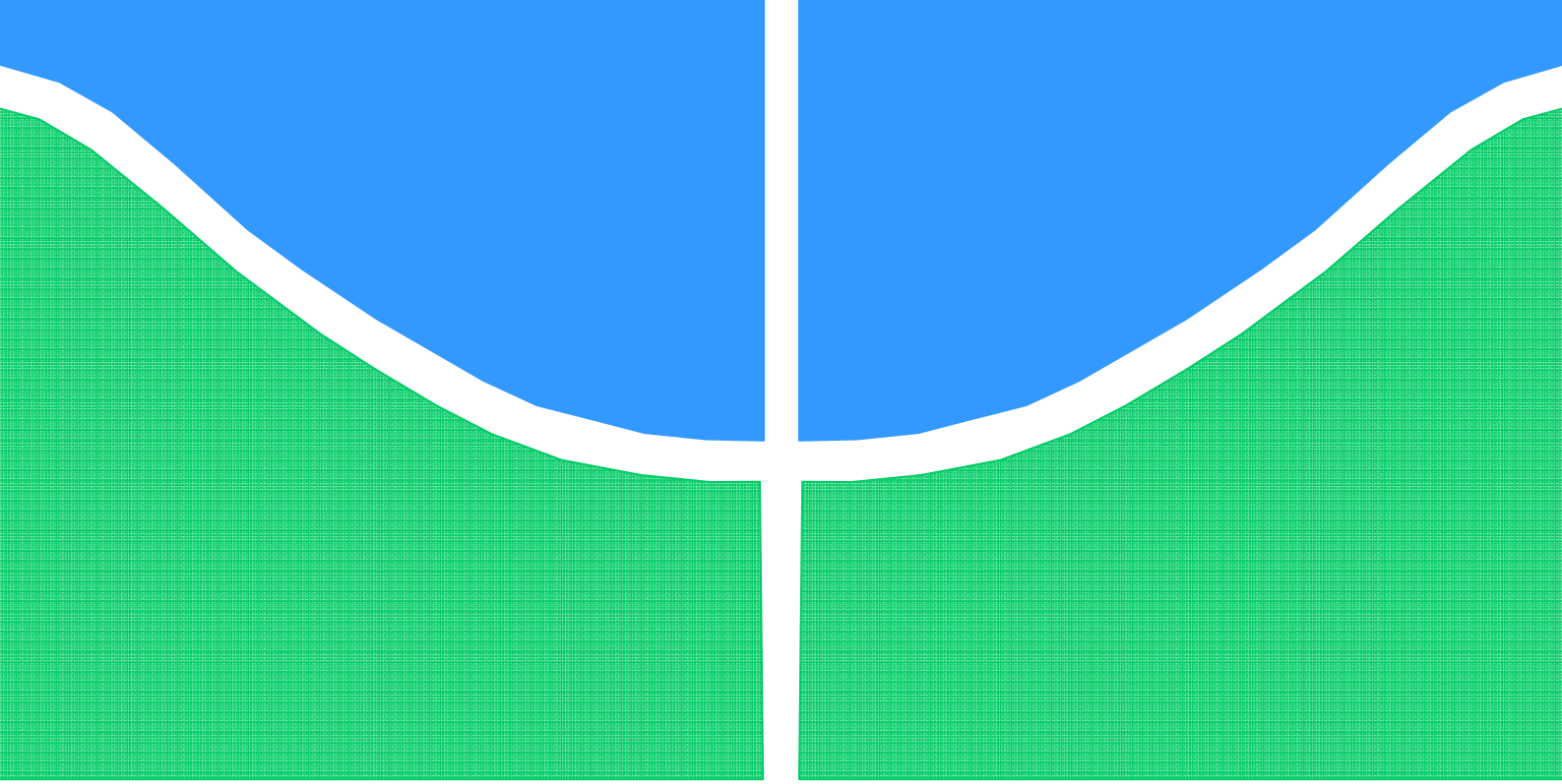


DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA

SDN - DMM: REDES DEFINIDAS POR SOFTWARE PARA GERENCIAMENTO DE MOBILIDADE DISTRIBUÍDO EM REDES IP MÓVEIS HETEROGÊNEAS

Rodrygo Torres Córdova

Orientador: Paulo Roberto de Lira Gondim

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SDN - DMM: REDES DEFINIDAS POR SOFTWARE PARA GERENCIAMENTO DE MOBILIDADE DISTRIBUÍDO EM REDES IP MÓVEIS HETEROGÊNEAS

RODRYGO TORRES CÓRDOVA

ORIENTADOR: PAULO ROBERTO DE LIRA GONDIM

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA
PUBLICAÇÃO: 652/2016
BRASÍLIA/DF: DEZEMBRO - 2016

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

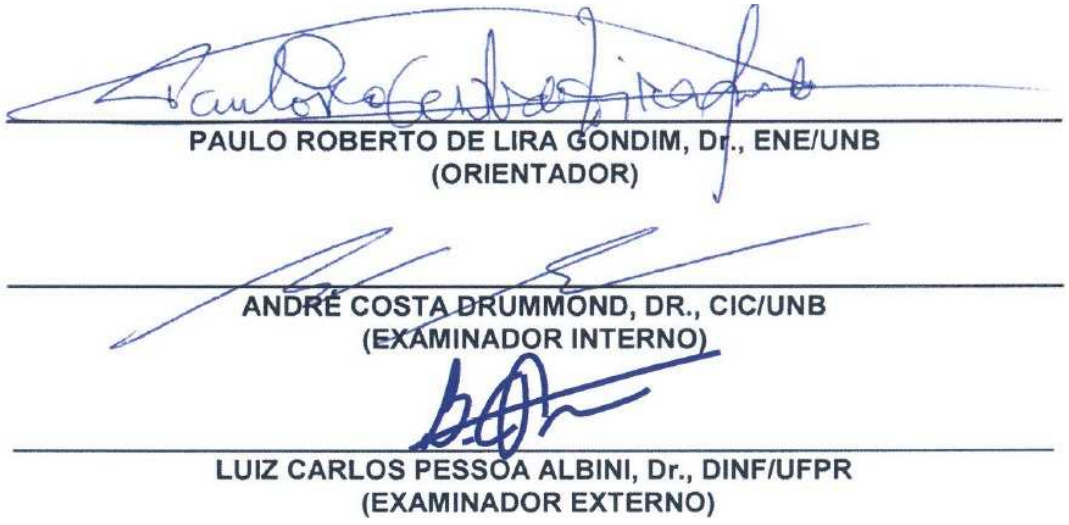
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SDN - DMM: REDES DEFINIDAS POR SOFTWARE PARA GERENCIAMENTO DE MOBILIDADE DISTRIBUÍDO EM REDES IP MÓVEIS HETEROGÊNEAS

RODRYGO TORRES CÓRDOVA

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE.

APROVADA POR:



Brasília, 20 de Dezembro de 2016.

FICHA CATALOGRÁFICA

CORDOVA, RODRYGO TORRES

SDN - DMM: Redes Definidas por Software para Gerenciamento de Mobilidade Distribuído em Redes IP Móveis Heterogêneas

[Distrito Federal] 2016.

x, 167p., 210 x 297 mm (ENC/FT/UnB, Mestre, Engenharia Elétrica, 2016).

Dissertação de Mestrado – Universidade de Brasília. Faculdade de Tecnologia.

Departamento de Engenharia Elétrica

1. 2.

3. 4.

I. II.

REFERÊNCIA BIBLIOGRÁFICA

CORDOVA, R. T. (2016). SDN - DMM: Redes Definidas por Software para Gerenciamento de Mobilidade Distribuído em Redes IP Móveis Heterogêneas. Dissertação de Mestrado em Engenharia Elétrica, Publicação 652/2016, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 167p.

CESSÃO DE DIREITOS

AUTOR: Rodrygo Torres Córdova.

TÍTULO: SDN - DMM: Redes Definidas por Software para Gerenciamento de Mobilidade Distribuído em Redes IP Móveis Heterogêneas.

GRAU: Mestre ANO: 2016

_____

Rodrygo Torres Córdova
Brasília – DF – Brasil.

Dedicatória

Este trabalho é primeiramente o fruto da dedicação e do amor dado por meus pais a mim, sem os quais nada seria possível. Dedico a eles, Rosineide e Córdova, este trabalho. A minha irmã, conselheira e amiga, Nathalya, por me inspirar a ser uma pessoa cada vez melhor. E aos meus amigos que sempre acreditaram e me incentivaram durante a minha jornada.

Rodrygo Torres Córdova

Agradecimentos

Agradeço a todos os meus amigos, familiares e professores que desde sempre contribuíram para o meu crescimento, preenchendo este caminho com boas recordações. Agradeço em especial ao meu orientador, Paulo Gondim, pelo empenho, responsabilidade, confiança e principalmente pelo apoio, sem os quais não seria possível chegar ao final deste processo de amadurecimento acadêmico, profissional e pessoal, tornando-se um grande amigo. Agradeço ao professor Jaime Lloret pelas sugestões e referências fornecidas. Agradeço a Deus por tornar tudo possível, sempre iluminando o meu caminho e as pessoas que passam por ele.

Rodrygo Torres Córdova

RESUMO

SDN - DMM: REDES DEFINIDAS POR SOFTWARE PARA GERENCIAMENTO DE MOBILIDADE DISTRIBUÍDO EM REDES IP MÓVEIS HETEROGÊNEAS

O gerenciamento de mobilidade aplicado na arquitetura tradicional da Internet tornou-se um grande desafio devido ao crescimento exponencial do tráfego móvel e do número de dispositivos que podem se conectar à rede. Soluções baseadas em DMM – *Distributed Mobility Management* tem sido consideradas nos últimos anos, permitindo lidar com os desafios apresentados nas abordagens de gerenciamento centralizada, como a escalabilidade e o desempenho.

Por outro lado, uma forte tendência em termos de evolução das redes de comunicação, especialmente da Internet, envolve as SDN- *Software-Defined Networks*, que se caracterizam pelo desacoplamento entre o plano de controle e o plano de dados da infraestrutura de rede de comunicação, sendo a função de controle da rede realizada em um ponto central responsável pela inteligência da rede (o controlador SDN) atuando somente no plano de controle, enquanto os demais ativos de rede ficam responsáveis apenas pelo encaminhamento de pacotes no plano de dados, inserindo assim flexibilidade, otimização e programabilidade da infraestrutura de rede.

Propõe-se aqui uma abordagem baseada em redes definidas por *software*, intitulada SDN-DMM, em que se considera uma arquitetura de rede que permite não somente obter os benefícios intrínsecos ao SDN, mas também lidar de forma simplificada e eficiente com o gerenciamento de mobilidade de modo distribuído em redes de acesso heterogêneas, garantindo a continuidade de sessão IP durante o deslocamento do usuário entre as redes. Nessa proposta, o conceito de *intents* é utilizado para que o plano de dados da rede seja automaticamente ajustado de acordo com a mobilidade executada pelo usuário para manter ativa a comunicação, permitindo que esta seja executada de forma otimizada.

A proposta SDN-DMM é comparada, com base em modelagem analítica, com duas propostas recentemente publicadas (a primeira, parcialmente distribuída e a segunda, completamente distribuída), considerando custos de sinalização, de entrega de pacote, de latência de *handover* e de roteamento em um cenário de integração de redes heterogêneas. Verifica-se que a proposta SDN-DMM, parcialmente distribuída, mostra-se vantajosa em relação às outras duas nos aspectos de entrega de pacote, latência de *handover* e roteamento, e desvantajosa sob o aspecto de sinalização quando o usuário executa um grande número de *handovers* em um curto período de tempo. A proposta SDN-DMM é então viabilizada em implementação em um cenário real de experimentação.

Conclui-se então que o gerenciamento de mobilidade distribuído pode se beneficiar do paradigma das redes definidas por *software*, devido à flexibilidade inserida e à programabilidade, que tornam o gerenciamento de mobilidade mais um dos serviços que podem ser prestados pela infraestrutura de forma otimizada através das aplicações executadas no controlador SDN, capaz de realizar os ajustes necessários dos fluxos IP bidirecionais no plano de dados de modo a suportar a mobilidade do usuário.

Palavras chave: Gerenciamento de mobilidade distribuído; DMM; Gerenciamento de mobilidade IP; Redes IP móveis; Redes Heterogêneas; Redes definidas por *software*; SDN; *OpenFlow*.

ABSTRACT

SDN - DMM: SOFTWARE-DEFINED NETWORKING FOR DISTRIBUTED MOBILITY MANAGEMENT ON HETEROGENEOUS MOBILE IP NETWORKS

Mobility management applied on Internet traditional architecture has become a major challenge due to the exponential growth in the mobile traffic and in the number of devices which could connect to the network. Solutions based on DMM - Distributed Mobility Management have been considered in the past few years, allowing dealing to challenges presented on centralized management approaches such as scalability and performance.

On the other hand, a strong tendency in terms of evolution of communication networks, especially the Internet, involves the Software-Defined Networking (SDN), characterized by the decoupling between control plane and data plane of the Communication Networks' infrastructure, where the network control function is performed in a central point (SDN Controller) acting only at control plane, while the other networks assets are responsible only for the routing packets in the data plane, thus inserting flexibility, optimization and programmability into the network infrastructure.

This work proposes an approach based on Software-Defined Networking, entitled SDN-DMM, in which it is considered a network architecture that allows not only to obtain the intrinsic benefits of SDN but also to deal in a simplified and efficient way with distributed mobility management in heterogeneous access networks, guaranteeing mainly the continuity of IP Session during the user's displacement between networks. The concept of intents is used so that the network data plane is automatically adjusted according to the mobility performed by the user to keep the communication active, allowing it to be optimally executed.

SDN-DMM proposal is compared with partially and fully distributed proposals, considering signaling, packet delivery, handover latency and routing costs in an integrated heterogeneous network scenario. Partially distributed SDN-DMM proposal is considered advantageous in relation to the other two in aspects such as packet delivery, handover latency and routing, and it is disadvantageous in the aspect of signaling when the user executes a large number of handovers in a short period of time. The SDN-DMM proposal is then implemented in a real experimentation scenario, using equipament of the telecommunications market.

This work concludes that distributed mobility management can benefit itself from the Software-Defined Networking paradigm due to the introduced flexibility and to the programmability in a optimized way throught the applications running on the SDN Controller, able to accomplish the necessary adjustments of bidirectional IP flows in the Data Plane in order to support the user mobility.

Keywords: Distributed Mobility Management; DMM; IP Mobility Management; Mobile IP networks; Heterogeneous Networks; Software-Defined Networking; SDN; OpenFlow.

SUMÁRIO

1 INTRODUÇÃO 1
1.1 ASPECTOS GERAIS E MOTIVAÇÃO 1
1.2 OBJETIVOS 2
1.2.1 OBJETIVO GERAL 2
1.2.2 OBJETIVOS ESPECÍFICOS 2
1.3 METODOLOGIA. 3
1.4 JUSTIFICATIVA 3
1.5 CONTRIBUIÇÕES (RESULTADOS OBTIDOS) 4
1.6 ORGANIZAÇÃO DO TRABALHO 5

2 REVISÃO BIBLIOGRÁFICA 6
2.1 GERENCIAMENTO DE MOBILIDADE 6
2.1.1 GERENCIAMENTO DE MOBILIDADE: CENTRALIZADO E DISTRIBUÍDO 6
2.1.2 GERENCIAMENTO DE MOBILIDADE: BASEADO NO MÓVEL E NA REDE 7
2.1.3 GERENCIAMENTO DE MOBILIDADE: OUTRAS PROPOSTAS DE CATEGORIZAÇÃO 8
2.1.4 GERENCIAMENTO DE MOBILIDADE: SINGLE-HOMED E MULTI-HOMED 8
2.2 REDES HETEROGÊNEAS 9
2.2.1 REDES MÓVEIS 3GPP: ARQUITETURA SAE E LTE (4G) 10
2.2.2 REDES MÓVEIS IEEE: WLAN 13
2.3 REDES DEFINIDAS POR SOFTWARE (SDN) E OPENFLOW 18
2.3.1 PARADIGMA SDN 19
2.3.2 PROTOCOLO OPENFLOW 22
2.4 CONSIDERAÇÕES FINAIS 24

3 TRABALHOS RELACIONADOS 25
3.1 MOBILE IPV6 (MIPV6) 25
3.2 PROXY MOBILE IPV6 (PMIPV6) 28
3.3 DRAFT IETF - JAEHWOON 29
3.4 DRAFT IETF - BERNARDOS 32
3.5 CONSIDERAÇÕES FINAIS 34

4 PROPOSTA SDN-DMM: ARQUITETURA BASEADA EM SDN PARA DMM 36
4.1 MOBILIDADE COM O USO DE INTENT EM REDES SDN 36
4.2 DESCRIÇÃO DO CENÁRIO E OPERAÇÃO SDN-DMM 39
4.3 CONSIDERAÇÕES FINAIS 45

5 AVALIAÇÃO DE DESEMPENHO DAS PROPOSTAS 46
5.1 CUSTO DE SINALIZAÇÃO 46
5.1.1 DRAFT IETF - JAEHWOON 47
5.1.2 DRAFT IETF - BERNARDOS 47
5.1.3 SDN-DMM 48
5.2 CUSTO DE ENTREGA DE PACOTE 49
5.2.1 DRAFT IETF - JAEHWOON 50
5.2.2 DRAFT IETF - BERNARDOS 50
5.2.3 SDN-DMM 51
5.3 CUSTO DE LATÊNCIA DE HANDOVER 51
5.3.1 DRAFT IETF - JAEHWOON 52
5.3.2 DRAFT IETF - BERNARDOS 53
5.3.3 SDN-DMM 54
5.4 CUSTO DE ROTEAMENTO 54

5.4.1 DRAFT IETF – JAEHWOON E BERNARDOS 57
5.4.2 SDN-DMM 61
5.5 RESUMO COMPARATIVO DAS CARACTERÍSTICAS 61
5.6 RESULTADOS OBTIDOS 62
5.6.1 SINALIZAÇÃO 62
5.6.2 ENTREGA DE PACOTE 65
5.6.3 LATÊNCIA DE HANDOVER 67
5.6.4 ROTEAMENTO 70
5.7 CONSIDERAÇÕES FINAIS 73

6 IMPLEMENTAÇÃO SDN-DMM 74
6.1 DESCRIÇÃO DO CENÁRIO 74
6.2 ARQUITETURA E IMPLEMENTAÇÃO 77
6.2.1 IMPLEMENTAÇÃO DO AMBIENTE DE REDE 79
6.2.2 IMPLEMENTAÇÃO DO CONTROLADOR SDN 79
6.2.3 IMPLEMENTAÇÃO DO PROTOCOLO OPENFLOW 82
6.2.4 IMPLEMENTAÇÃO DA MOBILIDADE SDN-DMM POR INTENT 83
6.3 TESTES E RESULTADOS 84
6.3.1 CENÁRIO ROTEAMENTO TRADICIONAL SEM HANDOVER 84
6.3.2 CENÁRIO SDN-DMM SEM HANDOVER 87
6.3.3 CENÁRIO SDN-DMM COM HANDOVER POR CHAVEAMENTO MANUAL 89
6.3.4 CENÁRIO SDN-DMM COM HANDOVER WIRELESS AUTOMATIZADO 93
6.3.5 ANÁLISE DOS RESULTADOS 103
6.4 CONSIDERAÇÕES FINAIS 104

7 CONCLUSÕES E TRABALHOS FUTUROS 106
7.1 CONCLUSÕES 106
7.2 TRABALHOS EM ANDAMENTO E FUTUROS 107

REFERÊNCIAS BIBLIOGRÁFICAS 109

APÊNDICE A – CONFIGURAÇÕES DA IMPLEMENTAÇÃO 115
A.1 AMBIENTE DE REDE 115
A.2 CONTROLADOR SDN 116
A.3 PROTOCOLO OPENFLOW 120
A.4 INTENT 129

APÊNDICE B – ARTIGO PUBLICADO E APRESENTADO 137
B.1 XXXIV SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADRES E SISTEMAS DISTRIBUÍDOS (SBRC2016) 137

APÊNDICE C – ARTIGO SUBMETIDO 142
C.1 TRANSACTIONS ON EMERGING TELECOMMUNICATIONS TECHNOLOGIES (2016) 142

APÊNDICE D – ARTIGOS ELABORADOS 152
D.1 PACKET DELIVERY AND ROUTING OPTIMIZATION FOR DMM WITH SDN-DMM 152
D.2 AVALIAÇÃO EXPERIMENTAL DE PROPOSTA BASEADA EM SDN PARA GERENCIAMENTO DE MOBILIDADE DISTRIBUÍDO 160

LISTA DE TABELAS

Capítulo 5

Tabela 5.1 - Comparação de aspectos referentes a DMM..... 62

Capítulo 6

Tabela 6.1 - Especificação técnica dos equipamentos utilizados na experimentação..... 74
Tabela 6.2 - Descrição dos softwares utilizados na experimentação. 76
Tabela 6.3 - Plano de Endereçamento IP. 78
Tabela 6.4 - Resultados para o cenário de roteamento tradicional..... 87
Tabela 6.5 - Resultados para o cenário SDN-DMM. 89
Tabela 6.6 - Resultados para o cenário SDN-DMM com handover manual por chaveamento. 92
Tabela 6.7 - Resultados para o cenário SDN-DMM com handover automático – MN1 conectado à rede A. 96
Tabela 6.8 - Resultados para o cenário SDN-DMM com handover automático – MN1 conectado à rede B. 99
Tabela 6.9 - Resultados para o cenário SDN-DMM com handover automático por wireless 103
Tabela 6.10 - Comparação das métricas entre Roteamento Tradicional e SDN-DMM..... 103
Tabela 6.11 - Métricas SDN-DMM com handover manual e automático. 104

LISTA DE FIGURAS

Capítulo 2

Figura 2.1 - Evolução 2G e 3G (modificada de [44]). 10
Figura 2.2 - Arquitetura plana SAE (modificado de [31]). 11
Figura 2.3 - Domínios da Arquitetura SAE [31]. 12
Figura 2.4 - Camadas do RM-OSI especificadas pela família IEEE 802.11 (modificado de [48]). 14
Figura 2.5 - Arquiteturas WLAN [50]. 17
Figura 2.6 - Arquitetura SDN (baseado em [36]). 20
Figura 2.7 - Arquitetura e características OpenFlow [51]. 22
Figura 2.8 - Campos de identificação para Fluxos do OpenFlow [36]. 23

Capítulo 3

Figura 3.1 - Operação MIPv6 (modificado [33]). 26
Figura 3.2 - Operação PMIPv6 [16]. 28
Figura 3.3 - Sinalização e encaminhamento de tráfego entre os MAGs [35]. 31
Figura 3.4 - Draft-bernardos-dmm-pmip [16]. 33
Figura 3.5 - Troca de mensagens de controle draft-Bernardos [16]. 34

Capítulo 4

Figura 4.1 - Relação entre a camada northbound, camada southbound e as informações de topologia. 38
Figura 4.2 - Arquitetura SDN-DMM. 40
Figura 4.3 - Cenário de conexão para as situações s1, s2 e s3.. 42
Figura 4.4 - Handover e estabelecimento do fluxo IP bidirecional em s2. 44
Figura 4.5 - Handover e estabelecimento do fluxo IP bidirecional em s3. 44

Capítulo 5

Figura 5.1 - Topologia para custo de roteamento. 56
Figura 5.2 - Custos de roteamento para m = 2 e n = 1,2 e 3. 58
Figura 5.3 - Custo de Sinalização: SDN-DMM vs Draft-Jaehwoon. 63
Figura 5.4 - Custo de Sinalização: SDN-DMM vs Draft-Bernardos. 65
Figura 5.5 - Custo de entrega de pacote: SDN-DMM vs draft-Jaehwoon. 66
Figura 5.6 - Custo de entrega de pacotes: SDN-DMM vs draft-Bernardos. 66

Figura 5.7 - Custo de latência de handover: SDN-DMM vs draft-Jaehwoon. 68
Figura 5.8 - Custo de latência de handover: SDN-DMM vs draft-Bernardos..... 70
Figura 5.9 - Custo de roteamento: SDN-DMM vs draft-Jaehwoon/Bernardos..... 71
Figura 5.10 - Custo de roteamento: SDN-DMM vs drafts Jaehwoon/Bernardos (2 ≤m≤1000). 72

Capítulo 6

Figura 6.1- Topologia de implementação da proposta SDN-DMM..... 78
Figura 6.2 - Relação de softwares, sistemas operacionais e hardware para o Controlador SDN. 79
Figura 6.3 - Diferença entre o modelo de máquina virtual e container..... 80
Figura 6.4 - Topologia para acesso remoto ao container ONOS-Controller..... 81
Figura 6.5 - Teste de vazão e perda de pacotes UDP com Iperf3 para o roteamento tradicional. 85
Figura 6.6 - Teste de vazão máxima TCP com Iperf3 para o roteamento tradicional. 86
Figura 6.7 - Perda de pacotes e latência RTT com ICMP para o roteamento tradicional..... 86
Figura 6.8 - Teste de vazão e perda de pacotes UDP com Iperf3 para SDN-DMM 87
Figura 6.9 - Teste de vazão máxima TCP com Iperf3 para SDN-DMM.. 88
Figura 6.10 - Perda de pacotes e latência RTT com ICMP para SDN-DMM..... 88
Figura 6.11- Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com handover manual por chaveamento..... 90
Figura 6.12 - Teste de vazão máxima TCP com Iperf3 para SDN-DMM com handover manual por chaveamento. 90
Figura 6.13- Perda de pacotes e latência RTT com ICMP para SDN-DMM com handover manual por chaveamento..... 91
Figura 6.14 - Latência de handover TCP para SDN-DMM com handover manual por chaveamento..... 92
Figura 6.15 - Latência de handover ICMP para SDN-DMM com handover manual por chaveamento..... 92
Figura 6.16- Conexão do host MN1 ao access-point AP-A na rede A. 94
Figura 6.17 - Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com MN1 conectado à rede A 94
Figura 6.18- Teste de vazão máxima TCP com Iperf3 para SDN-DMM com MN1 na rede A. 95

Figura 6.19 - Perda de pacotes e latência RTT com ICMP para SDN-DMM com MN1 na rede A 95
Figura 6.20 - Conexão do host MN1 ao access-point AP-B na rede B. 96
Figura 6.21 - Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com MN1 conectado à rede B. 97
Figura 6.22 - Teste de vazão máxima TCP com Iperf3 para SDN-DMM com MN1 conectado à rede B. 98
Figura 6.23- Perda de pacotes e latência RTT com ICMP para SDN-DMM com MN1 conectado à rede B.. 98
Figura 6.24 - Movimentação do host MN1 entre as redes A e B durante o handover 99
Figura 6.25- Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com handover automático por wireless. 100
Figura 6.26 - Teste de vazão máxima TCP com Iperf3 para SDN-DMM com handover automático por wireless. 101
Figura 6.27- Perda de pacotes e latência RTT com ICMP para SDN-DMM com handover automático por wireless. 101
Figura 6.28 - Latência de handover TCP para SDN-DMM com handover automático por wireless. 102
Figura 6.29 - Latência de handover ICMP para SDN-DMM com handover automático por wireless. 102

Apêndice A

Figura A.2.1 - Instalação e execução do container ONOS-Controller. 117
Figura A.2.2 - Conexão lógica entre o host C1 e o container ONOS-Controller. 118
Figura A.2.3 - Acesso remoto SSH ao ONOS-Controller. 119
Figura A.2.4 - Acesso remoto WEB ao ONOS-Controller. 120
Figura A.3.1 - Verificação do suporte ao OpenFlow e informações gerais antes da implementação. 121
Figura A.3.2 - Protocolo OpenFlow ativado no switch OFS1. 122
Figura A.3.3 - Fluxos IP e tabelas no switch OFS1. 123
Figura A.3.4 - Tentativa de estabelecimento de sessão TCP pelo switch OFS1 com o controlador C1 para a troca de mensagens OpenFlow. 124
Figura A.3.5 - Estabelecimento da sessão TCP para troca de mensagens OpenFlow. 125
Figura A.3.6 - Troca de mensagens OpenFlow entre o switch OFS1 e o controlador C1. 125

Figura A.3.7 - Troca de mensagens OpenFlow OFPMP_DESC entre o switch OFS1 e o controlador C1. 126
Figura A.3.8 - Reconhecimento do switch OF1 pela interface WEB do ONOS. 126
Figura A.3.9 - Ativação do OpenFlow e reconhecimento do switch pela CLI do ONOS-Controller. 127
Figura A.3.10 - Tabela de Fluxos do switch OFS1 com as regras definidas pelo controlador instaladas. 128
Figura A.3.11 - Solicitação OpenFlow do controlador C1 para adição de um fluxo no switch OFS1..... 129
Figura A.4.1 - Reconhecimento dos hosts conectados a infraestrutura pela interface WEB..129
Figura A.4.2 - Informações de topologia e identificação dos hosts pela CLI do ONOS-Controller. 130
Figura A.4.3 - Comunicação por intent e os fluxos OpenFlow associados no ONOS-Controller. 131
Figura A.4.4 - Pacotes OpenFlow para adição do fluxo IP gerados pela Intent. 131
Figura A.4.5 - Tabela de fluxos do switch OFS1 com os fluxos da comunicação indicada instalados. 132
Figura A.4.6 - Criação da comunicação entre os hosts e caminho da topologia utilizado..... 132
Figura A.4.7 - Mensagem OpenFlow informando ao controlador sobre a mudança topológica. 133
Figura A.4.8 - Solicitação de remoção dos fluxos pelo controlador ONOS-Controller devido à mudança topológica de rede. 134
Figura A.4.9 - Novos fluxos OpenFlow para estabelecer a comunicação indicada na intent 134
Figura A.4.10 - Novas mensagens OpenFlow para instalação do fluxo IP depois da mudança. 135
Figura A.4.11 - Tabela de encaminhamento do switch OF1 com os novos fluxos..... 135
Figura A.4.12 - Nova localização do host MN1 e novo caminho utilizado para comunicação. 136

LISTA DE SÍMBOLOS, NOMECLATURAS E ABREVIAÇÕES

ABNT	Associação Brasileira de Normas Técnicas
ACK	*ACKnowledgement*
AES	*Advanced Encryption Standard*
AID	*Association ID*
ARP	*Address Resolution Protocol*
AS	*Autonomous System*
BSS	*Basic Service Set*
CDMA	*Code Division Multiple Access*
CN	*Correspondent Node*
CoA	*Care-of Address*
CP	*Control Plane*
CPU	*Central Processing Unit*
CSMA	*Carrier Sense Multiple Access*
CSMA/CA	*Carrier Sense Multiple Access/Colission Avoidance*
DS	*Distribution Systems*
DSSS	*Direct Sequence Spread Spectrum*
ENB	*Evolved Node B*
EGP	*Exterior gateway Protocol*
EPC	*Evolved Packet Core*
ESS	*Extended Service Set*
E-UTRAN	*Evolved Universal Terrestrial Radio Access Network*
GPRS	*General Packet Radio Services*
GSM	*Global System for Mobile Communications*
GTP	GPRS *Tunneling Protocol*
GW	*Gateway*
HÁ	*Home Agent*
HN	*Home Network*
HoA	*Home Address*
IBSS	*Independent Basic Service Set*
ICMP	*Internet Control Message Protocol*
IEEE	*Institute of Electrical and Electronics Engineers*
IETF	*Internet Engineering Task Force*
IGP	*Interior Gateway Protocol*

IP	*Internet Protocol*
ISDN	*Integrated Service Digital Network*
ISO	*International Organization for Standardization*
ISP	*Internet service provider*
LAN	*Local Area Network*
LCC	*Logical Link Control*
LTE	*Long Term Evolution*
MAC	*Media Access Control*
MD5	*Message-Digest algorithm 5*
MIMO	*Multiple Input Multiple Output*
MIPv6	*Mobile IPv6*
MN	*Mobile Node*
NB	*Node B*
NBR	Norma Brasileira
OFDM	*Orthogonal Frequency Division Multiplexing*
OOB	*Out-of-Band*
OSI	*Open Systems Interconnection*
PMIPv6	*Proxy Mobile IPv6*
PSTN	*Public Switched Telephone Network*
P-GW	*Packet Data Network Gateway*
QOE	*Quality of Experience*
QOS	*Quality of Service*
RAN	*Radio Access Network*
RFC	*Request For Comments*
RTT	*Round Trip Time*
SAE	*System Architecture Evolution*
SDN	*Software-Defined Networking*
S-GW	*Serving Gateway*
SSID	*Service Set Identifier*
TCP	*Transport Control Protocol*
TKIP	*Temporal Key Integrity Protocol*
UDP	*User Datagram Protocol*
UMTS	*Universal Mobile Telecommunication System*
UP	*User Plane*
UTRAN	*Universal Terrestrial Radio Access Network*
VLAN	*Virtual Local Area Network*

WAN	*Wide Area Networks*
WEP	*Wired Equivalent Privacy*
Wi-FI	*Wireless-Fidelity*
WLAN	*Wireless Local Area Network*

1 – INTRODUÇÃO

Este capítulo apresenta os principais aspectos em relação à motivação e justificativas que levaram à elaboração deste trabalho de dissertação, assim como os objetivos que o nortearam, a metodologia empregada para alcançar tais objetivos, as contribuições obtidas e a estruturação do mesmo.

1.1 – ASPECTOS GERAIS E MOTIVAÇÃO

As redes de comunicação móveis tornaram-se o método de acesso principal dos usuários à Internet, resultando em um aumento significativo do número de dispositivos móveis conectados à rede global [13].

Como os serviços ofertados pelas operadoras de redes de comunicações móveis tendem a envolver soluções completamente baseadas no protocolo IP, seja para o serviço de voz, seja para o serviço de dados [1], e as sessões de comunicação precisam ter continuidade nessas redes, o gerenciamento de mobilidade IP torna-se um fator de enorme importância para as redes de comunicação a fim de suportar tal crescimento [27].

Tem-se então um cenário de crescimento exponencial do número de dispositivos móveis conectados à rede, favorecido pelo aumento da disponibilidade e do uso de aplicações que necessitam de mobilidade, especialmente ferramentas colaborativas em tempo real. Para esse cenário, as soluções atuais de gerenciamento de mobilidade não são adequadas para atender satisfatoriamente aos requisitos impostos às infraestruturas das redes de comunicação.

Os padrões de gerenciamento de mobilidade IP observados no IETF, tais como MIPv6 [8] e PMIPv6 [9], são dependentes de unidades centrais, que gerenciam os tráfegos de controle e de dados; elaborados de acordo com o roteamento tradicional de pacotes IP, apresentando problemas como roteamento sub-otimizado, baixa escalabilidade, sobrecarga de processamento dos ativos que realizam o transporte dos dados e pouca granularidade no serviço de gerenciamento de mobilidade.

Adicionalmente, a utilização de redes de acesso heterogêneas (HetNets), que pode ser considerada basicamente como a utilização de várias tecnologias, arquiteturas, tipos de transmissão e estações base de acesso a radio para aumentar a capacidade da rede móvel, impõe outras dificuldades a esse gerenciamento, como o gerenciamento integrado de recursos e as transições suaves entre as redes.

Como alternativa para lidar com os problemas intrínsecos da centralização no gerenciamento de mobilidade, um novo conceito chamado de *distributed mobility management* (DMM) tem sido recentemente utilizado em várias abordagens [10].

Neste contexto, o gerenciamento de mobilidade enfrenta um constante desafio relativo à eficiência da rede de comunicação, sem aumentar a sua complexidade. A adição e a utilização de novos protocolos, novas mensagens de sinalização ou novos processos que geram *overhead* devido ao encapsulamento e ao tráfego de controle, são exemplos de como o gerenciamento de mobilidade impacta diretamente o OPEX (*Operational Expenditure*) e o CAPEX (*Capital Expenditure*) de uma rede de comunicação.

Este trabalho propõe uma arquitetura utilizando o paradigma SDN com o protocolo *OpenFlow* para implementar uma solução DMM *network-based* em um ambiente de redes heterogêneas para lidar com os desafios de gerenciamento de mobilidade IP citados. Assim como a análise do seu desempenho em comparação com outros trabalhos relacionados e por fim a sua implementação em um ambiente real de experimentação.

1.2 – OBJETIVOS

1.2.1 - Objetivo Geral

Avaliar os principais aspectos e trabalhos relacionados com o gerenciamento de mobilidade distribuído em um ambiente de crescimento exponencial do número de dispositivos móveis conectados às redes heterogêneas, bem como propor, avaliar e implementar uma abordagem de DMM utilizando o paradigma de redes definidas por *software*.

1.2.2 - Objetivos Específicos

- Apresentar as principais características de arquiteturas e protocolos de gerenciamento de mobilidade distribuído;
- Apresentar uma proposta baseada em SDN para o gerenciamento de mobilidade distribuído;
- Avaliar o desempenho da proposta em comparação com trabalhos recentes, com base em custos de sinalização, entrega de pacote, latência de *handover* e roteamento;
- Implementar e avaliar a proposta em um ambiente real de experimentação, utilizando métricas de desempenho de redes.

1.3 – METODOLOGIA

Uma vez definido o tema e observado os problemas envolvidos no gerenciamento de mobilidade para o cenário atual das redes de comunicação, o trabalho foi elaborado em quatro partes.

Inicialmente foi realizado um estudo para o aprofundamento dos conceitos, protocolos e ambientes relacionados à mobilidade em redes baseadas em IP, tanto de forma específica quanto em redes móveis heterogêneas, principalmente as redes LTE e Wi-Fi, seguido de um levantamento e pesquisa bibliográfica, que permitiu identificar propostas recentes com diferentes abordagens, focadas principalmente no gerenciamento de mobilidade distribuído baseado na rede, considerando que tal forma de gerenciamento não impõem alterações nos terminais móveis, atendendo ao primeiro objetivo específico.

A partir do entendimento das abordagens utilizadas nos trabalhos relacionados levantados, e uma vez identificados os principais problemas existentes, buscou-se investigar como o paradigma de redes definidas por *software* poderia contribuir para solucionar problemas relativos ao gerenciamento de mobilidade, permitindo chegar à elaboração de uma proposta baseada em SDN para tratar do gerenciamento de mobilidade distribuído baseado na rede, de modo a lidar com os problemas de desempenho identificados. Buscou-se atender, assim, ao segundo objetivo específico.

Para alcançar o terceiro objetivo específico do trabalho, foi então realizada a avaliação de desempenho da proposta citada e de duas publicações recentes, com base em funções de custo relativas a sinalização, entrega de pacotes, latência de *handover* e roteamento da proposta elaborada com outros trabalhos relacionados utilizando modelagem analítica.

Para finalizar, foi realizada uma implementação real da proposta em um ambiente de experimentação, utilizando equipamentos e *software*s de mercado de telecomunicações, permitindo a análise do comportamento operacional e a obtenção de métricas de rede, atendendo assim ao último objetivo específico do trabalho.

1.4 – JUSTIFICATIVA

Com a diversificação cada vez maior das tecnologias de rede de acesso e o aumento exponencial do tráfego de dados gerados por dispositivos móveis, frente aos recursos finitos de banda e capacidade de processamento da infraestrutura de redes de comunicações, o

gerenciamento de mobilidade IP tornou-se um fator de grande importância tanto para a experiência final do usuário, quanto para a própria rede de comunicação.

De modo a comportar a demanda exponencial por conectividade móvel em um ambiente de redes de acesso heterogêneas, sem que ocorra o aumento da complexidade da infraestrutura para que se tenha um uso eficiente dos recursos disponível, tornando-a escalável, um estudo aprofundado das abordagens, protocolos e tecnologias envolvidos no gerenciamento de mobilidade se faz necessário.

A infraestrutura de rede de comunicação deve ser capaz de acompanhar o crescimento do número de dispositivos conectados com a máxima otimização dos recursos disponíveis de modo a suportar aplicações avançadas e o atendimento de novos cenários.

Para tal, é fundamental a avaliação das propostas de gerenciamento de mobilidade (o que pode se dar por meio de modelagens analíticas), assim como a implementação real permite obter resultados experimentais que favorecem a análise dos impactos causados no desempenho das redes de comunicações.

1.5 – CONTRIBUIÇÕES (RESULTADOS OBTIDOS)

As seguintes contribuições foram obtidas como consequência do desenvolvimento deste trabalho de dissertação:

1. Publicação e apresentação de 01 (um) artigo em conferência nacional;
2. Submissão de 01 (um) artigo para periódico internacional;
3. Elaboração de 02 (dois) artigos para submissão em periódicos;
4. Categorização dos trabalhos recentes para o gerenciamento de mobilidade distribuído;
5. Proposta de uma abordagem SDN para o gerenciamento de mobilidade distribuído de forma otimizada;
6. Proposta de métrica de roteamento para avaliação do desempenho das propostas;
7. Avaliação de desempenho da proposta, em comparação com os trabalhos recentes, com base em aspectos de sinalização, entrega de pacote, latência de *handover* e roteamento, por meio de modelagem analítica;
8. Implementação da proposta em cenário real, utilizando equipamentos e *software*s de mercado;
9. Avaliação de desempenho da proposta em um cenário real.

1.6 – ORGANIZAÇÃO DO TRABALHO

O presente documento está estruturado da seguinte maneira: no capítulo 02 são apresentados os principais conceitos, arquiteturas e operações dos temas utilizados como base para ambientação e contextualização das análises e proposta contidas nos capítulos subsequentes, sendo este capítulo composto pela revisão bibliográfica sobre gerenciamento de mobilidade centralizado e distribuído, redes heterogêneas, redes definidas por *software* e o protocolo *OpenFlow*.

Logo após a revisão bibliográfica, o capítulo 03 apresenta trabalhos relacionados, onde ocorre a descrição de diferentes abordagens utilizadas para prover um gerenciamento de mobilidade centralizado, distribuído, baseado na rede e baseado no móvel.

A proposta elaborada para DMM com base em uma arquitetura SDN, intitulada de SDN-DMM, é apresentada no capítulo 04, onde são descritos os principais aspectos de operação do plano de controle e do plano de dados.

A avaliação de desempenho da proposta SDN-DMM em comparação com trabalhos relacionados é realizada no capítulo 05, focando nos aspectos de sinalização e latência de *handover* do plano de controle e nos aspectos de entrega de pacote e roteamento do plano de dados, sendo apresentados os resultados obtidos por meio de uma modelagem analítica.

No capítulo 06 são apresentados os resultados obtidos e observações realizadas com a implementação da proposta SDN-DMM em um ambiente real de experimentação.

O capítulo 07 encerra o trabalho, apresentando as conclusões e sugestões para trabalhos futuros.

2 – REVISÃO BIBLIOGRÁFICA

Neste capítulo é realizada uma revisão bibliográfica dos principais conceitos, tecnologias e protocolos envolvidos no ambiente de gerenciamento de mobilidade distribuído, apresentando a definição e a categorização do mesmo, assim como os aspectos de redes heterogêneas e o novo paradigma de arquitetura de Redes Definidas por *Software*, com base no protocolo *OpenFlow*.

2.1 – GERENCIAMENTO DE MOBILIDADE

O gerenciamento de mobilidade é um termo amplo que pode estar relacionado a vários aspectos e ser aplicado em diferentes camadas da pilha de protocolos do modelo de referência de rede OSI. Alguns tipos de mobilidade mais comumente conhecidos são [41]:

- Mobilidade de Dispositivos: é a mobilidade que permite ao usuário utilizar o mesmo dispositivo para se mover entre redes heterogêneas mantendo o acesso aos mesmos conjuntos de serviços permitidos;
- Mobilidade Pessoal: é a mobilidade que permite ao usuário iniciar e receber sessões de modo global através de um identificador pessoal a partir de qualquer dispositivo;
- Mobilidade de Sessão: é a mobilidade que permite a continuidade das sessões em curso enquanto um usuário muda de rede ou dispositivo.

Para o contexto adotado neste trabalho, o gerenciamento de mobilidade está relacionado com a mobilidade executada na camada de rede do modelo de referencia OSI, com o intuito de manter a continuidade das sessões IP em andamento durante a movimentação do usuário entre diferentes redes de acesso.

2.1.1 – Gerenciamento de Mobilidade: Centralizado e Distribuído

A principal categorização do gerenciamento de mobilidade é em relação ao modo que o tráfego do nó móvel (do inglês, *Mobile Node – MN)* será tratado pelo plano de dados (do inglês, *Data Plane*) da rede de comunicação. Neste sentido, podemos separar as abordagens de gerenciamento de mobilidade em duas categorias tradicionais:

- Gerenciamento de mobilidade centralizado: nesta abordagem todo o tráfego originado e destinado aos *mobile nodes* dentro de um domínio de mobilidade é

tratado por uma entidade central na rede, não importando a localização do *mobile node* dentro do domínio de mobilidade;

- Gerenciamento de mobilidade distribuído (do inglês, *Distributed Mobility Management – DMM*): não existe uma entidade central responsável pelo tratamento de todo o tráfego dos *mobile nodes* dentro do domínio de mobilidade, o tratamento é realizado de forma distribuída entre as entidades da rede.

A principal característica do DMM é a separação clara entre o plano de dados e o plano de controle (do inglês, *Control Plane – CP*), com o plano de dados sendo distribuído ao longo dos equipamentos na borda da rede de acesso, com o objetivo de aproximar o agente de mobilidade (do inglês, *mobile agent*) do usuário final, implementando uma abordagem de mobilidade de forma mais plana na rede [11]. Onde o nível de distribuição do plano de controle categoriza as soluções DMM em dois tipos [4, 11, 12]:

- Parcialmente distribuído (do inglês, *Partially distributed*): o plano de dados é completamente distribuído entre os elementos da rede e o plano de controle é centralizado em pontos de controle na rede;

- Completamente distribuído (do inglês, *Fully distributed*): o plano de dados e o plano de controle são completamente distribuídos entre os elementos da rede, não existe uma entidade central de controle.

Deste modo, nas soluções DMM, o tráfego destinado ao *mobile node* não precisa mais atravessar um ponto central específico na rede, o âncora de mobilidade (do inglês, *mobility anchor*), como ocorre nas soluções centralizadas; no DMM, o tráfego é encaminhado pelos agentes de mobilidade que estão localizados mais próximos ao usuário em mobilidade.

2.1.2 – Gerenciamento de Mobilidade: Baseado no Móvel e na Rede

Outra categorização tradicional do gerenciamento de mobilidade é em relação aos envolvidos nos processos de sinalização do plano de controle da rede. Neste sentido, são apresentadas duas categorias:

- Gerenciamento de mobilidade baseado no móvel (do inglês, *Client-based*): o dispositivo final participa ativamente da troca de mensagens de sinalização no plano de controle da rede, como solicitações de registro, tomando conhecimento dos processos executados para prover a mobilidade;

- Gerenciamento de mobilidade baseado na rede (do inglês, *Network-based*): toda a troca de mensagens de sinalização realizada no plano de controle é feita de modo transparente para o dispositivo final, ele não participa ou toma conhecimento dos processos executados para prover a mobilidade.

2.1.3 – Gerenciamento de Mobilidade: outras propostas de categorização

Além das categorias tradicionais descritas nos itens anteriores, em [4] os autores propõem mais quatro categorizações para o gerenciamento de mobilidade, sendo elas:

- *clean-state*: são propostas de novas arquiteturas de rede para lidar com os principais problemas da mobilidade, não utilizam as abordagens tradicionais de forma evolutiva, que muitas vezes apenas permitem soluções de contorno;
- *architecture-dependent*: são as soluções de mobilidades dependentes de uma arquitetura específica de rede, como algumas abordagens e os esforços para realizar o escoamento de tráfego que podem ser utilizadas somente em redes 3GPP;
- *peer-to-peer* (P2P): onde as funções de gerenciamento de mobilidade são distribuídas em redes P2P;
- *based on or extending existing IETF protocols*: são as abordagens que utilizam como base os padrões de mobilidade definidos nas RFCs do IETF, propondo funcionalidades completamente novas ou estendendo as existentes nos protocolos.

2.1.3 – Gerenciamento de Mobilidade: Single-homed e Multi-homed

O desempenho de determinada abordagem de gerenciamento de mobilidade é influenciado pela topologia de rede na qual ela será utilizada. Em relação à mobilidade global, um dos principais aspectos relacionados ao desempenho de como o tráfego destinado para fora do domínio de mobilidade é encaminhado, como o tráfego de Internet, está relacionado se o domínio possui apenas uma ou mais saídas para o ambiente externo. Neste sentido, o cenário de conexão externa de um domínio de mobilidade pode ser categorizado como:

- *Single-homed*: o domínio de mobilidade possui apenas um ponto central para escoamento do tráfego destinado ao ambiente externo;
- *Multi-homed*: o domínio de mobilidade possui mais de um ponto para escoamento do tráfego destinado ao ambiente externo.

Soluções de gerenciamento de mobilidade centralizadas tendem a ter um melhor desempenho em cenários de conexão *single-homed*, quando as unidades centrais responsáveis pelo plano de dados são posicionadas próximas ou utilizadas como a saída para escoar o tráfego externo ao domínio de mobilidade. Isso ocorre devido ao tráfego relacionado aos usuários em mobilidade, que é direcionado para a unidade central que trata da mobilidade, seguir uma rota que seria utilizada de qualquer maneira para o escoamento do tráfego.

Em cenários de conexão *multi-homed*, as soluções de gerenciamento de mobilidade distribuídas tendem a ter um melhor desempenho, uma vez que as múltiplas saídas para escoar o tráfego externo podem ser utilizadas de forma mais otimizada devido à distribuição do plano de dados, aproximando então o encaminhamento do tráfego de determinado usuário em mobilidade com a saída mais próxima, sem necessitar passar por pontos centrais.

2.2 – REDES HETEROGÊNEAS

O conceito de redes heterogêneas surgiu juntamente com o avanço tecnológico que propiciou o aparecimento e implantação de diversos padrões e infraestruturas utilizadas nas redes de acesso. A ideia de convergência dessas diferentes abordagens de acesso tem como objetivo permitir a unificação e uso de forma mais eficiente da infraestrutura disponível, fornecendo vantagens como [42]:

- Aumento da área de cobertura ofertada para os usuários;
- Aumento da flexibilidade e interoperabilidade no acesso do usuário;
- Possível redução do custo dos serviços para o usuário;
- Implementação de novos serviços;
- Possibilidade de melhorar a experiência do usuário através de técnicas de QoS e *data offloading*.

A arquitetura para prover a integração entre as redes heterogêneas pode ser realizada basicamente de dois modos [43]:

- Fracamente acoplada: neste modo o processo de sinalização ocorre somente no núcleo de uma das redes, com o tráfego sendo encaminhado para a outra rede. Existe uma independência entre as redes principalmente em relação ao plano de dados;

- Fortemente acoplada: neste modo tanto o processo de sinalização quanto o encaminhamento de tráfego é direcionado para uma das redes, devido a ligação entre os núcleos das redes.

2.2.1 – Redes móveis 3GPP: Arquitetura SAE e LTE (4G)

A evolução das interfaces de radio utilizadas desde as redes de comunicação da segunda geração baseadas em GSM (*Global System for Mobile Communications*), onde houve a introdução da comutação por pacotes com a rede GPRS (*General Packet Radio Services*), até as redes de terceira geração baseadas nestas, que utilizam o sistema UMTS (*Universal Mobile Telecommunication System*) com a implantação de uma nova estrutura RAN (*Radio Access Network*), chamada de UTRAN (*Universal Terrestrial Radio Access Network*), como pode ser observado na figura 2.1, começou a deixar claro que não somente era necessário que as interfaces de radio evoluíssem, mas que toda a arquitetura do sistema também precisava evoluir [31].

A tendência geral de se otimizar o sistema para fornecer serviços baseados na comutação de pacotes, ou seja, no mundo IP, já era um grande indício da necessidade desta evolução. O que era reforçado na medida em que a evolução da RAN permitia que esta absorvesse de forma mais eficiente funções que antes estavam concentradas no núcleo da rede, como o processo de *handover* entre as estações radio base.

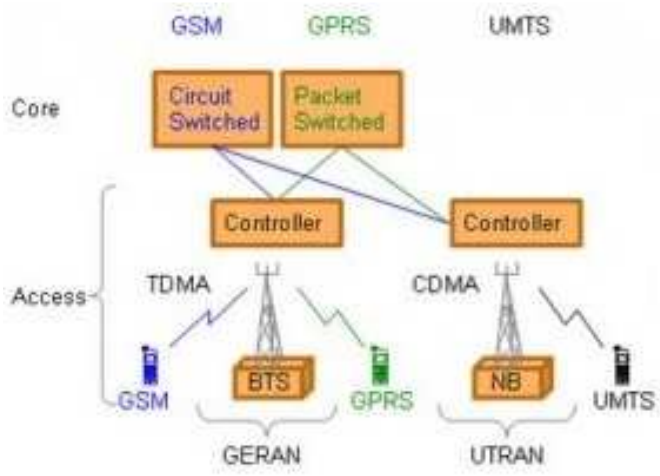


Figura 2.1 – *Evolução 2G e 3G (modificada de [44]).*

Estes pontos facilitaram para que fosse iniciado um debate sobre uma nova arquitetura, mais simples, mas que deveria atender aos seguintes objetivos:

- Simplificação geral do sistema;
- Otimização para comutação de pacotes em geral;
- Otimização para altas taxas de *throughput*, uma vez que a RAN estava proporcionando taxas de transmissão cada vez mais elevadas;

- Redução do tempo para ativar e estabelecer a comunicação no sistema;
- Redução da latência na entrega de pacotes;
- Otimização para interoperabilidade com outras redes de acesso 3GPP;
- Otimização para interoperabilidade com outras redes *wireless*.

Muitos desses objetivos teoricamente resultam em arquiteturas contraditórias, por exemplo, no caso de se buscar um aumento das taxas de transmissão e uma redução da latência, uma arquitetura mais plana com a redução de números de nós seria mais indicada. Mas por outro lado, quando o objetivo é a otimização da interoperabilidade com outras redes, a adição de novos nós de controle para alcançar tal objetivo, aumenta a complexidade e diâmetro da rede.

Deste modo a evolução do sistema resultou em uma arquitetura modular, chamada de arquitetura SAE (*System Architecture Evolution*), definida pelo grupo 3GPP como a arquitetura para a quarta geração das tecnologias definidas por este grupo, comumente conhecida como 4G-LTE (*Long Term Evolution*).

A SAE é baseada em uma arquitetura plana, *Flat Architecture*, com menos nós envolvidos, o que simplifica a implementação, reduz a latência e aumenta o desempenho. A evolução das arquiteturas até a arquitetura SAE pode ser observado na figura 2.2.

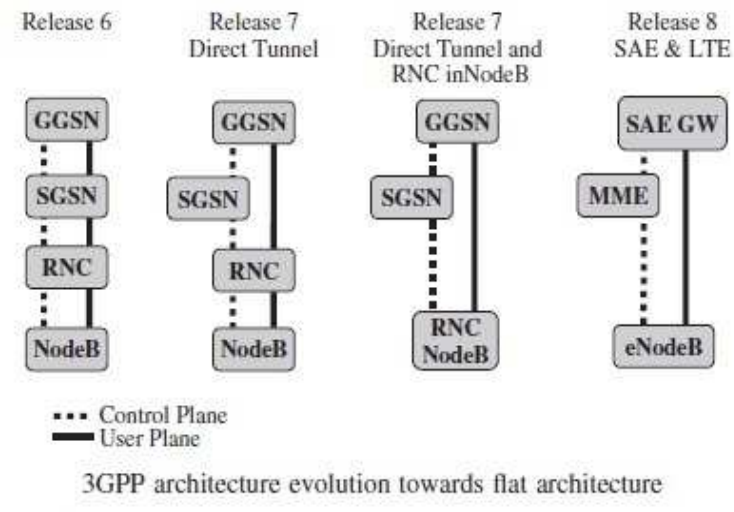


Figura 2.2 – *Arquitetura plana SAE (modificado de [31]).*

Visto que na prática nenhuma implementação deverá suportar todas as interoperabilidades possíveis entre diferentes redes, a especificação SAE foi dividida em duas trajetórias [31]:

- *GPRS enchacements for Evolved UTRAN (E-UTRAN) access*: arquitetura e funções em ambientes nativos 3GPP com E-UTRAN e outras 3GPP ANs, definindo os procedimentos de interoperabilidade entre elas, comumente utilizando GTP (*GPRS Tunneling Protocol*) como protocolo de mobilidade de rede.

- *Architecture enhancements for non-3GPP access*: arquitetura e funções na interoperabilidade com redes de acesso não 3GPP, como cdma2000 (*Code Division Multiple Access*) e HRPD (*High Rate Packet Data*), comumente utilizando os protocolos do IETF como o MIP (*Mobile Internet Protocol*) e o PMIP (*Proxy MIP*) para mobilidade.

Tomando como referência o sistema formado apenas pelo acesso E-UTRAN, para descrever os elementos básicos da arquitetura SAE, podemos segmenta-la em quatro partes, domínios, principais, como pode ser observado na figura 2.3, sendo eles [31]:

1. Service Domain – domínios externos para ofertas outros serviços como Internet e IMS;
2. User Equipament (UE);
3. Evolved-UTRAN;
4. Envolved Packet Core (EPC) Network

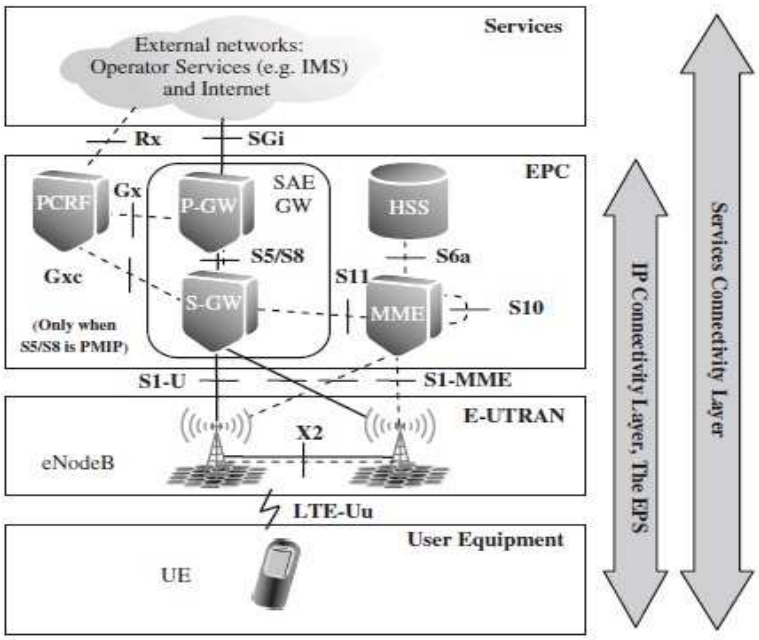


Figura 2.3 – *Domínios da Arquitetura SAE [31]*.

O Service Domain e o UE possuem funções equivalentes aos outros sistemas 3GPP, já o E-UTRAN e EPC, são os domínios que sofreram modificações significativas na arquitetura. A comunicação realizada entre os domínios UE, E-UTRAN e EPC representa a camada de conectividade IP do sistema, denominada como EPS (*Evolved Packet System*), onde todas as comunicações são baseadas na comutação de pacotes IP, não existe mais a presença dos nós baseados na comutação por circuitos e as interfaces vistas nas arquiteturas anteriores do 3GPP, sendo tudo designado para operar com IP.

Já a comunicação realizada entre o UE até o Service Domain, representa a camada de conectividade de Serviço. Nesta camada sistemas como o IMS (*IP Multimedia Sub-system*), podem ser utilizados como parte desta camada para prover serviços utilizando a camada de conectividade IP fornecida pelas camadas mais baixas para suportar, por exemplo, serviços VoIP e conectividade com sistemas legados como PSTN (*Public Switched Telephone Network*) e ISDN (*Integrated Service Digital Network*) por meio de *Media Gateways* (MG). Assim, o EPC não contém e não possui necessidade de conexão direta com as redes legadas e comutadas por circuitos, como ISDN e PSTN. Em questão de funcionalidades, o EPC possui função equivalente ao *packet switch domain* das outras redes 3GPP existentes, entretanto o arranjo das funções e dos nós do EPC são considerados completamente novos.

Em relação à evolução da parta a radio da rede, a parte RAN, o desenvolvimento no E-UTRAN é concentrado em um único nó, chamado de eNodeB (*Evolved Node B)*. Todas as funções de rádio são concentradas nele, sendo o ponto final para todos os protocolos relacionados à rádio. Do ponto de vista de rede, E-UTRAN é formado por uma topologia *mesh* onde os eNodeBs se conectam entre eles utilizando uma interface lógica chamada de X2.

O tratamento do tráfego de dados dos usuários dentro do EPC, ou seja, tratamento do UP, é realizado pelo SAE GW (*Gateway*), formado pelo S-GW (*Serving Gateway*) e P-GW (*Packet Data Network Gateway*). Existe a possibilidades de implementar o SAE GW em um único nó, mas o padrão define as interfaces específicas entre o S-GW e P-GW, assim como todas as operações realizadas por cada um.

2.2.2 – Redes móveis IEEE: WLAN

As redes sem fio definidas pelo IEEE na família de padrões da 802.11, chamadas de redes WALN (*Wireless LAN*) ou comumente de redes Wi-Fi, sendo este termo na verdade o nome utilizado por produtos certificados de acordo com a associação *Wi-Fi Alliance*, permite a comunicação entre os dispositivos através de ondas de radio e a interoperabilidade de comunicação destes com os dispositivos conectados por meio de cabeamento. Algumas das principais características em relação a um ambiente WLAN são [47]:

- Redução de custos e maior flexibilidade e mobilidade de implementação;
- Questões de segurança e privacidade;
- Transmissões sujeitas a interferências do ambiente;
- Regulamentações dependem da localidade;

- Quadros (*Frames*) e acesso ao meio realizado de modo diferente do que no padrão Ethernet.

Os padrões da família IEEE 802.11 definem os aspectos relacionados à camada física (PHY) e à camada de enlace (MAC) do modelo de referência OSI de acordo com a figura 2.4, sendo a camada de enlace neste caso dividida em duas subcamadas, onde as principais funções desempenhadas por cada uma delas são [48]:

- Camada Física (PHY): responsável pela transmissão utilizando radiofrequência ou infravermelho, onde são especificados os parâmetros de banda, largura de banda, codificação, modulação e filtragem da transmissão a radio.
- Camada de Enlace: responsável pelos métodos de controle de acesso ao meio e troca de quadros entre os dispositivos através de um meio físico comum, dividida em duas subcamadas:
    - Subcamada MAC (Media Access Control): responsável pelo controle e métodos de acesso ao meio, como o CSMA/CA (*Carrier Sense Multiple Access/Colission Avoidance*);
    - Subcamada LLC (*Logical Link Control*): responsável pelas conexões lógicas e realizar a interface com as camadas superiores, desempenhando funções como esquemas de retransmissão e detecção de erros, controle de admissão, gerenciamento de conexões e controle de recursos rádio.

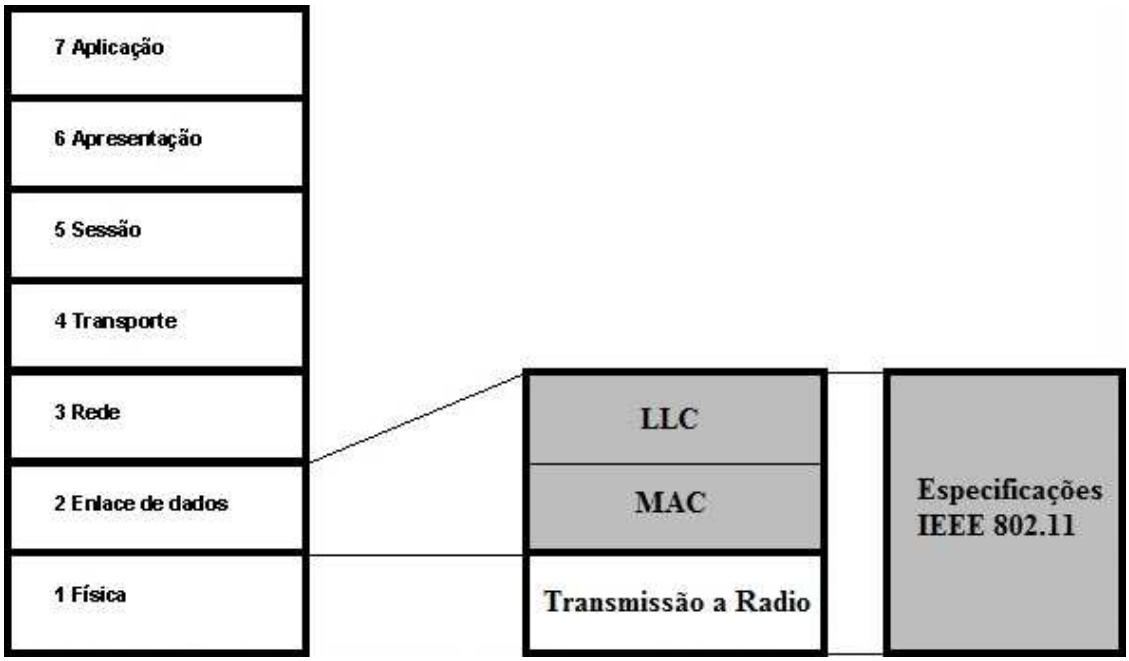


Figura 2.4 – *Camadas do RM-OSI especificadas pela família IEEE 802.11 (modificado de [48]).*

A família IEEE 802.11 especifica vários padrões para a comunicação em redes WLAN, onde a escolha de qual padrão utilizar para determinado cenário, é influenciada basicamente pela taxa de transmissão de dados alcançada pelo padrão e pela faixa de frequência utilizada, sendo que as taxas de dados são influenciadas de acordo com a modulação utilizada, como a modulação OFDM (*Orthogonal Frequency Division Multiplexing*) que permite taxas mais elevadas, sendo mais complexo e a modulação DSSS (*Direct Sequence Spread Spectrum*), que é mais simples, porém permite taxas menores. Os principais padrões utilizados e suas características são descritos abaixo:

- IEEE 802.11a
  - Aprovação oficial em 1999 [49];
  - Modulação OFDM;
  - Faixa de frequência utilizada de 5,8 GHz;
  - Taxa de transmissão de até 54 Mbps;
  - Vantagens: redução de problemas de interferência devido à frequência utilizada, antenas menores, velocidade superior em comparação aos padrões da época e 12 canais não sobrepostos;
  - Desvantagens: custo mais elevado, maior absorção por obstáculos reduzindo o alcance e baixo desempenho por bloqueios.

- IEEE 802.11b
  - Aprovação oficial em 1999 [49];
  - Modulação DSSS
  - Faixa de frequência utilizada de 2,4 GHz;
  - Taxa de transmissão de até 11 Mbps;
  - Vantagens: baixo custo, menor absorção com maior alcance e menor bloqueio;
  - Desvantagens: propenso a interferência de outros equipamentos devido a faixa de frequência utilizada e menores taxas de transmissão.

- IEEE 802.11g
  - Aprovação oficial em 2003 [49];
  - Modulação OFDM e DSSS para compatibilidade com o 802.11b
  - Faixa de frequência utilizada de 2,4 GHz;
  - Taxa de transmissão de até 54 Mbps;

- Vantagens: maior taxa de transmissão, menor absorção com maior alcance e menor bloqueio;
- Desvantagens: propenso a interferência de outros equipamentos devido a faixa de frequência utilizada.

- IEEE 802.11n
  - Aprovação oficial em 2009 [49];
  - Modulação MIMO-OFDM (*Multiple-Input Multiple-Output – OFDM*), baseada na melhoria dos algoritmos e no de múltiplas antenas;
  - Faixa de frequência utilizada de 2,4 GHz e/ou 5 GHz;
  - Taxa de transmissão de até 600 Mbps quando utilizado o modo MIMO 4x4;
  - Vantagens: maior taxa de transmissão e maior alcance;
  - Desvantagens: custos e possibilidade de equipamentos que foram lançados antes do lançamento oficial do padrão não serem compatíveis com este.

- IEEE 802.11i
  - Aprovação oficial em 2004 [49];
  - Criado para aperfeiçoar as funções de segurança da família 802.11;
  - WEP, TKIP, AES, IEEE 802.1x;
  - Permite a adição de novas técnicas sem a substituição de hardware.

Em relação à arquitetura das redes WLAN, existem basicamente quatro modalidades de conexão para a infraestrutura de comunicação sem fio, apresentadas na figura 2.5, sendo [47]:

- IBSS (*Independent Basic Service Set*)

Números de pontos de acesso utilizados	0
Conexão	P2P (*Peer-to-Peer*)
Modo	Ad Hoc
Cobertura	BSA (*Basic Service Area*)

- BSS (*Basic Service Set*)

Números de pontos de acesso utilizados	1
Conexão	Cliente ao AP
Modo	Infraestrutura
Cobertura	BSA (*Basic Service Area*)

- ESS (*Extended Service Set*)

Números de pontos de acesso utilizados	≥ 2
Conexão	Cliente ao AP
Modo	Infraestrutura
Cobertura	ESA (*Extented Service Area*)

- DS (*Distribution Systems*) – Mobilidade

Números de pontos de acesso utilizados	≥ 2
Conexão	Cliente ao AP
Modo	Infraestrutura
Cobertura	ESA (*Extented Service Area*)

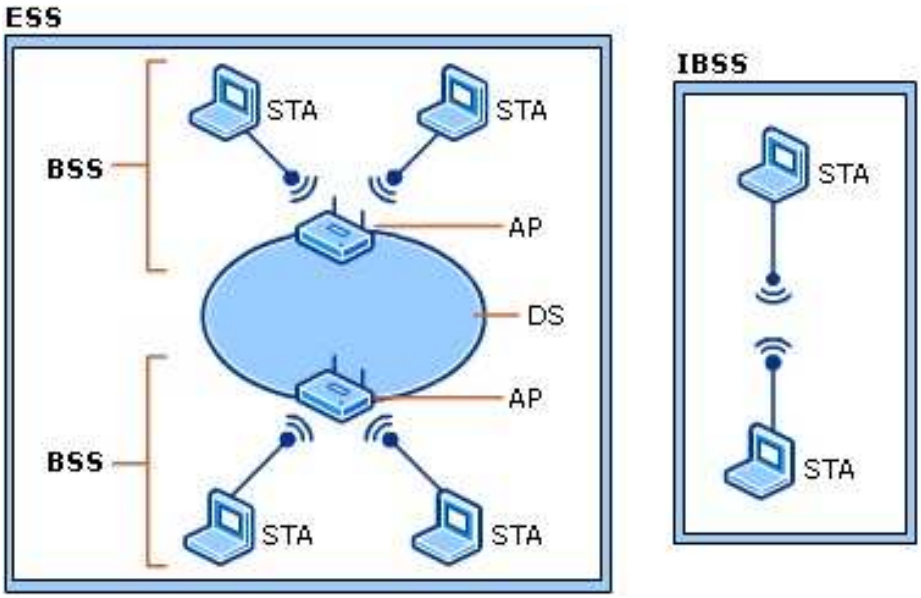


Figura 2.5 – *Arquiteturas WLAN [50]*.

