UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
ESPECIALIZAÇÃO EM REDES DE COMPUTADORES

EDENILSON TONDO DA SILVA

SOFTWARE LIVRE NO MONITORAMENTO DE SERVIÇOS E BACKUP DE DADOS POR MEIO DE REDES DE COMPUTADORES

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO
2015

EDENILSON TONDO DA SILVA

SOFTWARE LIVRE NO MONITORAMENTO DE SERVIÇOS E BACKUP DE DADOS POR MEIO DE REDES DE COMPUTADORES

Trabalho de Conclusão de Curso, apresentado ao II Curso de Especialização em Redes de Computadores – Configuração e Gerenciamento de Servidores e Equipamentos de Redes, da Universidade Tecnológica Federal do Paraná, câmpus Pato Branco, como requisito parcial para obtenção do título de Especialista.

Orientador: Prof. Dr ou MSc. Christian Carlos Souza Mendes

PATO BRANCO
2015

TERMO DE APROVAÇÃO

Software Livre no Monitoramento de Serviços e Backup de Dados por Meio de Redes de Computadores

por

Edenilson Tondo Da Silva

Esta monografia foi apresentada às 10h00min do dia 24 de outubro de 2015, como requisito parcial para obtenção do título de ESPECIALISTA, no II Curso de Especialização em Redes de Computadores – Configuração e Gerenciamento de Servidores e Equipamentos de Redes, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. O acadêmico foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho **aprovado**.

Banca Examinadora

Prof. M.Sc. Christian Carlos Souza Mendes
Orientador / UTFPR-CT

Prof. Dr. Fábio Favarim
UTFPR-PB

Prof. Dr. Éden Ricardo Dosciatti

Prof. Dr. Fábio Favarim
Coordenador do II Curso de Especialização
em Redes de Computadores

RESUMO

SILVA, Edenilson Tondo da. Software livre no monitoramento de serviços e backup de dados por meio de redes de computadores. 64f. Monografia de Trabalho de Conclusão de Curso (II Curso de Especialização em Redes de Computadores), Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

Grandes empresas implementam soluções de monitoramento e *backup* dos dados, infelizmente, para pequenas empresas, essa prática tem um custo relativamente alto e por essa razão muitas delas não investem nas áreas de monitoramento dos serviços e cópia redundante dos dados. Para esses casos, existe a possibilidade de implementação de soluções disponibilizadas em *software* livre. Este trabalho propõe-se a apresentar os conceitos e utilização das soluções de monitoramento Zabbix, e de *backup*, Bacula, com enfoque em pequenas organizações para que possivelmente facilite e amplie a utilização dessas tecnologias.

Palavras-chave: *Backup.* Monitoramento. Zabbix. Bacula.

ABSTRACT

SILVA, Edenilson Tondo da. Free Software free software applied in monitoring services and data backup via computer networks. 64f. Monografia de Trabalho de Conclusão de Curso (II Curso de Especialização em Redes de Computadores), Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

Large companies implement monitoring solutions and backup solutions, unfortunately, for small businesses, this practice has a relatively high cost and for that reason many of them do not invest in the areas of monitoring of services and redundant copy of the data. For those cases, there is the possibility of implementing this services in open source solutions. This paper proposes to present the concepts and use of Zabbix monitoring solutions, and backup, Bacula, with a focus on small organizations to possibly help and expand the use of those technologies.

Keywords: Backup. Monitoring. Zabbix. Bacula.

Lista de Figuras

LISTA DE SIGLAS

CD-ROM	Compact Disc-Read-Only Memory
CPU	Central Processing Unit
DHCP	Dynamic Host Configuration Protocol
DVD	Digital Versatile Disc
FSF	Free Software Fundation
GPL	General Public License
HD	Hard Drive
IDE	Integrated Drive Eletronics
IETF	Internet Engineering Task Force
IPMI	Intelligent Platform Management Interface
ISO	Internacional Organization for Standardization
LTS	Long Time Support
MIB	Management Information Base
NMS	Network Management Stations
PHP	PHP: Hypertext Preprocessor
RFC	Request for Comments
SATA	Serial Advanced Technology Attachment
SGBD	Sistema de Gerenciamento de Banco de Dados
SGMP	Simple Gateway Management Protocol
SMI	Structure of Management Information
SNMP	Simple Network Management Protocol
SSD	Solid-State Drive
TCP/IP	Transmission Control Protocol / Internet Protocol
USB	Universal Serial Bus

SUMÁRIO

1 INTRODUÇÃO

Este capítulo apresenta as considerações iniciais apresentando uma visão geral do trabalho, os objetivos, a justificativa e a organização do texto.

