Maicky Müller

Implantação de alta disponibilidade em PABX IP

São José - SC

Fevereiro/2017

Maicky Müller

Implantação de alta disponibilidade em PABX IP

Monografia apresentada à Coordenação do curso de tecnologia em Sistemas de Telecomunicações do Instituto Federal de Santa Catarina para a obtenção do diploma tecnólogo em Telecomunicações.

Instituto Federal de Santa Catarina – IFSC

Campus São José

Sistemas de Telecomunicações

Orientador: Prof. Msc. Ederson Torresini

São José - SC

Fevereiro/2017

Maicky Müller
Implantação de alta disponibilidade em PABX IP/ Maicky Müller. – São José - SC, Fevereiro/2017-
61 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Msc. Ederson Torresini

Monografia (Graduação) – Instituto Federal de Santa Catarina – IFSC
Campus São José
Sistemas de Telecomunicações, Fevereiro/2017.

1. Palavra-chave1. 2. Palavra-chave2. 2. Palavra-chave3. I. Orientador. II. Instituto Federal de Santa Catarina. III. Campus São José. IV. Título

Maicky Müller

Implantação de alta disponibilidade em PABX IP

Monografia apresentada à Coordenação do curso de tecnologia em Sistemas de Telecomunicações do Instituto Federal de Santa Catarina para a obtenção do diploma tecnólogo em Telecomunicações.

Trabalho aprovado. São José - SC, 15 de outubro de 2015:

Prof. Msc. Ederson Torresini
Orientador

Professor
Convidado 1

Professor
Convidado 2

São José - SC
Fevereiro/2017

Este trabalho é dedicado especialmente a minha família, que sempre me apoiou diante das dificuldades impostas pela vida. Também dedico aos amigos e próximos que me acompanham nesta jornada.

Agradecimentos

Os agradecimentos principais são direcionados ao meu professor e orientador Ederson Torresini, pela paciência para comigo e pelo empenho dedicado à elaboração deste trabalho. Também agradeço a todos que direta e indiretamente fizeram parte da minha formação, o meu muito obrigado.

Resumo

Atualmente a telefonia IP é um assunto muito visado pela grande indústria tecnológica, considerando que a voz transportada através de pacotes é uma saída bastante atrativa por oferecer baixo custo e qualidade aceitável.

Com o surgimento da telefonia IP, surgiram também as centrais telefônicas IP. Uma central telefônica IP é tratada como um servidor, e esta deve atender o máximo de usuários possíveis com alta qualidade e alta disponibilidade. No entanto, problemas no dia a dia levam esses equipamentos a eventuais falhas e até desligamento, consequentemente usuários antes autenticados na central perdem seu registro e ficam sem o serviço de telefonia, que é essencial e até primordial em inúmeras empresas.

Falhas como queda de energia e problemas com o *hardware* da central estão sujeitos a ocorrer a qualquer momento. Baseado nestes problemas, existem diversas maneiras de se contornar a indisponibilidade, os quais serão abordados neste trabalho. Atualmente o mundo corporativo depende essencialmente da telefonia para execução e manutenção de seus serviços. Hospitais, sistemas bancários, call centers, entre outros segmentos, dependem de alta disponibilidade do serviço, pois atuam em setores primordiais para população e não podem cogitar a possibilidade de terem seus serviços prejudicados. Com base nesta necessidade do mercado, surge a necessidade do estudo e implementação de alta disponibilidade em PABX IP.

Palavras-chave: telefonia IP. alta disponibilidade.

Abstract

Nowadays IP telephony is a subject very targeted by the great technological industry, considering that the voice transported through packages is a very attractive exit for offering low cost and acceptable quality.

With the emergence of IP telephony, there were also the IP telephone exchanges. An IP PBX is treated as a server, and it must serve as many users as possible with high quality and high availability. However, day-to-day problems lead these devices to eventual failures and even shutdown, consequently users before authenticated in the central lose their registration and are left without the telephone service, which is essential and even primordial in numerous companies.

Failures such as power outages and problems with the hardware of the control panel are subject to occur at any time. Based on these problems, there are several ways to avoid unavailability, which will be addressed in this work. Currently the corporate world depends essentially on the telephony for the execution and maintenance of its services. Hospitals, banking systems, call centers, among other segments, depend on the high availability of the service, since they operate in sectors that are primordial for the population and can not consider the possibility of having their services undermined. Based on this market need, there is a need to study and implement IP PBX high availability.

Keywords: IP telephony. high availability.

Lista de ilustrações

Sumário

1 Introdução

Atualmente a telefonia IP é um assunto muito visado pela grande indústria tecnológica, considerando que a voz transportada através de pacotes é uma saída bastante atrativa por oferecer baixo custo e boa qualidade.

As centrais telefônicas analógicas privadas surgiram em meados dos anos 1980. Nesta época os computadores e microcontroladores eram extremamente limitados. Haviam pouquíssimas soluções de redes de dados locais, mas ainda com muitas falhas ou de alto custo. Frente as necessidades da época, os PABX analógicos já eram considerados eficientes, devido ao equilíbrio entre funcionalidade e serviços de telefonia prestados (JUNIOR, 2014).

Após o surgimento das centrais telefônicas analógicas surgiu a telefonia IP, uma aplicação de Voz sobre IP (VoIP), ou seja, a voz transportada através de pacotes trafegando pela rede de dados. A primeira aplicação a realizar este tipo de serviço foi o *Internet Phone Software*, em 1995, construído pela empresa Vocaltec nc., de Israel. O objetivo era o desenvolvimento de um sistema que permitisse a utilização dos recursos multimídia de um computador doméstico para iniciar conversas de voz pela internet. Na época a qualidade era muito baixa, com muitos cortes e atrasos.

Com o surgimento da telefonia IP, surgiram também as centrais telefônicas IP. Estas utilizam protocolos de sinalização, como por exemplo o protocolo SIP, para iniciar, modificar ou terminar sessões, que contém qualquer tipo de tráfego multimídia, seja esse áudio, vídeo ou compartilhamento de tela. Este protocolo utiliza uma arquitetura semelhante ao modelo cliente-servidor: o terminal que solicita o chamado atua como agente cliente (*User Agent Client* - UAC) e o que recebe o chamado atua como servidor (*User Agent Server* - UAS).

Uma central telefônica IP é tratada como um servidor. Esta deve atender o máximo de usuários possíveis com alta qualidade e alta disponibilidade. No entanto, problemas no dia a dia levam esses equipamentos a eventuais falhas e até desligamento, consequentemente usuários antes autenticados na central perdem seu registro e ficam sem o serviço de telefonia, que é essencial e até primordial em inúmeras empresas.

Falhas como queda de energia e problemas com o *hardware* da central estão sujeitos a ocorrer a qualquer momento. Baseado nestes problemas, existem diversas maneiras de se contornar a indisponibilidade, os quais serão abordados neste trabalho e explícitos nos capítulos seguintes.

1.1 Motivação

Atualmente o mundo corporativo depende essencialmente da telefonia para execução e manutenção de seus serviços. Hospitais, sistemas bancários, *call centers*, entre outros segmentos, dependem de alta disponibilidade do serviço, pois atuam em setores primordiais para população e não podem cogitar a possibilidade de terem seus serviços prejudicados. Alguns minutos sem a disponibilidade de telefonia nestes segmentos pode custar caro para o faturamento destes setores mesmas ou até salvar vidas, como por exemplo um hospital, que necessita trivialmente da telefonia para comunicação interna de seus médicos e profissionais da saúde.

Para empresas do setor bancário, outro exemplo, a telefonia é ainda um meio muito utilizado por clientes para comunicação com seus gerentes e funcionários do banco. É de extrema importância que a telefonia esteja operando em perfeitas condições para que o cliente tenha um atendimento de qualidade e consequentemente adquira produtos e serviços que são oferecidos ao mesmo. Este é apenas mais um exemplo dentre tantos outros setores que dependem da telefonia para ter lucratividade, qualidade e excelência no atendimento aos seus clientes.

Com base nesta necessidade do mercado, surge a possibilidade do estudo e possível implementação de alta disponibilidade em um PABX IP. O modo escolhido para implantar alta disponibilidade no trabalho foi do tipo Ativo-Passivo, motivado pelos fatores que será vistos nos próximos capítulos do documento.

1.2 Objetivos

Este trabalho tem por objetivo principal a implantação de alta disponibilidade para uma central telefônica IP caso a mesma apresente alguma falha que a torne indisponível para o usuário. Serão estudados métodos e protocolos que garantam o registro dos telefones IP da central telefônica principal, em uma central telefônica idêntica secundária em caso de falha na central principal. Além da garantia de registro dos terminais, serão preservados os arquivos de bilhetagem da central principal e os arquivos de configuração das centrais.

Para que seja possível esta implementação, deverão ser estudados os seguintes itens:

- Protocolos de redundância: Utilizados para virtualização de IP das centrais;
- Ferramentas de sincronismo: Para sincronização dos arquivos de configuração das centrais principal e secundária;
- Sistemas gerenciadores de banco de dados: Para garantir a integridade e sincronismo dos arquivos de bilhetagem das centrais telefônicas principal e secundária;

Não é um objetivo do trabalho garantir a integridade das ligações em curso (áudio) em caso de falha e convergência para central telefônica secundária. Pretendemos garantir apenas o registro dos ramais, de forma que tenham o mínimo tempo com seus terminais sem registro em caso de falha na central principal.

1.3 Organização do texto

O texto está organizado da seguinte forma: no capítulo 2 apresentamos um resumo dos protocolos de redundância e sincronismo de dados que serão empregados ao longo deste trabalho em conjunto com alguns trabalhos relevantes da área. No capítulo 3 são apresentadas as propostas para o desenvolvimento do trabalho. No capítulo 4 serão apresentados os resultados e conclusões do trabalho.

2 Revisão Bibliográfica

2.1 Centrais telefônicas IP

As centrais telefônicas analógicas privadas surgiram em meados dos anos 1980. Nesta época os computadores e microcontroladores eram muito limitados e a rede de dados ainda era bastante escassa. Haviam pouquíssimas soluções de redes de dados locais, mas ainda com muitas falhas ou de alto custo. O que havia, à época para soluções dentro das instituições, eram as centrais analógicas.

As centrais analógicas utilizam a comutação de circuitos para conectar usuários internos. Para se comunicar com outros usuários ou centrais de locais distintos, utilizam o Sistema de Telefonia Fixa Comutada (STFC). Estes PABXs utilizavam uma implementação proprietária, o que dificultava ao usuário uma possível adição de novas funcionalidades. A figura 2.1 apresenta o exemplo de arquitetura de um PABX analógico.

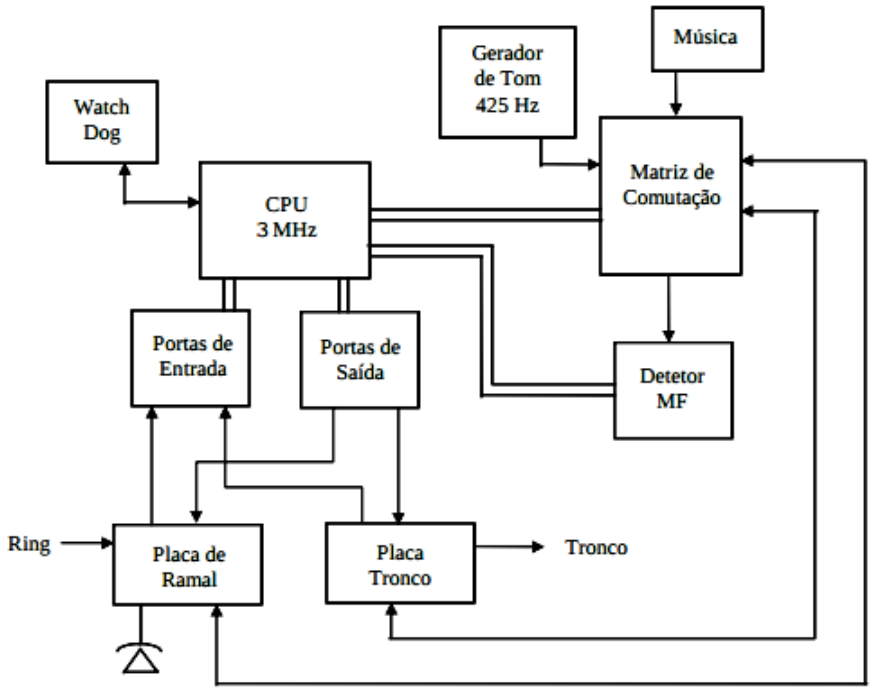

Figura 1 – Arquitetura de um PABX analógico.

Com o advento da Internet, e principalmente da popularização dos protocolos da arquitetura TCP/IP, surgiram alternativas, como as centrais baseadas em IP. A telefonia IP, portanto, é uma aplicação de voz sobre IP (VoIP), onde a transmissão de voz se dá via rede de dados, permitindo uma comutação entre usuários semelhante ao sistema de telefonia fixa comutada.

