# Análise da Viabilidade de uso do protocolo IP *Anycast* em Soluções de Alta Disponibilidade e Balanceamento de Carga

**Braian Jacomelli Piovesan[1], Ricardo Tombesi Macedo[2], Sidnei Renato Silveira[2]**

[1]Curso de Bacharelado em Sistemas de Informação – [2]Departamento de Tecnologia da Informação (DTecInf) - Universidade Federal de Santa Maria (UFSM) – Campus Frederico Westphalen – RS – Brasil

braian.jacomelli@gmail.com, rmacedo1987@gmail.com, sidneirenato.silveira@gmail.com

***Abstract.*** *This paper presents a study about the IP Anycast protocol's usage as alternative to achieve high availability and load balance in geographic distributed services. It makes a comparation with another alternatives to achieve the desired results, by presenting some related articles. A simulation of a test environment obtained the results shown here. It concludes that the IP Anycast protocol is a viable alternative to achieve high availability and load balance in geographic distributed services. At the test environment, was possible to see a decrease of about 3 times in the latency between the client and the server, besides ensuring the availability of the service even part of the servers fail.*

***Resumo.*** *Este artigo apresenta um estudo sobre o uso do protocolo IP Anycast como alternativa para obtenção de alta disponibilidade e balanceamento de carga em serviços geograficamente distribuídos. Uma comparação com outras alternativas para obter os resultados desejados é realizada após a apresentação de artigos relacionados ao tema. Os resultados obtidos por meio de simulações revelam que o protocolo IP Anycast é uma alternativa viável para implementação de alta disponibilidade em serviços geograficamente distribuídos. Em ambiente de testes, observou-se uma redução de até 3 vezes na latência entre o cliente e o servidor, além de garantir a disponibilidade do serviço mesmo quando ocorre a falha em parte dos servidores.*

## 1. Introdução

O crescimento da Internet e a transformação do conteúdo acessado faz com que, cada vez mais, seja necessário adotar técnicas que permitam um maior desempenho, com baixa latência, alta disponibilidade e escalabilidade. Conforme dados do projeto *Internet Live Stats,* o número de usuários conectados à internet entre os anos 2000 e 2015 passou de pouco mais de 1 bilhão para mais de 3 bilhões. Parte desse crescimento pode ser atribuído a maior acessibilidade dos dados, ao crescimento dos rendimentos globais e ao interesse das pessoas em ter acesso a internet, o que pode ser explicado pela pluralidade de opções disponíveis na internet, como entretenimento, compartilhamento de informações, acesso a serviços, entre outros. Uma estimativa da *Cisco* revela que em 2021, cerca de 81% do tráfego da internet será destinado a usuários finais composto por

vídeos e downloads. Nesse cenário, é preciso adotar medidas que permitam acompanhar as necessidades dos usuários e permitir a continuidade desse crescimento.

A Internet é um conjunto de Sistemas Autônomos ou ASs *(Autonomous Systems)* que se conectam por meio do protocolo IP (*Internet Protocol*) e trocam informações de roteamento com o uso do protocolo BGP (*Border Gateway Protocol*), conectando usuários a serviços (TANENBAUM, 2003). A conexão entre ASs normalmente é feita através de redes de alta capacidade e muitas vezes, essas redes percorrem grandes distâncias, como os cabos submarino de fibra ótica que ligam um continente a outro. Com a internet, os usuários conseguem obter informações e fazer o uso de serviços disponibilizados por provedores de conteúdo que podem estar em qualquer parte do mundo. Alguns usuários buscam entretenimento, como vídeos, blogs, música e troca de mensagens de texto. Outros utilizam a Internet como ferramenta de trabalho, utilizando sistemas *online*, videoconferências, serviços de email ou, até mesmo, sistemas de mapas GPS capazes de calcular com precisão o tempo estimado de um percurso, considerando informações recebidas de outros usuários desse mesmo sistema. Com uma variedade de serviços quase ilimitada, o acesso à Internet tem se tornado essencial na vida de muitas pessoas.

Apesar da Internet funcionar muito bem na maior parte do tempo, eventualmente ela está sujeita a falhas. Rompimentos de cabos em ASs de trânsito, defeitos em equipamentos como roteadores e servidores ou, até mesmo, situações que comprometam o fornecimento de energia para um datacenter inteiro são falhas que podem ocasionar a indisponibilidade ou perda de desempenho em serviços. Um exemplo recente foi a falha de uma das redes que conectam o Brasil ao resto do mundo, após o furacão Maria atingir Porto Rico, fazendo com que os equipamentos fossem desligados como medida de proteção. Essa situação fez com que usuários do Brasil sentissem dificuldades para acessar sites que dependiam dessa rede. (AMARAL, 2017). Além disso, nem sempre os servidores que disponibilizam algum serviço possuem os recursos adequados para suportar o crescimento do volume de acessos. Nessas situações, redes e serviços que não foram planejados para tolerar falhas normalmente ficam indisponíveis ou apresentam baixo desemprenho, o que pode causar prejuízos para as partes interessadas, seja o usuário, o provedor de conteúdo ou uma loja virtual que deixa de vender.

Existem várias formas de tornar um serviço disponível na Internet tolerante a falhas, porém, nem todas as formas se aplicam a qualquer tipo de serviço. Com o objetivo de eliminar o ponto de falha criado pelo balanceador de carga em conjuntos de servidores virtuais Linux, Alvim, Grossmann e Dantas (2002), propuseram o uso de uma solução envolvendo Linux-AD (Alta Disponibilidade), possibilitando a inclusão de um balanceador de carga redundante. Já Oliveira (2011), objetivando aprimorar a escalabilidade em Ambientes Virtuais Distribuídos, propôs a criação do *AnyAVDNet*, um *framework anycast* na camada de aplicação, que iria distribuir a carga entre servidores com base em métricas. Alcântara Filho e Rodrigues (2016) fizeram um estudo sobre o índice de falhas em conexões TCP persistentes em função do tempo de vida em um ambiente Anycast, onde identificaram que em situações onde o tempo de vida das sessões TCP é inferior a 15 segundos, esse valor é de 0,08%, ou seja, um valor quase insignificante. Essas abordagens resolvem parte do problema, pois a primeira garante que o serviço não irá falhar, mas não contempla a possibilidade de que todo o datacenter fique indisponível, enquanto a segunda concentra-se apenas em Ambientes Virtuais Distribuídos, criando um framework específico para esse caso.

A descentralização dos servidores utilizando o protocolo IP *Anycast* para obter tolerância a falhas, desempenho e escalabilidade é o objetivo deste trabalho. A implementação de alta disponibilidade baseada em roteamento IP (*Internet Protocol*) *Anycast* consiste em manter servidores geograficamente distribuídos, visíveis para o usuário por meio do mesmo endereço IP, que se comunicam entre si para manter os serviços atualizados e não dependem de uma gerência centralizada para realizar a distribuição da carga. Isso é possível pois, conforme Partridge, Mendez e Milliken (1993), quando um host transmite um datagrama para um endereço *anycast,* os protocolos de roteamento são responsáveis por entregar esse datagrama para, ao menos um servidor *anycast*. Essa técnica permite obter ganhos de desempenho, considerando que o serviço fica mais próximo ao usuário, podendo ser distribuído em vários datacenters conforme o interesse do provedor de serviços. O protocolo de roteamento se encarrega de encontrar um servidor próximo ao usuário, e, caso este falhe, encaminhar o tráfego para outros nós que possuem o mesmo endereço IP. Ganhos em escalabilidade também são possíveis, já que a adição de novos servidores para o balanceamento de carga nas regiões onde o número de usuários cresce é bastante simples. Por permitir a descentralização, o IP Anycast reduz o impacto das falhas na rede e também nas falhas de servidores, além de aproximar o usuário do serviço consumido.

Esse artigo apresenta uma avaliação de desempenho do uso do protocolo IP *Anycast* para prover balanceamento de carga e alta disponibilidade em serviços geograficamente distribuídos, tirando proveito do uso de protocolos de roteamento para realizar a distribuição de carga entre os servidores disponíveis A implementação proposta será avaliada quanto aos ganhos de desempenho, redução de latência do usuário até o servidor e balanceamento de carga.

As simulações para avaliar a solução proposta foram conduzidas em um ambiente de testes, utilizando o software (GNS3) para virtualizar uma série de roteadores, servidores e clientes. Nesse cenário, foram feitos testes onde os clientes enviaram pacotes ICMP para os servidores, gerando carga nos mesmos. Essa carga foi distribuída por meio do protocolo de roteamento aplicado nos roteadores. Também foram estabelecidas sessões BGP (*Border Gateway Protocol*) entre os clientes e os servidores *Anycast* para avaliar o funcionamento no caso de falha de um dos servidores. Percebeu-se que, após a falha de um dos servidores, a sessão BGP automaticamente passou a ser estabelecida com o outro servidor.

O artigo está estruturado da seguinte forma: a Seção 2 apresenta o referencial teórico, onde são abordados os conceitos sobre os principais temas que envolvem este trabalho. Na Seção 3, são apresentados alguns trabalhos relacionados para a composição do estado da arte. A Seção 4 apresenta a solução implementada, onde são efetuadas as configurações e testes propostos no trabalho. E por fim, temos a Seção 5, onde são tratadas as considerações finais com base em tudo o que foi visto ao longo do artigo.

## 2. Referencial Teórico

Esta seção apresenta o referencial teórico que será utilizado neste artigo, apresentando conceitos de protocolos de roteamento, serviços de rede, aplicações distribuídas e alta disponibilidade.