1.1 Considerações iniciais

No século XXI, graças aos avanços da tecnologia como um todo, em especial na área de telecomunicações, tem-se observado grandes mudanças no modo como nos organizamos em sociedade. A Internet nos proporciona a possibilidade de mitigar as complicações geográficas; um documento que levaria dias, no mais rápido dos casos, horas, para chegar a um destino, pode ser recebido em segundos se enviado por meio digital, essa dinamicidade tem feito sua adesão ser cada vez maior, tanto por indivíduos quanto por instituições; há poucos anos atrás, nas décadas de 80 e 90, fotos eram impressas e guardadas, gravações, tanto de áudio ou quanto de vídeo eram armazenas em fitas; essas mídias ainda continuam sendo usadas, mas paralelamente, a comodidade e facilidade de armazená-las em formato digital tem feito muitas pessoas a optarem apenas por esta alternativa. Com essa tendência em vista e com a concentração de várias informações distintas, como fotos, gravações, documentos, etc., em um mesmo meio físico, como por exemplo, um *Hard Disk*, a importância do equipamento aumenta, devido à grande gama de informações distintas que podem estar presentes. Desse modo, colocamo-nos em uma situação paradoxal, cada vez mais conseguimos armazenar mais informações de fontes diversas e em maior quantidade em um único dispositivo, proporcionalmente, aumenta-se o dano causado na eventual inacessibilidade das informações presentes neste aparelho.

Portanto, tendo a necessidade de garantir redundância da informação para evitar prejuízos oriundos de uma perda de dados em ambientes corporativos, é necessária a implementação de uma política de *backup* por parte das empresas, sendo que, em alguns casos, a indisponibilidade aos dados pode colocar toda a credibilidade e confiabilidade da instituição em risco. Conforme abordado por Faria

(2014), a inacessibilidade aos dados pode ocorrer por falhas de *hardware*, *software*, invasões, vírus, erros humanos, desastres naturais, entre outros.

Ainda de acordo com o autor, não importa como os dados que foram perdidos (exclusões acidentais, corrupção dos dados, etc.), um *backup* funcional consiste em minimizar os impactos dessa perda, possibilitando a restauração do arquivo ou serviço no menor tempo possível e com o mínimo de defasagem em termos de alteração das informações.

Para contornar esse problema, existem várias soluções de softwares conceituados de *backup* corporativo disponibilizados por empresas privadas especialistas nesse segmento, que geralmente são utilizados por empresas de médio ou grande porte. Entretanto, pequenas empresas também necessitam e devem aplicar uma política de *backup*; para essas empresas, a contratação de um *software* completo para gestão do *backup* dos seus dados não se torna vantajosa pela relação entre custo e benefício recebida; empresas nessa situação, quando realizam *backup* dos dados, geralmente delegam essa função para um ou dois funcionários que serão os responsáveis, esses, muitas vezes não estando familiarizados com os conceitos de *backup* e retirados de seus cargos para realizarem essa tarefa, acabam cometendo erros que podem pôr em risco os dados salvos.

Para não correr esse risco, existem soluções em *software* livre, que também proporcionam um gerenciamento de *backups* de confiabilidade e eficiência satisfatória, diminuindo assim, os custos com a aquisição de licença de *softwares* proprietários e facilitando o controle, o que vai diminuir a quantidade e tempo de pessoas necessárias trabalhando nessa tarefa. Segundo a *Free Software Fundation* (FSF, 2015)*, software* livre é qualquer programa de computador que pode ser usado, copiado, estudado, modificado e redistribuído com algumas restrições. A maneira usual de distribuição de *software* livre é anexar a esse uma licença de *software* livre e tornar o código fonte do programa disponível; essa licença na maioria dos casos permite a utilização do aplicativo sem a necessidade de pagar para adquiri-lo ou utilizá-lo, por essa razão, comumente *software* livre é confundido com *software* gratuito (FSF, 2015).