Conforme (FILHO, 2003), os terminais os terminais implementam parte da infraestrutura, ao processar pacotes, e o processamento e a realização das chamadas ocorrem em vários equipamentos que podem estar localizados em qualquer parte da rede. Na figura 2.2 é apresentado um exemplo de arquitetura de rede para a telefonia IP.

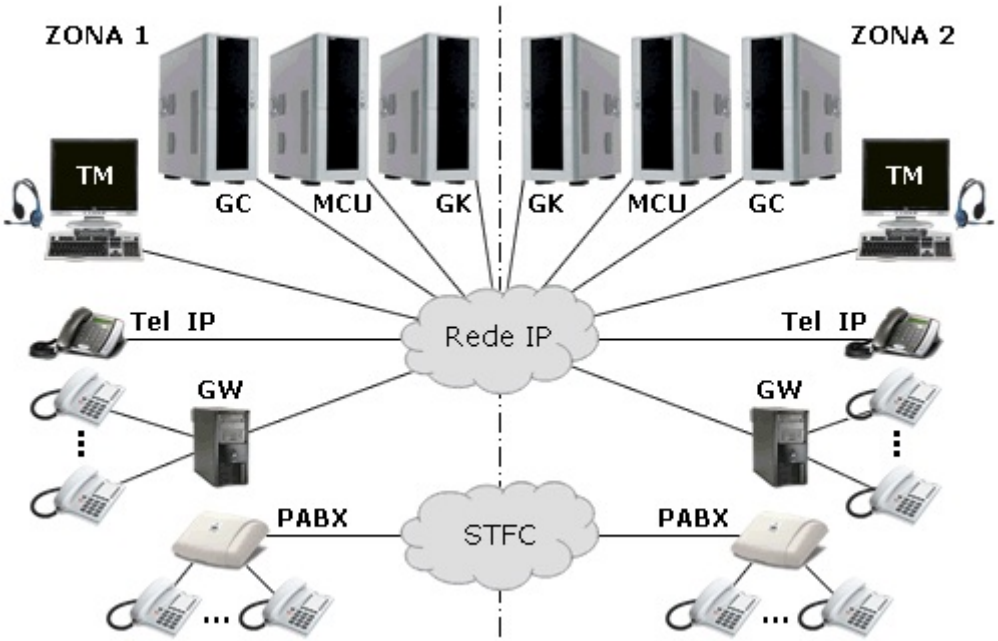

Figura 2 – Exemplo de arquitetura de rede para telefonia IP (FILHO, 2003).

Para (RIBEIRO, 2011) a presença do VoIP em ambientes domésticos e corporativos está em constante crescimento. Segundo dados do site Teleco, o ano de 2009 terminou com cerca de 1,8 milhão de assinantes VoIP no Brasil. E o site Jornal Floripa (conforme artigo acessado em: 17 set. 2010) também revela que os serviços baseados em VoIP cresceu na América Latina entre 2005 e 2011: o número de linhas com a tecnologia cresce 87,5% ao ano, sendo o Brasil o maior responsável por essa adoção.[1]

Assim, quinze anos após o surgimento das centrais privadas analógicas, os fabricantes de PABX vem introduzindo uma radical mudança em sua arquitetura. Segundo (SATO, 2004) a tecnologia que está ocasionando esta mudança é o VoIP, ou seja, a possibilidade de utilizar a mesma infraestrutura de uma rede IP para trafegar voz com uma boa qualidade, uma boa velocidade (pouco atraso) e com confiabilidade aceitável.

Surgem então os PABX IP, os quais têm como objetivo fazer o gerenciamento de chamadas utilizando um *software*, geralmente proprietário, via rede de dados. Estes, são baseados na arquitetura de um servidor onde um sistema operacional livre como o GNU/Linux pode ser empregado.

A seguir será apresentada uma breve explicação sobre a plataforma escolhida como PABX neste trabalho.

[1] A Anatel não discrimina VoIP como serviço de telefonia, mas como Serviço de Comunicação Multimídia (SCM). Assim, não há como obter dados oficiais recentes a partir dos seus Anuários.

2.1.1 Asterisk

Asterisk é um *software* de código aberto baseado na plataforma GNU/Linux que implementa uma central telefônica totalmente flexível quanto a configurações e funcionalidades. O *software* básico do Asterisk inclui características encontradas até então somente em sistemas telefônicos proprietários: correio de voz, respostas interativas, distribuição automática de chamadas e conferência em chamadas. É possível ainda criar novas funcionalidades escrevendo aplicativos ou mesmo módulos da plataforma, além de outras formas de modificação (ROSA, 2007).

Segundo (ROCHA et al., 2013), o Asterisk é o que chamamos de *Back-to-Back* (B2B) *User Agent*: em uma chamada telefônica estabelecida, cada "perna"(*leg*) da chamada entre origem e destino é processada pela central, monitorando assim todo o tráfego de áudio entre esses pontos.

Com base nas necessidades de se manter a alta disponibilidade de uma central telefônica IP, neste caso a plataforma escolhida o Asterisk, a seguir será apresentado uma breve explicação sobre alta disponibilidade.

2.2 Alta disponibilidade

A disponibilidade de um sistema consiste na capacidade de um usuário incluir ou modificar dados existentes neste sistema, com a garantia de integridade de quaisquer alterações realizadas em qualquer intervalo de tempo. Caso o usuário não consiga acessar estes dados, o sistema é definido como indisponível.

Máquinas normalmente apresentam disponibilidade na faixa de 99,99% a 99,999%, podendo ficar indisponíveis por período de pouco mais de cinco minutos até uma hora em um ano de operação. Aqui se encaixam grande parte das aplicações comerciais de alta disponibilidade, como centrais telefônicas. (GUINDANI, 2008)

Atingir altas taxas de disponibilidade só é possível por intermédio de dispositivos de *software* ou *hardware* que sejam capazes de detectar e recuperar sistemas que sofreram falhas (DANTAS, 2009).

Uma possível solução para problemas com disponibilidade de sistemas é a redundância de *hardware* nos equipamentos e nas suas ligações à rede. Outra solução é a construção de um aglomerado (*cluster*). Conforme (FERREIRA; SANTOS; ANTUNES, 2005), um aglomerado é um conjunto de computadores independentes, designados de nós, que colaboram entre si para atingir um determinado objetivo comum, podendo classificar-se como de alta disponibilidade ou de alto desempenho. A figura 3 apresenta um exemplo de um *cluster* típico.

As arquiteturas de alta disponibilidade são baseadas em:

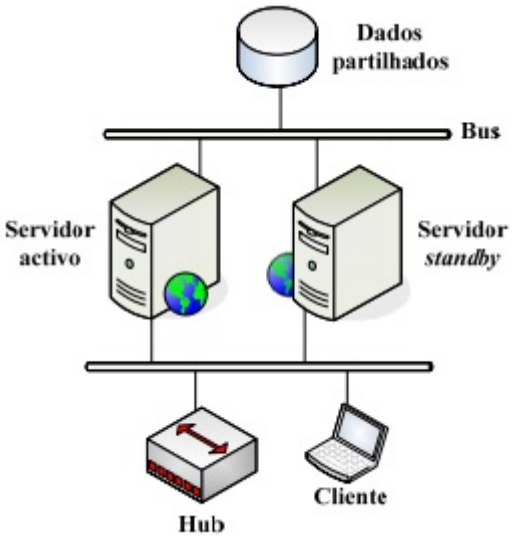

Figura 3 – Exemplo de um *cluster* típico.(FERREIRA; SANTOS; ANTUNES, 2005)

- Ativo/passivo: Um dos equipamentos atua como principal (ativo) e os demais atuam como secundários (passivos), sendo que o ativo envia todas as informações de estado para os passivos e, se o ativo falhar, um dos passivos é promovido a ativo e assume as sessões previamente estabelecidas.

- Ativo/ativo: Todos os equipamentos são configurados para o estado ativo, compartilhando as sessões entre eles (com possibilidade de balanceamento dinâmico de carga) e se um dos equipamentos falha, os demais assumem as sessões e os seus respectivos tráfegos (HASHIMOTO, 2009).

Para aplicação de alta disponibilidade no trabalho, foram estudados alguns protocolos e aplicações. A seguir serão apresentadas as ferramentas estudadas.

2.2.1 Projeto Linux-HA

O projeto Linux-HA começou em 1997 como uma lista de discussão criada por Harald Milz para discutir como se pode criar um conjunto de recursos de alta disponibilidade para GNU/Linux.

À época, o autor trabalhou para a Bell Labs, e estava investigando o Linux como uma plataforma para usar em produtos e por um ambiente de desenvolvimento. Como parte deste trabalho, ele foi convidado a investigar que tipo de funcionalidades o Linux-HA possuía. Em 1998, o autor desenvolveu um *software* que iria realizar a função de "batimento cardíaco". Naturalmente, o *software* foi assim chamado de *heartbeat*. Era um projeto simples, operando apenas através de portas seriais e fornecendo detecção apenas da "morte"do servidor através de seus "batimentos cardíacos". A intenção deste programa foi dar embasamento para um salto no projeto Linux-HA. Embora o projeto tem se expandido

muito além da função de batimento cardíaco inicial, o *software* é ainda frequentemente chamado apenas de *heartbeat*(ROBERTSON, 2004).

Ao longo do tempo o Projeto linux-HA teve um amadurecimento considerável, com a afirmação do *software* Heartbeat, cujo maior objetivo é desenvolver soluções GNU/Linux com o intuito de promover confiabilidade e disponibilidade. Este *software* é responsável por monitorar os servidores presentes no aglomerado para que, em caso de falhas do nó primário, sejam estas de *hardware* ou *software*, sejam realizados automaticamente os procedimentos necessários para disponibilizar os recursos e/ou aplicativos através do nó secundário - preservando assim a alta disponibilidade do aglomerado (Sá; HIURI, 2012).

Para que o *software* faça o monitoramento dos servidores, estes podem estar conectados fisicamente por uma interface dedicada (conexão serial ou paralela) ou mesmo rede multiponto, a exemplo de Ethernet.

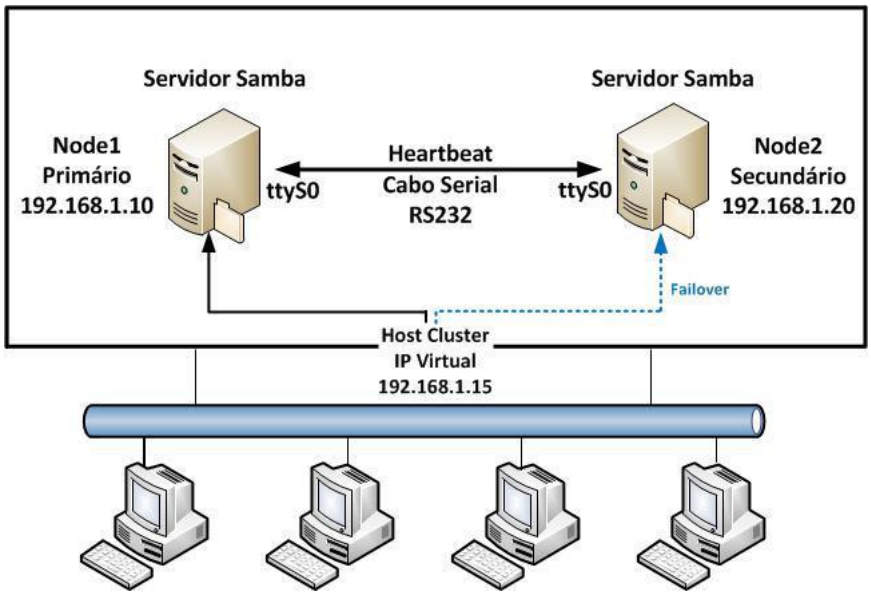

Figura 4 – Funcionamento do Heartbeat (Sá; HIURI, 2012).

A figura 4 mostra o funcionamento do Heartbeat usando conexão serial, e o endereço IP 192.168.1.15 é um IP virtual gerenciado pelo Heartbeat. Os clientes têm acesso ao serviço Samba (SERVIDOR..., 2013) do servidor por este IP. Os endereços IPs 192.168.1.10 e 192.168.1.20 são fixos, usados apenas para permitir o gerenciamento remoto, independente de qual servidor estiver com o IP virtual configurado. Pode-se definir o funcionamento do Heartbeat:

1. O Heartbeat no servidor secundário troca mensagens com o servidor primário através da conexão serial assíncrona RS232.

2. Em caso de falha na comunicação entre os servidores, o Heartbeat do servidor secundário inicia o serviço Samba e configura o IP 192.168.1.15, passando a funcionar como o servidor principal.