A conexão de um *host* a determinada rede WLAN, ou seja, a determinado ponto de acesso, é realizada através do processo chamado de *Join Process*, conhecido como processo de associação. O processo de associação é basicamente constituído de três principais etapas: 802.11 Probing (Investigação), 802.11 Authentication (Autenticação) e 802.11 Association (Associação) [47].

Na primeira etapa, 802.11 Probing, os *host*s iniciam uma varredura para coletar informações das redes WLAN que estão disponíveis dentro do seu alcance. Esta varredura pode ser realizada em modo passivo, com os *host*s identificando as redes através de pacotes

BEACON enviados pelos pontos de acesso para anunciar a presença da rede, onde as principais informações contidas nestes pacotes são o SSID da rede, as taxas suportadas e a segurança utilizada ou de modo ativo, com os *hosts* enviando pacotes de *probe request* para procurar por uma rede específica ou redes em que os pontos de acesso estão configurados para responder a este tipo de requisição também com pacotes BEACON.

Depois da etapa de investigação, o *host* define qual rede irá se conectar iniciando a etapa de 802.11 *Authentication*, onde neste estágio são verificados os mecanismos de autenticação para verificar se o *host* tem autorização para conectar-se a rede, onde verificações adicionais como de SSID e de endereço MAC são realizadas.

Após o *host* ser autenticado na rede e possuir as informações necessárias para estabelecer um enlace de comunicação entre o *host* e o ponto de acesso, a última etapa, 802.11 *Association*, consiste na troca de *association messages* onde o enlace de comunicação é estabelecido com o *access point* mapeando uma porta lógica *Association ID* (AID) para o *host*. O AID funciona como uma porta de switch e auxilia a infraestrutura controlar os quadros destinados ao *host*.

2.3 – REDES DEFINIDAS POR SOFTWARE (SDN) E OPENFLOW

Tradicionalmente as redes de comunicação são constituídas por equipamentos que desempenham tanto as funções de encaminhamento de pacotes do plano de dados, quanto as funções de controle e sinalização do plano de controle. Embora muitos destes equipamentos possuam em sua arquitetura a separação destes planos, com a operação em cada um deles possuindo certa independência, no intuito de prover uma maior estabilidade e disponibilidade do equipamento e assim da rede, este característica não é aplicada na infraestrutura de rede como um todo.

Nestas redes cada equipamento possui a sua própria inteligência e autonomia para as tomadas de decisão de ambos os planos, operando de forma autônoma dentro da infraestrutura de rede. Além disso, em redes constituídas por equipamentos de diferentes fabricantes, que é o caso da maioria das redes de grande porte de *Internet Service Provider*, cada fabricante e as vezes até mesmo equipamentos diferentes do mesmo fabricante, possuem a sua própria linguagem e arquitetura, tornando o gerenciamento do plano de controle da infraestrutura de rede bastante complexa.

Esta arquitetura tradicional possui alguns desafios frente ao dinamismo e velocidade dos negócios realizados nos dias atuais, onde os serviços *on-demand* tem se tornado um forte fator competitivo no mercado de telecomunicações, como:

- Sobrecarga administrativa para realizar a operação do plano de controle da infraestrutura de rede;
- Dependência da infraestrutura em relação à arquitetura e funcionalidades implementadas nos equipamentos de acordo com o fabricante;
- Complexidade e alta demanda de tempo na implementação de mudanças ou de novos serviços;
- Uso não otimizado da infraestrutura disponível e retardo na convergência em certas situações de falha;
- Maior rigidez da infraestrutura para se adaptar a mudanças de estratégias do negócio;

Neste cenário surgiu um novo paradigma de arquitetura de redes chamado de Redes Definidas por *Software* (do inglês, *Software-Defined Networking - SDN*), com o objetivo principal de adicionar uma maior flexibilidade e independência da infraestrutura de rede em relação aos fabricantes, tornando a rede programável. Do qual o protocolo *OpenFlow* vem se estabelecendo como o padrão aberto para a implementação de redes SDN.

2.3.1 – Paradigma SDN

O paradigma de arquitetura de redes SDN tem como característica fundamental a separação das funções de plano de controle e plano de dados da infraestrutura de rede como um todo, onde os dispositivos de rede passam apenas a desempenhar funções de plano de dados, como o encaminhamento de pacotes. Toda a inteligência e controle da rede, ou seja, as funções de plano de controle, passam a ser desempenhadas de modo centralizado em um dispositivo chamado de controlador SDN [36].

A origem do paradigma vem da arquitetura de redes *Ethane*, onde as políticas de controle de acesso eram executadas de modo distribuído na rede a partir de um dispositivo central responsável pelas políticas de segurança. Quando um novo fluxo de dados era identificado por algum dispositivo da rede, este consultava o dispositivo central de políticas para buscar informações de como o novo fluxo deveria ser tratado [36].

Deste modo a tomada de decisão era realizada de forma centralizada e de acordo com as políticas definidas no dispositivo central, que repassava as informações para os demais

dispositivos, através de programação de entradas nas tabelas de encaminhamento destes, de como os fluxos deveriam ser tratados.

Na arquitetura tradicional a rede é constituída por diversos dispositivos que possuem um sistema operacional embarcado, geralmente proprietário da fabricante, para controlar o hardware disponível. O plano de controle precisa então ser configurado de acordo com a especificação e comandos de cada fabricante em cada dispositivo da rede para que os dados possam ser encaminhados no plano de dados de acordo com o desejado.

No paradigma SDN os dispositivos de rede não possuem mais a função de plano de controle, passam a ser partes integrantes apenas do plano de dados, executando as funções de encaminhamento de acordo com o hardware fornecido, sendo considerados apenas como peças de hardware que realizam o encaminhamento de dados da rede. Um dispositivo central, o controlador SDN, é o responsável pelas funções de plano de controle, definindo como o hardware disponível na infraestrutura, os dispositivos de rede, devem encaminhar os pacotes. Essa separação permite que cada elemento da arquitetura fique dedicado em desempenhar apenas uma das funções, concentrando todo o poder computacional disponível nesta função.

A arquitetura SDN é basicamente constituída por três camadas: a camada superior *Northbound*, a camada do meio *Controller* e a camada inferior *Southbound*. A figura 2.6 apresenta a arquitetura básica SDN.

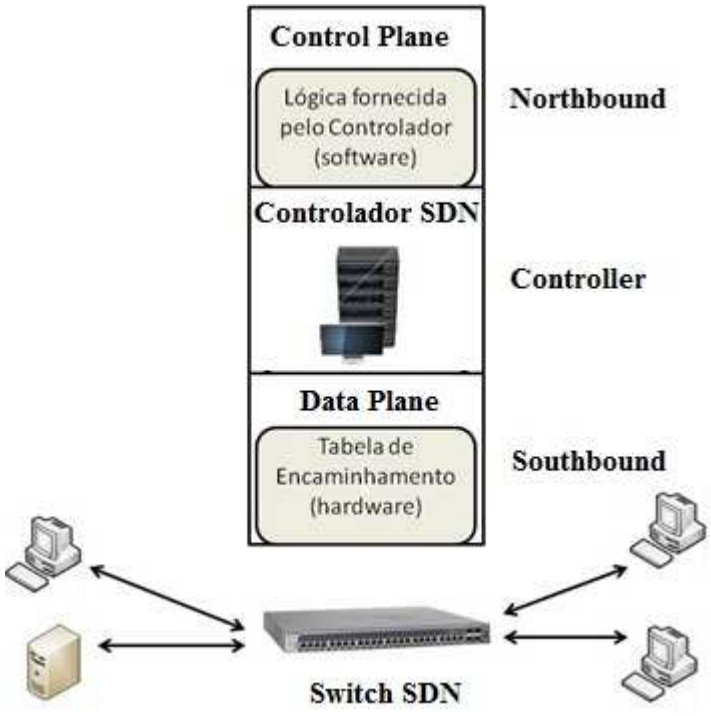


Figura 2.6 – *Arquitetura SDN (baseado em [36]).*

A camada *Controller* é formada pelo controlador SDN, responsável por conectar as duas outras camadas, realizando a interface com a camada superior e inferior, provendo a separação entre o plano de controle e o plano de dados da rede. A camada *Northbound* é responsável pelos serviços fornecidos de acordo com as aplicações executadas, realizando o controle efetivo das funções de plano de controle da rede, como virtualização, firewall, políticas de controle, políticas de encaminhamento, mobilidade e balanceamento de carga. A camada *Southbound* é responsável pela comunicação com a rede física, onde é executado o plano de dados da rede para o encaminhamento de pacotes.

As redes SDN permitem uma maior flexibilidade na medida em que além do plano de controle de toda a infraestrutura de rede estar concentrado em um ponto central, é inserido também a programabilidade da rede pela camada *Northbound*, onde as aplicações que definem os serviços ofertados na rede são executadas. Isso permite que toda a rede possa ser controlada de acordo com as necessidades específicas de modo rápido e prático. Alguns benefícios da arquitetura SDN são [36]:

- Plano de controle centralizado e programável;
- Controle do plano de dados de toda a rede como um todo;
- Flexibilidade e alto grau de adaptação;
- Automatização da configuração dos dispositivos de rede;
- Automação das políticas de segurança;
- Melhor visualização e gerenciamento da infraestrutura de rede;
- Engenharia de tráfego mais flexível;
- Redução dos custos operacionais;
- Redução do erro humano na configuração dos dispositivos de rede, na medida em que as configurações são realizadas de modo centralizado no concentrador e não em cada ativo da rede;
- Detecção e filtragem de tráfego indesejado;
- Independência da infraestrutura de rede em relação à arquitetura e funcionalidades implementadas pelos equipamentos de acordo com o fabricante;
- Baixa complexidade para a implementação de mudanças ou novos serviços;
- Otimização da infraestrutura disponível;
- Rápida convergência em caso de falhas;
- Melhor alinhamento entre a infraestrutura de rede e a estratégia de negócios.

2.3.2 – Protocolo OpenFlow

O protocolo *OpenFlow* é um padrão aberto que foi originado na Universidade de Stanford com o objetivo de validar novas propostas e protocolos de rede desenvolvidos por pesquisadores que necessitavam testa-las na rede acadêmica. Sendo atualmente mantido e desenvolvido pela *Open Network Foundation* (ONF), organização destinada a promoção e adoção de redes SDN através do desenvolvimento de padrões abertos.

É o primeiro padrão de interface de comunicação definido entre a camada de controle e de encaminhamento da arquitetura SDN. Especifica como o controlador deve se comunicar com os dispositivos de rede através do protocolo, permitindo a manipulação e o acesso direto do plano de dados dos dispositivos sem a necessidade de exploração ou abertura dos códigos proprietários dos fabricantes.

Existem várias versões do protocolo, sendo a primeira lançada em maio de 2008, versão 0.2.0 e a mais recente em março de 2015, versão 1.5.1. As duas versões mais amplamente utilizadas nas redes SDN são a versão 1.0 e a 1.3, que especificam que as principais características que um switch *OpenFlow* deve implementar são:

- Tabelas de fluxos;
- Canal de comunicação seguro com o controlador SDN;
- Protocolo *OpenFlow*.

A figura 2.7 ilustra as principais características e a arquitetura básica *OpenFlow*.

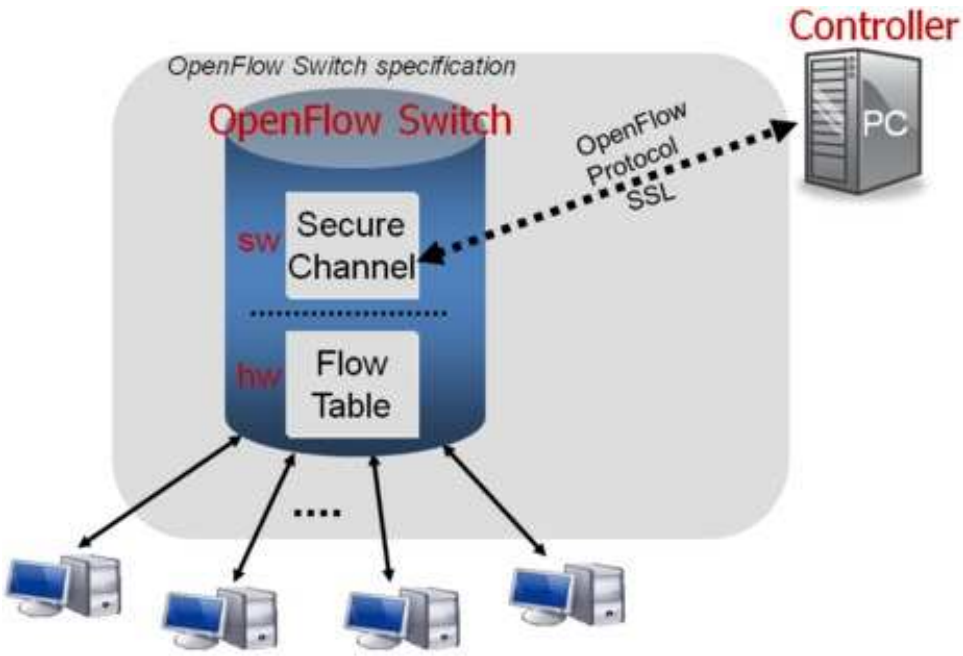


Figura 2.7 – *Arquitetura e características OpenFlow [51]*.

A tabela de fluxo contém a identificação de um fluxo, a ação associada e contadores relacionados. O canal de comunicação seguro entre o switch *OpenFlow* e o controlador SDN deve permitir que todas as mensagens de controle e os comandos sejam transmitidos de modo seguro na rede. E o protocolo utilizado para está comunicação deve ser o *OpenFlow*.

O principal objetivo do *OpenFlow* é a definição de uma interface aberta bem definida para o controle do plano de dados da infraestrutura de rede e a sua separação clara do plano de controle nos dispositivos de rede, sendo implementado através de módulos para que a comunicação com o controlador SDN seja feita de modo independente, segura e segmentada do tráfego real de dados na rede encaminhados de acordo com os fluxos estabelecidos.

Os switches convencionais executam o encaminhamento dos quadros de acordo com a correspondência de determinado tráfego com um conjunto de regras inseridas estaticamente ou por meio do aprendizado automático, onde as regras são inseridas em sua tabela de encaminhamento na medida em que ele toma conhecimento de qual interface determinado elemento se encontra. Os switches *OpenFlow* realizam o encaminhamento de forma semelhante, realizando a correspondência do tráfego com as entradas presentes na sua tabela de fluxos, mas sendo estas entradas criada e atualizadas pelo controlador SDN e não de maneira autônoma. O protocolo *OpenFlow* define que estes fluxos são identificados pela combinação dos campos da camada física até a camada de transporte de acordo com a figura 2.8 [36].

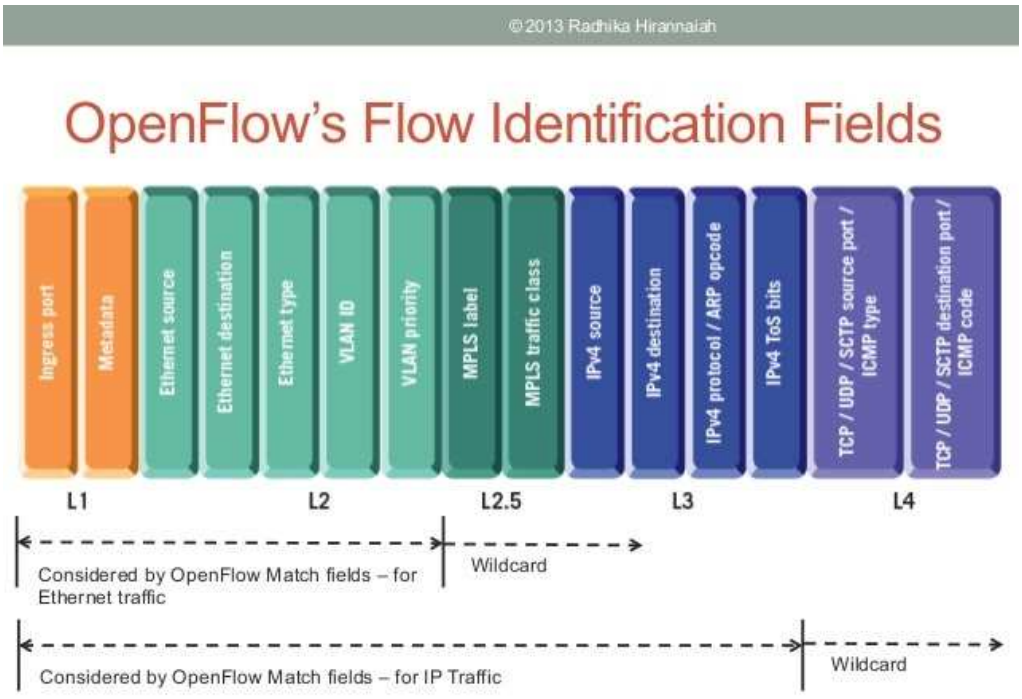


Figura 2.8 – *Campos de identificação para Fluxos do OpenFlow [36]*.

2.4 – CONSIDERAÇÕES FINAIS

Este capítulo abordou, inicialmente, o gerenciamento de mobilidade em redes baseadas em IP, categorizado principalmente de acordo com dois aspectos: como o encaminhamento dos dados é realizado no plano de dados e quais unidades participam do processo de sinalização no plano de controle.

Em relação ao encaminhamento dos dados no plano de dados, temos a categorização de gerenciamento centralizado, quando uma única unidade central é responsável por realizar o encaminhamento dos dados dos *hosts* em mobilidade, e gerenciamento distribuído, quando o encaminhamento dos dados não é vinculado à uma única unidade específica. Esta última possui duas subcategorizações: parcialmente distribuído e completamente distribuído. No gerenciamento parcialmente distribuído as funções de plano de controle ainda são centralizadas de algum modo em um ponto de controle da rede e no completamente distribuído estas funções são executadas por diferentes unidades da rede.

A categorização de acordo com o processo de sinalização no plano de controle, é definida como baseada no móvel, quando o *host* final tem consciência dos processos de mobilidade participando destes e como baseado na rede, quando toda a sinalização ocorre de modo transparente para o *host* final, sem que este tenha conhecimento ou participe.

Além destas categorizações principais são propostas outras como *clean-state*, *architecture-dependent*, *peer-to-peer* e *based on or extending existing IETF protocols* [4].

O capítulo aqui tratado abordou também o ambiente de redes heterogêneas, compostas por diferentes tipos de tecnologias de acesso como as redes móveis 4G-LTE (definidas pelo 3GPP) e as redes WLAN (definidas pelo IEEE), fornecendo benefícios como uma maior área de cobertura e flexibilidade para o usuário final. Em tais redes, o gerenciamento de mobilidade é um aspecto de fundamental importância para prover um serviço adequado ao usuário final quando este realiza o *handover* entre as redes.

Neste contexto, o novo paradigma de arquitetura de redes chamado de SDN, possibilita adicionar flexibilidade e otimização da infraestrutura de rede, podendo ser utilizado eficientemente em prol do gerenciamento de mobilidade distribuído em redes heterogêneas.

3 – TRABALHOS RELACIONADOS

Este capítulo é dedicado à apresentação e discussão de trabalhos voltados ao gerenciamento de mobilidade, apresentando quatro abordagens distintas do IETF (*Internet Engineering Task Force*), sendo dois padrões para o gerenciamento de mobilidade centralizado e dois *drafts* para o gerenciamento de mobilidade distribuído.

3.1 – MOBILE IPv6 (MIPv6)

Como o primeiro dentre os padrões estabelecidos pelo IETF para mobilidade baseada no protocolo IPv6, o Mobile IPv6 (MIPv6) [8] apoia-se em uma entidade central chamada de *home agent* (HA) localizada na rede de origem do *mobile node* (MN), chamada de *home network* (HN).

O HA é responsável pelas funções de plano de controle e de plano de dados. No plano de controle este agente realiza funções como o *binding cache*, que armazena informações de localização do MN associando o seu endereço IP permanente, chamado de *home address* (HoA), com o seu endereço IP temporário, chamado de *care-of address* (CoA). Já no plano de dados, realiza o redirecionamento dos pacotes recebidos que são destinados ao HoA para o MN através de tunelamento.

A operação do MIPv6 pode ser descrita com o *mobile node* possuindo um endereço IP permanente, seu *home address*, que faz parte do mesmo escopo de endereçamento de rede da sua *home network*. Este endereço é sempre utilizado pelo *mobile node* para se comunicar com os demais *hosts*, ou seja, deve ser sempre utilizado como o endereço de origem de todos os pacotes IP enviados por ele. Enquanto o *mobile node* estiver conectado em sua *home network* nenhum serviço de mobilidade é utilizado, uma vez que o roteamento padrão permite normalmente a comunicação do *mobile node* utilizando o *home address* com o *correspondent node* (CN).

Mas quando o *mobile node* está em um cenário de mobilidade conectado a outras redes que não seja a sua *home network*, conectando-se a uma *foreign network*, é necessário o uso de um endereço IP que esteja associado a sua nova localização para refletir a sua posição atual, uma vez que o *home address*, utilizado para realizar todas as comunicações e manter a continuidade de sessão IP, já não é capaz de refletir esta posição e permitir que através do roteamento padrão a comunicação com o *correspondent node* possa ser mantida. Neste

momento o *mobile node* também deve possuir um IP para ser utilizado no serviço de mobilidade, o endereço IP *care-of address*.

O *care-of address* é utilizado para que o *home agent* localizado na *home network* do *mobile node* possa saber a localização deste e assim encaminhar todos os pacotes destinados ao *home address* do *mobile node* corretamente para ele em sua nova localização, uma vez que o roteamento padrão da infraestrutura de rede envia todos os pacotes destinados ao *home address* para a *home network* e não para realmente onde está localizado o *mobile node*. O encaminhamento dos pacotes realizado pelo *home agent* para o *mobile node* é realiado através de um túnel estabelecido diretamente com o *mobile node* utilizando o seu endereço *care-of address*. A operação básica do protocolo pode ser observada na figura 3.1.

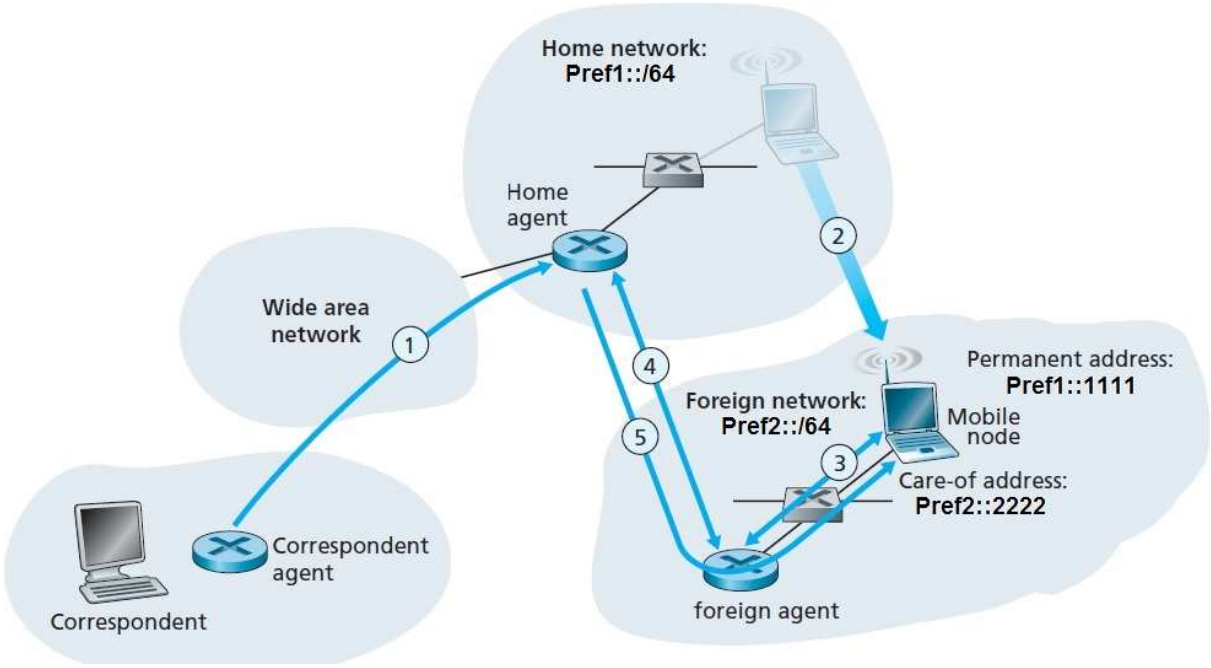


Figura 3.1 – *Operação MIPv6 (modificado [33]).*

São definidos dois tipos de *care-of address*, o *Foreign Agent care-of address* e o *Co-located care-of address*. A utilização de um ou outro determina como será o funcionamento e a atuação do *mobile node* e do *mobility agent* no processo de mobilidade, sendo:

- *Foreign Agent (FA) care-of address*: é um endereço do próprio *foreign agent* onde o *mobile node* está registrado, disponibilizado por ele através de *Agent Advertisement message* para que o *mobile nodes* utilize o serviço de mobilidade na *visited network* através dele.
- *Co-located care-of address*: é um endereço alocado diretamente ao *mobile node*, a uma de suas interfaces de rede, por exemplo, através de um servidor DHCP, para

que ele utilize diretamente o serviço de mobilidade sem a necessidade de um *foreign agent*.

Quando se utiliza o *FA care-of address* o túnel para o encapsulamento e entrega dos pacotes para o *mobile node* é estabelecido entre o *home agent* e o *foreign agent*, com estes realizando o *mobility binding* entre *home address* e *care-of address*. O *home agent* é responsável por encapsular os pacotes e entrega-los para o *foreign agent* através do túnel e este é responsável por desencapsular os pacotes e entrega-los diretamente para o *mobile node*.

Este modo é indicado como preferencial, pois além do *mobile node* não se envolver com ações de encapsulamento e estabelecimento de túnel, preservando seu processamento e bateria, possibilita que o *foreign agent* possa realizar uma espécie de NAT (*Network Address Translation*) na formação dos túneis, utilizando um único IP *FA care-of address* para atender a mais de um *mobile node*, aumentando a eficiência.

Ao se utilizar o *Co-located care-of address*, o túnel é estabelecido diretamente entre o *home agent* e o *mobile node*, uma vez que o *care-of address* está alocado em uma das interfaces de rede do *mobile node*. Neste modo o *mobile node* passa a ser um dos responsáveis na formação do túnel e pelo desencapsulamento dos pacotes recebidos pelo *home agent*, com este mantendo as mesmas funções que no modo anterior.

Uma das vantagens deste modo é que não existe a participação do *foreign agent* no processo, podendo então não ser necessário a sua presença para que o *mobile node* utilize o serviço de mobilidade, uma vez que este possui um endereço próprio da *visited network* e é um dos *end-points* do túnel. A desvantagem é a sobrecarga do *mobile node* com os processos de encapsulamento, formação do túnel e *mobility binding*, que acabam utilizando mais recursos deste, como, processamento e bateria. Outra desvantagem é a necessidade de alocação de um único *IP care-of address* para atender a apenas um *mobile node*, que em um cenário com alta mobilidade e grande número de *mobile nodes* aumenta consideravelmente o uso de recursos de alocação IP, tornando a manutenção e operação do serviço mais custosa.

No MIPv6 o *home agent* é responsável tanto pelo plano de controle, quanto pelo plano de dados de todos os *mobile node* em processo de mobilidade, com este atuando ativamente neste processo, trocando informações sobre a sua localização e até mesmo estabelecendo túneis, sendo então uma solução de gerenciamento categorizada como centralizada, por todo o tráfego, de controle e de dados, ser processado pelo *home agent* e *client-based*, onde o *mobile node* atua efetivamente nos procedimentos de sinalização do plano de controle.

3.2 – PROXY MOBILE IPv6 (PMIPv6)

Considerado como uma evolução do MIPv6, o Proxy MIPv6 [9] transfere funções importantes de plano de controle presentes no MN para outra unidade na rede chamada de *mobile access gateway* (MAG) e substitui o HA por uma entidade chamada de *local mobility anchor* (LMA).

O LMA e o MAG são responsáveis pelos processos de sinalização do plano de controle para fornecer a mobilidade de modo transparente para o MN. Um endereço IP chamado de *home network prefix* (HNP) é alocado para o MN dentro de um domínio conhecido como *PMIPv6 domain* ou *localized mobility domain*, formado pelo conjunto de LMA e MAGs, onde o LMA é responsável pelo encaminhamento do tráfego destinado ao HNP e os MAGs são responsáveis por prover conectividade com suporte à mobilidade ao MN. A operação básica do PMIPv6 pode ser observado na figura 3.2.

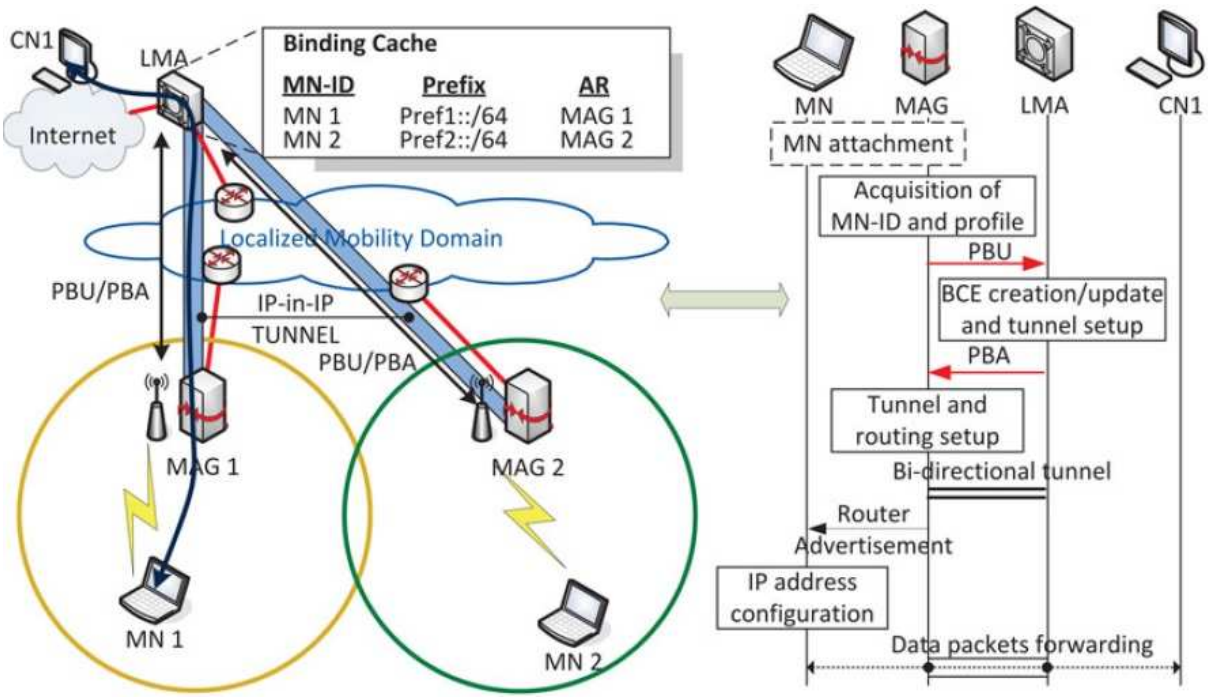


Figura 3.2 – *Operação PMIPv6 [16]*.

Durante a movimentação do MN dentro deste domínio, os processos de sinalização utilizando mensagens de *proxy binding update* (PBU) e *proxy binding acknowledgment* (PBA), são realizados entre o LMA e os MAGs com o objetivo principal de informar a atual localização do MN para o LMA. Estas informações são armazenadas no LMA através de *binding cache entry* (BCE) que contém o prefixo HPN do MN, a sua identificação e o

endereço do MAG que está o servindo, chamado de *Proxy Care-of Address* (Proxy-CoA). As informações contidas no BCE são utilizadas para estabelecer um túnel entre o LMA e os MAGs que estão fornecendo o serviço de mobilidade para os MN, de modo que o LMA possa redirecionar todo o tráfego destinado ao MN para o MAG correspondente e este encaminhar de forma transparente para o MN.

Como neste padrão o MN não participa ativamente dos procedimentos de sinalização do plano de controle, sendo a mobilidade executada de forma transparente para ele, mas ainda existe a centralização das funções de plano de dados em uma unidade central, o PMIPv6 é categorizado como uma solução centralizada e *network-based*.

3.3 – DRAFT IETF – JAEHWOON [22]

Este *draft* do IETF apresenta uma proposta *network-based fully* DMM baseada no protocolo PMIPv6, onde todas as funções do plano de controle e do plano de dados são distribuídas entre os *mobile access gateways* (MAGs).

Não existe uma unidade central, como o *local mobility anchor* (LMA), responsável pelos processos de sinalização e de encaminhamento do tráfego para todos os *mobile nodes* presentes no domínio PMIPv6, estas responsabilidades são distribuídas entre as entidades localizadas na borda da rede de acesso, as executando de forma transparente para o *mobile node*.

A operação do protocolo baseia-se em atribuir um prefixo de rede chamado de PREF (em um conceito de *supernet*) ao domínio de mobilidade, onde para cada MAG deste domínio é alocado uma sub-rede diferente pertencente ao PREF. Embora cada MAG seja responsável por uma sub-rede específica, todos anunciam o mesmo prefixo PREF como a rede a qual pertencem pelas interfaces que fornecem o serviço de mobilidade.

Quando o *mobile node* entra pela primeira vez no domínio, ele recebe um endereço IP da sub-rede correspondente ao MAG a qual está se conectando, com o prefixo PREF sendo indicado como a rede. Ao *mobile node* mudar o seu ponto de conexão à rede para um novo MAG, recebendo novas mensagens *Router Advertisement* (RA) com o campo *option information* contendo o mesmo prefixo de rede PREF, o *mobile node* considera que ainda está conectado à mesma rede e mantém o seu endereçamento de acordo com o que foi estabelecido na conexão com o primeiro MAG. Nesse momento, o novo MAG identifica que o *mobile node* está utilizando um IP pertencente ao prefixo PREF, mas que não faz parte da sua sub-rede, iniciando um processo de sinalização com o MAG responsável pelo endereço IP

utilizado pelo *mobile node*, a fim de se estabelecer um túnel entre eles para o redirecionamento de todo o tráfego pertencente ao *mobile node*.

Novos procedimentos e mensagens de *Distributed Proxy* e *FLUSH* são adicionadas ao protocolo PMIPv6 para que a sinalização e o próprio encaminhamento de tráfego sejam realizados diretamente entre os MAGs de modo completamente distribuído na rede. Neste cenário, o MAG de origem do *mobile node*, ou seja, o MAG que é responsável pelo endereço IP utilizado pelo *mobile node* que executou um *handover*, nomeado a partir de então para referências futuras como *anchor*-MAG (A-MAG), atua como se fosse um LMA, desempenhando funções como criar e manter o *binding cache entry* (BCE), realizar o *buffer* e o redirecionamento de pacotes através dos túneis estabelecidos.

O processo de sinalização é iniciado quando um MAG identifica a movimentação de um *mobile node* dentro do domínio. Na associação do *mobile node* ao MAG, este envia uma mensagem RA indicando o prefixo PREF como sendo a rede a qual o *mobile node* irá se conectar. Se logo após está mensagem o MAG receber um DHCP *request* do *mobile node*, o MAG considera que o *mobile node* está entrando pela primeira vez no domínio, ou seja, não se trata de um *handover*.

Mas se ao invés disso o MAG receber qualquer outro pacote com um endereço IP de origem pertencente ao prefixo PREF e que não faz parte do escopo da sua sub-rede, ele identifica que o *mobile node* está executando um *handover* e inicia o processo de sinalização, sendo chamado a partir de então para referências futuras como *serving*-MAG (S-MAG), que é o MAG que está provendo conectividade para um *mobile node* que executou um *handover*.

O S-MAG envia uma mensagem *Distributed Proxy Binding Update* (DPBU) para o A-MAG, informando-o sobre a nova localização do *mobile node* e criando um *Binding Update List* (BUL) para estabelecer um túnel entre eles.

A figura 3.3 ilustra a troca de mensagens no plano de controle e o encaminhamento do tráfego no plano de dados entre os MAGs quando um *mobile node* entra pela primeira vez no domínio e realiza dois *handovers* mantendo uma comunicação ativa com um *correspondent node* (CN). Quando o A-MAG recebe o DPBU do S-MAG, cria uma entrada BCE para o *mobile node*, respondendo com uma mensagem *Distributed Proxy Binding Acknowledgement* (DPBA) para o estabelecimento do túnel bidirecional que será utilizado para o redirecionamento de todo o tráfego do *mobile node*.

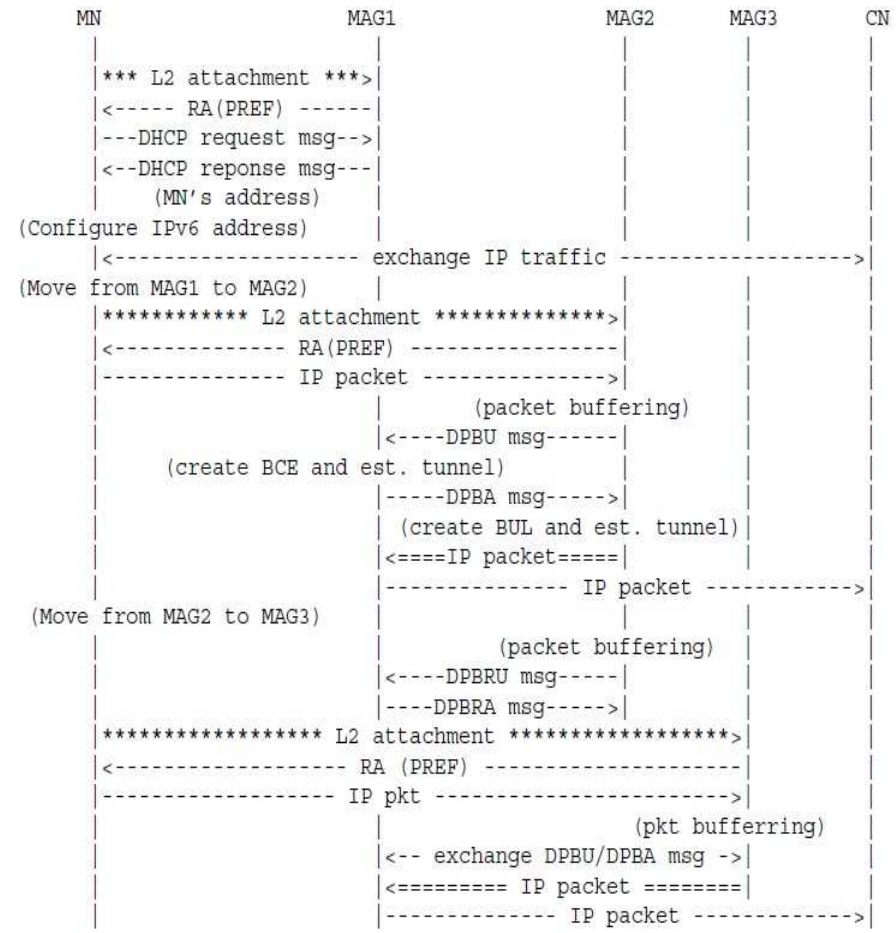


Figura 3.3 – *Sinalização e encaminhamento de tráfego entre os MAGs [35]*.

A mensagem DPBU é enviada em um pacote com o endereço IP de origem sendo o endereço no S-MAG, chamado de *proxy care-of address* (Proxy-CoA) e com o endereço IP de destino sendo o utilizado pelo *mobile node*. Este pacote é enviado através do roteamento normal da topologia, sendo encaminhado então em direção ao A-MAG. Quando o A-MAG intercepta o pacote contendo o DPBU, armazena o endereço do S-MAG na entrada BCE para o *mobile node* sinalizado, responde com uma mensagem DPBA e inicia o estabelecimento do túnel com o S-MAG.

Como o A-MAG não tem como saber quando o *mobile node* realizará novamente um *handover*, o S-MAG ao perceber que o *mobile node* foi ou será desconectado, envia uma mensagem de *Distributed Proxy Binding Release Update* (DPBRU) para o A-MAG informando sobre a desconexão do *mobile node*. O A-MAG ao receber o DPBRU passa a armazenar os pacotes destinados ao *mobile node* e responde com uma mensagem de *FLUSH* para o S-MAG, este ao receber a mensagem limpa os registros do *mobile node* da sua BUL e retransmite a mensagem *Flush* de volta para o A-MAG, com este limpando os registros atuais da BCE e aguardando até receber um novo DPBU do novo S-MAG para estabelecimento do novo túnel.

3.4 – DRAFT IEFT – BERNARDOS [23]

Esta proposta apresenta uma solução *network-based partially DMM* baseada no protocolo PMIPv6, onde uma unidade central chamada de *central mobility database* (CMD) desempenha as funções principais do plano de controle, enquanto a responsabilidade de encaminhamento de tráfego, no plano de dados, fica a cargo de unidades chamadas de *mobility anchor and access router* (MAAR).

O CMD assume a figura do LMA, mas diferente deste, ele não se envolve no encaminhamento de tráfego para o *mobile node*, ou seja, não participa das operações do plano de dados. Sendo o CMD a entidade central do plano de controle, realizando funções como o gerenciamento de sessões, registro de IP, registro de localização e *binding cache*.

Já a figura do MAG é substituída pela entidade de rede MAAR, que passa a realizar funções do LMA, tratando de aspectos do plano de controle como *local binding cache* e aspectos do plano de dados como prover a conectividade ao *mobile node*.

Cada MAAR gerencia um conjunto de prefixos IP específicos não sobrepostos que são atribuídos ao *mobile node* no momento em que este se conecta ao MAAR. A cada nova conexão do *mobile node* a um novo MAAR, um novo endereço IP sob a responsabilidade deste MAAR é alocado ao *mobile node*, mudando o conceito de endereço *home network prefix* (HNP) do PMIPv6, onde apenas um endereço IP é alocado ao *mobile node* para todo o domínio de mobilidade, para o conceito de endereço chamado de *local network prefix* (LNP).

A figura 3.4 ilustra a operação geral da proposta e como é tratado o *handover* executado pelo *mobile node*. Quando o *mobile node* se conecta pela primeira vez a um MAAR, este cria uma identificação única para o *mobile node* dentro do domínio de mobilidade (MN-ID) e reserva um endereço exclusivo LNP que será atribuído ao *mobile node*. O MAAR insere estas informações na sua tabela local *binding cache entry* (BCE) e as envia através de uma mensagem padrão *proxy binding update* (PBU) para o CMD.

Ao CMD receber a mensagem PBU, inclui as informações recebidas na sua tabela BCE, tabela com escopo global ao domínio de mobilidade, e cria uma sessão de mobilidade para o *mobile node*, respondendo ao MAAR com uma mensagem padrão *proxy binding acknowledgement* (PBA). Ao MAAR receber o PBA, envia uma mensagem RA incluindo o LNP reservado anteriormente para o *mobile node*, que passa a utilizar este endereço IP e o MAAR atual ser referenciado como *serving*-MAAR (S-MAAR), ou seja, o MAAR que está provendo conexão ao *mobile node*.

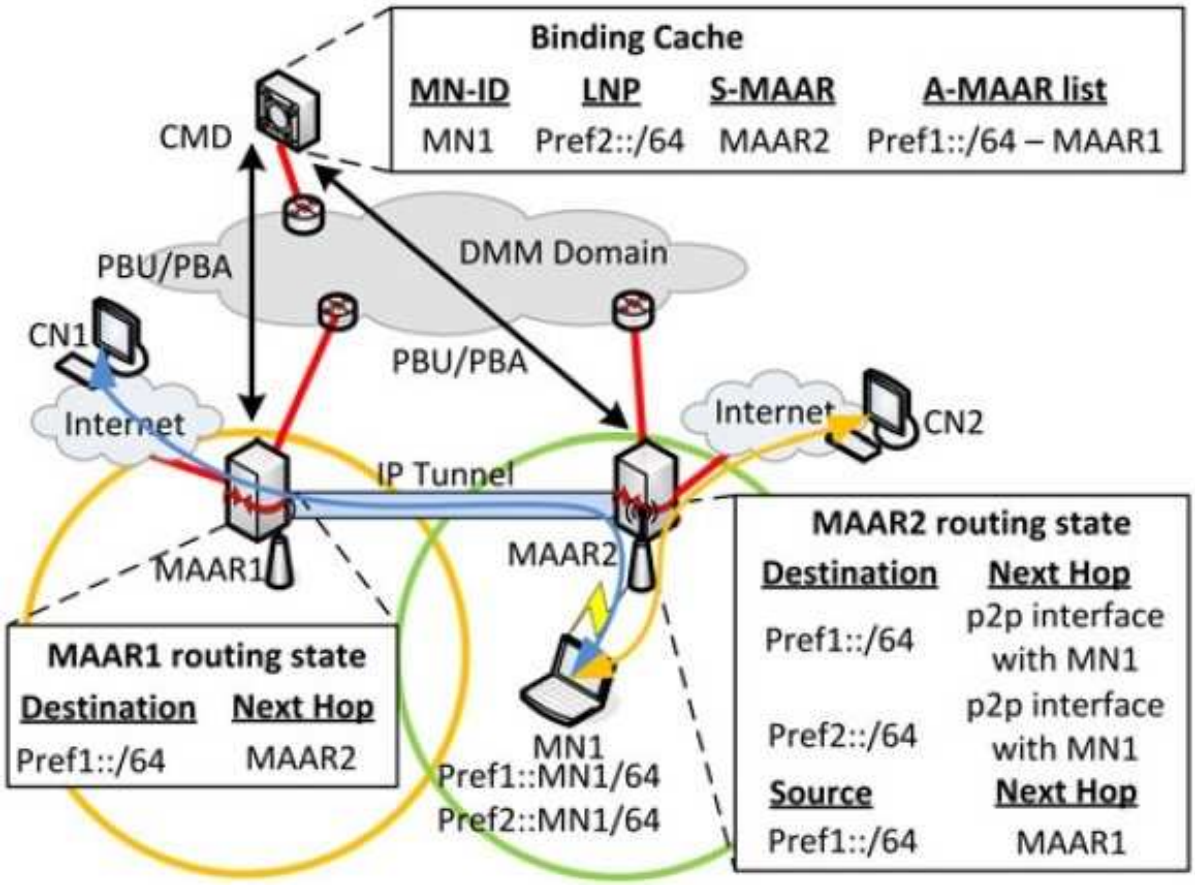


Figura 3.4 – *Draft-bernardos-dmm-pmip [16]*.

Ao *mobile node* realizar um *handover* para um novo MAAR, este realiza os mesmos procedimentos descritos anteriormente, alocando um novo NLP para o *mobile node* e enviando estas informações para o CMD. Mas ao CMD consultar os registros da BCE e verificar que o *mobile node* estava conectado anteriormente a outro MAAR, chamado de *anchor*-MAAR (A-MAAR) e utilizando um endereço NLP alocado por este, chamado de *previous*-LNP (pLNP), o CMD atualiza as entradas em sua tabela BCE e envia mensagens informando ao A-MAAR sobre a nova localização do *mobile node* e informando ao atual MAAR, o S-MAAR, sobre o pLNP sendo utilizado e o endereço do A-MAAR responsável.

Neste momento, o S-MAAR e A-MAAR estabelecem um túnel IP-in-IP para redirecionamento do tráfego relacionado ao pLNP, garantindo assim a continuidade das sessões IP que estavam em curso quando o *mobile node* executou o *handover* para o novo MAAR. A troca de mensagens entre o CMD e os MAARs pode ser observada na figura 3.4.

O *mobile node* passa a possuir dois endereços IP, o novo endereço fornecido pelo S-MAAR, o LNP e o endereço anterior fornecido pelo A-MAAR, o pLNP. Todo o tráfego relacionado ao pLNP é enviado através do túnel entre os MAARs e todo o tráfego relacionado

ao LNP é tratado diretamente pelo S-MAAR, sem utilizar técnicas de encapsulamento, onde novas sessões IP são estabelecidas utilizando este endereço.

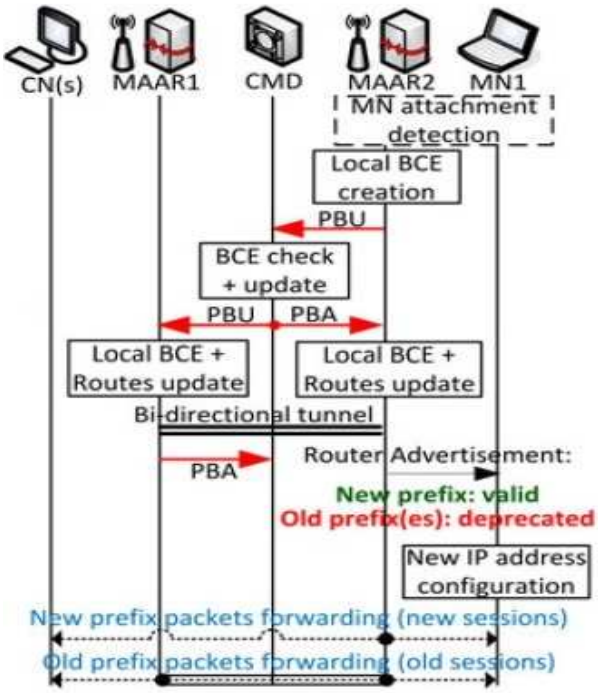


Figura 3.5 – *Troca de mensagens de controle draft-Bernardos [16]*.

3.5 – CONSIDERAÇÕES FINAIS

As abordagens para o gerenciamento de mobilidade analisadas neste capítulo representam as principais categorizações levantadas, sendo duas destas abordagens padrões do IETF e duas *drafts*.

O MIPv6 é um padrão do IETF para o gerenciamento de mobilidade centralizado baseado no móvel, onde o *host* final participa dos processos de sinalização de mobilidade no plano de controle para que um túnel seja estabelecido entre ele ou o *foreign agente* e o *home agent*. Onde este é o responsável pelo encaminhamento, através do túnel estabelecido, de todo o tráfego relacionado ao *host* em mobilidade.

No padrão IETF de gerenciamento de mobilidade centralizado baseado na rede, PMIPv6, as funções de sinalização do plano de controle não são mais executadas pelo *host* final, sendo todo o processo de gerenciamento de mobilidade transparente para o *host*. O túnel para o encaminhamento de pacotes para o *host* final, é estabelecido entre o LMA e o MAG, sendo estes os responsáveis por prover o serviço de mobilidade para o *host*.

O *draft* Jaehwoon apresenta uma proposta para realizar o gerenciamento de mobilidade baseado na rede de modo completamente distribuído, onde os MAGs são responsáveis tanto pelas funções de plano de controle, quanto pelas funções de plano de dados. Através de um conceito de *supernet* os MAGs identificam e gerenciam a mobilidade do *host* dentro do domínio, realizando o processo de estabelecendo de túneis entre si e o encaminhamento de dados para os *hosts* em mobilidade.

A proposta para o gerenciamento de mobilidade parcialmente distribuído baseado na rede apresentada no *draft* Bernardos, utiliza uma unidade central de controle chamada de CMD que assume as funções de plano de controle do LMA, atuando somente neste plano, ou seja, não se envolvendo com o encaminhamento de tráfego e unidades chamadas de MAAR que passam a desempenhar as funções do MAG, atuando no plano de controle e plano de dados. O CMD é o responsável por gerenciar a mobilidade executando ações como manter o registro da localização do *host* dentro do domínio de mobilidade o controlar o estabelecimento de túneis entre os MAARs para o encaminhamento do tráfego dos *hosts*.

4 – PROPOSTA SDN-DMM: ARQUITETURA BASEADA EM SDN PARA DMM

Neste capítulo é apresentada a proposta SDN-DMM para o gerenciamento de mobilidade distribuído baseado no paradigma SDN de modo a comportar a continuidade da sessão IP e atender principalmente os aspectos de roteamento otimizado, arquitetura distribuída, transparência para o nó móvel, escalabilidade, desempenho e baixa complexidade. Inicialmente é descrito o conceito e funcionamento da abstração da camada *northbound* SDN chamada de *intent* para prover a mobilidade IP, encerrando o capítulo com a apresentação do cenário e operação geral da proposta.

4.1 – MOBILIDADE COM O USO DE INTENT EM REDES SDN

A separação clara entre o plano de controle e o plano de dados, na visão da infraestrutura de rede como um todo, empregado pelo paradigma SDN, permite que a inteligência da rede possa ser centralizada em um único ponto, possibilitando o aumento da eficiência e flexibilidade de como as funções desempenhadas pela rede de comunicação e os novos serviços podem ser implementados.

Na proposta SDN-DMM, focada no serviço de mobilidade IP, a comunicação fim-a-fim entre os *host*s através da infraestrutura de rede é baseada em fluxos IP bidirecionais e não através do roteamento IP tradicional. Com estes fluxos sendo implementados pelo controlador SDN utilizando o protocolo *OpenFlow* na camada *southbound* da arquitetura SDN.

O *OpenFlow* é responsável por inserir, através de regras chamadas de fluxos, entradas nas tabelas de encaminhamento dos ativos de rede para determinar como a infraestrutura deve encaminhar determinado tipo de tráfego.

Um fluxo, de modo geral, é identificado pela correspondência do tráfego analisado com os parâmetros especificados na regra (como os endereços IP de origem e destino de uma comunicação) e tratado de acordo com a ação definida pela regra. O conjunto dessas regras aplicadas a um grupo de ativos determina como um fluxo IP será encaminhado fim-a-fim através da infraestrutura de rede.

Os fluxos criados com as regras *OpenFlow* são estáticos e locais, ou seja, uma vez instalados localmente na tabela de encaminhamento de determinado ativo de rede, eles permanecem na tabela independente se houver alguma alteração na infraestrutura de rede que afete o fluxo IP fim-a-fim associado à regra local. Não possibilitando que a infraestrutura ou

os fluxos locais se adaptem as condições de falhas ou mudanças topológicas, aspecto fundamental para o cenário de mobilidade IP.

Deste modo, com o objetivo de suportar a mobilidade IP utilizando uma arquitetura SDN com o protocolo *OpenFlow*, a infraestrutura de rede deve ser capaz de adaptar os fluxos IP a cada nova mudança, fornecendo um novo enlace de comunicação através da infraestrutura que permita a comunicação fim-a-fim entre os *hosts*.

Esta função pode ser desempenhada pelo controlador SDN, uma vez que ele possui a visão completa da infraestrutura de rede sob o seu controle, utilizando uma abstração da camada *northbound* chamada de *intent* em conjunto com o correto tratamento dos fluxos de acordo com a arquitetura de rede e os cenários de mobilidade.

A *intent* pode ser considerada como uma intenção de comunicação executada em alto nível que passa ao controlador informações sobre quais pontos da rede se deseja estabelecer um canal de comunicação [7].

O seu funcionamento é baseado na coleta das informações topológicas da rede pelo controlador, como a identificação dos *hosts*, dos ativos de rede e das conexões entre estes. De modo que através destas informações e da intenção de comunicação definida, o controlador cria e adpta as regras *OpenFlow* para estabelecimento dos fluxos IP de acordo com o cenário de mobilidade.

A identificação dos *host*s é realizada quando o switch envia um pacote de determinado fluxo IP que obteve correspondência com uma regra *OpenFlow* específica com a ação de encaminhamento para o controlador ou quando não houve nenhuma correspondência com os fluxos instalados na tabela de encaminhamento, utilizando então a regra local de encaminhamento do pacote para o controlador.

Neste momento o controlador analisa as informações contidas no pacote recebido juntamente com as informações passadas pelo switch, obtendo informações como o endereço MAC, endereço IP, dispositivo e porta a qual os *hosts* estão conectados, atribuindo um identificador único para cada *host*.

Desta maneira, o controlador possui uma visão completa da topologia de rede, não somente dos ativos *OpenFlow* e suas conexões, mas também da disposição e identificação dos *hosts* finais e demais dispositivos não *OpenFlow* conectados que participam da topologia.

Sem estas informações topológicas ou apenas pela criação direta de fluxos com o protocolo *OpenFlow* pela camada *southbound* para se estabelecer a comunicação entre *hosts* por meio

de um fluxo IP Bidirecional, não seria possível uma adaptação automática do plano de dados da infraestrutura de rede para comportar a mobilidade IP ou algum possível cenário de falha.

Sendo assim, é necessário que os fluxos *OpenFlow* sejam ajustados pelo controlador de acordo com as informações de topologia e com a comunicação fim-a-fim que deve ser estabelecida. Para isso, é indicada ao controlador na camada superior *northbound* a comunicação fim-a-fim que se deseja estabelecer e deve ser mantida pela infraestrutura de rede.

De posse desta indicação, o controlador verifica as informações de topologia para criar os fluxos *OpenFlow* na camada inferior *southbound* a fim de prover tal comunicação. Quando ocorrem mudanças de topologia, o controlador é informado pelos ativos de rede, verifica se as comunicações fim-a-fim indicadas nas *intents* foram afetadas e caso tenham sido, recalcula novos fluxos *OpenFlow* para comportar as mudanças e manter a comunicação.

A relação entre a comunicação indicada na camada *northbound* com a *Intent* e as regras *OpenFlow* na camada *southbound*, de acordo com as informações de topologia obtidas pelo controlador, é apresentada na figura 4.1.

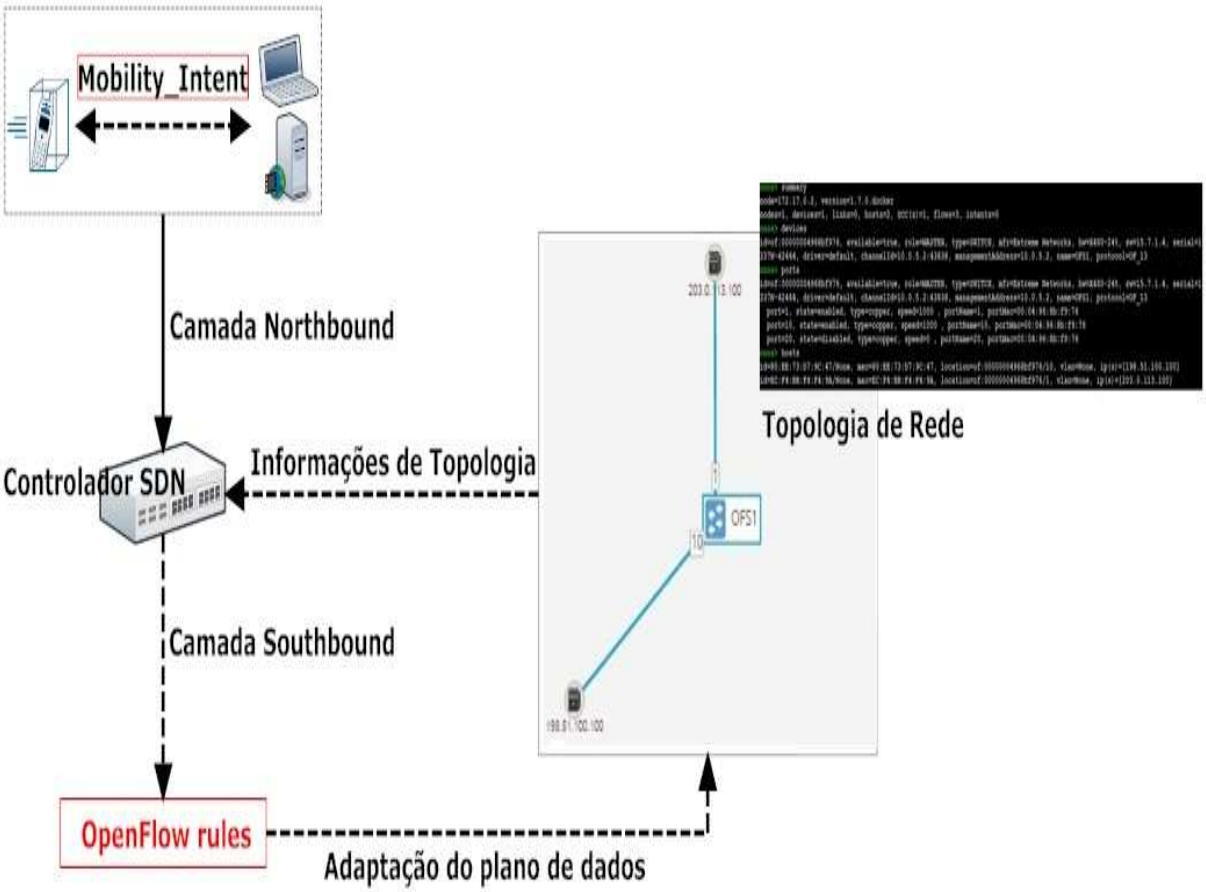


Figura 4.1 – *Relação entre a camada northbound, camada southbound e as informações de topologia.*

Podemos observar na figura 4.1 a separação clara entre a indicação da comunicação realizada na camada *northbound* através do uso da *Intent* e as regras *OpenFlow* que são criadas e reajustadas na camada *southbound* de acordo com as informações topológicas obtidas pelo controlador para adaptar automaticamente os fluxos IP bidirecionais do plano de dados para manter ativa a comunicação.

4.2 – DESCRIÇÃO DO CENÁRIO E OPERAÇÃO SDN-DMM

As soluções de mobilidade IP tradicionais, como as padronizadas pelo IETF, apoiam-se de alguma forma no roteamento tradicional de pacotes IP e em ativos específicos nas redes de comunicação para desempenhar papeis chaves nos processos de mobilidade e no tratamento da comunicação, como processos de registro e técnicas de tunelamento de pacotes.

Tais soluções acabam adicionando uma sobrecarga para o ambiente, tanto em relação aos ativos envolvidos, quanto para o transporte de dados em si, aumentando a complexidade para a operação e a manutenção da rede.

Os ativos que assumem responsabilidades dentro do contexto de mobilidade nestas soluções, passam a ter que se preocupar não apenas em desempenhar a sua função principal, por exemplo, o roteamento de pacotes em roteadores, mas também em desempenhar funções extras, como tunelamento, processos de registro e descoberta, ações de *proxy* e análise de pacotes.

A utilização de técnicas de tunelamento de dados e o roteamento sub-otimizado, baseado no endereço de destino do pacote IP, são inapropriados para o cenário de mobilidade, causando o uso ineficiente da infraestrutura de rede de comunicação, com a redução da taxa efetiva de dados transmitidos e a sobrecarga de certos enlaces de comunicação enquanto outros são subutilizados.

Esses pontos negativos são potencializados com o aumento do número de dispositivos móveis conectados à rede, por exemplo, com a Internet das Coisas (IoT) e que devido ao surgimento e implementação de novas tecnologias e paradigmas, como as redes definidas por *software* (SDN), tornam as abordagens utilizadas nas soluções citadas não adequadas para o cenário presente.

Esta seção apresenta uma proposta de solução *network-based partially* DMM baseada no paradigma SDN para lidar com os desafios apresentados até aqui. Tal proposta utiliza o

protocolo *OpenFlow* e o controlador ONOS para prover o gerenciamento de mobilidade IP em redes de acesso heterogêneas, focando na continuidade das sessões IP.

Por meio da separação entre o plano de controle e o plano de dados da rede de comunicação e do tratamento de pacotes de acordo com os fluxos IP, a proposta visa promover uma redução da complexidade geral do sistema e dos custos operacionais (OPEX) e de investimento (CAPEX), atendo principalmente aos aspectos de:

- Continuidade de sessão IP;
- Roteamento otimizado;
- Arquitetura distribuída;
- Transparência para o *mobile node*;
- Escalabilidade;
- Desempenho;
- Baixa complexidade.

A figura 4.2 apresenta uma arquitetura SDN básica de referência para o entendimento da proposta, composta por um controlador ONOS (C1), que pode ser implementado em modo cluster hierárquico distribuído para eliminar pontos únicos de falha, tratando também do aspecto de escalabilidade do plano de controle, switches *OpenFlow*, um serviço de mídia na Internet (MD), uma rede de *backbone* de transporte e redes de acesso heterogêneas.

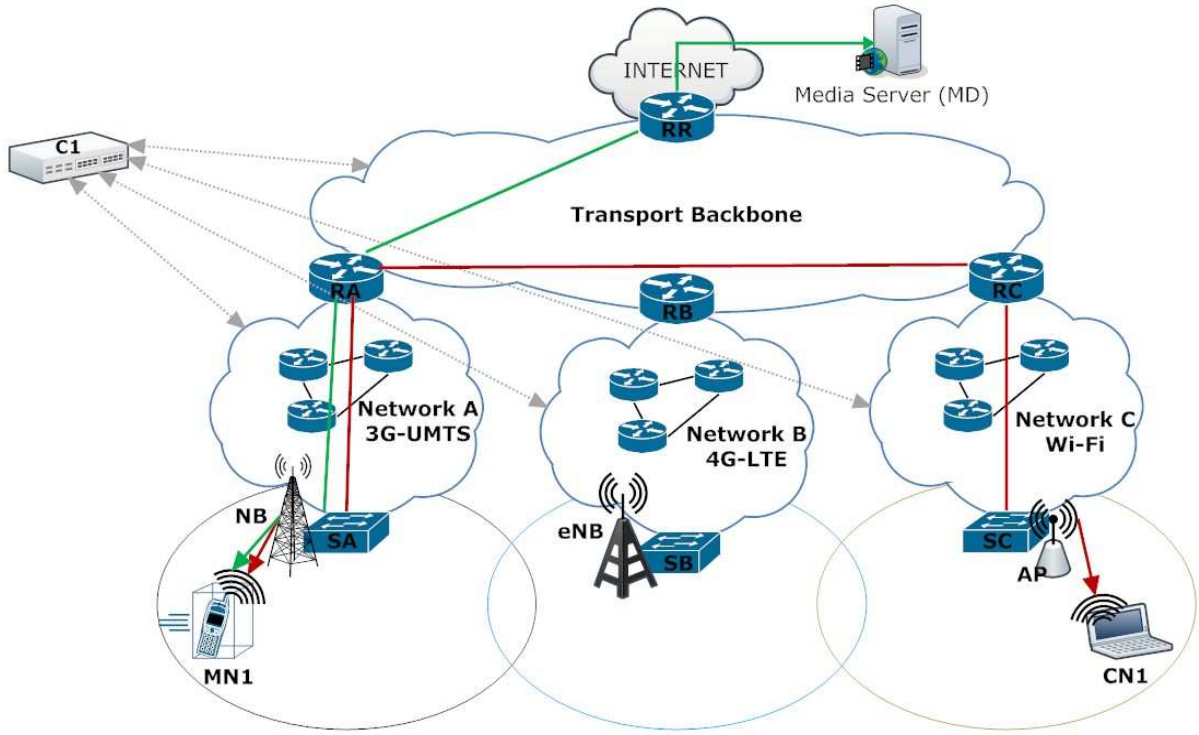


Figura 4.2 – *Arquitetura SDN-DMM.*

O cenário geral analisado envolve a mobilidade realizada pelo *mobile node* MN1, movimentando-se a partir da rede de acesso A (3G-UMTS) para a rede de acesso B (4G-LTE) e depois para a rede de acesso C (Wi-Fi) enquanto mantém suas comunicações ativas com o *Media Server* MD e com o *correspondent node* CN1.

A comunicação fim-a-fim entre os *hosts* é estabelecida na rede através de fluxos IP bidirecionais, sendo um fluxo do MN1 para o CN1 e outro do CN1 para o MN1 (representado na figura 4.2 pelo fluxo vermelho) e um fluxo do MN1 para o MD e outro do MD para o MN1 (representado na figura 4.2 pelo fluxo verde).

Estes fluxos são estabelecidos por meio de regras *OpenFlow* que inserem nas tabelas de encaminhamento dos ativos de redes envolvidos entradas correspondentes aos *links* de comunicação que devem ser utilizados para a formação do caminho que irá transportar o fluxo fim-a-fim.

As situações específicas analisadas neste processo de mobilidade são ilustradas na figura 4.3, sendo:

- $\boldsymbol{s_1}$= o MN1, o MD e o CN1 estão conectados as suas redes de origem e estabelecem a comunicação normalmente pela infraestrutura de rede, ainda não ocorreu mobilidade;
- $\boldsymbol{s_2}$= com as comunicações estabelecidas na situação anterior em andamento, o MN1 altera o seu ponto de conexão da rede A (3G-UMTS) para a rede B (4G-LTE), ocorrendo o processo de mobilidade;
- $\boldsymbol{s_3}$ = com as sessões ainda em curso, o *mobile node* MN1 realiza novo *handover*, alterando sua conexão da rede B (4G-LTE) para a rede C (WiFi).

No cenário s_1, ao switch SA receber um fluxo IP originado pelo MN1 com destino ao CN1 e ao MD, ele identifica em sua tabela de encaminhamento que existem entradas correspondentes a tais fluxos e toma as ações apropriadas.

Neste caso, encaminhando os pacotes pertencentes aos dois fluxos para o roteador RA, que por sua vez ao analisar a sua tabela de encaminhamento repassa o fluxo destinado ao MD para o roteador RR e o fluxo destinado ao CN1 para o roteador RC.

Quando o MN1, no cenário $\boldsymbol{s_2}$, realiza o processo de *handover* da *NodeB* (NB), que está inserida no contexto da rede A, para a *Evolved NodeB* (eNB), que está inserida no contexto da rede B, inicia-se o processo de mobilidade.