Ainda nesse sentido, Faria (2014), alerta que uma ferramenta proprietária pode ocasionar em aprisionamento tecnológico, sendo que o administrador dos serviços não poderá trocar de solução posteriormente, sem ter de renovar as licenças.

Pelos motivos apresentados, o *software* escolhido para análise nesse trabalho é o Bacula, por ser um *software* livre, esse será apresentado na sequência.

Somado a necessidade de garantir acessibilidade a informação, é também preciso um monitoramento desse e outros serviços, por este motivo, neste trabalho serão apresentadas as funcionalidades e implementação do *software* Zabbix, que também é um *software* livre.

1.2 Objetivos

Na seção a seguir são apresentados: o objetivo geral e os objetivos específicos do sistema proposto.

1.2.1 Objetivo Geral

Apresentar uma solução de monitoramento e *backup* com base em ferramentas de *software* livre em uma rede de computadores para empresas na qual a implementação de ferramentas proprietárias cujo custo monetário com aquisição de licenças de uso possa vir a inviabilizar sua implantação.

1.2.2 Objetivos Específicos

- Estudar os conceitos sobre monitoramento de redes com foco no uso do SNMP (*Simple Network Management Protocol*);
- Estudar os conceitos sobre *backup* e redundância de dados;
- Demonstrar os problemas enfrentados por pequenas empresas com relação a monitoramento e redundância de dados;
- Apresentar as aplicações das ferramentas Bacula e Zabbix em um ambiente de testes.
- Demonstrar as funcionalidades dos serviços Bacula e Zabbix para um ambiente corporativo focado em pequenas empresas.
- Sugerir as possíveis soluções e melhorias para o ambiente, bem como suas limitações.

1.3 Justificativa

O presente trabalho pretende ser um guia inicial para que empresas ou pequenos negócios que não utilizam os recursos tecnológicos de monitoramento e redundância das informações possam aplica-las no seu dia a dia, para isso são apresentadas as soluções já existente e amplamente utilizadas pelas grandes instituições, mostrando que com pouco investimento é possível utiliza-las também em ambientes coorporativos com tamanho reduzido, em grande parte devido a utilização de *software* livre, que dentre as várias vantagens, uma é a não necessidade de adquirir licenças para a utilização do *software*. Com isso espera-se agregar valor as instituições, as tornando mais eficientes devido ao controle dos dados que atualmente tem um valor estratégico alto e com o monitoramento de serviços para uma tomada de decisão mais rápida possível.

1.4 Estrutura do trabalho

O trabalho está organizado em cinco capítulos, sendo este o primeiro. Ele contém as considerações iniciais, os objetivos e a justificativa para elaboração do trabalho o escopo onde a pesquisa irá ser conduzida. O restante do documento está organizado da seguinte forma:

O Capítulo 2 apresenta o referencial teórico do trabalho, que inclui: gestão da informação, *backup*, tipos de mídias, o *software* BACULA, seu gerenciamento e monitoramento, o protocolo SMTP, MIB e também o *software* ZABBIX. Todos estes temas foram tratados e possuem um detalhamento no capítulo apresentado.

O Capítulo 3 apresenta as ferramentas utilizadas no trabalho, que envolvem o Ubuntu, PostresSQL, MySQL, o Virtual Box e o Google Calendar,

No Capítulo 4, são apresentados os Resultados e Discussões que envolvem o *software* Bacula, como também apresenta o ambiente corporativo real onde um experimento foi conduzido.

Por fim, o Capítulo 5 apresenta as conclusões obtidas com esta pesquisa, e também as contribuições, relevância e algumas limitações do trabalho executado. Apresenta-se possíveis trabalhos futuros, que podem contribuir para aprimorar a pesquisa aqui relatada.