3. Quando o servidor primário estiver novamente disponível, um pacote é enviado através da comunicação serial para avisar sobre sua disponibilidade para o servidor secundário.

4. O servidor secundário então para o serviço Samba, libera o IP 192.168.1.15 configurado anteriormente, e envia uma mensagem para o servidor primário, avisando-o sobre a possibilidade dele assumir novamente os serviços.

5. O servidor primário então inicia o Samba e configura novamente o IP 192.168.1.15, voltando assim a ser o servidor em produção.

A seguir apresentamos duas opções de protocolos de redundância, onde ambos aproveitam o mesmo conceito de IP virtual para conexão com outros computadores fora do aglomerado.

2.3 Protocolos de redundância

2.3.1 Protocolo VRRP

O protocolo *Virtual Router Redundancy Protocol* (VRRP) é um protocolo da arquitetura Internet especificado em Hinden (2015) definido como um roteador virtual, foi desenvolvido pela Cisco e ficou conhecido como HSRP antes de ser definida sua RFC. Provê maior confiabilidade aos ambientes de rede, resolvendo o problema do ponto único de falha de uma rede quando esta usa um único endereço IP (HINDEN, 2004).

Resumidamente o VRRP é um protocolo de eleição e funciona da seguinte maneira:

- Um dos *Hosts* da rede é eleito para assumir o IP virtual, baseado na sua prioridade (definida em arquivo de configuração, ver Seção A.2).

- Em caso de alguma falha com o *host master*, o próximo *host* com maior prioridade assumirá o IP virtual.

- Assim que o *host master* volta a funcionar, automaticamente volta a ter maior prioridade e consequentemente assume o IP virtual novamente.

Segundo (HASHIMOTO, 2009) o protocolo VRRP é um protocolo de redundância de roteamento IP desenvolvido para permitir uma transparente comutação do roteador da rede. Esse método provê a redundância sem a intervenção manual do usuário ou mesmo sem configuração adicional nos elementos de rede.

Um roteador virtual é definido pelo seu identificador virtual (VRID) e um conjunto de endereços IP. Um roteador VRRP pode associar um roteador virtual com seus endereços reais em uma interface.

São chamadas de *Link-State Advertisement* (LSA) as trocas de mensagens entre os roteadores da rede. O roteador principal, chamado de mestre, envia essas mensagens para os roteadores de *backup*. Os LSAs são encaminhados a cada 1 (um) segundo por padrão. Os roteadores em estado de *backup* podem ser configurados para reconhecer o intervalo de tempo em que o roteador mestre encaminha os LSAs. Essas mensagens são enviadas através do endereço *multicast* `224.0.0.18`. Quando o roteador mestre fica inoperante, ocorre o processo de eleição, e um roteador *backup* torna-se mestre. Os roteadores em estado *backup* percebem a ausência de LSA e então o roteador que possui a prioridade mais alta assume como roteador mestre, sendo que a prioridade varia entre 1 até 254 e é definida em arquivo de configuração (como pode ser visto na Seção A.2. A prioridade igual a 100 é considerada como valor padrão.

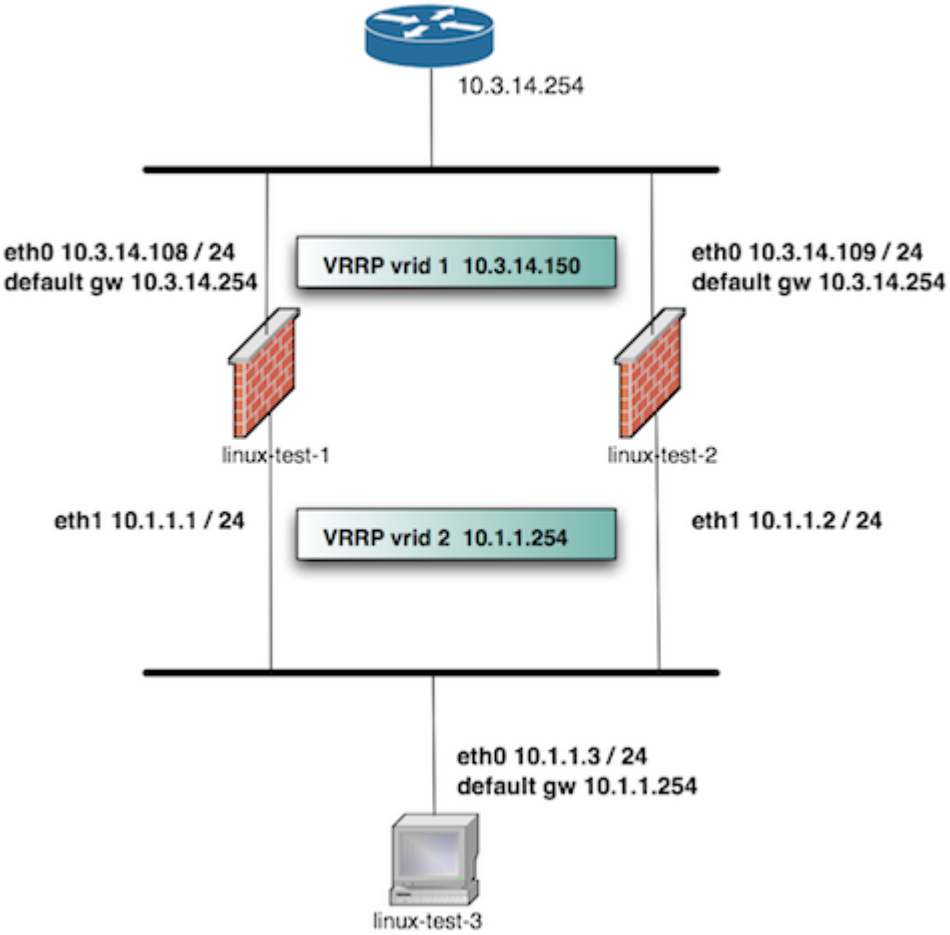

Figura 5 – Cenário de uso do protocolo VRRP rodando em dois servidores linux. Fonte: (HASHIMOTO, 2009).

Na figura 5 é apresentado um exemplo de uso do protocolo VRRP. Neste, temos duas máquinas GNU/Linux executando o protocolo VRRP para *failover* formando alta disponibilidade entre um par de *firewalls*, e uma terceira máquina usará o *firewall* em operação. As Máquinas `linux-test-1` e `linux-test-2` são os *firewalls* e `linux-test-3` é a estação de trabalho. Todos os testes são feitos em uma rede isolada usando endereços

IP privados, a sub-rede "externa"dos *firewalls* é 10.3.14.0/255.255.255.0 e a sub-rede "interna"dos *firewalls* é 10.1.1.0/255.255.255.0.

As máquinas `linux-test-1` e `linux-test-2` executam o serviço VRRP para criar o endereço IP virtual em ambas as sub-redes. O mecanismo então acrescenta um mesmo endereço IP para as interfaces `eth0` e `eth1`. Uma das máquinas torna-se mestre e toma posse do endereço virtual, adicionando-o à interface. Ele envia um datagrama UDP por segundo para o endereço de *multicast* `224.0.0.18` para declarar que está funcionando e possui o endereço. Se a máquina que está executando o protocolo é desligada por qualquer motivo, este fluxo de pacotes do mestre para e depois de um tempo limite pré-determinado, a segunda máquina se torna o mestre e assume o endereço IP virtual. O serviço também substitui o endereço MAC da interface por um endereço MAC virtual, de modo que quando o endereço IP virtual é transferido de uma máquina para outra, todas as máquinas da sub-rede correspondente não têm de atualizar suas tabelas ARP pois o endereço MAC permanece o mesmo.

A seguir será apresentado o protocolo de redundância CARP, uma alternativa aberta em relação a esse protocolo.

2.3.2 Protocolo CARP

O protocolo *Common Address Redundancy Protocol* (CARP) , assim como o VRRP, possibilita que múltiplos *computadores* da mesma rede local concorram por um mesmo endereço IP. Conforme (OpenBSD, 2017), CARP é uma alternativa segura e livre para o *Virtual Router Redundancy Protocol* (VRRP) (Seção 2.3.1) assim como do protocolo de *Hot Standby Router* (HSRP) (LI et al., 1998).

Em termos de funcionamento, o protocolo CARP é equivalente ao protocolo VRRP (Seção 2.3.1). Há um grupo de hospedeiros referidos como grupo de redundância. A esse grupo de redundância é atribuído um endereço IP virtual, o qual é compartilhado entre os membros do grupo. Dentro do grupo, um desses é designado como "mestre"e os demais como "backups". O host designado mestre, que atualmente possui o IP compartilhado, é o host que responde a todo tráfego ou solicitações de ARP direcionadas para ele. Cada hospedeiro pode pertencer a mais do que um grupo de redundância de cada vez. Desta forma, a diferença entre CARP e VRRP é que o protocolo CARP é livre conforme citado anteriormente.

Um uso comum para o CARP é criar um grupo de *firewalls* redundantes. O IP virtual que é atribuído ao grupo de redundância é configurado nas máquinas clientes como o *gateway* padrão. No caso em que o *firewall* mestre sofra uma falha ou seja desligado, o IP virtual será assumido por um dos *firewalls backup* e o serviço continuará disponível.

2.4 Ferramentas de Sincronismo

Uma vez analisadas as soluções de redundância em rede, serão vistos agora as opções de sincronização de dados: RSYNC, DRBD e Galera Cluster.

2.4.1 Rsync

Rsync é um aplicativo de rede que serve para sincronização de arquivos de um diretório a outro, local ou remoto. Seu funcionamento se baseia na transferência em delta; ou seja, o envio de arquivos é baseado na diferença do(s) arquivo(s) atual(is) e do(s) novo(s), o que reduz o tráfego na rede e permite que a mesma seja bastante utilizada para a realização de backups e espelhamento de arquivos.

Há duas maneiras diferentes de uso no rsync: usando um programa de controle remoto, como por exemplo *ssh* ou *rsh*, ou utilizando o serviço rsyncd - diretamente via TCP (RSYNC, 2015).

A seguir o estudo do DRBD, outra solução em sincronização de dados.

2.4.2 DRBD

O DRBD (dispositivo de bloco distribuído replicado) é um software baseado em replicação e espelhamento de conteúdo de dispositivos de bloco como: discos rígidos, partições, volumes lógicos etc; entre servidores. O DRBD resume-se em um módulo do kernel do sistema operacional Linux que, em conjunto com scripts e programas, disponibilizam blocos de armazenamento distribuídos, altamente utilizado em clusters de alta disponibilidade (Sá; HIURI, 2012). Semelhante ao funcionamento do RAID-1 (Redundant Array of Inexpensive Disks) que trabalha com espelhamento entre dois discos localizados em uma mesma máquina, o DRBD também opera espelhando dados, porém entre servidores interligados em rede, como ilustrado na Figura 6.

Existem duas formas de fazer este espelhamento: de forma síncrona ou assíncrona. Quando o espelhamento é síncrono, significa que o sistema de arquivos no nó ativo é notificado de que a escrita do bloco só foi concluída quando o bloco fez para ambos os discos de um aglomerado. Este espelhamento é a escolha certa para aglomerados de alta disponibilidade (mínimo de 99,999%). A outra opção é o espelhamento assíncrono. Isso significa que a entidade que emitiu as solicitações de gravação é informada sobre a conclusão assim que os dados são gravados no disco local. Para o espelhamento assíncrono é necessária a construção de espelhos em longas distâncias, ou seja, o tempo de interconexão da rede de ida e volta é maior do que a latência de gravação que você pode tolerar para sua aplicação.(LINBIT, 2016)

Na Figura 7 , as duas caixas em laranja representam dois servidores que formam

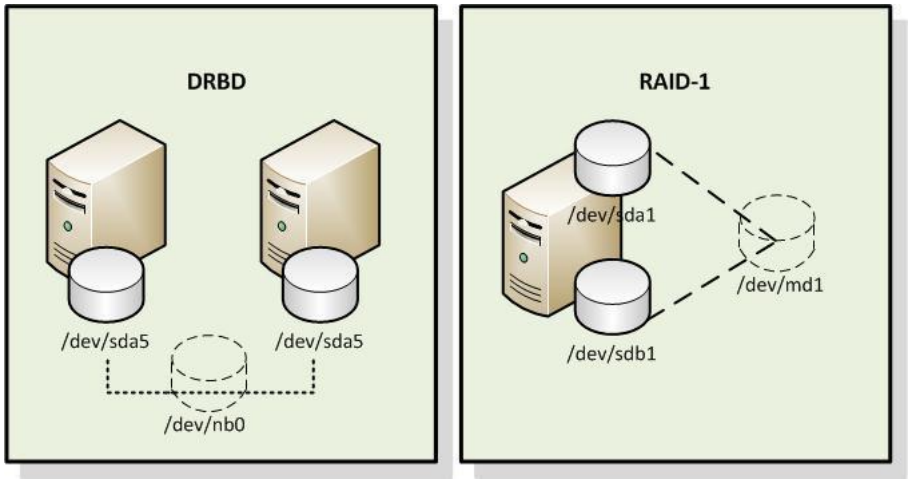

Figura 6 – DRBD X RAID 1 (Sá; HIURI, 2012).