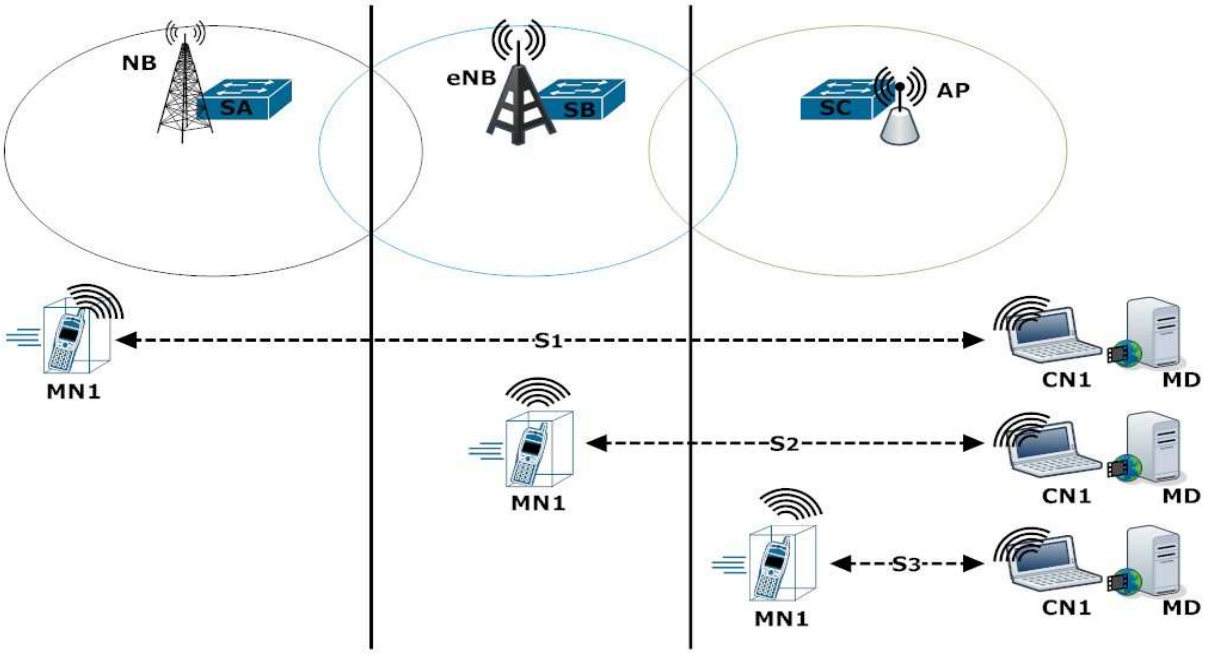


Figura 4.3 – *Cenário de conexão para as situações s_1, s_2 e s_3.*

Para manter a continuidade das sessões, o MN1 mantém o seu endereço IP quando realiza o processo de *handover* da rede A para a rede B, evitando alterações na comunicação fim-a-fim, principalmente em relação à camada 3 e superiores, como a realocação de *sockets* TCP/IP.

Mas devido à sua nova localização e a utilização de um endereço IP diferente do escopo de endereçamento de rede atribuído ao seu novo ponto de conexão, a rede B, é necessário que os pacotes destinados ao endereço IP do MN1 possam ser corretamente encaminhados pela infraestrutura de rede até a sua nova posição.

Para que isso ocorra de modo transparente para os *hosts* finais, a rede de comunicação deve se auto-adaptar, ou seja, deve fornecer um novo canal de comunicação que permita a comunicação fim-a-fim entre os *hosts* para o novo cenário, entretanto sem afetar as demais comunicações em curso na rede e o roteamento destas.

De modo a prover um uso mais eficiente da infraestrutura disponível e evitar sobrecargas, como técnicas de registro ou de tunelamento de pacote, a solução utiliza uma abstração da camada *northbound* SDN chamada de *intent*, readequando o plano de dados da rede para que a infraestrutura realize o encaminhamento dos pacotes destinados ao *mobile node* baseado no fluxo IP bidirecional e não no roteamento tradicional de pacotes pelo endereço IP de destino.

O controlador C1, de posse da intenção de comunicação definida pela *intent* entre o *mobile node* MN1 e os *hosts* MD e CN1, e do conhecimento da topologia física da rede, escolhe o melhor caminho para redefinir os canais de comunicação através do envio de regras de

controle de baixo nível *OpenFlow* pela interface *southbound* SDN para os ativos envolvidos, readequando as suas tabelas de encaminhamento para permitir os novos fluxos IP bidirecionais para o *mobile node* em sua nova localização.

A solução então é constituída com o switch S2 informando, ao identificar o *mobile node* ou um fluxo IP com endereço de origem diferente da rede B, a mudança topológica ao controlador C1.

O controlador ao constatar que ocorreu uma mudança de topologia envolvendo a localização do MN1, readequa o plano de dados da rede de comunicação, alterando as regras de *OpenFlow* para remover entradas antigas que não serão mais utilizadas e inserir novas entradas nas tabelas de encaminhamento dos ativos envolvidos na mudança, reestabelecendo um novo conjunto de enlaces para criar um novo fluxo IP bidirecional entre os *host*s.

A figura 4.4 ilustra as principais etapas realizadas no processo de gerência de mobilidade, onde:

1. O MN1 realiza o *handover* da NB para eNB, mantendo o seu IP de origem;
2. O switch SB identifica a presença do *mobile node* com um IP de rede diferente do seu escopo e informa o evento ao controlado C1;
3. O controlador C1 verifica que ocorreu uma mudança de topologia na mobilidade do MN1, então recalcula um novo caminho e envia novas regras *OpenFlow* para readequar os enlaces de comunicação utilizados;
4. Os novos fluxos IP bidirecionais são estabelecidos na rede, permitindo que os pacotes destinados ao MN1 sejam corretamente encaminhados a ele baseado no fluxo IP e não no roteamento tradicional IP, não afetando o roteamento geral e as demais comunicações em curso na rede.

Nesse cenário o tráfego entre os *hosts* é roteado baseado no novo fluxo IP bidirecional estabelecido pela alteração do plano de dados realizada pelo controlador, permitindo que os pacotes destinados ao MN1, ou seja, destinados a um endereço da rede A, possam ser entregues diretamente a ele, mesmo estando conectado a rede B.

Deste modo o encaminhamento do fluxo IP é realizado de forma otimizada, transparente e sem adicionar *overhead* para o transporte dos dados na rede ou para o processamento dos ativos envolvidos, uma vez que não ocorre o roteamento sub-otimizado e o tunelamento de pacotes.

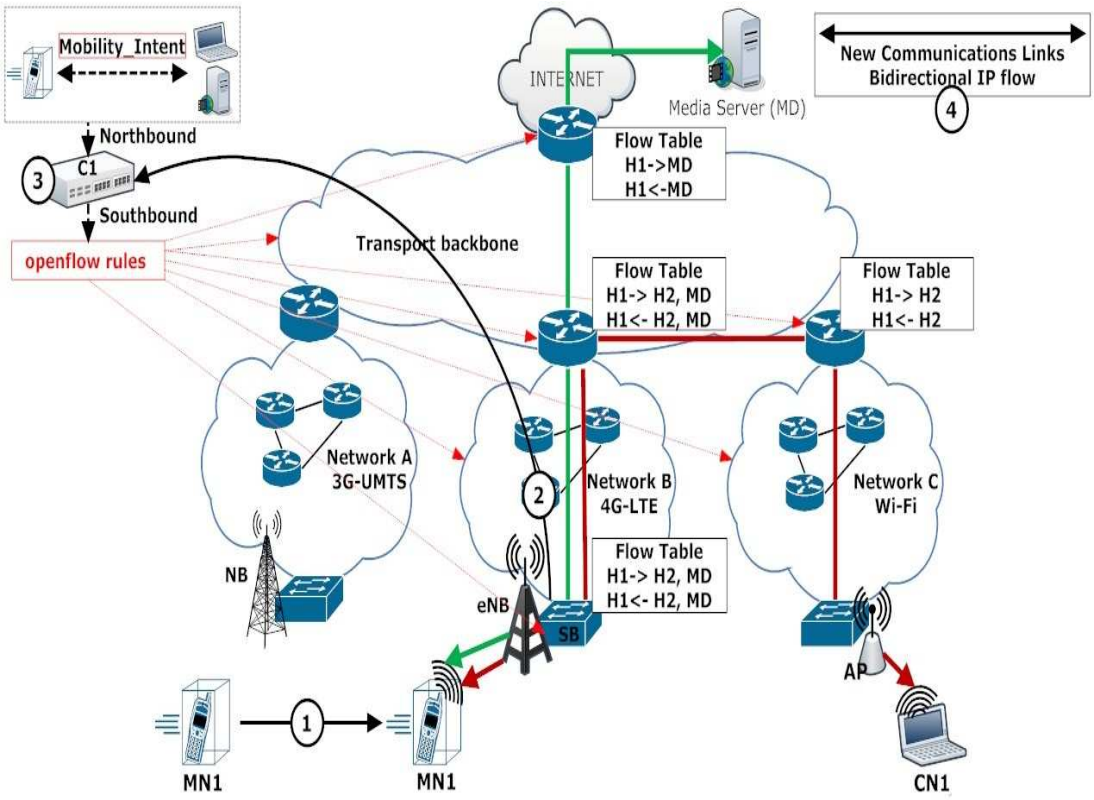


Figura 4.4 – *Handover e estabelecimento do fluxo IP bidirecional em* s_2.

Ao MN1 realizar nova mobilidade se conectando ao *access point* AP, inserido no contexto da rede C, o processo descrito anteriormente se repete. O switch SC informa ao controlador C1 da nova mudança de topologia ocorrida, que por sua vez reestabelece o fluxo IP bidirecional entre os *hosts* de modo que a comunicação entre o MN1 e o CN1 é realizada diretamente na rede C, conforme pode ser observado na figura 4.5.

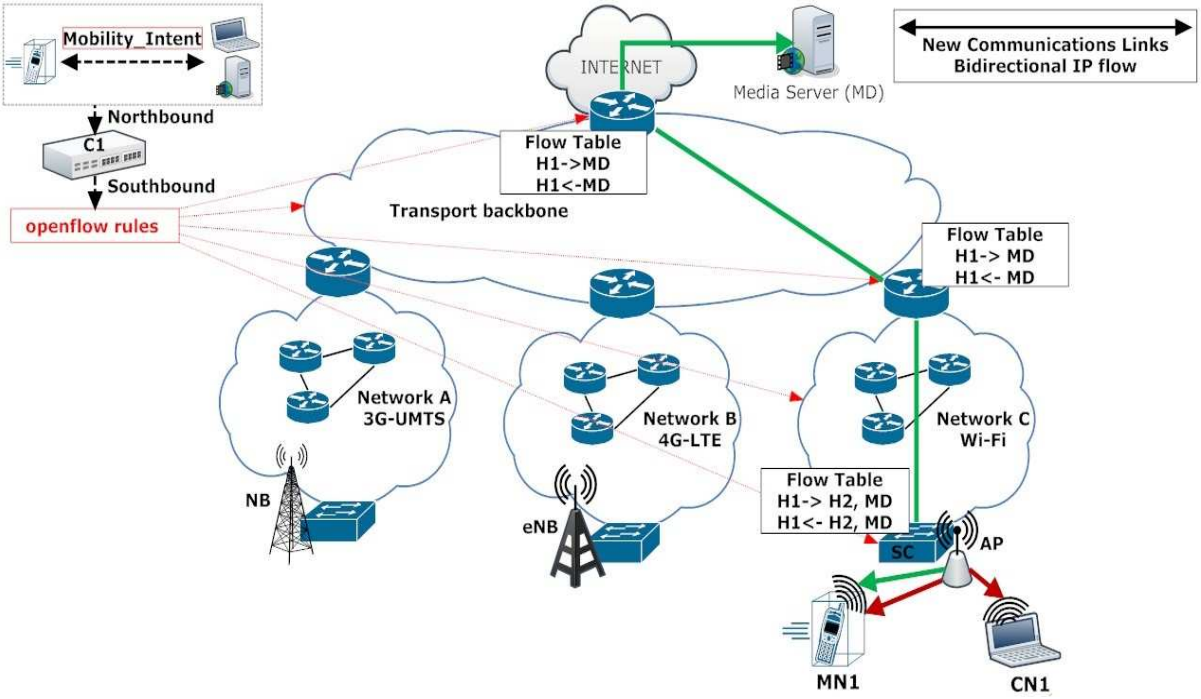


Figura 4.5 – *Handover e estabelecimento do fluxo IP bidirecional em* s_3.

4.3 – CONSIDERAÇÕES FINAIS

Utilizando o paradigma SDN, a proposta SDN-DMM apresentada, baseia-se em uma arquitetura de rede definida por *software* para prover o serviço de gerenciamento de mobilidade distribuído baseado na rede de modo otimizado e flexível, sem necessitar utilizar técnicas de tunelamento ou registros.

A partir da unidade central de controle, o controlador SDN, que atua somente no plano de controle tendo o conhecimento total da topologia de rede, é possível ajustar o plano de dados da infraestrutura de acordo com a mobilidade executada pelo *mobile node* de modo que o tráfego seja entregue diretamente a ele.

O ajuste do plano de dados para suportar a mobilidade do *mobile node* é feito de forma automática através da utilização da *intent* na camada *northbound* SDN, criando uma separação clara entre a intenção de comunicação e as regras *OpenFlow* na camada *southbound* criadas para adaptar os fluxos IP bidirecionais de acordo com as mudanças topológicas.

5 – AVALIAÇÃO DE DESEMPENHO DAS PROPOSTAS

Neste capítulo, é realizada a avaliação de desempenho das propostas por meio de modelagem analítica. São utilizadas as métricas de custo de sinalização [16], custo de entrega de pacote [16], custo de latência de *handover* [16] e custo de roteamento proposto, considerando as propostas *draft*-Jaehwoon [22], *draft*-Bernardos [23] e a proposta SDN-DMM, sendo apresentada ao final uma tabela comparativa em relação aos aspectos de roteamento, desempenho, requisitos para os *hosts* finais, arquitetura/complexidade e escalabilidade, assim como os resultados obtidos nessa avaliação.

5.1 – CUSTO DE SINALIZAÇÃO

O custo de sinalização está relacionado com a quantidade de mensagens de controle geradas especificamente para suportar o processo de mobilidade, com o objetivo principal de manter a continuidade das sessões do *mobile node*.

Pode ser entendido como um *overhead* de pacotes que os ativos de rede precisam processar e que a infraestrutura de rede de comunicação precisa transmitir em prol da mobilidade. Os três principais componentes do custo de sinalização são [16]:

i. C_{update}= Custo de atualização da localização do MN com mensagens de binding update;

ii. $C_{refresh}$ = Custo com atualizações periódicas;

iii. C_{de-reg}= Custo de desregistro devido a uma nova mobilidade executada dentro do domínio.

Desconsiderando o processo inicial de registro do *mobile node* quando este entra pela primeira vez no domínio de mobilidade e o processo final de desregistro, quando este deixa o domínio, uma vez que nestes momentos não é executado em si o processo de mobilidade, o custo de sinalização total C_{SIG}, apresentado em [16], é dado por:

$$C_{SIG} = \mu_{SN}\left(C_{update} + C_{refresh} + C_{de-reg}\right) \quad (1)$$

Em que μ_{SN} representa a taxa de cruzamento de *subnet*s executada pelo *mobile node*.

O custo individual de cada um dos três componentes (C_{update}, $C_{refresh}$ e C_{de-reg}) é formado pela soma do custo C_{X-Y}, que é o custo simétrico para transmitir um pacote de

controle entre duas unidades de rede X e Y, com o custo PC_X, que é o custo envolvido no processamento deste pacote de controle pela unidade de rede X [16].

A seguir são apresentados os custos de sinalização para cada uma das propostas avaliadas.

5.1.1 – Draft IETF - Jaehwoon

Nesta proposta, quando um *mobile node* executa um *handover*, o novo S-MAG envia uma mensagem DPBU para o A-MAG e este responde com uma mensagem DPBA. Essas mensagens são trocadas para informar a nova localização do *mobile node* e estabelecer o túnel para o redirecionamento do tráfego, assim o custo C_{update} é definido por:

$$C_{update}^{\text{Draft}-\text{Jaehwoon}} = 2C_{SMAG-AMAG} + PC_{AMAG} + PC_{SMAG} \quad (2)$$

Considerando que as atualizações periódicas são realizadas da mesma forma que no PMIPv6, as mesmas mensagens descritas acima são trocadas entre os MAGs a uma taxa igual a $R_{BCE} = 1/(\mu_{SN} T_{BCE})$, onde T_{BCE} é o tempo de vida de uma entrada BCE [16]. Assim $C_{refresh}$ é dado por:

$$C_{refresh}^{\text{Draft}-\text{Jaehwoon}} = R_{BCE}(2C_{SMAG-AMAG} + PC_{AMAG} + PC_{SMAG}) \quad (3)$$

Em relação ao custo C_{de-reg}, quando o *mobile node* se move para um novo MAG, mensagens de DPBRU/DPBRA e *Flush* são trocadas para que os registros antigos sejam removidos das tabelas. Assim o custo de desregistro devido à nova localização do *mobile node* é definido por:

$$C_{de-reg}^{\text{Draft}-\text{Jaehwoon}} = 4C_{SMAG-AMAG} + 2PC_{AMAG} + 2PC_{SMAG} \quad (4)$$

5.1.2 – Draft IETF - Bernardos

Quando o *mobile node* executa um *handover*, o S-MAAR envia uma mensagem PBU para o CMD para realizar o registro do novo LNP alocado ao *mobile node*, atualizando então a localização deste na entrada BCE relacionada no CMD.

Ao CMD verificar que o *mobile node* estava anteriormente conectado a outro MAAR (A-MAAR), envia uma mensagem *extended* PBA, informando o S-MAAR sobre os demais

prefixos ativos no *mobile node* (pLNP) e o endereço dos A-MAARs correspondentes; uma mensagem *extended* PBU também é enviada pelo CMD para os A-MAARs envolvidos atualizarem a nova localização do *mobile node* e estabelecer um novo túnel com o S-MAAR.

Assim o custo C_{update} depende da quantidade de prefixos ativos no *mobile node*, N_{PR}, no momento do *handover*, sendo expresso por [16]:

$$C_{update}^{\text{Draft}-\text{Bernardos}} = (N_{PR} + 1)(2C_{CMD-MAAR} + PC_{CMD} + PC_{MAAR}) \quad (5)$$

Similar ao processo de *refresh* que ocorre no PMIPv6, mensagens de atualizações periódicas PBU/PBA são trocadas apenas entre o S-MAAR e o CMD, para manter ativos e atualizados os registros, que considerando uma taxa R_{BCE}, o custo $C_{refresh}$ é dado por [16]:

$$C_{refresh}^{\text{Draft}-\text{Bernardos}} = R_{BCE}(2C_{CMD-MAAR} + PC_{CMD} + PC_{MAAR}) \quad (6)$$

Por fim, o processo de desregistro é executado quando um A-MAAR identifica que o pLNP não está sendo mais utilizado, trocando mensagens com o CMD e este com o S-MAAR, para que os registros sejam removidos das entradas. O custo C_{de-reg} é dado por [16]:

$$C_{de-reg}^{\text{Draft}-\text{Bernardos}} = 4C_{CMD-MAAR} + 2PC_{CMD} + 2PC_{MAAR} \quad (7)$$

5.1.3 – SDN-DMM

Na proposta SDN-DMM, o switch *OpenFlow* envia uma mensagem para o controlador ao identificar que o *mobile node* executou um *handover*. O controlador verifica a mudança ocorrida na topologia, recalcula um novo caminho e envia mensagens *OpenFlow* para que os switches envolvidos estabeleçam um novo enlace de comunicação através de fluxos IP bidirecionais.

Considerando um cenário de simetria entre a movimentação do *mobile node* no *draft*-Bernardos, que resulta na alocação de novos endereços LNP e os switches afetados no reajuste do plano de dados, com a quantidade de switches afetados sendo igual a quantidade de endereços LNP alocados, exceto o switch que identificou a mobilidade. A quantidade de switches afetados é representada por N_{PR}, com o custo C_{update} representado por:

$$C_{update}^{\text{SDN}-\text{DMM}} = 2C_{CTR-OFS} + PC_{CTR} + PC_{OFS} + (N_{PR})(C_{CTR-OFS} + PC_{OFS}) \quad (8)$$

Onde CTR significa o custo relacionado ao controlador e OFS o custo relacionado ao switch *OpenFlow*.

Como o switch que recebeu uma mensagem do controlador para ajustar o plano de dados só responde ao controlador caso ocorra algum problema para o estabelecimento do fluxo IP solicitado, não é adicionado à segunda parcela do custo C_{update} o processamento do controlador e o custo para trasmissão desta mensagem.

Em relação às atualizações periódicas e aos processos de desregistro, a abordagem SDN-DMM não trabalha com o conceito de registro e *binding cache*, que são utilizados para armazenar informações de localização com o objetivo de se estabelecer túneis para o redirecionamento de pacotes quando o *mobile node* realiza um *handover*. O controlador através do conhecimento da topologia física, apenas ajusta o plano de dados da rede sob demanda de acordo com a movimentação do *mobile node*.

As entradas nas tabelas de encaminhamento para os fluxos IP oriundos do *handover* do *mobile node* nos switches *OpenFlow*, analogamente as entradas de registro da BCE, são inseridas/removidas de duas formas:

1) Durante o processo de atualização, onde o controlador envia mensagens para os switches envolvidos para suportar a mobilidade, inserindo ou removendo entradas;
2) Pela expiração das entradas nos switches que não estão mais encaminhando tráfego relacionado ao fluxo IP bidirecional estabelecido.

Deste modo o custo de sinalização $C_{SIG}^{\mathrm{SDN-DMM}} = C_{update}^{\mathrm{SDN-DMM}}$.

5.2 – CUSTO DE ENTREGA DE PACOTE

O custo de entrega de pacote na perspectiva do gerenciamento de mobilidade pode ser entendido como o custo para que os pacotes sejam entregues entre o *mobile node* (MN) e o correspondent node (CN) dentro do domínio de mobilidade. De modo geral, o custo total de entrega de pacote é formado pela soma de três custos de acordo com os três trechos principais na rota de entrega do pacote do CN até o MN [16]:

i. C_{CN-NMA}= Custo de entrega do pacote entre o CN e o native mobility anchor (NMA) responsável pelo encaminhamento inicial dos pacotes destinados ao MN dentro do domínio de mobilidade;

ii. $C_{NMA-SMA}$= Custo de entrega do pacote entre o NMA e o serving MA (SMA) que está servindo atualmente o MN;

iii. C_{SMA-MN}= Custo de entrega do pacote entre o SMA e o MN através de um meio sem fio.

Considera-se que o custo para entregar um pacote sem encapsulamento tem um valor unitário, o custo para entregar o mesmo pacote através de tunelamento sofre uma penalização $\tau > 1$ e o custo para entregar este pacote através de um enlace sem fio sofre uma penalização ω. Para uma taxa de transmissão de envio e recepção de pacotes λ, considerando o uso de encapsulamento entre o NMA e o SMA e com a entrega dos pacotes deste para o MN sendo realizada por meio de um enlace sem fio, o custo total de entrega de pacotes C_{PD}, apresentado em [16], é dado por:

$$C_{PD} = \lambda(C_{CN-NMA} + \tau C_{NMA-SMA} + \omega C_{SMA-MN}) \quad (9)$$

Onde ao adicionarmos o impacto causado pela distância, *D*, entre os elementos que realizam o encapsulamento dos pacotes destinados ao *mobile node*, afetando o custo $C_{NMA-SMA}$, temos a nova expressão para o custo de entrega de pacote C_{EP}:

$$C_{EP} = \lambda(C_{CN-NMA} + \tau D C_{NMA-SMA} + \omega C_{SMA-MN}) \quad (10)$$

A seguir são apresentados os custos de entrega de pacote para cada uma das propostas avaliadas.

5.2.1 – Draft IETF - Jaehwoon

Nesta proposta apenas um túnel é estabelecido entre o A-MAG e o S-MAG. Então de acordo com a expressão (10), o C_{EP} para esta proposta, com o NMA e SMA sendo representados respectivamente por AMAG e o SMA, é dado por:

$$C_{EP}^{\text{Draft-Jaehwoon}} = \lambda(C_{CN-AMAG} + \tau D C_{AMAG-SMAG} + \omega C_{SMAG-MN}) \quad (11)$$

5.2.2 – Draft IETF - Bernardos

Nesta proposta o *mobile node* sempre recebe um novo endereçamento quando realiza o *handover* entre os MAARs, então para cada prefixo ativo em mobilidade é adicionado um novo custo de entrega de pacotes na medida em que novos túneis são estabelecidos em pares entre os A-MAARs e S-MAAR. Deste modo, o custo de entrega de pacotes de acordo com a expressão (10) é dado por:

$$C_{EP}^{\text{Draft}-\text{Bernardos}} = \lambda(C_{CN-AMAAR} + N_{PR}\tau DC_{AMAAR-SMAAR} + \omega C_{SMAAR-MN}) \quad (12)$$

Onde N_{PR} é o número de prefixos ativos em mobilidade; com o AMAAR sendo o NMA e o SMAAR o SMA.

5.2.3 – SDN-DMM

Na proposta SDN-DMM não existe a penalização do custo de entrega de pacotes com o uso de técnicas de encapsulamento entre as entidades de rede dentro do domínio de mobilidade, uma vez que o tráfego é encaminhado por meio de fluxos IP bidirecionais de forma otimizada. Como o tráfego originado pelo CN com destino o MN ao entrar no domínio de mobilidade é entregue diretamente para a entidade de rede que está atualmente servindo o *mobile node*, não existe o custo $C_{NMA-SMA}$. O C_{EP} para a solução SDN-DMM, com os MAs sendo os switches *OpenFlow* (OFS), é dado por:

$$C_{EP}^{SDN-DMM} = \lambda(C_{CN-OFS} + \omega C_{OFS-MN}) \quad (13)$$

5.3 – CUSTO DE LATÊNCIA DE HANDOVER

Este custo é dado pelo tempo desde a desconexão da rede antiga, até a finalização do procedimento de mobilidade que permite ao *mobile node* obter novamente conectividade global das sessões IP que estavam em curso.

A operação de *handover* de modo geral pode ser dividida em três etapas de acordo com [16]:

i. t_{L2} = intervalo de tempo entre a desconexão do enlace anterior até a conexão ao novo enlace;

ii. t_{auth} = intervalo de tempo gasto com os processos de autenticação e autorização devido à nova associação;

iii. $t_{binding}$ = intervalo de tempo relacionado com a fase de mobilidade. Fase executada de acordo com o protocolo responsável, onde são desempenhadas funções como bindings, ajuste de roteamento e estabelecimento de túneis.

Como t_{L2} e t_{auth} são independentes do protocolo de mobilidade utilizado, para uma análise focada no custo de latência de *handover*, $C_{handover}$, gerado de fato por cada protocolo, consideramos apenas o componente $t_{binding}$, sendo assim temos:

$$C_{handover} = t_{binding} \tag{14}$$

O $t_{binding}$, apresentado na equação (15), depende da operação e da abordagem para a mobilidade em cada proposta, mas de modo geral pode ser entendido como a soma dos tempos necessários para transmitir e processar as mensagens de controle.

$$t_{binding} = \sum RTT_{X-Y} + T_{proc}^{X} \tag{15}$$

Onde RTT_{X-Y} é o tempo para a troca de pacotes entre as entidades de rede X e Y; e T_{proc}^{X} é o tempo de processamento de um pacote pela entidade X.

A seguir são apresentados os custos de latência de *handover* para cada uma das propostas avaliadas.

5.3.1 – Draft IETF - Jaehwoon

O processo $t_{binding}$ inicia-se com o S-MAG anunciando o prefixo de rede PREF através de uma mensagem RA para o *mobile node* recém conectado. Este, ao visualizar o prefixo PREF na mensagem RA, considera que ainda está conectado à mesma rede, mantém o seu endereço de IP e transmite normalmente seus pacotes.

Ao S-MAG receber tais pacotes, verifica que o *mobile node* executou um processo de *handover*, então envia uma mensagem DPBU para que o A-MAG seja devidamente informado sobre a nova localização do *mobile node*. O A-MAG responde ao S-MAG com uma mensagem DPBA e estabelece o túnel entre eles para o encaminhamento dos pacotes do *mobile node*. Neste momento o *mobile node* retoma efetivamente a conectividade global, podendo enviar e receber os pacotes através do túnel estabelecido entre os MAGs.

Sendo assim, o custo de latência de *handover* é dado por:

$$C_{handover}^{Draft-Jaehwoon} = RTT_{MN-MAG} + RTT_{MAG-MAG} + T_{proc}^{MN} + 3T_{proc}^{MAG} \tag{16}$$

Onde RTT_{MN-MAG} é o tempo de envio da mensagem RA do MAG para o MN e do retorno do tráfego IP do MN para o MAG; $RTT_{MAG-MAG}$ é o tempo para a troca de mensagens

DPBU/DPBA entre os MAGs; T_{proc}^{MN} é o tempo de processamento da mensagem RA pelo MN; e T_{proc}^{MAG} é o tempo de processamento do MAG, sendo executado uma vez quando o S-MAG recebe o tráfego e duas vezes na comunicação entre os MAGs.

5.3.2 – Draft IETF - Bernardos

Nesta proposta, a fase de mobilidade é iniciada quando o *mobile node* se conecta ao novo S-MAAR solicitando um novo endereço IP. O S-MAAR ao realizar o registro deste IP com o CMD, é informado sobre a existência de outros prefixos que ainda estão sendo utilizados pelo *mobile node* e que possuem sessões ativas, devendo ser tratados pelo processo de mobilidade a fim de manter a continuidade destas sessões IP.

O S-MAAR então estabelece em conjunto com cada A-MAAR o túnel para encaminhamento do tráfego correspondente a estes prefixos.

Sendo assim, o custo de latência de *handover* é composto pelo envio da mensagem RS do *mobile node* para o S-MAAR, com este enviando uma mensagem PBU para o CMD, que responde ao S-MAAR com uma mensagem extended PBA e envia para os demais A-MAAR uma mensagem extended PBU e por fim a troca de mensagens entre os MAARs para o estabelecimento do túnel.

Considerando que todos os MAAR recebem e processam simultaneamente as mensagens extended PBA/PBU enviadas pelo CMD, o custo de latência de *handover* é dado por:

$$C_{handover}^{Draft-Bernardos} = \frac{1}{2}RTT_{MN-MAAR} + RTT_{CMD-MAAR} + RTT_{MAAR-MAAR} + 3T_{proc}^{MAAR} + T_{proc}^{CMD} \quad (17)$$

Com ½ $RTT_{MN-MAAR}$ sendo o tempo da mensagem RS do *mobile node* para o MAAR; $RTT_{CMD-MAAR}$ é o tempo da troca de mensagens PBU/PBA entre o CMD e o MAAR; $RTT_{MAAR-MAAR}$ é o tempo entre a troca de mensagens entre o S-MAAR e o A-MAAR para estabelecer o túnel; T_{proc}^{MAAR} é o tempo de processamento do MAAR, que ocorre ao receber a mensagem RS do *mobile node*, ao receber o extended PBA do CMD e ao receber a mensagem de estabelecimento de túnel; e T_{proc}^{CMD} é o tempo de processamento do CMD ao receber a mensagem PBU.

5.3.3 – SDN-DMM

Logo após o *mobile node* se conectar ao novo enlace mantendo o seu endereço IP de origem, ele passa a transmitir os pacotes normalmente. Quando o switch open flow identifica que o *mobile node* realizou um *handover*, envia uma mensagem OF para o controlador informando-o sobre o evento e aguarda as instruções.

O controlador calcula um novo caminho por onde será estabelecido o novo fluxo IP bidirecional para encaminhar corretamente os pacotes do *mobile node* devido a sua nova localização, enviando para os switches afetados uma mensagem OF para estes insiram na tabela de encaminhamento o fluxo IP correspondente.

Neste momento os pacotes relacionados com o *mobile node* são encaminhados corretamente pela infraestrutura de rede de acordo com o fluxo IP bidirecional estabelecido, fornecendo novamente conectividade global com continuidade das sessões IP.

Considerando que todos os switches *OpenFlow* recebem e processam simultaneamente as mensagens enviadas pelo controlador, a equação (18) apresenta o custo de latência de *handover* para a abordagem SDN-DMM.

$$C_{handover}^{SDN-DMM} = \frac{1}{2} RTT_{MN-OFS} + RTT_{CTR-OFS} + 2T_{proc}^{OFS} + T_{proc}^{CTR} \qquad (18)$$

Onde $\frac{1}{2}RTT_{MN-OFS}$ é o tempo de envio do pacote do *mobile node* para o switch *OpenFlow*; $RTT_{CTR-OFS}$ é o tempo da troca de pacotes entre o controlador e o switch *OpenFlow*; T_{proc}^{OFS} é o tempo de processamento do switch *OpenFlow*, executado quando recebe o pacote do *mobile node* e quando recebe a mensagem *OpenFlow* do controlador; e T_{proc}^{CTR} é o tempo de processamento do controlador.

5.4 – CUSTO DE ROTEAMENTO

O roteamento de pacotes possui grande impacto nos cenários de mobilidade, tanto da perspectiva do usuário final, quanto da própria infraestrutura de rede de comunicação.

Um roteamento otimizado permite utilizar de maneira eficiente os enlaces de comunicação disponíveis e reduzir a latência da comunicação final, evitando assim a formação de gargalos para o encaminhamento de tráfego e melhorando a qualidade de experiência (do inglês, *Quality of Experience* - *QoE*) do usuário.

Definimos a métrica custo de roteamento com o objetivo de analisar o impacto causado pelo roteamento empregado nos cenários de mobilidade de acordo com a abordagem DMM utilizada em cada proposta. O custo de roteamento, $C_{routing}$, é definido na equação (19) que para simplificação considera a direção *downstream* da comunicação, ou seja, do *correspondent node* localizado fora do domínido de mobilidade, para o *mobile node* dentro do domínio.

$$C_{routing} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{H^{m}_{MN-CN} + L^{m}_{I} + 2L^{m}_{E}}{H^{l}_{MN-CN} + L^{l}_{I} + 2L^{l}_{E}} \quad (19)$$

Onde $C_{m-routing}$ é o custo de roteamento calculado de acordo com a entrega de um pacote do *correspondent node* para o *mobile node* e $C_{l-routing}$é o custo de roteamento calculado de acordo como a entrega deste mesmo pacote, mas para um *host* local à rede na qual se encontra o *mobile node*. Considera-se também que:

i. $H_{MN-CN} =$ Número de saltos que o pacote executa dentro do domínio de mobilidade até alcançar o *mobile node*/*host* local;

ii. $L_I =$ Número de vezes que os enlaces internos são utilizados para o encaminhamento dos pacotes destinados ao *mobile node*/*host* local. Enlaces internos são os enlaces que conectam os roteadores de modo horizontal, roteadores localizados no mesmo nível/camada em relação ao núcleo da rede;

iii. $L_E =$ Número de vezes que os enlaces externos são utilizados para o encaminhamento dos pacotes destinados ao *mobile node*/*host* local. Enlaces externos são os enlaces que conectam roteadores de modo vertical, roteadores localizados em diferentes níveis/camadas em relação ao núcleo da rede. São os chamados *link*s de comunicação *upstream*/*downstream*.

O custo de roteamento tem o objetivo de expressar, para determinada abordagem DMM, o aumento deste custo em relação a um roteamento otimizado de acordo com a localização topológica do *mobile node*.

Quando obtemos um valor igual a 1 (um) para o custo de roteamento $C_{routing}$, significa que a solução DMM não provoca o aumento do custo de roteamento, fornecendo então um roteamento otimizado. Isto é, os pacotes roteados para o *mobile node*, possuem o mesmo custo que os pacotes roteados para os *hosts* locais.

O peso 2 (dois) atribuído para o componente L_E na equação (19) contabiliza o uso de recursos mais caros de infraestrutura de rede e a maior latência inserida. Os enlaces de *upstream* são geralmente mais caros e percorrem uma maior distância devido ao objetivo de agregação de tráfego entre uma camada de rede inferior e uma camada de rede superior.

Além disso, no projeto de redes hierárquicas, os enlaces de *upstream* conectam equipamentos de um custo e desempenho inferior, como roteadores de acesso, a equipamentos de rede de um custo e desempenho superior, como roteadores agregadores metropolitanos [14, 52 e 53]. Portanto o peso atribuído ao componente L_E leva em consideração o custo e a latência envolvido no uso de enlaces *upstream*.

Deste modo o componente L_E possui um maior impacto na determinação do custo de roteamento do que o componente L_I, que representa os enlaces que conectam ativos de rede de uma mesma camada de rede, com o objetivo de resolução de tráfego local, utilizando enlaces e ativos de rede que possuem uma menor capacidade e custo.

Possuindo também um maior impacto do que o componente H_{MN-CN}, que representa o número geral de ativos que o pacote necessita percorrer, onde a latência inserida pelo ativo pode ser considerada a mesma, independente da camada de rede em que ele se encontra.

Pelo roteamento depender da topologia de rede, o custo de roteamento de determinada abordagem DMM também depende da topologia de rede utilizada. Assim, será considerada uma topologia em árvore simples, apresentada na figura 5.1, como referência para a análise do custo de roteamento das propostas apresentadas.

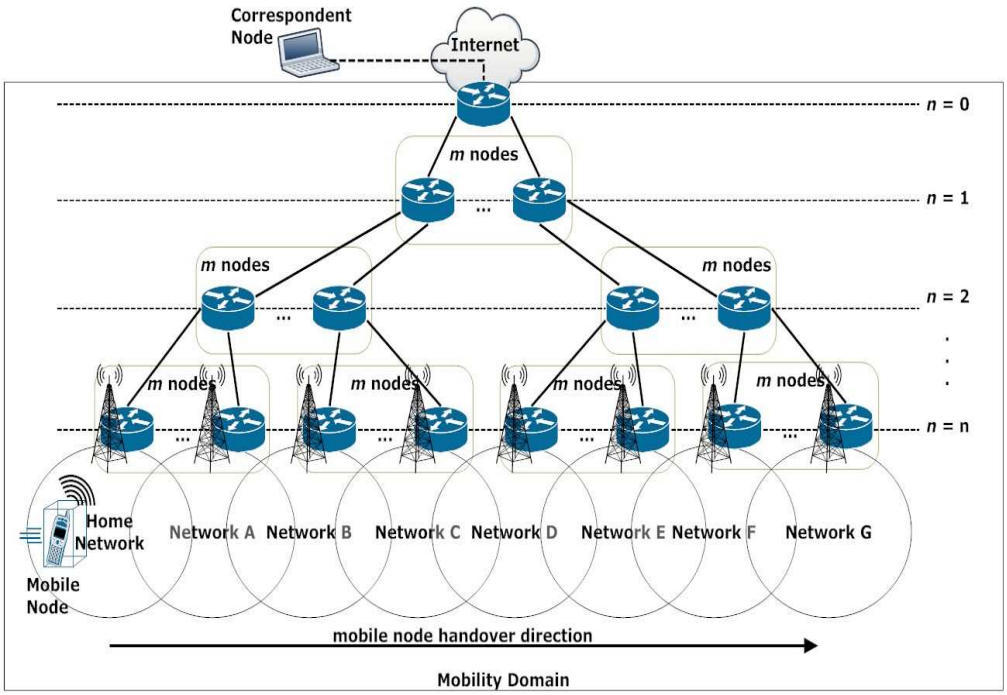


Figura 5.1 – *Topologia para custo de roteamento.*

Outro fator que impacta no custo de roteamento é a movimentação do *mobile node* pela infraestrutura de rede. Para simplificar a determinação de uma expressão que represente o custo de roteamento médio das propostas, será considerado que o *mobile node* executa uma movimentação linear entre os pontos de acesso adjacentes que pertencem ao mesmo nível de rede, calculando então a média dos custos de roteamento de cada *handover*.

A partir destas considerações, fixando a *home network* do *mobile node* em uma das extremidades da rede e com o tráfego do *correspondent node* entrando no domínio de mobilidade por um ponto central, é analisado o custo de roteamento para as propostas apresentadas de acordo com a topologia em árvore e movimentação do *mobile node* da figura 5.1, onde n representa o número de níveis verticais da rede, a profundidade, e m o número de nós de um nível inferior que são conectados a um nó de nível superior, começando a partir do nível mais superior, o nível raiz ($n = 0$) com um único nó ($m = 1$).

5.4.1 – Draft IETF – Jaehwoon e Bernardos

Como ambos os *draft*s utilizam técnicas de tunelamento para o encaminhamento do tráfego do *mobile node*, onde o tráfego precisa ser enviado para o *mobility anchor*, localizado na *home network* e responsável pelo tratamento dos pacotes destinados e originados pelo *mobile node*, estas propostas apresentam o mesmo custo de roteamento.

O roteamento é realizado com os pacotes originados ou destinados pelo *mobile node* sendo encaminhados primeiramente para o *mobility anchor* responsável de acordo com o roteamento empregado na rede. Depois estes pacotes são encaminhados através de um túnel para o *mobile node* na direção *downstream* e através da infraestrutura para o *correspondent node* na direção *upstream*, ambos se baseando novamente no roteamento empregado na rede.

Para uma análise inicial do custo de roteamento destas propostas, consideramos uma topologia em árvore simples, de acordo com a figura 5.1, calculando o custo de roteamento para $m = 2$ e $n = 1, 2\ e\ 3$, conforme apresentado na figura 5.2.

Neste cenário, para $n = 1$, quando o *mobile node* realiza o *handover* da sua *home network* para o ponto de acesso adjacente, localizado em outra rede, temos que um pacote enviado pelo *correspondent node* ao *mobile node* realiza 4 saltos e utiliza 3 vezes *links* de *upstream*.

O pacote destinado ao *mobile node* ao entrar no domínio de mobilidade, passa pelo roteador conectado a Internet, que através do roteamento tradicional da rede envia o pacote para a *home network* do *mobile node*, para o roteador responsável pelo tratamento dos pacotes

destinados a ele, que por fim encaminha o pacote através do túnel estabelecido com o roteador localizado na rede a qual *mobile node* está conectado.

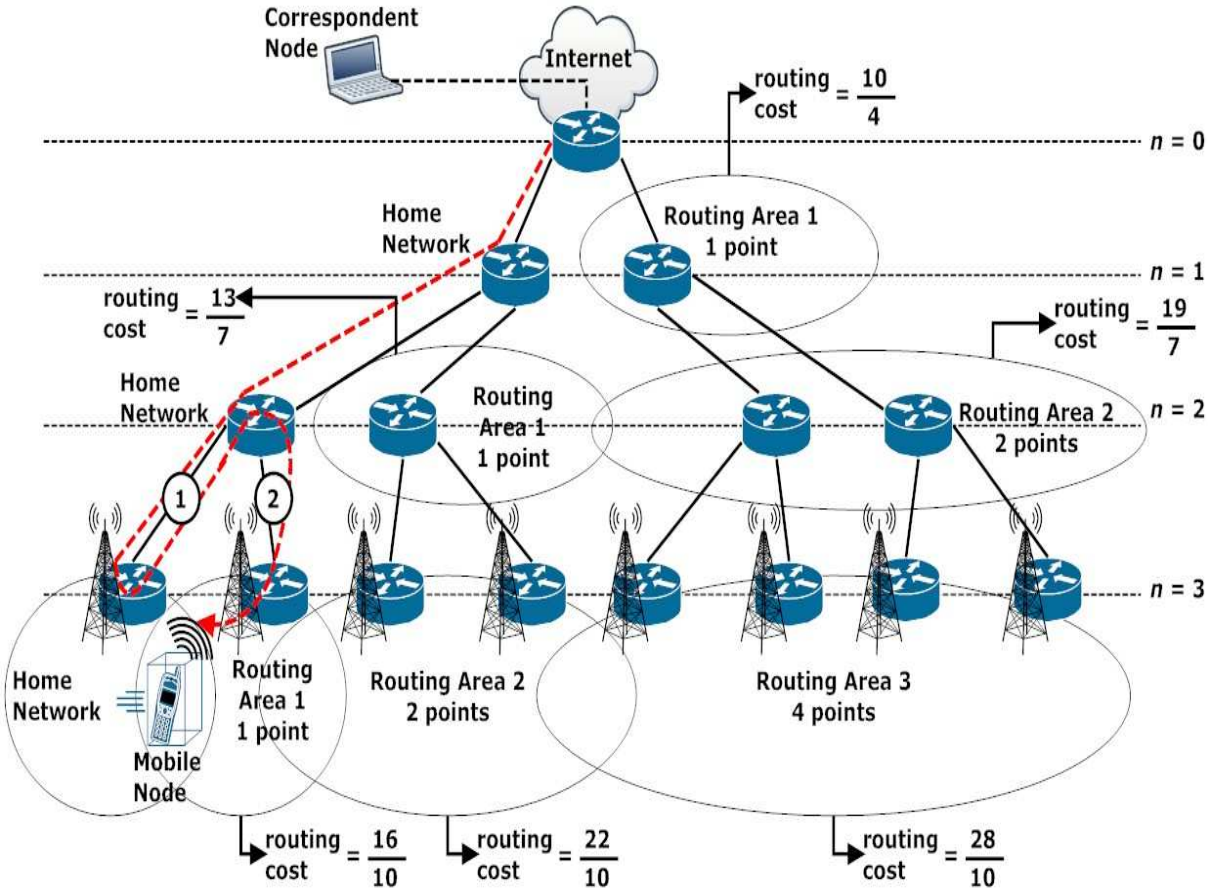


Figura 5.2 – *Custos de roteamento para* $m = 2$ *e* $n = 1, 2$ *e* 3.

Já um pacote enviado para um *host* local da rede a qual o *mobile node* esta conectado, de acordo com o roteamento tradicional, realiza apenas 2 saltos e utiliza 1 vez o *link* de *upstream*. Sendo assim, o custo de roteamento é dado por:

$$C_{routing}^{n=1} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{4 + 0 + 2 \times 3}{2 + 0 + 2 \times 1} = \frac{10}{4}$$

Para $n = 2$, temos que o *mobile node* pode executar três *handovers*, para o primeiro *handover* o custo de roteamento é igual a:

$$C_{routing}^{n=2;\,1°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{5 + 0 + 2 \times 4}{3 + 0 + 2 \times 2} = \frac{13}{7}$$

Para os demais dois *handovers* possíveis que o *mobile node* pode executar, temos que o custo de roteamento é dado por:

$$C_{routing}^{n=2;\ 2°\ e\ 3°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{7+0+2\ \times 6}{3+0+2\ \times 2} = \frac{19}{7}$$

Por fim, para $n = 3$, temos que o custo de roteamento dos 7 *handovers* que o *mobile node* pode executar, é dado por:

- Para o primeiro *handover*:

$$C_{routing}^{n=3;\ 1°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{6+0+2\times 5}{4+0+2\times 3} = \frac{16}{10}$$

- Para os *handovers* de 2 e 3:

$$C_{routing}^{n=3;\ 2°\ e\ 3°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{8+0+2\times 7}{4+0+2\times 3} = \frac{22}{10}$$

- Para os *handovers* de 4 a 7:

$$C_{routing}^{n=3;\ 4°\ a\ 7°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{10+0+2\times 9}{4+0+2\times 3} = \frac{28}{10}$$

Na figura 5.2, a linha tracejada representa o caminho dos pacotes enviados do *correspondente node* para o *mobile node* para $n = 3$ quando o primeiro *handover* é executado da sua *home network* para o primeiro ponto de acesso adjacente.

Observamos que o tráfego utiliza de forma ineficiente a infraestrutura de comunicação, utilizando duas vezes o *link* de *upstream* (identificado pelo número 1 na figura 5.2) entre o roteador da *home network* e o roteador localizado na camada superior, assim como é necessário utilizar duas vezes este roteador, resultando em $L_E^m = 5$ e $H_{MN-CN}^m = 6$.

Já para um roteamento otimizado, o *link* de *upstream* mencionado acima não é se quer utilizado, uma vez que apenas o *link* identificado pelo número 2 na figura 5.2 é utilizado, com o roteador de *upstream* que se coneta aos *link*s 1 e 2 sendo utilizado somente uma vez, resultando então em $L_E^l = 3$ and $H_{MN-CN}^l = 4$.

Agora para uma topologia em árvore com $m = 3$ e $n = 1, 2\ e\ 3$, temos:

- $n = 1$

$$C_{routing}^{n=1;1° \; e \; 2°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{4 + 0 + 2 \times 3}{2 + 0 + 2 \times 1} = \frac{10}{4}$$

- $n = 2$

$$C_{routing}^{n=2;\; 1° \; e \; 2°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{5 + 0 + 2 \times 4}{3 + 0 + 2 \times 2} = \frac{13}{7}$$

$$C_{routing}^{n=2;\; 3° \; a \; 8°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{7 + 0 + 2 \times 6}{3 + 0 + 2 \times 2} = \frac{19}{7}$$

- $n = 3$

$$C_{routing}^{n=3;\; 1° \; e \; 2°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{6 + 0 + 2 \times 5}{4 + 0 + 2 \times 3} = \frac{16}{10}$$

$$C_{routing}^{n=3;\; 3° \; a \; 8°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{8 + 0 + 2 \times 7}{4 + 0 + 2 \times 3} = \frac{22}{10}$$

$$C_{routing}^{n=3;\; 9° \; a \; 26°} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{10 + 0 + 2 \times 9}{4 + 0 + 2 \times 3} = \frac{28}{10}$$

Observamos que o uso da equação (19) para o cálculo da média do custo de roteamento com um valor fixo para $n > 0$ e um valor variável para $m \geq 2$, causa uma progressão geométrica de razão m e primeiro termo $m - 1$ do número de *handovers* que possuem o mesmo custo de roteamento.

Estes *handovers* formam uma área comum chamada de Área de Roteamento (*Routing Area*), onde os *handovers* executados resultam no mesmo custo de roteamento, como pode ser observado na figura 5.2.

Agora ao fixarmos um valor para $m \geq 2$ e variamors o valor de $n > 0$, uma progressão aritmética do componente $C_{m-routing}$ com razão 6 é observada no cálculo do custo de roteamento para as Áreas de Roteamento para dado n. Com outra progressão aritmética dos componentes $C_{m-routing}$ e $C_{l-routing}$ com razão 3 sendo observada, mas agora entre as Áreas de Roteamento de níveis de rede adjacentes (n e $n + 1$). Estas relações podem ser observadas na figura 5.2.

Sendo assim, o custo de roteamento médio dos *draft*s Jaehwoon e Bernardos para uma topologia em árvore simples com *handover* linear do *mobile node*, depende do número de níveis de rede da topologia, n, e do número de nós conectados por cada nó superior, m, sendo definido por:

$$C_{routing}^{Drafts} = \frac{\sum_{i=1}^{n}(m-1)m^{i-1} \times \frac{10+(n-1)3+(i-1)6}{4+(n-1)3}}{m^n - 1} \quad (20)$$

5.4.2 – SDN-DMM

O roteamento empregado na proposta SDN-DMM para tratar da mobilidade do *mobile node* é baseado no encaminhamento de pacotes por meio de fluxos IP bidirecionais de acordo com a nova localização do *mobile node* quando este realiza um *handover*.

Como os pacotes não são roteados levando em consideração a relação entre o IP utilizado pelo *mobile node* e o escopo de endereçamento da rede a qual o *mobile node* se conecta, o roteamento é feito de forma otimizada.

Se considerarmos que a decisão de implementação do fluxo IP bidirecional segue a rota definida pelo IGP utilizado na rede, temos que para qualquer que seja a posição do *mobile node* em relação a sua *home network*, o roteamento empregado sempre será um roteamento otimizado, não adicionando custo.

Deste modo, temos que o custo de roteamento $C_{m-routing}$ para o *mobile node* é igual ao custo de roteamento $C_{l-routing}$ para um *host* local a rede na qual o *mobile node* se conecta. Com o custo de roteamento da proposta SDN-DMM sendo igual a 1 para qualquer topologia ou cenário utilizado quando se baseado no IGP utilizado, conforme a equação (21).

$$C_{routing}^{SDN-DMM} = 1 \quad (21)$$

5.5 – RESUMO COMPARATIVO DAS CARACTERÍSTICAS

A tabela 5.1 apresenta o resumo comparativo dos principais aspectos relacionados ao gerenciamento de mobilidade distribuído dos trabalhos analisados.

Tabela 5.1 – *Comparação de aspectos referentes a DMM.*

Solução:	Draft-Jaehwoon	Draft-Bernardos	SDN-DMM
Roteamento	É sub-otimizado: tráfego passa sempre por ponto central (MAG)	É sub-otimizado: o tráfego passa por ponto central (A-MAAR)	É otimizado: sempre é definido o melhor caminho para o fluxo IP
Desempenho	Baixo custo de sinalização com sobrecarga de processos nos ativos (*cache* e *binding*) envolvidos e para entrega de pacotes	Relativo custo de sinalização com sobrecarga de processos nos ativos envolvidos (*cache* e *binding*) e para entrega de pacotes via A-MAAR	Relativo custo de sinalização, sem sobrecarga de processo e com transporte otimizado
Requisitos para os hosts finais	Não requer modificação	Requer que o host final suporte múltiplos endereçamento IP na interface de conexão	Não requer modificação
Arquitetura e Complexidade	Totalmente distribuído e baseado na rede	Parcialmente distribuído e baseado na rede	Parcialmente distribuído e baseado na rede
Escalabilidade	Escalável	Escalável	Escalável (controlador em cluster)

5.6 – RESULTADOS OBTIDOS

Esta seção apresenta os resultados obtidos com a avaliação analítica das propostas, permitindo discutir os possíveis benefícios e desvantagens obtidos com uma abordagem SDN-DMM em relação ao custo de sinalização, custo de entrega de pacote, custo de latência de *handover* e custo de roteamento.

5.6.1 – Sinalização

Na avaliação do custo de sinalização, de modo a obter um resultado comparativo geral, devido ao fato da topologia de rede afetar diferentemente o custo de acordo com a abordagem DMM utilizada, consideramos que o custo para transmitir um pacote de controle entre as entidades da rede, custo C_{X-Y}, é igual para todas as soluções analisadas, ou seja, $C_{MAG-MAG} = C_{CMD-MAAR} = C_{CTR-OFS} = C$ [16].

Além disso, para que o custo de processamento do pacote de controle pela entidade X, PC_X, não afete significativemnete o valor final do custo de sinalização, prejudicando a avaliação de uma perspectiva de abordagem DMM utilizada, considerados o mesmo custo de processamento para todas as entidades nas soluções, onde $PC_X = PC$.

A comparação entre o custo de sinalização entre a proposta SDN-DMM e o *draft*-Jaehwoon, pode ser observada na equação (22), obtida pela razão entre o custo de sinalização SDN-DMM, equação (8) e o custo de sinalização *draft*-Jaehwoon, soma das equações (2), (3) e (4).

$$\frac{C_{SIG}^{SDN-DMM}}{C_{SIG}^{\text{Draft-Jaehwoon}}} = \frac{2 + N_{PR}}{6 + 2R_{BCE}} \tag{22}$$

Para o contexto de comparação entre a proposta SDN-DMM e o *draft*-Jaehwoon, uma vez que esta utiliza o conceito de endereço HNP e o processo de sinalização sempre envolve no máximo dois pares de MAG, podemos considerar como N_{PR} o número de switches *OpenFlow* envolvidos no processo de sinalização para realizar uma atualização do fluxo IP bidirecional para suportar o *handover* de um *mobile node*.

Deste modo, a figura 5.3 apresenta a comparação do custo de sinalização entre a proposta SDN-DMM e o *draft*-Jaehwoon, com a variação do tempo de permanência na *subnet* para diferentes números de switches envolvidos no processo.

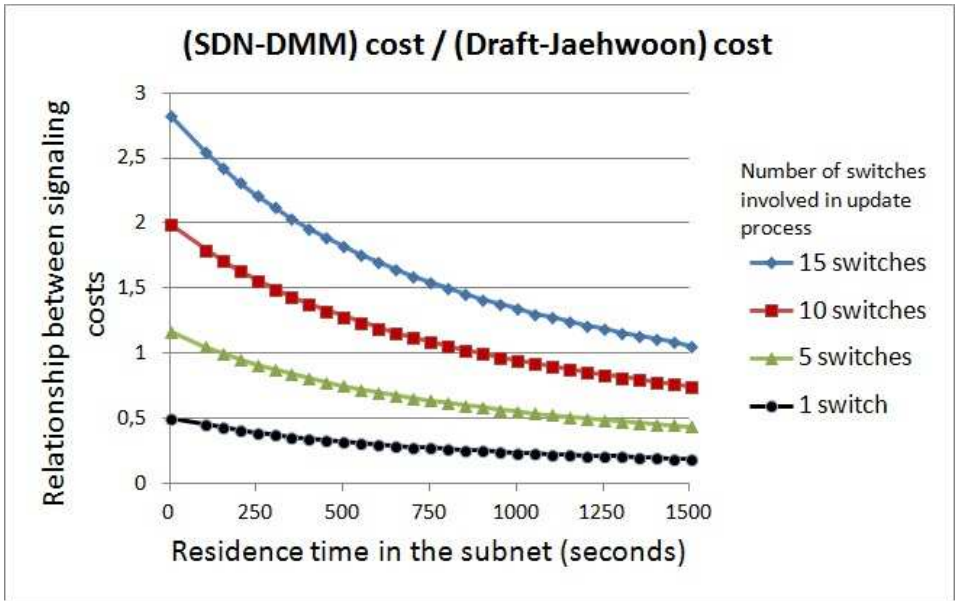


Figura 5.3 – *Custo de Sinalização: SDN-DMM vs Draft-Jaehwoon.*

Como o fluxo IP bidirecional precisa ser estabelecido entre os switches afetados pelo processo de *handover* do *mobile node*, podemos observar na figura 5.3 que quanto maior o número de switches *OpenFlow* afetados pelo processo de atualização de um novo fluxo IP bidirecional, maior o custo de sinalização da solução SDN-DMM em comparação com a proposta *draft*-Jaehwoon. Isso ocorre porque na proposta *draft*-Jaehwoon a troca de sinalização é limitada no máximo entre dois pares de MAG.

Considerando que as alterações de fluxo IP bidirecional ocorrem na borda da rede, ou seja, na rede de acesso, onde podemos considerar que o número de switches afetados pelo processo de atualização provavelmente não ultrapassa 10 switches [15], fato que torna o custo de sinalização da proposta SDN-DMM rapidamente inferior ao custo do *draft*-Jaehwoon com o aumento do tempo de permanência do *mobile node* na *subnet*.

O custo de sinalização da proposta SDN-DMM torna-se menor na medida em que o tempo de permanência do *mobile node* na *subnet* aumenta, devido ao fato de nesse cenário o custo com atualizações periódicas possuírem um peso maior na determinação do custo de sinalização total. Como a proposta SDN-DMM não apresenta tal custo de atualização periódica e o *draft*-Jaehwoon apresenta, a proposta SDN-DMM possui um menor custo de sinalização neste cenário.

Quando comparamos a proposta SDN-DMM com a proposta do *draft*-Bernardos, obtemos a equação (23), que representa a razão entre o custo de sinalização SDN-DMM de acordo com a equação (8) e o custo de sinalização *draft*-Bernardos, que é a soma das equações (5), (6) e (7).

$$\frac{C_{SIG}^{SDN-DMM}}{C_{SIG}^{\text{Draft}-\text{Bernardos}}} = \frac{2 + N_{PR}}{6 + 2N_{PR} + 2R_{BCE}} \tag{23}$$

A comparação entre o custo de sinalização da proposta SDN-DMM com o *draft*-Bernardos pode ser observada na figura 5.4, onde a relação entre custos de sinalização é apresentada de acordo com a variação do tempo de permanência do *mobile node* na *subnet*, que pode ser entendido como a variação da sua taxa de *handover*, para quatro diferentes números de prefixos ativos com sessões correntes.

A proposta SDN-DMM possui um custo de sinalização pelo menos 50% menor quando comparada com o draf-Bernados devido a dois fatores principais:

1) O primeiro fator é que os switches *OpenFlow* que recebem as mensagens do controlador em um processo de atualização para o estabelecimento de um novo fluxo IP bidirecional, não precisam responder a esta mensagem, o que faz o custo C_{update} da solução SDN-DMM, que aumenta de acordo com o crescimento do número de prefixos LNP, ser inferior;
2) O segundo fator é que devido à proposta SDN-DMM não utilizar mensagens para atualizações periódicas e para desregistro, quando o tempo de permanência do *mobile node* na *subnet* aumenta, situação onde as atualizações periódicas tem um peso maior no cálculo do custo de sinalização, o custo relativo da solução SDN, por não possuir este componente, diminui.

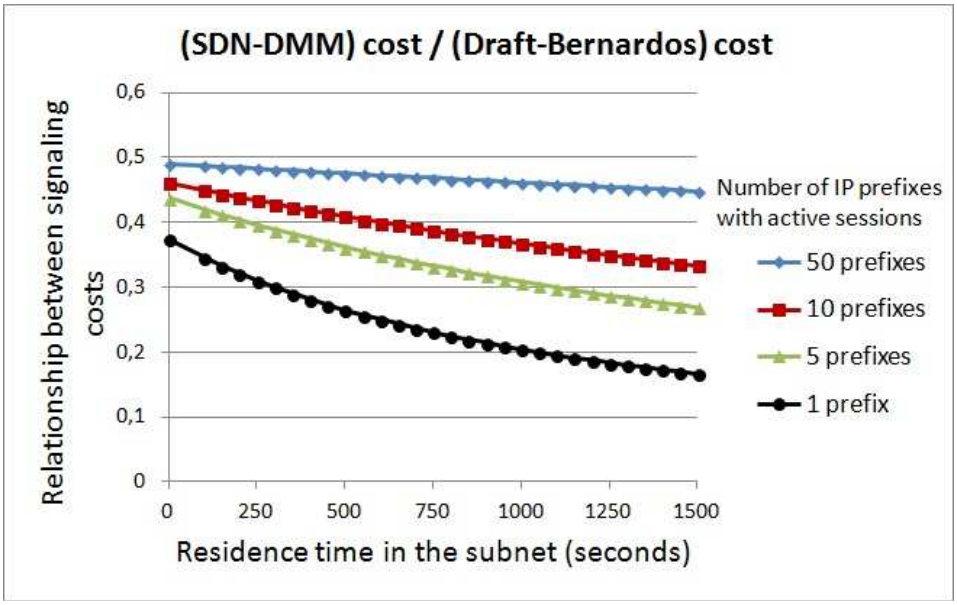


Figura 5.4 – *Custo de Sinalização: SDN-DMM vs Draft-Bernardos.*

5.6.2 – Entrega de pacote

Para realizar uma avaliação de custo de entrega de pacote mais igualitária das propostas analisadas, o custo C_{SMA-MN} relacionado com a entregua final do pacote para o *mobile node* pelo enlace sem fio é desconsiderado [16] e o custo C_{CN-NMA} é considerado igual e com valor unitário para todas as soluções, ou seja, o custo para que um pacote entre no domínio e seja entregue para a primeira entidade envolvida no cenário de mobilidade é dado por $C_{CN-AMAG} = C_{CN-AMAAR} = C_{CN-OFS} = 1$.

Deste modo o foco é mantido na avaliação da abordagem utilizada em cada proposta e no custo respectivo para a entrega do pacote dentro do domínio de mobilidade. A comparação do custo de entrega de pacotes na proposta SDN-DMM e na proposta do *draft*-Jaehwoon pode ser observada na figura 5.5 que apresenta a razão obtida entre a equação (13) e a equação (11), representada pela expressão (24).

$$\frac{C_{EP}^{SDN-DMM}}{C_{EP}^{\text{Draft-Jaehwoon}}} = \frac{1}{1 + \tau D} \tag{24}$$

A figura 5.5 apresenta a relação entre os custos de entrega de pacote com a variação da penalização de tunelamento para a variação da distância entre os A-MAG e o S-MAG que realizam o tunelamento, onde a distância igual a 1 pode ser entendido como a distância base onde o A-MAG e o S-MAG estão diretamente conectados, a distância igual a 2 como o dobro da distância base e assim respectivamente para as distâncias 5 e 10.

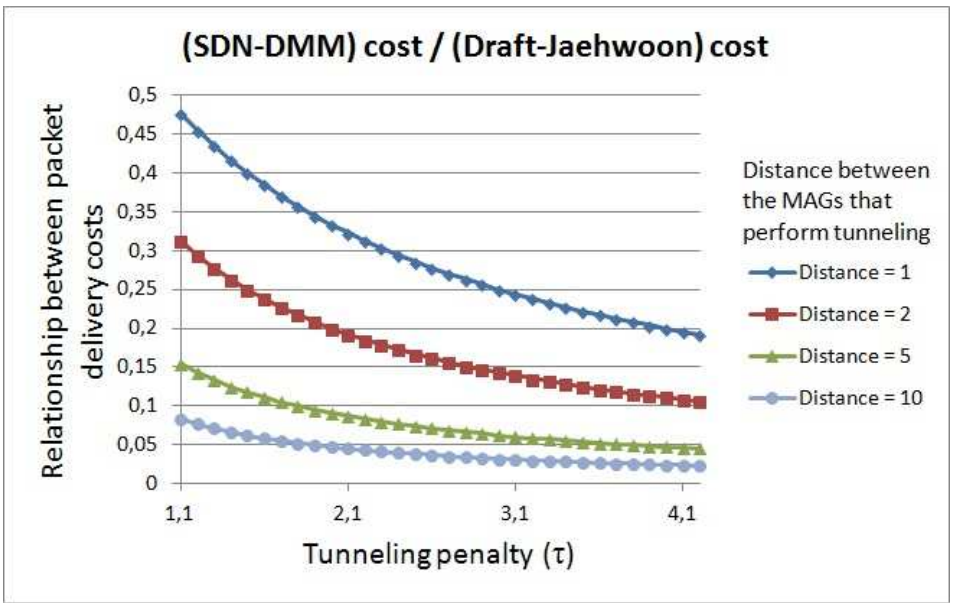


Figura 5.5 – *Custo de entrega de pacote: SDN-DMM vs draft-Jaehwoon.*

Para a comparação do custo de entrega de pacote com a proposta apresentada no *draft*-Bernardos, a figura 5.6, apresenta a razão obtida entre a equação (13) e a equação (12), representada pela expressão (25).

$$\frac{C_{EP}^{SDN-DMM}}{C_{EP}^{\text{Draft}-\text{Bernardos}}} = \frac{1}{1 + N_{PR}\tau D} \tag{25}$$

A relação entre os custos de entrega de pacote entre a proposta SDN-DMM e a proposta do *draft*-Bernardos é apresentada na figura 5.6 para uma penalidade $\tau = 2$ com a variação do número de prefixos IP com sessões ativas.

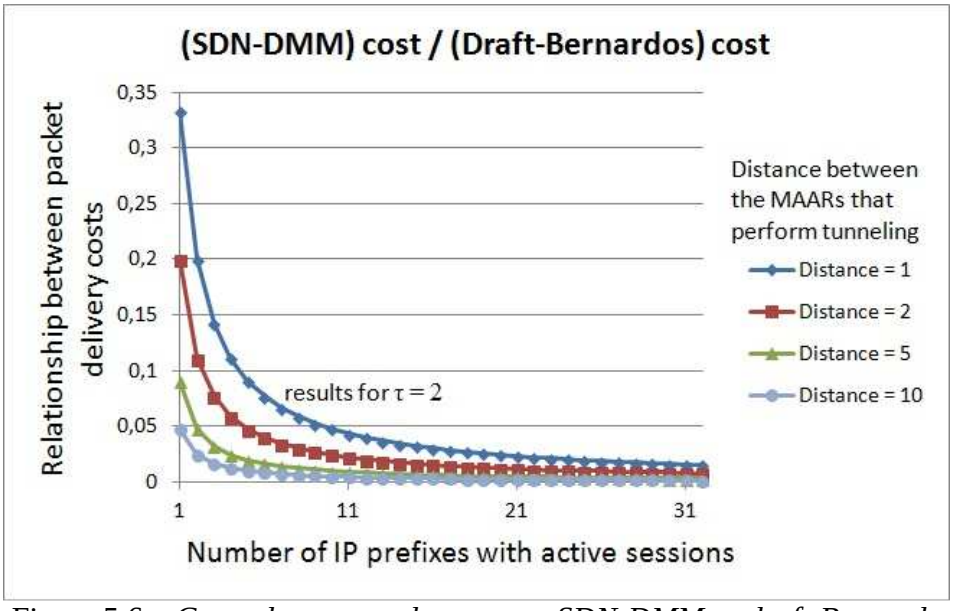


Figura 5.6 – *Custo de entrega de pacotes: SDN-DMM vs draft-Bernardos.*

Quando comparada com as propostas apresentadas no *draft*-Jaehwoon e no *draft*-Bernardos, a solução SDN-DMM apresenta um menor custo de entrega de pacotes, isto porque o tráfego destinado ao *mobile node* é entregue diretamente à entidade de rede na qual o *mobile node* está conectado sem utilizar técnicas de tunelamento.

Como na proposta do *draft*-Jaehwoon o tráfego é entregue por meio de um túnel entre os MAGs, na medida em que a penalização do túnel e a distância entre os MAGs aumentam, a solução SDN-DMM apresenta um menor custo de entrega de pacote.

O custo de entrega de pacote da solução SDN-DMM em comparação com o *draft*-Bernardos é ainda menor devido ao conceito de pLNP utilizado neste *draft*. Onde é necessário estabelecer vários túneis, um para cada prefixo pLNP que ainda está sendo utilizado pelo *mobile node*.

Assim com o aumento do número de prefixos pLNP, que pode ser entendido com o *mobile node* executando mais processos de *handover* mantendo as sessões ativas, a proposta SDN-DMM torna-se mais atrativa.

A solução SDN-DMM sempre permite uma entrega otimizada de pacotes por se basear nos fluxos IP bidirecionais que são estabelecidos de acordo com os cenários de mobilidade, onde o plano de dados é a justado para utilizar o melhor caminho.

Não existe a penalização devido ao uso de tunelamento, não sendo influenciado também pela distância entre os elementos que o realizam.

5.6.3 – Latência de Handover

A fim de se avaliar o impacto da etapa $t_{binding}$ para o custo de latência de *handover*, o tempo gasto nas etapas de operação de *handover* t_{L2} e t_{auth} e para a troca de pacotes entre o *mobile node* e o seu ponto de acesso à rede, RTT_{MN-PA}, serão considerados iguais em todas as propostas analisadas. Tornando assim a variação dos componentes que constituem o $t_{binding}$ o fator mais impactante para o custo total de latência de *handover*.

Baseado em [16], consideramos a relação apresentada na expressão (26), assim como um processamento igualitário de pacote pelas unidades de rede, demonstrado na expressão (27) e uma distância fixa entre o elemento central de controle e cada ponto de acesso à rede na rede de acesso nas propostas DMM parcialmente distribuídas, expressão (28).

$$t_{l2} + t_{auth} + RTT_{MN-MAG} + 2T_{proc}^{MAG} = t_{l2} + t_{auth} + RTT_{MN-MAAR} + T_{proc}^{MAAR} + T_{proc}^{CMD} = t_{l2} + t_{auth} + RTT_{MN-OFS} + T_{proc}^{OFS} + T_{proc}^{CTR} = T_{commun} \quad (26)$$

$$T_{proc}^{MN} = T_{proc}^{MAAR} = T_{proc}^{CMD} = T_{proc}^{OFS} = T_{proc}^{CTR} = T_{proc}^{Unit} \quad (27)$$

$$RTT_{CMD-MAAR} = RTT_{CTR-OFS} = RTT_{control} \quad (28)$$

Para T_{commun}é atribuído um valor de 50 ms, obtido nos experimentos realizados em [16]; para T_{proc}^{Unit} um valor de 1 ms, considerando que o tempo de processamento de um pacote pela unidade é igual à latência inserida pelo processamento de um pacote em um salto de roteamento, que em uma rede de *backbone* é tipicamente inferior a 1 ms [14]; e para $RTT_{control}$ utilizamos os valor de 1 ms, 5 ms, 10ms e 20 ms, valores comuns para a comunicação entre os ativos de borda e o concentrador localizado em um Ponto de Presença ao longo de uma rede de *backhall* [15].