## 2.1 Redes de computadores

Uma rede de computadores é formada por um conjunto de módulos processadores (equipamentos) capazes de trocar informações e compartilhar recursos através de um sistema de comunicação (SOARES, LEMOS e COLCHER, 1995). As redes de computadores podem ser classificadas, de acordo com sua área de abrangência, em PAN (*Personal Area Network*), LAN (*Local Area Network*), MAN (*Metropolitan Area Network*), WAN (*Wide Area Network*) e Internets (TANENBAUM, 2003).

Para o bom desenvolvimento desse trabalho, os conceitos de LAN, WAN e Internets serão analisados. Uma rede LAN é uma rede particular que opera dentro de um espaço geográfico restrito, geralmente dentro do mesmo edifício, residência ou empresa. Já as redes WAN operam em áreas geográficas maiores, podendo chegar a abranger um país ou continente. Redes WAN normalmente são utilizadas por empresas que possuem filiais em diversas cidades e precisam interconectar suas LANs. Por fim, as Internets, ou Redes Interligadas, são um conjunto de redes distintas se interligam (TANENBAUM, 2003).

### 2.1.1 O Protocolo IP

O protocolo IP pode ser considerado o elemento que mantém a Internet unida. Desde sua concepção, foi projetado com o objetivo de interligar redes, fornecendo a melhor forma possível de transportar datagramas da origem para o destino (TANENBAUM, 2003).

Um datagrama IP consiste em uma parte de cabeçalho e uma parte de texto. No cabeçalho do datagrama IP são encontrados os seguintes campos: o campo *Versão,* que contém a versão do protocolo usado; o campo *IHL (Internet Header Length),* que informa o tamanho do cabeçalho; o campo *Serviços diferenciados*, que é usado para marcar os pacotes com sua classe de serviço e transportar informações de notificação de congestionamento; o campo *Tamanho total*, que informa o tamanho total do datagrama, incluindo o cabeçalho e dados, seu tamanho máximo é 65.535 bytes; o campo *Identificação,* que permite ao *host* de destino que determine a qual datagrama pertence um fragmento recém-chegado; o campo *DF* (*Don't fragment*, ou *não fragmentar*) indica que o datagrama não deve ser fragmentado pelos roteadores no percurso; o campo *MF* significa *Mais fragmentos*, o último fragmento de um datagrama não contém esse conjunto de *bits*, indicando que todos os fragmentos já chegaram; o campo *Deslocamento de fragmento,* que indica a que ponto do datagrama atual o fragmento pertence; o campo *TTL* (*Time to live* ou *Tempo de vida*), que é um contador usado para limitar a vida útil de um pacote; o campo *Protocolo*, que informa a que processo o de transporte o datagrama deve ser entregue, sendo que os mais comuns são TCP e UDP (descrever as siglas); o campo *Checksum,* que é usado para detectar erros quando o pacote atravessa a rede; os campos *Endereço de origem* e *Endereço de destino*¸ que indicam o IP das interfaces de rede de origem e de destino; e, por último, o campo *Opções*, que foi projetado para que versões posteriores do protocolo incluam informações não presentes no original, porém, é aceito apenas parcialmente e pouco usado (TANENBAUM; WETHERALL, 2011).

Os endereços *IPv4 (Internet Protocol versão 4)* são compostos por 32 *bits*, organizados em grupos de 4 *bytes* separados por ponto, escritos em notação decimal, de

0 a 255. Cada endereço de 32 *bits* é composto por uma parte de rede, que é igual para todos os *hosts* na mesma rede e uma parte de *host*. Uma rede também é conhecida como prefixo, e são escritos dando o menor endereço IP no bloco de endereços. O tamanho do prefixo é determinado pelo número de *bits* na parte de rede, enquanto os *bits* restantes fazem parte do campo de *host* e podem variar. O tamanho do prefixo não pode ser deduzido a partir do endereço IP, portanto, os protocolos de roteamento devem transportar os prefixos para os roteadores, o que as vezes é descrito apenas pelo seu tamanho, como em um '/16' que é pronunciado como 'barra 16' (TANENBAUM; WETHERALL, 2011).

### 2.1.2 Algoritmos de Roteamento

O algoritmo de roteamento é a parte do *software* da camada de rede responsável pela decisão sobre a interface de saída a ser usada na transmissão do pacote de entrada (TANENBAUM, 2003).

Os algoritmos de roteamento podem ser classificados como estáticos (ou algoritmos não adaptativos) e dinâmicos (ou algoritmos adaptativos). Algoritmos de roteamento estáticos possuem suas tabelas de roteamento fixa ou alteradas de forma manual. Em contrapartida, os algoritmos adaptativos sofrem alterações em suas tabelas de roteamento em casos de mudanças na topologia de rede.

O modelo de roteamento *IP Anycast* propõe que um pacote será entregue ao membro ou *host* mais próximo de um grupo (PARTRIDGE et al., 1993). No roteamento por *anycast*, todos os membros de um grupo recebem um mesmo endereço IP e o algoritmo de roteamento irá encontrar o nó mais próximo.

Na Internet, grupos de roteadores sob o controle de uma autoridade administrativa são considerados *Sistemas Autônomos (Autonomous System, AS)*. Um AS pode ser um provedor de Internet, uma grande organização ou uma universidade (COMER, 2007).

Existem duas categorias de protocolos de roteamento: os IGPs (*Internal Gateway Protocol*) e os EGPs (*External Gateway Protocol*). Os roteadores dentro de um AS usam Protocolos de Roteador Interior, enquanto os roteadores que precisam trocar informações de roteamento com outro Sistema Autônomo usam Protocolos de Roteador Externo (EGP) (COMER, 2007).

A existência dessas duas categorias de protocolos de roteamento é necessária porque os objetivos de um IGP e de um EGP são diferentes. Um protocolo de *gateway* interior precisa movimentar pacotes da origem até o destino da forma mais eficiente possível. Ele não precisa se preocupar com política. Já um protocolo de *gateway* exterior precisa se preocupar com política, que, em geral, indicam quais pacotes devem ser transportados através daquele sistema autônomo (TANENBAUM, 2003). Neste trabalho, será utilizado o protocolo BGP (*Border Gateway Protocol*), um EGP utilizado para propagação de rotas na internet.

### 2.1.2 *Border Gateway Protocol* - BGP

Como já foi falado anteriormente, protocolos de *gateway* exterior tem como característica a necessidade de se preocuparem com política. As escolhas de políticas a

serem seguidas podem ser altamente individuais, porém, algumas políticas comuns são as políticas de serviço de trânsito e *peering*. Políticas de trânsito são implementadas quando um cliente paga outro ISP (*Internet Service Provider*) para transportar pacotes a qualquer outro destino na Internet. Já as políticas de *peering* são implementadas quando um AS se conecta a outro AS, geralmente em IXPs (*Internet Exchange Points*), com o objetivo de trocar tráfego, sem custos.

Para que um serviço de trânsito funcione, o provedor de trânsito deve anunciar as rotas de todos os destinos na Internet ao cliente pelo enlace que os conecta. Já o cliente, deve anunciar rotas somente para os destinos em sua rede, que serão propagadas para o resto da Internet pelo provedor de trânsito. Essas são políticas básicas para esse tipo de serviço.

As políticas básicas de *peering* consistem em dois ASs enviarem anúncios de roteamento um para o outro, para os endereços que residem em suas redes. As rotas recebidas não são propagadas para outros ASs, por isso se diz que *peering* não é transitivo.

O BGP é uma forma de protocolo por vetor de distância, no qual cada roteador registra o caminho utilizado por uma rota. O caminho consiste no roteador do próximo *hop* e na sequência de ASs, também chamada de caminho do AS. Essas informações são trocadas entre roteadores por meio de conexões TCP, o que oferece comunicação confiável. Um roteador, ao enviar uma rota para outro AS, inicia a mesma com seu próprio número de AS, construindo o caminho do AS e evitando *loops* de roteamento, pois, ao receber uma rota, um roteador verifica se seu próprio número de AS está no caminho do AS, e, caso esteja, a rota é descartada (TANENBAUM; WETHERALL, 2011).

Apesar do BGP ser um protocolo do tipo EGP, ele possui uma variante chamada iBGP (*Internal BGP*), usada para propagar as rotas de um lado para o outro dentro de um ISP. Conforme Tanenbaum e Wetherall (2011), a regra é que cada roteador no limite do ISP descobre todas as rotas vistas por todos os outros roteadores de limite.

Um roteador BGP pode descobrir rotas para um destino a partir de outros roteadores aos quais está conectado nos próximos ASs, podendo descobrir várias rotas para um mesmo destino. A construção de políticas para definir qual rota deve ser escolhida fica a critério do AS, porém, algumas estratégias básicas costumam ser usadas: que rotas de *peering* ou de clientes devem ter preferência em relação a rotas de trânsito, caminhos de AS mais curtos costumam ser melhores e, por último, rotas que têm um menor custo dentro do AS devem ser preferidas. As estratégias devem ser implementas em todos os roteadores, pois cada roteador escolhe suas rotas preferidas de forma individual (TANENBAUM; WETHERALL, 2011).