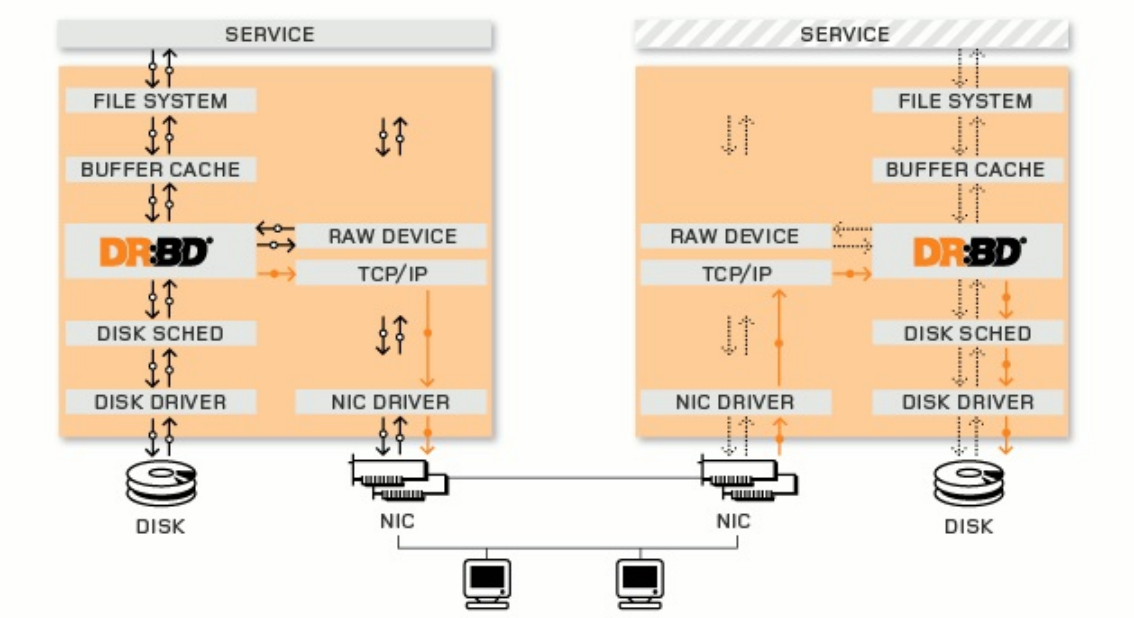

Figura 7 – Funcionamento do DRBD. Fonte: LINBIT (2016).

um aglomerado de alta disponibilidade. As caixas contêm os componentes habituais de um *kernel* Linux: sistema de arquivos, memória *cache*, disco programador, *drivers* de disco, pilha TCP/IP e interface de rede (*Network Interface Card* - NIC). As setas pretas mostram o fluxo de dados entre esses componentes. As setas laranja mostram o fluxo de dados: O DRBD espelha os dados de um serviço altamente disponível, a partir do nó ativo do aglomerado de HA para o nó de espera do grupo HA.

2.4.3 Galera Cluster

Além de dados organizados em arquivos e distribuídos em rede, como visto anteriormente, também há a replicação de dados entre sistemas gerenciadores de bancos de dados (SGBDs). Define-se como conjunto de banco de dados ou aglomerado de banco de

dados um sistema de replicação distribuída, onde todos compartilham o mesmo tipo de informação e mecanismo de sincronização.

Para replicações do tipo mestre-escravo o servidor principal registra as atualizações dos dados e propaga os logs (conjunto de operações que modificam a estrutura ou os dados) através da rede para os escravos. Estes por sua vez recebem o fluxo de dados e aplicam estas alterações. A figura abaixo ilustra o funcionamento da replicação Mestre-Escravo.

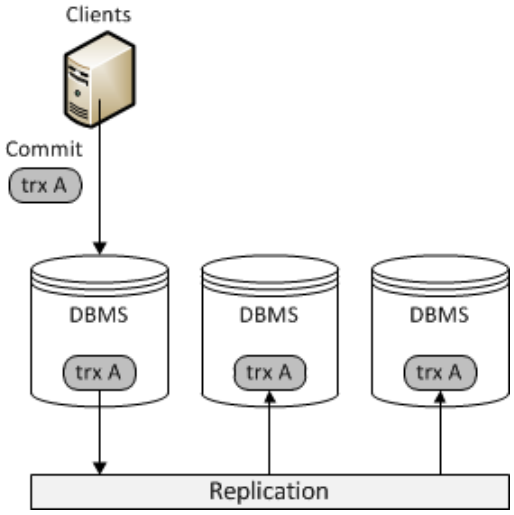

Figura 8 – Replicação Mestre-Escravo. Fonte: Galera... (2016).

Em um sistema de replicação multi-mestre ou mestre-mestre - como também é conhecida - você pode enviar atualizações a qualquer nó de banco de dados. Essas atualizações, em seguida, propagam-se através da rede para os outros nós de banco de dados. Todos os nós de banco de dados funcionam como mestres. A imagem abaixo ilustra o funcionamento da replicação multi-mestre.

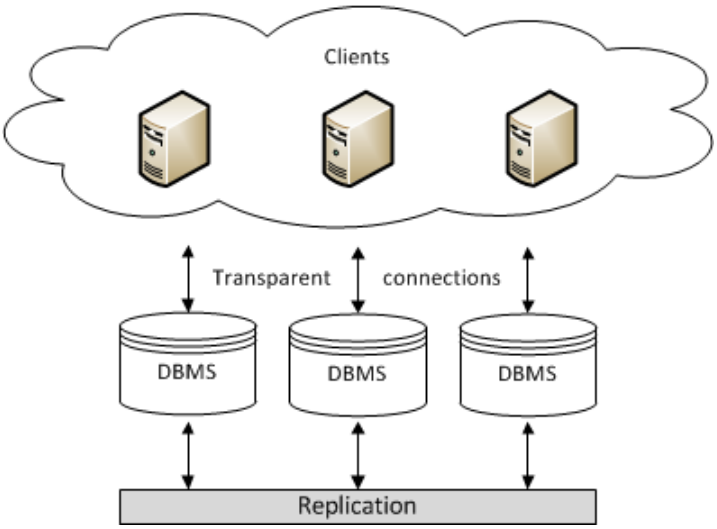

Figura 9 – Replicação Multi-Mestre / mestre-mestre. Fonte: Galera... (2016).

Uma implementação existente para o SGBD Oracle MySQL e semelhantes, como

por exemplo MariaDB e Percona XtraDB, para o método de replicação multi-mestre é o Galera Cluster (GALERA. . . , 2016). Trata-se de um grupo de banco de dados que utiliza sincronismo do tipo multi-mestre, ou seja, todos os nós compartilham a mesma informação.

No modelo de sincronismo multi-mestre, temos duas implicações, são elas:

1. Todos os nós são mestres, o que significa que não há uma fonte primária de leitura e escrita; ou seja, todos os nós estão aptos a processar tanto leitura (consultas) quanto escrita (inserção/modificação);

2. A sincronização é bidirecional; ou seja, é preciso um controle de versão dos dados armazenados para comparação na linha do tempo (o que foi modificado por último, etc.)

Assim, no modelo mestre-mestre o nó e a falha são isolados e o restante do ambiente continua em funcionamento. Em caso de falha de algum servidor e retorno imediato do mesmo, os dados serão sincronizados pelo próprio Galera, através do controle de versões que é feito pela ordem dos logs. Em caso de retorno com alto retardo - e portanto uma longa sequência de logs -, os dados poderão ser sincronizados através do mecanismo Rsync, conforme apresentado no item 2.4.1.

2.5 Trabalhos Relacionados

A necessidade de implementação de sistemas de alta disponibilidade (veja Seção 2.2) levou a pesquisa e estudo de diversos trabalhos relacionados ao assunto. Um assunto amplo e complexo que necessita de atenção e dedicação para que haja o entendimento e então a implementação do mesmo.

Nesta seção serão apresentados alguns trabalhos relacionados com o uso dos protocolos de redundância estudados anteriormente , veja Seção 2.3. Entre eles estão VRRP, CARP e Projeto Linux-HA.

Com a crescente variedade de dispositivos de acesso a rede de dados que aparecem no mercado, cresce a necessidade de manter a disponibilidade dos serviços utilizados através da rede de dados. A fim de complementar as capacidades dos roteadores genéricos, tais como *firewall*, detecção de intrusão, tradução de endereço de rede (NAT, veja em Egevang e Francis (1994)), controle de tráfego, otimização de rotas, o autor Shinn (2009), centra-se na análise de vários protocolos e uma implementação do RGP (*Redundant Gateway Protocol*), que auxiliam a detecção e a solução de indisponibilidade de serviços com o uso de redundância de hardware.

Em Shinn (2009) o autor desenvolveu seu trabalho baseado em análises de alguns protocolos de redundância. São eles: HSRP (LI et al., 1998), VRRP (KNIGHT et al., 1998)

e o Protocolo de espera IP (proprietário da *Digital Equipment Corporation*). Também promoveu a implementação do RGP (protocolo de redundância de *gateway*) a fim de tratar problemas de disponibilidade em redes de acesso internet. O título do trabalho é *Tolerância a falhas de roteadores virtuais para servidores virtuais Linux*.

O esquema proposto no artigo em discussão de Shinn (2009) pode ser implantado em qualquer rede Ethernet baseada em uma topologia que permite duas conexões entre uma sub-rede e um *backbone* (núcleo da rede). Para tanto usam-se roteadores virtuais. Roteador virtual é um conjunto de dois ou mais roteadores reais que fornecem um esquema de tolerância a falhas de conectividade do *backbone* com nós da rede local. Enquanto um dos dois roteadores funciona como roteador principal, o outro funciona como roteador de *backup*. Por padrão, todo o tráfego de saída para o *backbone* da Internet a partir de máquinas locais passa pelo roteador principal, que é configurado como *gateway* padrão para os nós da rede local. Quando o roteador principal falhar, todos os nós da rede que têm seus *gateways* configurados para ser o roteador principal terão a conexão com a Internet perdida. O uso de roteadores virtuais pode evitar essa situação, porém há necessidade de reconfigurar todos os nós da rede para apontar seu *gateway* para o roteador de *backup* quando o roteador principal estiver fora de operação, esta tarefa, apesar de trivial, dispende bastante tempo. O roteador de *backup* verifica o estado do roteador principal periodicamente, caso detecte que o mesmo está fora de operação, então alterna para o modo *failover*. No modo *failover* o roteador *backup* irá assumir automaticamente um endereço IP virtual (endereço da camada de rede) e um endereço MAC virtual (endereço da camada de enlace de dados) para servir os nós da rede local como seu *gateway* padrão. Por este caminho, a conectividade com a Internet continua sem interrupção.

No artigo de Shinn (2009) o autor faz uma breve análise do HSRP, VRRP e Protocolo de espera IP para tratar o problema de disponibilidade. O RGP (Redundant Gateway Protocol) foi desenvolvido por meio da otimização do protocolo VRRPv2 (KNIGHT et al., 1998) e projetado para prover redundância na camada de rede. O protocolo de redundância de *gateway* não troca nenhuma mensagem com as camadas acima da camada de rede (por exemplo, camada de transporte). Consequentemente, qualquer *failover* é feito no nível de rede, assim os estados das conexões TCP serão mantidas. A implementação do RGP se deu através de três casos: *backup* único, *backup* mútuo e rastreamento de porta.

No caso de *backup* único (Figura 10) todas as máquinas usam roteador A com IP padrão. O roteador B faz o backup do roteador A. Se o roteador B não recebe qualquer anúncio VRRP do roteador A durante o intervalo que o principal ficou inoperante (padrão em torno de 3 segundos), o roteador B assume o papel de principal e começa a responder ao endereço IP do roteador virtual ou seja, IP A. Porém se o roteador B falhar, o roteador A não assumirá.