2 REFERENCIAL TEÓRICO

A forma na qual nos organizamos em sociedade no século XXI tem cada vez mais tornado os indivíduos e corporações dependentes das informações armazenadas no formato digital. Isso traz um desafio para gestores das áreas de tecnologias: como assegurar uma quantidade tão grande de dados? Neste capítulo são descritos os conceitos a respeito da gestão das informações, cuidados com a segurança das informações, principais definições de uma política de cópia de segurança e a análise do *software* Bacula como ferramenta de cópias de segurança no intuito de esclarecer a questão em aberto.

Também nesse capítulo são apresentadas as definições de monitoramento da rede e de seus serviços somado com a necessidade da redundância dos dados com a apresentação do *software* livre Zabbix.

2.1 Gestão das Informações

A gestão das informações tem intrínseca ligação no desenvolvimento de praticamente qualquer organização, do mesmo modo que também é vital para sua continuidade. Diante o exposto, fica claro que a implantação de uma política de cópia de segurança é necessária, para isso, deve-se realizar a aplicação de uma boa gestão das informações, que estipulam como administrar e planejar os dados de uma organização. Conforme explana Toigo (1990), a gestão de dados tem as seguintes abrangências:

- Identificar o modo como as informações são armazenadas;
- Definir níveis de segurança para acesso aos dados;
- Ilustrar as regras de manutenção e controle de dados;
- Definir os mecanismos para garantir o armazenamento e disponibilidade dos dados.

Ainda conforme Toigo (1990), elaborar com atenção o planejamento da cópia de segurança é a melhor forma de garantir o acesso e segurança das informações; no contexto empresarial elas são classificadas como segue:

- Informações críticas do negócio: informações que a instituição necessita para ter continuidade a seus negócios e atividades primárias;
- Informações redundantes: documentos que possuam mais do que uma cópia;
- Informações temporárias: informações usadas de modo pontual, que não têm necessidade de armazenamento por período longo;
- Informações de configuração: arquivos de configurações de serviços, como por exemplo, regras para o funcionamento de um servidor DHCP (*Dynamic Host Configuration Protocol*).
- Informações desnecessárias: informações que não são utilizadas nas atividades da instituição.

Em observância a essas definições, para uma cópia eficiente das informações e que utilize o mínimo de recursos, é necessário realizar uma escolha inteligente dos dados a serem guardados, ou seja, copiar apenas arquivos relevantes ao funcionamento da instituição. Toigo (1990) também ressalta algumas observações para um *backup* ser efetivo e seguro. Segundo o autor, o local de armazenamento dos *backups* deve ser restrito, para evitar que pessoas não autorizadas tenham acesso aos dados. Outro item levantado é armazenar uma cópia em outro local. Importante também ao realizar os *backups* é utilizar criptografia para garantir o sigilo das informações em caso de roubo e ter atenção com o descarte de *backups* antigos, pois uma mídia mal descartada compromete a segurança da instituição. Também é necessário definir com atenção a mídia utilizada, levando em consideração alguns parâmetros como vida útil da mídia, tamanho da cópia e frequência de atualização, como será ilustrado nos tópicos seguintes.

2.2 *Backup*

No caso dos *backups* pessoais, na maioria dos casos, o usuário necessita apenas do acesso aos seus arquivos em sua última versão e dificilmente terá a necessidade de analisar um histórico das modificações deles, além disso, a quantidade de dados a ser salva não costuma ser grande para esse perfil de utilizador. Para essa demanda, armazenar a última versão dos seus arquivos em um *hard disk*

externo mensalmente e salvar arquivos mais importantes em um e-mail diária ou semanalmente são medidas suficientes para assegurar o acesso aos dados; já no contexto empresarial, Faria (2014) explica que o *backup* é uma cópia segura das informações, de modo que permita assegurar a continuidade do negócio em casos de sinistros, sendo por isso, de suma importância para a continuidade do empreendimento. Para que isso seja alcançável, é preciso organizar políticas de backup para garantir o acesso as informações com o mínimo de defasagem (em alguns casos é necessário manter histórico de modificações de arquivos por tempo determinado para fins de auditoria) na melhor eficiência possível balanceando tempo de restauração com espaço disponível.