Atributos de caminho do BGP, inclusive os opcionais, que podem ser utilizados nas políticas implementadas pelo AS para influenciar o tráfego são apresentados na RFC (*Request For Comments*) 4271. Entre eles, estão os atributos *AS-PATH, MULTI_EXIT_DISC e LOCAL_PREFERENCE.* Neste trabalho, as políticas que foram criadas nos roteadores são limitadas a manipular o atributo *AS-PATH*. Este atributo é obrigatório e representa o caminho de AS pelo qual as informações de rotas passaram até chegar no roteador, pois, toda vez que um roteador BGP propaga uma

informação de rota para outro roteador BGP em outro AS, o primeiro deve adicionar a informação sobre seu próprio número de AS no *AS-PATH.*

## 2.2 Sistemas Distribuídos

Ao longo dos anos, diversas definições foram dadas para sistemas distribuídos, muitas delas divergentes entre si. Em linhas gerais, podemos definir que um sistema distribuído é um conjunto de computadores independentes que se apresenta a seus usuários como um sistema único e coerente (TANENBAUM; VAN STEEN, 2007).

Em sistemas distribuídos é possível utilizar vários computadores diferentes, e essas diferenças devem estar ocultas ao usuário, que pode interagir com o sistema de forma consistente, independentemente de onde ocorre essa interação. Além disso, espera-se que seja relativamente fácil expandir um sistema distribuído, devido a sua característica de possuir computadores diferentes que estão ocultos ao usuário. Esse sistema deve ficar continuamente disponível, mesmo que parte de seus nós esteja inoperante, sem que os usuários percebam, até que esses nós voltem a ficar operacionais.

Um sistema distribuído tem, por objetivo, facilitar o acesso a recursos compartilhados como dados, páginas *web* e serviços, podendo implementar níveis de transparência, ou seja, ocultar questões como acesso, localização, migração, relocação, replicação, concorrência e falha (TANENBAUM; VAN STEEN, 2007).

# 3. Estado da arte

Nesta seção serão apresentados alguns trabalhos relacionados, como o objetivo de identificar diferentes abordagens para o problema de alta disponibilidade e balanceamento de carga em serviços distribuídos.

## 3.1 Implementação de uma Arquitetura para Serviço Distribuído com Grande Disponibilidade em Ambiente Linux

Em seu trabalho, Alvim, Grossmann e Dantas (2002) abordam os conceitos de *IP Virtual* e de *Servidor Virtual,* desenvolvendo uma solução para o problema de falhas no balanceador de cargas de um servidor virtual, utilizando Linux Alta-Disponibilidade.

Para entender o conceito de Linux-AD é importante conhecer os conceitos de *IP virtual* e *heartbeat.* O primeiro se refere ao endereço *IP* de um serviço que está sendo executado em uma determinada máquina, e não o seu endereço *IP* real. O *heartbeat* é o programa responsável por monitorar os serviços em ambientes de alta disponibilidade, fazendo com que, em caso de falha do servidor principal, o servidor de *backup* assuma o endereço *IP virtual*.

O Linux-AD funciona nos dois servidores por meio do *heartbeat,* que faz com que o servidor principal assuma o *IP virtual* durante a inicialização, e, em seguida, promove a troca de mensagens entre os servidores para monitorar suas disponibilidades. Caso o *heartbeat* do servidor secundário identifique que o servidor primário está

indisponível, o servidor secundário irá assumir o endereço *IP virtual,* passando a prover os serviços que estão configurados. Caso o servidor principal volte a ficar disponível, a configuração de *nice failback* irá indicar qual dos dois deve ficar com o *IP virtual*. (ALVIM; GROSSMANN; DANTAS, 2002)

É importante destacar que essa implementação pode falhar quando ocorre um problema de rede, em que apenas um dos servidores fica indisponível e o *heartbeat* faz com que ambos assumam o *IP virtual*. Quando este servidor que havia sido desconectado da rede volta a se conectar, a rede é levada a um estado inconsciente, pois ambos os servidores estão respondendo pelos serviços, de forma que um dos servidores terá que ser reinicializado.

O servidor virtual Linux é um conjunto de servidores e um balanceador de carga que são vistos pelo cliente como se fossem um único servidor. Isso acontece pois o balanceador de carga se encarrega de repassar os pacotes para os servidores reais baseando-se nas informações de endereçamento *IP*, assemelhando-se ao funcionamento de um servidor proxy. O principal objetivo do servidor virtual é prover alto desempenho e alta disponibilidade para aplicações, através de redundância e balanceamento de carga entre máquinas. (ALVIM; GROSSMANN; DANTAS, 2002)

Neste trabalho, o programa *ldirectord* é executado no balanceador de carga, que faz a distribuição dos serviços e utiliza o envio de mensagens para monitorar o *status* de cada máquina, removendo-as do *cluster* caso apresentem falhas. O escalonamento é realizado por meio de um algoritmo que pode ser implementado de diversas formas, sendo que as mais utilizadas são *round-robin* e *conexão-mínima.* No *round-robin* cada servidor pode receber um peso. Assim, em uma rodada, cada servidor atenderá tantas requisições quanto indicado por seu peso. Já o algoritmo de conexão-mínima direciona as conexões para o servidor que possuir menor número de conexões estabelecidas, sendo interessante em situações onde exista uma coleção de servidores que possuem capacidade de processamento similar.

Algumas vantagens do servidor virtual, segundo Alvim, Grossmann e Dantas (2002):

- A utilização de soluções de *software* abertas e de *hardware* simples permitem implementações de baixo custo;
- Fornece redundância e alta disponibilidade permitindo que as máquinas reais recebam manutenção, sem gerar indisponibilidade nos serviços;
- Provê adaptabilidade, permitindo incrementar o processamento com a adição de novas máquinas ao *cluster* facilmente, assim como remover essas máquinas quando for desejado.

No artigo, o autor identifica o balanceador de carga como um ponto crítico de falha. Para sanar esse ponto de falha, faz a implementação do Linux-AD no balanceador de carga, utilizando um ambiente com cinco servidores, dos quais dois fazem o papel de balanceador de carga e três fazem o papel de servidores reais. Os balanceadores de carga foram chamados de LAICOZ e LAICOY, enquanto os servidores reais foram chamados de LAICO2, LAICO3 e LAICO4. O serviço HTTP (*HyperText Transfer Protocol*) foi utilizado nos testes.

Os testes consistiram em retirar de operação alguns servidores e depois inseri-los novamente no *cluster*. Os testes obtiveram sucesso, sendo que o serviço HTTP que foi

testado manteve-se disponível, assim como o balanceador de carga de *backup* funcionou.

## 3.2 Utilização do Protocolo *Anycast* em Ambientes Virtuais Distribuídos visando à Escalabilidade do Sistema

Segundo Singal e Zyda (1999), citado por Oliveira (2011), AVDs (Ambientes Virtuais Distribuídos) são sistemas que permitem a interação de múltiplos usuários em um ambiente compartilhado, mesmo que os usuários estejam geograficamente distantes, e possuem como características a sensação de espaço compartilhado, sensação de presença compartilhada, sensação de tempo compartilhado, meio de comunicação e meio de compartilhamento.

A camada de comunicação de um AVD precisa ser cuidadosamente planejada para atender a requisitos de baixa latência, confiabilidade e largura de banda. Os dados trafegam pela rede dentro de PDUs (*Protocol Data unit*) que não possuem um formato padronizado, existindo PDUs específicos para cada serviço. Segundo o artigo analisado, a topologia de comunicação de um AVD pode ser definida como cliente-servidor, *peer-to-peer* ou híbrida.

A topologia cliente-servidor é composta por um servidor central que mantém todas as informações sobre o ambiente. As vantagens dessa topologia são a facilidade de implementação e a garantia de que não irão ocorrer inconsistências dos dados entre os participantes (CORREIA, 2005 apud OLIVEIRA, 2011). As desvantagens dessa topologia são que o servidor pode ser um fator limitante, pois tem que processar cada mensagem enviada no ambiente, assim como, pode se tornar um ponto de falha. Além disso, o número de participantes aceitos em uma arquitetura cliente-servidor é inferior ao obtido em uma arquitetura peer-to-peer (JUNIOR, 2000 apud OLIVEIRA, 2011).

Permitindo a comunicação direta entre qualquer elemento da rede (DUNDES, 2008 apud OLIVEIRA, 2011)a topologia *peer-to-peer* elimina o problema do ponto de falha existente na topologia cliente-servidor. Nessa topologia, se a comunicação entre os clientes for *unicast,* cada participante deve estabelecer uma conexão com cada outro participante (DUNDES, 2008 apud OLIVEIRA, 2011)Uma alternativa para reduzir o número de conexões e de mensagens é o uso de *multicast* (JUNIOR, 2000 apud OLIVEIRA, 2011).

Ao unir a topologia cliente-servidor à topologia *peer-to-peer*, cria-se a topologia híbrida. Essa topologia contorna o problema do uso de um servidor, acrescentando uma grande escalabilidade ao sistema. Nessa topologia, o servidor central apenas mantém uma visão consistente do ambiente. A troca de mensagens entre os clientes é feita usando *multicast*. O servidor também recebe essas mensagens *multicast* para manter o ambiente atualizado (LEE et al., 2007 apud OLIVEIRA, 2011).

No artigo é identificado que, conforme o número de usuários aumenta, a quantidade de dados trafegando na rede também aumenta, o que pode gerar congestionamentos, que podem corromper dados, provocando atrasos na entrega dos mesmos, gerando inconsistência entre os usuários. Para contornar isso, existem esquemas de gerenciamento de interesse, divididos em abordagem baseada em proximidade, abordagem baseada em classe e em uma abordagem híbrida (GREENHALGH; BENFORD, 1997 apud OLIVEIRA, 2011) (SINGHAL; ZYDA,

1999 apud OLIVEIRA, 2011). O ambiente é dividido em regiões, assim, cada região pode ser gerenciada por um servidor, conforme o esquema de gerenciamento de interesse adotado.