A equação apresentada em (29), obtida pela razão entre a equação (18) e (16), representa a relação entre custos de latência de *handover* da proposta SDN-DMM com o *draft*- Jaehwoon.

$$\frac{C_{handover}^{SDN-DMM}}{C_{handover}^{\text{Draft}-\text{Jaehwoon}}} = \frac{T_{commun} + 4T_{proc}^{Unit} + 2RTT_{CTR-OFS}}{2T_{commun} + 4T_{proc}^{Unit} + 2RTT_{MAG-MAG}} \quad (29)$$

A figura 5.7 ilustra o impacto na relação entre custos de latência de *handover* entre a proposta SDN-DMM e o *draft*-Jaehwoon, de acordo com a variação da latência na medida em que o *mobile node* executa novos processos de *handover* na rede, o que pode ser entendido como o aumento da distância entre os MAGs que estabelecem o túnel para encaminhamento dos pacotes do *mobile node*, para fornecer a este conectividade global.

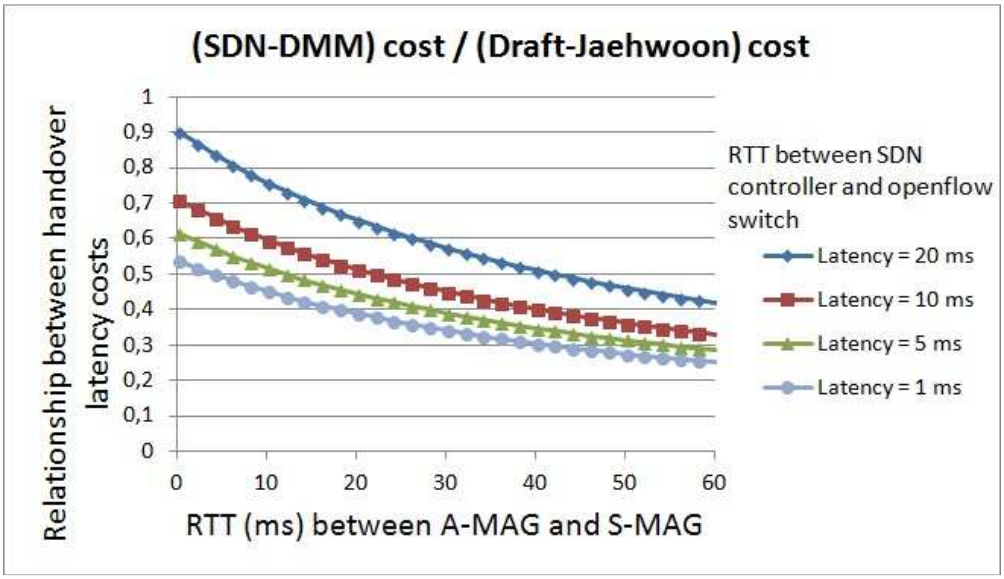


Figura 5.7 – *Custo de latência de handover: SDN-DMM vs draft-Jaehwoon.*

Podemos obsevar que devido à abordagem utilizada para prover a mobilidade das sessões IP no *draft*-Jaehwoon, onde é necessário duas vezes o processo T_{commun} em comparação com a proposta SDN-DMM, esta apresenta sempre um custo de latência de *handover* menor quando a distância entre o concentrador e o switch *OpenFlow* é igual a distância entre os MAGs que realizam o tunelamento. No caso de um RTT típico de 5 ms para ambos, a redução do custo chega a ser aproximadamente 44%.

Outro aspecto importante é que a implementação do controlador em relação aos switches *OpenFlow* nas redes de acesso é geralmente realizado de forma que a sua localização resulte na menor latência coletiva entre eles, ou seja, o controlador assume uma posição central de forma que na medida em que o *mobile node* realiza o *handover* entre as redes de acesso, a latência entre o switch *OpenFlow* e o controlador não é alterada significativamente.

Outro forma de implementação que reduz a latência, é a implementação do controlador em modo cluster hierárquico, onde o controlador responsável por determinadas redes de acesso é implementado próximo a elas.

Deste modo, como os MAGs estão localizados na borda da rede de acesso, existe um cenário onde a movimentação do *mobile node* pelas redes de acesso aumenta a latência entre o S-MAG e o A-MAG, o que não ocorre na proposta SDN-DMM, contribuindo para que esta proposta mantenha um menor custo de latência de *handover* durante a movimentação do *mobile node*, como pode ser observado pelas curvas apresentadas na figura 5.7.

A relação entre custos de latência de *handover* entre a proposta SDN-DMM e o *draft*-Bernardos pode ser observada na equação (30), obtida pela razão entre a equação (18) e (17).

$$\frac{C_{handover}^{SDN-DMM}}{C_{handover}^{\text{Draft}-\text{Bernardos}}} = \frac{T_{commun} + 4T_{proc}^{Unit} + 2RTT_{CTR-OFS}}{T_{commun} + 6T_{proc}^{Unit} + 2RTT_{CMD-MAAR} + 2RTT_{MAAR-MAAR}} \quad (30)$$

O custo de latência de *handover* é maior no *draft*-Bernardos quando comparado com a proposta SDN-DMM devido ao maior número de interações na etapa de $t_{binding}$.

Primeiro é necessário o registro e atualização sobre os prefixos que ainda estão ativos no *mobile node* durante o *handover*, realizado entre o S-MAAR e o CMD que resulta no componente $RTT_{CMD-MAAR}$.

Depois é preciso estabelecer o túnel entre o A-MAAR e o S-MAAR para encaminhar o tráfego dos prefixos que ainda estão ativos, adicionando o componente $RTT_{MAAR-MAAR}$. O que resulta também em uma maior quantidade de processamento T_{proc}^{Unit}.

A figura 5.8 ilustra os resultados obtidos com a equação (30).

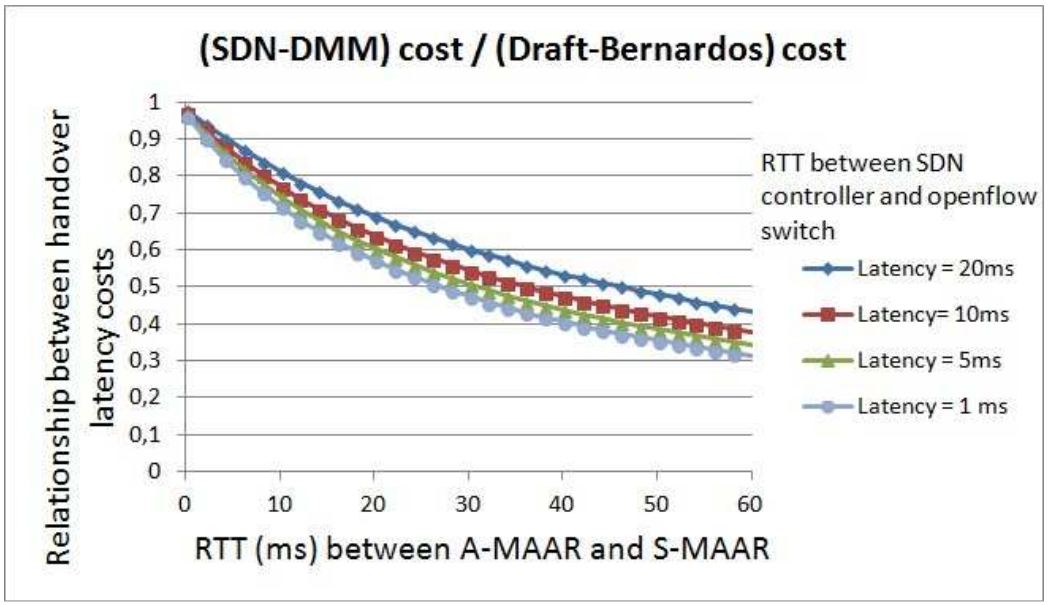


Figura 5.8 – *Custo de latência de handover: SDN-DMM vs draft-Bernardos.*

Como consideramos que a distância entre o elemento de controle e o ponto de acesso à rede são iguais nas duas propostas, ou seja, $RTT_{CTR-OFS} = RTT_{CMD-MAAR}$, o fator predominante para o custo de latência de *handover* é o $RTT_{MAAR-MAAR}$.

Do mesmo modo que no *draft*-Jaehwoon, como os MAAR estão localizados na borda das redes de acesso, ocorre a possibilidade de aumento da latência entre o S-MAAR e o A-MAAR na medida em que o *mobile node* executa o *handover*, aumentando assim o custo de latência de *handover* quando comparado com a proposta SDN-DMM.

Isto pode ser obsevado no comportamento das curvas da figura 5.8 com o aumento da latência no eixo das abscissas.

5.6.4 – Roteamento

A equação (31) representa a relação entre os custos de roteamento da proposta SDN-DMM com os *draft*s Jaehwoon e Bernardos, obtida pela razão entre a equação (21) e a equação (20).

$$\frac{C_{routing}^{SDN-DMM}}{C_{routing}^{Drafts}} = \frac{m^n - 1}{\sum_{i=1}^{n}(m-1)m^{i-1} \times \frac{10+(n-1)3+(i-1)6}{4+(n-1)3}} \quad (31)$$

Tendo como base cenários reais de implementação e assim melhor avaliar a relação entre os custos de roteamento das propostas apresentadas, a figura 5.9 ilustra a relação dos custos para

quatro níveis de rede ($n = 1, 2, 3\ e\ 4$) com a variação do número de nós conectados por um nó de nível superior (m) de 1 a 20 [14, 15].

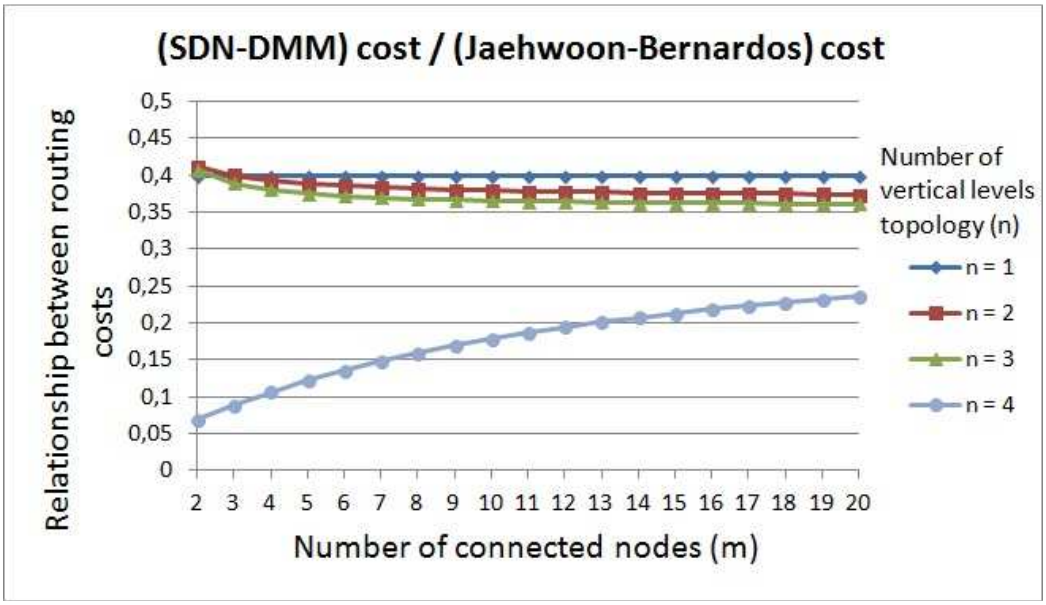


Figura 5.9 – *Custo de roteamento: SDN-DMM vs draft-Jaehwoon/Bernardos.*

Devido ao roteamento otimizado da proposta SDN-DMM, podemos observar que o custo de roteamento é pelo menos 50% inferior que o custo das demais propostas. Isso ocorre porque na proposta SDN-DMM o tráfego destinado ao *mobile node* é encaminhado diretamente a este, como se fosse tratado como um *host* local da rede a qual está conectado.

Já nos *draft*s Jaehwoon e Bernardos, o tráfego precisa primeiro alcançar o *mobility anchor* responsável pelo endereçamento do *mobile node* para depois ser encaminhado para a nova localização do *mobile node* através de tunelamento, utilizando então mais recursos da rede e de maneira redundante.

Para uma topologia em árvore simples com $n = 1$, o custo de roteamento da proposta SDN-DMM é 40% do custo dos *draft*s, não importando a quantidade de nós conectados ao nó de nível superior (m). Isto ocorre porque a quantidade de saltos e de *links* utilizados para o encaminhamento de pacotes para o *mobile node* realizando *handover* de maneira horizontal e linear, como descrito anteriormente, é constante.

Quando se aumenta o número de níveis da rede para $n = 2\ e\ 3$, observamos na figura 5.9 um aumento, embora baixo (menos de 3%), do custo inicial de roteamento para $m = 2$ em relação ao obtido com $n = 1$. Isso é devido à diminuição da quantidade proporcional de recursos utilizados na comparação entre a proposta SDN-DMM e os *draft*s Jaehwoon e Bernardos nestes níveis de rede.

O custo de roteamento para estes níveis de rede sofre uma leve redução em $2 \leq m \leq 5$, cerca de até 6% em comparação ao custo para $n = 1$, onde a partir de $m > 5$ temos o início da estabilização do custo.

Este cenário ocorre porque na medida em que o m aumenta, considerando a movimentação de *handover* horizontal e linear adotada, temos o aumento da área em que o custo de roteamento permanece o mesmo para os *handovers* executados (Área de Roteamento) e que devido à relação com a quantidade de enlaces *upstream* utilizados, acaba por causar a estabilidade da média do custo de roteamento.

A partir do nível de rede $n \geq 4$, temos uma redução drástica de mais de 90% do custo de roteamento na comparação da proposta SDN-DMM com os *draft*s Jaehwoon e Bernardos.

Isso ocorre devido ao aumento significativo da quantidade proporcional de recursos de rede utilizados para realizar o roteamento pelos *draft*s quando comparado com a proposta SDN-DMM. Em especial o aumento relacionado ao número de *links upstream* necessários, que possuem maior peso no cálculo do custo de roteamento.

A figura 5.10 apresenta a relação entre custos de roteamento para $2 \leq m \geq 1000$, apenas com o objetivo de ilustrar o comportamento do custo de roteamento para $n = 4$, dado que de acordo com a topologia considerada e os cenários reais de implementação, não é comum se ter um $m > 20$ [14, 15].

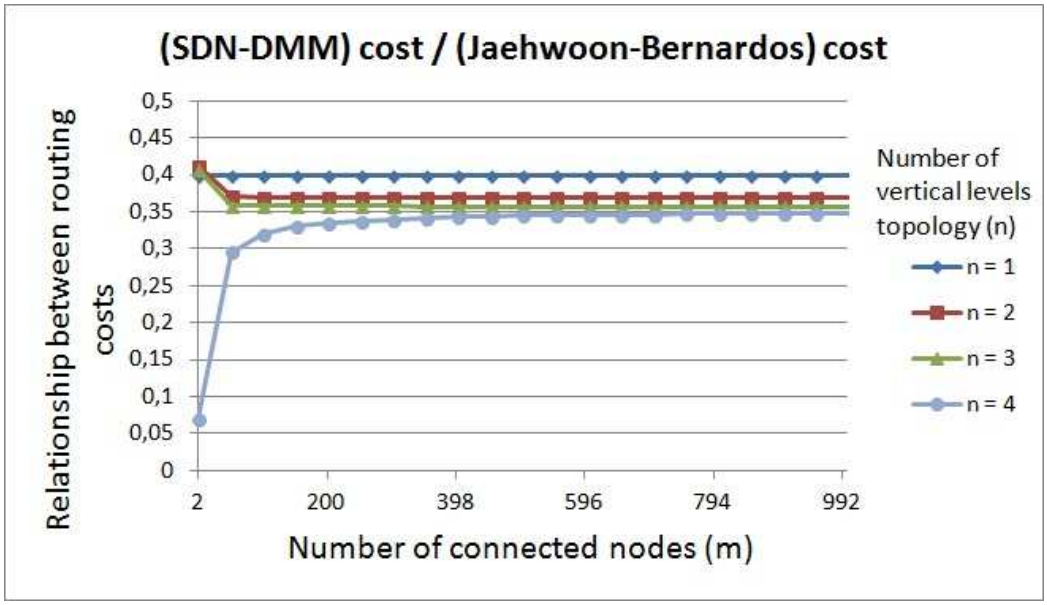


Figura 5.10 – *Custo de roteamento: SDN-DMM vs drafts Jaehwoon/Bernardos (*$2 \leq m \leq 1000$*).*

Neste cenário, $n = 4$, para aproximadamente $2 \leq m \leq 30$ temos um aumento rápido do custo de roteamento devido ao crescimento da Área de Roteamento onde as propostas

apresentam um custo de roteamento mais próximo entre si, elevando assim a média da relação entre os custos de roteamento.

A partir, aproximadamente, de $m \geq 150$ incia-se a estabilização do custo de roteamento, devido à predominância da última área de custo de roteamento, que pode ser observada pelo componente $\frac{m^n - 1}{(m-1)m^{i-1}}$ da equação (31).

5.7 – CONSIDERAÇÕES FINAIS

A análise das métricas de custo de sinalização, de entrega de pacote, de latência de *handover* e de roteamento entre a proposta SDN-DMM e os *drafts* Jaehwoon e Bernardos, demonstra que o uso de uma abordagem SDN para o gerenciamento de mobilidade distribuído permite a otimização dos recursos e da infraestrutura disponível, acarretando em um desempenho satisfatório de modo geral.

O custo de sinalização é inferior quando comparado ao *draft* Bernardos devido a não existir mensagens relacionadas às atualizações periódicas na proposta SDN-DMM e não ser necessária uma mensagem de resposta ao controlador SDN quando solicitada a adição de um fluxo IP bidirecional. Já na comparação com o *draft* Jaehwoon, o custo de sinalização relativo é reduzido na medida em que o tempo de permanência do *host* na *subnet* aumenta, devido ao aumento do peso no cálculo do custo com as atualizações periódicas.

Considerando que na proposta SDN-DMM não é necessário utilizar técnicas de tunelamento para entrega de pacotes ao *host* em mobilidade, onde o tráfego é entregue diretamente a ele através do encaminhamento baseado em fluxos IP bidirecionais, o custo de entrega de pacote da proposta é sempre inferior ao obtido nos *drafts* Jaehwoon e Bernardos, nos quais são utilizadas técnicas de tunelamento para a entrega dos pacotes baseado no roteamento tradicional.

A latência de *handover* é inferior na proposta SDN-DMM devido à menor quantidade de mensagens necessárias no plano de controle para o estabelecimento de um novo caminho de comunicação na infraestrutura de rede com o uso dos fluxos IP bidirecionais quando o *host* executa a mobilidade.

Como a entrega de pacotes é realizada sempre de modo otimizado através dos fluxos IP seguindo as rotas definidas pelo IGP ao considerar o *mobile node* em mobilidade como um *host* local à rede a qual está conectado, o custo de roteamento da proposta mostrou-se comumente inferior aos *drafts* Jaehwoon e Bernardos, nos quais o tunelamento e roteamento triangular provocam um uso ineficiente da infraestrutura.

6 – IMPLEMENTAÇÃO DA PROPOSTA SDN-DMM

Este capítulo descreve a implementação da proposta SDN-DMM em um cenário de experimentação real, utilizando equipamentos e *software*s de mercado, assim como os resultados obtidos. Inicialmente é feita uma descrição da arquitetura e das topologias consideradas, equipamentos, *software*s, cenários implementados e métricas analisadas. Logo após são tratados aspectos de configuração e de realização de testes, seguidos da apresentação e discussão dos resultados.

6.1 – DESCRIÇÃO GERAL

A proposta SDN-DMM foi implementada em um cenário real de experimentação utilizando equipamentos e *softwares* de mercado. Os equipamentos utilizados são apresentados na tabela 6.1, que apresenta as principais especificações técnicas relevantes para a experimentação.

Tabela 6.1 – *Especificação técnica dos equipamentos utilizados na experimentação.*

Fabricante:	Extreme Networks
Série:	Summit
Modelo:	Summit X460-24t
Tipo:	Ethernet Switch Layer 2/Layer 3
Número de portas:	24 portas
RJ-45 10/100/1000Base-T:	20 portas
SFP 10/100/1000Base-X:	04 portas
Shared Ports RJ-45/SFP	04 portas
Console RS-232:	01 porta
Expansão SFP+ 10GBASE-X:	02 portas
Capacidade de encaminhamento:	176 Gbps
Transmissão de pacotes:	130.9 Mpps
Memória RAM:	1 GB ECC DRAM
Memória Flash:	1 GB Compact Flash
CPU:	64-bit MIPS 600 MHz
Tabela MAC:	32k
VLANs:	4.094
Tamanho máximo do pacote:	9216 Bytes
Fabricante:	Cisco Systems
Série:	Linksys
Modelo:	WRT160N
Tipo:	Wireless Router
Número de portas:	05 portas
RJ-45 LAN 10/100Base-T:	04 portas
RJ-45 WAN 10/100Base-T:	01 portas
Padrões:	IEEE 802.11g, IEEE 802.11n, IEEE 802.3, IEEE 802.3u
Número de antenas:	02 antenas
RF Pwr (EIRP):	16,5 dBm
Ganho de antena:	1,8 dBi
Fabricante:	Sony
Série:	Vaio
Modelo:	VAIO Fit15F
Tipo:	Notebook
CPU:	Intel Core i3-5005U 2.00GHz, 3MB L2 Cache, Dual Core
Memória RAM:	4GB RAM SO-DIMM DDR3L
Conectividade Wireless:	Wi-FI IEEE 802.11 b/g/n
Conectividade Cabeada:	Gigabit Ethernet 10/100/1000 Mbps
Sistema Operacional:	Windows 7

No total, foram utilizados para a implementação do cenário de experimentação 01 Switch Extreme X460-24t, 02 *Access Points* Cisco WRT160N e 03 *Notebooks* VAIO Fit15F, com as funções desempenhadas por cada um sendo:

- Switch Extreme X460-24t = Switch *Openflow*, denominado OFS1;
- 01 *Access Point* Cisco WRT160N = *Access Point*, denominado AP-A;
- 01 *Access Point* Cisco WRT160N = *Access Point*, denominado AP-B;
- 01 *Notebook* VAIO Fit15F = *Mobile Node*, denominado MN1;
- 01 *Notebook* VAIO Fit15F = *Media Server*, denominado MD1;
- 01 *Notebook* VAIO Fit15F = Servidor de virtualização, denominado N1, para o controlador SDN denominado C1.

As métricas de rede selecionadas para análise foram:

1. Taxa de vazão UDP (*Throughput* UDP): capacidade de transmissão entre o MN1 e o MD antes e depois do MN1 realizar o *handover*;
2. Taxa de vazão máxima TCP (*Throughput* TCP): capacidade de transmissão entre o MN1 e o MD antes e depois do MN1 realizar o *handover*;
3. Perda de pacotes (UDP) entre o MN1 e o MD para o cenário sem mobilidade e o cenário com mobilidade;
4. Perda de pacotes (ICMP) entre o MN1 e o MD para o cenário sem mobilidade e o cenário com mobilidade;
5. Latência RTT (ICMP) entre o MN1 e o MD para o cenário sem mobilidade e o cenário com mobilidade;
6. Latência de *handover*: intervalo de tempo entre o último pacote da sessão recebido pelo MD antes do MN1 realizar o *handover* e o primeiro pacote recebido da sessão depois do MN1 ter realizado o *handover*.

As métricas foram analisadas para os dois principais cenários de *handover* a seguir:

- Cenário A – *Handover* por Chaveamento Manual: Com o objetivo de verificar diretamente o impacto que a proposta SDN-DMM tem sobre as métricas de rede analisadas, neste cenário os *hosts* MN1 e MD1 são conectados diretamente ao switch *OpenFlow* através de cabos UTP. O *handover* do *host* MN1 da rede A para a rede B é realizado de forma manual através da sua desconexão física da porta associada à rede A e a sua conexão a porta associada à rede B diretamente no switch *OpenFlow* enquanto mantém sessões ativas com o MD1. O intuito deste cenário é eliminar o possível impacto causado pelo *handover* na parte a rádio da rede sobre as

métricas de rede analisadas, deste modo os resultados obtidos representam mais fielmente o real impacto da proposta em relação à abordagem DMM utilizada.

- Cenário B – *Handover Wireless* Automatizado: Com o objetivo de verificar como a proposta se comporta em um cenário real de *handover*, ou seja, quando é implementada em conjunto com o *handover* realizado na parte a rádio da rede, neste cenário o MN1 se conecta ao *Access Point* AP-A e realiza o *handover* para o *Access Point* AP-B enquanto mantém sessões ativas com o *host* MD1. Deste modo os resultados obtidos refletem melhor uma implementação completa da proposta para mobilidade.

Os principais *softwares*, sistemas operacionais e protocolos utilizados para a implementação da proposta, geração e coleta das estatísticas, são descritos na tabela 6.2.

Tabela 6.2 – *Descrição dos softwares utilizados na experimentação.*

Nome do *Software*	Versão	Descrição
iPerf	2 e 3.1.3	Ferramenta para geração de tráfego UDP/TCP que permite a mensuração de diferentes parâmetros de rede, principalmente a vazão, latência e perda de pacotes.
Wireshark	2.2.0	Ferramenta para análise de protocolos que realiza a captura de pacotes mostrando informações como marcadores de tempo e encapsulamentos.
ONOS	1.7.0	Sistema operacional SDN aberto para provedores de serviço com arquitetura escalável, de alta disponibilidade e desempenho baseado na linguagem JAVA.
Docker	1.11.2	Ferramenta utilizada para criação de *container* com conceito semelhante as das máquinas virtuais.
Ubuntu Server	14.04. LTS	Sistema Operacional de código aberto criado a partir do Linux baseado no Debian personalizado para utilização em servidores de rede.
Windows 7	Professional	Sistema Operacional proprietário da Microsoft para utilização em *host*s final de usuários.
VMware Workstation Pro	12.1.1	*Software* utilizado para criação de máquinas virtuais.
EXOS	15.7.1.4	Sistema Operacional proprietário da Extreme Networks utilizado no switch Summit X460.
OpenFlow	1.3	Protocolo aberto utilizado para implementação SDN.
ICMP	N/A	Protocolo de rede definido pela IETF na RFC 792 utilizado comumente para a verificação de conectividade de camada de rede.
inSSIDer	2.0	Ferramenta de análise de Wi-Fi utilizada para realizar site survey.

O sistema operacional Windows 7 foi utilizado nos três *hosts* MN1, MD1 e N1. Sendo que nos *hosts* MN1 e MD1 foi utilizado como sistema final, onde os *softwares* iPerf e Wireshark foram instalados.

A ferramenta iPerf foi utilizada no *host* MN1 e no servidor MD1 para a geração do tráfego TCP/UDP com o intuito de medir os parâmetros de vazão e perda de pacotes, sendo que este último parâmetro também foi analisado através da geração de mensagens com o protocolo ICMP entre os *hosts*.

Já o Wireshark foi utilizado para a captura do tráfego gerado com o objetivo de mensurar a latência do *handover*, através da utilização das marcações de tempo dos pacotes recebidos pelo servidor MD1.

Já para o *host* N1, cuja função é virtualizar o controlador SDN, foi utilizado o *software* VMware Workstation Pro para a criação de um servidor virtual com o sistema operacional Ubuntu Server, que foi utilizado como hospedeiro para o sistema operacional SDN ONOS, implementado na modalidade *container* através do *software* Docker.

No *host* N1 também foi utilizado o *software* Wireshark para a captura dos pacotes de controle *OpenFlow* entre o controlador ONOS e o switch *OpenFlow*.

6.2 – ARQUITETURA E IMPLEMENTAÇÃO

A arquitetura geral, utilizada como base para a implementação dos cenários descritos, é composta por cinco redes principais, sendo seus elementos e respectivos objetivos:

- Rede Internet: servidor MD1, utilizada para o tráfego destinado a redes externas;
- Rede Cliente A: *host* MN1 e o *access point* AP-A, utilizada para o tráfego de dados de clientes;
- Rede Cliente B: *access point* AP-B, utilizada para o tráfego de dados de clientes;
- Rede de Gerência: controlador C1 e switch OFS1, utilizada para o tráfego de controle SDN e para a gerência do switch;
- Rede de Acesso Out-of-Band (OOB): Terminal remoto e controlador C1, utilizada para acesso remoto ao controlador SDN.

A topologia ilustrando as conexões, portas utilizadas e a disposição dos elementos descritos acima é apresentada na figura 6.1.

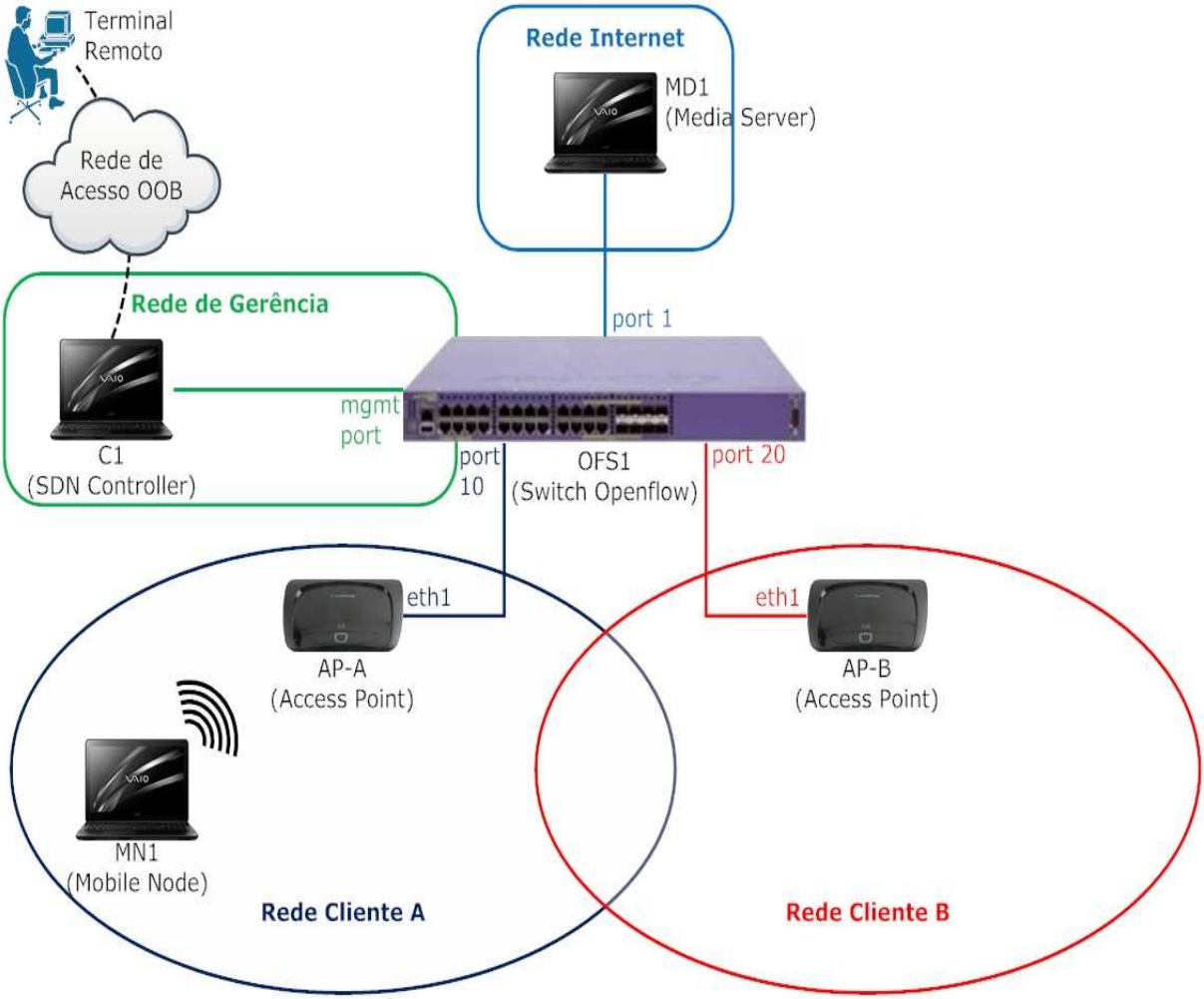


Figura 6.1 – *Topologia de implementação da proposta SDN-DMM.*

O plano de endereçamento IP utilizado é apresentado na tabela 6.3.

Tabela 6.3 – *Plano de Endereçamento IP.*

Rede Cliente A – 198.51.100.0/24 (Gateway 198.51.100.1)	
MN1	198.51.100.100
AP-A	198.51.100.50
Rede Cliente B – 192.0.2.0/24 (Gateway 192.0.2.1)	
AP-B	192.0.2.50
Rede Internet – 203.0.113.0/24 (Gateway 203.0.113.1)	
MD1	203.0.113.100
Rede de Gerência – 10.0.5.0/30	
C1	10.0.5.1
OFS1	10.0.5.2
Rede de Acesso OOB – 192.168.25.0/24	
Terminal Remoto	192.168.25.100
C1	192.168.25.118

As configurações para o ambiente apresentado na topologia e de implementação da proposta SDN-DMM são apresentadas nas seções seguintes e descritas em detalhes no apêndice A.

6.2.1 – Implementação do Ambiente de Rede

A criação das redes presentes na topologia da figura 6.1 foi realizada através do uso de VLANs no switch OFS1, onde a comunicação entre estas redes é realizada através do encaminhamento de pacotes diretamente pelo switch OFS1 entre as VLANs.

O endereçamento IP dos *hosts* MN1, MD1 e C1 e dos *access points*, apresentados na tabela 6.3, foram realizados de modo estático, ou seja, são mantidos os mesmos independente de qual rede estes *hosts* estejam conectados.

As configurações de rede e de endereçamento realizadas podem ser observadas no apêndice A.1.

6.2.2 – Implementação do Controlador SDN

O controlador utilizado para a implementação da proposta SDN-DMM foi o *Open Network Operating System* (ONOS), sendo implementado na modalidade *container* utilizando o *software* Docker no sistema operacional Ubuntu Server, com este virtualizado no *host* N1 através do *software* VMware Workstation Pro para Windows 7. A figura 6.2 ilustra a relação entre os *softwares*, sistemas e *hardware* utilizados para a implementação do controlador.

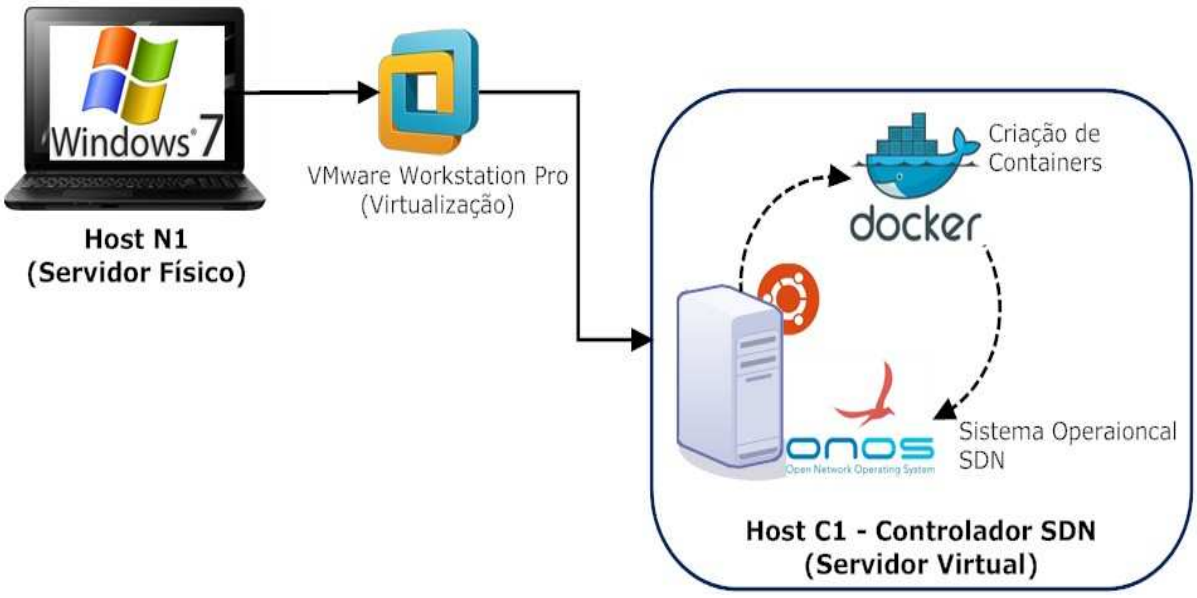


Figura 6.2 – *Relação de softwares, sistemas operacionais e hardware para o Controlador SDN.*

Através do *host* hospedeiro N1 utilizando o sistema operacional Windows 7, foi instalado o *software* para virtualização VMware Workstation Pro, onde se criou um servidor virtual (*host* C1) utilizando o sistema operacional Ubuntu Server, no qual foi instalado o *software* Docker para a execução de *containers*, implementando desta maneira o controlador ONOS na modalidade *container*.

Focando na implementação do controlador SDN, o *software* Docker utilizado, permite a criação e execução de *containers* que se assemelham às máquinas virtuais. A principal diferença entre o modelo de máquina virtual e o modelo de *container*, está relacionada ao compartilhamento do *kernel* com o sistema operacional.

No modelo de máquina virtual cada máquina possui seu próprio *kernel*, ou seja, temos a presença de vários *kernels* com vários módulos (rede, memória, tabela de partição de disco, serviço de escalonamento de processos) por máquina executada em cima do *Hypervisior* utilizado, consequentemente em cima do *hardware* do hospedeiro.

Já no modelo de *container*, os aplicativos executados compartilham o mesmo *kernel* e módulos, ou seja, ha uma redução da sobrecarga gerada em cima do sistema operacional do hospedeiro, consequentemente em cima do *hardware*. A figura 6.3 ilustra esta diferença.

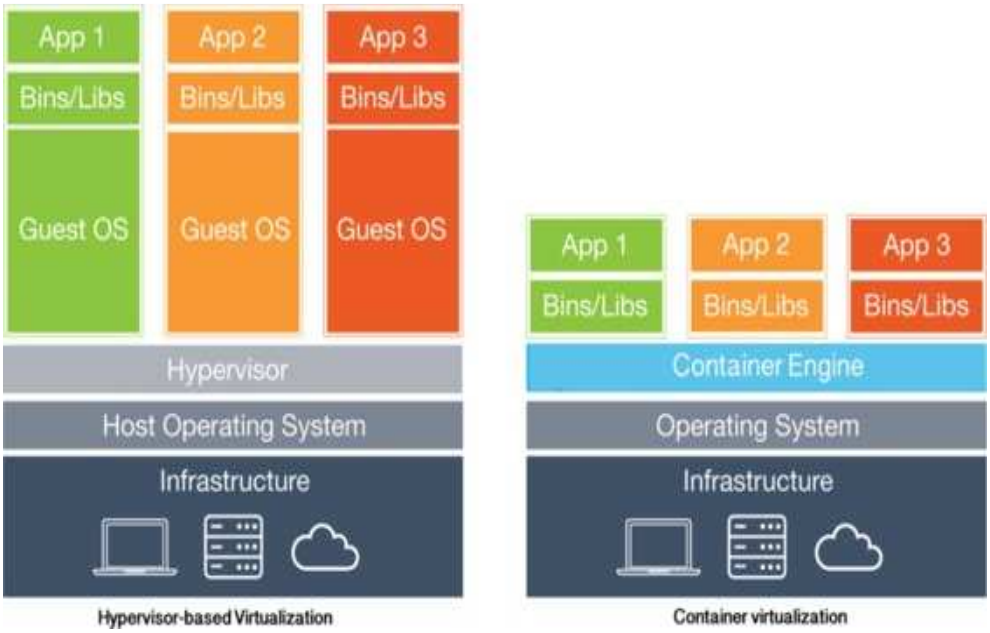


Figura 6.3 – *Diferença entre o modelo de máquina virtual e container.*

A partir da instalação do Docker é possível baixar do repositório Docker Hub o *container* oficial do controlador SDN ONOS criado pelo ONOS Project. O Docker Hub é um serviço de registro baseado em nuvem que permite armazenar e enviar imagens de *containers*, assim como o compartilhamento destes.

A comunicação entre o *host* local e os *containers* executados pelo Docker, é realizada através de uma conexão de rede lógica criada localmente entre eles. Ao se instalar o Docker e executar um *container*, um IP vinculado a esta conexão lógica é alocado ao *host* e ao *container* para permitir a comunicação local.

O acesso ao sistema operacional SDN ONOS executado no *host* C1 foi realizado de modo remoto através do redirecionamento de portas no *host* C1 e de uma rede de acesso *out-of-band*, ou seja, através de uma infraestrutura de rede não compartilhada com o cenário de implementação da proposta SDN-DMM.

Este tipo de arquitetura de acesso tem o objetivo de facilitar o acesso ao controlador na mesma medida em que evita interferências no ambiente e nos resultados, resultando em um acesso ao controlador como se este fosse realizado localmente pelo *host* C1.

A figura 6.4 ilustra a arquitetura utilizada para o acesso remoto ao controlador e a comunicação *OpenFlow* entre ele e o switch OFS1 através da rede *out-of-band* e do redirecionamento de portas.

No apêndice A.2 são apresentados em detalhes os comandos e configurações da implementação do controlador SDN.

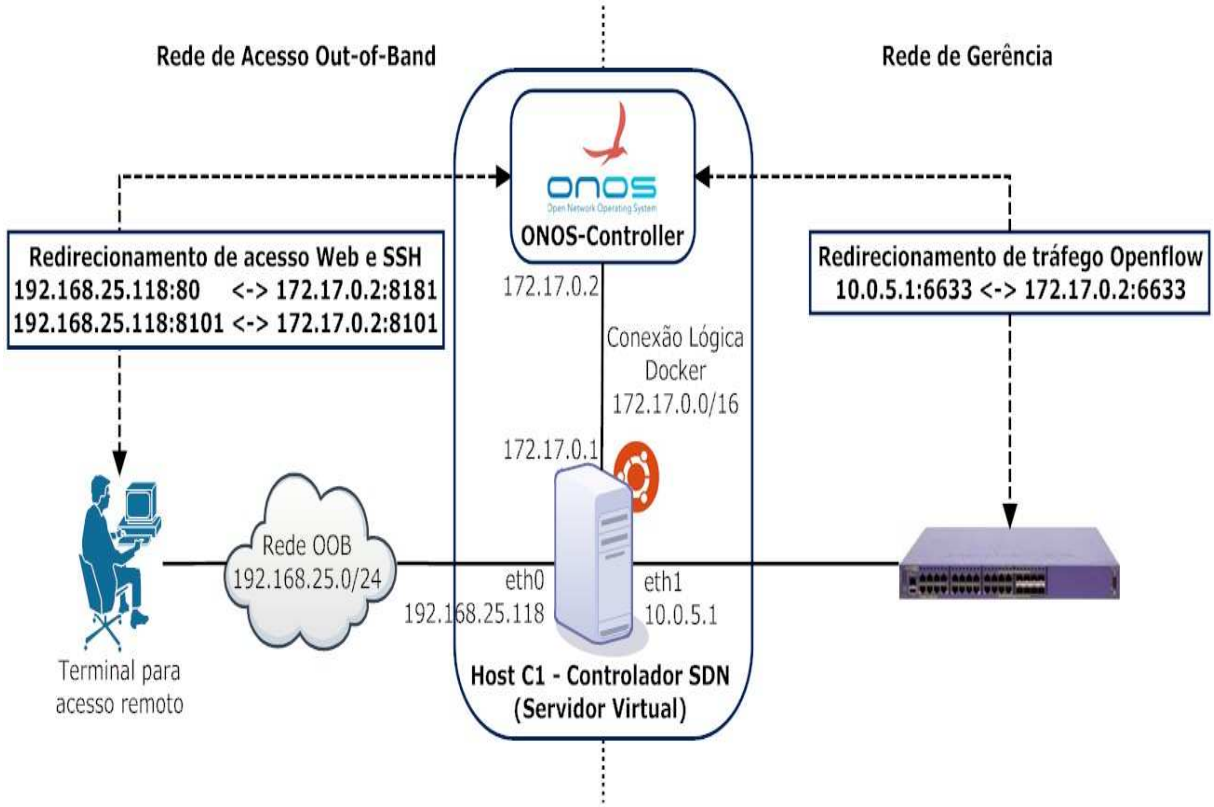


Figura 6.4 – *Topologia para acesso remoto ao container ONOS-Controller.*

6.2.3 – Implementação do protocolo OpenFlow

O protocolo *OpenFlow* foi utilizado como o protocolo da camada *southbound* na arquitetura SDN para realizar a comunicação entre o controlador C1 executando o *container* ONOS-Controller e o switch OFS1, com o objetivo de adaptar o plano de dados da rede através de regras *OpenFlow* manipulando os fluxos IP na medida em que o *host* MN1 executa o *handover* da sua rede de origem, Rede Cliente A, para a rede visitada, Rede Cliente B, enquanto mantém sessões ativas com o *host* MD1.

Para que isso ocorra, o switch OFS1 deve transferir a sua autonomia de decisão de encaminhamento de pacotes baseado no roteamento IP, ou seja, o seu tradicional plano de controle local, para o controlador C1, que passará a desempenhar a função de plano de controle da rede como um todo, tomando as decisões de encaminhamento de pacotes baseado em fluxos IP.

Deste modo, é necessário que o switch OFS1 se comunique através do protocolo *OpenFlow* com o controlador C1 e estabeleça a relação SDN de plano de controle e plano de dados da rede.

Para a configuração de implementação do *OpenFlow* no switch OFS1, primeiro é verificado o suporte ao protocolo no equipamento, confirmando se a imagem para o módulo *OpenFlow* está corretamente instalada, assim como as informações gerais do protocolo.

Depois é criada uma VLAN para ser utilizada como o plano de dados para o encaminhamento dos pacotes de acordo com os fluxos IP estabelecidos pelas regras *OpenFlow* recebidas do controlador C1.

Então o protocolo é ativado no switch OFS1, com a definição do modo de operação, do controlador primário responsável pelo plano de controle, no caso o controlador C1, e as ações locais para determinados tipos de fluxos caso não haja uma regra definida pelo controlador.

Nesse momento o switch OFS1 tenta estabelecer uma sessão com o controlador C1 para iniciar as trocas de mensagens *OpenFlow* para estabelecer a relação de plano de controle da rede. Onde a VLAN criada anteriormente para ser utilizada como o plano de dados, passa a encaminhar pacotes através dos fluxos IP definidos pelas regras *OpenFlow* e não mais baseado no roteamento tradicional de pacotes.

Neste momento o switch OFS1 toma as decisões de encaminhamento de pacotes baseando-se nos fluxos IP definidos pelo protocolo *OpenFlow*.

Primeiramente o switch verifica a correspondência de determinado fluxo IP com as entradas presentes na tabela chamada de ACL, tabela que contém todos os fluxos IP que foram definidos pelo controlador SDN através das regras *OpenFlow*.

Caso o switch não encontre uma correspondência para determinado fluxo IP com a tabela ACL, ele pode utilizar entradas configuradas localmente, chamadas de *default flows*, para tomar ações relacionadas ao encaminhamento do fluxo, onde quatro *default flows* foram implementadas no switch OFS1.

Para que a comunicação SDN ocorra efetivamente entre o switch OF1 e o controlador C1, é necessário também habilitar o protocolo *OpenFlow* para ser utilizado na camada *southbound* no controlador C1, indicando que este protocolo será utilizado para a comunicação de plano de controle na rede.

Logo após o *OpenFlow* ser ativado em ambos, é estabelecida uma sessão TCP e ocorre a troca de mensagens *OpenFlow* para divulgar as informações relativas a topologia, funcionalidades e recursos da rede SDN.

Depois de finalizada a troca inicial de mensagens *OpenFlow*, o controlador C1 reconhece o switch OFS1 e passa a ser o seu controlador primário, definindo a partir de então os fluxos IP que devem ser implementados no switch para o encaminhamento dos pacotes.

Neste momento a rede SDN está estabelecida utilizando o protocolo *OpenFlow* e o encaminhamento de pacotes é realizado através de fluxos IP bidirecionais e não mais por meio do roteamento tradicional de pacotes. Os detalhes da implementação *OpenFlow* estão apresentados no apêndice A.3.

6.2.4 – Implementação da mobilidade SDN-DMM por Intents

O princípio da mobilidade SDN-DMM através de *intents* pode ser definido em três partes:

1) Coleta das informações topológicas da infraestrutura de rede;
   Coleta das informações topológicas pelo controlador SDN através do protocolo *OpenFlow*.
2) Indicação da comunicação na camada superior SDN *northbound*;
   Definição da comunicação que deve ser estabelecida e mantida pela infraestrutura.
3) Criação e ajustes dos fluxos IP bidirecionais pela camada inferior SDN *southbound*.
   Definição das regras *OpenFlow* para tratamento da comunicação.

As configurações e ações da implementação da *intent* estão descritas no apêndice A.4.

6.3 – TESTES E RESULTADOS

Esta seção apresenta os testes realizados e os resultados obtidos na implementação real da proposta SDN-DMM com a análise das métricas de rede de taxa de vazão, perda de pacotes, latência RTT e latência de *handover* de acordo com o descrito na seção 6.1, sendo os cenários implementados:

- Cenário Roteamento Tradicional sem *handover*;
- Cenário SDN-DMM sem *handover*;
- Cenário SDN-DMM com *handover* por chaveamento manual;
- Cenário SDN-DMM com *handover wireless* automatizado.

Ao final da seção é apresentado de forma consolidada todos os resultados obtidos na implementação dos cenários, assim como a análise comparativa dos mesmos.

6.3.1 – Cenário Roteamento Tradicional sem Handover

Neste cenário a comunicação entre os *hosts* MN1 e MD1 foi implementada através do roteamento tradicional IP através da conexão direta dos *hosts* por cabos UTP ao switch OFS1 em suas redes de origem de acordo com a topologia de rede apresentada na figura 6.1 da seção 6.2. Não foi utilizado nenhum mecanismo de SDN ou cenário de mobilidade, tratando-se de uma comunicação realizada por meio do roteamento de pacotes entre as redes baseado na tabela de roteamento do switch OFS1.

O objetivo deste cenário é avaliar o desempenho da taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT, de modo a obter dados para realizar uma análise comparativa destas métricas com o ambiente SDN.

Com a comunicação devidamente estabelecida, o *software* Iperf3 foi utilizado para geração de tráfego UDP, de modo a avaliar a taxa de vazão e perda de pacotes, e para geração de tráfego TCP, de modo a avaliar a taxa de vazão máxima, durante o intervalo de 5 minutos. Sendo executado em modo cliente no *host* MN1 e em modo servidor no *host* MD1.

O tráfego UDP foi gerado a partir do *host* MN1 para o *host* MD1 a uma taxa de 100 Mbps de acordo com os seguintes parâmetros:

```
#Lado Cliente MN1: Geração de tráfego UDP a 100Mbps
iperf3 –c 203.0.113.100 –u –t 300 –b 100M –i 10

#Lado Servidor MD1
iperf3 –s –i 10
```

Onde o parâmetro *–s* indica o modo servidor, *–c 203.0.113.100* indica o modo cliente para conexão ao servidor com o IP especificado, *–u* para geração de tráfego UDP, *–t* para definição do tempo em segundos de geração de tráfego, *–b* sendo a taxa em que o tráfego será gerado, no caso em Mbps, e *–i* o intervalo de tempo para serem apresentados os reportes durante o andamento do teste. O resultado obtido é apresentado na figura 6.5.

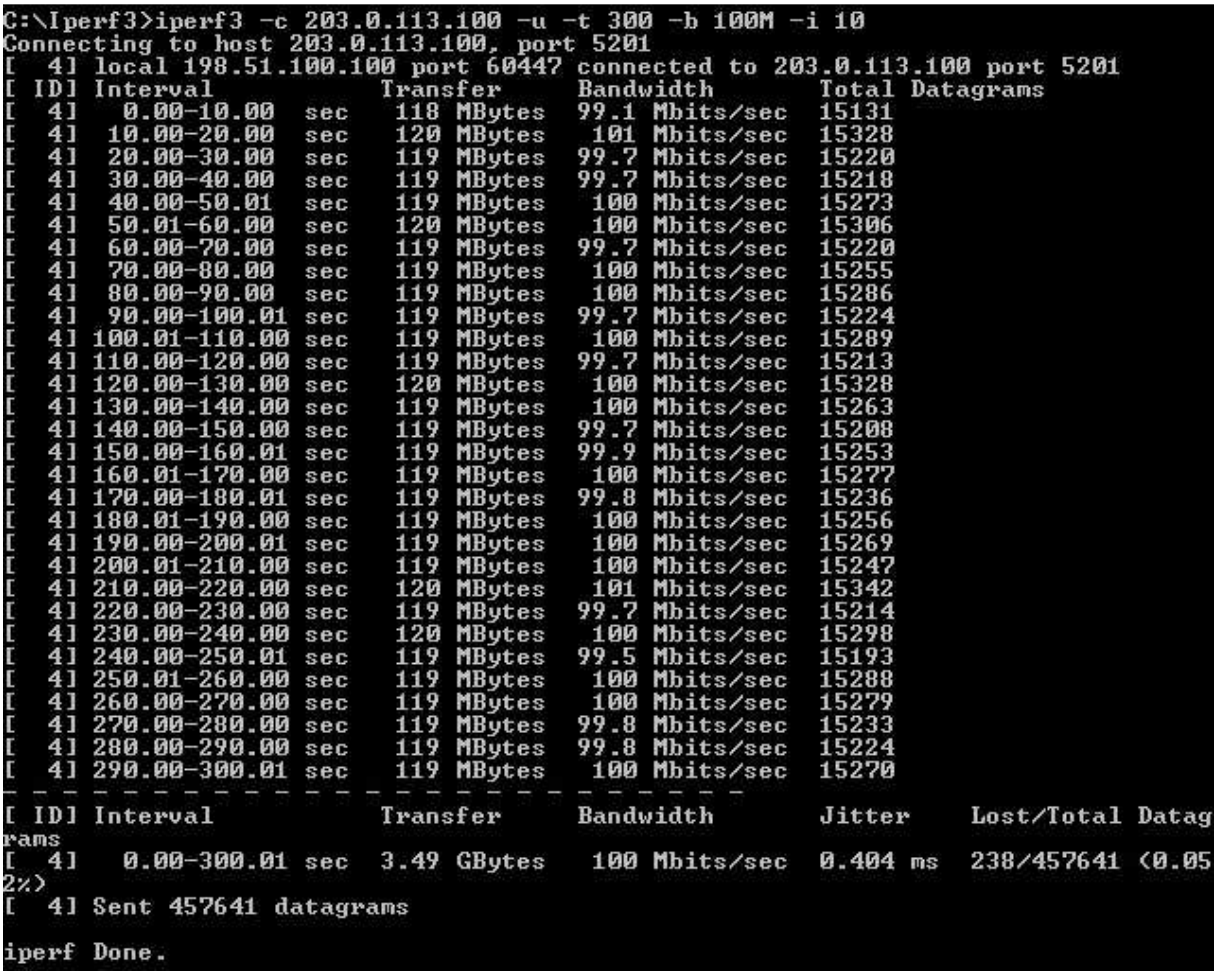


Figura 6.5 – *Teste de vazão e perda de pacotes UDP com Iperf3 para o roteamento tradicional.*

Observamos na figura 6.5 que a taxa média de vazão UDP obtida foi igual à definida no comando, de 100 Mbps, e que ocorreu uma perda de pacote de 0,052%.

Para o teste da vazão máxima TCP, foram utilizados os seguintes parâmetros:

```
#Lado Cliente MN1: Geração de tráfego TCP
iperf3 –c 203.0.113.100 –t 300 –i 10

#Lado Servidor MD1
iperf3 –s –i 10
```

Onde o parâmetro *–s* indica o modo servidor, *–c 203.0.113.100* indica o modo cliente para conexão ao servidor com o IP especificado, *–t* para definição do tempo em segundos de geração de tráfego e *–i* o intervalo de tempo para serem apresentados os reportes durante o

andamento do teste. No caso de geração de tráfego TCP, não é necessário indicar o tipo de tráfego a ser gerado, uma vez que este é o tráfego default do Iperf, não sendo possível também estabelecer a taxa de vazão em que se deseja gerar o tráfego, uma vez que o Iperf irá utilizar as características do TCP para obter a vazão máxima possível. O resultado obtido é apresentado na figura 6.6.

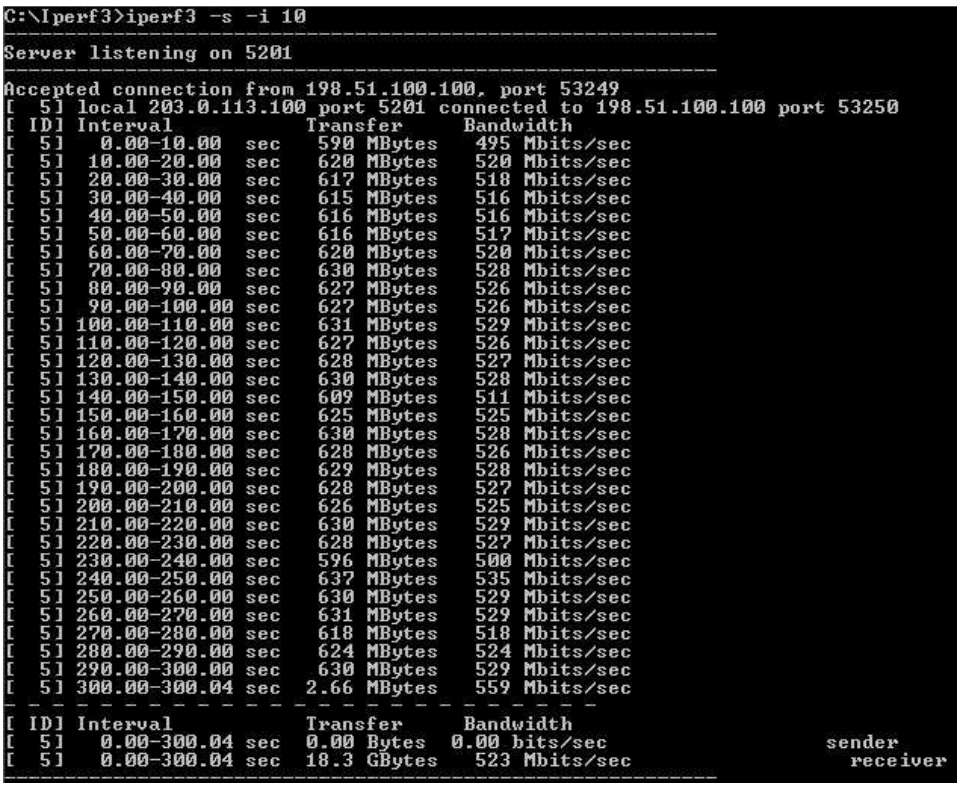


Figura 6.6 – *Teste de vazão máxima TCP com Iperf3 para o roteamento tradicional.*

Observamos na figura 6.6 que o teste de vazão máxima TCP para o roteamento tradicional obteve o resultado médio de 523 Mbps. Para a perda de pacotes ICMP e latência RTT, durante o mesmo intervalo de tempo utilizado nos testes com o Iperf3, obtivemos perda igual a 0% e latência inferior a 1 ms, no envio de pacotes ICMP do *host* MN1 para o *host* MD1, conforme apresentado na figura 6.7.

Figura 6.7 – *Perda de pacotes e latência RTT com ICMP para o roteamento tradicional.*

Os resultados obtidos para o cenário de roteamento tradicional são apresentados de forma consolidada na tabela 6.4.

Tabela 6.4 – *Resultados para o cenário de roteamento tradicional.*

Teste	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	100 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	523 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,052%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	<1 ms

6.3.2 – Cenário SDN-DMM sem Handover

Utilizando a abordagem SDN-DMM, neste cenário a comunicação entre os *host*s MN1 e MD1 foi implementada através de fluxos IP bidirecionais definidos pelo controlador C1 através do uso de *Intents*. Os *host*s foram conectados diretamente ao switch OFS1 por cabos UTP em suas redes de origem do mesmo modo que no cenário de roteamento tradicional, realizando os mesmos testes para obter os dados de taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT, a fim de se comparar o desempenho com o ambiente de roteamento tradicional.

Com o uso dos mesmos parâmetros utilizados no cenário de roteamento tradicional para o *software* Iperf3, a taxa de vazão e perda de pacotes UDP obtidas são apresentadas na figura 6.8.

```
C:\Iperf3>iperf3 -c 203.0.113.100 -u -t 300 -b 100M -i 10
Connecting to host 203.0.113.100, port 5201
[  4] local 198.51.100.100 port 52293 connected to 203.0.113.100 port 5201
[ ID] Interval           Transfer     Bandwidth       Total Datagrams
[  4]   0.00-10.00  sec   118 MBytes  99.2 Mbits/sec  15141
[  4]  10.00-20.00  sec   119 MBytes   100 Mbits/sec  15274
[  4]  20.00-30.00  sec   120 MBytes   100 Mbits/sec  15329
[  4]  30.00-40.00  sec   118 MBytes  99.3 Mbits/sec  15149
[  4]  40.00-50.00  sec   119 MBytes   100 Mbits/sec  15277
[  4]  50.00-60.00  sec   119 MBytes   100 Mbits/sec  15255
[  4]  60.00-70.00  sec   120 MBytes   100 Mbits/sec  15296
[  4]  70.00-80.00  sec   119 MBytes   100 Mbits/sec  15259
[  4]  80.00-90.00  sec   119 MBytes  99.9 Mbits/sec  15249
[  4]  90.00-100.01 sec   119 MBytes  99.8 Mbits/sec  15235
[  4] 100.01-110.00 sec   119 MBytes   100 Mbits/sec  15279
[  4] 110.00-120.01 sec   119 MBytes   100 Mbits/sec  15264
[  4] 120.01-130.00 sec   119 MBytes   100 Mbits/sec  15275
[  4] 130.00-140.00 sec   119 MBytes   100 Mbits/sec  15253
[  4] 140.00-150.00 sec   119 MBytes  99.9 Mbits/sec  15252
[  4] 150.00-160.00 sec   119 MBytes  99.7 Mbits/sec  15219
[  4] 160.00-170.00 sec   119 MBytes   100 Mbits/sec  15294
[  4] 170.00-180.00 sec   119 MBytes   100 Mbits/sec  15264
[  4] 180.00-190.00 sec   119 MBytes  99.9 Mbits/sec  15239
[  4] 190.00-200.00 sec   119 MBytes  99.9 Mbits/sec  15239
[  4] 200.00-210.00 sec   120 MBytes   100 Mbits/sec  15315
[  4] 210.00-220.00 sec   119 MBytes  99.7 Mbits/sec  15207
[  4] 220.00-230.00 sec   119 MBytes   100 Mbits/sec  15283
[  4] 230.00-240.00 sec   119 MBytes   100 Mbits/sec  15278
[  4] 240.00-250.00 sec   119 MBytes  99.9 Mbits/sec  15237
[  4] 250.00-260.00 sec   119 MBytes   100 Mbits/sec  15266
[  4] 260.00-270.01 sec   119 MBytes  99.8 Mbits/sec  15234
[  4] 270.01-280.00 sec   120 MBytes   101 Mbits/sec  15327
[  4] 280.00-290.00 sec   119 MBytes  99.8 Mbits/sec  15236
[  4] 290.00-300.01 sec   119 MBytes  99.8 Mbits/sec  15238
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total Datag
rams
[  4]   0.00-300.01 sec  3.49 GBytes   100 Mbits/sec  0.317 ms  492/457663 (0.11
%)
[  4] Sent 457663 datagrams

iperf Done.
```

Figura 6.8 – *Teste de vazão e perda de pacotes UDP com Iperf3 para SDN-DMM.*

Onde observamos na figura 6.8 que a taxa média de vazão UDP alcançada foi a mesma que a definida para o teste, de 100 Mbps, e que ocorreu uma perda de 0,11%. Para o teste da vazão máxima TCP foi obtido o resultado da figura 6.9.

```
C:\Iperf3>iperf3 -s -i 10
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 198.51.100.100, port 63212
[  5] local 203.0.113.100 port 5201 connected to 198.51.100.100 port 63213
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-10.00  sec   617 MBytes   518 Mbits/sec
[  5]  10.00-20.00  sec   618 MBytes   518 Mbits/sec
[  5]  20.00-30.00  sec   624 MBytes   523 Mbits/sec
[  5]  30.00-40.00  sec   621 MBytes   521 Mbits/sec
[  5]  40.00-50.00  sec   622 MBytes   522 Mbits/sec
[  5]  50.00-60.00  sec   623 MBytes   523 Mbits/sec
[  5]  60.00-70.00  sec   628 MBytes   527 Mbits/sec
[  5]  70.00-80.00  sec   628 MBytes   527 Mbits/sec
[  5]  80.00-90.00  sec   615 MBytes   515 Mbits/sec
[  5]  90.00-100.00 sec   625 MBytes   524 Mbits/sec
[  5] 100.00-110.00 sec   626 MBytes   525 Mbits/sec
[  5] 110.00-120.00 sec   626 MBytes   525 Mbits/sec
[  5] 120.00-130.00 sec   629 MBytes   528 Mbits/sec
[  5] 130.00-140.00 sec   616 MBytes   516 Mbits/sec
[  5] 140.00-150.00 sec   617 MBytes   517 Mbits/sec
[  5] 150.00-160.00 sec   630 MBytes   528 Mbits/sec
[  5] 160.00-170.00 sec   626 MBytes   525 Mbits/sec
[  5] 170.00-180.00 sec   630 MBytes   528 Mbits/sec
[  5] 180.00-190.00 sec   632 MBytes   530 Mbits/sec
[  5] 190.00-200.00 sec   630 MBytes   528 Mbits/sec
[  5] 200.00-210.00 sec   628 MBytes   527 Mbits/sec
[  5] 210.00-220.00 sec   629 MBytes   527 Mbits/sec
[  5] 220.00-230.00 sec   586 MBytes   492 Mbits/sec
[  5] 230.00-240.00 sec   554 MBytes   465 Mbits/sec
[  5] 240.00-250.00 sec   631 MBytes   530 Mbits/sec
[  5] 250.00-260.00 sec   628 MBytes   527 Mbits/sec
[  5] 260.00-270.00 sec   630 MBytes   528 Mbits/sec
[  5] 270.00-280.00 sec   627 MBytes   526 Mbits/sec
[  5] 280.00-290.00 sec   628 MBytes   527 Mbits/sec
[  5] 290.00-300.00 sec   631 MBytes   529 Mbits/sec
[  5] 300.00-300.03 sec  2.45 MBytes   686 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-300.03 sec  0.00 Bytes  0.00 bits/sec                  sender
[  5]   0.00-300.03 sec  18.2 GBytes   522 Mbits/sec                  receiver
-----------------------------------------------------------
```

Figura 6.9 – *Teste de vazão máxima TCP com Iperf3 para SDN-DMM.*

Observamos na figura 6.9 que a taxa média de vazão máxima TCP alcançada foi de 522 Mbps. Para a perda de pacotes e latência RTT com ICMP, obtivemos os resultados da figura 6.10, com perda de 0% e latência inferior a 1 ms.

```
C:\>ping 203.0.113.100 -t

Disparando 203.0.113.100 com 32 bytes de dados:
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
.
.
.
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128

Estatísticas do Ping para 203.0.113.100:
    Pacotes: Enviados = 326, Recebidos = 326, Perdidos = 0 (0% de
             perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 0ms, Máximo = 3ms, Média = 0ms
```

Figura 6.10 – *Perda de pacotes e latência RTT com ICMP para SDN-DMM.*

Os resultados obtidos para o cenário SDN-DMM são apresentados de forma consolidada na tabela 6.5.

Tabela 6.5 – *Resultados para o cenário SDN-DMM.*

Teste	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	100 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	522 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,11%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	<1 ms

6.3.3 – Cenário SDN-DMM com Handover por Chaveamento Manual

A partir da implementação do cenário SDN-DMM sem *handover* anterior, com os *host*s conectados diretamente ao switch OFS1, o *handover* do *host* MN1 foi realizado de forma manual através do seu chaveamento entre a porta do switch associada à rede A e a porta do switch associada à rede B durante a realização dos testes para coletar a taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT durante o *handover* do *host*.

Neste cenário também foi analisada a métrica de latência de *handover* através de pacotes TCP gerados pelo Iperf3 e através de pacotes ICMP. Para fins de melhor visualização dos resultados, para geração do tráfego UDP neste cenário foi utilizado a versão 2 do Iperf, que gera saídas resumidas no modo servidor sobre a perda e recebimento de pacotes fora de ordem em situações de *handover*.

Com o uso dos mesmos parâmetros utilizados anteriormente para o *software* Iperf3, mas agora utilizando a versão 2, a taxa de vazão e perda de pacotes UDP obtidas são apresentadas na figura 6.11.

Podemos observar na figura 6.11 que o *handover* é realizado no intervalo entre 50 a 60 segundos da execução do teste, onde ocorre perda de pacotes e uma redução da taxa de vazão momentaneamente, onde logo após a execução do *handover* voltam a normalidade. Obtendo ao final uma taxa média de vazão UDP de 99 Mbps, com uma perda de pacotes de 0,96%.

Para a taxa de vazão máxima TCP com o *handover* manual foi obtido o resultado apresentado na figura 6.12.