Para alcançar essa eficiência é necessário intercalar modos diferentes de realizar o backup dos dados, Barros (2007), define o *backup* nas seguintes classificações:

- *Backup* Completo, *Full*, Normal ou Total: É o *backup* integral, abrange todos os arquivos selecionados para a mídia externa. Deve ser feito em intervalos de tempo maiores, pois ele inicia o ciclo dos *backups* diferenciais e incrementais.
- *Backup* Diferencial: É um *backup* cumulativo de todos os arquivos criados ou alterados depois do último *backup* completo, normalmente longo e de restauração rápida, em conjunto, um *backup* completo e um *backup* diferencial incluem todos os arquivos selecionados para cópia, alterados e inalterados, sendo então, necessário esses para restaurar todos os dados.
- *Backup* Incremental: Apenas os arquivos criados ou alterados desde o último *backup* completo ou incremental são salvos, por isso são mais rápidos para a cópia dos dados, exigindo menos espaço de armazenamento, por outro lado, a recuperação dos arquivos é mais lenta, pois é necessário restaurar todas as cópias em sequência.

Percebe-se como essas três formas de *backups* são realizadas analisando a Figura 1 e a Figura 2; no modo diferencial o processo do *backup* exige mais processamento e a restauração é mais simples, no modo incremental, tem-se um processo mais fácil e econômico de cópia, mas uma complexidade maior para restauração, tendo em vista que é necessário o último *backup* completo e todos as cópias adicionais subsequentes. Por esse motivo, em cenários em que o tempo sem

acesso aos arquivos causará grandes prejuízos, recomenda-se maiores investimentos em espaço em disco para implementação de *backups* totais e diferencias, já em cenários em que o tempo sem acesso aos arquivos não cause prejuízos, pode-se utilizar *backups* totais e incrementais.

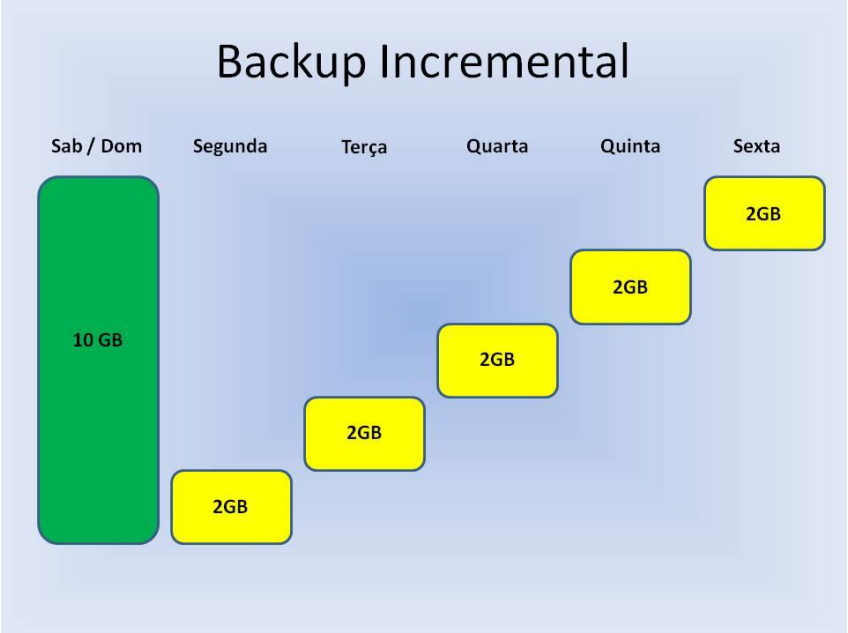

Figura 1 - Backup Diferencial.
Fonte:http://fernandopsalmeida.blogspot.com.br/2011/11/backup-diferencial-e-backup-incremental.html)