Com o intuito de fornecer um balanceamento de carga entre os servidores primários de cada região usando o protocolo de comunicação *anycast* em nível de aplicação, o autor propôs a criação do *framework AnyAVDNet.* Nessa abordagem, a escolha do servidor é feita na camada de aplicação com base em alguns indicadores de desempenho (WU; HWANG; HO, s.d. apud OLIVEIRA, 2011). Na implementação do *AnyAVDNet,* um servidor *anycast* recebe as conexões dos novos usuários e os envia uma lista contendo os endereços de todos os servidores primários daquela região. Se não existir nenhum servidor primário, o primeiro cliente passa a ser o servidor primário dessa região. Após receber a lista com os endereços dos servidores, por meio da troca de mensagens com cada servidor primário, o cliente pode definir em qual se conectar, com base nas métricas recebidas.

Com a implementação do *AnyAVDNet,* Oliveira (2011) conseguiu atingir o objetivo de contornar as limitações de comunicação existentes no desenvolvimento de sistemas AVD, por meio do balanceamento de carga entre os servidores primários da arquitetura AVDNet usando o protocolo *anycast* em nível de aplicação.

## 3.3 Impactos do Uso de Endereçamento IP Anycast em Conexões TCP Persistentes

Em seu trabalho, Alcântara Filho e Rodrigues (2016), fazem uma análise das probabilidades de interrupções de conexões TCP persistentes considerando as atualizações das tabelas de roteamento em um ambiente que usa IP *Anycast*.

Como exemplos do uso de *anycasting* o artigo cita os servidores de raiz do *Domain Name System* (DNS) e provedores de redes de distribuição de conteúdo (CDN), identificando que esses serviços enfrentam desafios extras inerentes ao uso desse modo de endereçamento, como a localização do servidor mais próximo e a afinidade das conexões entre clientes e servidores, tendo em vista que cabe ao roteamento assegurar o caminho entre o cliente e servidor.

Entre as dificuldades que precisam tratamento especial, está a ausência de afinidade das conexões entre clientes e servidores, o que pode gerar a interrupção de conexões baseadas em estado, como as TCP, caso ocorra uma mudança no roteamento que leve à escolha de outro servidor, o que pode ser considerado como uma das características mais significativas. Além disso, é possível que as vantagens referentes à localidade das conexões, como a baixa latência, não sejam necessariamente obtidas sem o devido planejamento das operadoras de trânsito.

No trabalho, uma estimativa do impacto causado em conexões TCP foi realizada com base em dados reais da tabela de roteamento da Internet, coletada pelo projeto *Route Views*, onde foram selecionados cinco Sistemas Autônomos com pelo menos dois provedores de trânsito. Uma análise sobre as atualizações de rotas relativas aos prefixos desses ASs foi feita, identificando eventos de mudança de topologia (*AS-PATH*) que poderiam levar a uma quebra de conexão TCP, assumindo o pior caso, ou seja, qualquer mudança de AS-PATH incorreria em uma mudança de servidor.

O estudo identificou que em conexões TCP com tempo de vida próximos a 5000 segundos, o índice de quebra na conexão é considerado alto, cerca de 0,1%. Porém, conforme o tempo de vida das conexões é reduzido, esse percentual tende a cair expressivamente. No caso de conexões com tempo de vida inferior a 15 segundos, o índice de quebra na conexão é próximo a 0,02%. Assim, o autor conclui que a ausência de afinidade das conexões TCP no endereçamento IP *Anycast* pode ser um fator limitante no caso de aplicações que necessitam prolongados tempos de vida em suas conexões. No entanto, pode ser uma alternativa viável para aplicações onde não existe a exigência de um tempo de vida muito longo para as conexões.

## 3.4 Estudo comparativo

Com base nos artigos trabalhos relacionados apresentados anteriormente, o Quadro 1 apresenta um comparativo de características identificadas, comparando-as com a solução implementada neste trabalho.

**Quadro 1 - Estudo comparativo**

| **Características** | **Trabalho 1** (ALVIM; GROSSMANN; DANTAS, 2002) | **Trabalho 2** (OLIVEIRA, 2011) | **Trabalho 3** (ALCÂNTARA FILHO; RODRIGUES, 2016) | **Solução Implementada** |
|---|---|---|---|---|
| Utiliza balanceador de carga? | Sim | Sim | Não se aplica. | Não |
| Tira proveito da distribuição geográfica para reduzir a latência? | Não | Sim | Não se aplica. | Sim |
| Provê alta disponibilidade? | Sim | Sim | Não se aplica. | Sim |
| Demonstra vantagens com o uso da solução? | Sim | Sim | Sim | Sim |

Conforme pode ser visto no Quadro 1, nos artigos relacionados, existem diversas formas de tratar alta disponibilidade, escalabilidade e balanceamento de carga. Em seu artigo Alvim, Grossmann e Dantas (2002), propõe a adoção de Linux AD (Alta Disponibilidade), com o uso do *Heartbeat* para contornar o problema de falha no balanceador de cargas em *clusters* Linux. Essa solução não apresenta vantagens relacionadas à distribuição geográfica dos serviços, apenas em relação à alta disponibilidade e balanceamento de carga.

Já o artigo desenvolvido por Oliveira (2011) apresenta uma solução Anycast baseada na camada de aplicação para Ambientes Virtuais Distribuídos. Essa solução apresenta vantagens por tirar proveito da distribuição geográfica, distribuindo as cargas com base em métricas avaliadas pela solução proposta pelo autor. Essa solução, apesar de interessante, precisa ser desenvolvida na camada de aplicação, o que gera um nível de dificuldade maior.

Uma estimativa de falhas geradas por interrupções em conexões TCP persistentes é apresentada por Alcântara Filho e Rodrigues (2016). Essa estimativa utiliza dados reais, onde podemos concluir que, para serviços que utilizem conexões com tempos de vida inferiores a 15 segundos, o número de falhas, no pior cenário possível, será de cerca de 0,08%. Assim, sabe-se que o uso do protocolo IP *Anycast* deve ser implementado após estudos sobre os serviços que farão uso do mesmo.

A solução implementada neste trabalho consiste em analisar, por meio de testes, a funcionalidade do protocolo IP *Anycast*. Espera-se obter resultados que demonstrem a viabilidade do seu uso para balanceamento de carga e alta disponibilidade, assim como apresentar meios que tornem possível a sua implementação e operação.

## 4. Solução Avaliada

Com o objetivo de avaliar as vantagens da implementação do IP *Anycast* como meio para obter balanceamento de carga e ganhos em disponibilidade, foram realizados testes usando o *software* GNS3, onde uma série de roteadores Cisco 3745 foram organizados formando uma rede onde cada roteador faz parte de um Sistema Autônomo. Nessa rede, o protocolo BGP é responsável pela propagação de rotas entre os ASs.

Também foram usadas sessões BGP, nesse caso *multihop,* para a análise do protocolo IP *Anycast* com relação a alta disponibilidade. As sessões foram estabelecidas entre os dois servidores *anycast* e os hosts clientes, que utilizam o mesmo sistema operacional. Esse sistema foi escolhido pois possui diversas funcionalidades embarcadas, entre elas, o suporte ao protocolo BGP. Em um primeiro momento, pretendíamos utilizar outro serviço, como DNS ou HTTP com uma ferramenta capaz de gerar ataques DOS para simulação de carga, porém o ambiente de testes se mostrou instável. Nesse caso, as sessões BGP configuradas permitiram visualizar a capacidade de redirecionar os serviços em caso de falhas. Em um ambiente real, uma série de outros serviços podem tirar proveito disso, principalmente os que tem conexões com baixo tempo de vida.

O balanceamento de carga foi analisado com base no processamento dos servidores *anycast* e volume de pacotes recebido. Para gerar carga, foram utilizados pacotes ICMP (*Internet Control Message Protocol*) originados a partir dos *hosts* clientes. Em um cenário real, o pequeno volume de pacotes ICMP não seria suficiente para gerar uma carga considerável, porém, para fins de pesquisa em um ambiente de testes, esse volume foi suficiente. Com o objetivo de redistribuir as cargas, as políticas de roteamento do protocolo BGP foram alteradas no servidor Anycast-SRV-02, fazendo com que parte do tráfego fosse migrada para o servidor Anycast-SRV-01.

A avaliação de desempenho teve como base a análise dos tempos de resposta dos pacotes ICMP enviados por um dos nós cliente. Nesse caso, observou-se que, no momento em que o servidor *anycast* mais próximo ao nó estava disponível, o tempo de

resposta era cerca de três vezes inferior ao momento em que esse nó estava indisponível. Esse dado indica que a solução apresenta vantagens por tirar proveito dos protocolos de roteamento para encontrar o servidor mais próximo.

Os testes conduzidos apresentaram resultados que indicam que a solução analisada se apresenta como uma alternativa simples e eficiente na implementação de serviços que precisam de alta disponibilidade, balanceamento de carga e desempenho. Essas características são possíveis pois a escolha do servidor que atenderá o cliente é feita pelo protocolo de roteamento, que encontra o servidor mais próximo com base em sua tabela de rotas.

## 4.1 Ambiente de testes

Nessa seção são abordados os aspectos relacionados ao ambiente de testes, o que envolve os *softwares* utilizados, configurações de endereçamento IP, topologia de rede do ambiente, protocolos de roteamento, ferramentas de testes e coleta de dados.