O *backup* mútuo (Figura 11) permite o balanceamento de carga e alta disponibili-

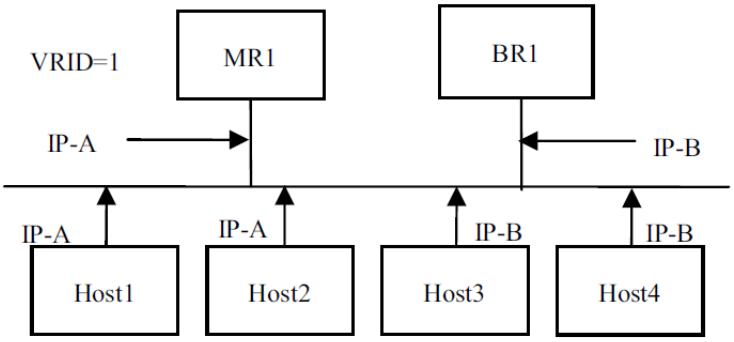

Figura 10 – RGP por Backup único (SHINN, 2009).

dade. Metade das máquinas usam o roteador A como o roteador padrão, e a outra metade usa o roteador B. Assim, a carga é dividida entre A e B. Os roteadores A e B controlam mutuamente uns aos outros e nenhum deles pode assumir o controle se o outro falhar. Deste forma, o serviço só ficará indisponível se ambos falharem ao mesmo tempo. Quanto maior o número de roteadores monitorando uns aos outros, maior será a disponibilidade caso algum roteador falhe fisicamente, ou ocorra algum problema com seu *software* que o deixe inoperante.

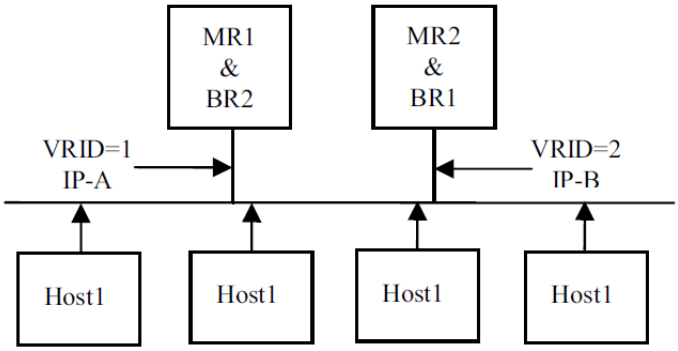

Figura 11 – RGP por Backup mútuo (SHINN, 2009).

Em alguns casos, pode ser útil associar o comportamento de vários grupos virtuais de roteadores. Com o rastreamento de porta isto é possível. O protocolo RGP o implementa através de dois mecanismos: Ele pode lançar um *script* depois de uma transição de estado e obriga as transições de estado de recepção do sinal (`SIGUSR1`) ser o mestre e (`SIGUSR2`) ser *backup*. Por exemplo, se a interface `eth1` do `R1` falhar, o mecanismo muda o seu estado de *backup* e pode lançar um *script*. O *script* envia uma `SIGUSR2` ao roteador e o obriga a passar para o estado de *backup*. Por exemplo, na Figura 12 dois *firewalls* estão encaminhando pacotes a partir de um enlace 0 para um enlace 1 e vice-versa. A `eth0` e `eth1` estão conectadas: se `eth1` de `R1` falhar, ele deve parar de aceitar pacotes de `eth0`, e vice-versa.

O autor concluiu que RGP é adequado em casos de falha e tentativa de reconexão

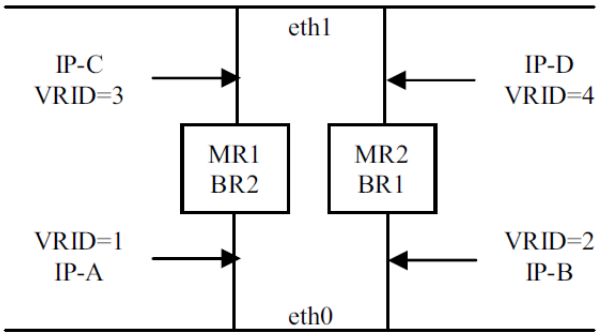

Figura 12 – RGP por rastreamento de portas(SHINN, 2009).

automática com o servidor do cliente: por exemplo roteadores IP com pacotes IP podem ser perdidos pela rede. Ainda para Shinn (2009) entre as funções auxiliares de um roteador IP, alta disponibilidade é a função mais necessária. Como estes dispositivos de acesso à rede de borda (roteadores) são implantados no caminho crítico entre um site local do usuário e seu provedor de serviços de Internet de alta disponibilidade (tolerância a falhas) é crucial para a sua concepção, a conectividade ininterrupta do site local do usuário à internet. Quando testado um desligamento de um roteador virtual, foi visto que o restante dos roteadores poderia responder momentaneamente. Para o autor, o sistema é satisfatório para os ambientes e situações testadas, mas não é perfeito para ser utilizado comercialmente.

Já no trabalho proposto pelo autor Wu et al. (2007), o mesmo define que servidores SIP Proxy são importantes blocos de construção de uma rede SIP, podem ser desenhados em forma de um *cluster* (FERREIRA; SANTOS; ANTUNES, 2005) para evitar indisponibilidade de serviços causados por falhas de software ou hardware. O projeto de um *cluster* de servidor SIP proxy com um único atendente em uma rede SIP enfrenta o problema de um único ponto de falha. No trabalho do autor Wu et al. (2007) é proposta a utilização de uma arquitetura composta por dois servidores SIP utilizando o mecanismo de detecção de falhas VRRP.

Propõe-se um esquema de rápida detecção de falhas e recuperação (*Fast Failure detection and Failover* - FFF). O *cluster* de servidores SIP Proxy é gerenciado pelos controladores de trabalho centralizados, despachantes SIP. A Figura 13 mostra a arquitetura de FFF por redes SIP de alta disponibilidade. Um SAP (*Service Access Point*) é um IP virtual que está associado com um despachante SIP, ativo e responsável por tratar as mensagens SIP recebidas. Com um SAP, os clientes SIP podem acessar os serviços corretamente sem conhecer o projeto de alta disponibilidade no lado do servidor. Clientes em diferentes regiões (por exemplo, as regiões A e B na Figura 13) pode acessar respectivos SAP para maior escalabilidade.

Após o estudo de alguns trabalhos relacionados às ferramentas possíveis a serem utilizadas, no capítulo 3 foram definidos os protocolos e mecanismos escolhidos para

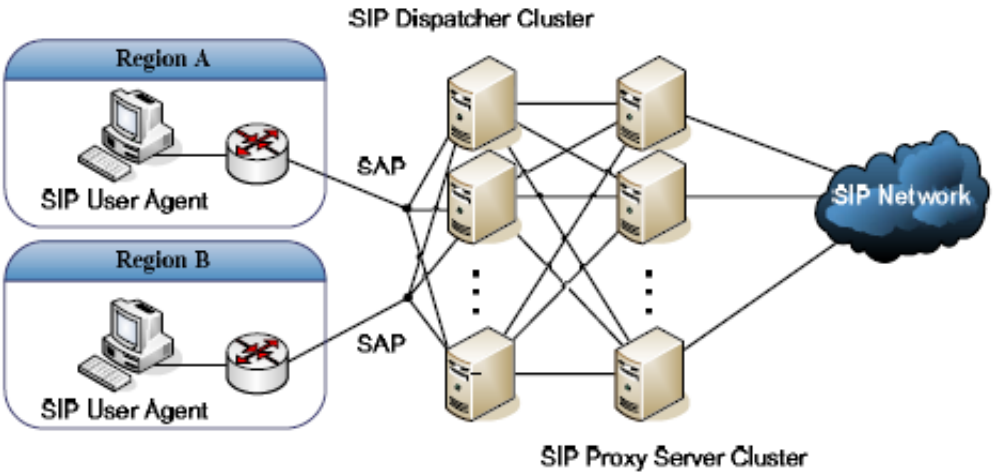

Figura 13 – Projeto de arquitetura do sistema FFF proposto para redes SIP de alta disponibilidade. FOnte: Wu et al. (2007).

implementação deste projeto.

3 Desenvolvimento da Proposta

A proposta deste trabalho consiste na implantação de alta disponibilidade para um PABX IP. O modo de alta disponibilidade que se estou e implantou no desenvolvimento do trabalho foi do tipo ativo-passivo, devido à utilização combinada do protocolo VRRP/-CARP e um SGDB, os quais se adequam à aplicação do protocolo SIP empregado neste trabalho, o Asterisk. A escolha desta modalidade se deu devido a utilização do protocolo VRRP/CARP juntamente com um SGBD - estudados na Seção 2.2.

A implantação de alta disponibilidade ocorre mediante ao chaveamento entre servidores primário e secundário, quando o servidor principal apresentar alguma indisponibilidade, que através de uma ferramenta de monitoramento, deve fazer a convergência para o servidor secundário. Desta forma, é possível garantir que os clientes registrados anteriormente no servidor principal se autentiquem no servidor secundário. Este processo é possível pois a manutenção da sessão SIP ocorre de forma regular; ou seja, os terminais reenviam registros SIP regularmente (com intervalo configurável). Assim, após detecção de perda de registro, o terminal reenvia o pedido de registro, autenticando assim no servidor secundário de maneira transparente (mesmo endereço de registro). Cabe reforçar que o objetivo deste trabalho é garantir apenas o registro dos UACs, sem, portanto, atender as sessões de mídia - o que implicaria analisar os tempos de chaveamento entre servidores e os tempos de encerramento de chamada por falta de transmissão dos pacotes de mídia -, o que foge ao escopo original.

Para realizar esta implantação de alta disponibilidade, foi necessário realizar a implementação de um mecanismo de chaveamento entre os servidores principal e secundário. Com base nos objetivos citados na Seção 1.2, foram realizados testes e análises dos mecanismos VRRP (Seção 2.3) - que ficou responsável pela virtualização do IP e monitoramento dos processos vitais dos servidores -, Rsync (Seção 2.4.1) - responsável pela sincronização bidirecional dos arquivos de configuração dos dois servidores - e o Galera Cluster (Seção 2.4.3) - que se encarregou do cluster de banco de dados, garantindo a integridade dos arquivos de configuração necessários da duas centrais telefônicas (Asterisk).

Com base nos estudos realizados e nas ferramentas citadas acima, o conjunto de mecanismos - aplicados ao Asterisk -, executam uma ação de contingência, que é acionada mediante alguma mudança operacional, mapeada no descritivo deste trabalho. Esta ação consiste na ativação de um segundo Asterisk (central espelho) assumindo o papel de servidor principal. A central espelho é idêntica a central primária, pois possui configurações de hardware e software idênticas a mesma.

O mecanismo de monitoramento citado (VRRP) deve monitorar o estado dos dois

servidores Asterisk (primário e secundário). Este estado é definido como *online* ou *offline*, seguindo os requisitos estabelecidos abaixo:

O estado da central é considerado *online* caso atenda a **todos** os de requisitos abaixo:

- Central devidamente energizada e ligada;
- Central conectada a rede, respondendo a requisições IP;
- Serviço de gerenciamento de chamadas funcionando adequadamente.

Por outro lado, o estado da central é considerado *offline*, caso atenda a **pelo menos um** dos requisitos abaixo:

- Central com o cabo de alimentação desconectado ou desligada;
- Cabo de rede desconectado, não respondendo a solicitações IP;
- Problemas de *hardware* (fonte de alimentação, placa de rede, disco rígido, processador e/ou memória principal;
- Indisponibilidade do serviço de gerenciamento de chamadas da central.

Em caso de detecção de servidor *offline* o mecanismo deve informar o servidor secundário imediatamente para que este então assuma o controle, que antes estava com o servidor primário. Até que o problema no servidor primário seja resolvido, o controle fica com o servidor secundário. Porém, assim que o problema for resolvido no servidor principal, o mecanismo deve lhe repassar o controle. Inicialmente as sessões de mídia que estão em andamento no momento da troca de controle dos servidores são ignoradas quanto ao seu estado. Isto é, as ligações em andamento caem e não recebem qualquer tratativa. Esta decisão foi tomada devido a complexidade em manter os estados das sessões replicados. Dentre as ferramentas de sincronismo existentes, convencionou-se utilizar Rsync com Galera Cluster discutidos na Seção 2.4 para realizar o sincronismo do banco de dados das centrais primária e secundária.

Para que os ramais antes registrados na central primária se autentiquem na central secundária, em caso falha no servidor primário, é necessário que o banco de dados das duas centrais estejam sincronizados. Este sincronismo será realizado com o auxílio do protocolo RSYNC operando em conjunto com o Galera Cluster.

O cenário proposto pode ser visualizado na figura 14.

Para que o trabalho fosse realizado, o IFSC câmpus São José disponibilizou máquinas e virtuais e infraestrutura de rede para o desenvolvimento do projeto.