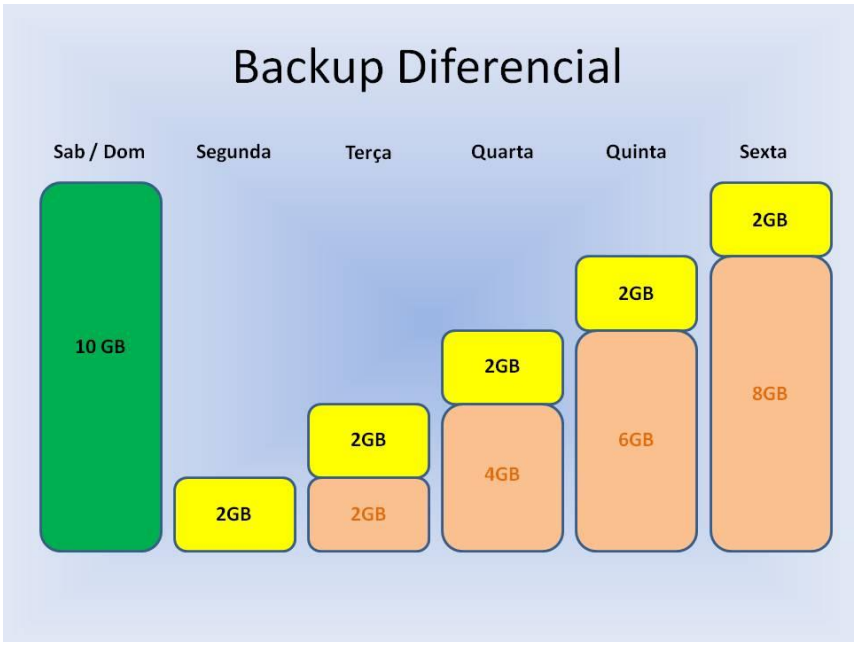

Figura 2 - *Backup* Incremental
Fonte: (BACKUP, 2015).

2.2.1 Topologias de *Backup*

Guise (2009) exemplifica sobre a topologia de *backup*, o autor afirma que existem dois modos que os *backups* podem ser realizados: descentralizado e centralizado, a grosso modo, um ambiente de *backup* centralizador significa a utilização de um *software* único e um ambiente de *backup* descentralizado pode ter diversas soluções mais simples operando em conjunto.

Na topologia de *backups* descentralizados, geralmente cada estação realiza suas próprias cópias de segurança em um dispositivo que está fisicamente conectado a esse, em geral pequenas organizações quando realizam o *backup* tendem a operar dessa forma, o que, como foi apresentado, não é o ideal. Um backup descentralizado assemelha-se ao modo apresentado na Figura 3, em geral *backups* descentralizados representam alta demanda dos administradores e dos usuários e pode acarretar em inacessibilidade de dados devido a uma maior operação manual nas operações das cópias. Esse modo também pode resultar em unidades maiores de mídia que exigidas. Apesar do exposto, *backups* descentralizados apresentam a vantagem de apresentar um menor tempo de recuperação, pois cada usuário, em teoria, pode realizar o processo de forma pontual e isolada, a Figura 3 ilustra esse funcionamento.