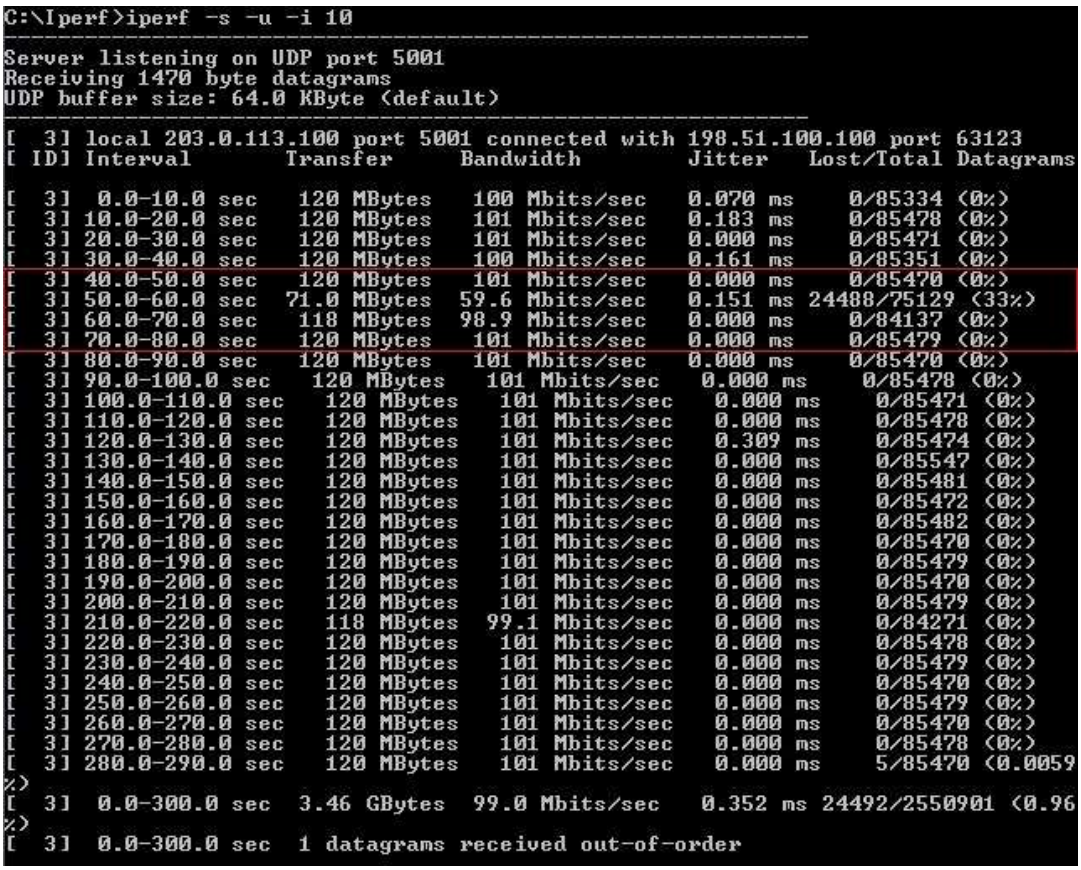


Figura 6.11 – *Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com handover manual por chaveamento.*

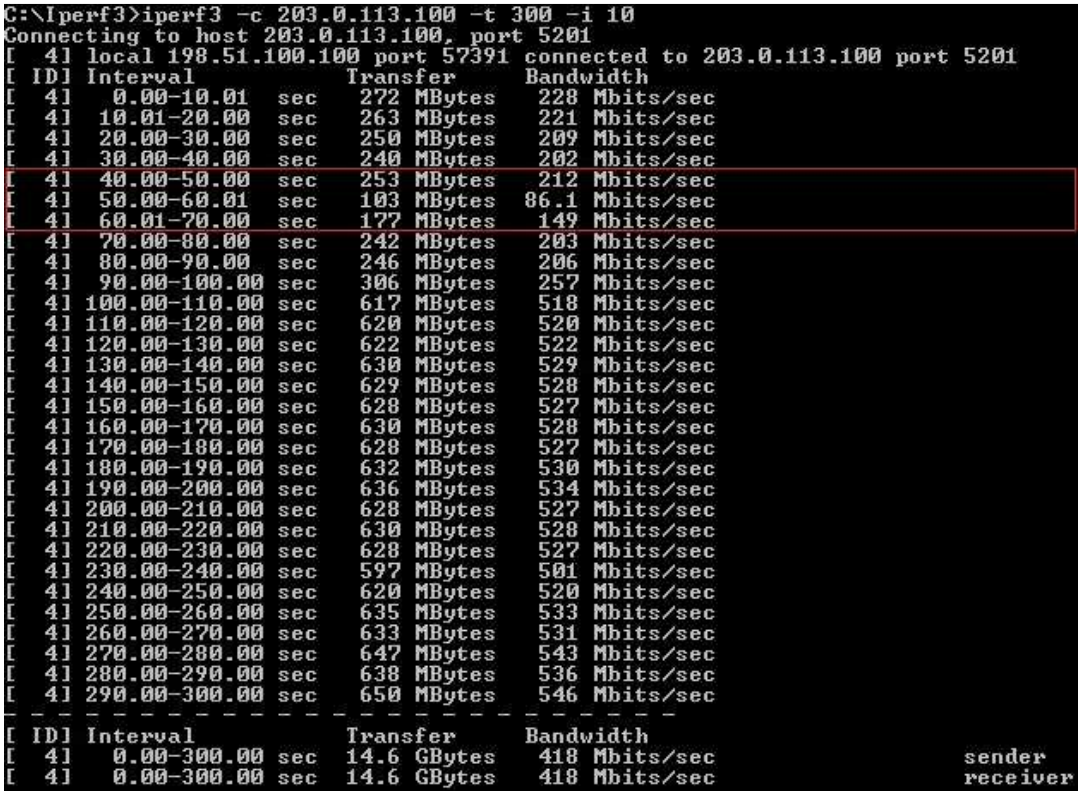


Figura 6.12 – *Teste de vazão máxima TCP com Iperf3 para SDN-DMM com handover manual por chaveamento.*

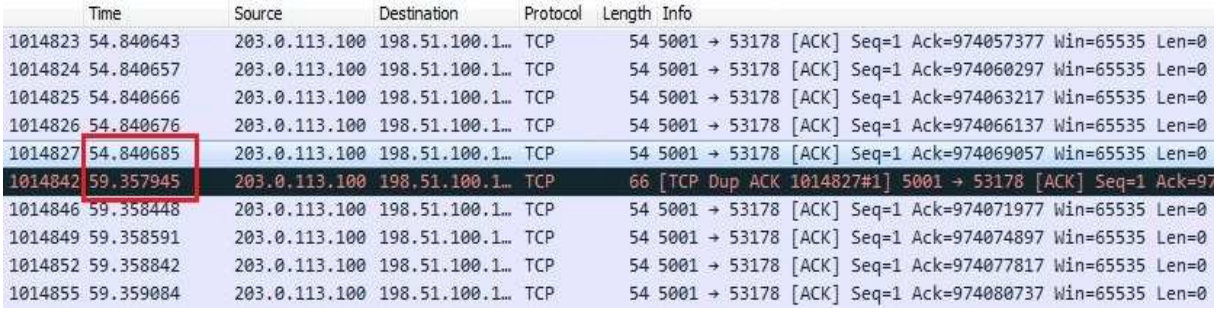
Na figura 6.12 observamos que o *handover* foi executado no intervalo entre 50 e 60 segundos, onde ocorreu momentaneamente a redução da taxa de vazão TCP, obtendo ao final a média de 418 Mbps. Para a perda de pacotes e latência RTT com ICMP, obtivemos uma perda de 1% e latência média inferior a 1 ms de acordo com a figura 6.13.

```
C:\>ping 203.0.113.100 -t

Disparando 203.0.113.100 com 32 bytes de dados:
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
.
.
.
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Falha geral.
Falha geral.
Esgotado o tempo limite do pedido.
Esgotado o tempo limite do pedido.
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
.
.
.
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128
Resposta de 203.0.113.100: bytes=32 tempo<1ms TTL=128

Estatísticas do Ping para 203.0.113.100:
    Pacotes: Enviados = 288, Recebidos = 284, Perdidos = 4 (1% de
             perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 0ms, Máximo = 1ms, Média = 0ms
```

Figura 6.13 – *Perda de pacotes e latência RTT com ICMP para SDN-DMM com handover manual por chaveamento.*

Observamos que o *handover* é executado no momento em que ocorre a mensagem de *Falha geral* na figura 6.13, devido à desconexão física do *host* e é finalizado em seguida a perda de pacotes indicada pela mensagem *Esgotado o tempo limite do pedido* ao receber respostas positivas do protocolo ICMP.

Para análise da latência de *handover*, durante a realização do teste de vazão TCP e ICMP, foram coletados no servidor os pacotes recebidos e comparados os intervalos entre o último pacote recebido ao *host* MN1 ser desconectado e o primeiro depois de executado o *handover*. O resultado para a latência de *handover* TCP pode ser observado na figura 6.14.

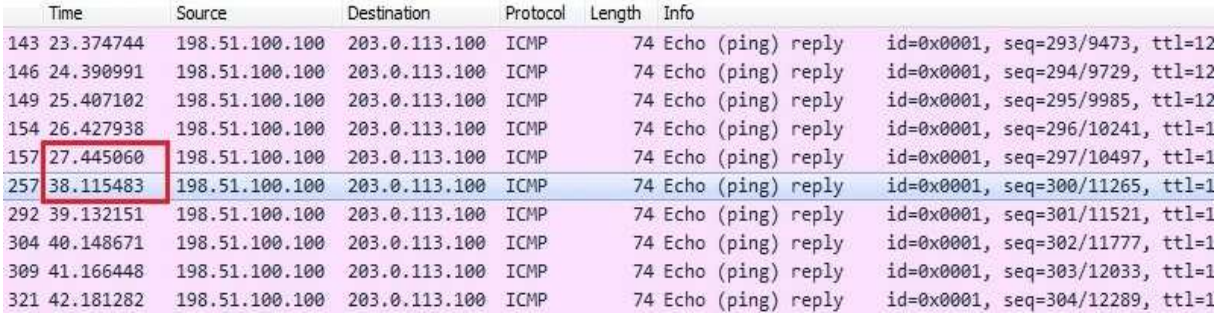


	Time	Source	Destination	Protocol	Length	Info
1014823	54.840643	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974057377 Win=65535 Len=0
1014824	54.840657	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974060297 Win=65535 Len=0
1014825	54.840666	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974063217 Win=65535 Len=0
1014826	54.840676	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974066137 Win=65535 Len=0
1014827	54.840685	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974069057 Win=65535 Len=0
1014842	59.357945	203.0.113.100	198.51.100.1…	TCP	66	[TCP Dup ACK 1014827#1] 5001 → 53178 [ACK] Seq=1 Ack=97
1014846	59.358448	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974071977 Win=65535 Len=0
1014849	59.358591	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974074897 Win=65535 Len=0
1014852	59.358842	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974077817 Win=65535 Len=0
1014855	59.359084	203.0.113.100	198.51.100.1…	TCP	54	5001 → 53178 [ACK] Seq=1 Ack=974080737 Win=65535 Len=0

Figura 6.14 – *Latência de handover TCP para SDN-DMM com handover manual por chaveamento.*

Observamos na figura 6.14 que houve uma latência de aproximadamente 4,517 segundos entre os pacotes TCP recebidos pelo MD1 durante o processo de *handover* executado pelo MN1.

A latência de *handover* ICMP pode ser observada na figura 6.15, onde obtivemos aproximadamente 10,67 segundos.



	Time	Source	Destination	Protocol	Length	Info
143	23.374744	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=293/9473, ttl=12
146	24.390991	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=294/9729, ttl=12
149	25.407102	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=295/9985, ttl=12
154	26.427938	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=296/10241, ttl=1
157	27.445060	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=297/10497, ttl=1
257	38.115483	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=300/11265, ttl=1
292	39.132151	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=301/11521, ttl=1
304	40.148671	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=302/11777, ttl=1
309	41.166448	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=303/12033, ttl=1
321	42.181282	198.51.100.100	203.0.113.100	ICMP	74	Echo (ping) reply id=0x0001, seq=304/12289, ttl=1

Figura 6.15 – *Latência de handover ICMP para SDN-DMM com handover manual por chaveamento.*

Os resultados obtidos para o cenário SDN-DMM com *handover* manual por chaveamento são apresentados de forma consolidada na tabela 6.6.

Tabela 6.6 – *Resultados para o cenário SDN-DMM com handover manual por chaveamento.*

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	99 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	418 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,96%
Perda de pacotes	Ping/ICMP	300 segundos	1%
Latência RTT	Ping/ICMP	300 segundos	<1 ms
Latência de *handover*	Iperf/TCP	300 segundos	4,517 segundos
Latência de *handover*	Ping/ICMP	300 segundos	10,67 segundos

6.3.4 – Cenário SDN-DMM com Handover Wireless Automatizado

Este cenário compreende a implementação da topologia da figura 6.1 da seção 6.2, onde estão presentes duas redes *wireless* as quais o *host* MN1 realizará um *handover* mantendo ativa a comunicação com o *host* MD1.

Para realizar o *handover* de forma automática sem necessitar de protocolos adicionais, os *access points* foram configurados com o mesmo SSID e senha de autenticação, utilizado a banda 2,4 GHz com o canal 6 para o *acess point* AP-A, responsável pela conectividade à rede A e o canal 11 para o *access point* B, responsável pela conectividade à rede B. Estes foram dispostos de acordo com a topologia de modo a existir uma sobreposição da áreas de cobertura que permita o *host* MN1 executar o *handover* adequadamente, selecionando o *access point* com o sinal mais intenso para dado momento. A ferramenta inSSIDer foi utilizada a analise *wireless* e *site survey*.

Por ser tratar de um ambiente sem fio, sujeito a interferências diversas, para obtermos resultados que permitam analisar de maneira consistente o impacto que a proposta SDN-DMM possui sobre as métricas de rede, os testes foram executados em três etapas:

1. Etapa 1 – MN1 conectado à Rede A
   Com o *host* MN1 conectado somente a rede A através do *access point* AP-A mantendo comunicação com o *host* MD1;
2. Etapa 2 – MN1 conectado à Rede B
   Com o *host* MN1 conectado somente a rede B através do *access point* AP-B mantendo comunicação com o *host* MD1;
3. Etapa 3 – *Handover* do MN1
   Com o *host* MN1 realizando o *handover* da rede A, *access point* AP-A, para a rede B, *access* point AP-B, mantendo comunicação com o *host* MD1.

A seguir são apresentados os resultados obtidos para cada uma das etapas.

Etapa 1 – MN1 conectado a Rede A

A figura 6.16 ilustra o estado da conexão Wi-Fi do *host* MN1 conectado ao *access point* AP-A a partir de onde os testes de taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT foram realizados.

O teste para a taxa de vazão e perda UDP foi realizado do mesmo modo que no cenário para o *handover* executado de forma manual, utilizado a versão 2 do Iperf e os mesmos parâmetros anteriores. Os resultados podem ser observados na figura 6.17.

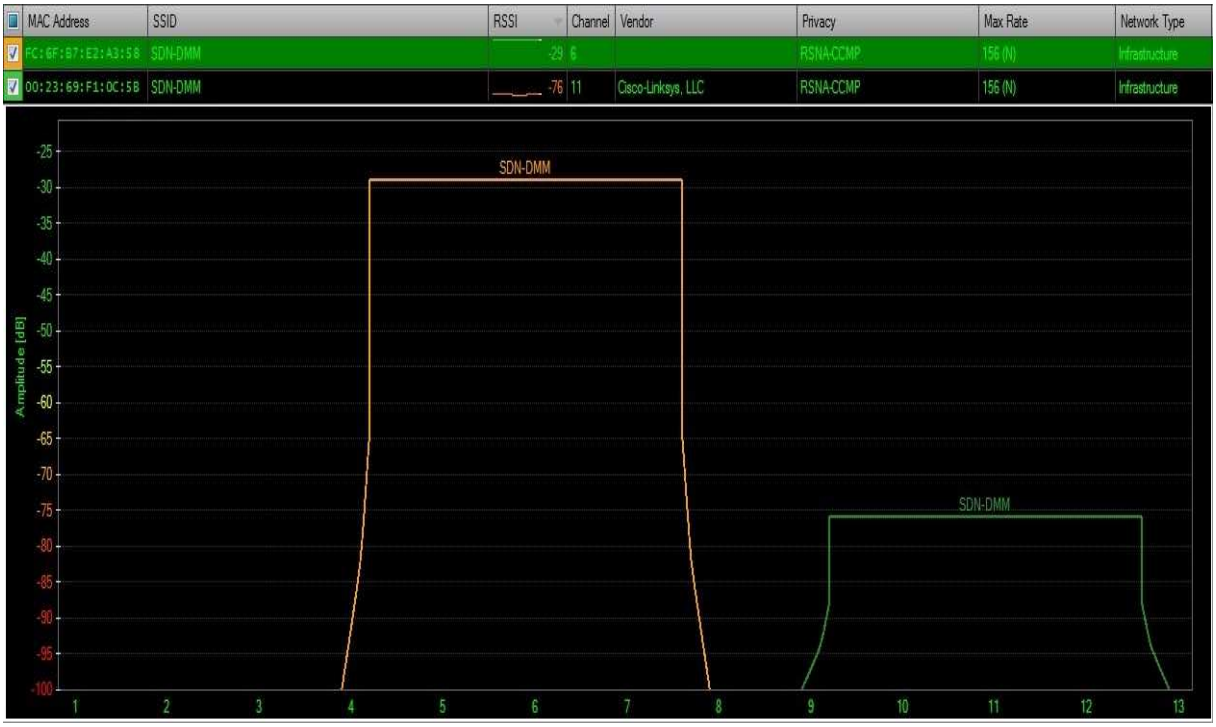


Figura 6.16 – *Conexão do host MN1 ao access-point AP-A na rede A.*

```
C:\Iperf>iperf -s -u -i 10
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 64.0 KByte (default)
------------------------------------------------------------
[  3] local 203.0.113.100 port 5001 connected with 192.51.100.100 port 57552
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0-10.0 sec   103 MBytes  86.4 Mbits/sec   0.641 ms    0/73470 (0%)
[  3] 10.0-20.0 sec   107 MBytes  89.5 Mbits/sec   0.399 ms    0/76089 (0%)
[  3] 20.0-30.0 sec   106 MBytes  88.8 Mbits/sec   0.627 ms    0/75486 (0%)
[  3] 30.0-40.0 sec   107 MBytes  90.0 Mbits/sec   0.518 ms    0/76517 (0%)
[  3] 40.0-50.0 sec   106 MBytes  88.9 Mbits/sec   0.708 ms    0/75607 (0%)
[  3] 50.0-60.0 sec   106 MBytes  89.1 Mbits/sec   0.536 ms    0/75774 (0%)
[  3] 60.0-70.0 sec   106 MBytes  89.0 Mbits/sec   1.062 ms    0/75689 (0%)
[  3] 70.0-80.0 sec   107 MBytes  89.7 Mbits/sec   0.008 ms    0/76235 (0%)
[  3] 80.0-90.0 sec   107 MBytes  89.6 Mbits/sec   0.174 ms    0/76216 (0%)
[  3] 90.0-100.0 sec   106 MBytes  88.9 Mbits/sec   1.065 ms    0/75555 (0%)
[  3] 100.0-110.0 sec   106 MBytes  89.1 Mbits/sec   0.108 ms    0/75742 (0%)
[  3] 110.0-120.0 sec   106 MBytes  89.0 Mbits/sec   0.027 ms    0/75640 (0%)
[  3] 120.0-130.0 sec   106 MBytes  88.6 Mbits/sec   0.026 ms    0/75315 (0%)
[  3] 130.0-140.0 sec   103 MBytes  86.8 Mbits/sec   0.002 ms    0/73823 (0%)
[  3] 140.0-150.0 sec   108 MBytes  90.3 Mbits/sec   0.028 ms    0/76825 (0%)
[  3] 150.0-160.0 sec   103 MBytes  86.3 Mbits/sec   0.015 ms    0/73382 (0%)
[  3] 160.0-170.0 sec   105 MBytes  87.9 Mbits/sec   0.011 ms    0/74732 (0%)
[  3] 170.0-180.0 sec   108 MBytes  91.0 Mbits/sec   0.016 ms    0/77351 (0%)
[  3] 180.0-190.0 sec   107 MBytes  89.5 Mbits/sec   0.014 ms    0/76083 (0%)
[  3] 190.0-200.0 sec   107 MBytes  90.2 Mbits/sec   0.005 ms    0/76671 (0%)
[  3] 200.0-210.0 sec   110 MBytes  92.3 Mbits/sec   0.064 ms    0/78455 (0%)
[  3] 210.0-220.0 sec   102 MBytes  85.9 Mbits/sec   0.007 ms    0/73018 (0%)
[  3] 220.0-230.0 sec   104 MBytes  87.2 Mbits/sec   0.588 ms    0/74170 (0%)
[  3] 230.0-240.0 sec   105 MBytes  88.3 Mbits/sec   0.046 ms    0/75121 (0%)
[  3] 240.0-250.0 sec   106 MBytes  88.8 Mbits/sec   0.069 ms    0/75489 (0%)
[  3] 250.0-260.0 sec   107 MBytes  89.9 Mbits/sec   0.086 ms    0/76420 (0%)
[  3] 260.0-270.0 sec   108 MBytes  90.4 Mbits/sec   0.126 ms    0/76842 (0%)
[  3] 270.0-280.0 sec   105 MBytes  88.4 Mbits/sec   0.091 ms    0/75162 (0%)
[  3] 280.0-290.0 sec   106 MBytes  89.0 Mbits/sec   0.244 ms    0/75706 (0%)
[  3] 290.0-300.0 sec   107 MBytes  90.0 Mbits/sec   0.080 ms    0/76513 (0%)
[  3]  0.0-300.0 sec  3.11 GBytes  88.9 Mbits/sec   1.569 ms    0/2269173 (0%)
[  3]  0.0-300.0 sec  1 datagrams received out-of-order
```

Figura 6.17 – *Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com MN1 conectado à rede A.*

Observamos na figura 6.17 que a taxa média de vazão UDP alcançada foi de 88,9 Mbps, inferior à definida para o teste de 100 Mbps e que não ocorreram perda de pacotes. Para o teste da vazão máxima TCP foi obtido o resultado da figura 6.18.

```
C:\Iperf3>iperf3 -c 203.0.113.100 -i 10 -t 300
Connecting to host 203.0.113.100, port 5201
[  4] local 198.51.100.100 port 60465 connected to 203.0.113.100 port 5201
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-10.00  sec   102 MBytes  85.6 Mbits/sec
[  4]  10.00-20.00  sec   102 MBytes  85.4 Mbits/sec
[  4]  20.00-30.00  sec   102 MBytes  86.0 Mbits/sec
[  4]  30.00-40.00  sec   101 MBytes  84.6 Mbits/sec
[  4]  40.00-50.00  sec   101 MBytes  84.4 Mbits/sec
[  4]  50.00-60.00  sec   102 MBytes  85.7 Mbits/sec
[  4]  60.00-70.00  sec   101 MBytes  84.5 Mbits/sec
[  4]  70.00-80.00  sec  99.2 MBytes  83.3 Mbits/sec
[  4]  80.00-90.00  sec  99.1 MBytes  83.2 Mbits/sec
[  4]  90.00-100.00 sec  99.4 MBytes  83.4 Mbits/sec
[  4] 100.00-110.00 sec   102 MBytes  85.7 Mbits/sec
[  4] 110.00-120.00 sec   101 MBytes  84.4 Mbits/sec
[  4] 120.00-130.00 sec   102 MBytes  85.5 Mbits/sec
[  4] 130.00-140.00 sec   101 MBytes  84.6 Mbits/sec
[  4] 140.00-150.00 sec   102 MBytes  85.8 Mbits/sec
[  4] 150.00-160.00 sec  95.9 MBytes  80.4 Mbits/sec
[  4] 160.00-170.00 sec   100 MBytes  84.0 Mbits/sec
[  4] 170.00-180.00 sec  98.5 MBytes  82.6 Mbits/sec
[  4] 180.00-190.00 sec  92.0 MBytes  77.2 Mbits/sec
[  4] 190.00-200.00 sec  99.6 MBytes  83.6 Mbits/sec
[  4] 200.00-210.00 sec   101 MBytes  84.7 Mbits/sec
[  4] 210.00-220.00 sec   100 MBytes  83.9 Mbits/sec
[  4] 220.00-230.00 sec  99.1 MBytes  83.2 Mbits/sec
[  4] 230.00-240.00 sec  98.6 MBytes  82.7 Mbits/sec
[  4] 240.00-250.00 sec   100 MBytes  84.2 Mbits/sec
[  4] 250.00-260.00 sec   101 MBytes  84.6 Mbits/sec
[  4] 260.00-270.00 sec   101 MBytes  84.5 Mbits/sec
[  4] 270.00-280.00 sec   100 MBytes  83.9 Mbits/sec
[  4] 280.00-290.00 sec   101 MBytes  84.4 Mbits/sec
[  4] 290.00-300.00 sec  99.6 MBytes  83.6 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-300.00 sec  2.93 GBytes  84.0 Mbits/sec                  sender
[  4]   0.00-300.00 sec  2.93 GBytes  84.0 Mbits/sec                  receiver
```

Figura 6.18 – *Teste de vazão máxima TCP com Iperf3 para SDN-DMM com MN1 na rede A.*

Observamos na figura 6.18 que a taxa média de vazão máxima TCP alcançada foi de 84 Mbps. Para a perda de pacotes e latência RTT com ICMP, obtivemos os resultados da figura 6.19, com perda de 0% e latência média de 2 ms.

```
C:\>ping 203.0.113.100 -t

Disparando 203.0.113.100 com 32 bytes de dados:
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=6ms TTL=64
.
.
.
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=4ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64

Estatísticas do Ping para 203.0.113.100:
    Pacotes: Enviados = 300, Recebidos = 300, Perdidos = 0 (0% de
             perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 1ms, Máximo = 14ms, Média = 2ms
```

Figura 6.19 – *Perda de pacotes e latência RTT com ICMP para SDN-DMM com MN1 na rede A.*

Os resultados obtidos para a conexão do *host* MN1 à rede A no cenário SDN-DMM com *handover* automático por *wireless* são apresentados de forma consolidada na tabela 6.7.

Tabela 6.7 – *Resultados para o cenário SDN-DMM com handover automático – MN1 conectado à rede A.*

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	88,9 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	84 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	2 ms

Etapa 2 – MN1 conectado a Rede B

A figura 6.20 ilustra o estado da conexão Wi-Fi do *host* MN1 conectado ao *access point* AP-B a partir de onde os testes de taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT foram realizados.

Figura 6.20 – *Conexão do host MN1 ao access-point AP-B na rede B.*

O teste para a taxa de vazão e perda UDP foi realizado do mesmo modo que no cenário para o *handover* executado de forma manual, utilizado a versão 2 do Iperf e os mesmos parâmetros anteriores. Os resultados podem ser observados na figura 6.21.

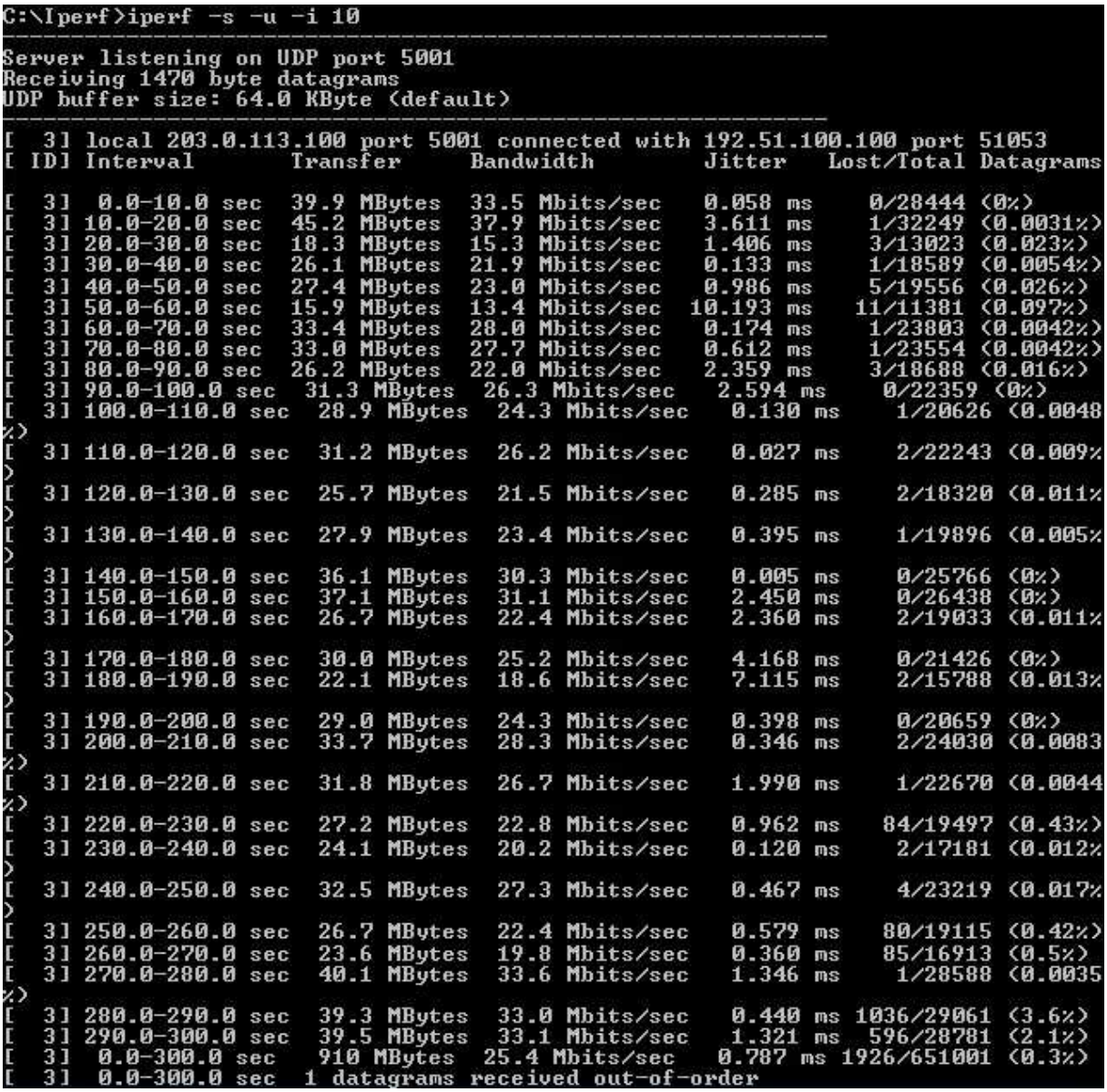


Figura 6.21 – *Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com MN1. conectado à rede B.*

Observamos na figura 6.21 que a taxa média de vazão UDP alcançada foi de 25,4 Mbps, inferior à definida para o teste de 100 Mbps e com uma perda de pacotes de 0,3%. Para o teste da vazão máxima TCP foi obtido o resultado da figura 6.22.

```
C:\Iperf3>iperf3 -c 203.0.113.100 -i 10 -t 300
Connecting to host 203.0.113.100, port 5201
[  4] local 198.51.100.100 port 61446 connected to 203.0.113.100 port 5201
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-10.00  sec  4.38 MBytes  3.67 Mbits/sec
[  4]  10.00-20.00  sec  8.12 MBytes  6.82 Mbits/sec
[  4]  20.00-30.00  sec  11.9 MBytes  9.96 Mbits/sec
[  4]  30.00-40.00  sec  9.12 MBytes  7.65 Mbits/sec
[  4]  40.00-50.00  sec  7.75 MBytes  6.50 Mbits/sec
[  4]  50.00-60.00  sec  11.9 MBytes  9.96 Mbits/sec
[  4]  60.00-70.00  sec  18.0 MBytes  15.1 Mbits/sec
[  4]  70.00-80.00  sec  16.2 MBytes  13.6 Mbits/sec
[  4]  80.00-90.00  sec  12.4 MBytes  10.4 Mbits/sec
[  4]  90.00-100.00 sec  20.4 MBytes  17.1 Mbits/sec
[  4] 100.00-110.00 sec  19.1 MBytes  16.0 Mbits/sec
[  4] 110.00-120.00 sec  19.1 MBytes  16.0 Mbits/sec
[  4] 120.00-130.00 sec  20.6 MBytes  17.3 Mbits/sec
[  4] 130.00-140.00 sec  14.9 MBytes  12.5 Mbits/sec
[  4] 140.00-150.00 sec  12.1 MBytes  10.2 Mbits/sec
[  4] 150.00-160.00 sec  8.75 MBytes  7.34 Mbits/sec
[  4] 160.00-170.00 sec  9.00 MBytes  7.55 Mbits/sec
[  4] 170.00-180.00 sec  9.75 MBytes  8.18 Mbits/sec
[  4] 180.00-190.00 sec  8.00 MBytes  6.71 Mbits/sec
[  4] 190.00-200.00 sec  14.4 MBytes  12.1 Mbits/sec
[  4] 200.00-210.00 sec  11.6 MBytes  9.75 Mbits/sec
[  4] 210.00-220.00 sec  11.2 MBytes  9.44 Mbits/sec
[  4] 220.00-230.00 sec  12.5 MBytes  10.5 Mbits/sec
[  4] 230.00-240.00 sec  10.5 MBytes  8.81 Mbits/sec
[  4] 240.00-250.00 sec  11.8 MBytes  9.86 Mbits/sec
[  4] 250.00-260.00 sec  11.6 MBytes  9.75 Mbits/sec
[  4] 260.00-270.00 sec  11.6 MBytes  9.75 Mbits/sec
[  4] 270.00-280.00 sec  10.0 MBytes  8.39 Mbits/sec
[  4] 280.00-290.00 sec  14.2 MBytes  12.0 Mbits/sec
[  4] 290.00-300.00 sec  11.0 MBytes  9.23 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-300.00 sec   372 MBytes  10.4 Mbits/sec                  sender
[  4]   0.00-300.00 sec   372 MBytes  10.4 Mbits/sec                  receiver
```

Figura 6.22 – *Teste de vazão máxima TCP com Iperf3 para SDN-DMM com MN1 conectado à rede B.*

Observamos na figura 6.22 que a taxa média de vazão máxima TCP alcançada foi de 10,4 Mbps. Para a perda de pacotes e latência RTT com ICMP, obtivemos os resultados da figura 6.23, com perda de 0% e latência média de 15 ms.

```
C:\>ping 203.0.113.100 -t

Disparando 203.0.113.100 com 32 bytes de dados:
Resposta de 203.0.113.100: bytes=32 tempo=34ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=20ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=5ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=27ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=16ms TTL=64
.
.
.
Resposta de 203.0.113.100: bytes=32 tempo=7ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=19ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=10ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=27ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=4ms TTL=64

Estatísticas do Ping para 203.0.113.100:
    Pacotes: Enviados = 300, Recebidos = 300, Perdidos = 0 (0% de
             perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 2ms, Máximo = 226ms, Média = 15ms
```

Figura 6.23 – *Perda de pacotes e latência RTT com ICMP para SDN-DMM com MN1 conectado à rede B.*

Os resultados obtidos para a conexão do *host* MN1 à rede B no cenário SDN-DMM com *handover* automático por *wireless* são apresentados de forma consolidada na tabela 6.8.

Tabela 6.8 – *Resultados para o cenário SDN-DMM com handover automático – MN1 conectado à rede B.*

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	25,4 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	10,4 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,3%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	15 ms

Etapa 3 – *Handover* do MN1

A figura 6.24 ilustra o estado da conexão Wi-Fi do *host* MN1 durante o seu *handover* da rede A para a rede B, ilustrando o aumento da intensidade de sinal em relação ao tempo do *access point* AP-B em relação ao *access point* AP-A durante a sua movimentação. O *handover* foi executado com os testes de taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT em andamento.

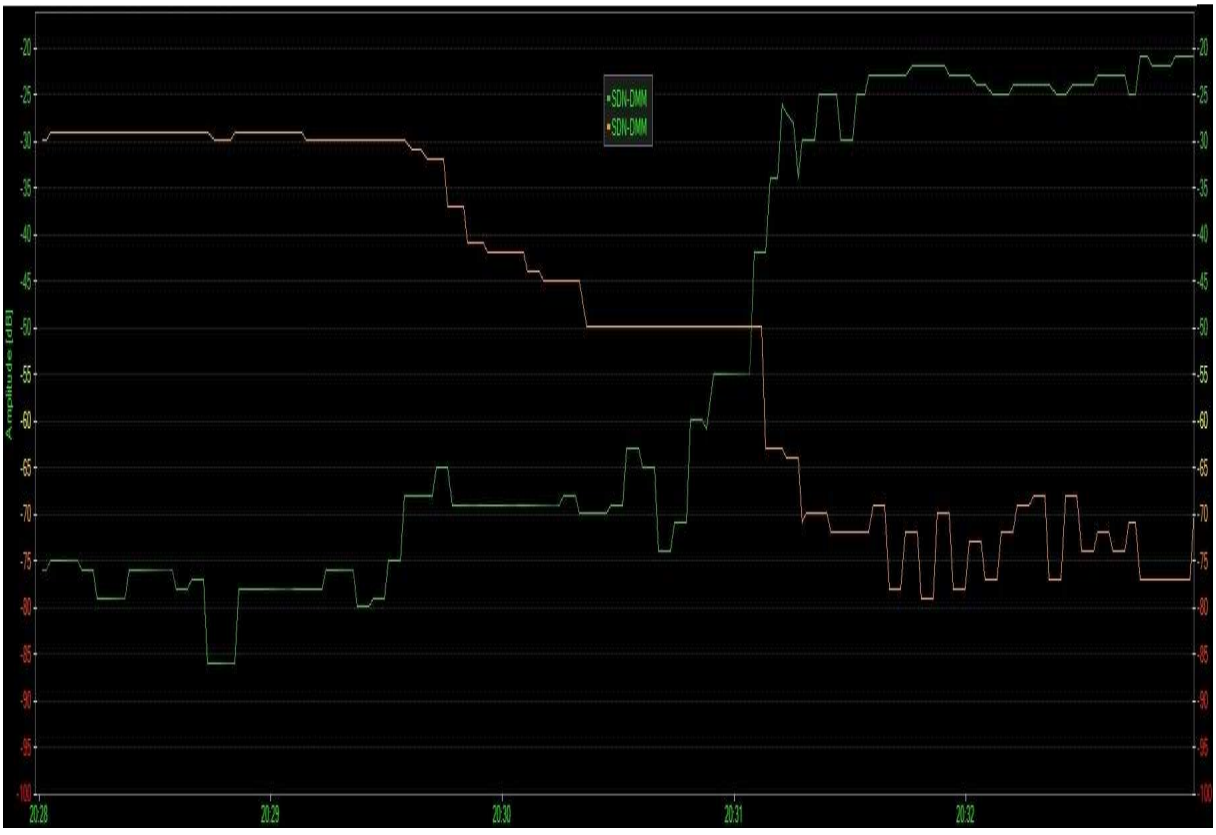


Figura 6.24 – *Movimentação do host MN1 entre as redes A e B durante o handover.*

Podemos observar na figura 6.24, que apresenta no eixo das abscissas a relação crescente de tempo, que durante a movimentação do MN1 a intensidade do sinal do AP-A diminui, na medida em que o sinal do AP-B aumenta. Quando o sinal do AP-B fica suficientemente maior do que o AP-A, o MN1 realiza o *handover* para a rede B.

Com o uso dos mesmos *softwares* e parâmetros utilizados anteriormente, os resultados dos testes da taxa de vazão e perda de pacotes UDP, iniciado antes do processo de *handover* e finalizado após, são apresentadas na figura 6.25.

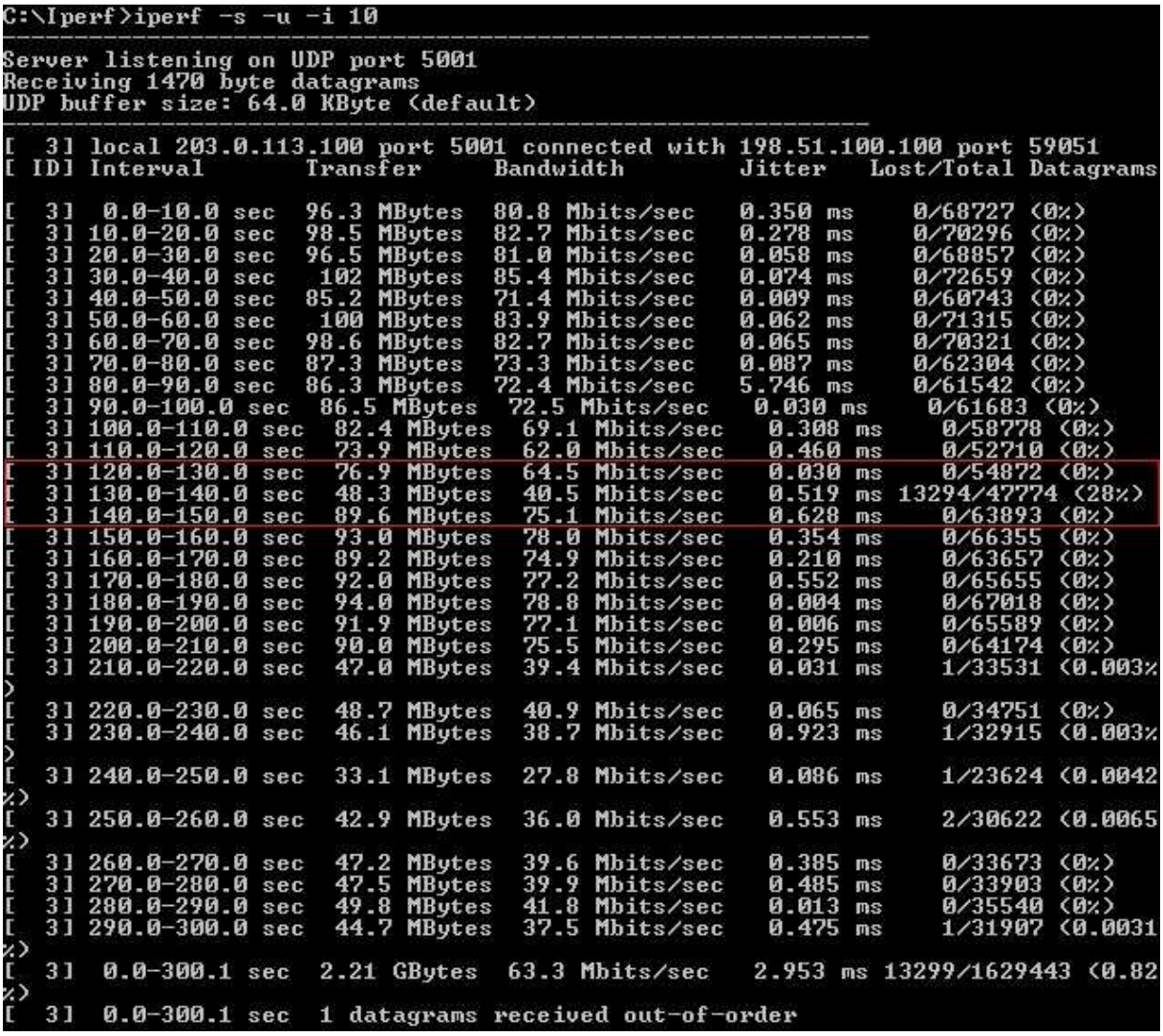


Figura 6.25 – *Teste de vazão e perda de pacotes UDP com Iperf2 para SDN-DMM com handover automático por wireless.*

Podemos observar na figura 6.25 que o *handover* é realizado no intervalo entre 130 a 140 segundos da execução do teste, onde ocorre perda de pacotes e uma redução da taxa de vazão, que volta a normalidade para as estatísticas obtidas para a rede B. Onde ao final foi obtida uma taxa média de vazão UDP de 63,3 Mbps, com uma perda de pacotes de 0,82%.

Para a taxa de vazão máxima TCP com o *handover* manual foi obtido o resultado apresentado na figura 6.26.

```
C:\Iperf3>iperf3 -c 198.51.100.100 -i 10 -t 300
Connecting to host 198.51.100.100, port 5201
[  4] local 203.0.113.100 port 53928 connected to 198.51.100.100 port 5201
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-10.01  sec  95.1 MBytes  79.7 Mbits/sec
[  4]  10.01-20.00  sec  80.0 MBytes  67.2 Mbits/sec
[  4]  20.00-30.00  sec  70.4 MBytes  59.0 Mbits/sec
[  4]  30.00-40.00  sec  76.5 MBytes  64.2 Mbits/sec
[  4]  40.00-50.00  sec  75.9 MBytes  63.6 Mbits/sec
[  4]  50.00-60.01  sec  77.2 MBytes  64.8 Mbits/sec
[  4]  60.01-70.01  sec  68.8 MBytes  57.6 Mbits/sec
[  4]  70.01-80.01  sec  14.6 MBytes  12.3 Mbits/sec
[  4]  80.01-90.00  sec  18.2 MBytes  15.3 Mbits/sec
[  4]  90.00-100.00 sec  35.9 MBytes  30.1 Mbits/sec
[  4] 100.00-110.00 sec  25.9 MBytes  21.7 Mbits/sec
[  4] 110.00-120.01 sec  26.6 MBytes  22.3 Mbits/sec
[  4] 120.01-130.00 sec  34.8 MBytes  29.2 Mbits/sec
[  4] 130.00-140.01 sec  29.6 MBytes  24.8 Mbits/sec
[  4] 140.01-150.00 sec  28.2 MBytes  23.7 Mbits/sec
[  4] 150.00-160.00 sec  32.8 MBytes  27.5 Mbits/sec
[  4] 160.00-170.01 sec  31.2 MBytes  26.2 Mbits/sec
[  4] 170.01-180.01 sec  27.2 MBytes  22.8 Mbits/sec
[  4] 180.01-190.00 sec  26.9 MBytes  22.6 Mbits/sec
[  4] 190.00-200.01 sec  32.5 MBytes  27.2 Mbits/sec
[  4] 200.01-210.00 sec  41.1 MBytes  34.5 Mbits/sec
[  4] 210.00-220.01 sec  43.5 MBytes  36.5 Mbits/sec
[  4] 220.01-230.01 sec  20.1 MBytes  16.9 Mbits/sec
[  4] 230.01-240.01 sec  33.0 MBytes  27.7 Mbits/sec
[  4] 240.01-250.01 sec  34.1 MBytes  28.6 Mbits/sec
[  4] 250.01-260.01 sec  38.0 MBytes  31.9 Mbits/sec
[  4] 260.01-270.01 sec  37.9 MBytes  31.8 Mbits/sec
[  4] 270.01-280.00 sec  29.0 MBytes  24.4 Mbits/sec
[  4] 280.00-290.00 sec  29.5 MBytes  24.7 Mbits/sec
[  4] 290.00-300.02 sec  31.2 MBytes  26.2 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  4]   0.00-300.02 sec  1.22 GBytes  34.8 Mbits/sec                  sender
[  4]   0.00-300.02 sec  1.22 GBytes  34.8 Mbits/sec                  receiver
```

Figura 6.26 – *Teste de vazão máxima TCP com Iperf3 para SDN-DMM com handover automático por wireless.*

```
C:\>ping 203.0.113.100 -t

Disparando 203.0.113.100 com 32 bytes de dados:
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
.
.
.
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=3ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=5ms TTL=64
Esgotado o tempo limite do pedido.
Esgotado o tempo limite do pedido.
Resposta de 203.0.113.100: bytes=32 tempo=17ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=19ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=12ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=4ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
.
.
.
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=30ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=14ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=24ms TTL=64
Resposta de 203.0.113.100: bytes=32 tempo=2ms TTL=64

Estatísticas do Ping para 203.0.113.100:
    Pacotes: Enviados = 292, Recebidos = 290, Perdidos = 2 (0% de
             perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 2ms, Máximo = 397ms, Média = 10ms
```

Figura 6.27 – *Perda de pacotes e latência RTT com ICMP para SDN-DMM com handover automático por wireless.*

Na figura 6.26 observamos que o *handover* foi executado no intervalo entre 70 e 80 segundos, onde ocorreu momentaneamente a redução da taxa de vazão TCP e que logo após é mantida uma taxa média de acordo com o esperado para a rede B. Obtendo ao final a média de 34,8 Mbps.

Para a perda de pacotes e latência RTT com ICMP, obtivemos uma perda de 0% e latência média de 10 ms de acordo com a figura 6.27. Observamos que o *handover* é executado no momento em que ocorre a mensagem de *Esgotado o tempo limite do pedido* e que é finalizado logo após receber respostas positivas do protocolo ICMP.

Para análise da latência de *handover*, durante a realização do teste de vazão TCP e ICMP, foram coletados no servidor os pacotes recebidos e comparados os intervalos entre o último pacote recebido antes do *host* MN1 realizar o *handover* da rede A e o primeiro pacote recebido logo após o MN1 possuir conectividade na rede B. O resultado para a latência de *handover* TCP pode ser observado na figura 6.28.

	Time	Source	Destination	Protocol	Length	Info
603072	74.270288	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585095554 Ack=1 Win=212992 Le
603073	74.270565	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585097014 Ack=1 Win=212992 Le
603074	74.270611	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585098474 Ack=1 Win=212992 Le
603075	74.270630	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585099934 Ack=1 Win=212992 Le
603076	74.270651	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585101394 Ack=1 Win=212992 Le
603088	78.844312	203.0.113.100	198.51.100.100	TCP	1514	[TCP Spurious Retransmission] 53928 → 5201 [ACK] Seq
603090	78.962652	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585102854 Ack=1 Win=212992 Le
603091	78.968679	203.0.113.100	198.51.100.100	TCP	1186	53928 → 5201 [PSH, ACK] Seq=585104314 Ack=1 Win=2129
603092	78.968887	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585105446 Ack=1 Win=212992 Le
603093	78.968967	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585106906 Ack=1 Win=212992 Le
603094	78.969061	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585108366 Ack=1 Win=212992 Le
603068	74.268228	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585092634 Ack=1 Win=212992 Le
603069	74.268246	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq=585094094 Ack=1 Win=212992 Le

Figura 6.28 – *Latência de handover TCP para SDN-DMM com handover automático por wireless.*

Observamos na figura 6.28 que houve uma latência de aproximadamente 4,573 segundos entre os pacotes TCP recebidos pelo MD1 durante o processo de *handover* executado pelo MN1.

A latência de *handover* ICMP pode ser observada na figura 6.29, onde obtivemos aproximadamente 5,893 segundos.

	Time	Source	Destination	Protocol	Length	Info
386	159.426374	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1823/7943, ttl=128 (reply in 387)
388	160.416258	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1824/8199, ttl=128 (reply in 389)
392	161.416328	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1825/8455, ttl=128 (reply in 393)
394	162.417606	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1826/8711, ttl=128 (reply in 395)
396	163.433690	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1827/8967, ttl=128 (reply in 397)
402	169.327373	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1829/9479, ttl=128 (reply in 403)
404	170.326571	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1830/9735, ttl=128 (reply in 405)
408	171.329384	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1831/9991, ttl=128 (reply in 409)
410	172.328496	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1832/10247, ttl=128 (reply in 411)
412	173.326193	192.51.100.100	203.0.113.100	ICMP	74	Echo (ping) request id=0x0001, seq=1833/10503, ttl=128 (reply in 413)

Figura 6.29 – *Latência de handover ICMP para SDN-DMM com handover automático por wireless.*

Os resultados obtidos para o cenário SDN-DMM com *handover* automático por wireless são apresentados de forma consolidada na tabela 6.9.

Tabela 6.9 – *Resultados para o cenário SDN-DMM com handover automático por wireless.*

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	63,3 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	34,8 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,82%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	10 ms
Latência de *handover*	Iperf/TCP	300 segundos	4,573 segundos
Latência de *handover*	Ping/ICMP	300 segundos	5,893 segundos

6.3.5 – Análise dos Resultados

Verificamos que o ambiente SDN não causa impactos negativos para as métricas de rede em comparação com o ambiente de roteamento tradicional. Onde os resultados obtidos demonstram o mesmo desempenho alcançado para o encaminhamento de pacotes baseado no endereço de destino IP e para o encaminhamento de pacotes baseado no fluxo IP bidirecional utilizado na proposta SDN-DMM. A tabela 6.10 apresenta os resultados destes dois cenários.

Tabela 6.10 – *Comparação das métricas entre Roteamento Tradicional e SDN-DMM.*

Métrica	Roteamento Tradicional	SDN-DMM
Vazão UDP	100 Mbps	100 Mbps
Vazão máxima TCP	523 Mbps	522 Mbps
Perda de pacotes UDP	0,052%	0,11%
Perda de pacotes ICMP	0%	0%
Latência RTT	<1 ms	<1 ms

Analisando em conjunto a tabela 6.10, a tabela 6.11 e os resultados obtidos nos experimentos realizados, observamos que a proposta SDN-DMM também não acarreta em perda de desempenho quando o *mobile node* está conectado a uma rede estrangeira. Ao *mobile node* executar um *handover* para uma nova rede, os valores obtidos para as métricas

analisadas são os mesmos para um *host* nativo àquela rede. Isso pode ser observado nos testes realizados nos cenários de *handover* com o SDN-DMM e na tabela 6.11.

Tabela 6.11 – *Métricas SDN-DMM com handover manual e automático.*

Métrica	SDN-DMM com *handover* manual por chaveamento	SDN-DMM com *handover* automático por *wireless*
Vazão UDP	99 Mbps	63,3 Mbps
Vazão máxima TCP	418 Mbps	34,8 Mbps
Perda de pacotes (UDP)	0,96%	0,82%
Perda de pacotes (ICMP)	1%	0%
Latência RTT	<1 ms	10 ms
Latência de *handover* (TCP)	4,517 segundos	4,573 segundos
Latência de *handover* (ICMP)	10,67 segundos	5,893 segundos

Os valores próximos obtidos para a métrica de latência de *handover* (TCP) para o cenário de *handover* manual por chaveamento e para o cenário de *handover* automático por *wireless*, são devidos ao balanceamento entre os fatores que prejudicam a medição dos mesmos em cada cenários.

No cenário de *handover* manual, à desconexão manual do MN1 no chaveamento entre as portas do switch OF1 causa uma interrupção abrupta da comunicação e do estado da porta nos equipamentos, que ao serem reconectados, é necessário aguardar um intervalo de tempo para que o estado da porta fique ativo, interferindo assim na medição do tempo de latência de *handover* devido ao processo da abordagem SDN-DMM.

Já para o cenário de *handover* automático, o fator da interferência do ambiente *wireless* sem utilizar algum protocolo específico para *handover* entre *access point*, é o ponto que prejudica a medição da latência de *handover* da proposta, embora neste caso a transição entre as redes seja mais suave.

6.4 – CONSIDERAÇÕES FINAIS

A implementação da proposta SDN-DMM em um ambiente real de experimentação e os testes realizados demonstram que a abordagem utilizada para o gerenciamento de mobilidade distribuído baseada em uma arquitetura SDN atende de maneira satisfatória aos requisitos de

comunicação e não acarreta em prejuízos as métricas de rede quando comparado ao cenário tradicional de roteamento.

O desempenho obtido para a taxa de vazão quando o *mobile node* encontra-se em mobilidade vai de encontro ao custo de entrega de pacotes analisado na modelagem analítica do capítulo 5, demonstrando que a proposta sempre emprega a entrega de forma otimizada, possuindo então um menor custo em relação as outras abordagens analisadas neste trabalho.

A perda de pacotes e a latência de *handover* obtidas nos testem realizados podem ser considerados coerentes com os resultados obtidos com a modelagem analítica realizada anteriormente, considerando a baixa quantidade de mensagens necessárias no plano de controle para prover a mobilidade, reajustando o plano de dados.

7 – CONCLUSÕES E TRABALHOS FUTUROS

7.1 – CONCLUSÕES

Considerando um cenário de crescimento exponencial do número de dispositivos móveis e de crescente disseminação das redes heterogêneas, o presente trabalho focou o gerenciamento de mobilidade distribuído e a elaboração e avaliação comparativa de uma abordagem de DMM baseada no paradigma de redes definidas por *software*.

A partir dos resultados obtidos na análise de desempenho realizada, observamos que a proposta SDN-DMM lida de maneira satisfatória com os principais desafios de gerenciamento de mobilidade para o cenário considerado.

Adotou-se uma abordagem *network-based partially* DMM associada à arquitetura SDN, buscando incorporar à solução os benefícios intrínsecos deste paradigma; verificou-se que a proposta permite, de modo transparente ao *mobile node* e às redes de acesso heterogêneas, o uso eficiente e escalável dos recursos disponíveis da infraestrutura de rede de comunicação, tanto do ponto de vista operacional (OPEX), quanto do ponto de vista de investimento e evolutivo (CAPEX).

A escalabilidade para o plano de dados é obtida na medida em que não existe a dependência ou sobrecarga de um nó central para tratar a mobilidade do *mobile node*. Os ativos na borda da rede de acesso são responsáveis somente pelo encaminhamento de tráfego para os *mobile nodes* que estão conectados a eles, não se envolvendo no redirecionamento de tráfego para *mobile nodes* que estão conectados a outros ativos.

Em relação ao plano de controle, o controlador que é o ponto central de inteligência da rede e responsável pelo tratando da mobilidade no domínio, pode ser implementado em modo *cluster* de maneira hierárquica. Onde determinado controlador fica como responsável principal por uma parte da infraestrutura, enquanto os demais ficam como *backup* para esta parte e atuando como principal para outras partes da rede. Isto permite tratar questões de escalabilidade e ponto único de falha para o plano de controle.

Como os *mobile nodes* não se envolvem e não precisam executar nenhum processo adicional em relação à mobilidade e a infraestrutura de rede não precisa suportar nenhum novo protocolo para tal, a proposta possui o benefício de manter o mesmo nível de complexidade tanto para a borda da rede e para os dispositivos finais, quanto para o núcleo da rede, considerando a arquitetura baseada em SDN. As questões de processamento e de consumo energético no *mobile node* não são afetadas e os ativos de rede não precisam criar ou

manter tabelas e processos específicos como tunelamentos e relações de *bindings* para prover a mobilidade.

A otimização do roteamento, onde os pacotes são encaminhados diretamente para o *mobile node* baseado no fluxo IP bidirecional, melhora a experiência do usuário e usa de forma eficiente os enlaces de comunicação disponíveis. Uma vez que o custo de roteamento não é afetado, quando comparado às outras propostas, pela localização do *mobile node*, escopo de endereçamento ou uso inapropriado de enlaces, por exemplo, em cenários de roteamento triangular.

Esta otimização, aliada a não utilização de técnicas de tunelamento (que adicionaria um *overhead* tanto para o processamento quanto para o transporte do *payload* pela rede de comunicação), permite que a solução mantenha um nível de desempenho superior para a comunicação realizada em um cenário de mobilidade, quando comparado às demais abordagens utilizadas.

Sendo assim, a proposta SDN-DMM trata adequadamente as questões de escalabilidade, desempenho, roteamento e complexidade, envolvidas no gerenciamento de mobilidade distribuído.

A implementação real e os testes descritos no capítulo 6 demonstram que a proposta baseada na arquitetura SDN garante o mesmo desempenho para a comunicação dos *hosts* em mobilidade do que os ambientes tradicionais de roteamento IP, não adicionando *overhead* para a transmissão dos dados pela infraestrutura de rede.

Em síntese, a proposta SDN-DMM permite obter os seguintes benefícios:

- Transparência para os *host*s finais;
- Uso otimizado dos recursos de infraestrutura de rede;
- Complexidade baixa de acordo com a arquitetura SDN;
- Adiciona escalabilidade e flexibilidade;
- Melhor Desempenho.

7.2 TRABALHOS EM ANDAMENTO E FUTUROS

Trabalhos em andamento envolvem a validação de um modelo e análises estatísticas com o foco para IP Flow data offloading e suporte a possibilidade de um modelo híbrido com a extensão do SDN para o *mobile node*, utilizando simultaneamente uma abordagem *client-based* e *network-based.*

Outros temas relevantes que não foram abordados sendo indicados para continuidade da pesquisa são as relações de mobilidade entre diferentes sistemas autônomos.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] KARIMZADEH M; VALTULINA, L; KARAGIANNIS, G. *Applying SDN/OpenFlow in virtualized LTE to support Distributed Mobility Management (DMM).* In: Proceedings of the 4th International Conference on Cloud Computing and Services Science, CLOSER 2014, 3-5 Barcelona, Espanha. 86. SciTePress, Abril 2014.

[2] KARIMZADEH, M; ZHAO, Z; HENDRIKS, L; SCHMIDT, R; FLEUR, S; BERG, H; PRAS, A; BRAUN, T; CORICI, M. *Mobility and Bandwidth Prediction as a Service in Virtualized LTE Systems*. 4th IEEE International Conference on Cloud Networking (CLOUDNET), 2015.

[3] VALTULINA, L; KARIMZADEH, M; KARAGIANNIS, G; HEIJENK, A; PRAS, A. *Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems*. IEEE GLOBECOM 2014.

[4] GIUST, F; BERNARDOS, C; OLIVIA, A. *HDMM: deploying client and network-based distributed mobility management*. Journal Telecommunications Systems, Volume 59 Edição 2, Pag. 247-270, Junho 2015.

[5] SEITE, P; BERTIN, P; *Distributed Mobility Anchoring*. Draft IETF at https://tools.ietf.org/html/draft-seite-dmm-dma-00, Fevereiro 2012.

[6] LANTZ, B; O'CONNOR, B; HART, J; BERDE, P; RADOSLAVOV, P; KOBAYASHI, M; KOIDE, T; HIGUCHI, Y; GEROLA, M; SNOW, W; PARULKAR, G. *ONOS: Towards an Open, Distributed SDN OS*. SIGCOMM, Chicago, Agosto 2014.

[7] *Introducing ONOS - a SDN network operating system for Service Providers*. OnLab Whitepaper, 2014.

[8] Perkins, C., Johnson, D., & Arkko, J. (2011). Mobility Support in IPv6. RFC 6275.

[9] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K.,&Patil, B. (2008). Proxy Mobile IPv6. RFC 5213.

[10] Chan,H.A.,Yokota,H., Xie, J., Seite, P.,&Liu,D. (2011). Distributed and dynamicmobility management in mobile internet:Current approaches and issues. Journal of Communications, 6(1), 4–15.

[11] Chan, H. (2014). Requirements for Distributed Mobility Management. RFC 7333.

[12] Yokota, H., Seite, P., Demaria, E., & Cao, Z. (2010). Use case scenarios for Distributed Mobility Management. Internet-Draft (work in progress), draft-yokota-dmm-scenario-00.txt.

[13] Karimzadeh, M., Sperotto, A., & Pras, A. (2014). Software defined networking to improve mobility management performance. In Monitoring and securing virtualized networks and services, lecture notes in computer science (LNCS) (pp. 118–122). Berlin: Springer.

[14] RNP. Estatísticas de operação da Rede Ipê – Backbone da Rede Nacional de Ensino e Pesquisa (RNP), 2016.

[15] RNP. Estatísticas de operação do Ponto de Presença da RNP no Distrito Federal (PoP-DF), 2016.

[16] Giust F., Bernardos C., Oliva A. (2014). Analytic Evaluation and Experimental Validation of a Network-Based IPv6 Distributed Mobility Management Solution. In IEEE Transactions on Mobile Computing, vol. 13, no. 11, pp. 2484-97, November 2014.

[17] L. Valtulina, M. Karimzadeh, G. Karagiannis, G. Heijenk, A. Pras, "Performance evaluation of a SDN/openflow-based distributed mobility management (DMM) approach in virtualized LTE systems", Proc. IEEE GLOBECOM, pp. 18-23, Dec. 2014.

[18] FEAMSTER, N; REXFORD, J; ZEGURA, E. The road to SDN: an intellectual history of programmable networks, ACM SIGCOMM Computer Communication Review, v.44, n.2, April 2014. [doi>10.1145/2602204.2602219]

[19] M. Kobayashi et al., "Maturing of OpenFlow and software-defined networking through deployments", Comput. Netw., vol. 61, Special Issue on Future Internet Testbeds—Part I, pp. 151-175, 2014.

[20] CASADO, M; FOSTER, N; GUHA, A. Abstractions for software-defined networks, Communications of the ACM, v.57, n.10, October 2014. [doi>10.1145/2661061.2661063]

[21] B. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks", IEEE Commun. Surv. Tut., vol. 16, no. 3, pp. 1617-1634, 2014.

[22] J. Lee and Y. Kim, "PMIPv6-based Distributed Mobility Management," IETF draft, draft-jaehwoon-dmm-pmipv6-04, June 18 2015.

[23] BERNARDOS, C.; OLIVIA, A.; GIUST, F. A PMIPv6-based solution for Distributed Mobility Management. In IETF draft, draft-bernardos-dmm-pmip-06, Março 2016.

[24] GIUST, F.; OLIVIA, A.; BERNARDOS, C.; COSTA, R. A network-based localized mobility solution for Distributed Mobility Management. In Wireless Personal Multimedia Communications (WPMC), 14th International Symposium, October 2011.

[25] Lee, H.-B; Min, S.-G; Han, Y.-H; Lee, K.-H; Lee, H.-W; Ryu, W. IP flow mobility scheme in scalable network-based mobility management architecture. Telecommunication Systems, 1–11, 2015.

[26] Lee, K.-H., Lee, H.-W., Ruy, W., & Han, Y.-H. A scalable network-based mobility management framework in heterogeneous IP-based networks. Telecommunication Systems Journal, 52(4), 1989–2002, 2013.

[27] BERNARDOS C., OLIVA A., SERRANO P., et al. (2014). *An Architecture for Software Defined Wireless Networking*. In IEEE Wireless Communications, vol. 21, no.3, pag 52-61.

[28] PAIVA, R; BAHIA, A; ISHIMORI, A; PINHEIRO, B; FARIAS, F; ABELEM, A. *Gerenciamento de Recursos no Processo de Handoff em Redes Sem fio Definidas por Software*. 31º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Porto Alegre - RS: Editora da SBC, v. 1. p. 23-28, 2013.

[29] SUN, K; KIM, Y. *Flow Mobility Management in PMIPv6-based DMM (Distributed Mobility Management) Networks*. Journal of Wireless Mobile Networks, Ubiquitous Computing and Dependable Applications, volume: 5, number: 4, pp. 120, 2014.

[30] CHOI, H; MIN, S; HAN, Y; KOODLI, R. *Desing and Simulation of a Flow Mobility Scheme Based on Proxy Mobile IPv6*. Jornal of Information Processing Systems, Volume 8, N°4, Dezembro, 2012.

[31] HOLMA, H; TOSKALA, A. *LTE for UMTS – Evolution to LTE-Advanced*. 2.ed. Finland: Wiley, 2011.

[32] REFORD, J; FEAMSTER, N; CEASER, M; CADWELL, D; SHAIKH, A; MERWE, J. *Design and implementation of a Routing Control Plataform*. Proceeding NSDI'05 Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2, Pages 15-28, 2005.

[33] KUROSE, J.; ROSS, K. *Redes de computadores e a internet: uma abordagem top-down*. 6. ed. São Paulo: Pearson Addison Wesley, 2009.

[34] REFORD, J; FEAMSTER, N; GUPTA, A; VANBEVER, L; SHAHBAZ, M; DONOVAN, S; SCHLINKER, B; SHENKER, S; CLARK, R; KATZ-BASSET, E. *SDX- A Software Defined Internet Exchange*. SIGCOMM, Chicago, Agosto 2014.

[35] 3GPP TS 23.401, 23.203, 36.413 and 23.060 at http://www.3gpp.org/specifications

[36] Guedes, D; Vieira, L; Vieira, M; Rodrigues, H. e Nunes, R. “Redes Definidas por Software: uma abordagem sistêmica para o desenvolvimento de pesquisas em Redes de Computadores”. Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC 2012, p. 160–210, 2012.

[37] CASADO, M; FRREDMAN, J; PETTIT, J; LUO, J; MCKEOWN, N. AND SHENKER, S. "Ethane: taking control of the enterprise". Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications, August 27-31, Kyoto, Japan, 2007. [doi>10.1145/1282380.1282382]

[38] CASADO, M; KUPONEN, T; RAMANATHAN, R. AND SHENKER, S. "Virtualizing the network forwarding plane". Proceedings of the Workshop on Programmable Routers for Extensible Services of Tomorrow, November 30-30, Philadelphia, Pennsylvania, 2010. [doi>10.1145/1921151.1921162]

[39] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, "Openflow: enabling innovation in campus networks", ACM SIGCOMM Computer Commun. Review, vol. 38, no. 2, pp. 69-74, 2008.

[40] Sherwood, R., Gibb, G., Yap, K.-K., Apenzeller, G., Casado, M., McKeown, N., and Parulkar, G. *Flowvisor: A network virtualization layer*. Tech. Rep. OPENFLOWTR, OpenFlowSwitch.org, 2009

[41] Baptista, Gustavo Luiz B., Endler, Markus, Sacramento, Vagner. *Gerenciamento de Mobilidade e Tratamento de Desconexão Baseado em SIP*. Rio de Janeiro, 2009. 100p. Dissertação de Mestrado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

[42] Trineto, José Benício Menezes. *Avaliação de Desempenho de Protocolos de Autenticação em Redes Sem Fio Heterogêneas*. Brasília, 2016. 147 p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Universidade de Brasília.

[43] *Heterogeneous Wireless Access Networks Architectures and Protocols*. Edited by Ekram Hossain.© 2009Springer US. ISBN: 978-0-387-09776-3.

[44] 3GPP. *Long Term Evolution*. Em http://www.3gpp.org/technologies/keywords-acronyms/98-lte [acessado julho-2015].

[45] IEEE802.11a - IEEE Doc. IEEE P802.11-96/49C. "802.11 Tutorial – 802.11 MAC Entity: MAC Basic Access Mechanism Privacy and Access Control". U.S.A., 1996.

[46] IEEE802.11b - IEEE Standard 802.11. "The IEEE 802.11 Standard". U.S.A., 1997.

[47] Odom, Wendel. *CCNA 640-802 Official Cert Library*. 3. ed. USA: Cisco Press, novembro 2011.

[48] PUC RIO MAXWELL. *Trabalho sobre redes sem fio IEEE 802.11*. Em http://www.maxwell.vrac.puc-rio.br/5688/5688_4.PDF [acessado fevereiro-2015].

[49] Magno, Ricardo; Rechart, Diogo; Gonçalves, Daniel; Ferrão, David. *Como Evoluíram as Normas Wi-Fi IEEE 802.11*. Porto, 2013. Relatório de Mestrado Integrado em Engenharia Eletrotécnica e de Computadores, Faculdade de Engenharia - Universidade do Porto.

[50] Bulhman, Haroldo José; Cabianca, Luis Antonio. *Redes LAN/MAN Wireless II*. São Paulo, 2016. Artigo publicado pela empresa Teleco-Inteligência em telecomunicações, disponível em http://www.teleco.com.br/tutoriais/tutorialrwlanman2/default.asp [acessado em julho-2016];

[51] Universidade de Stanford – *Overview: How does OpenFlow Work* http://yuba.stanford.edu/cs244wiki/index.php/Overview [acessado em julho-2015];

[52] CISCO SYSTEM. Routers Products and Solotions. http://www.cisco.com/c/en/us/products/routers/buyers-guide.html [accessed May-2016];

[53] JUNIPER NETWORKS. MX Series Plataforms. http://www.juniper.net/us/en/products-services/routing/mx-series/compare/#a=VMX,P5,P10,P40,P80,P104,P240,P480,P960,P2010,P2020 [accessed May-2016].