O ambiente foi composto por um total de 6 roteadores Cisco 3745, três *hosts NETem*, que simulam latência similar a enlaces de longa distância e cinco dispositivos rodando o sistema operacional *RouterOS*. Parte dos testes também foram realizados utilizando 3 computadores com sistema operacional Ubuntu. Esses equipamentos foram virtualizados por um *software* chamado GNS3 (https://www.gns3.com/). O GNS3 utiliza sistemas de virtualização como o QEMU, *Dynamips* e o *VirtualBox*. Dentro desse ambiente, também foi adicionado um *switch* e uma nuvem, chamada Gerencia. Esses últimos dispositivos fazem parte do GNS3 e tem como funcionalidade permitir que exista uma conexão de ponte do computador hospedeiro com os *hosts* virtualizados. Neste caso, essa conexão de ponte permitirá que os dispositivos sejam gerenciados por outro computador, evitando que a performance seja prejudicada durante os testes. Também é possível gerar gráficos por meio de uma *Appliance Zabbix* que foi virtualizada em outro computador, utilizando o *VirtualBox*.

O GNS3 é um *software* de código aberto, desenvolvido e suportado por uma comunidade de mais de 800 mil usuários e já teve mais de 10 milhões de *downloads*. É utilizado em companhias ao redor do mundo para virtualizar dispositivos reais, permitindo testar e verificar implantações no mundo real. Ao longo de seus dez anos de existência, o GNS3, que originalmente suportava apenas dispositivos Cisco emulados por um *software* chamado *Dynamips*, evoluiu para suportar diversos fabricantes de equipamentos de redes e também a integração com outros sistemas de virtualização, como *VirtualBox* e *VMWare*. Esses atributos foram considerados ao escolher o GNS3 como *software* para a condução dos testes, pois o mesmo permite obter resultados similares aos obtidos com a utilização de equipamentos e ambientes reais, resguardadas as limitações impostas pelo *hardware* onde o *software* está instalado (BOMBAL; DUPONCHELLE, 2017).

Os gráficos gerados por meio da *Appliance Zabbix* criaram a possibilidade de entender o comportamento do ambiente de forma visual. O *Zabbix* é um *software* de código aberto, desenvolvido para monitorar a disponibilidade e o desempenho de componentes de infra-estrutura de TI. Ele possui recursos de coleta e armazenamento em tempo real de dados de dispositivos de rede, recursos para visualização dos dados armazenados, tais como gráficos que podem ser personalizados e recursos de análise de

dados para fins de envio de alertas. Outra vantagem do *Zabbix* em relação a outros sistemas para monitoramento é o fato de que é possível efetuar o *download* de uma máquina virtual (*Appliance*) pronta para uso, reduzindo o tempo gasto com itens como instalação de Sistemas Operacionais, pacotes e configuração de banco de dados. A facilidade de implantação e a flexibilidade da ferramenta tornam o *Zabbix* uma das melhores alternativas de código aberto para monitoramento em ambientes de TI. (ZABBIX LLC, 2017)

Os 6 roteadores Cisco 3745 fazem parte de ASs diferentes, representados pelos números 100, 200, 300, 400, 500 e 600, enquanto dois dos dispositivos *RouterOS* fazem parte do AS 1000. Para interligar esses roteadores, foram usados endereços IP da rede 10.0.0.0/16. Cada Sistema Autônomo recebeu um prefixo "/16", que é propagado por meio do protocolo BGP, conforme descrito na Tabela 1.

**Tabela 1 – IPs alocados para cada AS do ambiente de testes**

| Número do AS | Prefixo |
|---|---|
| 100 | 10.101.0.0/16 |
| 200 | 10.102.0.0/16 |
| 300 | 10.103.0.0/16 |
| 400 | 10.104.0.0/16 |
| 500 | 10.105.0.0/16 |
| 600 | 10.106.0.0/16 |
| 1000 | 10.110.0.0/16 |

Fonte: Dos autores, 2017

A configuração dos roteadores é bastante similar, onde cada roteador se conecta com dois Sistemas Autônomos diferentes, e propaga todas as rotas conhecidas. Nesse caso, as políticas individuas que cada AS poderia implementar não são relevantes, com exceção do AS 1000, que implementa políticas com o objetivo de manipular e balancear o tráfego destinado os dispositivos *RouterOS*. A configuração básica do roteador AS-100 pode ser vista na Figura 1, omitindo o que não for relevante para o funcionamento dos testes.

```
AS-100#show running-config
hostname AS-100
interface FastEthernet0/0
 description to-Internet
 ip address dhcp
 duplex auto
 speed auto
interface FastEthernet1/0
 description to-Anycast-SRV-01
 ip address 10.0.11.1 255.255.255.0
 duplex auto
 speed auto
interface FastEthernet3/0
 description to-AS200
 no switchport
 ip address 10.0.100.1 255.255.255.0
interface FastEthernet4/0
 no ip address
 shutdown
 duplex auto
 speed auto
interface Vlan1
 no ip address
router bgp 100
 no synchronization
 bgp log-neighbor-changes
 network 10.101.0.0 mask 255.255.0.0
 neighbor 10.0.11.2 remote-as 1000
 neighbor 10.0.100.2 remote-as 200
 no auto-summary
ip route 10.101.0.0 255.255.0.0 Null0
snmp-server community anycast RO
end
```

**Figura 1 – Configuração básica do roteador AS-100 utilizado no ambiente de testes**
**Fonte: Dos atores, 2017**

As configurações apresentadas na Figura 1 permitem que o roteador estabeleça sessões BGP com os roteadores dos outros ASs aos quais está conectado. Também permitem que o roteador seja monitorado por meio do protocolo SNMP, o que permitirá que a *appliance* Zabbix colete dados de tráfego para geração de gráficos. Essa *appliance* está disponível para *download* em https://www.zabbix.com/download e foi virtualizada por meio do *VirtualBox*. A topologia de rede construída pode ser vista na Figura 2.

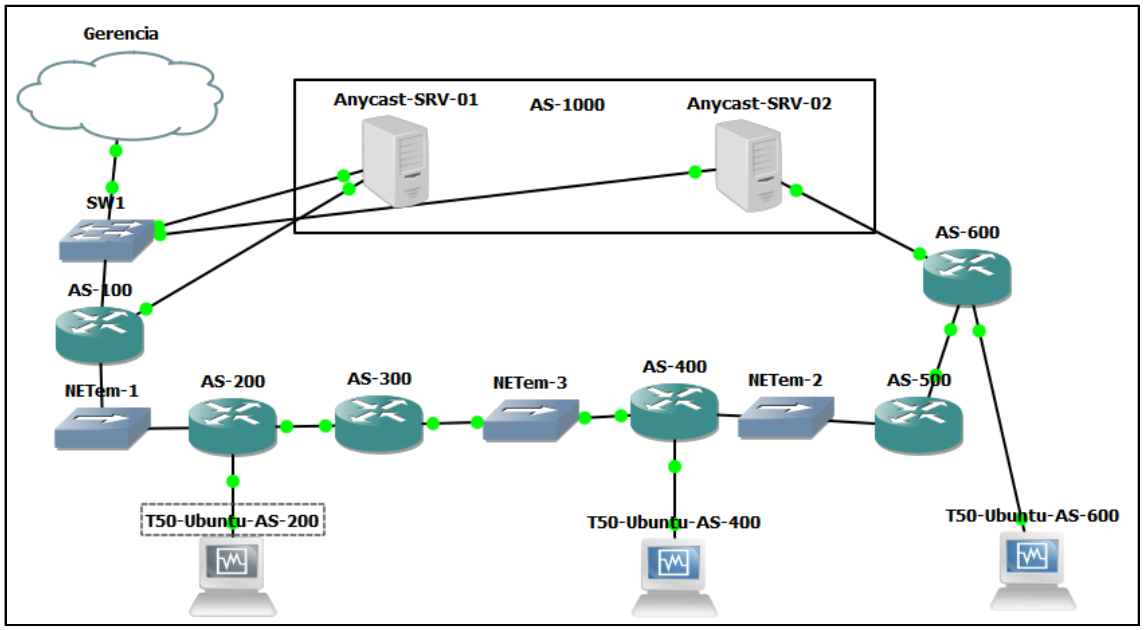


**Figura 2 – Ambiente de testes no software GNS3. Fonte: Dos autores, 2017**

Após todas as sessões BGP serem estabelecidas, quando analisamos a tabela de rotas do roteador AS-300, na Figura 3, podemos ver que o mesmo possui duas rotas para o prefixo 10.110.0.0/16, preferindo a rota que possui o atributo AS-PATH menor. Esse comportamento também ocorre no roteador AS 400, indicando que ambos os roteadores já possuem rotas redundantes para o prefixo *anycast* do AS-1000.

```
AS-300#show ip bgp 10.110.0.1
BGP routing table entry for 10.110.0.0/16, version 21
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  200 100 1000
    10.0.200.1 from 10.0.200.1 (10.102.100.1)
      Origin IGP, localpref 100, valid, external, best
  400 500 600 1000
    10.0.30.2 from 10.0.30.2 (10.0.40.1)
      Origin IGP, localpref 100, valid, external
```

**Figura 3 – Rotas do servidor AS 300 no ambiente de testes. Fonte: Dos autores, 2017**

Para gerar carga no sistema, nos testes de balanceamento de carga, 3 dispositivos com sistema operacional Linux com a ferramenta T50 foram escolhidos, porém, o ambiente de testes se mostrou instável nessa situação. Para que os testes pudessem ser concluídos, os *hosts* Ubuntu foram substituídos por dispositivos com sistema operacional RouterOS. Nesse caso, as avaliações foram feitas analisando o volume de pacotes recebido, o consumo de CPU de cada servidor *Anycast* e suas variações quando as políticas de roteamento são atualizadas.