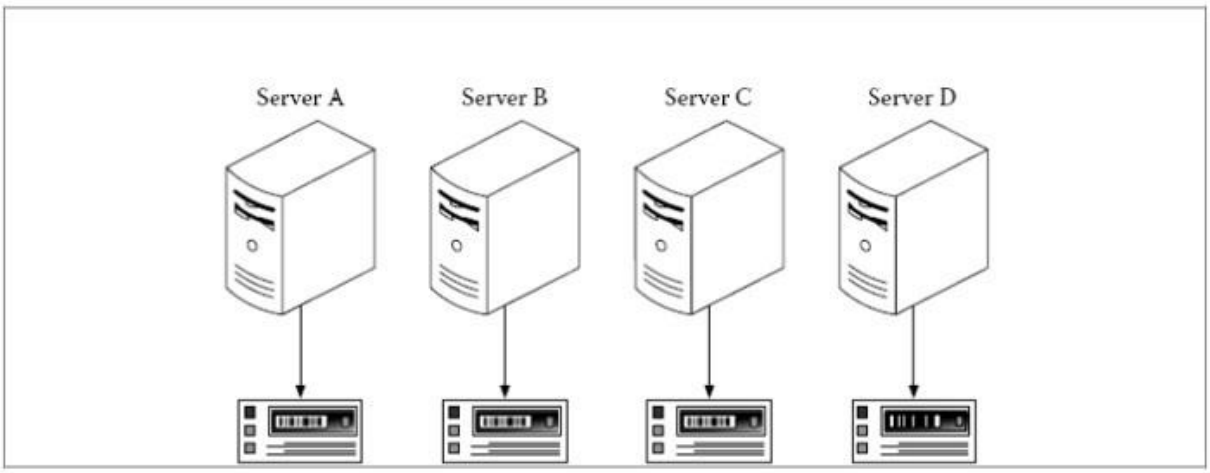

Figura 3 - *Backup* Descentralizado.
Fonte: Guise, P. D. *apud* (JUNIOR, B. P., 2010)

Ainda conforme Guise (2009), em uma topologia centralizada várias estações enviam dados para um único servidor, conforme pode-se observar na Figura 4, em que apenas um único servidor é quem gerencia os *backups* de todas as estações; os

computadores no ambiente enviam os dados ao servidor de backup, que escreve, restaura e mantém os dados nas mídias de *backup*.

2.3 Mídias

Conforme explica FERREIRA (2003), as mídias para *backup* são as memórias não voláteis, sendo as mais utilizadas as magnéticas e ópticas; os componentes físicos onde os dados são armazenados, possuem tamanho, volume, durabilidade e velocidades de leitura e gravação diferentes. Nos próximos tópicos as mídias mais comuns para *backup* serão apresentadas.

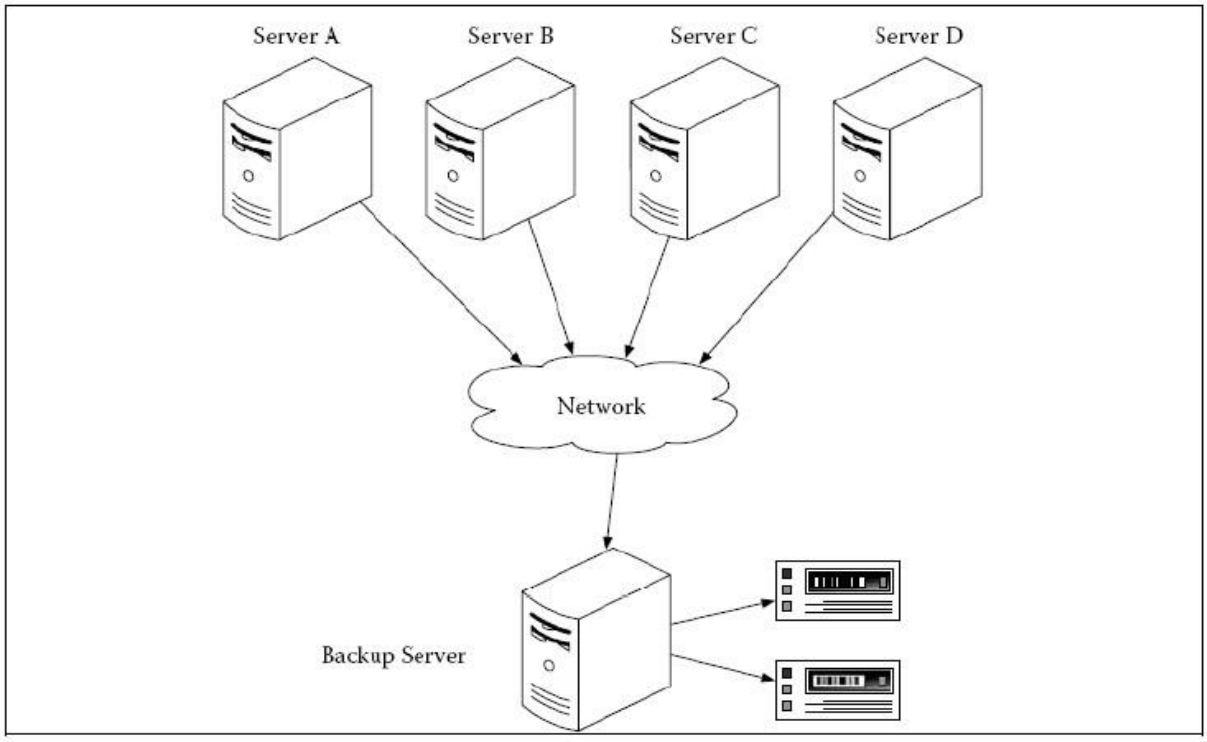

Figura 4 - *Backup* Centralizado.
Fonte: Guise, P. D. *apud* (JUNIOR, B. P., 2010).