APÊNDICES

APÊNDICE A – CONFIGURAÇÕES DA IMPLEMENTAÇÃO

A.1 – AMBIENTE DE REDE

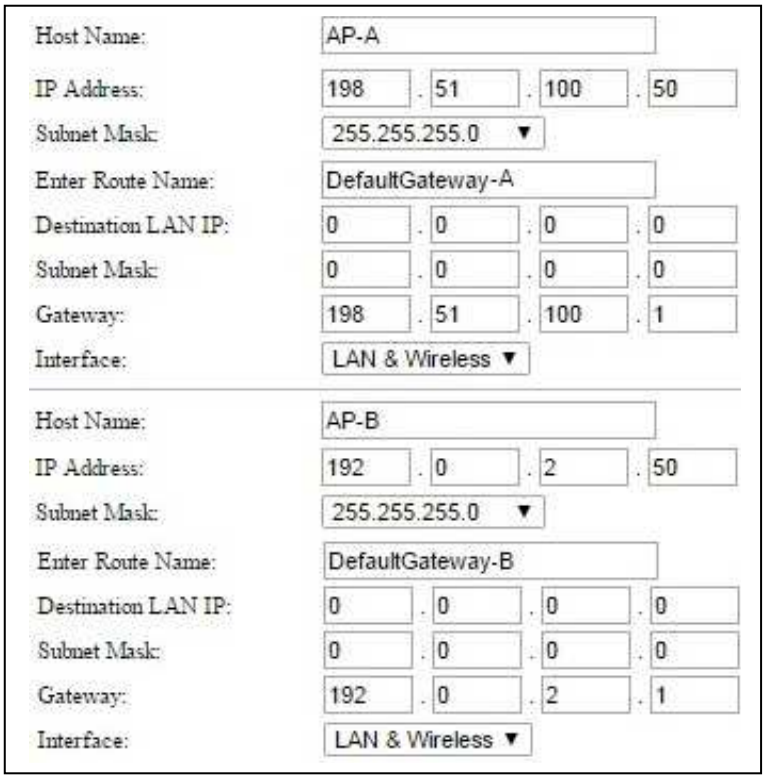
A configuração e endereçamento das VLANs utilizadas no switch OFS1 foram realizados conforme os comandos abaixo:

```
#Rede Internet
create vlan Rede-Internet tag 203
configure vlan Rede-Internet add ports 1 untagged
configure vlan Rede-Internet ipaddress 203.0.113.1/24

#Rede Cliente A
create vlan Rede-Cliente-A tag 198
configure vlan Rede-Cliente-A add ports 10 untagged
configure vlan Rede-Cliente-A ipaddress 198.51.100.1/24

#Rede Cliente B
create vlan Rede-Cliente-B tag 192
configure vlan Rede-Cliente-B add ports 20 untagged
configure vlan Rede-Cliente-B ipaddress 192.0.2.1/24

#Rede de Gerência
configure vlan Mgmt ipaddress 10.5.0.2/30

#Roteamento entre as Redes
enable ipforwarding ipv4 vlan Rede-Internet
enable ipforwarding ipv4 vlan Rede-Cliente-A
enable ipforwarding ipv4 vlan Rede-Cliente-B
```

O endereçamento IP dos *hosts* MN1, MD1 e C1 foi realizado de modo estático, sendo:

```
#host MN1
netsh interface ip set address name=”Conexão de Rede sem Fio” static
198.51.100.100 255.255.255.0 198.51.100.1

#host MD1
netsh interface ip set address name=”Conexão local” static 203.0.113.100
255.255.255.0 203.0.113.1

#host C1
netsh interface ip set address name=”Conexão local” static 192.168.25.118
255.255.255.0
```

As configurações de endereçamento IP dos *access points*, também realizadas de modo estático, podem ser visualizadas abaixo:



A.2 – CONTROLADOR SDN

O controlador ONOS foi implementado na modalidade *container* utilizando o *software* Docker no sistema operacional Ubuntu Server. Para a instalação do *software* Docker é necessário utilizar um sistema operacional de 64 bits que execute pelo menos a versão 3.10 do *kernel*. A instalação, ativação e verificação do Docker no *host* C1 foram realizadas com os seguintes comandos:

```
#Atualização do repositório APT, ativação do HTTPS e Certificados
apt-get update
apt-get install apt-transport-https ca-certificates

#Adição da chave GPG
apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys
58118E89F3A912897C070ADBF76221572C52609D

#Instalação e ativação
apt-get install docker-engine
service docker start

#Verificação da instalação e versão
docker version
```

Para baixar e executar o *container* do ONOS foram executados os seguintes comandos:

```
#Download do container ONOS do Docker Hub
docker pull onosproject/onos

#Verificação da imagem docker do ONOS
docker images

#Executando o container ONOS, nomeando-o como ONOS-Controller e realizando o redirecionamento de portas
docker run -td -p 8101:8101 -p 80:8181 -p 6633:6633 --name ONOS-Controller onosproject/onos

#Verificação da execução do container ONOS
docker ps -a
```

A figura A.2.1 ilustra o resultado dos comandos descritos acima na instalação e execução do *container* ONOS-Controller.

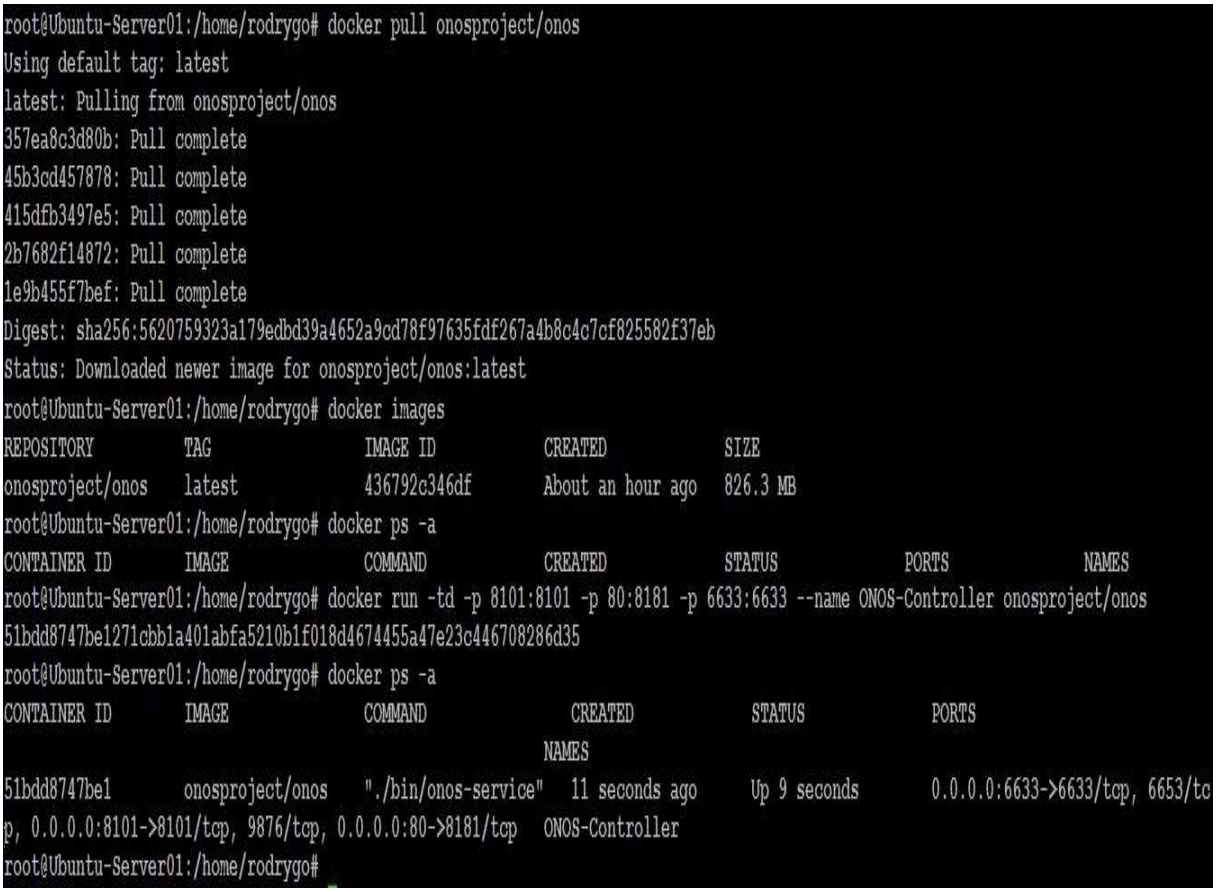


Figura A.2.1 – *Instalação e execução do container ONOS-Controller.*

Quando executamos um *container* com o comando *docker run* utilizando o parâmetro *–p*, estamos criando um redirecionamento de portas para que todas as conexões recebidas pelo

host local em uma determinada porta, sejam redirecionadas para o *container* em uma determinada porta. Deste modo o *container* pode ser acessado de maneira remota através do *host* local.

No caso da implementação da proposta SDN-DMM, ao executarmos o *container* ONOS com os parâmetros *–p 8101:8101 –p 80:8181 e –p 6633:6633*, o *host* C1 redireciona todo o tráfego recebido nas portas definidas no primeiro valor do par de portas, para o *container* nas portas definidas no segundo valor do par de portas.

Assim o tráfego recebido nas portas 8101, 80 e 6633 do *host* C1 é redirecionado respectivamente para as portas 8101, 8181 e 6633 do *container*, que no caso do *container* ONOS executam em ordem os serviços de SSH, Acesso Web e comunicação *OpenFlow*.

A conexão lógica e os IP atribuídos para o *host* C1 e o *container* ONOS-Controller podem ser visualizados na figura A.2.2, assim como a comunicação efetiva entre eles através desta conexão.

```
root@Ubuntu-Server01:/home/rodrygo# ifconfig
docker0   Link encap:Ethernet  HWaddr 02:42:ea:c7:49:bc
          inet addr:172.17.0.1  Bcast:0.0.0.0  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@Ubuntu-Server01:/home/rodrygo# docker inspect ONOS-Controller | grep -w IPAddress
            "IPAddress": "172.17.0.2",
                    "IPAddress": "172.17.0.2",

root@Ubuntu-Server01:/home/rodrygo# ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.209 ms
64 bytes from 172.17.0.2: icmp_seq=2 ttl=64 time=0.096 ms
64 bytes from 172.17.0.2: icmp_seq=3 ttl=64 time=0.081 ms
^C
--- 172.17.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.081/0.128/0.209/0.058 ms
root@Ubuntu-Server01:/home/rodrygo#
```

Figura A.2.2 – *Conexão lógica entre o host C1 e o container ONOS-Controller.*

Para realizar o acesso remoto ao controlador ONOS, foram configuradas duas interfaces de rede no *host* C1, a eth0 para permitir o acesso remoto pela rede de acesso OOB e a eth1 para permitir a troca de mensagens de controle *OpenFlow* e de gerência com o switch OFS1.

Os seguintes comandos foram inseridos no arquivo de configuração /etc/network/interfaces para alcançar este objetivo:

```
#Interface eth0 para a rede de acesso remoto OOB
auto eth0
iface eth0 inet static
address 192.168.25.118
netmask 255.255.255.0
broadcast 192.168.25.255
gateway 192.168.25.1
dns-nameservers 192.168.25.1 8.8.8.8

#Interface eth0 para a rede de gerência
auto eth1
iface eth1 inet static
address 10.0.5.1
netmask 255.255.255.252
broadcast 10.0.5.3
```

Uma vez realizadas as ações descritas até aqui, é possível acessar remotamente o controlador através do terminal remoto pela rede de acesso *out-of-band* tanto via linha de comando utilizando SSH, quanto via acesso WEB por HTTP.

A figura A.2.3 ilustra o acesso à CLI e a figura A.2.4 o acesso WEB realizados a partir do terminal remoto para o controlador.

```
# ssh onos@192.168.25.118 -p 8101
Password authentication
Password:
Welcome to Open Network Operating System (ONOS)!

Documentation: wiki.onosproject.org
Tutorials:     tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown ONOS.

onos>
```

Figura A.2.3 – *Acesso remoto SSH ao ONOS-Controller.*

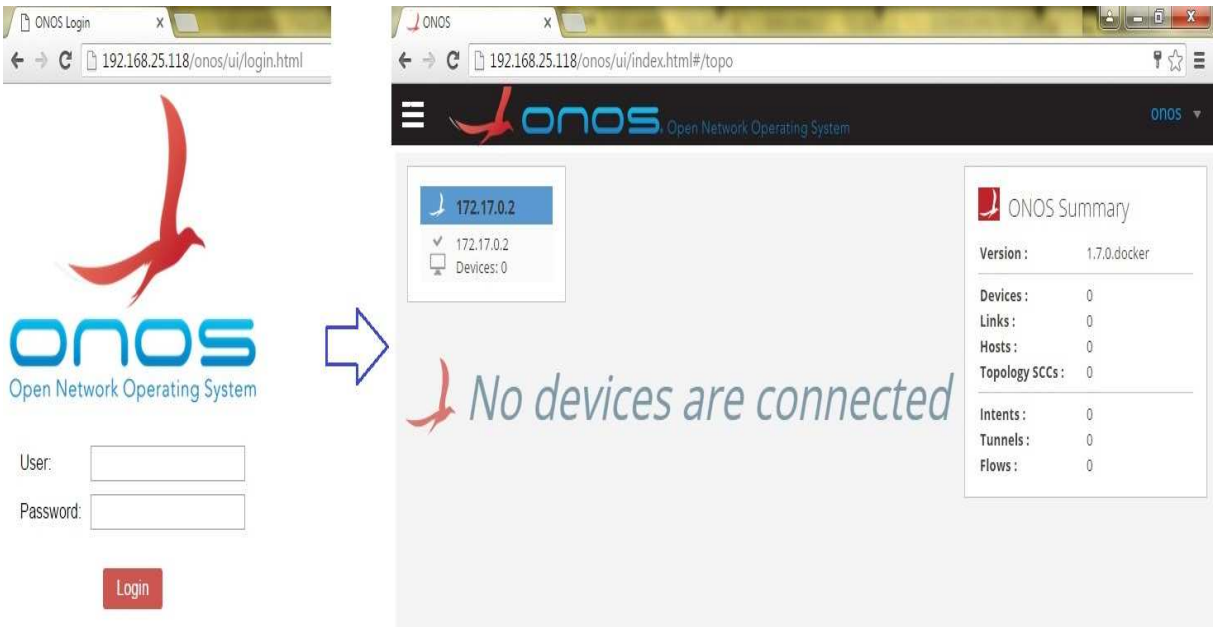


Figura A.2.4 – *Acesso remoto WEB ao ONOS-Controller.*

A.3 – PROTOCOLO OPENFLOW

Abaixo são apresentadas as configurações necessárias para habilitar e ativar o protocolo *OpenFlow* no switch OFS1 para se comunicar com o controlador C1 e estabelecer a relação de plano de controle e plano de dados da rede SDN:

```
#Verificação da imagem, suporte OpenFlow, configurações, fluxos e tabelas
show version images
show version process “openflow”
show openflow
show openflow flows
show openflow tables

#Criação da VLAN para o data-plane da arquitetura SDN
create vlan SDN-Data-Plane tag 10
configure vlan SDN-Data-Plane add ports 1 untagged
configure vlan SDN-Data-Plane add ports 10 untagged
configure vlan SDN-Data-Plane add ports 20 untagged

#Ativação do openflow e definição do modo de operação, controlador e fluxos locais
enable openflow
configure openflow vlan SDN-Data-Plane mode standard
configure openflow controller primary out-of-band active ipaddress 10.0.5.1 vr VR-Mgmt
configure openflow default-rule iparp controller
configure openflow default-rule lldp controller
configure openflow default-rule bddp controller
```

A figura A.3.1 ilustra o resultado para a verificação do suporte ao protocolo *OpenFlow* e as informações gerais antes de realizar as configurações de ativação do protocolo no switch OFS1.

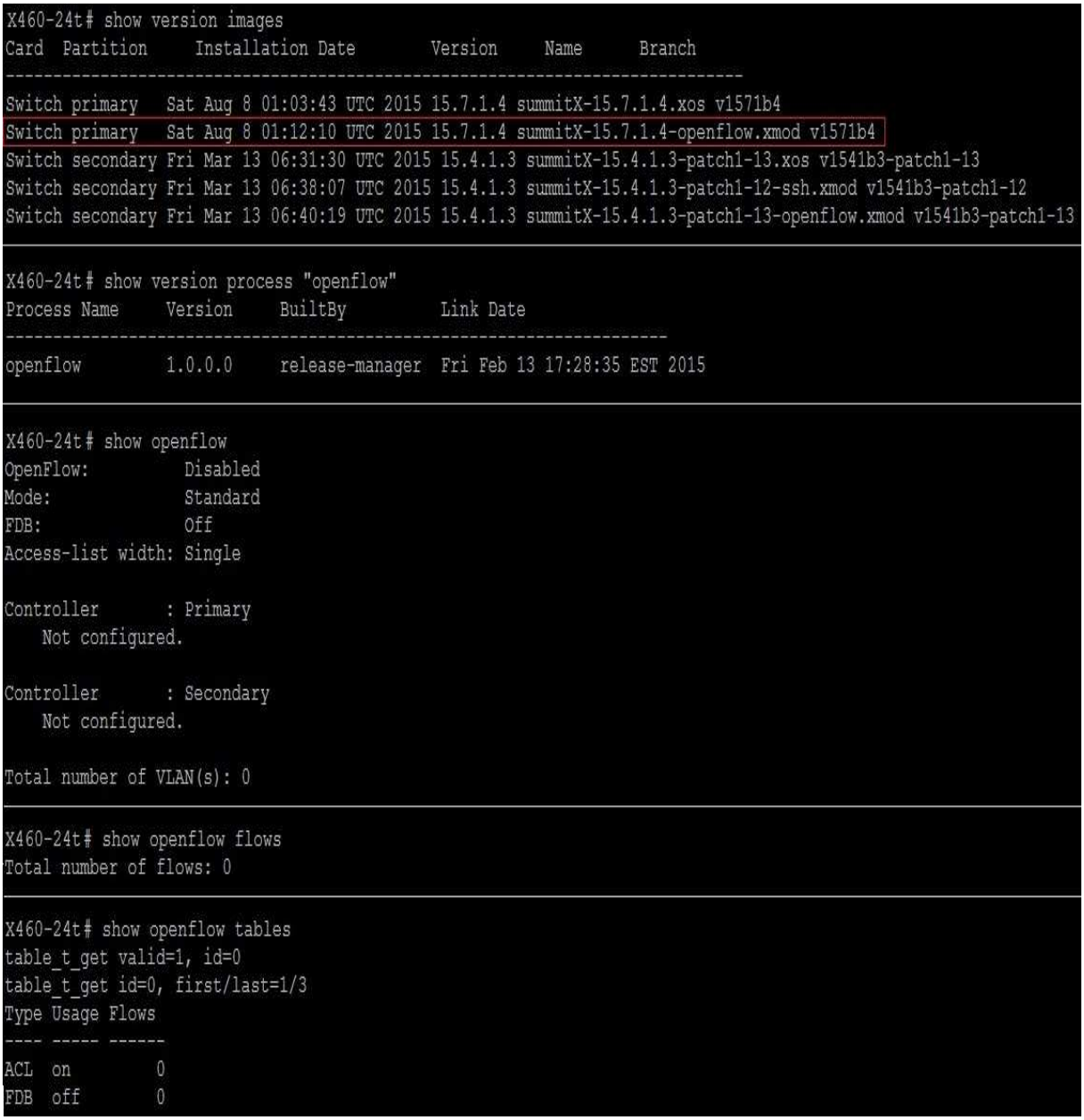


Figura A.3.1 – *Verificação do suporte ao OpenFlow e informações gerais antes da implementação.*

Podemos observar na figura A.3.1 que o switch OFS1 possui instalada a imagem para o modulo *OpenFlow* da versão 15.7.1.4 do sistema operacional EXOS na sua partição principal, partição utilizada para carregar o sistema e módulos para a operação do switch e que a versão do processo *OpenFlow* utilizada é a 1.0.0.0.

O *OpenFlow* encontra-se desabilitado sem quaisquer configurações, portanto não existe a presença de uma VLAN para realizar o plano de dados da rede SDN, assim não constando nenhum tipo de fluxos e entradas nas tabelas.

Após a execução das configurações de implementação do *OpenFlow* no switch OFS1, o protocolo é ativado e o switch tenta ativamente iniciar uma sessão para a troca de mensagens *OpenFlow* através da porta padrão TCP 6633 com o controlador C1, que foi definido como o controlador primário de IP 10.0.5.1, de modo a estabelecer a relação de plano de controle da rede SDN.

A VLAN SDN-Data-Plane passa a ser utilizada como a VLAN de plano de dados para o encaminhamento de pacotes no modo de operação *standard*, este modo estabelece que o encaminhamento de pacotes seja realizado somente através dos fluxos IP definidos pelas regras *OpenFlow* e não mais baseado no roteamento tradicional de pacotes.

A figura A.3.2 apresenta a saída do comando de verificação da operação do protocolo *OpenFlow* no switch OFS1, logo após executar os comandos de implementação do *OpenFlow* no switch, que ilustra o comportamento descrito acima.

```
X460-24t # show openflow
OpenFlow:          Enabled
Versions:          OpenFlow10, OpenFlow13
Mode:              Standard
FDB:               Off
Access-list width: Single

Controller       : Primary
    Status       : CONNECTING
    Datapath ID  : 00000004968bf976
    VR           : VR-Default
    Mode         : out-of-band Active
    Target       : tcp:10.0.5.1:6633
    Uptime(secs) : 0

Controller       : Secondary
    Not configured.

                                                  Flows
VLAN                            VID  Mode     Ports Active Error
------------------------------- ---- -------- ----- ------ ------
SDN-Data-Plane                   10 Standard     3      0      0

Total number of VLAN(s): 1
```

Figura A.3.2 – *Protocolo OpenFlow ativado no switch OFS1.*

A tabela ACL, assim como as *default flow*s, podem ser observadas na figura A.3.3, que apresenta a saída do comando de verificação dos fluxos IP e tabelas no switch OFS1 depois da implementação do *OpenFlow*.

Quatro *default flows* foram implementadas no switch OFS1. Sendo as três primeiras realizando correspondência com o tráfego relacionado ao protocolo ARP, protocolo LLDP e protocolo BDDP e a última realizando correspondência com todo o tráfego restante, ou seja, uma espécie de regra *default*. Todas as *default flows* foram implantadas com a ação de encaminhar o fluxo correspondente para o controlador C1.

```
X460-24t# show openflow flows
Total number of OpenFlow flows : 0
Total number of default flows  : 4

Flow name        Type     Duration (secs)  Prio              Packets
---------------- ---- -------------------- ----- --------------------
ofDefault_0      ACL                  196     0                    0
    Match  :
    Actions:  CONTROLLER
ofDefault_1      ACL                  190     0                    0
    Match  :  Ethernet Type       : 0x0806 (ARP)
    Actions:  CONTROLLER
ofDefault_2      ACL                  184     0                    0
    Match  :  Ethernet Type       : 0x88cc (LLDP)
    Actions:  CONTROLLER
ofDefault_3      ACL                  180     0                    0
    Match  :  Ethernet Type       : 0x8942
    Actions:  CONTROLLER

Packets:  (*) Cumulative packet count for all FDB flows
```

```
X460-24t# show openflow tables
table_t_get valid=1, id=0
table_t_get id=0, first/last=1/3
Type Usage Flows
---- ----- ------
ACL  on         0
FDB  off        0
```

Figura A.3.3 – *Fluxos IP e tabelas no switch OFS1*.

Podemos observar na figura A.3.3 que o switch OFS1 ainda não possui nenhum fluxo *OpenFlow* definido pelo controlador C1, uma vez que a conexão com este ainda não foi estabelecida, apresentando então o valor 0 para o número total de fluxos *OpenFlow* na parte superior da figura e para o número de *flows* contidos na tabela ACL na parte inferior da mesma.

O não estabelecimento da conexão entre o switch OFS1 e o controlador C1, pode ser verificado pela informação *CONNECTING* no parâmetro *Status* relacionado ao controlador na

figura A.3.2, que indica que o switch OFS1 está tentando estabelecer uma conexão com o controlador C1, mas ainda sem sucesso.

Por padrão, o switch tenta estabelecer uma nova conexão com o controlador em intervalos de 8 segundos. A tentativa do estabelecimento da sessão TCP pode ser observado na figura A.3.4, que apresenta a captura de pacotes realizada com o *software* Wireshark na interface física de rede do hospedeiro N1, utilizada pelo controlador C1 para a comunicação com o switch OFS1.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.5.2	10.0.5.1	TCP	74	37953 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
2	0.000501	10.0.5.1	10.0.5.2	TCP	60	6633 → 37953 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3	8.002393	10.0.5.2	10.0.5.1	TCP	74	37954 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
4	8.002938	10.0.5.1	10.0.5.2	TCP	60	6633 → 37954 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
5	16.012487	10.0.5.2	10.0.5.1	TCP	74	37955 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
6	16.013102	10.0.5.1	10.0.5.2	TCP	60	6633 → 37955 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
10	24.032069	10.0.5.2	10.0.5.1	TCP	74	37956 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
11	24.032569	10.0.5.1	10.0.5.2	TCP	60	6633 → 37956 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
12	31.999427	10.0.5.2	10.0.5.1	TCP	74	37957 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
13	31.999974	10.0.5.1	10.0.5.2	TCP	60	6633 → 37957 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
14	40.041862	10.0.5.2	10.0.5.1	TCP	74	37958 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
15	40.042447	10.0.5.1	10.0.5.2	TCP	60	6633 → 37958 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
19	47.999083	10.0.5.2	10.0.5.1	TCP	74	37959 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
20	47.999634	10.0.5.1	10.0.5.2	TCP	60	6633 → 37959 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22	56.019036	10.0.5.2	10.0.5.1	TCP	74	49141 → 6633 [SYN] Seq=0 Win=5840 Len=0 MSS=1460
23	56.048072	10.0.5.1	10.0.5.2	TCP	60	6633 → 49141 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figura A.3.4 – *Tentativa de estabelecimento de sessão TCP pelo switch OFS1 com o controlador C1 para a troca de mensagens OpenFlow.*

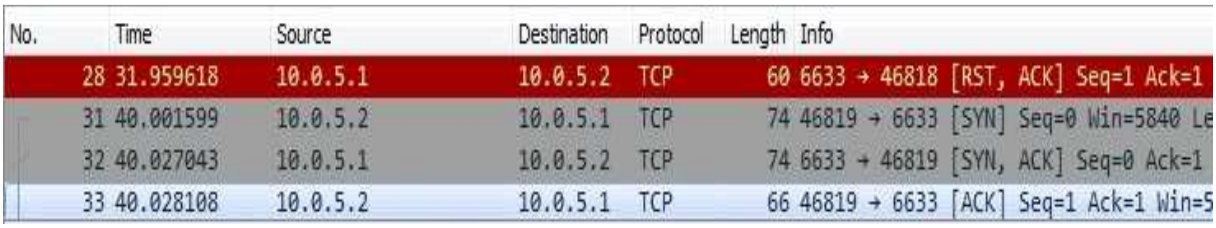
Podemos observar na figura A.3.4 que o controlador C1 recebe os pacotes SYN de inicialização da sessão TCP enviados pelo switch OFS1 para a troca de mensagens *OpenFlow*, mas os rejeita respondendo com pacotes RST, ACK.

Isto ocorre porque o protocolo *OpenFlow* ainda não foi implementado como o protocolo da camada *southbound* no controlador C1, apenas no switch OFS1, assim o controlador C1 considera que está solicitação é indevida.

Para que a sessão TCP possa ser estabelecida e a troca de mensagens *OpenFlow* realizada, permitindo a implementação da relação de plano de controle da rede SDN entre o switch OFS1 e o controlador C1, basta ativar o protocolo *OpenFlow* na camada *southbound* deste controlador com o comando:

```
#Ativação do protocolo OpenFlow como protocolo da camada southbound
app activate org.onosproject.openflow
```

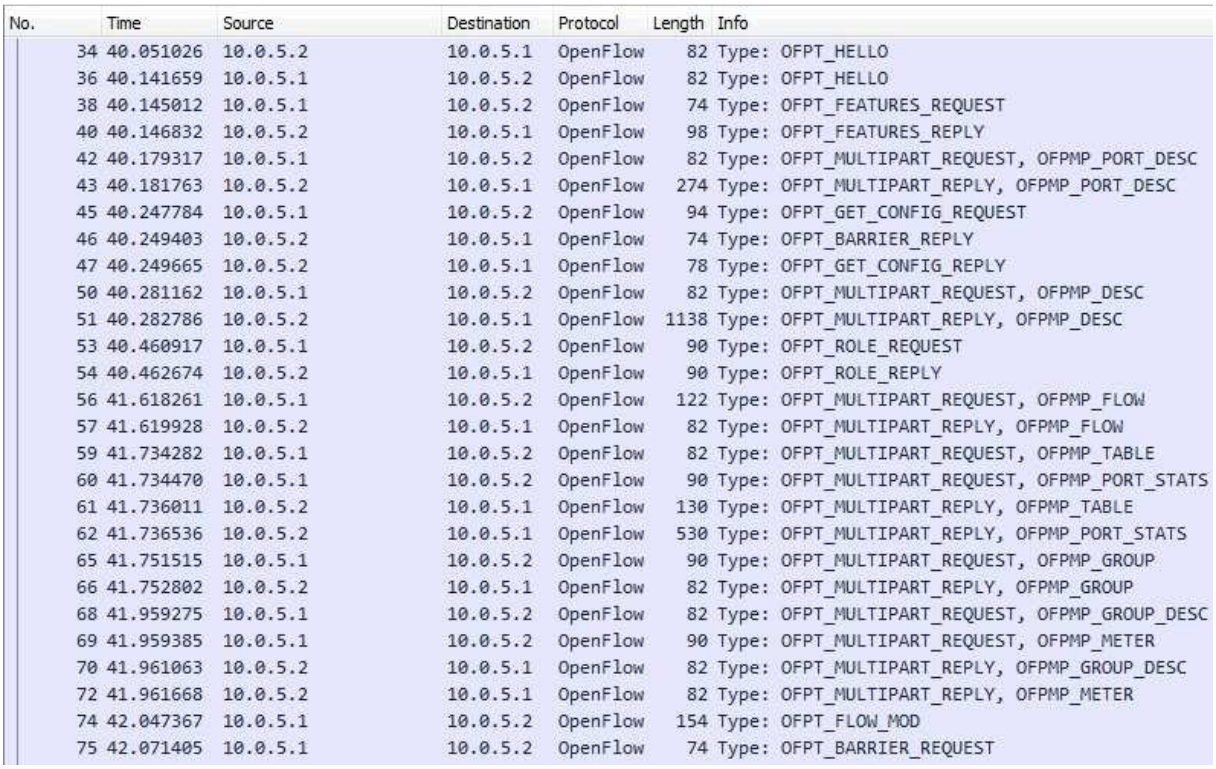
Ao ativar o protocolo *OpenFlow* no controlador C1, a sessão TCP é estabelecida, como pode ser observado na captura de pacotes apresentada na figura A.3.5.



No.	Time	Source	Destination	Protocol	Length	Info
28	31.959618	10.0.5.1	10.0.5.2	TCP	60	6633 → 46818 [RST, ACK] Seq=1 Ack=1
31	40.001599	10.0.5.2	10.0.5.1	TCP	74	46819 → 6633 [SYN] Seq=0 Win=5840 Le
32	40.027043	10.0.5.1	10.0.5.2	TCP	74	6633 → 46819 [SYN, ACK] Seq=0 Ack=1
33	40.028108	10.0.5.2	10.0.5.1	TCP	66	46819 → 6633 [ACK] Seq=1 Ack=1 Win=5

Figura A.3.5 – *Estabelecimento da sessão TCP para troca de mensagens OpenFlow.*

Logo após o estabelecimento da sessão TCP, o switch OFS1 e o controlador C1 iniciam uma troca de mensagens *OpenFlow* para divulgar as informações relativas a topologia, funcionalidades e recursos da rede SDN, como pode ser observado na figura A.3.6.



No.	Time	Source	Destination	Protocol	Length	Info
34	40.051026	10.0.5.2	10.0.5.1	OpenFlow	82	Type: OFPT_HELLO
36	40.141659	10.0.5.1	10.0.5.2	OpenFlow	82	Type: OFPT_HELLO
38	40.145012	10.0.5.1	10.0.5.2	OpenFlow	74	Type: OFPT_FEATURES_REQUEST
40	40.146832	10.0.5.2	10.0.5.1	OpenFlow	98	Type: OFPT_FEATURES_REPLY
42	40.179317	10.0.5.1	10.0.5.2	OpenFlow	82	Type: OFPT_MULTIPART_REQUEST, OFPMP_PORT_DESC
43	40.181763	10.0.5.2	10.0.5.1	OpenFlow	274	Type: OFPT_MULTIPART_REPLY, OFPMP_PORT_DESC
45	40.247784	10.0.5.1	10.0.5.2	OpenFlow	94	Type: OFPT_GET_CONFIG_REQUEST
46	40.249403	10.0.5.2	10.0.5.1	OpenFlow	74	Type: OFPT_BARRIER_REPLY
47	40.249665	10.0.5.2	10.0.5.1	OpenFlow	78	Type: OFPT_GET_CONFIG_REPLY
50	40.281162	10.0.5.1	10.0.5.2	OpenFlow	82	Type: OFPT_MULTIPART_REQUEST, OFPMP_DESC
51	40.282786	10.0.5.2	10.0.5.1	OpenFlow	1138	Type: OFPT_MULTIPART_REPLY, OFPMP_DESC
53	40.460917	10.0.5.1	10.0.5.2	OpenFlow	90	Type: OFPT_ROLE_REQUEST
54	40.462674	10.0.5.2	10.0.5.1	OpenFlow	90	Type: OFPT_ROLE_REPLY
56	41.618261	10.0.5.1	10.0.5.2	OpenFlow	122	Type: OFPT_MULTIPART_REQUEST, OFPMP_FLOW
57	41.619928	10.0.5.2	10.0.5.1	OpenFlow	82	Type: OFPT_MULTIPART_REPLY, OFPMP_FLOW
59	41.734282	10.0.5.1	10.0.5.2	OpenFlow	82	Type: OFPT_MULTIPART_REQUEST, OFPMP_TABLE
60	41.734470	10.0.5.1	10.0.5.2	OpenFlow	90	Type: OFPT_MULTIPART_REQUEST, OFPMP_PORT_STATS
61	41.736011	10.0.5.2	10.0.5.1	OpenFlow	130	Type: OFPT_MULTIPART_REPLY, OFPMP_TABLE
62	41.736536	10.0.5.2	10.0.5.1	OpenFlow	530	Type: OFPT_MULTIPART_REPLY, OFPMP_PORT_STATS
65	41.751515	10.0.5.1	10.0.5.2	OpenFlow	90	Type: OFPT_MULTIPART_REQUEST, OFPMP_GROUP
66	41.752802	10.0.5.2	10.0.5.1	OpenFlow	82	Type: OFPT_MULTIPART_REPLY, OFPMP_GROUP
68	41.959275	10.0.5.1	10.0.5.2	OpenFlow	82	Type: OFPT_MULTIPART_REQUEST, OFPMP_GROUP_DESC
69	41.959385	10.0.5.1	10.0.5.2	OpenFlow	90	Type: OFPT_MULTIPART_REQUEST, OFPMP_METER
70	41.961063	10.0.5.2	10.0.5.1	OpenFlow	82	Type: OFPT_MULTIPART_REPLY, OFPMP_GROUP_DESC
72	41.961668	10.0.5.2	10.0.5.1	OpenFlow	82	Type: OFPT_MULTIPART_REPLY, OFPMP_METER
74	42.047367	10.0.5.1	10.0.5.2	OpenFlow	154	Type: OFPT_FLOW_MOD
75	42.071405	10.0.5.1	10.0.5.2	OpenFlow	74	Type: OFPT_BARRIER_REQUEST

Figura A.3.6 – *Troca de mensagens OpenFlow entre o switch OFS1 e o controlador C1.*

Um exemplo das informações trocadas entre o switch OFS1 e o controlador C1, pode ser observado na figura A.3.7, que apresenta detalhes dos pacotes números 50 e 51 capturados na figura A.3.6.

Apresentando respectivamente a solicitação do controlador C1 com um pacote OFPT_MULTIPART_REQUEST do tipo OFPMP_DESC para que o switch OFS1 informe a

descrição da sua plataforma e a resposta do switch OFS1 com um pacote OFPT_MULTIPART_REPLY do mesmo tipo.

```
▷ Frame 50: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface 0
▷ Ethernet II, Src: Vmware_15:a0:57 (00:0c:29:15:a0:57), Dst: ExtremeN_8b:f9:76 (00:04:96:8b:f9:76)
▷ Internet Protocol Version 4, Src: 10.0.5.1, Dst: 10.0.5.2
▷ Transmission Control Protocol, Src Port: 6633 (6633), Dst Port: 46819 (46819), Seq: 69, Ack: 277, Len: 16
◢ OpenFlow 1.3
    Version: 1.3 (0x04)
    Type: OFPT_MULTIPART_REQUEST (18)
    Length: 16
    Transaction ID: 4294967289
    Type: OFPMP_DESC (0)
  ▷ Flags: 0x0000
    Pad: 00000000

▷ Frame 51: 1138 bytes on wire (9104 bits), 1138 bytes captured (9104 bits) on interface 0
▷ Ethernet II, Src: ExtremeN_8b:f9:76 (00:04:96:8b:f9:76), Dst: Vmware_15:a0:57 (00:0c:29:15:a0:57)
▷ Internet Protocol Version 4, Src: 10.0.5.2, Dst: 10.0.5.1
▷ Transmission Control Protocol, Src Port: 46819 (46819), Dst Port: 6633 (6633), Seq: 277, Ack: 85, Len: 1072
◢ OpenFlow 1.3
    Version: 1.3 (0x04)
    Type: OFPT_MULTIPART_REPLY (19)
    Length: 1072
    Transaction ID: 4294967289
    Type: OFPMP_DESC (0)
  ▷ Flags: 0x0000
    Pad: 00000000
    Manufacturer desc.: Extreme Networks
    Hardware desc.: X460-24t
    Software desc.: 15.7.1.4
    Serial no.: 1337N-42464
    Datapath desc.: None
```

Figura A.3.7 – *Troca de mensagens OpenFlow OFPMP_DESC entre o switch OFS1 e o controlador C1.*

Depois de finalizada a troca inicial de mensagens *OpenFlow*, o controlador C1 reconhece o switch OFS1 e passa a ser o seu controlador primário, definindo a partir de então os fluxos IP que devem ser implementados no switch para o encaminhamento dos pacotes. Isto pode ser observado na figura A.3.8, que apresenta o reconhecimento do switch na interface WEB.

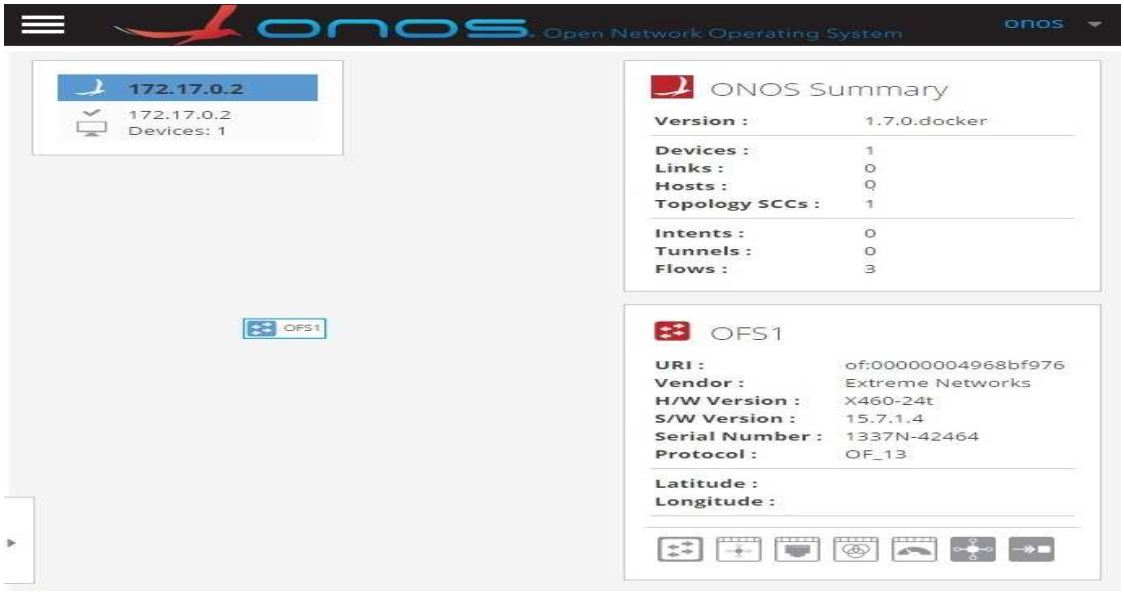


Figura A.3.8 – *Reconhecimento do switch OF1 pela interface WEB do ONOS.*

Por padrão, o controlador ONOS instala três fluxos ao reconhecer um switch, um para tratar o protocolo ARP, um para o protocolo BDDP e um para o protocolo LLDP, todos com a ação de encaminhar estes tipos de tráfegos para o controlador.

A figura A.3.9 mostra a ativação do protocolo *OpenFlow* pela linha de comando do ONOS-Controller, assim como um resumo das informações gerais do ambiente SDN, as informações do reconhecimento do switch OFS1 e as regras *OpenFlow* enviadas para que ele instale os fluxos correspondentes em sua tabela ACL.

```
onos> app activate org.onosproject.openflow
onos> onos:summary
node=172.17.0.2, version=1.7.0.docker
nodes=1, devices=1, links=0, hosts=0, SCC(s)=1, flows=3, intents=0
onos> devices
id=of:00000004968bf976, available=true, role=MASTER, type=SWITCH, mfr=Extreme Networks, hw=X460-24t, sw=15.7.1.4, serial=1337N-42464,
driver=default, channelId=10.0.5.2:46819, managementAddress=10.0.5.2, protocol=OF_13
onos> flows
deviceId=of:00000004968bf976, flowRuleCount=3
    id=100002d3e8a40, state=ADDED, bytes=0, packets=0, duration=964, priority=40000, tableId=0, appId=org.onosproject.core, payLoad=nu
ll, selector=[ETH_TYPE:arp], treatment=DefaultTrafficTreatment{immediate=[OUTPUT:CONTROLLER], deferred=[], transition=None, meter=None
, cleared=false, metadata=null}
    id=100002f55aef3, state=ADDED, bytes=0, packets=0, duration=964, priority=40000, tableId=0, appId=org.onosproject.core, payLoad=nu
ll, selector=[ETH_TYPE:bddp], treatment=DefaultTrafficTreatment{immediate=[OUTPUT:CONTROLLER], deferred=[], transition=None, meter=Non
e, cleared=false, metadata=null}
    id=10000407f8112, state=ADDED, bytes=0, packets=0, duration=964, priority=40000, tableId=0, appId=org.onosproject.core, payLoad=nu
ll, selector=[ETH_TYPE:lldp], treatment=DefaultTrafficTreatment{immediate=[OUTPUT:CONTROLLER], deferred=[], transition=None, meter=Non
e, cleared=false, metadata=null}
```

Figura A.3.9 – *Ativação do OpenFlow e reconhecimento do switch pela CLI do ONOS-Controller.*

O comando *onos:summary* executado na figura A.3.9, apresenta um resumo do ambiente SDN, identificando o IP do ONOS-Controller (172.17.0.2, que é o IP atribuído ao *container* na conexão lógica com o *host* C1 através do Docker), a versão do controlador ONOS utilizada (1.7.0.docker) e a presença de um dispositivo SDN (devices=1), no caso o switch OFS1.

As informações detalhadas deste dispositivo, podem ser obtidas com o comando *devices*, listando a sua descrição, contendo informações como o tipo de dispositivo, o fabricante, versão do sistema operacional, modelo, número serie, o IP e porta de conexão e a versão do protocolo *OpenFlow* utilizada.

Ainda observando a figura A.3.9, ao executar o comando *flows*, são listados todos os fluxos relacionados a determinado dispositivo. Para o switch OFS1, que recebeu o ID 00000004968bf976, definido a partir do seu endereço MAC, são apresentados os três fluxos padrões que o controlador ONOS solicita para adição na tabela de fluxos do switch.

Cada fluxo possui um ID de identificação, informa o estado da regra na tabela de fluxos do switch, se foi adicionada ou não, o tipo de tráfego para realizar a correspondência e qual ação deve ser tomada.

Verificamos que os fluxos devem estar instalados na tabela ACL do switch OFS1, uma vez que o estado presente no parâmetro *state* dos fluxos listados está como *ADDED*.

Isso pode ser verificado ao consultar a tabela de fluxos do switch OFS1, como observado na figura A.3.10, que apresenta os três fluxos instalados na tabela ACL e as informações principais como o nome do fluxo, a prioridade, o tipo de correspondência a ser executada e a ação a ser tomada.

```
X460-24t.14 # show openflow tables
table_t_get valid=1, id=0
table_t_get id=0, first/last=1/3
Type Usage Flows
---- ----- ------
ACL  on          3
FDB  off         0

X460-24t# show openflow flows
Total number of OpenFlow flows : 3
Total number of default flows  : 4

Flow name         Type     Duration (secs)  Prio                Packets
---------------- ---- -------------------- ----- --------------------
of_0              ACL                  3662 40000                     0
    Match   :  Ethernet Type        : 0x0806 (ARP)
    Actions:  CONTROLLER:65535
    Cookie                       : 0x100002d3e8a40
of_1              ACL                  3662 40000                     0
    Match   :  Ethernet Type        : 0x8942
    Actions:  CONTROLLER:65535
    Cookie                       : 0x100002f55aef3
of_2              ACL                  3662 40000                     0
    Match   :  Ethernet Type        : 0x88cc (LLDP)
    Actions:  CONTROLLER:65535
    Cookie                       : 0x10000407f8112
```

Figura A.3.10 – *Tabela de Fluxos do switch OFS1 com as regras definidas pelo controlador instaladas.*

A mensagem *OpenFlow* enviada pelo controlador C1 para solicitar a adição de um fluxo IP na tabela do switch OFS1, pode ser observada na captura de pacotes apresentada na figura A.3.11, onde o controlador solicita a inclusão de uma regra relacionada ao tráfego ARP.

```
▷ Internet Protocol Version 4, Src: 10.0.5.1, Dst: 10.0.5.2
▷ Transmission Control Protocol, Src Port: 6633 (6633), Dst Port: 46819 (46819)
◢ OpenFlow 1.3
    Version: 1.3 (0x04)
    Type: OFPT_FLOW_MOD (14)
    Cookie: 0x000100002d3e8a40
    Priority: 40000
  ◢ Match
      Type: OFPMT_OXM (1)
      Length: 10
    ◢ OXM field
        Class: OFPXMC_OPENFLOW_BASIC (0x8000)
        0000 101. = Field: OFPXMT_OFB_ETH_TYPE (5)
        .... ...0 = Has mask: False
        Length: 2
        Value: ARP (0x0806)
      Pad: 000000000000
  ◢ Instruction
      Type: OFPIT_APPLY_ACTIONS (4)
      Length: 24
      Pad: 00000000
    ◢ Action
        Type: OFPAT_OUTPUT (0)
        Length: 16
        Port: OFPP_CONTROLLER (0xfffffffd)
        Max length: OFPCML_NO_BUFFER (0xffff)
        Pad: 000000000000
```

Figura A.3.11 – *Solicitação OpenFlow do controlador C1 para adição de um fluxo no switch OFS1.*

A.4 – INTENT

A identificação dos *hosts* na topologia de rede pelo controlador ONOS-Controller, pode ser observado na figura A.4.1, ilustrando quando os *hosts* MN1 (198.51.100.100) e MD1 (203.0.113.100) são conectados diretamente ao switch OFS1, respectivamente, nas portas 10, que está associada à Rede Cliente A e na porta 1, associada à Rede Internet.

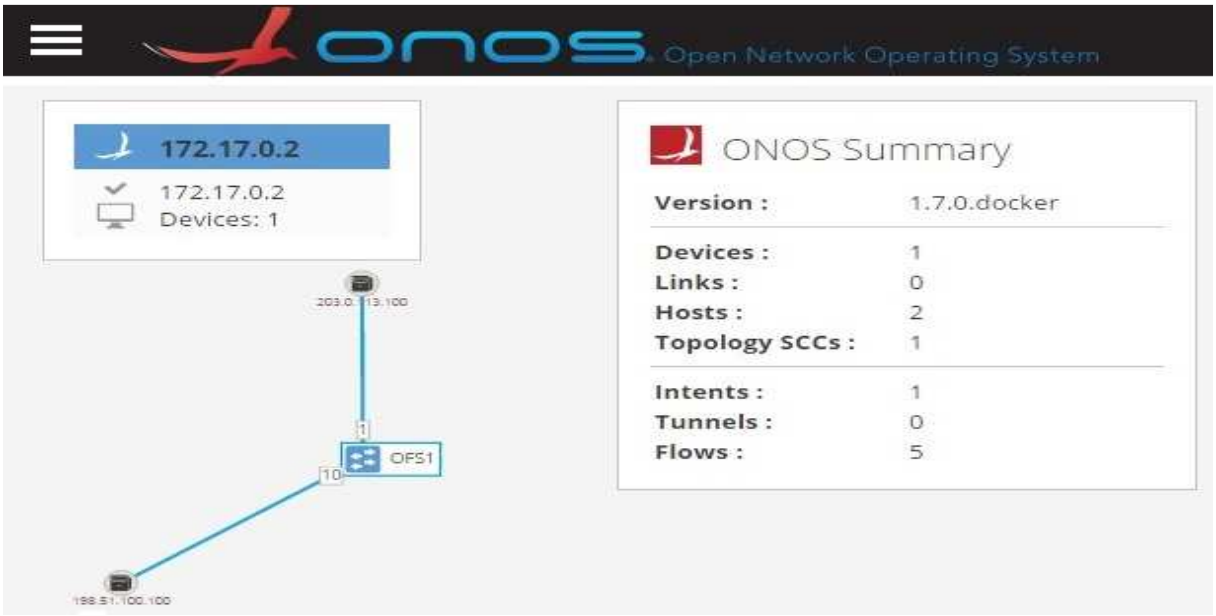


Figura A.4.1 – *Reconhecimento dos hosts conectados a infraestrutura pela interface WEB.*

Estas informações topológicas completas obtidas pelo controlador, incluindo a identificação atribuída aos *hosts*, podem ser observadas na figura A.4.2, que ilustra o resultado para os comandos *summary*, *devices*, *ports* e *hosts* executados na CLI do ONOS-Controller, fornecendo as principais informações referentes à topologia de rede.

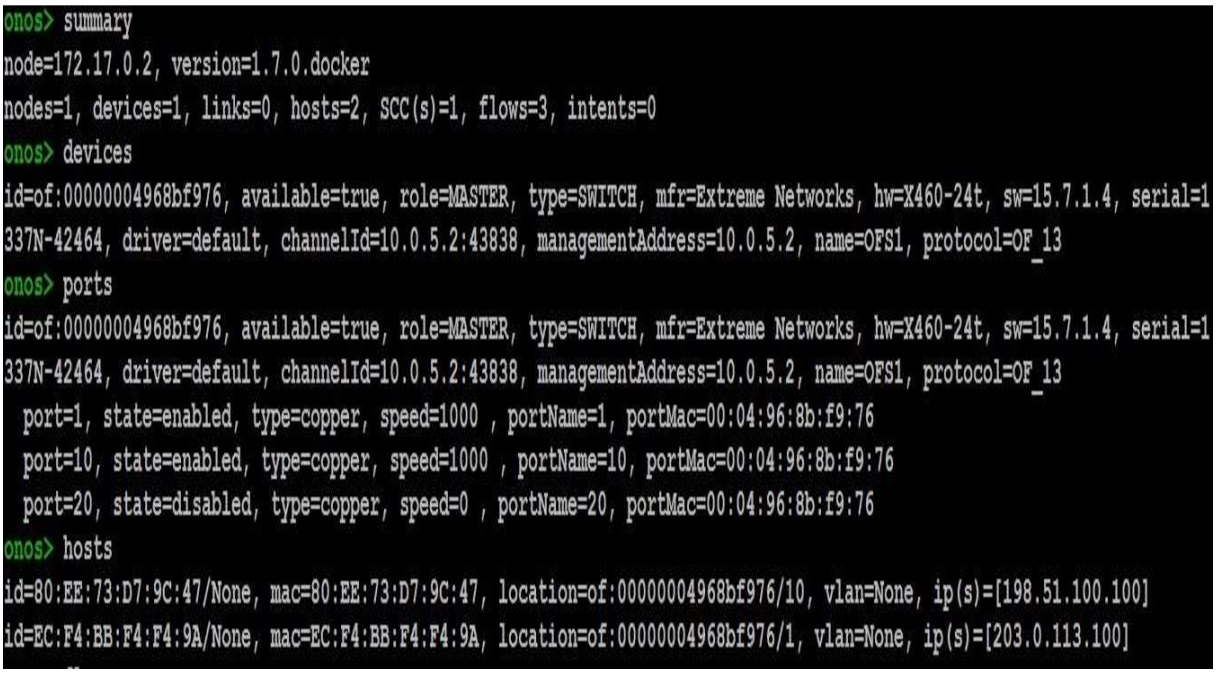


Figura A.4.2 – *Informações de topologia e identificação dos hosts pela CLI do ONOS-Controller.*

A partir desta identificação dos *hosts*, é possível utilizar a *intent* na camada *northbound* para indicar ao controlador a comunicação fim-a-fim que deve ser estabelecida e mantida pela infraestrutura de rede através do comando:

```
#Ativação da intent na camada northbound para prover a mobilidade
add-host-intent [options] “host-id-1” “host-id-2”
```

Podemos observar na figura A.4.3 a *intent* criada no ONOS-Controller para a comunicação entre os *hosts* MN1 e MD1, ilustrando a separação clara entre a indicação da comunicação na camada *northbound*, observada com o comando *intents*, e os fluxos *OpenFlow* criados na camada *southbound*, observados com o comando *flows*.

```
onos> intents
id=0x200000, state=INSTALLED, key=0x200000, type=HostToHostIntent, appId=org.onosproject.cli
    resources=[80:EE:73:D7:9C:47/None, EC:F4:BB:F4:F4:9A/None]
    treatment=[NOACTION]
    constraints=[LinkTypeConstraint{inclusive=false, types=[OPTICAL]}]
    host1=80:EE:73:D7:9C:47/None, host2=EC:F4:BB:F4:F4:9A/None
onos> flows
deviceId=of:00000004968bf976, flowRuleCount=5
    id=5b0000c025f957, state=ADDED, bytes=0, packets=499, duration=79, priority=100, tableId=0, appId=org.onosproject.net.
intent, payLoad=null, selector=[IN_PORT:1, ETH_DST:80:EE:73:D7:9C:47, ETH_SRC:EC:F4:BB:F4:F4:9A], treatment=DefaultTraffic
Treatment{immediate=[OUTPUT:10], deferred=[], transition=None, meter=None, cleared=false, metadata=null}
    id=5b0000fb8e3068, state=ADDED, bytes=0, packets=225563, duration=79, priority=100, tableId=0, appId=org.onosproject.n
et.intent, payLoad=null, selector=[IN_PORT:10, ETH_DST:EC:F4:BB:F4:F4:9A, ETH_SRC:80:EE:73:D7:9C:47], treatment=DefaultTra
fficTreatment{immediate=[OUTPUT:1], deferred=[], transition=None, meter=None, cleared=false, metadata=null}
```

Figura A.4.3 – *Comunicação por intent e os fluxos OpenFlow associados no ONOS-Controller.*

Verificamos na figura A.4.3 que a partir da indicação de comunicação entre os *hosts* MN1 e MD1 através da *intent* com id 0x200000, o controlador gerou duas regras *OpenFlow* para permitir o fluxo IP bidirecional fim-a-fim. Sendo o primeiro fluxo apresentado, id 5b0000c025f957, no sentindo MD1 para MN1 e o segundo fluxo, id 5b0000fb8e3068, no sentido MN1 para MD1. A figura A.4.4 apresenta os dois pacotes *OpenFlow* enviados pelo ONOS-Controller para que o switch OFS1 instale em sua tabela de encaminhamento os fluxos criados relacionados a comunicação indicada.

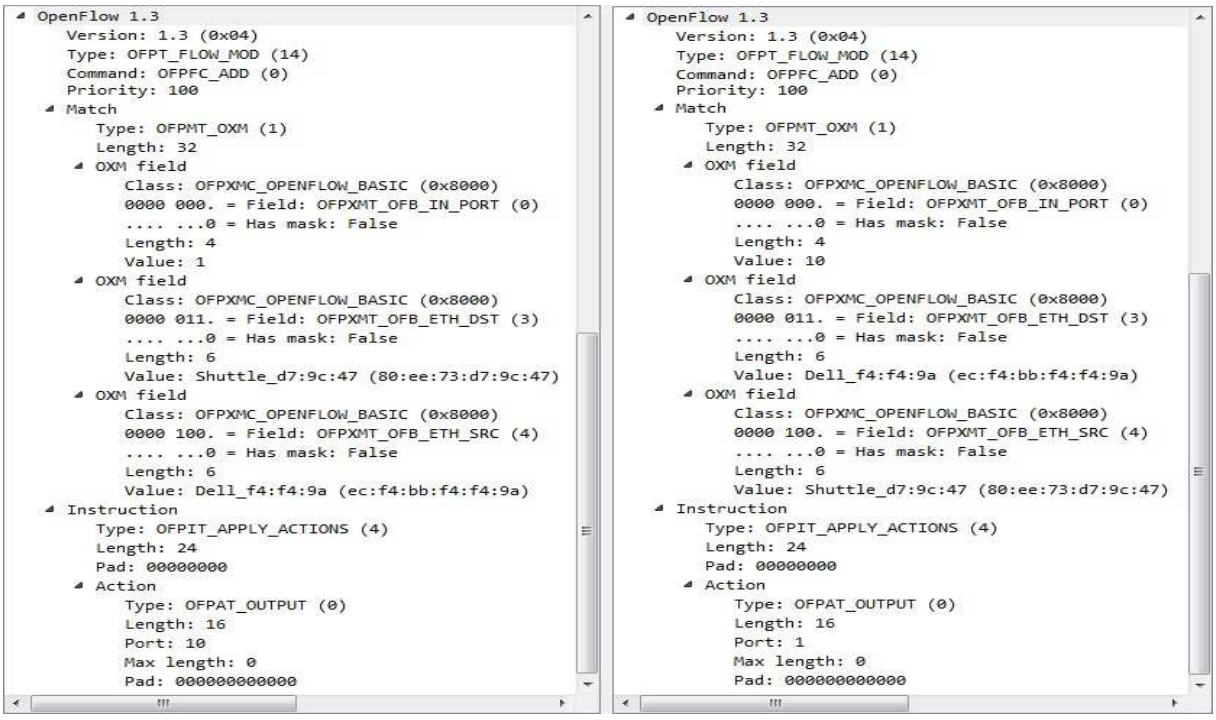


Figura A.4.4 – *Pacotes OpenFlow para adição do fluxo IP gerados pela Intent.*

Podemos constatar na figura A.4.5 que o switch OFS1 instalou corretamente os fluxos em sua tabela de encaminhamento.

```
Total number of OpenFlow flows : 5
Total number of default flows  : 4

Flow name        Type     Duration (secs)  Prio                Packets
---------------- ---- ------------------- ----- --------------------
of_3             ACL                   4   100                     0
    Match   :  Input Port          : 1
               Src MAC             : ec:f4:bb:f4:f4:9a
               Dst MAC             : 80:ee:73:d7:9c:47
    Actions:   output:10
    Cookie                         : 0x5b0000c025f957
of_4             ACL                   4   100                     0
    Match   :  Input Port          : 10
               Src MAC             : 80:ee:73:d7:9c:47
               Dst MAC             : ec:f4:bb:f4:f4:9a
    Actions:   output:1
    Cookie                         : 0x5b0000fb8e3068
```

Figura A.4.5 – *Tabela de fluxos do switch OFS1 com os fluxos da comunicação indicada instalados.*

Por fim, podemos observar na figura A.4.6 a interface WEB do ONOS-Controller indicando a criação do fluxo IP para a comunicação entre os *hosts* MN1 e MD1, assim como o caminho da topologia utilizado.

Figura A.4.6 – *Criação da comunicação entre os hosts e caminho da topologia utilizado.*

Para validarmos o funcionamento da mobilidade IP pela adaptação do plano de dados executado pelo controlador para manter a comunicação indicada, através do reajuste das regras *OpenFlow* de acordo com a mudança ocorrida na topologia, desconectamos o *host* MN1 da porta 10 e o conectamos na porta 20.

Ao realizarmos tal operação, o switch OFS1 informa ao controlador sobre o evento, informando que a porta 10 anteriormente no estado *up* foi para o estado *down*. Como pode ser observado na figura A.4.7, que apresenta parte do pacote *OpenFlow* enviado ao controlador pelo switch.

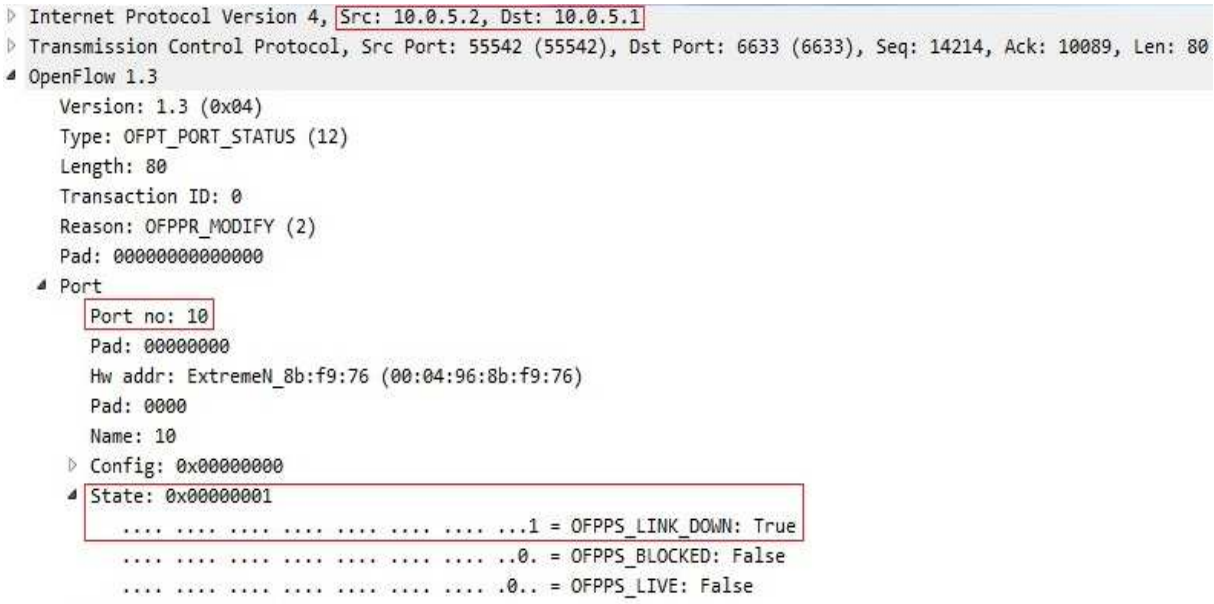


Figura A.4.7 – *Mensagem OpenFlow informando ao controlador sobre a mudança topológica.*

Ao receber a informação de mudança de topologia, o controlador verifica que os fluxos IP bidirecionais criados anteriormente para atender a comunicação indicada, foram afetados, neste caso não sendo possível mais a sua utilização.

Então o controlador recalcula novos fluxos IP bidirecionais para que a infraestrutura de rede possa continuar provendo a comunicação que foi indicada através da *intent.* Onde primeiramente o controlador solicita que o switch OFS1 remova os fluxos IP antigos e depois insira os novos fluxos para comportar as alterações da topologia.

Isso pode ser observado na figura A.4.8 que ilustra as mensagens *OpenFlow* enviadas pelo controlador ONOS-Controller ao switch OFS1 solicitando a remoção de tais fluxos.

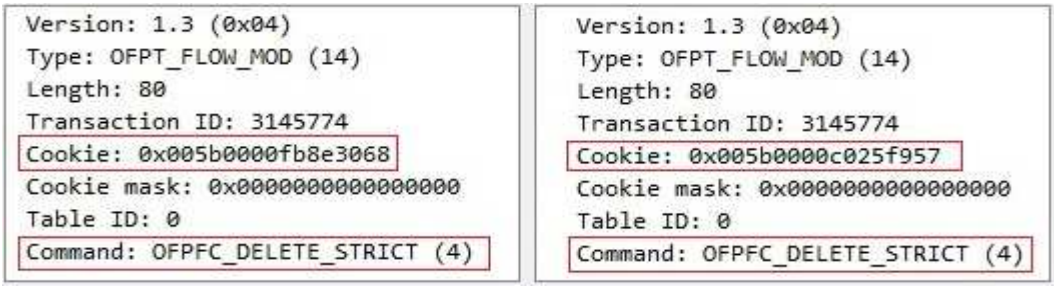


Figura A.4.8 – *Solicitação de remoção dos fluxos pelo controlador ONOS-Controller devido à mudança topológica de rede.*

Logo após conectarmos o *host* MN1 na porta 20 associada à Rede Cliente B e o controlador estar ciente da mudança topológica com a nova posição do *host* MN1, o ONOS-Controller redefine os fluxos *OpenFlow* anteriormente criados para estabelecer um novo fluxo IP bidirecional entre os *hosts* de acordo com a comunicação indicada pela *intent*.

A criação das novas regras *OpenFlow* para atender a nova localização do *host* MN1 pode ser observada na figura A.4.9, ilustrando que a *intent* anteriormente criada foi mantida, *intent* com id 0x200000, mas agora com novos fluxos *OpenFlow* referentes a esta *intent*.

```
onos> intents
id=0x200000, state=INSTALLED, key=0x200000, type=HostToHostIntent, appId=org.onosproject.cli
    resources=[80:EE:73:D7:9C:47/None, EC:F4:BB:F4:F4:9A/None]
    treatment=[NOACTION]
    constraints=[LinkTypeConstraint{inclusive=false, types=[OPTICAL]}]
    host1=80:EE:73:D7:9C:47/None, host2=EC:F4:BB:F4:F4:9A/None
onos> flows
deviceId=of:00000004968bf976, flowRuleCount=5
    id=5b0000c025f957, state=ADDED, bytes=0, packets=431, duration=75, priority=100, tableId=0, appId=org.onosproject.net.
intent, payLoad=null, selector=[IN_PORT:1, ETH_DST:80:EE:73:D7:9C:47, ETH_SRC:EC:F4:BB:F4:F4:9A], treatment=DefaultTraffic
Treatment{immediate=[OUTPUT:20], deferred=[], transition=None, meter=None, cleared=false, metadata=null}
    id=5b0000ea60d722, state=ADDED, bytes=0, packets=15662, duration=75, priority=100, tableId=0, appId=org.onosproject.ne
t.intent, payLoad=null, selector=[IN_PORT:20, ETH_DST:EC:F4:BB:F4:F4:9A, ETH_SRC:80:EE:73:D7:9C:47], treatment=DefaultTraf
ficTreatment{immediate=[OUTPUT:1], deferred=[], transition=None, meter=None, cleared=false, metadata=null}
```

Figura A.4.9 – *Novos fluxos OpenFlow para estabelecer a comunicação indicada na intent.*

Observamos na figura A.4.9 que o primeiro fluxo, com id 5b0000c025f957 responsável pelo fluxo IP unidirecional do MD1 para o MN1, teve a sua ação modificada para encaminhar o tráfego pela porta 20, nova porta de conexão do MN1 e não mais pela porta 10 como anteriormente.

E um novo fluxo, com id 5b0000ea60d722, foi adicionado para permitir o fluxo IP unidirecional do MN1 para o MD1. A figura A.4.10 ilustra os novos pacotes *OpenFlow*

gerados pelo ONOS-Controller para que o switch OFS1 instale os novos fluxos em sua tabela de encaminhamento.

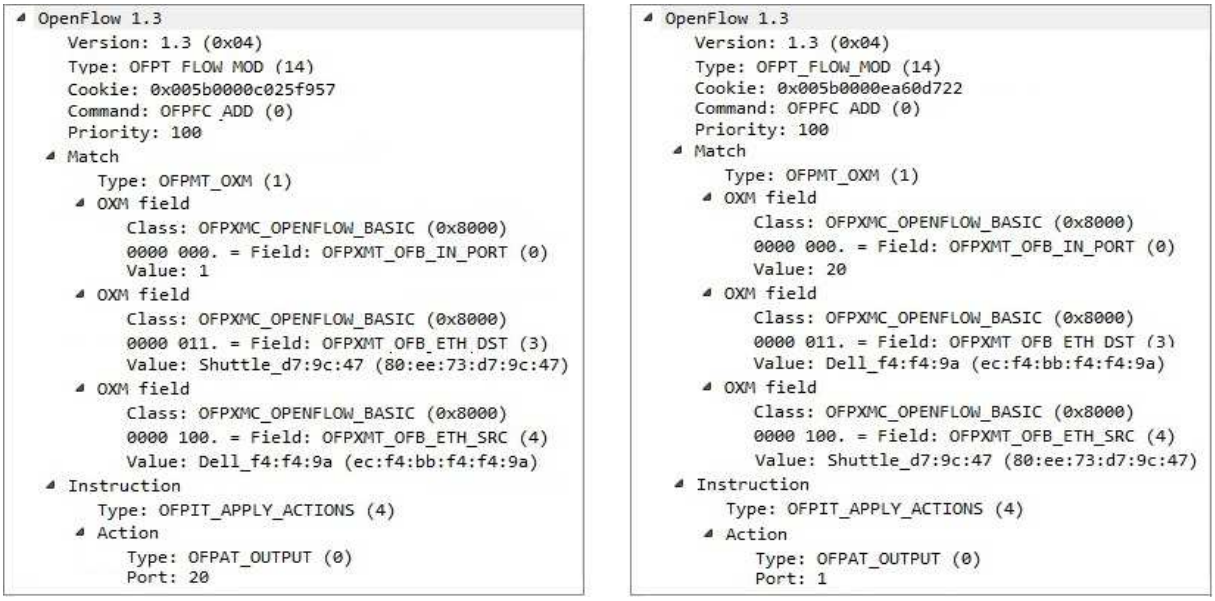


Figura A.4.10 – *Novas mensagens OpenFlow para instalação do fluxo IP depois da mudança.*

Confirmamos que os novos fluxos foram instalados corretamente na tabela de encaminhamento do switch OFS1 observando a figura A.4.11.