## 4.3 Metodologia de testes e validação

Os testes realizados tiveram, como objetivo, o de analisar a eficiência do protocolo IP *Anycast* como solução para alta disponibilidade e balanceamento de carga. Nesse sentido, é importante que mesmo nos casos em que um dos servidores *Anycast* esteja inoperante ou inalcançável, os clientes continuem a ter acesso ao serviço, que deve ser redirecionado para o servidor *Anycast* disponível. O consumo de recursos dos servidores, como CPU e rede também foi considerado, servindo como métrica para a eficiência do balanceamento baseado em *Anycast*. Por último, a latência do cliente até o servidor escolhido pelo roteamento serviu como métrica para identificar ganhos em performance associados ao uso do *Anycast*.

A ferramenta Zabbix foi usada para gerar os gráficos de tráfego e de uso de CPU dos servidores *Anycast*, enquanto os testes de latência foram feitos a partir dos dispositivos RouterOS chamados T50-Ubuntu-AS-200, T50-Ubuntu-AS-400 e T50-Ubuntu-AS-600. A partir das informações coletadas, foi possível visualizar as alterações na carga de cada servidor ao longo dos testes, assim como as variações de latência ocasionadas pelas simulações conduzidas.

A avaliação da disponibilidade do serviço foi feita a partir de sessões BGP *multihop*. Esse tipo de sessão é usado para divulgação de prefixos, como, por exemplo, o projeto *Team Cymru*, que utiliza sessões desse tipo para divulgar uma lista de prefixos IP que devem ser filtrados, como prefixos que ainda não foram designados para nenhum AS ou endereços IP não roteáveis, permitindo a criação de filtros automáticos. Nesse artigo, o único objetivo foi analisar a migração dessas sessões após a queda de um dos servidores *anycast*.

Para promover alterações nas tabelas de rotas, o que consequentemente deve influenciar no balanceamento de carga, as políticas de roteamento dos servidores *Anycast* foram alteradas, modificando o atributo *AS-PATH*, por meio da adição de um caminho mais longo. Essa prática, comum dentro de um Sistema Autônomo, é utilizada para balancear os *links* de trânsito.

A metodologia empregada consistiu em avaliar, ao longo de três etapas, as funcionalidades de tolerância a falhas, balanceamento de cargas e aumento de desempenho proporcionado pelo uso do protocolo IP *Anycast*. A característica de tolerância a falhas foi avaliada por meio do desligamento de um dos servidores, simulando uma indisponibilidade. Os ganhos em desempenho foram quantificados por meio da comparação do tempo de resposta (latência) medido no cenário em que dois servidores *anycast* estão disponíveis e no cenário em que apenas um servidor *anycast* está disponível, de forma a simular um aumento na distância geográfica. Foram coletados dados por meio do *Zabbix* para análise do balanceamento de carga, enquanto alterações nas tabelas de roteamento foram feitas utilizando o atributo *AS-PATH* do protocolo BGP durante o período em que os clientes estavam gerando carga através do envio de pacotes ICMP para os servidores. As descrições detalhadas das simulações conduzidas e suas conclusões encontram-se na próxima seção.

## 4.4 Testes e validação

Os testes iniciaram-se com todos os equipamentos do ambiente inicializados, sem nenhum tipo de consumo, exceto o necessário para gerenciar os dispositivos. A Figura 4

apresenta os dados de consumo de recursos dos dois servidores *Anycast,* chamados Anycast-SRV-01 e Anycast-SRV-02.

```
[admin@Anycast-SRV-01] > system resource print
                   uptime: 23m15s
                  version: 6.40.5 (stable)
               build-time: Oct/31/2017 13:05:15
             free-memory: 201.7MiB
            total-memory: 221.9MiB
                      cpu: QEMU
                cpu-count: 1
            cpu-frequency: 2494MHz
                 cpu-load: 12%
           free-hdd-space: 69.8MiB
          total-hdd-space: 95.3MiB
  write-sect-since-reboot: 984
         write-sect-total: 985
        architecture-name: x86_64
               board-name: CHR
                 platform: MikroTik
```

```
[admin@Anycast-SRV-02] > system resource print
                   uptime: 24m48s
                  version: 6.40.5 (stable)
               build-time: Oct/31/2017 13:05:15
             free-memory: 201.8MiB
            total-memory: 221.9MiB
                      cpu: QEMU
                cpu-count: 1
            cpu-frequency: 2494MHz
                 cpu-load: 9%
           free-hdd-space: 69.8MiB
          total-hdd-space: 95.3MiB
  write-sect-since-reboot: 1664
         write-sect-total: 1665
        architecture-name: x86_64
               board-name: CHR
                 platform: MikroTik
```

**Figura 4 – Servidores *Anycast* sem carga Fonte: dos autores, 2017**

Os primeiros testes visaram avaliar a funcionalidade do ambiente utilizando protocolo IP *Anycast* e os ganhos de performance relacionados com a redução da latência. Isso foi feito utilizando testes básicos de conectividade a partir dos *hosts* Linux, o que permite analisar as variações de latência e de rotas.

Na Figura 5, que mostra as tabelas de rotas dos roteadores AS-200, AS-400 e AS-600, é possível visualizar que os pacotes do *host* T50-Ubuntu-AS-200 estão sendo enviados pelo protocolo de roteamento para o servidor chamado Anycast-SRV-01. Nessa mesma imagem, é possível ver que os pacotes dos hosts T50-Ubuntu-AS-400 e T50-Ubuntu-AS-600 estão sendo enviados para o servidor Anycast-SRV-02.

```
AS-200#show ip bgp 10.110.0.1
BGP routing table entry for 10.110.0.0/16, version 5
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  100 1000
    10.0.100.1 from 10.0.100.1 (10.0.100.1)
      Origin IGP, localpref 100, valid, external, best
```

```
AS-400#show ip bgp 10.110.0.1
BGP routing table entry for 10.110.0.0/16, version 7
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  300 200 100 1000
    10.0.30.1 from 10.0.30.1 (10.0.200.2)
      Origin IGP, localpref 100, valid, external
  500 600 1000
    10.0.40.2 from 10.0.40.2 (10.0.50.1)
      Origin IGP, localpref 100, valid, external, best
```

```
AS-600#show ip bgp 10.110.0.1
BGP routing table entry for 10.110.0.0/16, version 4
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  1000
    10.0.12.2 from 10.0.12.2 (10.110.0.1)
      Origin IGP, localpref 100, valid, external, best
```

**Figura 5 – Tabelas de rotas de parte do ambiente de testes. Fonte: Dos autores, 2017**

Nesse cenário, identificou-se que a latência do dispositivo T50-Ubuntu-AS-200 até o servidor *Anycast* 10.100.0.1 era em média 41 milissegundos, enquanto que do *host* T50-Ubuntu-AS-600 a latência média foi de 27 milissegundos. Já a latência do dispositivo T50-Ubuntu-AS-400 obteve uma média de 63 milissegundos. Essa variação de latência se justifica pois o dispositivo T50-Ubuntu-AS-600 está mais próximo ao servidor Anycast-SRV-02, tendo apenas um roteador em seu caminho, enquanto os outros dispositivos têm um maior número de saltos para alcançar o destino.

Para investigar o comportamento da rede, uma política de roteamento foi aplicada no servidor Anycast-SRV-02, com o objetivo de desviar o tráfego para o servidor Anycast-SRV-01. Essa política consiste em alterar o *AS-PATH* adicionando saltos, de forma que as rotas divulgadas sejam recalculadas pelos roteadores que as recebem. A Figura 6 mostra a configuração efetuada no dispositivo Anycast-SRV-02 e as tabelas de rotas dos roteadores AS-400 e AS-600.

```
[admin@Anycast-SRV-02] > routing filter add chain=out_as_600 action=accept set-bgp-prepend=6
```

```
AS-600#show ip bgp 10.110.0.1
BGP routing table entry for 10.110.0.0/16, version 21
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
  Advertised to update-groups:
     1
  500 400 300 200 100 1000
    10.0.50.1 from 10.0.50.1 (10.0.50.1)
      Origin IGP, localpref 100, valid, external
  1000 1000 1000 1000 1000 1000
    10.0.12.2 from 10.0.12.2 (10.110.0.1)
      Origin IGP, localpref 100, valid, external, best
```

```
AS-400#show ip bgp 10.110.0.1
BGP routing table entry for 10.110.0.0/16, version 23
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  300 200 100 1000
    10.0.30.1 from 10.0.30.1 (10.0.200.2)
      Origin IGP, localpref 100, valid, external, best
```

**Figura 6 – Alteração nas políticas de roteamento do host Anycast-SRV-02 e tabelas de roteamento de AS-600 e AS-400. Fonte: dos autores, 2017**

A partir dessa informação, é possível perceber que agora o AS-400 está preferindo o servidor Anycast-SRV-01, passando pelos ASs 300, 200 e 100 para chegar até o destino. Entre os roteadores AS-400 e AS300 existe o dispositivo NETem-3, que irá adicionar latência, simulando a distância geográfica. Agora, efetuando novos testes, identificamos que a latência de T50-Ubuntu-AS-600 e de T50-Ubuntu-AS-200 até o servidor 10.110.0.1 não sofreu alterações, porém, no caso do *host* T50-Ubuntu-AS-400 a latência média subiu para 109 milissegundos. Isso ocorre porque agora esse *host* precisa acessar um servidor que está geograficamente mais distante.