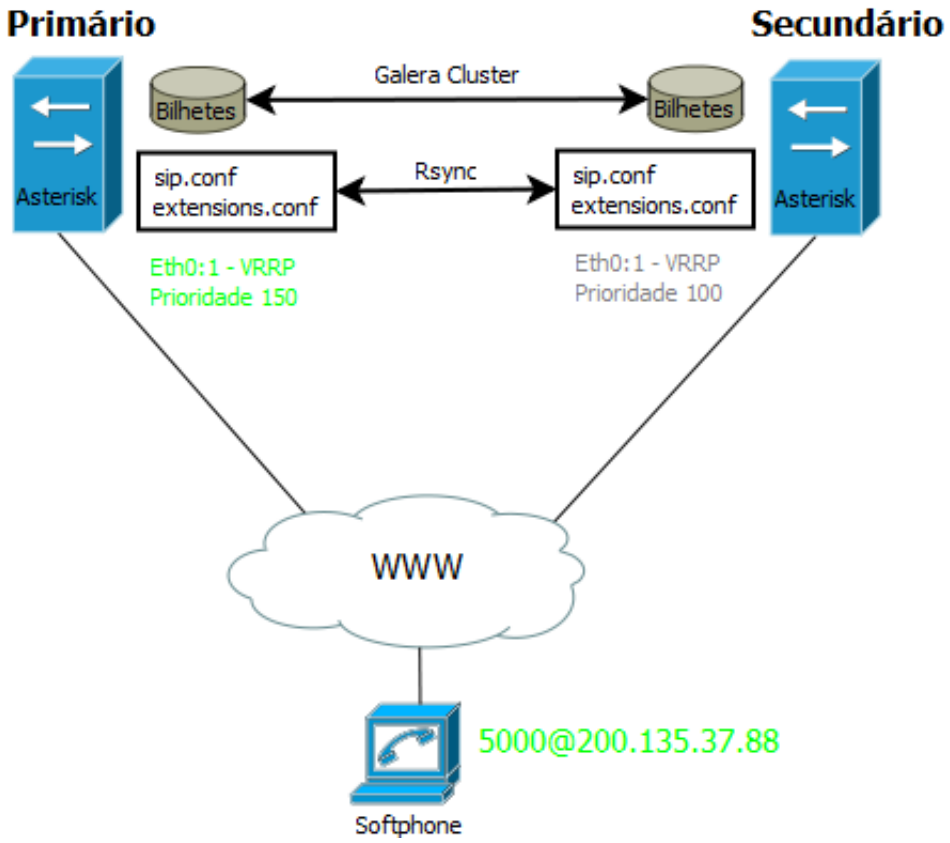

Figura 14 – Cenário proposto para implantação de alta disponibilidade. Fonte: Próprio autor.

A implementação com o passo a passo de configurações e demais orientações está descrita no apêndice A.

Na Seção a seguir serão apresentados os componentes implementados para a implantação de alta disponibilidade para um PABX IP.

3.1 Sincronização de arquivos de configuração do Asterisk

Para que fosse possível realizar a sincronização dos arquivos de configuração do Asterisk (sip.conf e extensions.conf) dos dois PABX envolvidos , o mecanismo de sincronismo de arquivos Csync2 (2.4.1) foi utilizado. O Csync2 é responsável por sincronizar os arquivos de forma bidirecional, ou seja, tanto escreve quanto apaga nos arquivos independente da direção (do primário para o secundário ou vice-versa) realizando uma comparação fiel entre os arquivos.

As configurações e implementação do Csync2 estão disponíveis no apêndice na Seção A.3.

A figura 15 ilustra o funcionamento desta ferramenta aplicada ao cenário proposto.

Em seguida será apresentado a solução para realização do sincronismo dos arquivos

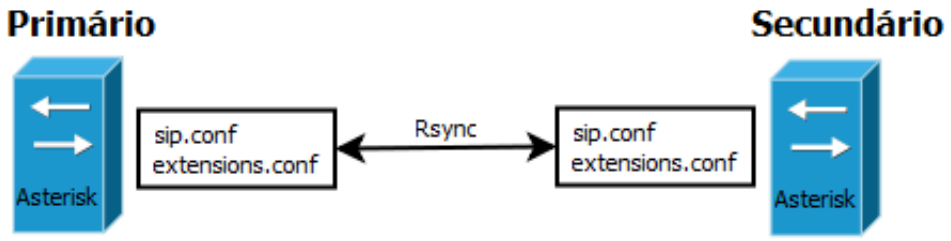

Figura 15 – Aplicação do Csync2 no cenário proposto. Fonte: Próprio autor.

de bilhetagem do Asterisk.

3.2 Sincronismo dos arquivos de bilhetagem

Os dados de bilhetagem do Asterisk ficam armazenados no arquivo CDR. Para garantir a integridade destas informações, foi utilizado um sistema gerenciador de banco de dados, o MySQL, responsável por armazenar e gerenciar tais informações. Para que houvesse o sincronismo destas informações foi utilizado o Galera Cluster. O Galera por sua vez cria um*cluster* de dados, onde todos os nós do *cluster* compartilham a mesma informação.

Para esta configuração, inicialmente foi necessário criar uma base de dados para a bilhetagem. Após a criação da Base, foi necessário criar as tabelas, utilizando a Base criada. Com os arquivos de bilhetagem sendo gravados diretamente no MySQL, bastou configurar o nó para compartilhar esta base de dados.

As configurações e orientações quanto a instalação do Galera Cluster podem ser vistas na Seção A.5.

3.3 Compartilhamento do IP virtual

Para realizar a virtualização do IP compartilhado entre os servidores principal e secundário, foi aplicado ao trabalho o protocolo de redundância VRRP. Conforme a Seção 2.3.1, o protocolo VRRP trata-se de um protocolo de eleição, onde o "roteador"com maior prioridade é detentor do IP virtual definido.

Para que fosse possível pôr em prática a definição do VRRP foi necessário realizar a instalação e configuração do mesmo, disponível na Seção A.2 de forma clara e detalhada. Três parâmetros principais foram definidos nos arquivos de configuração do VRRP aplicado no trabalho:

1. Prioridade de cada servidor;
2. ID do grupo VRRP (VRID);

3. IP virtual compartilhado entre os servidores;

Após estas definições e a inicialização dos serviços responsáveis pelo VRRP, o servidor primário assume então o IP virtual, por possuir o maior número de prioridade.

Na Figura 16 é possível visualizar os parâmetros que foram definidos nos arquivos de configuração do VRRP, além das mensagem regulares (para controle), chamadas LSA. Estas mensagens são enviadas regularmente do servidor primário para o secundário. Em caso de ausência da mesma pelo tempo de 1 minuto, o servidor secundário interpreta que existe uma falha no servidor primario e assume o controle do IP virtual (pois é o próximo servidor com número de prioridade maior definido no grupo VRRP).

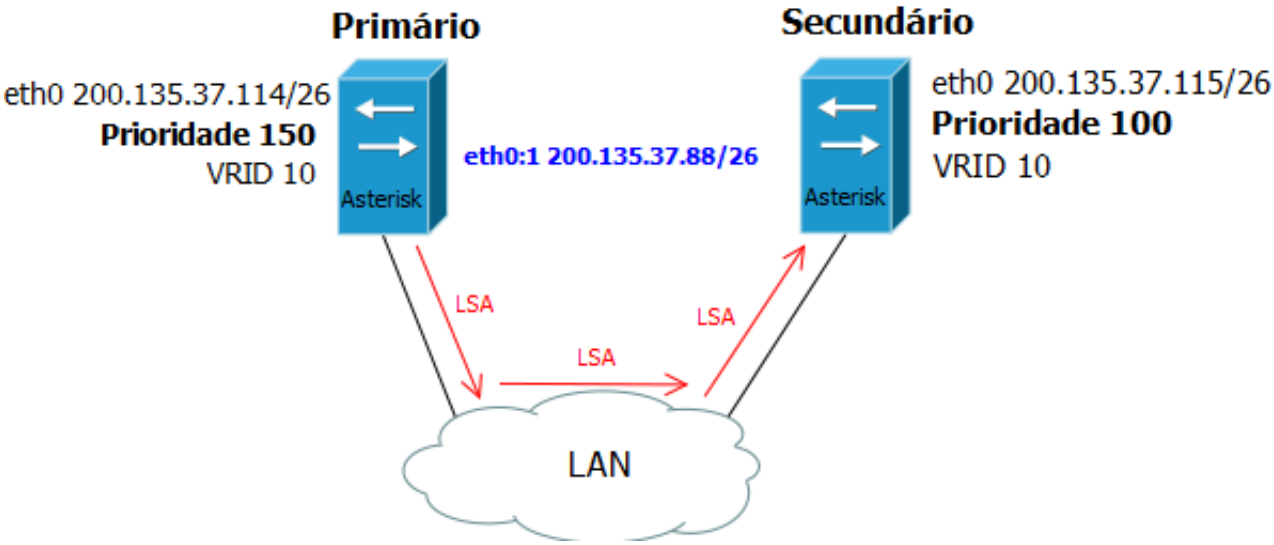

Figura 16 – Aplicação do VRRP no cenário proposto. Fonte: elaborado pelo autor.

Para que o chaveamento dos servidores ocorresse em caso de falhas que não fossem provenientes diretamente do VRRP, foi necessário implementar um monitor de "processos". Veja a Seção 3.4 a seguir.

3.4 Monitoramento dos servidores

Para realização do monitoramento dos processos nos servidores primário e secundário, foi utilizado o próprio protocolo VRRP, visto na Seção 3.3. Este por sua vez possui em seus arquivos de configuração, a opção de execução de um *script* para verificar o estado de determinado processo.

No trecho abaixo é possível verificar o *script* utilizado para validação do processo responsável pelo Asterisk. Em caso de resultado diferente do esperado, o VRRP então toma uma ação de contingência de forma imediata, liberando o IP virtual antes em posse do servidor primário para o servidor secundário.

```
vrrp_script chk_asterisk {
    script "/usr/bin/killall -0 asterisk"
    interval 2
    weight 0
}
```

As configurações completas podem ser vistas na Seção A.2.

Após a realização de configuração dos complementos que compõem o trabalho, foram realizados os testes para validação da solução proposta. Veja a Seção 3.5 a seguir.

3.5 Testes de Alteração de Cenário

Após a instalação de todas as ferramentas necessárias para a aplicação do trabalho, foram iniciados os testes para validação de Alta disponibilidade do cenário proposto.

1. Registro de ramal SIP em cada servidor isoladamente: O primeiro teste foi o registro de um ramal SIP em cada servidor de forma isolada. Para realizar o teste de registro do ramal SIP no Asterisk do servidor primário, foi utilizado um *softphone*. Após configurar o softphone para registro no servidor primário, o mesmo se registrou conforme o esperado. A mesma situação ocorreu com o registro deste mesmo ramal no servidor secundário.

2. Compartilhamento do IP virtual utilizando o VRRP: Para realizar este teste, os dois servidores envolvidos, primário e secundário, tiveram os serviços do VRRP inicializados, desta forma - conforme a configuração apresentada no apêndice A.2 - o servidor primário assumiu o IP virtual configurado no cenário. Para validar a troca do IP em caso de falha no servidor primário, foram realizadas as simulações abaixo:

   - Parada do serviço VRRP no servidor primário: o servidor secundário assumiu o endereço IP virtual com tempo inferior à 1s.
   - Desligamento do servidor primário: o servidor secundário assumiu o endereço IP virtual com tempo inferior à 1s.

   Para validação da convergência do IP compartilhado entre os servidores primário e secundário, foi dado o comando abaixo:

   ```
   ~# ifconfig
   ```

   Desta forma foi possível observar a mudança do IP virtual passando do servidor primário para o secundário.

Na Figura 17 é possível observar de forma ilustrativa os teste realizados: O Serviço VRRP é parado de forma forçada e automaticamente as mensagens LSA enviadas do servidor primário para o servidor secundário não são mais enviadas. Com isto, o servidor secundário assume o IP virtual.

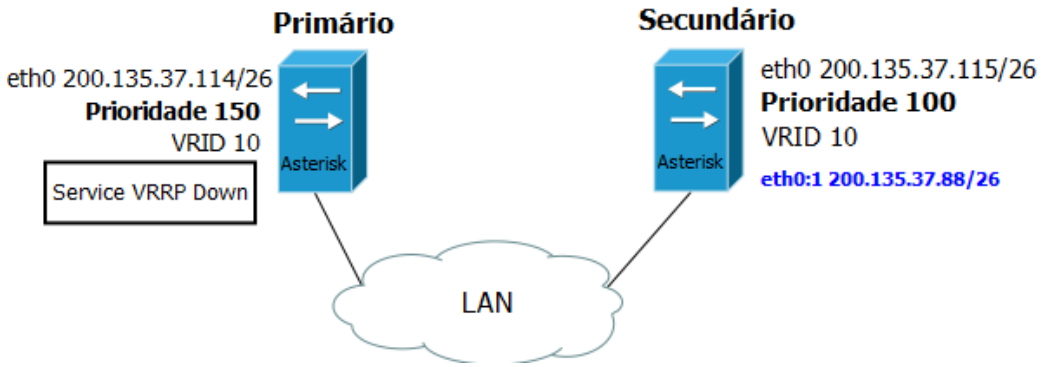

Figura 17 – Simulação de falha do VRRP. Fonte: Elaborado pelo autor.