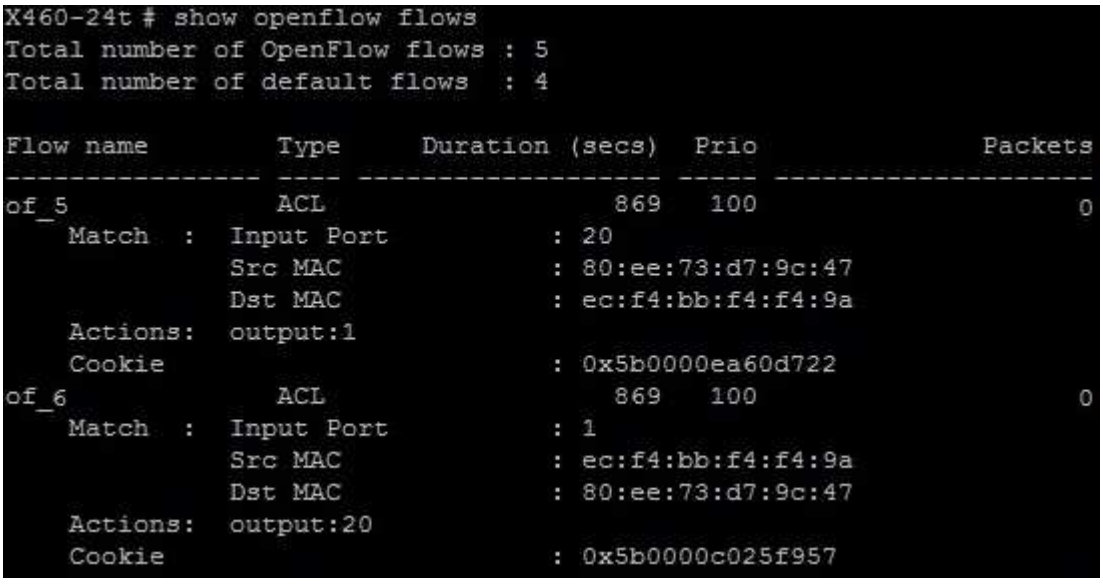


Figura A.4.11 – *Tabela de encaminhamento do switch OF1 com os novos fluxos.*

Concluímos a verificação observando a interface WEB do ONOS-Controller na figura A.4.12, mostrando a identificação do *host* MN1 em sua nova posição e o novo fluxo IP bidirecional utilizado para a comunicação indicada pela *intent.*

Figura A.4.12 – *Nova localização do host MN1 e novo caminho utilizado para comunicação.*

APÊNDICE B – ARTIGO PUBLICADO E APRESENTADO

B.1 – XXXIV SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS (SBRC2016)

O artigo abaixo foi aceito e apresentado durante o SBRC 2016 realizado em Salvador, BA, no dia 03/06/2016, sendo publicado nos ANAIS do Workshop de Pesquisa Experimental de Internet do Futuro.

Arquitetura baseada em SDN para Gerenciamento de Mobilidade Distribuído em Redes IP Móveis Heterogêneas

Rodrygo T. Córdova[1], Paulo R. L. Gondim[1]

[1]Departamento de Engenharia Elétrica – Universidade de Brasília (UNB)
Brasília – DF – Brazil

rodrygo.cordova@rnp.br, pgondim@unb.br

Abstract. *Mobility management applied in the traditional architecture of the Internet has become a great challenge due to the exponential growth in the number of devices that can connect to network and also because of the traditional routing based on the destination address field of the IP header. In this article we propose a network architecture based on Software Defined Networking, which not only allows to get the intrinsic benefits of SDN, but also deals in a simplified and efficient way to the distributed mode of mobility management in heterogeneous access networks, ensuring mainly the continuity of IP session as the user shifts between networks.*

Resumo. *O gerenciamento de mobilidade aplicado na arquitetura tradicional da Internet tornou-se um grande desafio devido ao crescimento exponencial do número de dispositivos que podem se conectar à rede e do roteamento tradicional baseado no endereço de destino do cabeçalho IP. Neste artigo propomos uma abordagem baseada em redes definidas por software, apresentando uma arquitetura de rede que não somente permite obter os benefícios intrínsecos do SDN, mas também lidar de forma simplificada e eficiente com o gerenciamento de mobilidade de modo distribuído em redes de acesso heterogêneas, garantindo principalmente a continuidade de sessão IP durante o deslocamento do usuário entre as redes.*

1. Introdução

A oferta de serviços por operadoras de redes de comunicações móveis tende a envolver soluções completamente baseadas no protocolo IP, seja para o serviço de voz, seja para o serviço de dados [Karimzadeh, 2014]. Por outro lado, sessões de comunicação precisam ter continuidade nessas redes, tornando o gerenciamento de mobilidade IP um fator de enorme importância para as redes de comunicação [Bernardos, 2014].

Atualmente, os padrões de gerenciamento de mobilidade IP do IETF e do 3GPP são dependentes de unidades centrais, que gerenciam os tráfegos de controle e de dados; elaborados de acordo com o roteamento tradicional de pacotes IP, apresentam problemas como roteamento sub-otimizado, baixa escalabilidade, sobrecarga de processamento dos

ativos de transporte no plano de dados e pouca granularidade no serviço de gerenciamento de mobilidade. Adicionalmente, a utilização de redes de acesso heterogêneas (HetNets) impõe dificuldades adicionais a esse gerenciamento.

Este artigo apresenta uma proposta de arquitetura utilizando o paradigma SDN com o protocolo *Open Flow* para DMM (do inglês, *Distributed Mobility Management*) em um ambiente de redes heterogêneas para lidar com os desafios de gerenciamento de mobilidade IP citados.

2. Arquitetura SDN para o Gerenciamento Distribuído de Mobilidade IP

A figura 01 apresenta uma arquitetura SDN básica de referência para o entendimento da proposta, composta por um controlador ONOS (C1), que pode ser implementado em modo cluster hierárquico distribuído, *switches* open-flow, um serviço de mídia na Internet (MD), uma rede de *backbone* de transporte e redes de acesso heterogêneas. O cenário analisado envolve a mobilidade realizada pelo *host* H1, movimentando-se a partir da rede de acesso A (3G-UMTS) para a rede de acesso B (4G-LTE), enquanto mantém suas comunicações ativas com o Media Server e com o *host* H2.

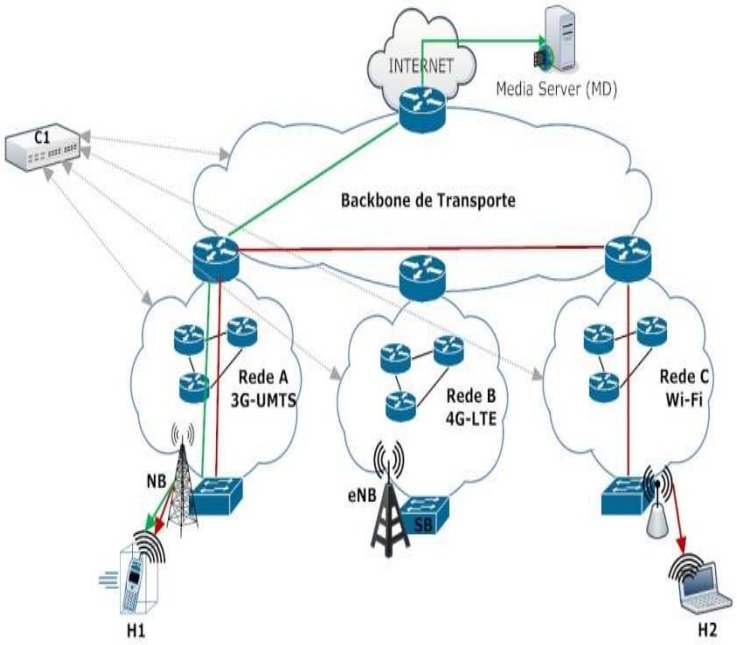


Figura 1. Comunicação do nó móvel H1 na arquitetura SDN

Para manter a continuidade das sessões, o nó móvel H1 mantém o seu endereço IP quando realiza o processo de *handover* da rede A para a rede B. Devido à sua nova localização e à utilização de um endereço IP diferente do escopo de endereçamento de rede atribuído ao seu novo ponto de conexão, a rede B, é necessário que os pacotes destinados ao endereço IP do nó móvel H1 possam ser corretamente encaminhados pela infraestrutura de rede até a sua nova posição.

De modo a prover um uso mais eficiente da infraestrutura disponível e evitar sobrecargas, como técnicas de registro ou de encapsulamento do pacote, a solução utiliza uma abstração modificada da camada *Northbound* chamada de *mobility_intent*, readequando o plano de dados da rede para que a infraestrutura realize o encaminhamento dos pacotes destinados ao nó móvel em mobilidade baseado no fluxo IP bidirecional e não no roteamento tradicional de pacotes pelo endereço IP de destino. O controlador, de posse do conhecimento da topologia física, escolhe o melhor caminho para redefinir os canais de comunicação através

do envio de regras de controle de baixo nível *Open Flow* pela interface *Southbound* para os ativos envolvidos, adequando as suas tabelas de encaminhamento para permitir o fluxo IP bidirecional para o nó móvel em mobilidade. A figura 2 ilustra as principais etapas realizadas no processo de gerência de mobilidade.

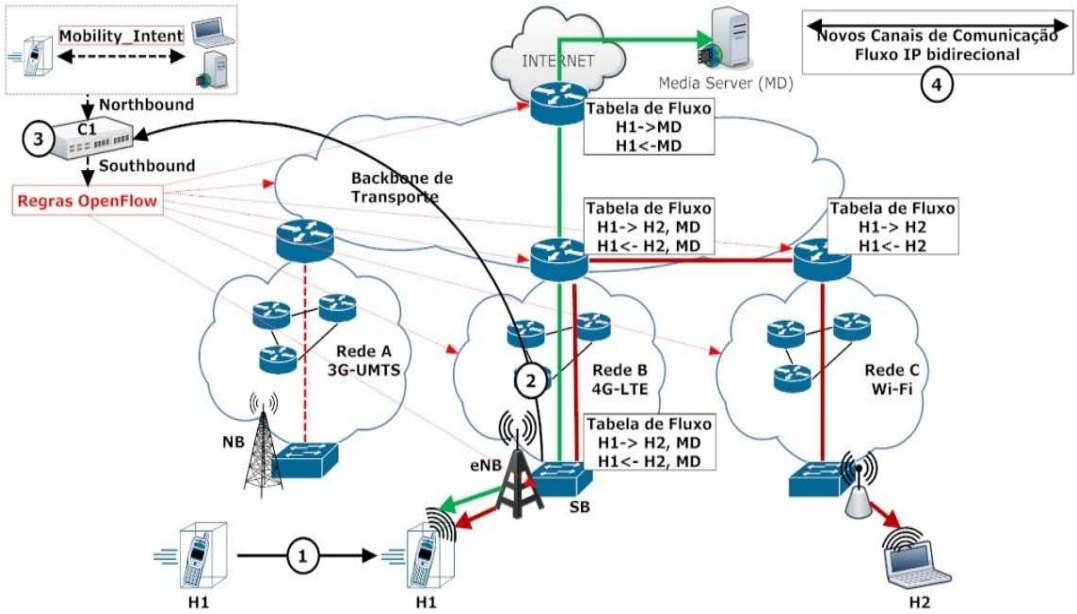


Figura 2. *Handover* e reestabelecimento do canal de comunicação

Em (1) o nó móvel H1 realiza o *handover* da NB para eNB, mantendo o seu IP de origem; em (2) o switch SB identifica a presença deste nó com um IP de rede diferente do seu escopo e informa o evento ao controlador; em (3) o controlador verifica que ocorreu uma mudança de topologia na mobilidade do nó móvel H1, então recalcula um novo caminho para readequar o canal de comunicação, enviando novas regras *Open Flow* e em (4) os novos fluxos IP bidirecionais são estabelecidos na rede, permitindo que os pacotes destinados ao nó móvel H1 sejam corretamente encaminhados a ele baseado no fluxo IP e não no roteamento tradicional IP, não afetando o roteamento geral e as demais comunicações em curso na rede.

3. Resultados obtidos e trabalhos em andamento

Os gráficos apresentados a seguir ilustram os resultados obtidos na comparação analítica relativa ao custo de entrega de pacote (apresentada em [Giust, 2014]) entre a solução proposta SDN-DMM e duas outras soluções, a solução centralizada PMIPv6, padrão do IETF e a proposta DMM baseada no PMIPv6 [Giust, 2014].

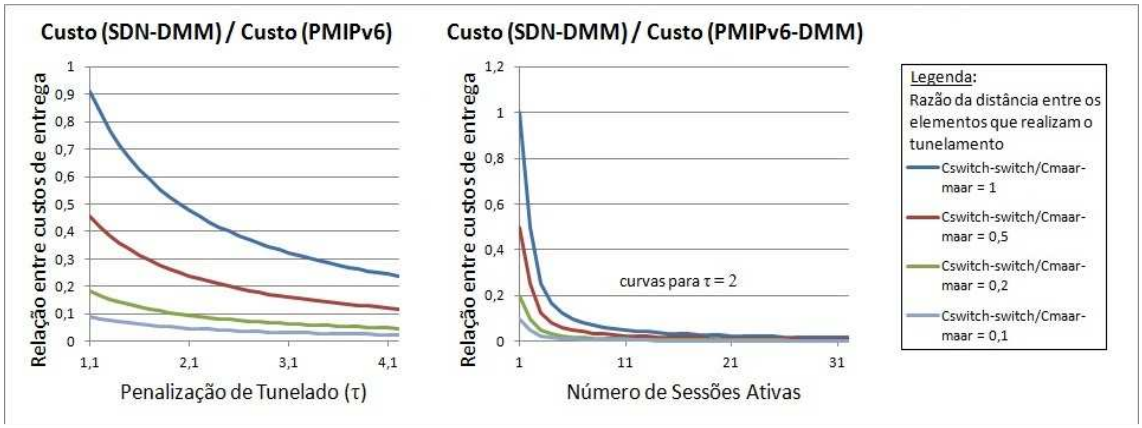


Figura 3. Relação de Custos de Entrega de Pacote

A proposta SDN-DMM apresenta um custo de entrega de pacote inferior às demais soluções. Na figura 3 o gráfico à esquerda compara a proposta com a solução centralizada PMIPv6, apresentando um custo inicial inferior e a redução deste na medida em que aumenta a penalização da entrega de pacotes com processo e transporte de tráfego tunelado (τ), que é utilizado no PMIPv6, e à medida em que aumenta a distância entre os elementos que realizam este tunelamento. Este resultado é devido à proposta SDN-DMM não precisar realizar processos de tunelamento do tráfego destinado ao nó móvel. Na mesma figura, observa-se no gráfico à direita, que a proposta SDN-DMM apresenta uma redução significativa do custo também em relação à solução PMIPv6-DMM, com o aumento do número de sessões ativas e da distância entre os elementos que realizam o tunelamento. Neste caso, o ganho acentuado é devido à proposta SDN-DMM realizar o transporte de pacotes de modo distribuído sem utilizar processos de tunelamento e empregar o conceito de "âncora de mobilidade", que criaria um ponto central para determinado tráfego, permitindo assim que a infraestrutura possa realizar a entrega otimizada sem tunelamento, que representa um ganho significativo.

A tabela 1 apresenta comparação entre soluções publicadas e a solução proposta.

Tabela 1. Comparação entre as soluções existentes e a proposta SDN-DMM.

Solução:	PMIPv6 [RFC 5213]	PMIPv6-DMM [Giust, 2014]	SDN-DMM
Roteamento	É sub-otimizado: tráfego passa sempre por ponto central (LMA)	É sub-otimizado: o tráfego passa por ponto central (A-MAAR)	É otimizado: sempre é definido o melhor caminho para o fluxo IP
Desempenho	Baixo custo de sinalização com sobrecarga de processos nos ativos (*cache* e *binding*) envolvidos e para entrega de pacotes	Relativo custo de sinalização com sobrecarga de processos nos ativos envolvidos (*cahe* e *binding*) e para entrega de pacotes via A-MAAR	Relativo custo de sinalização, sem sobrecarga de processo e com transporte otimizado
Requisitos para os *host*s finais	Não requer modificação	Requer que o *host* final suporte múltiplos endereçamento IP na interface de conexão	Não requer modificação
Complexidade e Arquitetura	Baseado na rede e de forma centralizada	Baseado na rede e de forma distribuída	Baseado na rede e de forma distribuída
Escalabilidade	Não escalável para o crescimento do número de usuários em mobilidade	Escalável	Escalável (implementação em cluster)

Trabalhos em andamento envolvem verificar o custo associado à perda de pacotes e ao possível aumento do custo de sinalização, além de lidar com o desafio nas relações de mobilidade entre diferentes sistemas autônomos. Outros focos incluem *data offloading* e balanceamento de carga em redes heterogêneas.

Referências

Giust F., Bernardos C., Oliva A. (2014). Analytic Evaluation and Experimental Validation of a Network-Based IPv6 Distributed Mobility Management Solution. *In IEEE Transactions on Mobile Computing*, vol. 13, no. 11, November 2014.

Karimzadeh M., Valtulina L., Karagiannis G. (2014). Applying SDN/OpenFlow in virtualized LTE to support Distributed Mobility Management (DMM). In *Proceedings of the 4th International Conference on Cloud Computing and Services Science*, Barcelona, Spain, pp. 639-644.

Bernardos C., Oliva A., Serrano P., et al. (2014). An Architecture for Software Defined Wireless Networking. In *IEEE Wireless Communications*, vol. 21, no.3, pag 52-61.

OnLab Whitepaper (2014). Introducing ONOS - a SDN network operating system for Service Providers. *http://onosproject.org/wp-content/uploads/2014/11/Whitepaper-ONOS-final.pdf* [accessed 12-Nov-2015].

APÊNDICE C – ARTIGO SUBMETIDO

C.1 – JOURNAL: TRANSACTIONS ON EMERGING TELECOMMUNICATIONS TECHNOLOGIES (2016)

O artigo abaixo foi elaborado e submetido ao jornal *Transactions on Emerging Telecommunicatins Technologies*.

SDN-DMM for Intelligent Mobility Management in Heterogeneous Mobile IP Networks

Rodrygo Torres Córdova[1], Paulo Roberto de Lira Gondim[1]

[1]Eletrical Engineering Department – Universidade de Brasília (UNB)
Brasília – DF – Brazil
rodrygo.cordova@rnp.br, pgondim@unb.br

ABSTRACT

Mobility management applied to the traditional architecture of the Internet has become a great challenge due to the exponential growth in the number of devices that can connect to network and the traditional routing based on the destination address field of the IP header. This article proposes a network architecture based on Software Defined Networking, which not only provides the intrinsic benefits of SDN, but also deals with the distributed mode of mobility management in heterogeneous access networks in a simplified and efficient way, ensuring mainly the continuity of IP session as the user shifts between networks and efficient use of the communication network infraestructure.

1. INTRODUCTION

Mobile communication networks have become the main users method of access to the Internet, which has significantly increased the number of mobile devices connected to the global network [12].

As the services offered by operators of mobile communication networks tend towards involving solutions totally based on the IP protocol for both voice and data services [1] and the communication sessions must be continued, the IP mobility management has become fundamental for communication networks, so that such an increase can be supported. [18].

The number of mobile devices connected to the network has increased exponentially, due to the expasion in both availability and use of applications that require mobility, especially real-time collaborative tools, and the current mobility management solutions are not adequate for satisfactorily meeting the requirements imposed on the infrastructures of communication networks.

The IETF IP mobility management standards, as MIPv6 [R7] and PMIPv6 [R8] depend on central units that manage both control and data traffic, elaborated according to the traditional routing of IP packets. They also pose problems, as sub-optimized routing, low scalability, overload of processing in network devices and low granularity in the mobility management service.

Additionally, the use of heterogeneous access networks (HetNets) imposes other difficulties to such a management, as the integrated management of resources and the soft transitions among networks.

The mobility management faces a constant challenge regarding the communication network efficiency with no increase in its complexity. The addition and use of new protocols, signaling messages or processes that cause overhead due to the encapsulation and control traffic are examples of how the mobility management directly impacts on the OPEX (Operational Expenditure) and CAPEX (Capital Expenditure) of a communication network.

An alternative for dealing with the intrinsic problems of centralization and costs in the mobility management is the new concept, called distributed mobility management (DMM) \cite{R9}. Its main characteristic is the clear separation between the data plane and the control plane. The data plane is distributed along the equipments on the access network edge for approximating the mobile agent to the end user, implementing a flatter mobility approach in the network [10]. Therefore, the traffic forwarded to the mobile node (MN) does not cross a specific central point in the network, i.e., the mobility anchor, as it occurs in centralized solutions. In DMM, the traffic is forwarded by the mobile agents nearest the user.

The DMM solutions can be categorized into two types, depending on the distribution level of the control plane [R4,R10,R11]:

- Partially distributed: the data plane is completely distributed among the network elements and the control plane is centralized in control points in the network; and
- Fully distributed: both data plane and control plane are completely distributed among the network elements and there exists no central entity of control.

This manuscript proposes an architecture that uses the SDN paradigm with the OpenFlow protocol for the implementation of a network-based DMM solution in an environment of heterogeneous networks for dealing with the above-mentioned challenges of IP mobility management.

The remainder of the manuscript is organized as follows: Section II describes our proposal for DMM, called SDN-DMM (Software-Defined Network for Distributed Mobility Management); Section III addresses the related work on network-based DMM; Section IV reports the analytical evaluation of the proposals; Section V provides the results and discussions. Finally, Section VI presents our conclusions, reports on the ongoing studies and suggests future work.

2. SDN-DMM: AN SDN ARCHITECTURE FOR THE MANAGEMENT OF DISTRIBUTED MOBILITY

Traditional IP mobility solutions, as those standardized by IETF, are somehow supported on the traditional routing of IP packets and specific network devices for playing key roles in the mobility and treatment of communication processes, as registration processes and techniques of packet encapsulation.

Such solutions cause overhead in the infrastructure, regarding the network devices involved and the data transportation, which increases the complexity for the operation and

maintenance of the network. These network devices must perform not only their main function, e.g., packet routing in routers, but also extra functions, as tunneling, proxy actions, packet analysis, among others.

The use of techniques of data encapsulation and sub-optimized routing based on the address of the IP packet is inappropriate for the mobility scenario and causes the inefficient use of the communication network infrastructure, reduction in the effective rate of data transmitted and overload of certain communication links, whereas others are underutilized. Furthermore, such negative points are potentialized by the expected increase in the number of mobile devices connected to the network (due, for example, to the Internet of Things - IoT).

This section proposes a network-based partially Distributed Mobility Management solution based on the SDN paradigm for dealing with the challenges addressed. It uses the OpenFlow protocol and ONOS controller to provide IP mobility management in heterogeneous access networks focusing on the continuity of IP sessions.

The proposal aims at a reduction in the general complexity of the system and the operational (OPEX) and investment (CAPEX) costs through the separation between the control and data planes of the communication and the treatment of packets according to the IP flows. The following aspects must be satisfied:

- Continuity of the IP session;
- Optimized routing;
- Distributed architecture;
- Transparency for the mobile node;
- Scalability;
- Performance;
- Low Complexity.

Fig. 01 shows a basic reference SDN architecture composed of an ONOS (C1) controller, which can be implemented in a distributed hierarchical cluster mode for the elimination of single failure points and treatment of the scalability aspect of the control plane, OpenFlow switches, a media service in the Internet (MD), a transport backbone network and heterogeneous access networks.

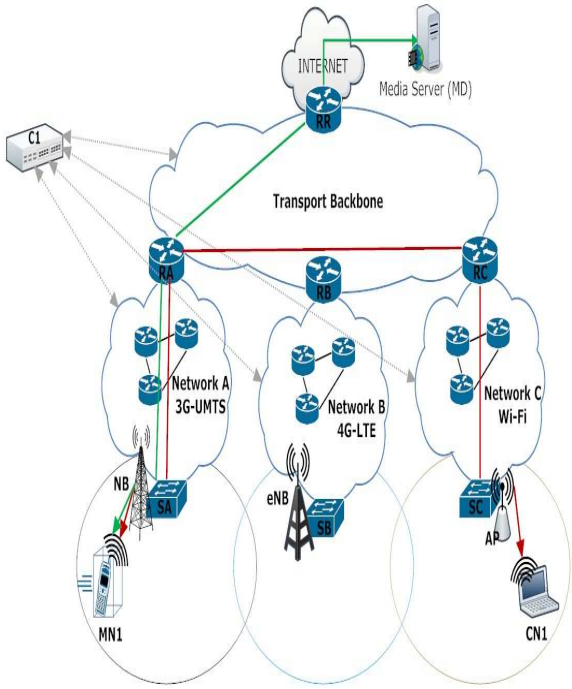


Figure 01. SDN-DMM architecture.

The general scenario analyzed involves the mobility performed by mobile node MN1, which moves from access network A (3G-UMTS) to access network B (4G-LTE) and then to access network C (WiFi), while keeping its communications with the Media Server and correspondent node CN1. The specific situations analyzed in this mobility process are shown in Fig. 02, where:

- s_1 = mobile node MN1, media Server MD and corresponding node CN1 are connected to their origin networks and establish communication normally through the network infrastructure (no mobility has occurred);
- s_2 = as the communications established in the previous situation are in progress, mobile node MN1 changes its

connection point from network A (3G-UMTS) to network B (4G-LTE), which causes the mobility process;

- s_3 = mobile node MN1 performs another handover, changing its connection from network B (4G-LTE) to network C (WiFi), while the sessions are still in progress.

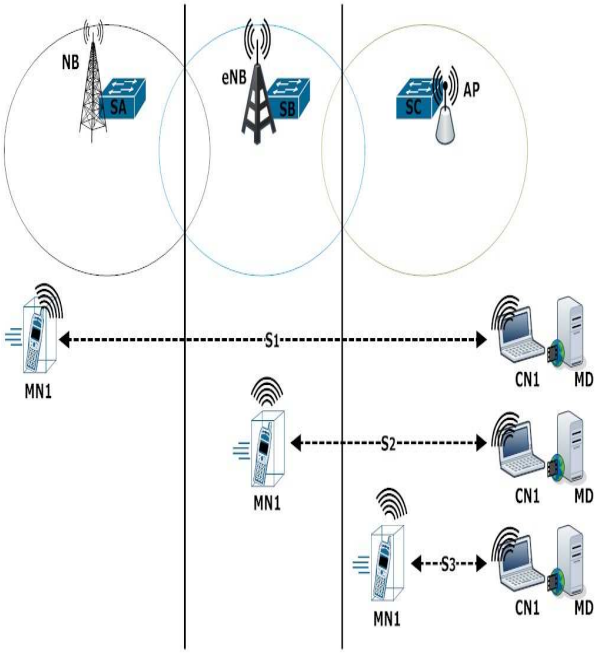


Figure 02. Connection scenarios.

The end-to-end communication between hosts in scenario s_1 is established in the network through bidirectional IP flows, i.e., a flow from MN1 to CN1 and another from CN1 to MN1 (represented in Fig. 01 by the red flow) and a flow from MN1 to MD and another from MD to MN1 (represented in Fig. 01 by the green flow). Such flows are established by OpenFlow rules that insert corresponding entries in the forwarding table of the network devices involved in the formation of the flow path. The process of establishment of the bidirectional IP flows through OpenFlow rules is shown in Fig. 03.

In such a scenario, switch SA, which has received an IP flow from mobile node MN1 to be forwarded to correspondent node CN1 and media server MD, identifies corresponding entries to such flows in its forwarding table and takes the appropriate actions. It forwards the packets that belong to both flows to router RA, which analyzes its forwarding table and sends the flow addressed to MD to router RR and the flow addressed to CN1 to router RC.

In scenario s_2, when mobile node MN1 perfoms the handover process from NodeB (NB), inserted in the context of network A, to eNodeB (eNB), inserted in the context of network B, the mobility process is started.

For the continuity of the IP sessions, mobile node MN1 keeps its IP address during the handover process from network A to network B for avoiding changes in the end-to-end communication, mainly in relation to layer 3 and higher layers, as the replacement of TCP/IP sockets. Due to its new location and use of an IP address different from the addressing scope of the network assigned to its new connection point, i.e., network B, the packets addressed to the MN1 IP address must be correctly forwarded by the network infrastructure to its new position.

For a transparent process for the final hosts, the communication network must self-adapt, i.e., provide a new communication channel that enables the end-to-end communication among the hosts for a new scenario. However, it must not affect the other ongoing communications in the network and their routing.

For a more efficient use of the available infrastructure and avoidance of overloads due to techniques of registration or packet encapsulation, the solution uses a modified abstraction of the Northbound layer, called mobility intent. It readjusts the data plane of the network, so that the infrastructure can forward the packets addressed to the mobile node based on the bidirectional IP flow, rather than on the traditional packet routing by the destination IP address.

An intent can be considered a communication intention performed at a high level; it informs the controller about

which network points should be established a communication path [R6].

After controller C1 has been known on both intention (mobility intent) and physical topology, it chooses the best way for redefining the communication paths by sending low level control rules through the Southbound interface (OpenFlow rules) to the network equipments involved and readjusts its forwarding tables for new bidirectional IP flows for the mobile node in its new location.

In the solution when switch SB has identified the mobile node or an IP flow whose original address is different from network B, it informs such an event to controller C1, which, after having detected a change in the topology involving the location of mobile node MN1, readjusts the data plane of the communication network changing the OpenFlow rules for inserting new entries in the forwarding tables of the network equipments involved in the changes. Then a new set of links for the creation of a new bidirectional IP flow between the hosts is reestablished. Fig. 03 shows the main stages of the mobility management process, where:

1. Mobile node MN1 performs handover from NB to eNB, keeping its original IP address;
2. Switch SB identifies the presence of the mobile node with a IP network address different from its scope and informs the event to controller C1;
3. Controller C1 detects a topology change in the mobility of mobile node MN1, therefore, it recalculates a new path and sends new OpenFlow rules for readjusting the communication links used;
4. The new bidirectional IP flows are established in the network, which enables the packets addressed to mobile node MN1 to be correctly forwarded according to the IP flow, rather than to the traditional IP routing. Therefore, the general routing and the other ongoing communications in the network are not affected.

In this scenario, the traffic between hosts is routed according to the new bidirectional IP flow established by the change in the data plane performed by the controller. The packets sent to mobile node MN1 with an address of network A, can be directly delivered to MN1 even when it is connected to network B, in an optimized and transparent way and with no overhead addition for the data transportation in the network.

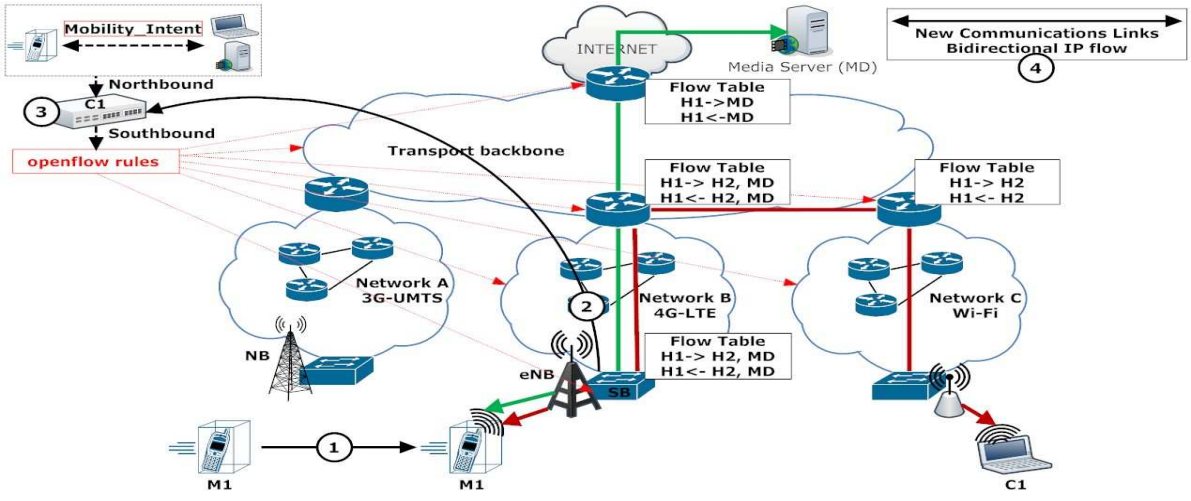


Figure 03. Handoff and establishment of the bidirectional IP flow in s_2.

The above-mentioned process is repeated when mobile node MN1 performs another handover by connecting to the access point AP (Fig. 03), inserted in the context of network C. Switch SC informs controller C1 on the new change in topology and controller C1 reestablishes the bidirectional IP flow between the hosts. The communication between mobile node MN1 and correspondent node CN1 now ocurrs directly in network C.

3. RELATED WORK

This section presents the main concepts and the functioning of two recent IETF drafts for DMM.

3.1 A Fully Distributed Solution for DMM [16]

Such an IETF draft (Draft-jaehwoon-dmm-pmipv6 [R16] presents a network-based fully DMM proposal based on the PMIPv6 protocol, in which all control and data plane functions are distributed among the mobile access gateways (MAGs). No central unit, as the local mobility anchor (LMA), is responsible for the processes of signaling and forwarding of traffic. Such responsibilities are distributed among the entities located on the network edges.

The protocol assigns a network prefix, called PREF (in a supernet concept) to the mobility domain, where a different subnetwork that belongs to PREF is allocated for each MAG of the domain. Although each MAG is responsible for a specific subnetwork, all MAGs inform the PREF prefix as the network to which they belong through its mobility service interfaces.

When the mobile node enters the domain for the first time, it is provided with an IP address of the subnet corresponding to the MAG it is connecting to and PREF prefix is indicated as the network address. Once the mobile node change its connection point to a new MAG and receive new router advertisement (RA) messages with the option information field containing the same network prefix PREF, it considers it is still connected to the same network and keeps its addressing according to what has been established in the connection with the first MAG.

The original MAG from the mobile node, i.e., the MAG responsible for the IP address used by the mobile node that performed a handover, named anchor-MAG (A-MAG) from now on for future references, acts as an LMA, performing functions such as creation and maintenance of the binding cache entry (BCE), buffer and packet redirection through the tunnels established.

The signaling process starts when an MAG has identified the movement of a mobile node within the domain. In the association between the mobile node and MAG, the latter sends an RA message indicating PREF prefix as the network to which the mobile node will connect. If MAG receives a DHCP request from the mobile node, it considers the mobile node is entering the domain for the first time, i.e., it is not a handoff.

However, if MAG receives any other packet with an source IP address of PREF prefix that does not belong to the scope of its subnetwork, it detects the mobile node is performing a handoff and starts the signaling process. This MAG is called, for future references, as serving-MAG (S-MAG), which is the MAG that provides connectivity to a mobile node that has performed a handover.

S-MAG sends a distributed proxy binding update (DPBU) message to A-MAG to inform the new location of the mobile node and creates a binding update list (BUL) for the establishment of a tunnel between them. When A-MAG receives the DPBU from S-MAG, it creates a BCE entry for the mobile node and responds with a distributed proxy binding acknowledgement (DPBA) message for the establishment of the bidirectional tunnel to be used for the redirection of all traffic of the mobile node.

As A-MAG cannot know when a mobile node will perform another handover, S-MAG sends a distributed proxy binding release update (DPBRU) message to A-MAG informing on the disconnection of the mobile node when it notices the mobile node has been or will be disconnected. After receiving the DPBRU, A-MAG starts to store the packets addressed to the mobile node and responds with a Flush message to S-MAG, which cleans the mobile node registrations from its BUL and retransmits the Flush to A-MAG, which cleans the current registrations of the BCE and waits for a new DPBU from the new S-MAG for the establishment of the new tunnel.

3.2 A Partially Distributed Solution for DMM [17]

This proposal presents a network-based partially distributed DMM solution (Draft-bernardos-dmm-pmip [17]) based on the PMIPv6 protocol, in which a central unit, called central mobility database (CMD) performs the main functions of control plane, whereas the forwarding of traffic, data plane, is the responsibility of units called mobility anchor and access router (MAAR).

CMD assumes the LMA figure, however, it does not forward traffic to the mobile node, i.e., it does not participate of the data plane operations. It is the central entity of the control plane and manages sessions and registration of IP, location and binding cache. MAG is replaced by network entity MAAR, which performs LMA functions, as treatment of aspects of control plane, as local binding cache, and aspects of the data plane, as supply of connectivity to the mobile node.

Each MAAR manages a set of nonnsuperposed specific IP prefixes assigned to the mobile node when it is connected to MAAR. At each new connection of the mobile node to a new MAAR, a new IP address is allocated in the mobile node, under the responsibility of MAAR. The home network prefix (HNP) address concept of PMIPv6, in which only one IP address is allocated to the mobile node for all mobility domain, is changed to the address concept called local network prefix (LNP).

When the mobile node connects to MAAR for the first time, MAAR creates a unique identification for the mobile node within the mobility domain (MN-ID) and stores an exclusive LNP address to be assigned to the mobile node. MAAR inserts such information in its local binding cache entry (BCE) table and sends it through a standard proxy binding update (PBU) message to CMD. The latter includes the information in its BCE table, whose scope is global for the mobility domain, creates a mobility session for the mobile node and responds to MAAR with a standard proxy binding acknowledgement (PBA) message. After MAAR has received the PBA, it sends an RA message that includes the LNP previously stored for the mobile node, which starts to use the IP address and the current MAAR is referenced as Serving-MAAR (S-MAAR), i.e., the MAAR that is providing connection to the mobile node.

When the mobile node performs a handover for the new MAAR, the latter follows the same procedures described and allocates a new NLP for the mobile node. However, when it sends the information to the CMD, which verifies the mobile node was previously connected to another MAAR using an NLP address allocated by MAAR, called, respectively, anchor-MAAR (A-MAAR) and previous-LNP (pLNP), CMD updates the entries in its BCE table and sends messages to A-MAAR containing the new location of the mobile node, and to the current MAAR, i.e., S-MAAR, about the pLNP used and the address of the responsible A-MAAR. At this moment, S-MAAR and A-MAAR establish an IP-in-IP tunnel for the redirectioning of the traffic related to pLNP, which guarantees the continuity of the IP sessions that were in progress when the mobile node performed the handover for a new MAAR.

Therefore, the mobile node has two IP addresses, i.e., the new address provided by S-MAAR, the LNP, and the previous address provided by A-MAAR, i.e., the pLNP. All traffic related to pLNP is sent through the tunnel between the MAARs and all traffic related to LNP is treated directly by S-MAAR, without the use of encapsulation techniques, through which new IP sessions are established with such address.

4. ANALYTICAL EVALUATION

This section addresses the analytical evaluation of the draft-Jaehwoon [R16], draft-Bernardos [R17], and SDN-DMM proposals conducted with metrics of signaling and handover latency costs [R15]. A comparative table of the aspects of routing, performance, requirements for the end hosts and architecture/complexity is also provided.

4.1 Signaling cost

The signaling cost is related to the number of control messages generated specifically for supporting the mobility process and keeping the continuity of the mobile node sessions. Its three main components are [R15]:

i. C_{update}= Cost of the location update of the MN with binding update messages;
ii. $C_{refresh}$ = Cost of periodical updates;
iii. C_{de-reg}= Cost of deregistration due to the execution of a new mobility within the domain.

The total signaling cost C_{SIG}, provided in [15], is given by:

$$C_{SIG} = \mu_{SN}(C_{update} + C_{refresh} + C_{de-reg}) \quad (1)$$

where μ_{SN} represents the subnet crossing rate performed by the mobile node. The individual cost of each of the three components is given by the sum of C_{X-Y} cost, which is the symetrical cost for the transmission of a control packet between two, X and Y, network units, and PC_Xcost, which is the cost involved in the processing of the control packet by network unit X [15].

The signaling costs for each proposal are evaluated below:

- Draft-Jaehwoon [16]

When a mobile node performs a handover, the new S-MAG sends a DPBU message to A-MAG, which responds with a DPBA message. Such messages are exchanged for informing the new location of the mobile node and establishing the tunnel for the redirectioning of the traffic, therefore, C_{update} cost is defined by:

$$C_{update}^{\text{Draft-Jaehwoon}} = 2C_{SMAG-AMAG} + PC_{AMAG} + PC_{SMAG} \quad (2)$$

As the periodical updates are performed as in PMIPv6, the same messages described above are exchanged between the MAGs at an$R_{BCE} = \frac{1}{\mu_{SN} T_{BCE}}$ rate, where T_{BCE} is the lifetime of a BCE entry [15]. Therefore, $C_{refresh}$ is given by:

$$C_{refresh}^{\text{Draft-Jaehwoon}} = R_{BCE}(2C_{SMAG-AMAG} + PC_{AMAG} + PC_{SMAG}) \quad (3)$$

Regarding C_{de-reg} cost, when the mobile node moves to a new MAG, DPBRU/DPBRA and Flush messages are exchanged for the removal of the old registrations from the tables. Therefore, the deregistration cost due to the new mobile node location is defined by:

$$C_{de-reg}^{\text{Draft-Jaehwoon}} = 4C_{SMAG-AMAG} + 2PC_{AMAG} + 2PC_{SMAG} \quad (4)$$

- Draft-Bernardos [17]

The following expressions are used for the signaling cost components [15]:

$$C_{update}^{\text{Draft-Bernardos}} = (N_{PR} + 1)(2C_{CMD-MAAR} + PC_{CMD} + PC_{MAAR}) \quad (5)$$

$$C_{refresh}^{\text{Draft-Bernardos}} = R_{BCE}(2C_{CMD-MAAR} + PC_{CMD} + PC_{MAAR}) \quad (6)$$

$$C_{de-reg}^{\text{Draft-Bernardos}} = 4C_{CMD-MAAR} + 2PC_{CMD} + 2PC_{MAAR} \quad (7)$$

- SDN-DMM

In the SDN-DMM proposal the OpenFlow switch sends a message to the controller after detecting the mobile node has performed a handover. The controller verifies the change that occurred in the topology, recalculates a new path and sends OpenFlow messages, so that the switches involved establish a new communication path through bidirectional IP flows.

Similarly to the scenario between the movement of the mobile node in draft-Bernardos, which results in the allocation of new LNP addresses, and the switches affected in the data plane readjustment, the number of switches involved (except the one that identified the mobility) can be represented by N_{PR}, where C_{update} cost is represented by

$$C_{update}^{\text{SDN-DMM}} = 2C_{CTR-OFS} + PC_{CTR} + PC_{OFS} + (N_{PR})(C_{CTR-OFS} + PC_{OFS}) \quad (8)$$

where CTR indicates the cost related to the controller and OFS indicates the cost related to the OpenFlow switch. As the switch that received a message from the controller for adjusting the data-plane responds to the controller only if a problem has occurred for the establishment of the IP flow requested, the processing of the controller and the cost for the transmission of this related message are not added to C_{update} cost.

Regarding the periodical updates and the deregistration processes, the SDN-DMM approach does not address the concepts of registration and binding cache used for storing information on location for the establishment of tunnels that redirect packets when the mobile node performs a handover.

Through the knowledge of physical topology, the controller only adjusts the data plane of the network on demand according to the movement of the mobile node. The entries in the forwarding tables for the IP flows, due the handover of the mobile node, in the OpenFlow switches are inserted and removed in two ways, i.e., during the update process, when the controller sends messages to the switches involved for supporting mobility, by inserting or removing entries, and in the expiration of entries in the switches that no longer forward traffic related to the bidirectional IP flow established. Therefore, signaling cost $C_{SIG}^{\text{SDN-DMM}}$ is equivalent to $C_{update}^{\text{SDN-DMM}}$.

4.2 Handover latency cost

This cost is given by the time spent from the disconnection of the old network to the ending of the mobility procedure that enables the mobile node to obtain global connectivity of the ongoing IP sessions again.

In general, the handover operation can be divided into three stages, according to [15]:

iv. t_{L2} = time interval between the disconnection of the previous link and the connection to the new link;

v. t_{auth} = time interval spent on the authentication and authorization processes due to the new association;

vi. $t_{binding}$ = time interval related to the mobility phase conducted according to the responsible protocol, through which functions, as bindings, routing adjustment and establishment of tunnels are performed.

As t_{L2} and t_{auth} are independent of the mobility protocol employed, only componet $t_{binding}$ is considered for an analysis of the handover latency cost ($C_{handover}$) generated for each protocol, therefore,

$$C_{handover} = t_{binding} \quad (09)$$

$t_{binding}$ (Eq.10) depends on both operation and approach for the mobility in each proposal. However, it can be understood as the sum of the times necessary for the transmission and processing of the control messages.

$$t_{binding} = \sum RTT_{X-Y} + T_{proc}^{X} \quad (10)$$

where RTT_{X-Y} is the time for the packet exchange between the network entities X and Y; and T_{proc}^{X} is the processing time of a packet by entity X.

The handover latency costs for each proposal are evaluated below:

- Draft-Jaehwoon [16]

The $t_{binding}$ process starts by S-MAG announcing the PREF network prefix through an RA message for the recently connected mobile node, which, after having visualized the PREF prefix, considers it is still connected to the same network, maintains its IP address and normally transmits its packets.

After S-MAG has received the packets, it verifies the mobile node has performed a handover process, therefore, it sends a DPBU message, so that A-MAG is informed about the new location of the mobile node. A-MAG responds to S-MAG with a DBPA message and establishes a tunnel between them for the addressing of the mobile node packets. At this moment, the mobile node is globally reconnected and can send and receive the packets through the tunnel established between the MAGs.

Therefore, the handover latency cost is given by:

$$C_{handover}^{Draft-Jaehwoon} = RTT_{MN-MAG} + RTT_{MAG-MAG} + T_{proc}^{MN} + 3T_{proc}^{MAG} \quad (11)$$

where RTT_{MN-MAG} is the time of the sending of the RA message from MAG to MN and return of the IP traffic from MN to MAG, $RTT_{MAG-MAG}$ is the time for the DPBU/DPBA message exchange between the MAGs, T_{proc}^{MN} is the processing time of the RA message by the MN, and T_{proc}^{MAG} is the MAG processing time performed once, when S-MAG receives the traffic, and twice, in the communication between the MAGs.

- Draft-Bernardos [17]

The mobility phase starts when the mobile node is connected to the new S-MAAR and requests a new IP address. After S-MAAR has registered such an IP with CMD, it is informed about the existence of other prefixes that are still being used by the mobile node and have active sessions. Therefore, they must be treated by the mobility process, so that such IP sessions can be continued. S-MAAR establishes a tunnel with each A-MAAR for addressing the traffic corresponding to those prefixes.

Therefore, the handover latency cost comprises the sending of the RS message from the mobile node to S-MAAR, which sends a PBU message to CMD, which responds to S-MAAR with an extended PBA message and sends an extended PBU message to the other A-MAAR, and finally the message exchange between MAARs for the establishment of the tunnel.

As all MAARs simultaneously receive and process the extended PBA/PBU messages sent by CMD, the handover latency cost is given by:

$$C_{handover}^{Draft-Bernardos} = \frac{1}{2}RTT_{MN-MAAR} + RTT_{CMD-MAAR} + RTT_{MAAR-MAAR} + 3T_{proc}^{MAAR} + T_{proc}^{CMD} \quad (12)$$

where ½ $RTT_{MN-MAAR}$ is the time of the RS message from the mobile node to

MAAR, $RTT_{CMD-MAAR}$ is the time of the PBU/PBA message exchange between CMD and MAAR, $RTT_{MAAR-MAAR}$ is the time between the message exchange between S-MAAR and A-MAAR for the tunnel establishment, T_{proc}^{MAAR} is the MAAR processing time, which occurs when it receives the RS message from mobile node, the extended PBA message from CMD, and the message for tunnel establishment from MAAR, and T_{proc}^{CMD} is the CMD processing time when it receives the PBU message from MAAR.

- SDN-DMM

Soon after the mobile node has connected to the new wireless network, maintaining its original IP address, it starts to normally transmit the packets. When the OpenFlow switch identifies the mobile node has performed a handover, it sends an OF message to the controller about the event and awaits instructions.

The controller calculates a new path through which the new bidirectional IP flow is established for the correct addressing of the mobile node packets due to its new location and sends to affected switches an OF message, so that they insert the corresponding IP flow in the forwarding table.

At this moment, the packets related to the mobile node are correctly forwarded by the network infrastructure according to the bidirectional IP flow established and global connectivity is again provided, which enables the continuity of the IP sessions.

As all OpenFlow switches simultaneously receive and process the messages sent by the controller, equation (13) gives the handover latency cost for the SDN-DMM approach:

$$C_{handover}^{SDN-DMM} = \frac{1}{2}RTT_{MN-OFS} + RTT_{CTR-OFS} + 2T_{proc}^{OFS} + T_{proc}^{CTR} \quad (13)$$

where $½RTT_{MN-OFS}$ is the time of the packet-delivery from the mobile node to the OpenFlow switch, $RTT_{CTR-OFS}$ is the time of the packet-exchange between the controller and the OpenFlow switch, T_{proc}^{OFS} is the processing time of the OpenFlow switch performed when it receives the packet from the mobile node and the OpenFlow message from the controller, and T_{proc}^{CTR} is the processing time of the controller.

Table 01 shows a comparative summary of the main aspects of proposals related to DMM.

Tabela 01 – Comparison among the proposals

Proposals:	Draft-Jaehwoon [16]	Draft-Bernardos[17]	SDN-DMM
Routing	Sub-optimized	Sub-optimized	Optimized
Performance	Lower signaling cost with processing and delivery packet overhead	Relative signaling cost with processing and delivery packet overhead via A-MAAR	Relative signaling cost without processing and delivery packet overhead
End host requirements	None	Support for multiple IP addressing	None
Network-based Architecture	Fully DMM	Partially DMM	Partially DMM

5. RESULTS AND DISCUSSION

This section provides the results of the analytical evaluation of the proposals that evidence the benefits of an SDN-DMM approach regarding costs of signaling and handover latency.

5.1 - Signaling

For a general comparative result in the signaling cost evaluation, as the network topology affects the cost according to the DMM approach used, the cost for the transmission of a control packet among the network entities, cost C_{X-Y}, is the same for all solutions analyzed, i.e., $C_{MAG-MAG} = C_{CMD-MAAR} = C_{CTR-OFS} = C$ [15].

Moreover, for the cost calculation of the control packet processing by entity X, PC_X does not affect the signaling cost significantly, hence, the evaluation of a DMM approach perspective. The same processing cost was considered for all entities in the solutions, where $PC_X = PC$.

The comparison of the signaling cost between SDN-DMM and draft-Jaehwoon is shown in equation (14). It was obtained by the ratio between the SDN-DMM signaling cost, equation (8), and the draft-Jaehwoon signaling cost, which is the sum of equations (2), (3) and (4).

$$\frac{C_{SIG}^{SDN-DMM}}{C_{SIG}^{\text{Draft-Jaehwoon}}} = \frac{2 + N_{PR}}{6 + 2R_{BCE}} \qquad (14)$$

As the comparison between SDN-DMM and draft-Jaehwoon uses the HNO address concept and the signaling process always involves at most two MAG pairs, N_{PR} can be considered the number of open flow switches involved in the signaling process for a bidirectional IP flow update and support to a mobile node handover.

Figure 04 shows the comparison of the signaling cost between SDN-DMM and draft-Jaehwoon with the dwell time variation in the subnet for different numbers of switches involved in the process.

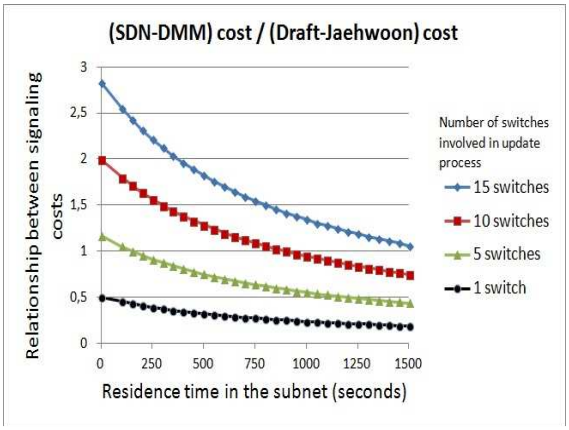


Figure 04 – Signaling cost: SDN-DMM vs Draft-Jaehwoon

As shown in Fig. 04, as the bidirectional IP flow must be established between the switches affected by the handover process of the mobile node, the larger the number of OpenFlow switches affected by the update process of a new bidirectional IP flow, the higher the signaling cost of the SDN-DMM solution in comparison with draft-Jaehwoon, because the signaling change in draft-Jaehwoon is limited to at most two MAG pairs.

As the bidirectional IP flow alterations occur on the network edge, the number of switches affected by the update process probably does not exceed 10 switches, which may quickly decrease the cost of the signaling cost of SDN-DMM to below that of draft-Jaehwoon in function of the time of permanence of the mobile node in the subnet.

The signaling cost of SDN-DMM becomes lower as the time of permanence of the mobile node in the subnet increases, because the cost of periodical updates has a higher weight in the determination of the total signaling cost. As SDN-DMM does not involve such a periodical update cost, differently from draft-Jaehwoon, SDN-DMM requires a lower signaling cost.

The comparison between SDN-DMM and draft-Bernardos provides equation (15), which represents the ratio between the SDN-DMM signaling cost, according to equation (8), and the draft-Bernardos signaling cost, which is the sum of equations (5), (6) and (7).

$$\frac{C_{SIG}^{SDN-DMM}}{C_{SIG}^{\text{Draft-Bernardos}}} = \frac{2 + N_{PR}}{6 + 2N_{PR} + 2R_{BCE}} \qquad (15)$$

Figure 05 shows the comparison between the signaling cost of SDN-DMM and draft-Bernardos, where the relation between signaling costs is shown according to the variation in the time of permanence of the mobile node in the subnet, which can be understood as the variation of its handover rate for four different numbers of active prefixes with current sessions.

The signaling cost of SDN-DMM is, at least, 50% lower in comparison with that of draft-Bernados because the OpenFlow switches that receive the messages from the controller in an update process for the establishment of a new bidirectional IP flow do not need to answer them. Therefore, the C_{update} cost of the SDN-DMM solution, which increases in function of the increase in the number of LNP prefixes, becomes lower.

Moreover, as SDN-DMM does not use messages for periodical updates and

deregistration, when the mobile node remains longer in the subnet, which is a situation of higher weight of periodical updates in the calculation of the signaling cost, the relative cost of SDN decreases, as it does not have such a component.

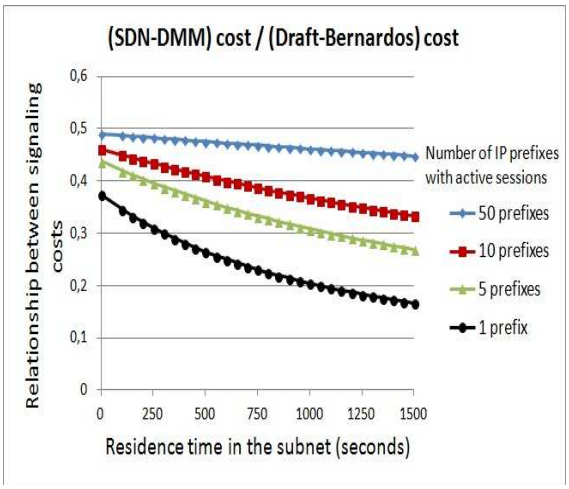


Figure 05 – Signaling cost: SDN-DMM vs Draft-Bernardos

5.2 - Handover latency

The time spent on the t_{L2} and t_{auth}handover operation stages and the packet exchange between the mobile node and its access point to the network, RTT_{MN-AP}, will be considered the same in all proposals analyzed, for the evaluation of the impact of stage $t_{binding}$ on the handover latency cost. Therefore, the variation of the components that comprise $t_{binding}$ becomes the most impacting factor on the total handover latency cost.

Based on [15], we consider the relation in (16) an equal packet processing performed by the network units, shown in (17), and a fixed distance between the central control element and each access point to the network in the access network in DMM partially distributed, expression (18).

$$t_{l2} + t_{auth} + RTT_{MN-MAG} + 2T_{proc}^{MAG} = t_{l2} + t_{auth} + RTT_{MN-MAAR} + T_{proc}^{MAAR} + T_{proc}^{CMD} = t_{l2} + t_{auth} + RTT_{MN-OFS} + T_{proc}^{OFS} + T_{proc}^{CTR} = T_{commun} \quad (16)$$

$$T_{proc}^{MN} = T_{proc}^{MAAR} = T_{proc}^{CMD} = T_{proc}^{OFS} = T_{proc}^{CTR} = T_{proc}^{Unit} \quad (17)$$

$$RTT_{CMD-MAAR} = RTT_{CTR-OFS} = RTT_{control} \quad (18)$$

A 50ms value, obtained by the experiments conducted in [15], is attributed to T_{commun}, whereas 1 ms is assigned to T_{proc}^{Unit}, as the time of packet processing by the unit is equal to the latency inserted by the packet processing in a hop between routers, which is typically lower than 1 ms [13] in a backbone network. 5 ms, 10ms and 20 ms, which are common values for the communication between the edge network equipment and the concentrator located in a Point of Presence along a backhall network [14] are use for $RTT_{control}$.

The equation in (19), obtained by the ratio between equations (13) and (11), represents the relation between handover latency costs of SDN-DMM and draft-Jaehwoon.

$$\frac{C_{handover}^{SDN-DMM}}{C_{handover}^{\text{Draft-Jaehwoon}}} = \frac{T_{commun} + 4T_{proc}^{Unit} + 2RTT_{CTR-OFS}}{2T_{commun} + 4T_{proc}^{Unit} + 2RTT_{MAG-MAG}} \quad (19)$$

Figure 06 shows the impact on the relation between handover latency costs between SDN-DMM and draft-Jaehwoon, according to the latency variation as the mobile node conducts new handover processes in the network, which can be understood as the increase in the distance between MAGs that establish the tunnel for the addressing of the packets of the mobile node, so that it can be provided with global connectivity.

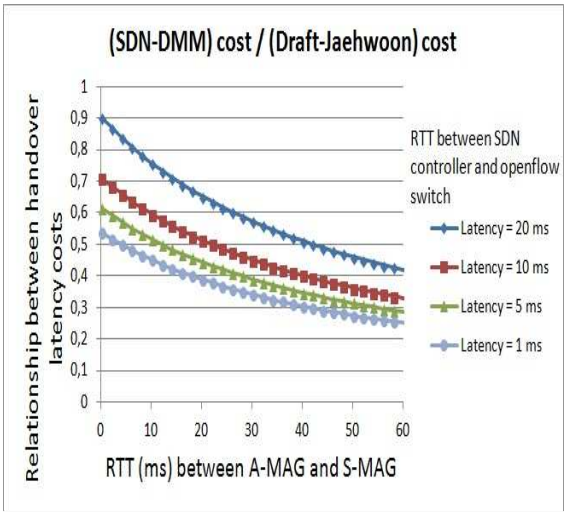


Figure 06 – Handover latency cost: SDN-DMM vs draft-Jaehwoon

Due to the approach employed for the supply of mobility to IP sessions in draft-Jaehwoon, in which process T_{commun} is required twice, in comparison with SDN-DMM, the handover latency cost is always lower when the distance between the concentrator and the OpenFlow switch is equal to the distance between the MAGs that perform tunneling. If a typical 5ms RTT is used for both, the cost reduction is approximately 44%.

Another important aspect is the implementation of the controller in relation to the OpenFlow switches in the access networks is generally performed in a way its location results in the shortest collective latency between them, i.e., the controller assumes a central position, so that as the mobile node performs a handover between the access networks, the latency between the OpenFlow switch and the controller is not significantly changed. Another implementation that reduces latency is the implementation of the controller in a hierarchical cluster mode, in which the controller responsible for certain access networks is implemented near them.

As MAGs are located on the edge of the access network, the movement of the mobile node in the access networks increases the latency between S-MAG and A-MAG, which does not occur in SDN-DMM and contributes to its lower handover latency cost during the movement of the mobile node, as shown by the curves in Fig. 06.

The relation betwen handover latency costs between SDN-DMM and draft-Bernardos is shown in equation (20), obtained by the ratio between equations (13) and (12).

$$\frac{C_{handover}^{SDN-DMM}}{C_{handover}^{\text{Draft-Bernardos}}} = \frac{T_{commun} + 4T_{proc}^{Unit} + 2RTT_{CTR-OFS}}{T_{commun} + 6T_{proc}^{Unit} + 2RTT_{CMD-MAAR} + 2RTT_{MAAR-MAAR}} \quad (20)$$

Figure 7 shows the results provided by equation (20).

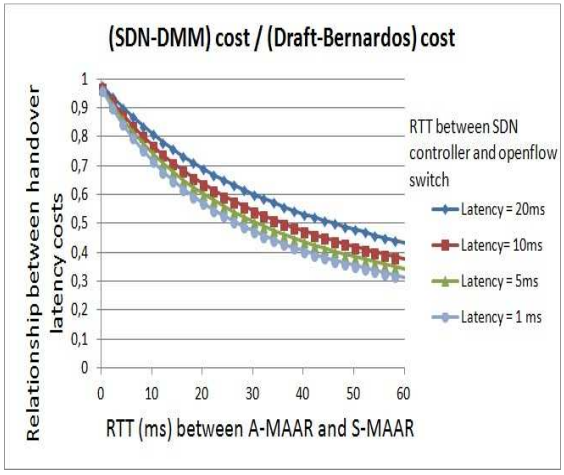


Figure 7 – Handover latency cost: SDN-DMM vs draft-Bernardos

The handover latency cost is higher in draft-Bernardos than in SDN-DMM due to the larger number of interactions in the $t_{binding}$ stage. First, the registration and update of the prefixes that are still active in the mobile node during the handover performed between S-MAAR and CMD are required, which results in component $RTT_{CMD-MAAR}$. The tunnel between A-MAAR and S-MAAR must then be established through the addition of component $RTT_{MAAR-MAAR}$, so that the traffic of the prefixes that are still active can be forwarded, which also results in a larger amount of T_{proc}^{Unit} processing.

As the distance between the control element and the point of access to the network are the same in the two proposals, i.e., $RTT_{CTR-OFS} = RTT_{CMD-MAAR}$, the predominant factor for the handover latency cost is $RTT_{MAAR-MAAR}$.

As in draft-Jaehwoon, as MAAR are located on the edge of the access networks, the latency between S-MAAR and A-MAAR may increase, as the mobile node performs handover, which increases the handover latency cost in comparison with SDN-DMM, as shown by the behavior of the curves in Fig. 7 with the latency increase in the x axis.

6. CONCLUSIONS AND FUTURE WORK

This paper discussed aspects and proposals related to mobility management functions in an IP-based heterogeneous communications network, and proposed an architecture based on Software Defined Networking for Distributed Mobility Management (SDN-DMM), considering the need for session continuity, scalability and performance. The results show SDN-DMM satisfactorily deals with the main challenges of mobility management for an exponential increase in the number of mobile devices.

The proposal provides an efficient and scalable use of the available resources of the communication network infrastructure for the mobile node and the heterogeneous access networks, from both operational (OPEX) and evolutive and investment-based viewpoints through a network-based partially DMM approach associated with SDN and incorporating the intrinsic benefits of the paradigm to the solution.

Scalability for the data plane is obtained as a central node is not dependent or overloaded for the mobility of the mobile node. The network equipment on the edge of the access network only forward the traffic to the mobile nodes connected to it and do not redirect the traffic to the mobile nodes connected to other network equipments.

Regarding the control plane, the controller is the central point of the network intelligence responsible for the mobility of the domain. It can be implemented hierarchically in cluster mode and a certain controller becomes responsible for part of the infrastructure, whereas the others act as backup and responsible for other parts of the network, which enables issues on scalability and the unique failure point to be addressed for the control plane.

As the mobile nodes are not involved and do not conduct any other additional process regarding mobility and the network infrastructure does not need to support any other new protocol, the benefit of the proposal regards the maintenance of the same level of complexity for both the network edge and final devices and the network core, according to the SDN-based architecture. The processing and energetic consumption in the mobile node are not affected and the network equipments do not need to create or maintain specific tables and processes, as encapsulation and binding relations to provide mobility.

We can conclude that SDN-DMM adequately addresses issues of scalability, performance and complexity involved in the distributed mobility management, allowing to ensure IP session continuity and using infrastructure resources related to mobility management tasks in an efficient way.

Additional studies have been conducted about SDN-DMM ([21]), considering metrics as routing cost and packet delivery cost (treated as the cost for packets to be delivered between the mobile node (MN) and the correspondent node (CN) within the mobility domain [15]), aiming to discuss latency reduction of end-to-end communication and treating the formation of bottlenecks for the traffic forwarding, with the the traffic being routed directly between the hosts without use of tunneling techniques.

For the routing optimization, we also consider that packets could be addressed directly to the mobile node based on the bidirectional IP flow and without the use of encapsulation techniques (which would add an overhead for both processing and transport of the payload through the communication network), allowing to improve the user quality of experience and efficiently using the available

communication resources in a domain mobility scenario.

Other studies involve the validation of the model in a scenario of real experimentation, considering the need for data offloading of IP flows and extensive statistical analysis. The support to a possible hybrid model (client-based + network-based) with SDN extension and the mobility management involving different autonomous systems has also been considered.

REFERENCES

[1] KARIMZADEH M; VALTULINA, L; KARAGIANNIS, G. *Applying SDN/OpenFlow in virtualized LTE to support Distributed Mobility Management (DMM)*. In: Proceedings of the 4th International Conference on Cloud Computing and Services Science, CLOSER 2014, 3-5 Barcelona, Spain. 86. SciTePress, Abril 2014.

[2] KARIMZADEH, M; ZHAO, Z; HENDRIKS, L; SCHMIDT, R; FLEUR, S; BERG, H; PRAS, A; BRAUN, T; CORICI, M. *Mobility and Bandwidth Prediction as a Service in Virtualized LTE Systems*. 4th IEEE International Conference on Cloud Networking (CLOUDNET), 2015.

[3] VALTULINA, L; KARIMZADEH, M; KARAGIANNIS, G; HEIJENK, A; PRAS, A. *Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems*. IEEE GLOBECOM 2014.

[4] GIUST, F; BERNARDOS, C; OLIVIA, A. *HDMM: deploying client and network-based distributed mobility management*. Journal Telecommunications Systems, Volume 59 Issue 2, Pages 247-270, Junho 2015.

[5] SEITE, P; BERTIN, P; *Distributed Mobility Anchoring*. Draft IETF at https://tools.ietf.org/html/draft-seite-dmm-dma-00, Fevereiro 2012.

[6] ONLAB. *Introducing ONOS - a SDN network operating system for Service Providers.* Whitepaper http://onosproject.org/wp-content/uploads/2014/11/Whitepaper-ONOS-final.pdf [accessed Nov-2015].

[7] Perkins, C., Johnson, D., & Arkko, J. (2011). Mobility Support in IPv6. RFC 6275.

[8] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K.,&Patil, B. (2008). Proxy Mobile IPv6. RFC 5213.

[9] Chan,H.A.,Yokota,H., Xie, J., Seite, P.,&Liu,D. (2011). Distributed and dynamicmobility management in mobile internet:Current approaches and issues. Journal of Communications, 6(1), 4–15.

[10] Chan, H. (2014). Requirements for Distributed Mobility Management. RFC 7333.

[11] Yokota, H., Seite, P., Demaria, E., & Cao, Z. (2010). Use case scenarios for Distributed Mobility Management. Internet-Draft (work in progress), draft-yokota-dmm-scenario-00.txt.

[12] KARIMAZDEH, *SDN Mobility Management*, 2014

[13] Estatísticas de operação da Rede Ipê – Backbone da Rede Nacional de Ensino e Pesquisa (RNP), 2016.

[14] Estatísticas de operação do Ponto de Presença da RNP no Distrito Federal (PoP-DF), 2016.

[15] Giust F., Bernardos C., Oliva A. (2014). Analytic Evaluation and Experimental Validation of a Network-Based IPv6 Distributed Mobility Management Solution. In IEEE Transactions on Mobile Computing, vol. 13, no. 11, November 2014.

[16] J. Lee and Y. Kim, "PMIPv6-based Distributed Mobility Management," IETF draft, draft-jaehwoon-dmm-pmipv6-04, June 18 2015.

[17] BERNARDOS, C.; OLIVIA, A.; GIUST, F. A PMIPv6-based solution for Distributed Mobility Management. In IETF draft, draft-bernardos-dmm-pmip-06, Março 2016.

[18] BERNARDOS C., OLIVA A., SERRANO P., et al. (2014). *An Architecture for Software Defined Wireless Networking*. In IEEE Wireless Communications, vol. 21, no.3, pag 52-61.

[19] CISCO SYSTEM. *Routers Products and Solotions.* http://www.cisco.com/c/en/us/products/routers/buyers-guide.html [accessed May-2016].

[20] JUNIPER NETWORKS. *MX Series Plataforms.* http://www.juniper.net/us/en/products-services/routing/mx-series/compare/#a=VMX,P5,P10,P40,P80,P104,P240,P480,P960,P2010,P2020 [accessed May-2016].
[21] CORDOVA, R; GONDIM, P. 2016. *Packet Delivery and Routing Optimization for DMM with SDN-DMM approach.*

APÊNDICE D – ARTIGOS ELABORADOS

D.1 – PACKET DELIVERY AND ROUTING OPTIMIZATION FOR DMM WITH SDN-DMM APPROACH

RESEARCH ARTICLE

PACKET DELIVERY AND ROUTING OPTIMIZATION FOR DISTRIBUTED MOBILITY MANAGEMENT WITH SDN-DMM

Rodrygo Torres Córdova[†] ***and*** **Paulo Roberto de Lira Gondim**[†]

SUMMARY Mobility management applied to the traditional architecture of the Internet has become a great challenge due to the exponential growth in the number of devices that can connect to network and the traditional routing based on the destination address field of the IP header. This article presents a analytical evaluation conducted with metrics of packet delivery cost and routing cost between a proposal of network architecture based on Software Defined Networking, called SDN-DMM, and two IETF drafts for DMM, which shows that the SDN approach not only provides the intrinsic benefits of SDN in compare with traditional architecture, but also deals with the distributed mode of mobility management in heterogeneous access networks in a simplified and efficient way.

key words: Software-Defined Networking (SDN); OpenFlow; Distributed Mobility Management (DMM); IP Mobility Management; Mobile IP networks.

1. Introduction

Mobile communication networks have become the main users method of access to the Internet, which has significantly increased the number of mobile devices connected to the global network [12].

As the services offered by operators of mobile communication networks tend towards involving solutions totally based on the IP protocol for both voice and data services [1] and the communication sessions must be continued, the IP mobility management has become fundamental for communication networks, so that such an increase can be supported [18].

The IETF IP mobility management standards, as MIPv6 [7] and PMIPv6 [8] depend on central units that manage both control and data traffic, elaborated according to the traditional routing of IP packets. They also pose problems, as sub-optimized routing, low scalability, overload of processing in network devices and low granularity in the mobility management service.

An alternative for dealing with the intrinsic problems of centralization and costs in the mobility management is the new concept, called distributed mobility management (DMM) [9]. Its main characteristic is the clear separation between the data plane and the control plane. The data plane is distributed along the equipments on the access network edge for approximating the mobile agent to the end user, implementing a flatter mobility approach in the network [10]. Therefore, the traffic forwarded to the mobile node (MN) does not cross a specific central point in the network, i.e., the mobility anchor, as it occurs in centralized solutions. In DMM, the traffic is forwarded by the mobile agents nearest the user.

The DMM solutions can be categorized into two types, depending on the distribution level of the control plane [4, 10, 11]: Partially distributed - the data plane is completely distributed among the network elements and the control plane is centralized in control points in the network; and Fully distributed - both data plane and control plane are completely distributed among the network elements and there exists no central entity of control.