Após validar o funcionamento do ambiente, foram feitos testes que analisaram a distribuição de carga. Esses testes foram realizados a partir de *hosts* RouterOS, inseridos onde originalmente estavam os dispositivos T50-Ubuntu-AS-200, T50-Ubuntu-AS-400 e T50-Ubuntu-AS-600. Para fins de documentação, os nomes foram mantidos. Em um primeiro momento, a ferramenta T50 foi considerada para esses testes, porém, os recursos de hardware limitados do ambiente de testes impediram seu uso. O uso da ferramenta HPING3 também foi descartado devido à capacidade de processamento do ambiente. Foram gerados gráficos em um servidor *Zabbix* para análise do comportamento da rede. Antes de iniciar os testes, as políticas de roteamento aplicadas no servidor Anycast-SRV-02 foram removidas, voltando as tabelas de roteamento ao estado da Figura 5.

Nos *hosts* T50-Ubuntu-AS-200, T50-Ubuntu-AS-400 e T50-Ubuntu-AS-600, foram inicializados testes ICMP com o comando “ping 10.110.0.1 interval=100ms”. Nessa configuração, o volume de pacotes enviados foi de 10 pacotes por segundo a partir de cada dispositivo RouterOS, o que já foi suficiente para aumentar o uso de CPU nos servidores Anycast-SRV-01 e Anycast-SRV-02. O aumento de CPU se justifica por ser um ambiente de testes virtualizado. Após alguns minutos, a política de roteamento voltou a ser aplicada no servidor Anycast-SRV-02. A Figura 7 exibe um gráfico de consumo de recursos durante o período de testes.

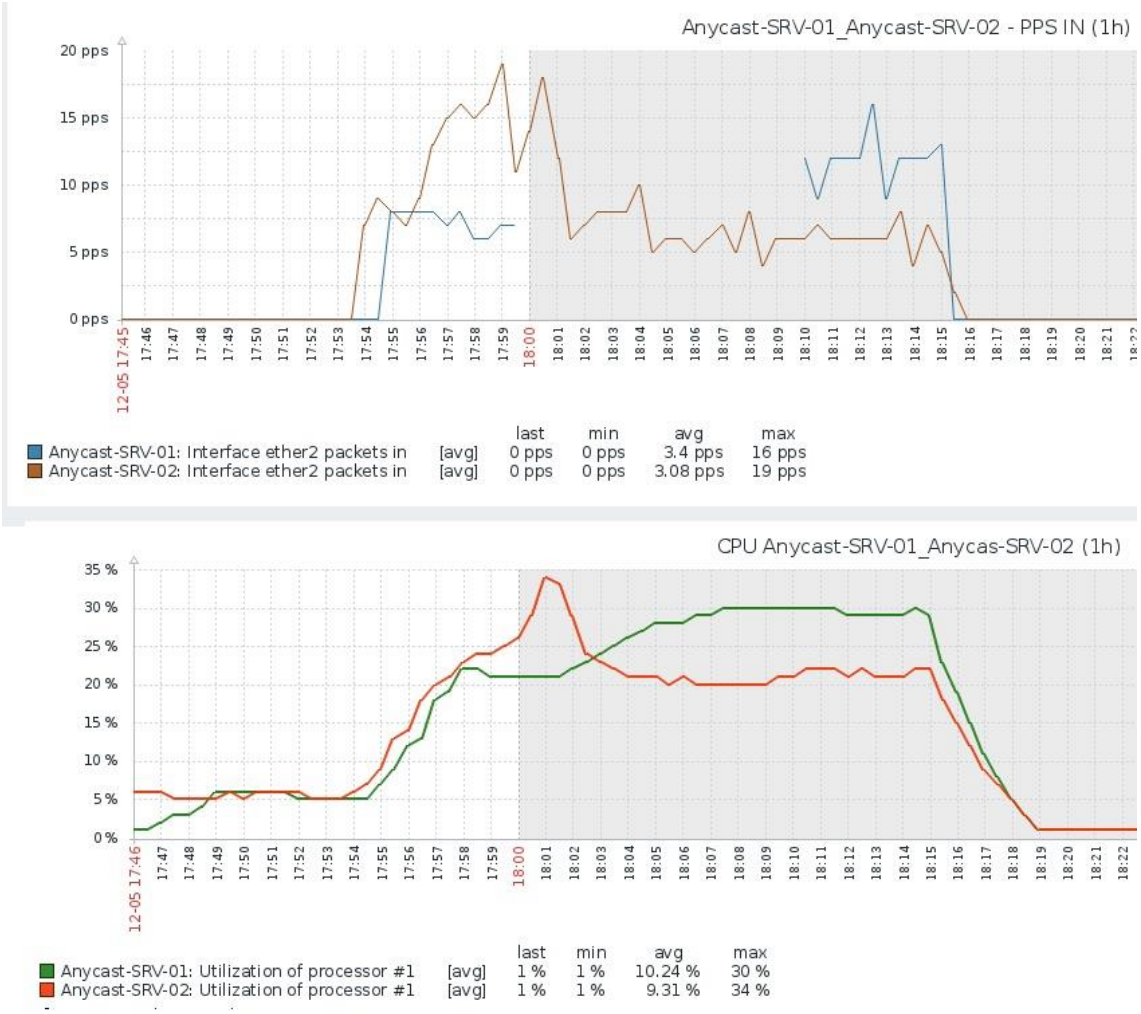


**Figura 7 - Consumo de recursos nos servidores Fonte: (dos autores, 2017)**

Na Figura 7, podemos visualizar o momento em que a política de roteamento voltou a ser aplicada no servidor Anycast-SRV-02, fazendo com que parte do tráfego fosse redirecionado para o servidor Anycast-SRV-01. Com a redução no número de pacotes processados pelo servidor Anycast-SRV-02, o consumo de CPU cai proporcionalmente. Consequentemente, o consumo de CPU em Anycast-SRV-01 aumenta.

Por se tratar de um ambiente de testes pequeno e com recursos de *hardware* limitados, não é possível fazer testes com um número maior de dispositivos que se comportem como clientes. Assim, não conseguimos obter vários níveis de balanceamento de carga, pois as alterações que são feitas refletem apenas em 3 dispositivos consumidores de recursos. Em ambientes maiores, é possível obter um balanceamento de carga mais homogêneo.

Por fim, os *hosts* RouterOS T50-Ubuntu-AS-200, T50-Ubuntu-AS-400 e T50-Ubuntu-AS-600 foram utilizados para validar o funcionamento do protocolo IP *Anycast* em relação à alta disponibilidade. Foram configuradas sessões BGP *multihop* nesses hosts, que podem ser estabelecidas tanto com o servidor Anycast-SRV-01 quanto com o

servidor Anycast-SRV-02. Para facilitar a análise dos resultados obtidos do ponto de vista do cliente, cada servidor *Anycast* recebeu um *Router* ID diferente nas configurações do BGP.

Antes de iniciar este teste, a política de roteamento existente em Anycast-SRV-02 foi removida, assim, as tabelas de roteamento voltaram ao estado da Figura 5. A Figura 8 exibe as telas dos servidores *Anycast* onde é possível encontrar a informação sobre as sessões BGP estabelecidas. Após a constatação de que todas as sessões BGP *Multihop* foram estabelecidas, o servidor Anycast-SRB-02 foi desligado, simulando uma indisponibilidade.

```
[admin@Anycast-SRV-01] > routing bgp peer print
Flags: X - disabled, E - established
 #    INSTANCE          REMOTE-ADDRESS                                REMOTE-AS
 0 E  default           10.0.11.1                                     100
 1 E  default           10.102.100.2                                  102
 2    default           10.104.100.2                                  104
 3    default           10.106.100.2                                  106
[admin@Anycast-SRV-02] > routing bgp peer print
Flags: X - disabled, E - established
 #    INSTANCE          REMOTE-ADDRESS                                REMOTE-AS
 0 E  default           10.0.12.1                                     600
 1    default           10.102.100.2                                  102
 2 E  default           10.104.100.2                                  104
 3 E  default           10.106.100.2                                  106
```

**Figura 8 - Sessões BGP estabelecidas Fonte: (dos autores, 2017)**

A Figura 9 apresenta a tela do *host* T50-Ubuntu-AS-600, onde é possível visualizar as perdas de pacote geradas pela indisponibilidade do servidor *Anycast*. Nessa mesma figura, podemos ver que todas as sessões BGP foram estabelecidas no servidor Anycast-SRV-01.

```
[admin@T50-Ubuntu-AS-600] > ping 10.110.0.1
  SEQ HOST                                     SIZE TTL TIME  STATUS
    0 10.110.0.1                                 56  63 27ms
    1 10.110.0.1                                 56  63 31ms
    2 10.110.0.1                                 56  63 58ms
    3 10.110.0.1                                 56  63 60ms
    4 10.110.0.1                                 56  63 38ms
    5 10.110.0.1                                             timeout
    6 10.110.0.1                                             timeout
    7 10.110.0.1                                             timeout
    8 10.110.0.1                                             timeout
    9 10.110.0.1                                             timeout
   10 10.110.0.1                                 56  58 148ms
   11 10.110.0.1                                 56  58 101ms
   12 10.110.0.1                                 56  58 98ms
   13 10.110.0.1                                 56  58 96ms
[admin@Anycast-SRV-01] > routing bgp peer print
Flags: X - disabled, E - established
 #    INSTANCE                    REMOTE-ADDRESS                                REMOTE-AS
 0 E  default                     10.0.11.1                                     100
 1 E  default                     10.102.100.2                                  102
 2 E  default                     10.104.100.2                                  104
 3 E  default                     10.106.100.2                                  106
```

**Figura 9 - Servidor Anycast-SRV-01 estabelece todas as sessões BGP após a indisponibilidade do servidor Anycast-SRV-02 Fonte: (dos autores, 2017)**

Assim, é possível afirmar que o protocolo IP *Anycast* se apresenta como uma solução viável na implementação de alta disponibilidade, pois permite que as conexões destinadas a um servidor inoperante sejam automaticamente redirecionadas para outro

servidor que esteja em operação. Trata-se também, de uma solução simples e eficiente para balanceamento de carga, distribuindo a carga entre os servidores disponíveis e permitindo que sejam feitas intervenções para aprimorar essa distribuição caso seja necessário.