Conforme apresentado no apêndice A, o VRRP foi devidamente configurado para realizar também o monitoramento do Asterisk. Como descrito anteriormente, o VRRP foi o mecanismo empregado para monitoramento dos servidores primário e secundário pois dispõe de tal característica.

3. Sincronismo dos arquivos de configuração do Asterisk: Conforme descrito na proposta do trabalho, a ferramenta de sincronismo utilizada foi o Csyn2. Após a criação dos arquivos de configuração dos Asterisk, os mesmos foram sincronizados através do CSYNC2. Para validar a replicação das alterações nos dois servidores foram realizados dois testes:

   - Adição de novas linhas nos arquivos de configuração do servidor primário: Para isto foi utilizado um editor de arquivos e criadas duas novas linhas no arquivo de configuração SIP.conf, o Csync2 se comportou como o esperado, replicando a alteração realizada para o servidor secundário.
   - Exclusão de informações nos arquivos de configuração: o Csync2 replicou a exclusão das informações para o servidor secundário, comprovando assim seu funcionamento bidirecional.

   Para que o Asterisk do servidor secundário realizasse a releitura do arquivo SIP.conf alterado, foi utilizado o comando abaixo:

   ```
   ~# asterisk -rx "sip reload"
   ```

4. Verificação do *cluster* do banco de dados: Após as configurações do *cluster* vistas na Seção A.5, foram realizados testes para validação do estado do nó, através do comando abaixo:

```
mysql -u root -p -e 'SELECT VARIABLE_VALUE as "cluster size"
FROM INFORMATION_SCHEMA.GLOBAL_STATUS
WHERE VARIABLE_NAME="wsrep_cluster_size"'
```

O banco retorna o numero de nós que existem no Cluster, neste caso foi igual a 2, confirmando que o teste foi executado e obteve êxito ao final.

```
+--------------+
| cluster size |
+--------------+
| 2            |
+--------------+
```

Outro teste realizado foi de conexão entre o Asterisk e a base MySQL. Para realização deste teste foi necessário conectar ao MYSQL e dar o comando abaixo:

```
SHOW PROCESS LIST;
```

Desta forma, todas as conexões com o Banco de Dados foram apresentadas, inclusive a conexão do Asterisk com sua Base de dados.

Após a finalização de todos os testes pontuais - utilizando cada ferramenta de maneira isolada - , foram realizados testes funcionais, para validação do trabalho como um todo, ou seja, utilizando todas as ferramentas.

Testes realizados:

1. Estado 0: Servidores primário e secundário em pleno funcionamento, com todos os processos vitais rodando.

   No estado 0, foi considerado o "cenário inicial", com todos os componentes necessários para o pleno funcionamento do trabalho iniciados sem qualquer falha.

   - VRRP rodando: IP virtual compartilhado em posse do servidor primário, conforme definido nas configurações A.2;
   - Asterisk: Rodando perfeitamente;
   - RSYNC: Rodando, com os arquivos sincronizados;
   - Galera Cluster: *Cluster* rodando, funcionando perfeitamente.

   Inicialmente foi registrado um ramal SIP apontando para o IP virtual compartilhado entre os servidores, que o servidor primário assumiu no estado 0. O registro ocorreu sem problemas, conforme é possível ver na Figura 18.

```
<--- SIP read from UDP:189.40.89.225:50104 --->
REGISTER sip:200.135.37.88:10985;transport=UDP SIP/2.0
Via: SIP/2.0/UDP 100.125.155.220:61716;branch=z9hG4bK-524287-1---7cfe71d83202b7fc
Max-Forwards: 70
Contact:
<sip:5000@100.125.155.220:61716;rinstance=d4cce9e710802320;transport=UDP>;expires=0
To: "5000"<sip:5000@200.135.37.88:10985;transport=UDP>
From: "5000"<sip:5000@200.135.37.88:10985;transport=UDP>;tag=1d003319
Call-ID: opF7K_vACFckyhgJihzujQ..
CSeq: 5 REGISTER
User-Agent: Zoiper rv2.8.30
Authorization: Digest
username="5000",realm="asterisk",nonce="3ee801b5",uri="sip:200.135.37.88:10985;trans
port=UDP",response="06edc9a8b7a5f6e732cbdad64e8aa25b",algorithm=MD5
Allow-Events: presence, kpml, talk
Content-Length: 0

<------------->
--- (12 headers 0 lines) ---
Sending to 189.40.89.225:50104 (NAT)

<--- Transmitting (NAT) to 189.40.89.225:50104 --->
SIP/2.0 200 OK
Via: SIP/2.0/UDP 100.125.155.220:61716;branch=z9hG4bK-524287-1---
7cfe71d83202b7fc;received=189.40.89.225;rport=50104
From: "5000"<sip:5000@200.135.37.88:10985;transport=UDP>;tag=1d003319
To: "5000"<sip:5000@200.135.37.88:10985;transport=UDP>;tag=as50a76f43
Call-ID: opF7K_vACFckyhgJihzujQ..
CSeq: 5 REGISTER
Server: Asterisk PBX 11.13.1~dfsg-2+b1
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH,
MESSAGE
Supported: replaces, timer
Expires: 0
Date: Mon, 27 Mar 2017 00:37:43 GMT
Content-Length: 0
```

Figura 18 – Solicitação de Registro do ramal ao servidor primário. Fonte: Elaborado pelo autor.

Na Figura 18 é possível observar que o ramal solicita o pedido de registro através da mensagem SIP REGISTER e em seguida é confirmado com a mensagem 200 OK vinda do servidor primário.

2. Estado 1: Simulação de queda na rede (derrubando o serviço do VRRP `keepalived`).

   Com o ramal SIP devidamente registrado no servidor primário, foi simulada uma falha no processo responsável pelo VRRP, o serviço keepalived.

   Nas Figuras 19 e 20 é possível visualizar o estado de registro do ramal nos servidores primário e secundário respectivamente, antes da simulação de falha.

```
maicky-0*CLI> sip show peers
Name/username             Host                                    Dyn Forcerport Comedia    ACL Port     Status      Description
5000/5000                 179.205.248.25                           D  Yes        Yes           61716    Unmonitored
5001/5001                 (Unspecified)                            D  Yes        Yes           0        Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 1 online, 1 offline]
```

Figura 19 – Estado do registro do ramal no servidor primário. Fonte: Elaborado pelo autor.

```
maicky-1*CLI> sip show peers
Name/username             Host                                    Dyn Forcerport Comedia    ACL Port     Status      Description
5000/5000                 (Unspecified)                            D  Yes        Yes           0        Unmonitored
5001/5001                 (Unspecified)                            D  Yes        Yes           0        Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 0 online, 2 offline]
```

Figura 20 – Estado do registro do ramal no servidor secundário. Fonte: Elaborado pelo autor.

É possível observar que o ramal 5000 está devidamente registrado no servidor primário, enquanto o servidor secundário não possui nenhum registro.

No status de rede dos servidores é possível observar que o IP virtual compartilhado entre os servidores pertence ao servidor primário, veja a Figura 21:

```
eth0      Link encap:Ethernet  Endereço de HW 00:0c:29:c0:96:c1
          inet end.: 200.135.37.114  Bcast:200.135.37.127  Masc:255.255.255.192
          endereço inet6: fe80::20c:29ff:fec0:96c1/64 Escopo:Link
          UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1
          RX packets:14512836 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11884455 errors:0 dropped:0 overruns:0 carrier:0
          colisões:0 txqueuelen:1000
          RX bytes:1533856098 (1.4 GiB)  TX bytes:1635936192 (1.5 GiB)

eth0:1    Link encap:Ethernet  Endereço de HW 00:0c:29:c0:96:c1
          inet end.: 200.135.37.88  Bcast:0.0.0.0  Masc:255.255.255.192
          UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1

lo        Link encap:Loopback Local
          inet end.: 127.0.0.1  Masc:255.0.0.0
          endereço inet6: ::1/128 Escopo:Máquina
          UP LOOPBACKRUNNING  MTU:65536  Métrica:1
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
          colisões:0 txqueuelen:0
          RX bytes:1884 (1.8 KiB)  TX bytes:1884 (1.8 KiB)
```

Figura 21 – IP virtual em posse do servidor primário. Fonte: Próprio autor.

Para simular a falha foi utilizado o comando abaixo:

```
/etc/init.d/keepalived stop
```

Desta forma, imediatamente o IP virtual que antes estava em posse do servidor primário, passa a pertencer ao servidor secundário, veja abaixo:

```
eth0      Link encap:Ethernet  Endereço de HW 00:0c:29:c1:f7:39
          inet end.: 200.135.37.115  Bcast:200.135.37.127  Masc:255.255.255.192
          endereço inet6: fe80::20c:29ff:fec1:f739/64 Escopo:Link
          UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1
          RX packets:12973420 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3711527 errors:0 dropped:0 overruns:0 carrier:0
          colisões:0 txqueuelen:1000
          RX bytes:1082098812 (1.0 GiB)  TX bytes:623079222 (594.2 MiB)

eth0:1    Link encap:Ethernet  Endereço de HW 00:0c:29:c1:f7:39
          inet end.: 200.135.37.88  Bcast:0.0.0.0  Masc:255.255.255.192
          UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1

lo        Link encap:Loopback Local
          inet end.: 127.0.0.1  Masc:255.0.0.0
          endereço inet6: ::1/128 Escopo:Máquina
          UP LOOPBACKRUNNING  MTU:65536  Métrica:1
          RX packets:99402 errors:0 dropped:0 overruns:0 frame:0
          TX packets:99402 errors:0 dropped:0 overruns:0 carrier:0
          colisões:0 txqueuelen:0
          RX bytes:8747300 (8.3 MiB)  TX bytes:8747300 (8.3 MiB)
```

Figura 22 – IP virtual em posse do servidor secundário. Fonte: Próprio autor.

Neste momento, o ramal 5000 antes registrado no servidor primário, solicita o registro novamente ao IP configurado para registro e se registra no servidor secundário. Este é o processo de Re-registro do ramal, com tempo de solicitação configurável. Devido à esta característica, o tempo para que o ramal se registre no servidor pode chegar a 60s. Caso a solicitação de registro seja manual, imediatamente o ramal se registrará.

Após o retorno do serviço keepalived, o IP virtual voltou ao servidor primário. Com este retorno, o ramal reenviou o registro e ficou registrado no servidor primário novamente.

3. Estado 2: Simulação de queda do serviço do Asterisk.

   Para simular a falha no Asterisk, tomou-se como premissa que o ramal SIP estivesse registrado no servidor primário. Considerando o cenário neste estado inicial, foi utilizado o comando abaixo para parar o serviço do Asterisk no servidor primário:

   ```
   /etc/init.d/asterisk stop
   ```

   Após a parada do serviço, imediatamente o servidor secundário assumiu o IP Virtual e 24s depois, o ramal antes registrado no servidor primário, passa a se registrar no servidor secundário, através do envio da mensagem `SIP REGISTER`, com a confirmação do servidor secundário através da mensagem `SIP 200 OK`.

   O comportamento deste teste se deu idêntico à parada do serviço VRRP, representado no Estado 1.

   Com os três testes funcionais acima, foi possível avaliar o funcionamento do desenvolvimento como um todo do trabalho e chegar aos resultados finais, conforme o capítulo a seguir.

4 Resultados e Considerações Finais

Com a realização dos testes isolados e testes funcionais, foi possível observar os resultados de cada um e os avaliar.

De maneira geral, os resultados dos testes isolados foram positivos em se tratando do funcionamento de cada ferramenta/mecanismo de forma isolada. O comportamento de cada ferramenta se deu conforme o estudado, sem qualquer surpresa.

Para os testes funcionais os resultados se dividiram conforme cada cenário montado:

- Cenário ideal: ou estado zero, para se ter como parâmetro para os estados 1 e 2 testados a seguir. O resultado do teste se deu conforme o esperado, o ramal SIP se registrou perfeitamente em cada servidor bem como o funcionamento de cada ferramenta/mecanismo empregado.
- Falha de rede: através da parada do serviço `keepalived` (protocolo VRRP). O resultado foi positivo, após a parada do serviço, a convergência do compartilhamento do IP ocorreu com menos 1s, mostrando assim que o servidor secundário estava disponível rapidamente para assumir o registro do(s) ramal(is) SIP. Logo após a convergência de IP, o registro do terminal também ocorreu e forma positiva, com um tempo hábil conforme proposto no trabalho.
- Falha no Asterisk: através da parada do serviço Asterisk no servidor primário. O resultado neste teste também foi positivo, o comportamento se deu idêntico ao teste realizado no Estado 1. Após a parada do serviço, com um tempo inferior a 1s, o servidor secundário já se encontrava em posse do IP virtual e pronto para receber o registro dos terminais SIP.