This manuscript presents a analytical evaluation conducted with metrics of packet delivery cost and routing cost between the SDN-DMM proposal, based on SDN with OpenFlow protocol, and two IETF drafts for DMM, for dealing with the above-mentioned challenges of IP mobility management.

The remainder of the manuscript is organized as follows: Section II describes the SDN-DMM proposal [21] and the related work on network-based DMM; Section III reports the analytical evaluation of the proposals; Section IV provides the results and discussions. Finally, Section V presents our conclusions, reports on the ongoing studies and suggests future work.

2. SDN-DMM and Related Work

This section introduces the SDN-DMM proposal

[21], which uses the SDN approach for distributed management mobility with IP Flows, and two IETF drafts for DMM - one with a fully distributed approach, (Draft-Jaehwoon [16]), and another with a partially distributed approach (Drafft-Bernardos [17]).

2.1 SDN-DMM [21]

SDN-DMM is a network-based partially DMM proposal based on the SDN architecture where the IP mobility is supported by the network infrastructure through the automatic adjustment of the IP flows in the data plane by the SDN controller with OpenFlow, according to the handover executed by the end host.

The proposal considers an abstraction of the northbound layer, called *Intent*, which informs the SDN controller on a communication to be implemented in the network. The controller, which has a full view of the topology, creates and adjusts OpenFlow rules according to the mobility scenario, so that the data plane of the network can deliver the data traffic directly to the end host in any network point of attachment. Figure 01 illustrates the process, where:

1. Mobile node MN1 performs handover from NB to eNB, keeping its original IP address;
2. Switch SB identifies the presence of the mobile node with an IP network address different from its scope and informs the event to controller C1;
3. Controller C1 detects a topology change in the mobility of mobile node MN1, recalculates a new path and sends new OpenFlow rules for the readjustment of the communication links used;
4. The new bidirectional IP flows are established in the network, which enables the packets addressed to mobile node MN1 to be correctly forwarded according to the IP flow, rather than to the traditional IP routing. Therefore, the general routing and the other ongoing communications in the network are not affected.

2.2 Draft-Jaehwoon: A Fully Distributed Solution for DMM [16]

Draft-Jaehwoon is a network-based fully DMM proposal based on the PMIPv6 protocol [8], where MAGs are responsible for all the control and data plane functions that support the IP mobility.

Its use a supernet concept for the entire mobility domain where the mobile node considers that is always connected to the same network when performed a handover. The MAGs identify the handover performed by the mobile node and exchange control messages in the control plane to create a tunnel between them in the data plane to properly forwarding the traffic to the mobile node in the data plane.

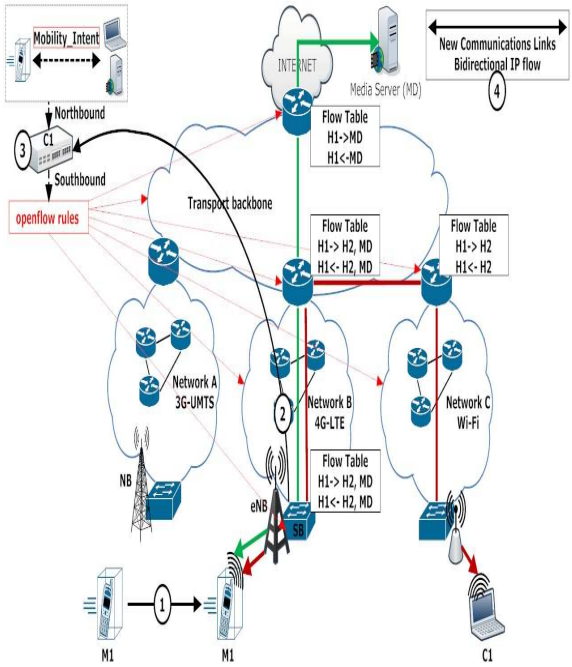


Fig. 1 Mobility support with IP Flows [21]

So when the mobile node perform a handover, a new tunnel is established between the current MAG that the mobile node is connect, called Seving MAG and the first MAG that the mobile node was connected when it entered in the mobility domain, called Native MAG.

2.3 Draft-Bernardos: A Partially Distributed Solution for DMM [17]

Draft-Bernardos is a network-based partially DMM proposal based on the PMIPv6 protocol [8], in which a central unit, called central mobility database (CMD) performs the main functions of control plane, whereas the forwarding of traffic, data plane, is the responsibility of units called mobility anchor and access router (MAAR).

CMD assumes the LMA figure, however, it does not forward traffic to the mobile node, i.e., it does not participate of the data plane operations. It is the central entity of the control plane and manages sessions and registration of IP, location and binding cache. Each MAAR manages a set of nonnsuperposed specific IP prefixes assigned to the mobile node when it is connected to MAAR. At each new connection of the mobile node to a new MAAR, a new IP address is allocated in the mobile node, under the responsibility of MAAR. The home network prefix (HNP) address concept of PMIPv6, in which only one IP address is allocated to the mobile node for all mobility domain, is changed to the address concept called local network prefix (LNP).

So when the mobile node perform a handover, a

new tunnel is established between the current MAAR that the mobile node is connect, called Seving MAAR, and the all other MAAR that the mobile node was connected and have active IP session using the prefix managed by these MAARs, called Natives MAAR.

3. Analytical Evaluation

This section addresses the analytical evaluation conducted with metrics of packet delivery cost [15] and routing cost proposed for draft-Jaehwoon [16], draft-Bernardos [17], and SDN-DMM proposals. The evaluation of other metrics, such as signaling and handover latency costs, is provided in [1].

3.1 Packet delivery cost

In a mobility management perspective, the packet-delivery cost can be understood as the cost for packets to be delivered between the mobile node (MN) and the corresponding node (CN) within the mobility domain. In general, the total packet-delivery cost comprises the sum of the three costs according to the three main distances in the packet-delivery route from CN to MN [15]:

iv. C_{CN-NMA}= Cost of the delivery packet between CN and the native mobility anchor (NMA) responsible for the initial forwarding of the packets addressed to the MN within the mobility domain;

v. $C_{NMA-SMA}$= Cost of the delivery packet between NMA and serving MA (SMA) that is currently serving MN; and

vi. C_{SMA-MN}= Cost of the delivery packet between SMA and MN through a wireless medium.

The delivery cost of a non-encapsulated packet has a unitary value; the delivery cost of the same packet through tunneling suffers a $\tau > 1$ penalty and the cost for the delivery of the packet through a wireless link suffers a ωpenalty. The total packet-delivery cost C_{PD} provided in [15], for the sending and reception transmission rate of packets λ, considering the use of encapsulation between NMA and SMA and the packet-delivery to MN through a wireless link, is given by

$$C_{PD} = \lambda(C_{CN-NMA} + \tau C_{NMA-SMA} + \omega C_{SMA-MN}) \quad (1)$$

The impact caused by distance D between the elements that encapsulate the packets addressed to the mobile node and that affect the $C_{NMA-SMA}$ cost generates the new expression for the packet-delivery cost C_{EP}:

$$C_{EP} = \lambda(C_{CN-NMA} + \tau DC_{NMA-SMA} + \omega C_{SMA-MN}) \quad (2)$$

Below are the packet-delivery costs for each proposal evaluated:

- Draft-Jaehwoon [16]

Only one tunnel is established between A-MAG and S-MAG. Therefore, according to expression (2), where NMA and SMA are represented, respectively, by AMAG and SMA, C_{EP} is given by

$$C_{EP}^{\text{Draft-Jaehwoon}} = \lambda(C_{CN-AMAG} + \tau DC_{AMAG-SMAG} + \omega C_{SMAG-MN}) \quad (3)$$

- Draft-Bernardos [17]

The mobile node is always provided with a new address when it performs handover between the MAARs, therefore, a new packet-delivery cost is added to each active prefix as new tunnels are established in pairs between A-MAARs and S-MAARs. According to expression (2), the packet-delivery cost is given by

$$C_{EP}^{\text{Draft-Bernardos}} = \lambda(C_{CN-AMAAR} + N_{PR}\tau DC_{AMAAR-SMAAR} + \omega C_{SMAAR-MN}) \quad (4)$$

where N_{PR} is the number of active prefixes and AMAAR is NMA and SMAAR is SMA.

- SDN-DMM

The encapsulation techniques are not used between network entities within the mobility domain in the SDN-DMM proposal, as the traffic is addressed by bidirectional IP flows in an optimized way, so that no packet-delivery cost penalty occurs. No $C_{NMA-SMA}$ cost is required, because the traffic originated by the CN and addressed to the MN is delivered directly to the network entity that is serving the mobile node when it enters the mobility domain. Packet-delivery cost C_{EP}, where the mobility anchors are the OpenFlow switches (OFS), is given by

$$C_{EP}^{SDN-DMM} = \lambda(C_{CN-OFS} + \omega C_{OFS-MN}) \quad (5)$$

3.2 Routing cost

The packet routing highly impacts on mobility scenarios, from both the final user´s perspective and the communication network infrastructure.

An optimized routing provides a more efficient use of the communication links available and reduces the latency of the final communication, which avoids the formation of bottlenecks for the traffic forwarding and improves the user´s quality of experience (QoE).

The routing cost metric has been defined for the analysis of the impact exerted by the routing on the

mobility scenarios according to the DMM approach used in each proposal. The routing cost, $C_{routing}$, is defined in equation (6), which considers the downstream direction in the communication, i.e., from the corresponding node to the mobile node, for simplification.

$$C_{routing} = \frac{C_{m-routing}}{C_{l-routing}} = \frac{H^m_{MN-CN} + L^m_I + 2L^m_E}{H^l_{MN-CN} + L^l_I + 2L^l_E} \quad (6)$$

where $C_{m-routing}$ is the routing cost for the mobile node and $C_{l-routing}$ is the routing cost for a local host connected to the network where the mobile node is located. Moreover,

iv. H_{MN-CN} = number of hops (hop count) performed by the packet within the mobility domain until it reaches the mobile node/ local host;

v. L_I = number of times internal links are used for the forwarding of packets to the mobile node/ local host. Internal links connect routers horizontally, i.e., routers located in the same level/layer in relation to the core network;

vi. L_E = number of times external links are used for the forwarding of packets to the mobile node/ local host. External links connect routers vertically, i.e., routers located in different levels/layers in relation to the core network, and are called upstream/downstream communication links.

The routing cost expresses how much a value is higher in relation to an optimized routing according to the topological location of the mobile node for a certain DMM approach. When the routing cost, $C_{routing}$, obtained is 1, solution DMM does not increase it and provides an optimized routing, i.e., the cost of the packets routed for both mobile node and local host is the same.

Weight 2 attributed to component L_E in equation (6) accounts for the use of more expensive infrastructure resources and the higher latency inserted. The upstream links are generally more expensive and cover a longer distance due to the traffic aggregation objective between a lower and a higher network layer. Moreover, in a hierarchical network, the upstream links connect network equipment of lower cost and performance, as access routers, to network equipment of higher cost and performance, as metropolitan aggregation routers [13, 19, 20]. Therefore, weight 2 accounts for the cost and latency involved in the use of the links.

Component L_E exerts a higher impact on the determination of the routing cost than component L_I, which represents the links that connect the network equipment of a same network layer for the local traffic resolution using links and network equipment of lower capacity and cost. L_E also exerts a higher impact than component H_{MN-CN}, which represents the general number of hops a packet must travel. The latency inserted by the hops can be considered the same, regardless of the network layer in which it is located.

The same way routing depends on the network topology, the routing cost of a certain DMM approach depends on the network topology employed. Therefore, a simple tree network topology (Figure 2) will be considered for the analysis of the routing cost of the proposals presented.

The movement of a mobile node through the network infrastructure also impacts on the routing cost. For a simpler determination of an expression that represents the average routing cost of the proposals, the mobile node will move linearly between the adjacent access points that belong to the same network level, where the routing cost is equal to the average of the routing costs of each handover.

The routing cost for the proposals is then analyzed according to the tree topology and movement of the mobile node of figure 2, where *n* represents the number of vertical levels of the network, i.e., the depth and *m* denotes the number of nodes of a lower level that are connected to a higher-level node, starting from the highest level, i.e., the root level ($n = 0$) with a unique node ($m = 1$). The home network of the mobile node is fixed to an extremity of the network in a certain level and the traffic of the correspondent node enters the mobility domain through a central point.

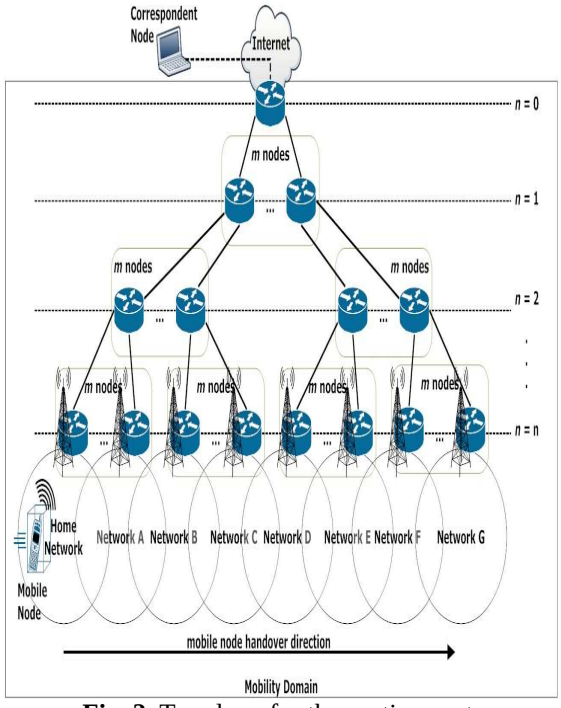


Fig. 2 Topology for the routing cost

- Draft-Jaehwoon [16] and draft-Bernardos [17]

As both drafts employ encapsulation techniques to forward the mobile node traffic (the traffic must be sent to the mobility anchor located in the home

network responsible for the treatment of packets addressed and originated by the mobile node), such proposals have the same routing cost.

The routing is performed with the packets originated or addressed by the mobile node, first forwarded to the mobility anchor responsible according to the routing employed in the network and then forwarded through a tunnel to the mobile node and through the infrastructure to the correspondent node, again according to the routing technique employed in the network.

The use of equation (6) and the topology in figure 2 for the calculation of the average routing cost with a fixed value to $n > 0$ and a variable value to ≥ 2 causes a geometrical progression whose common ratio is m and first term is $m - 1$ of the handover number that generates the same cost. A common area, called Routing Area, is then created and the handovers results in the same routing cost.

On the other hand, when a value is fixed to $m \geq 2$ and $n > 0$ is variable, an arithmetic progression is observed and its common ratio is 6 for $C_{m-routing}$ in the calculus of the routing cost between routing areas for an n previously determined. Another arithmetic progression is developed for $C_{m-routing}$and $C_{l-routing}$ in the calculus of the routing cost between the routing area for a given n and $n + 1$, whose common ratio is 3.

Figure 3 shows the behavior of the progressions for $m = 2$ and $n = 1, 2\ and\ 3$. The dashed line represents the path of the packets sent from the correspondent node to the mobile node with $n = 3$, when the mobile node makes a first handover at its home network to the first point located at the routing area 1.

The traffic uses the communications network infrastructure inefficiently, using two times the upstream link (identified by number 1 in figure 03) between the home network router and the one located in the top layer, such as this router, resulting in a $L_E^m = 5$ and $H_{MN-CN}^m = 6$.

Whereas for a optimized routing the above upstream link is not even used, once only the link identified by number 2 in figure 03 is used. And the upstream router is used just one time. Then the results of the otimized routing are $L_E^l = 3$ and $H_{MN-CN}^l = 4$. The different routing areas and their costs according to *n* are also provided.

Therefore, the average routing cost of drafts Jaehwoon and Bernardos for a simple tree topology with linear mobile node handover depends on the number of network levels of the topology, *n*, and number of nodes connected by each higher node, *m*, defined by

$$C_{routing}^{Drafts} = \frac{\sum_{i=1}^{n}(m-1)m^{i-1} \times \frac{10+(n-1)3+(i-1)6}{4+(n-1)3}}{m^n - 1} \quad (7)$$

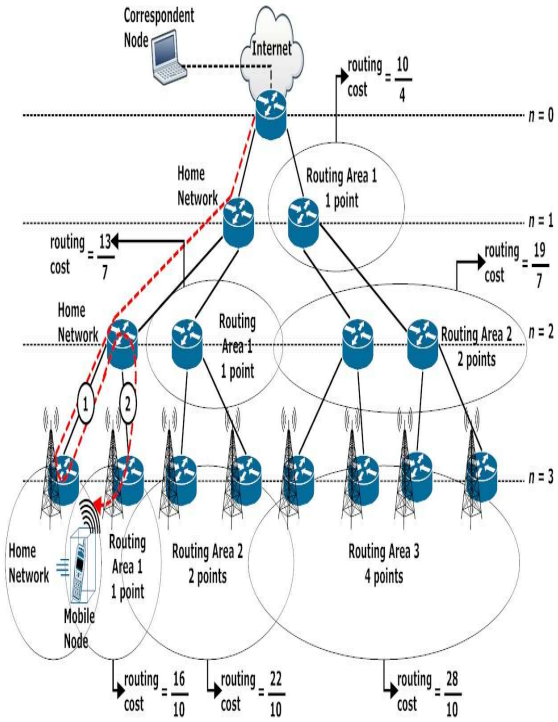


Fig. 3 Routing cost for m=2 and n =1,2 and 3

- SDN-DMM

The routing employed in SDN-DMM for the treatment of the mobility of the mobile node is based on the forwarding of packets by means of bidirectional IP flows according to the new location of the mobile node that performs a handover.

As the IP used by the mobile node and the scope of the addressing of the network to which the mobile node connects are not considered in the packet routing and the routing is performed in an optimized way.

If the decision on the implementation of the bidirectional IP flow is considered to follow the route defined by the IGP used in the network, the routing employed will always be optimized and require no additional cost for any position of the mobile node in relation to its home network.

Therefore, routing cost $C_{m-routing}$ for the mobile node is the same of routing cost $C_{l-routing}$ for a local host of the network to which the mobile node connects for any topology or scenario used, taking into account that the route defined by the implemented IGP is the optimized route, so the routing cost of SDN-DMM is always 1, according to equation (8),

$$C_{routing}^{SDN-DMM} = 1 \quad (8)$$

4. Results

This section addresses the benefits of an SDN-DMM approach regarding costs of packet delivery and routing and provides the results of its analytical evaluation, which evidences its efficiency.

4.1 Packet delivery

For a more similar evaluation of the packet-delivery cost of the proposals analyzed, cost C_{SMA-MN} related to the final packet-delivery to the mobile node through the wireless interlacing is disregarded [15] and cost C_{CN-NMA} is considered the same, with a unitary value for all solutions, i.e., the cost for a packet to enter the domain and be delivered to the first entity involved in the mobility scenario is given by $C_{CN-AMAG} = C_{CN-AMAAR} = C_{CN-OFS} = 1$. Therefore, the focus is maintained on both the evaluation of the approach employed in each proposal and the respective cost for the delivery of the packet within the mobility domain. The comparison of the packet-delivery cost in SDN-DMM and draft-Jaehwoon is shown in figure 4, which also displays the ratio between equations (5) and (3), represented by expression (9).

$$\frac{C_{EP}^{SDN-DMM}}{C_{EP}^{\text{Draft-Jaehwoon}}} = \frac{1}{1 + \tau D} \tag{9}$$

Figure 4 shows the relation between the packet-delivery costs with the variation in the tunneling penalty for the variation in the distance between A-MAG and S-MAG that perform tunneling. Distance 1 can be understood as the base distance, in which A-MAG and S-MAG are directly connected, distance 2 is the double of the base distance, and so on for distances 5 and 10.

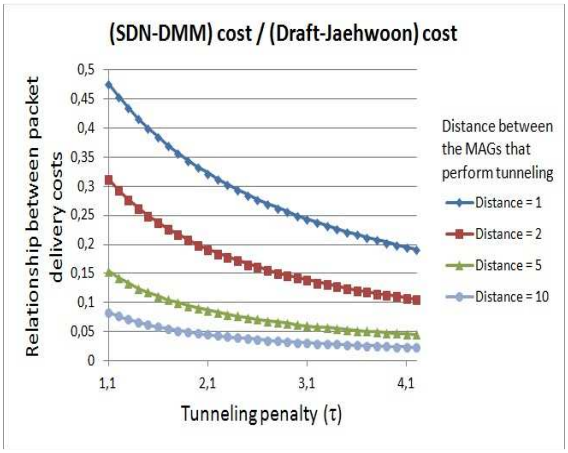


Fig. 4 Packet-delivery cost: SDN-DMM vs draft-Jaehwoon

Figure 5 shows the ratio between equations (5) and (4), represented by expression (10), for the comparison between the packet-delivery cost and the proposal presented in draft-Bernardos.

$$\frac{C_{EP}^{SDN-DMM}}{C_{EP}^{\text{Draft-Bernardos}}} = \frac{1}{1 + N_{PR}\tau D} \tag{10}$$

The figure also shows the relation between the packet-delivery costs in N-DMM and draft-Bernardos for a penalty $\tau = 2$ with the variation in the number of IP prefixes with active sessions.

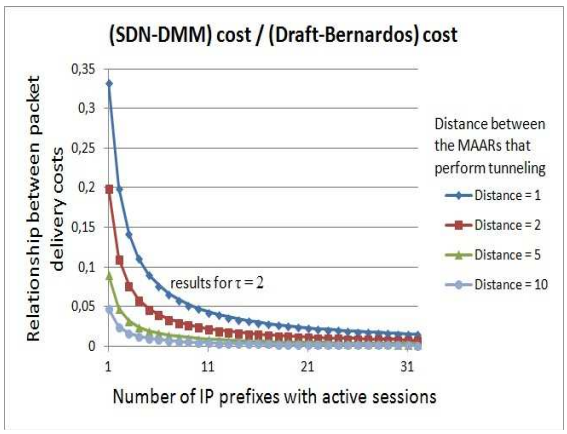


Fig. 5 Packet-delivery cost: SDN-DMM vs draft-Bernardos

In comparison with the proposals presented in draft-Jaehwoon and draft-Bernardos, SDN-DMM shows a lower packet-delivery cost because the traffic addressed to the MN is directly delivered to the entity of the network to which the mobile node is connected without the use of encapsulation techniques.

As in draft-Jaehwoon, the traffic is delivered through a tunnel between MAGs; as the tunnel penalty and the distance between the MAGs increase, the SDN-DMM solution requires a lower packet-delivery cost.

The packet-delivery cost of SDN-DMM is lower in comparison with draft-Bernardos, due to the pLNP concept employed in the draft. Several tunnels must be established, i.e., one for each pLNP prefix used by the mobile node. Therefore, the number of pLNP prefixes, which can be understood by the mobile node performing more handover processes and keeping the sessions active, makes SDN-DMM more attractive.

SDN-DMM always provides an optimized packet-delivery, as it is based on bidirectional IP flows established according to the mobility scenarios, where the data plane is adjusted so as to take the best path. No penalty is applied due to the use of tunneling, therefore, the distance between the elements that perform it does not affect the cost.

4.2 Routing

Equation (11) represents the relation between the routing costs of SDN-DMM and drafts Jaehwoon and Bernardos, obtained by the ratio between Equations (8) and (7).

$$\frac{C_{routing}^{SDN-DMM}}{C_{routing}^{Drafts}} = \frac{m^n - 1}{\sum_{i=1}^{n}(m-1)m^{i-1} \times \frac{10+(n-1)3+(i-1)6}{4+(n-1)3}} \tag{11}$$

Figure 6 illustrates the relation of costs for four network levels ($n = 1, 2, 3\ and\ 4$) according to the

1 to 20 variation in the number of nodes connected by a higher level node (m) and based on real implementation scenarios for a better evaluation of the relation among the routing costs of the proposals.

Due to the optimized routing of SDN-DMM, the routing cost is at least 50% lower than that of the other proposals. In SDN-DMM, the traffic addressed to the mobile node is forwarded directly to it, as if it were a local host of the network to which it is connected. On the other hand, in Jaehwoon and Bernardos drafts, the traffic must first reach the mobility anchor responsible for the forwarding of the mobile node to its new location through tunneling and the redundant use of more network resources.

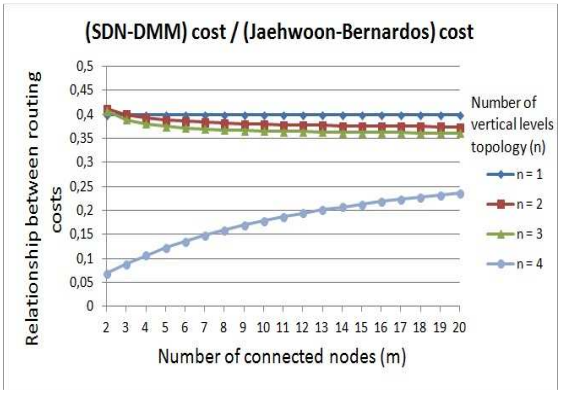


Fig. 6 Routing cost: SDN-DMM vs draft-Jaehwoon/Bernardos

The routing cost of SDN-DMM is 40% of the cost of the drafts, regardless of the number of nodes connected to the higher level node (m) for a simple tree topology with $n = 1$, due to the constant number of leaps and links used for the addressing of packets to the mobile node that is performing handover in a horizontal and linear way, as previously described.

When the number of network levels is increased to $n = 2\ and\ 3$, the initial routing cost for $m = 2$ in relation to that obtained for $n = 1$ is slightly increased (below 3%) (Fig. 6), due to the decrease in the proportional number of resources used in the comparison between SDN-DMM and Jaehwoon and Bernardos drafts at such network levels.

The routing cost for those network levels slightly decreases at $2 \leq m \leq 5$, i.e., approximately up to 6% in comparison with the cost for $n = 1$; from $m > 5$, it stabilizes. As m increases, according to the horizontal and linear movement of the handover adopted, the area in which the routing cost remains the same for the handovers performed increases. The relation with the number of upstream links used promotes the average stability of the routing cost. The routing cost is drastically reduced from network level $n \geq 4$, i.e., over 90%, in comparison with SDN-DMM in Jaehwoon and Bernardos drafts. Such a reduction is due to the significant increase in the proportional amount of network resources used for the routing performed by the drafts in comparison with SDN-DMM.

For approximately $2 \leq m \leq 30$, the routing cost is rapidly increased due to the increase in the area where the routing costs of the proposals are more similar, which increases the average of the relation between the routing costs. The routing costs stabilize from approximately $m \geq 150$ because of the predominance of the last area of the routing cost shown by component $\frac{m^n - 1}{(m-1)m^{i-1}}$ of equation (11).

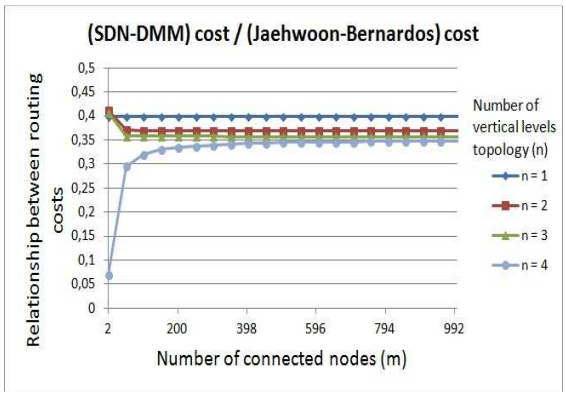


Fig. 7 Routing cost: SDN-DMM vs drafts Jaehwoon and Bernardos ($2 \leq m \geq 1000$)

Figure 7 shows the relation between the routing costs for $2 \leq m \geq 1000$ for illustrating the behavior of the curves, as, according to the topology considered and real implementation scenarios, a value of $m > 20$ is not common.

5. Conclusions and Future Work

The results show SDN-DMM satisfactorily deals with the main challenges of mobility management for an exponential increase in the number of mobile devices.

The proposal provides an efficient and scalable use of the available resources of the communication network infrastructure for the mobile node and the heterogeneous access networks, from both operational (OPEX) and evolutive and investment-based viewpoints through a network-based partially DMM approach associated with SDN and incorporating the intrinsic benefits of the paradigm to the solution.

The routing optimization, in which packets are addressed directly to the mobile node based on the bidirectional IP flow, improves the user´s experience and efficiently uses the available communication links, once the routing cost is not affected, in comparison to the other proposals, by the location of the mobile node, scope of addressing or inappropriate use of links, i.e., scenarios of triangular routing.

Such optimization combined with the non use of encapsulation techniques, which would add an overhead for both processing and transport of the payload through the communication network, provide a higher performance level to the solution

for communication in a mobility scenario, in comparison with the other approaches employed.

Therefore, SDN-DMM adequately addresses issues of scalability, performance, routing and complexity involved in the distributed mobility management. Ongoing studies involve the validation of the model and statistical analyses in a real scenario of experimentation, focus on IP Flow data offloading, support to a possible hybrid model with SDN extension to the mobile node (client-based + network-based) and challenges in the relations of mobility among different autonomous systems.

References

[1] KARIMZADEH M; VALTULINA, L; KARAGIANNIS, G. *Applying SDN/OpenFlow in virtualized LTE to support Distributed Mobility Management (DMM)*. In: Proceedings of the 4th International Conference on Cloud Computing and Services Science, CLOSER 2014, 3-5 Barcelona, Spain. 86. SciTePress, Abril 2014.

[2] KARIMZADEH, M; ZHAO, Z; HENDRIKS, L; SCHMIDT, R; FLEUR, S; BERG, H; PRAS, A; BRAUN, T; CORICI, M. *Mobility and Bandwidth Prediction as a Service in Virtualized LTE Systems*. 4th IEEE International Conference on Cloud Networking (CLOUDNET), 2015.

[3] VALTULINA, L; KARIMZADEH, M; KARAGIANNIS, G; HEIJENK, A; PRAS, A. *Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems*. IEEE GLOBECOM 2014.

[4] GIUST, F; BERNARDOS, C; OLIVIA, A. *HDMM: deploying client and network-based distributed mobility management*. Journal Telecommunications Systems, Volume 59 Issue 2, Pages 247-270, Junho 2015.

[5] SEITE, P; BERTIN, P; *Distributed Mobility Anchoring*. Draft IETF at https://tools.ietf.org/html/draft-seite-dmm-dma-00, Fevereiro 2012.

[6] ONLAB. *Introducing ONOS - a SDN network operating system for Service Providers.* Whitepaper http://onosproject.org/wp-content/uploads/2014/11/Whitepaper-ONOS-final.pdf [accessed Nov-2015].

[7] Perkins, C., Johnson, D., & Arkko, J. (2011). Mobility Support in IPv6. RFC 6275.

[8] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K.,&Patil, B. (2008). Proxy Mobile IPv6. RFC 5213.

[9] Chan,H.A.,Yokota,H., Xie, J., Seite, P.,&Liu,D. (2011). Distributed and dynamicmobility management in mobile internet:Current approaches and issues. Journal of Communications, 6(1), 4–15.

[10] Chan, H. (2014). Requirements for Distributed Mobility Management. RFC 7333.

[11] Yokota, H., Seite, P., Demaria, E., & Cao, Z. (2010). Use case scenarios for Distributed Mobility Management. Internet-Draft (work in progress), draft-yokota-dmm-scenario-00.txt.

[12] KARIMAZDEH, *SDN Mobility Management*, 2014

[13] Estatísticas de operação da Rede Ipê – Backbone da Rede Nacional de Ensino e Pesquisa (RNP), 2016.

[14] Estatísticas de operação do Ponto de Presença da RNP no Distrito Federal (PoP-DF), 2016.

[15] Giust F., Bernardos C., Oliva A. (2014). Analytic Evaluation and Experimental Validation of a Network-Based IPv6 Distributed Mobility Management Solution. In IEEE Transactions on Mobile Computing, vol. 13, no. 11, November 2014.

[16] J. Lee and Y. Kim, "PMIPv6-based Distributed Mobility Management," IETF draft, draft-jaehwoon-dmm-pmipv6-04, June 18 2015.

[17] BERNARDOS, C.; OLIVIA, A.; GIUST, F. A PMIPv6-based solution for Distributed Mobility Management. In IETF draft, draft-bernardos-dmm-pmip-06, Março 2016.

[18] BERNARDOS C., OLIVA A., SERRANO P., et al. (2014). *An Architecture for Software Defined Wireless Networking*. In IEEE Wireless Communications, vol. 21, no.3, pag 52-61.

[19] CISCO SYSTEM. *Routers Products and Solotions.* http://www.cisco.com/c/en/us/products/routers/buyers-guide.html [accessed May-2016].

[20] JUNIPER NETWORKS. *MX Series Plataforms.* http://www.juniper.net/us/en/products-services/routing/mx-series/compare/#a=VMX,P5,P10,P40,P80,P104,P240,P480,P960,P2010,P2020 [accessed May-2016].

[21] CORDOVA, R; GONDIM, P. 2016. *SDN-DMM for Intelligent Mobility Management in Heterogeneous Mobile IP Networks*.

D.2 – AVALIAÇÃO EXPERIMENTAL DE PROPOSTA BASEADA EM SDN PARA GERENCIAMENTO DE MOBILIDADE DISTRIBUÍDO

RESEARCH ARTICLE

AVALIAÇÃO EXPERIMENTAL DE PROPOSTA BASEADA EM SDN PARA GERENCIAMENTO DE MOBILIDADE DISTRIBUÍDO

Rodrygo Torres Córdova[†] *e* Paulo Roberto de Lira Gondim[†]

SUMMARY Mobility management applied to the traditional architecture of the Internet has become a great challenge due to the exponential growth in the number of devices that can connect to network and the traditional routing based on the destination address field of the IP header. This article presents a analytical evaluation conducted with metrics of packet delivery cost and routing cost between a proposal of network architecture based on Software Defined Networking, called SDN-DMM, and two IETF drafts for DMM, which shows that the SDN approach not only provides the intrinsic benefits of SDN in compare with traditional architecture, but also deals with the distributed mode of mobility management in heterogeneous access networks in a simplified and efficient way.

key words: Software-Defined Networking (SDN); OpenFlow; Distributed Mobility Management (DMM); IP Mobility Management; Mobile IP networks.

1. Introdução

As redes de comunicação móveis se tornaram o método principal de acesso a Internet, o que aumentou significativamente o número de dispositivos móveis conectados a rede global [12].

Como os serviços ofertados pelas operadoras de redes móveis tendem a envolver soluções completamente baseadas no protocolo IP [1] e a as sessões de comunicações necessitam ser mantidas ao longo da movimentação do usuário através das redes, o gerenciamento de mobilidade IP é um fator de extreme impacto para as redes de comunicação [18].

Os padrões de gerenciamento de mobilidade IP do IETF, como MIPv6 [7] e PMIPv6 [8], dependem de unidades centrais que gerenciam e atuam tanto no plano de controle, quanto no plano de dados da rede e por serem baseados no roteamento tradicional de pacotes IP, apresentam alguns desafios como roteamento sub-otimizado, pouca escalabilidade, sobrecarga de processamento nas unidades centrais e pouca granularidade do serviço de mobilidade.

Uma alternativa para lidar com os problemas intrínsecos da centralização e os custos associados, é um novo conceito chamado de gerenciamento de mobilidade distribuído [9]. A principal característica desse novo conceito é a separação clara entre as ações realizadas no plano de controle e no plano de dados. As funções de plano de dados são distribuídas ao longo dos equipamentos na borda da rede para aproximar o agente de mobilidade do usuário, implementando uma abordagem de rede mais plana [10]. Nessa abordagem o tráfego encaminhado para um *mobile node* não precisa atravessar um ponto central na rede, onde o tráfego é tratado pelo agente mais próximo do usuário.

As soluções DMM podem ser categorizadas em dois tipos, dependendo do nível de distribuição do plano de controle [4, 10, 11]: Parcialmente distribuído – o plano de dados é completamente distribuído na rede, mas o plano de controle é centralizado em pontos de controle na rede; e Completamente distribuído – tanto o plano de dados, quanto o plano de controle são distribuídos ao longo da rede, não existe pontos centrais de controle.

Este artigo apresenta a implementação e os resultados obtidos em um ambiente real de experimentação de uma proposta baseada no paradigma de rede SDN para o gerenciamento de mobilidade distribuido através de fluxos IP bidirecionais, chamada de SDN-DMM [21].

O restante deste artigo está organizado da seguinte maneira: Seção 2 descreve a proposta SDN-DMM [21] e o ambiente de implementação; Secção 3 apresenta os cenários implementados e os resultados obtidos e a secção 4 encerra o artigo com as conclusões e trabalhos futures..

2. Implementação SDN-DMM

Esta seção introduz a proposta SDN-DMM [21], a qual utilize a abordagem SDN para o gerenciamento de mobilidade distribuído com fluxos IP e apresenta a implementação da proposta SDN-DMM em um cenário de real de experimentação utilizando equipamentos e *software*s de mercado, descrevendo a arquitetura, topologia, cenários e métricas analisadas.

SDN-DMM é uma proposta DMM *network-*

based parcialmente distribuída baseada na arquitetura SDN onde a mobilidade IP é suportada pela infraestrutura de rede por meio do ajuste automático dos fluxos IP no plano de dados realizado pelo controlador SDN com o protocolos *OpenFlow* de acordo com o *handover* executado pelo usuário.

A proposta utiliza uma abstração da camada *northbound* SDN, chamada de *Intent*, que informa ao controlador SDN determinada comunicação entre *host*s que precisa ser implementada na rede. O controlador com uma visão geral da topologia de rede, cria e ajustas as regras *OpenFlow* de acordo com o cenário de mobilidade, então o plano de dados da rede pode encaminhar o tráfego direcionado ao *mobile node* de forma direta em qualquer ponto da rede que ele estiver conectado. A figura 01 ilustra o processo realizado, onde:

1. *Mobile node* MN1 executa um *handover* da NB para a eNB, mantendo o seu endereço IP original;
2. Switch SB identifica a presença do *mobile node* com um IP diferente do seu escopo de endereçamento e informa o evento ao controlador C1;
3. Controlador C1 detecta uma mudança de topologia em relação ao *mobile node* MN1, recalcula um novo caminho and envia novas regras *OpenFlow* para reajustar os links de comunicação utilizados;
4. O novo fluxo IP bidirecional é estabelecido na rede, o que permite que os pacotes endereçados ao *mobile node* MN1 sejam corretamente encaminhados de acordo com o fluxo IP, ao inveis do roteamento tradicional. Por tanto o roteamento geral e as outras comunicações na rede não são afetadas.

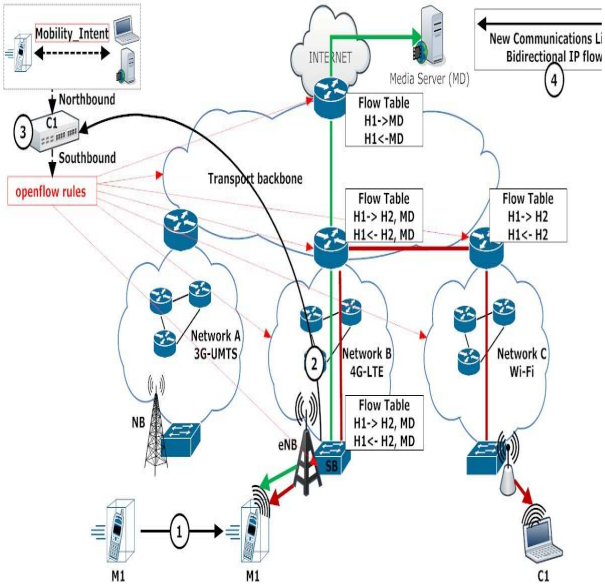


Figura 01 – proposta SDN-DMM para o gerenciamento de mobilidade IP [21]

A implementação consiste em utilizar a proposta SDN-DMM para prover a continuidade da sessão IP no gerenciamento de mobilidade, com um controlador SDN ajustando o plano de dados da rede, utilizando o protocolo *OpenFlow*, quando um *mobile node* realiza um *handover* da sua rede, rede A, para uma nova rede, rede B, com conexões ativas com um servidor.

O controlador utilizado para a implementação da proposta SDN-DMM foi o Open Network Operating System (ONOS), sendo implementado na modalidade *container* utilizando o *software* Docker no sistema operacional Ubuntu Server, utilizando os seguintes equipamentos com as respectivas funções:

- 01 Switch *OpenFlow* = Switch Extreme X460-24t, denominado OFS1;
- 02 *Access Point*s = Cisco WRT160N, denominados AP-A e AP-B;
- 03 Notebooks = VAIO Fit15F, denominados *Mobile Node* (MN1), Media Server (MD1) e Controlador SDN (C1).

As métricas de rede selecionados para avaliar o desempenho da implementação foram:

1. Taxa de vazão UDP (Throughput UDP);
2. Taxa de vazão máxima TCP (Throughput TCP);
3. Perda de pacotes UDP;
4. Perda de pacotes ICMP;
5. Latência de RTT com ICMP;
6. Latência de *Handover*

Sendo que inicialmente, para permitir a comparação do desempenho entre o encaminhamento de pacotes com roteamento tradicional e o encaminhamento por fluxos IP bidirecionais com *OpenFlow*, as métricas de 1 a 5 foram analisadas para estes dois ambientes sem aplicar o gerenciamento de mobilidade, a fim de verificar possíveis interferências nos resultados do gerenciamento de mobilidade utilizando SDN-DMM devido as características intrínsecas da abordagem SDN.

Todas as métricas foram analisadas para dois cenários de *handover*, sendo:

- SDN-DMM COM *HANDOVER* MANUAL POR CHAVEAMENTO

  Com o objetivo de verificar diretamente o impacto que a proposta SDN-DMM tem sobre as métricas de rede analisadas, neste cenário os *host*s MN1 e MD1 são conectados diretamente ao switch *OpenFlow* através de cabos UTP, onde o *handover* do *host* MN1 da rede A para a rede B é realizado de forma manual através da desconexão deste da

porta associada à rede A e a sua conexão a porta associada à rede B no switch *OpenFlow* enquanto mantém uma sessões ativas com o MD1. O intuito deste cenário é avaliar o impacto causado pela abordagem SDN-DMM desconsiderando o *handover* executado na parte à rádio da rede.

- SDN-DMM COM *HANDOVER* AUTMÁTICO POR *WIRELESS*

  Com o objetivo de verificar como a proposta se comporta em um cenário real de *handover*, ou seja, quando é implementada em conjunto com o *handover* realizado na parte à rádio da rede, neste cenário o MN1 se conecta ao *Access Point* AP-A e realiza o *handover* para o *Access Point* AP-B enquanto mantém uma sessão ativa com o *host* MD1. Deste modo os resultados obtidos refletem uma melhor aproximação da implementação completa da proposta para o gerenciamento de mobilidade.

Em relação aos *softwares*, sistemas operacionais e protocolos principais utilizados para avaliação das métricas de rede selecionadas, o *software* livre iPerf foi utilizado para a geração de tráfego TCP/UDP com o intuito de medir os parâmetros de *throughput* e de perda de pacotes, sendo que este último parâmetro também foi analisado através da geração de mensagens com o protocolo ICMP, utilizado também para a avaliação da latência RTT.

Já o *software* livre Wireshark foi utilizado para a captura do tráfego gerado com o objetivo de mensurar a latência do *handover*, através da utilização das marcações de tempo dos pacotes recebidos pelo servidor MD, onde também foi utilizado pacotes ICMP para a mensuração de tal métrica.

A arquitetura geral, utilizada como base para a implementação dos ambientes e cenários descritos, é composta por cinco redes principais e seus elementos de acordo com o descrito a seguir e observado na figura 02:

- Rede Internet: servidor MD1. Tráfego destinado as redes externas;
- Rede Cliente A: *host* MN1 e *access point* AP-A. Tráfego destinado aos clientes da rede A;
- Rede Cliente B: *access point* AP-B. Tráfego destinado aos clientes da rede B;
- Rede de Gerência: controlador C1 e switch OFS1. Tráfego de controle SDN e para gerência do switch;
- Rede de Acesso Out-of-band (OOB): Terminal remoto e controlador C1. Acesso gerencial ao controlador C1

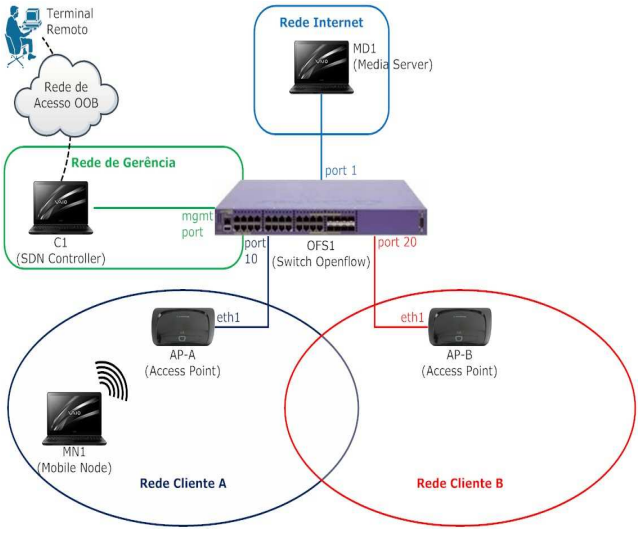


Figura 02 – Topologia de Implementação SDN-DMM

3. Testes e Resultados

Esta seção apresenta os testes realizados e os resultados obtidos na implementação real da proposta SDN-DMM com a análise das métricas de rede de taxa de vazão, perda de pacotes, latência RTT e latência de *handover* de acordo com o descrito na seção 2, sendo os ambientes e cenários implementados:

- Roteamento Tradicional sem *handover*;
- SDN-DMM sem *handover*;
- SDN-DMM com *handover* manual por chaveamento;
- SDN-DMM com *handover* automático por *wireless*.

3.1 Roteamento Tradicional sem *handover*

Neste cenário a comunicação entre os *hosts* MN1 e MD1 foi implementada através do roteamento tradicional IP através da conexão direta dos *hosts* por cabos UTP ao switch OFS1 em suas redes de origem de acordo com a topologia de rede apresentada na figura 02.

Não foi utilizado nenhum mecanismo de SDN ou cenário de mobilidade, tratando-se de uma comunicação realizada por meio do roteamento de pacotes entre as redes baseado na tabela de roteamento do switch OFS1.

O objetivo deste cenário é avaliar o desempenho da taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT, de modo a obter dados para realizar uma análise comparativa destas métricas com o ambiente SDN.

Com a comunicação devidamente estabelecida, o *software* iPerf foi utilizado para geração de tráfego UDP a uma taxa de 100 Mbps, de modo a avaliar a taxa de vazão e perda de pacotes, e para geração de tráfego TCP, de modo a avaliar a taxa de vazão máxima, durante o intervalo de 5 minutos. Sendo executado em modo cliente no *host* MN1 e em

modo servidor no *host* MD1. Os resultados obtidos podem ser observados na tabela 01.

Tabela 01 – Resultados para o cenário de roteamento tradicional

Teste	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	100 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	523 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,052%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	<1 ms

3.2 SDN-DMM sem *handover*

Utilizando a abordagem SDN-DMM, neste cenário a comunicação entre os *hosts* MN1 e MD1 foi implementada através de fluxos IP bidirecionais definidos pelo controlador C1 através do uso de Intents.

Os *hosts* foram conectados diretamente ao switch OFS1 por cabos UTP em suas redes de origem do mesmo modo que no cenário de roteamento tradicional, realizando os mesmos testes para obter os dados de taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT, a fim de se comparar o desempenho com o ambiente de roteamento tradicional. Obtendo os resultados apresentados na tabela 02.

Tabela 02 – Resultados para o cenário SDN-DMM

Teste	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	100 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	522 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,11%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	<1 ms

3.3 SDN-DMM com *handover* manual por chaveamento

A partir da implementação do cenário SDN-DMM sem *handover* anterior, com os *hosts* conectados diretamente ao switch OFS1, o *handover* do *host* MN1 foi realizado de forma manual através do seu chaveamento entre a porta do switch associada à rede A e a porta do switch associada à rede B durante a realização dos testes para coletar a taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT durante o *handover* do *host*.

Neste cenário também foi analisada a métrica de latência de *handover* através de pacotes TCP gerados pelo iPerf e através de pacotes ICMP. Com o uso dos mesmos parâmetros utilizados anteriormente para o *software* iPerf, a taxa de vazão e perda de pacotes UDP obtidas são apresentadas na figura 3, onde podemos observar que o *handover* é realizado no intervalo entre 50 a 60 segundos da execução do teste, onde ocorre perda de pacotes e uma redução da taxa de vazão momentaneamente, onde logo após a execução do *handover* voltam a normalidade. Obtendo ao final uma taxa média de vazão UDP de 99 Mbps, com uma perda de pacotes de 0,96%.

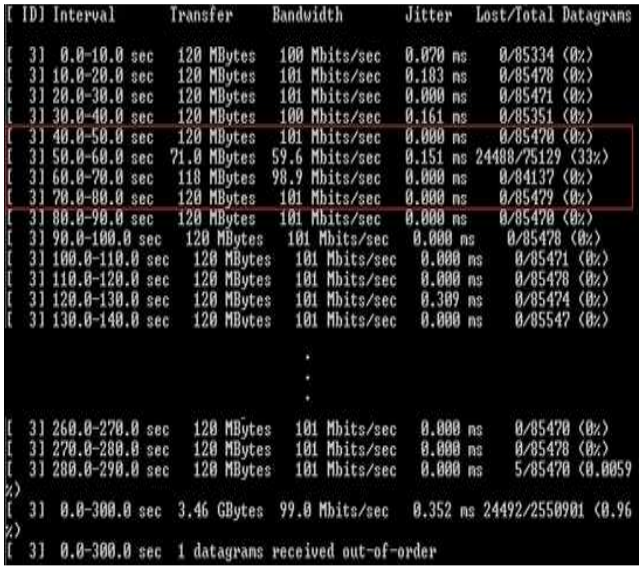


Figura 03 - Teste de vazão e perda de pacotes UDP para SDN-DMM com *handover* manual por chaveamento

Na figura 4 observamos que o *handover* foi executado no intervalo entre 50 e 60 segundos, onde ocorreu momentaneamente a redução da taxa de vazão TCP, obtendo ao final a média de 418 Mbps. Para a perda de pacotes e latência RTT com ICMP, obtivemos uma perda de 1% e latência média inferior a 1 ms de acordo com a figura 5.

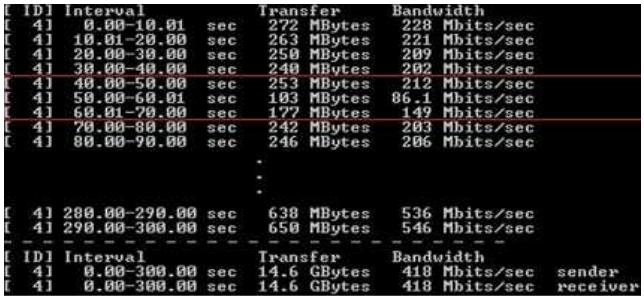


Figura 4 - Teste de vazão máxima TCP para SDN-DMM com *handover* manual por chaveamento.

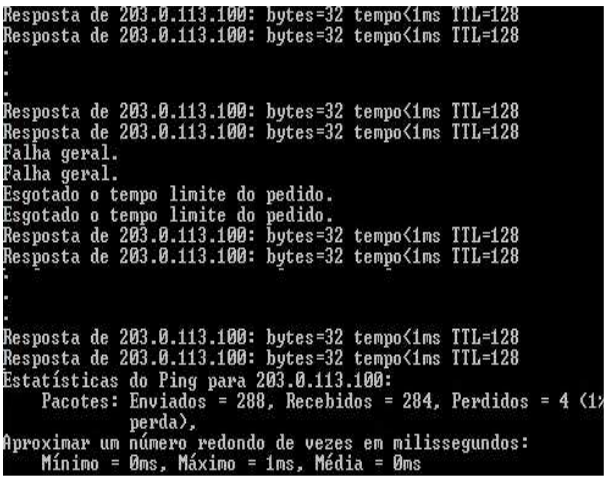


Figura 5 - Perda de pacotes e latência RTT com ICMP para SDN-DMM com *handover* manual por chaveamento.

Observamos que o *handover* é executado no momento em que ocorre a mensagem de Falha geral na figura 5, devido à desconexão física do *host* e é finalizado em seguida a perda de pacotes indicada pela mensagem Esgotado o tempo limite do pedido ao receber respostas positivas do protocolo ICMP.

Para análise da latência de *handover*, durante a realização do teste de vazão TCP e ICMP, foram coletados no servidor os pacotes recebidos e comparados os intervalos entre o último pacote recebido ao *host* MN1 ser desconectado e o primeiro depois de executado o *handover*. O resultado para a latência de *handover* TCP pode ser observado na figura 6.

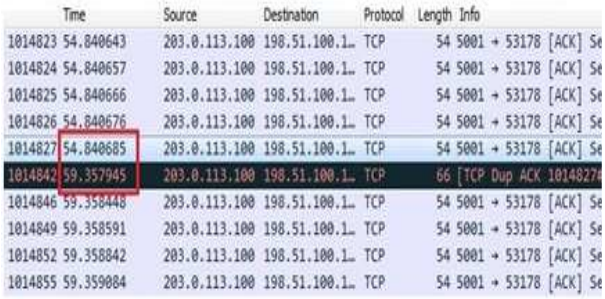


Figura 6 - Latência de *handover* TCP para SDN-DMM com *handover* manual por chaveamento.

Observamos na figura 6 que houve uma latência de aproximadamente 4,517 segundos entre os pacotes TCP recebidos pelo MD1 durante o processo de *handover* executado pelo MN1. A latência de *handover* ICMP pode ser observada na figura 7, onde obtivemos aproximadamente 10,67 segundos.

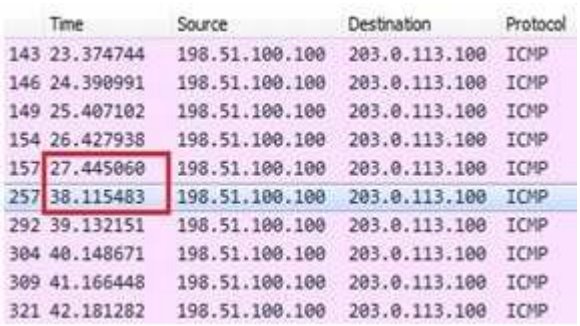


Figura 07 - Latência de *handover* ICMP para SDN-DMM com *handover* manual por chaveamento

Os resultados obtidos para o cenário SDN-DMM com *handover* manual por chaveamento são apresentados de forma consolidada na tabela 3.

Tabela 03 – Resultados para o cenário SDN-DMM com *handover* manual por chaveamento

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	99 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	418 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,96%
Perda de pacotes	Ping/ICMP	300 segundos	1%
Latência RTT	Ping/ICMP	300 segundos	<1 ms
Latência de *handover*	Iperf/TCP	300 segundos	4,517 segundos
Latência *handover*	Ping/ICMP	300 segundos	10,67 segundos

3.4 SDN-DMM com *handover* automático por *wireless*

Este cenário compreende a implementação da topologia da figura 2, onde estão presentes duas redes *wireless* as quais o *host* MN1 realizará um *handover* mantendo ativa a comunicação com o *host* MD1.

Para realizar o *handover* de forma automática sem necessitar de protocolos adicionais, os *access point*s foram configurados com o mesmo SSID e senha de autenticação, utilizado a banda 2,4 GHz com o canal 6 para o acess point AP-A, responsável pela conectividade à rede A e o canal 11 para o *access point* B, responsável pela conectividade à rede B. Estes foram dispostos de acordo com a topologia de modo a existir uma sobreposição da áreas de cobertura que permita o *host* MN1

executar o *handover* adequadamente, selecionando o *access point* com o sinal mais intenso para dado momento.

Por ser tratar de um ambiente sem fio, sujeito a interferências diversas, para obtermos resultados que permitam analisar de maneira consistente o impacto que a proposta SDN-DMM possui sobre as métricas de rede, os testes foram executados em três etapas:

1. Etapa 1 – MN1 conectado à Rede A;
2. Etapa 2 – MN1 conectado à Rede B;
3. Etapa 3 – *Handover* do MN1 .

A tabela 4 e 5 a seguir apresentam os resultados obtidos respectivamente para as etapas 1 e 2.

Tabela 04 – Resultados para o cenário SDN-DMM com *handover* automático – MN1 conectado à rede A

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	88,9 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	84 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	2 ms

Tabela 05 – Resultados para o cenário SDN-DMM com *handover* automático – MN1 conectado à rede B

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	25,4 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	10,4 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,3%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	15 ms

Para a etapa 3 de *handover* entre as redes *wireless*, A figura 8 ilustra o estado da conexão Wi-Fi do *host* MN1 durante o seu *handover* da rede A para a rede B, ilustrando o aumento da intensidade de sinal em relação ao tempo do *access point* AP-B em relação ao *access point* AP-A durante a sua movimentação. O *handover* foi executado com os testes de taxa de vazão UDP, taxa de vazão máxima TCP, perda de pacotes UDP, perda de pacotes ICMP e latência RTT em andamento.

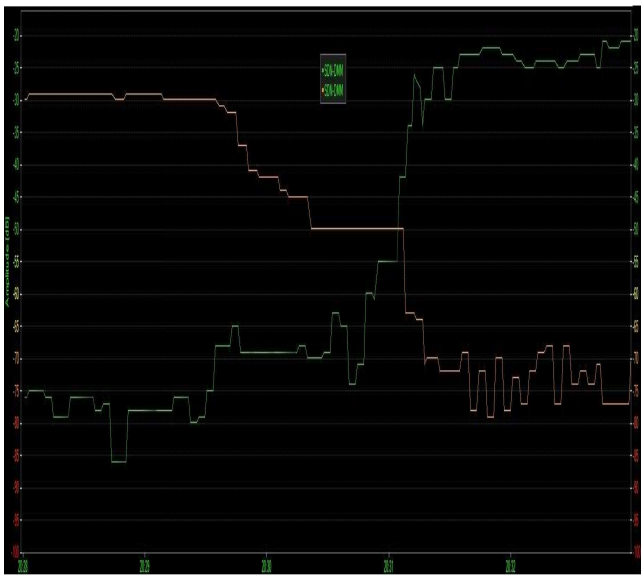


Figura 8 - Movimentação do *host* MN1 entre as redes A e B durante o *handover*.

Podemos observar na figura 8, que apresenta no eixo das abscissas a relação crescente de tempo, que durante a movimentação do MN1 a intensidade do sinal do AP-A diminui, na medida em que o sinal do AP-B aumenta. Quando o sinal do AP-B fica suficientemente maior do que o AP-A, o MN1 realiza o *handover* para a rede B.

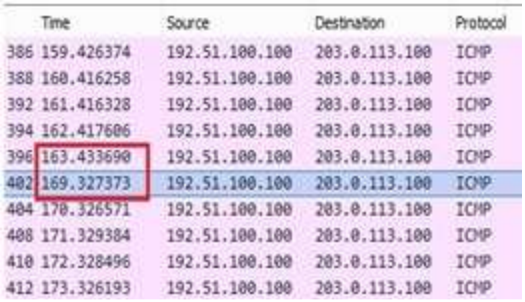
Para análise da latência de *handover*, durante a realização do teste de vazão TCP e ICMP, foram coletados no servidor os pacotes recebidos e comparados os intervalos entre o último pacote recebido antes do *host* MN1 realizar o *handover* da rede A e o primeiro pacote recebido logo após o MN1 possuir conectividade na rede B. O resultado para a latência de *handover* TCP pode ser observado na figura 9.

	Time	Source	Destination	Protocol	Length	Info
603072	74.270288	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603073	74.270565	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603074	74.270611	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603075	74.270630	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603076	74.270651	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603088	78.844312	203.0.113.100	198.51.100.100	TCP	1514	[TCP Spurious Retransm
603090	78.962652	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603091	78.968679	203.0.113.100	198.51.100.100	TCP	1186	53928 → 5201 [PSH, ACK
603092	78.968887	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603093	78.968967	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603094	78.969061	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603068	74.268228	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq
603069	74.268246	203.0.113.100	198.51.100.100	TCP	1514	53928 → 5201 [ACK] Seq

Figura 09 - Latência de *handover* TCP para SDN-DMM com *handover* automático por *wireless*.

Observamos na figura 9 que houve uma latência de aproximadamente 4,573 segundos entre os pacotes TCP recebidos pelo MD1 durante o processo de *handover* executado pelo MN1. A

latência de *handover* ICMP pode ser observada na figura 10, onde obtivemos aproximadamente 5,893 segundos.



Time	Source	Destination	Protocol
386 159.426374	192.51.100.100	203.0.113.100	ICMP
388 160.416258	192.51.100.100	203.0.113.100	ICMP
392 161.416328	192.51.100.100	203.0.113.100	ICMP
394 162.417606	192.51.100.100	203.0.113.100	ICMP
396 163.433690	192.51.100.100	203.0.113.100	ICMP
402 169.327373	192.51.100.100	203.0.113.100	ICMP
404 170.326571	192.51.100.100	203.0.113.100	ICMP
408 171.329384	192.51.100.100	203.0.113.100	ICMP
410 172.328496	192.51.100.100	203.0.113.100	ICMP
412 173.326193	192.51.100.100	203.0.113.100	ICMP

Figura 10 - Latência de *handover* ICMP para SDN-DMM com *handover* automático por *wireless.*

Com o uso dos mesmos *softwares* e parâmetros utilizados anteriormente, os resultados dos testes, iniciado antes do processo de *handover* e finalizado após, são apresentadas na tabela 6.

Tabela 06 – Resultados para o cenário SDN-DMM com *handover* manual por chaveamento

Métrica	Ferramenta/Protocolo	Tempo de execução	Resultado médio
Vazão UDP	Iperf/UDP	300 segundos	63,3 Mbps
Vazão máxima TCP	Iperf/TCP	300 segundos	34,8 Mbps
Perda de pacotes	Iperf/UDP	300 segundos	0,82%
Perda de pacotes	Ping/ICMP	300 segundos	0%
Latência RTT	Ping/ICMP	300 segundos	10 ms
Latência de *handover*	Iperf/TCP	300 segundos	4,573 segundos
Latência de *handover*	Ping/ICMP	300 segundos	5,893 segundos

4. Conclusões e Trabalhos Futuros

Verificamos que o ambiente SDN não causa impactos negativos para as métricas de rede em comparação com o ambiente de roteamento tradicional. Onde os resultados obtidos demonstram o mesmo desempenho alcançado para o encaminhamento de pacotes baseado no endereço de destino IP e para o encaminhamento de pacotes baseado no fluxo IP bidirecional utilizado na proposta SDN-DMM. A tabela 7 apresenta os resultados destes dois cenários.

Tabela 07 – Comparação das métricas entre Roteamento Tradicional e SDN-DMM.

Métrica	Roteamento Tradicional	SDN-DMM
Vazão UDP	100 Mbps	100 Mbps
Vazão máxima TCP	523 Mbps	522 Mbps
Perda de pacotes	0,052%	0,11%
Perda de pacotes	0%	0%
Latência RTT	<1 ms	<1 ms

Analisando em conjunto a tabela 7, a tabela 8 e os resultados obtidos nos experimentos realizados, observamos que a proposta SDN-DMM também não acarreta em perda de desempenho quando o *mobile node* está conectado a uma rede estrangeira. Ao *mobile node* executar um *handover* para uma nova rede, os valores obtidos para as métricas analisadas são os mesmos para um *host* nativo àquela rede. Isso pode ser observado nos testes realizados nos cenários de *handover* com o SDN-DMM e na tabela 8.

Tabela 08 – Métricas SDN-DMM com *handover* manual e automático

Métrica	SDN-DMM com *handover* manual por chaveamento	SDN-DMM com *handover* automático por *wireless*
Vazão UDP	99 Mbps	63,3 Mbps
Vazão máxima TCP	418 Mbps	34,8 Mbps
Perda de pacotes (UDP)	0,96%	0,82%
Perda de pacotes (ICMP)	1%	0%
Latência RTT	<1 ms	10 ms
Latência de *handover* (TCP)	4,517 segundos	4,573 segundos
Latência de *handover* (ICMP)	10,67 segundos	5,893 segundos

Os valores próximos obtidos para a métrica de latência de *handover* (TCP) para o cenário de *handover* manual por chaveamento e para o cenário de *handover* automático por *wireless*, são devidos ao balanceamento entre os fatores que prejudicam a medição dos mesmos em cada cenários.

No cenário de *handover* manual, à desconexão manual do MN1 no chaveamento entre as portas do switch OF1 causar uma interrupção abrupta da comunicação e do estado da porta nos equipamentos, que ao serem reconectados, é necessário aguardar um intervalo de tempo para que o estado da porta fique ativo, interferindo assim na medição do tempo de latência de *handover* devido ao processo da abordagem SDN-DMM.

Já para o cenário de *handover* automático, o fator da interferência do ambiente *wireless* sem utilizar algum protocolo específico para *handover* entre *access point*, é o ponto que prejudica a medição da latência de *handover* da proposta, embora neste

caso a transição entre as redes seja mais suave.

Outro fator importante que deve ser considerado na análise dos valores obtidos para a latência de *handover*, é que a própria medição desta métrica através das ferramentas utilizadas, prejudica os resultados. Ao se ativar a captura de pacotes no *host*, parte da capacidade de processamento e de rede é direcionada para a ação de captura, o que reduz o desempenho das métricas que estão sendo medidas. Sendo assim, para um cenário de produção, são esperados valores melhores do que o observado.

A implementação da proposta SDN-DMM em um ambiente real de experimentação e os testes realizados demonstram que a abordagem utilizada para o gerenciamento de mobilidade distribuído baseada em uma arquitetura SDN atende de maneira satisfatória aos requisitos de comunicação e não acarreta em prejuízos as métricas de rede quando comparado ao cenário tradicional de roteamento.

O desempenho obtido para a taxa de vazão quando o *mobile node* encontra-se em mobilidade vai de encontro ao custo de entrega de pacotes analisado na modelagem analítica [22], demonstrando que a proposta sempre emprega a entrega de forma otimizada, possuindo então um menor custo em relação as outras abordagens analisadas neste trabalho.

A perda de pacotes e a latência de *handover* obtidas nos testem realizados podem ser considerados coerentes com os resultados obtidos com a modelagem analítica [21], considerando a baixa quantidade de mensagens necessárias no plano de controle para prover a mobilidade, reajustando o plano de dados.

Trabalhos em andamento envolvem a validação de um modelo e análises estatísticas com o foco para IP Flow data offloading e suporte a possibilidade de um modelo híbrido com a extensão do SDN para o *mobile node*, utilizando simultaneamente uma abordagem *client-based* e *network-based.*

Outros temas relevantes que não foram abordados sendo indicados para continuidade da pesquisa são as relações de mobilidade entre diferentes sistemas autônomos.

References

[1] KARIMZADEH M; VALTULINA, L; KARAGIANNIS, G. *Applying SDN/OpenFlow in virtualized LTE to support Distributed Mobility Management (DMM)*. In: Proceedings of the 4th International Conference on Cloud Computing and Services Science, CLOSER 2014, 3-5 Barcelona, Spain. 86. SciTePress, Abril 2014.

[2] KARIMZADEH, M; ZHAO, Z; HENDRIKS, L; SCHMIDT, R; FLEUR, S; BERG, H; PRAS, A; BRAUN, T; CORICI, M. *Mobility and Bandwidth Prediction as a Service in Virtualized LTE Systems*. 4th IEEE International Conference on Cloud Networking (CLOUDNET), 2015.

[3] VALTULINA, L; KARIMZADEH, M; KARAGIANNIS, G; HEIJENK, A; PRAS, A. *Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems*. IEEE GLOBECOM 2014.

[4] GIUST, F; BERNARDOS, C; OLIVIA, A. *HDMM: deploying client and network-based distributed mobility management.* Journal Telecommunications Systems, Volume 59 Issue 2, Pages 247-270, Junho 2015.

[5] SEITE, P; BERTIN, P; *Distributed Mobility Anchoring*. Draft IETF at https://tools.ietf.org/html/draft-seite-dmm-dma-00, Fevereiro 2012.

[6] ONLAB. *Introducing ONOS - a SDN network operating system for Service Providers.* Whitepaper http://onosproject.org/wp-content/uploads/2014/11/Whitepaper-ONOS-final.pdf [accessed Nov-2015].

[7] Perkins, C., Johnson, D., & Arkko, J. (2011). Mobility Support in IPv6. RFC 6275.

[8] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K.,&Patil, B. (2008). Proxy Mobile IPv6. RFC 5213.

[9] Chan,H.A.,Yokota,H., Xie, J., Seite, P.,&Liu,D. (2011). Distributed and dynamicmobility management in mobile internet:Current approaches and issues. Journal of Communications, 6(1), 4–15.

[10] Chan, H. (2014). Requirements for Distributed Mobility Management. RFC 7333.

[11] Yokota, H., Seite, P., Demaria, E., & Cao, Z. (2010). Use case scenarios for Distributed Mobility Management. Internet-Draft (work in progress), draft-yokota-dmm-scenario-00.txt.

[12] KARIMAZDEH, *SDN Mobility Management*, 2014

[13] Estatísticas de operação da Rede Ipê – Backbone da Rede Nacional de Ensino e Pesquisa (RNP), 2016.

[14] Estatísticas de operação do Ponto de Presença da RNP no Distrito Federal (PoP-DF), 2016.

[15] Giust F., Bernardos C., Oliva A. (2014). Analytic Evaluation and Experimental Validation of a Network-Based IPv6 Distributed Mobility Management Solution. In IEEE Transactions on Mobile Computing, vol. 13, no. 11, November 2014.

[16] J. Lee and Y. Kim, "PMIPv6-based Distributed Mobility Management," IETF draft, draft-jaehwoon-dmm-pmipv6-04, June 18 2015.

[17] BERNARDOS, C.; OLIVIA, A.; GIUST, F. A PMIPv6-based solution for Distributed Mobility Management. In IETF draft, draft-bernardos-dmm-pmip-06, Março 2016.

[18] BERNARDOS C., OLIVA A., SERRANO P., et al. (2014). *An Architecture for Software Defined Wireless Networking*. In IEEE Wireless Communications, vol. 21, no.3, pag 52-61.

[19] CISCO SYSTEM. *Routers Products and Solotions.* http://www.cisco.com/c/en/us/products/routers/buyers-guide.html [accessed May-2016].

[20] JUNIPER NETWORKS. *MX Series Plataforms.* http://www.juniper.net/us/en/products-services/routing/mx-series/compare/#a=VMX,P5,P10,P40,P80,P104,P240,P480,P960,P2010,P2020 [accessed May-2016].

[21] CORDOVA, R; GONDIM, P. 2016. *SDN-DMM for Intelligent Mobility Management in Heterogeneous Mobile IP Networks*.

[22] CORDOVA, R; GONDIM, P. 2016. *Packet Delivery and Routing Optimization for DMM with SDN-DMM*.