A manipulação de rotas, possível pelo uso de protocolos de roteamento dinâmicos, permite que sejam feitos ajustes no balanceamento de carga entre os servidores, o que pode ser feito por interação humana ou, podem ser desenvolvidos *scripts* que façam intervenções em situações pré-estabelecidas.

## 5. Considerações finais

Este artigo apresentou uma análise da implementação do protocolo IP *Anycast* como meio para obter balanceamento de carga e alta disponibilidade em sistemas geograficamente distribuídos. Falhas em redes *backbone* ou nos próprios servidores podem gerar indisponibilidade ou perda de performance, o que pode ser mitigado com a distribuição geográfica dos servidores, deixando-os mais próximos aos usuários. Além disso, a escalabilidade dos serviços se torna mais simples, uma vez que é possível adicionar servidores para balancear a carga próximos ao local de consumo.

Por meio dos testes realizados em um ambiente de testes com o uso da ferramenta GNS3, foram demonstradas formas de realizar o balanceamento de carga e realizadas simulações de indisponibilidades. O ganho de performance obtido pela distribuição geográfica dos servidores pôde ser analisado com a queda do servidor Anycast-SRV-02, na Figura 9. Nessa figura, é possível ver que a latência a partir do dispositivo T50-Ubuntu-AS-600 chegou a ser três vezes maior quando o servidor mais próximo ficou indisponível.

Além de ser possível identificar uma melhora de desempenho, a implantação do protocolo IP *Anycast* mostrou-se bastante simples. Boa parte dessa simplicidade vem dos protocolos de roteamento, que já fazem parte de um ambiente como a Internet. Assim, não existe a necessidade de desenvolver protocolos ou aplicações específicas. A implantação pode ser feita utilizando equipamentos que já possuem suporte a protocolos de roteamento dinâmicos ou da instalação de *softwares* como o *Quagga*, que permite que um sistema operacional Linux passe a suportar esses protocolos. (QUAGGA ROUTING SUITE, 2017) A simplicidade e flexibilidade são características a serem consideradas na escolha do IP *Anycast* como uma solução para tornar serviços tolerantes a falhas e aumentar o desempenho.

A escolha do GNS3 como sistema para virtualizar o ambiente de teste mostrou-se acertada por permitir que fossem conduzidos testes similares ao uso de equipamentos reais. Porém, o desempenho ficou aquém do esperado. As limitações da ferramenta em relação ao desempenho não permitiram que fossem feitos testes mais elaborados, como a utilização de ferramentas que simulam ataques *DOS* ou a utilização de serviços como *DNS* nos servidores *anycast,* o que não reduz a precisão dos testes, apenas abre caminho para que novos trabalhos sejam realizados no futuro. É importante destacar que sem este *software*, o ambiente de testes precisaria uma grande quantidade de equipamentos, como computadores ou roteadores, cabos e energia.

Durante o desenvolvimento desse trabalho, foram enfrentadas dificuldades relacionadas com o ambiente de testes e coleta de informações de desempenho. Parte

dessas dificuldades podem ser justificadas devido a limitações de *hardware*, o que reduziu as simulações possíveis no ambiente de testes. Além disso, durante as simulações, não foi possível estressar ao máximo os dispositivos, pois, quando isso foi feito, a coleta de dados como tráfego nas interfaces dos servidores *anycast* apresentou falhas. Essas dificuldades foram contornadas ao reduzir o estresse dos servidores *anycast,* com o envio de um volume menor de pacotes, permitindo assim que fossem coletados dados para a geração de gráficos.

Com as informações e testes realizados neste artigo, foi possível verificar que existem vários benefícios do uso do protocolo IP *Anycast* que podem ser explorados, como sua facilidade de implementação e operação. No futuro podem ser realizados outros estudos relacionados à automação do balanceamento de carga, tais como a inserção de mais nós do tipo servidor e um número maior de serviços podem ser testados, além de outros protocolos de roteamento.

## Referências


ALCÂNTARA FILHO, R. P. D; RODRIGUES, W. **Impactos do Uso de Endereçamento IP Anycast em Conexões TCP Persistentes.** 2016. Disponível em: <https://www.iecom.org.br/encom2016/ENCOM_files/Encom2016_Artigos/14.pdf>. Acesso em: 03 dez. 2017.

ALVIM, R. M.; GROSSMANN, F. L. L.; DANTAS, M. A. R. **IMPLEMENTAÇÃO DE UMA ARQUITETURA PARA SERVIÇO DISTRIBUÍDO COM GRANDE DISPONIBILIDADE EM AMBIENTE LINUX.** 2002. 12 f. TCC (Graduação) - Curso de Ciência da Computação, Departamento de Ciência da Computação, Universidade de Brasília, Brasília, 2002. Disponível em: <https://www.researchgate.net/profile/Mario_Dantas2/publication/268329651_IMPLEMENTACAO_DE_UMA_ARQUITETURA_PARA_SERVICO_DISTRIBUIDO_COM_GRANDE_DISPONIBILIDADE_EM_AMBIENTE_LINUX/links/55643b8108ae86c06b6977a0.pdf>. Acesso em: 07 dez. 2017.

AMARAL, B. **Furacão Maria impacta infraestrutura de cabo submarino e afeta Internet no Brasil.** 2017. Disponível em: <http://teletime.com.br/2017/09/21/furacao-maria-impacta-infraestrutura-de-cabo-submarino-e-afeta-internet-no-brasil/>. Acesso em: 07 dez. 2017.

BOMBAL, D.; DUPONCHELLE, J. **Getting Started with GNS3.** 2017. Disponível em: <https://docs.gns3.com/1PvtRW5eAb8RJZ11maEYD9_aLY8kkdhgaMB0wPCz8a38/index.html>. Acesso em: 20 dez. 2017.

COMER, D. E. **Redes de computadores e internet.** 4. ed. Porto Alegre: Bookman, 2007. 640 p.

CORREIA, R. C. M. **AVDNet-Arquitetura para Ambientes Virtuais Distribuídos Escaláveis Baseada na Infra-Estrutura Atual da Internet**. Tese (Ciência da

Computação) — Instituto Tecnológico de Aeronáutica, São José dos Campos, SP., 2005.

DUNDES, L. W. M. Graduação em Ciência da Computação, **AVDNet6 - Framework de gerenciamento de aplicações distribuídas baseadas no protocolo IPv6**. Presidente Prudente: [s.n.], 2008. 61 f.

GREENHALGH, G.; BENFORD, S. Boundaries, awareness and interaction in collaborative virtual environments. In: **Proceedings of the Sixth Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises**. [S.l.: s.n.], 1997. p. 193–198.

JUNIOR, A. J. M. L. ATAXIA: **Uma arquitetura para viabilizaçcão de NVÉs voltados para a Educação a Distância através da Internet**. Dissertação (Ciência da Computação) — Universidade Federal do Ceará, Fortaleza, 2000.

LEE, D. et al. **ATLAS A Scalable Network Framework for Distributed Virtual Environments.** Korea, 2007.

OLIVEIRA, R. S. A. **Utilização do protocolo anycast em ambientes virtuais distribuídos visando à escalabilidade do sistema.** 2011. 38 f. TCC (Graduação) - Curso de Ciência da Computação, Departamento de Matemática, Estatística e Computação, Universidade Estadual Paulista, Presidente Prudente, 2011. Disponível em: <https://repositorio.unesp.br/bitstream/handle/11449/120316/oliveira_rsa_tcc_prud.pdf?sequence=1&isAllowed=y>. Acesso em: 07 dez. 2017.

PARTRIDGE, C.; MENDEZ, T.; MILLIKEN, W.. **Host Anycasting Service.** 1993. Disponível em: <https://tools.ietf.org/html/rfc1546>. Acesso em: 11 set. 2017.

QUAGGA ROUTING SUITE. **Quagga Routing Suite:** About Quagga. Disponível em: <http://www.nongnu.org/quagga/>. Acesso em: 22 nov. 2017.

SINGHAL, S.; ZYDA, M. **Networked Virtual Environments**. [S.l.]:Addison-Wesley, 1999.

SOARES, L. F. G.; LEMOS, G.; COLCHER, S.. **Redes de computadores:** das LANs, MANs, e WANs às redes ATM. 2. ed. Rio de Janeiro: Editora Campus, 1995. 705 p.

TANENBAUM, Andrew S.. **REDES DE COMPUTADORES.** 4. ed. Rio de Janeiro: Elsevier, 2003. 945 p.

TANENBAUM, A .S.; VAN STEEN, M. **Sistemas Distribuídos:** Principios e Paradigmas. 2. ed. São Paulo, Sp: Pearson Prentice Hall, 2007. 416 p.

TANENBAUM, A. S.; WETHERALL, D. **Redes de computadores.** 5. ed. São Paulo: Pearson Prentice Hall, 2011. 582 p.

WU, C.; HWANG, R.; HO, J. **A Scalable Overlay Framework for Internet Anycasting Service**. Taiwan, s.d.

ZABBIX LLC. **What is Zabbix.** 2017. Disponível em: <https://www.zabbix.com/product>. Acesso em: 22 nov. 2017.