Para validação dos testes foi realizada uma quantidade significativa de testes, um número de aproximadamente 30 testes por cenário proposto. Esta quantidade de testes foi necessária para obtenção de valores estaticamente significativos.

Para futuras melhorias/implementações deste trabalho, seria interessante estudar a possibilidade de manter a mesma sessão do ramal SIP já registrado no servidor primário, no servidor secundário. Para isto seria necessário buscar a informação deste registro dentro do Asterisk e manter a sessão ativa no servidor secundário. Desta forma, a nível de usuário, o registro dos ramais SIP no servidor secundário ficaria de maneira transparente, ou seja, não haveria a necessidade de aguardar o tempo de Re-registro do ramal SIP, conforme citado nos testes. Outra possível melhoria seria a manutenção da mídia na sessão SIP, caso o(s) ramal(is) SIP esteja(m) em conversação, desta forma o tráfego de áudio passaria

do servidor primário imediatamente para o servidor secundário, garantindo assim que não ocorra ausência dos pacotes de áudio durante a conversação e consequentemente queda na ligação.

Conclui-se que o objetivo proposto neste trabalho foi atingido, os ramais SIP tiveram seu registro garantido quando o servidor primário apresentou falha, tendo em vista os cenários aplicados e testes realizados. Foi possível validar o funcionamento da ferramenta responsável pelo monitoramento do estado dos servidores, que mediante à alguma falha, realizou o chaveamento do IP compartilhado entre os servidores primário e secundário. Por fim, os resultados mostraram que baseado em uma solução simples (plataforma aberta) e baixo custo, é possível empregar alta disponibilidade para um PABX IP.

Referências

DANTAS, J. R. Cluster de alta disponibilidade com arquitetura heartbeat, um projeto linux-ha para software livres. 2009. Citado na página 23.

EGEVANG, K.; FRANCIS, P. *The IP network address translator (NAT)*. [S.l.], 1994. Citado na página 32.

FERREIRA, F.; SANTOS, N.; ANTUNES, M. Clusters de alta disponibilidade–uma abordagem open source. *Relatorio desenvolvido na disciplina de Projecto I*, 2005. Citado 4 vezes nas páginas 13, 23, 24 e 35.

FILHO, H. B. Telefonia ip. 2003. Citado 2 vezes nas páginas 13 e 22.

GALERA Cluster. 2016. Disponível em: <http://galeracluster.com/documentation-webpages/>. Citado 3 vezes nas páginas 13, 31 e 32.

GUINDANI, A. Gestão da continuidade dos negócios. 2008. Citado na página 23.

HASHIMOTO, G. T. *Uma proposta de extensao para um protocolo para arquiteturas de alta disponibilidade*. Tese (Doutorado) — Universidade Federal de Uberlandia, 2009. Citado 4 vezes nas páginas 13, 24, 26 e 27.

HINDEN, R. Virtual router redundancy protocol (vrrp). 2004. Citado na página 26.

HINDEN, R. M. *Virtual Router Redundancy Protocol (VRRP)*. RFC Editor, 2015. RFC 3768. (Request for Comments, 3768). Disponível em: <https://rfc-editor.org/rfc/rfc3768.txt>. Citado na página 26.

JUNIOR, P. J. P. *VoIP I: Telefonia Convencional*. 2014. Disponível em: <http://www.teleco.com.br/tutoriais/tutorialvoipcp1/pagina_2.asp>. Citado na página 17.

KNIGHT, S. et al. *Virtual Router Redundancy Protocol*. [S.l.], 1998. Citado 2 vezes nas páginas 32 e 33.

LI, T. et al. *Cisco Hot Standby Router Protocol (HSRP)*. [S.l.], 1998. <http://www.rfc-editor.org/rfc/rfc2281.txt>. Disponível em: <http://www.rfc-editor.org/rfc/rfc2281.txt>. Citado 2 vezes nas páginas 28 e 32.

LINBIT. *drbd*. 2016. Disponível em: <http://www.drbd.org/home/>. Citado 3 vezes nas páginas 13, 29 e 30.

OpenBSD. *PF: Firewall Redundancy with CARP and pfsync*. 2017. Disponível em: <http://www.openbsd.org/faq/pf/carp.html>. Citado na página 28.

RIBEIRO, G. da S. *Voz sobre IP II: A Convergência de Dados e Voz*. 2011. Disponível em: <http://www.teleco.com.br/tutoriais/tutorialvoipconv2/pagina_4.asp>. Citado na página 22.

ROBERTSON, A. The evolution of the linux-ha project. In: *UKUUG LISA/Winter Conference High-Availability and Reliability*. [S.l.: s.n.], 2004. Citado na página 25.

ROCHA, W. et al. Servidor voip utilizando asterisk. *Pará, January*, 2013. Citado na página 23.

ROSA, R. H. da. Ferramenta para desenvolvimento de planos de discagem no asterisk. 2007. Citado na página 23.

RSYNC. 2015. Disponível em: <http://rsync.samba.org/ftp/rsync/rsync.html>. Citado na página 29.

SATO, A. M. PABX IP. 2004. Citado na página 22.

SERVIDOR Samba. 2013. Disponível em: <http://www.samba.org/>. Citado na página 25.

SHINN, S. K. Fault tolerance virtual router for linux virtual server. In: IEEE. *Future Networks, 2009 International Conference on*. [S.l.], 2009. Citado 5 vezes nas páginas 13, 32, 33, 34 e 35.

Sá, D. de; HIURI, V. Cluster de alta disponibilidade em linux. Faculdade Politec, Santa Bárbara D'Oeste, 2012. Citado 4 vezes nas páginas 13, 25, 29 e 30.

WU, W.-M. et al. A fast failure detection and failover scheme for sip high availability networks. In: IEEE. *Dependable Computing, 2007. PRDC 2007. 13th Pacific Rim International Symposium on*. [S.l.], 2007. p. 187–190. Citado 3 vezes nas páginas 13, 35 e 36.

Apêndices

APÊNDICE A – Configuração do Cenário

Para desenvolvimento do trabalho, foi necessário realizar as configurações em quatro etapas:

1. Instalação do Asterisk e configurações básicas.
2. Instalação e configuração do VRRP.
3. Instalação e configuração do Rsync.
4. Instalação e configuração do Galera.

Foi utilizada a distribuição Debian GNU/Linux, versão `jessie` (estável à época dos testes) e 64-bit.

A.1 Instalação e configuração do Asterisk

No Linux a instalação é bastante simples:

```
apt-get install asterisk
```

Após a instalação foram realizadas configurações de plano de discagem e ramais básicos, abaixo um exemplo do arquivo sip.conf (configurações dos ramais):

```
[general]
context = default
bindport = 10985
bindaddr = 0.0.0.0
srvlookup = yes
disallow = all
allow = all

[5000]
type = friend
callerid = "5000" <5000>
username = 5000
secret = 5000
host = dynamic
```

```
nat = yes
canreinvite = no
context = ramais

[5001]
type = friend
callerid = "5001" <5001>
username = 5001
secret = 5001
host = dynamic
nat = yes
canreinvite = no
context = ramais
```

A seguir, o arquivo extensions.conf (configuração das regras de discagem):

```
[ramais]

exten => 5000,1,Dial(SIP/5000,25)
exten => 5000,2,Hangup

exten => 5001,1,Dial(SIP/5001,25)
exten => 5001,2,Hangup
```

Após a instalação do Asterisk, foi realizada a instalação do VRRP para virtualização do IP dos servidores principal e secundário.

A.2 Instalação e configuração do VRRP

No Linux:

```
apt-get install vrrpd
apt-get intall keepalived
```

Após a instalação dos pacotes necessários para o devido funcionamento do VRRP, foi necessário realizar a configuração do VRRP nos dois servidores envolvidos, primário e sucundário:

Arquivo de configuração do servidor primário:

```
global_defs {
```

```
    lvs_id LVS_MAIN
}

vrrp_script chk_asterisk {
    script "/usr/bin/killall -0 asterisk"
    interval 2
    weight 0
}

vrrp_sync_group VG1 {
    group {
        VI_1
    }
}

vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 10
    priority 150
    advert_int 1
    garp_master_delay 2
    preempt_delay 20
    authentication {
        auth_type PASS
        auth_pass k@l!ve1
    }

    track_script {
        chk_asterisk
    }

    virtual_ipaddress {
        200.135.37.88/26 label eth0:1
    }
}
```

Arquivo de configuração do servidor secundário:

```
global_defs {
    lvs_id LVS_MAIN
}

vrrp_script chk_asterisk {
    script "/usr/bin/killall -0 asterisk"
    interval 2
    weight 0
}

vrrp_sync_group VG1 {
    group {
        VI_1
    }
}

vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 10
    priority 100
    advert_int 1
    garp_master_delay 2
    preempt_delay 20
    authentication {
        auth_type PASS
        auth_pass k@l!ve1
    }

    track_script {
        chk_asterisk
    }

    virtual_ipaddress {
        200.135.37.88/26 label eth0:1
    }
}
```

A.3 Instalação e configuração do Rsync/Csync2

Após a instalação e configuração do VRRP, foi realizada a instalação e configuração do Rsync, mecanismo responsável por realizar o sincronismo dos arquivos de configuração do Asterisk dos servidores primário e secundário:

A.4 Instalação e configuração de Rsync e Csync2

```
apt-get install rsync csync2
```

Após a instalação são nessesários alguns procedimento para o funcionamento do mecanismo de sincronismo. Inicialmente é necessário criar os certificados do Csync2 nas duas máquinas envolvidas, primária e secundária:

```
openssl genrsa -out /etc/csync2_ssl_key.pem 1024
openssl req -new -key /etc/csync2_ssl_key.pem \
    -out /etc/csync2_ssl_cert.csr
openssl x509 -req -days 600 -in /etc/csync2_ssl_cert.csr \
    -signkey /etc/csync2_ssl_key.pem -out /etc/csync2_ssl_cert.pem
```

Em seguida, criar a pasta csync2 em /etc/:

```
mkdir csync2/
```

Após criar a pasta, é necessário gerar uma chave em um dos servidores, em seguida copiar esta chave para o servidor secundário:

```
csync2 -k /etc/csync2/exemplo.key
```

Copiar para o servidor secundário:

```
scp /etc/csync2/exemplo.key server02:/etc/csync2
```

Agora, basta criar e editar o arquivo /etc/csync2.cfg com o seguinte conteúdo nos dois servidores:

```
group exemplo {
    host maicky-0 maicky-1;
    key /etc/csync2/exemplo.key;
    include /etc/asterisk/extensions.conf;
```

```
    include /etc/asterisk/sip.conf;
    include /etc/mysql/debian.cnf;
    auto none;
}
```

Por último, basta criar um agendamento para sincronização dos arquivos, a cada 1 minuto por exemplo dentro da crontab:

```
*/1 * * * * root /usr/sbin/csync2 -x > /dev/null 2> /dev/null
```

A.5 Instalação e configuração do Galera

Nos dois servidores, primário e secundário, foi realizada a instalação dos pacotes do Galera Cluster:

```
apt-get install galera-3 mariadb-server
```

Em seguida, realizar criar e editar o arquivo /etc/mysql/conf.d/galera.cnf com o conteúdo abaixo:

```
[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
innodb_doublewrite=1
query_cache_size=0
query_cache_type=0
bind-address=0.0.0.0
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so
wsrep_cluster_name="test_cluster"
wsrep_node_address=maicky-0
wsrep_cluster_address=gcomm://
wsrep_sst_method=rsync
```

Em seguida, parar o serviço Mysql nos dois servidores envolvidos:

```
systemctl stop mysql
```

Depois, copiar o conteúdo de /etc/mysql/debian.cnf do servidor primário para o servidor secundário, para que o dois servidores fiquem com o mesmo conteúdo.

Feito isto, executar o comando abaixo para criar e iniciar o cluster no servidor primário:

```
galera_new_cluster
```

Iniciar os serviços mysql em cada servidor do cluster:

```
systemctl start mysql
```

Após iniciar os serviços MySQL, verificar quantos nós estão ativos no cluster através do comando abaixo:

```
mysql -u root -p -e 'SELECT VARIABLE_VALUE as "cluster size"
FROM INFORMATION_SCHEMA.GLOBAL_STATUS
WHERE VARIABLE_NAME="wsrep_cluster_size"'
```

O resultado deverá ser este:

```
+--------------+
| cluster size |
+--------------+
| 2            |
+--------------+
```

A finalidade do Galera neste trabalho é garantir a integridade e compartilhamento da bilhetagem dos PABX envolvidos. Desta forma os registros serão compartilhados no nó do Cluster.

Para esta configuração, inicialmente é necessário criar uma base de dados para a bilhetagem. Após a criação da Base, é necessário criar as tabelas, utilizando a Base criada. Com os arquivos de bilhetagem sendo gravados diretamente no MySQL, basta configurar o nó para compartilhar esta base de dados.