2.3.1 Mídias Ópticas

Conforme detalha MORIMOTO (2010), durante a segunda metade da década de 70, as empresas Philips e Sony trabalharam na elaboração de uma mídia óptica para substituir os discos de vinil. Essa cooperação originou a tecnologia do CD-ROMs

(*Compact Disc-Read-Only Memory*), que evoluiu ao longo das décadas e posteriormente deu origem ao DVD e ao Blu-ray, que usamos atualmente.

Ainda de acordo com Morimoto (2010), dentro do CD, os dados são armazenados na forma de sulcos, que contrastam com a superfície lisa e reflexiva do disco. Durante a leitura, os sulcos dissipam o laser do leitor, enquanto o restante da superfície o reflete, permitindo que o leitor diferencie os bits "1" dos bits "0" com base na intensidade da reflexão. No CD os dados são armazenados na forma de uma espiral contínua, como em um disco de vinil.

O baixo custo de produção e grande capacidade de armazenamento (para a época) fizeram com que os CDs se tornassem rapidamente a mídia mais utilizada para armazenamento. Mesmo hoje em dia, os CDs ainda são populares como uma forma de armazenamento de dados, pois são mídias baratas e praticamente qualquer PC possui um drive óptico para leitura e gravação dos dados.

Para situações não coorporativas, realizar backups dos dados em mídias como o DVD (*Digital Versatile Disc*) e o Blu-ray são uma opção atrativa, pois possuem um custo relativamente baixo para um satisfatório armazenamento; entretanto, Faria (2014) explica que atualmente com o preço mais acessível de mídias como o HD (*Hard Drives)* e Fitas magnéticas, que serão explicadas na sequência, é mais aconselhável utilizá-las em empresas, visto que são mais resistentes e podem ser armazenadas por um tempo maior. Empresas pequenas e de médio porte, que em sua análise constatarem que o custo dos equipamentos para *backup* em fitas magnéticas for alto pelo retorno oferecido, podem paliativamente realizar o *backup* em *hard drives* externos combinado com cópias em DVD’s ou Blu-rays, tendo assim redundância do *backup* em duas mídias, o que garante acesso aos dados no caso de falha de um dos equipamentos.

2.3.2 *Hard Drive*

Zanatti (2014) define o HD como uma memória não volátil, o que significa que os dados não são perdidos quando a corrente elétrica é interrompida.

Conforme o auotr, a cabeça de leitura e gravação de um disco rígido funciona como um eletroímã extremamente preciso, sendo capaz de gravar trilhas de dados

medindo centésimos de milímetro de largura. Quando os dados estão sendo gravados no disco, a cabeça utiliza seu campo magnético para organizar as moléculas de óxido de ferro da superfície de gravação, o que faz com que os polos positivos das moléculas se alinhem com o polo negativo da cabeça; com sua polaridade alternada constantemente varia-se também a direção dos polos das moléculas na superfície magnética, desse modo, conforme a direção dos polos tem-se os bits 1 ou 0. Quando é preciso ler os dados gravados, a cabeça de leitura capta o campo magnético gerado pelas moléculas alinhadas e a variação entre os sinais magnéticos gera uma pequena corrente elétrica que caminha através dos fios da bobina, quando o sinal chega à placa lógica do HD ele é interpretado como uma sequência de bits.

Com base no exposto, é visto que o HD é uma opção viável para ser usado como mídia de backup para instituições de pequeno porte. Isso se deve ao fato de que a alta difusão que a tecnologia teve, o que faz com que o HD tenha espaço suficiente, velocidades de leitura e gravação adequados além de durabilidade por um valor atingível pela maioria das empresas, contando também como a comodidade de existirem HD externos, que podem ser ligados a entrada USB (*Universal Serial Bus*) do servidor, o que facilita a instalação e utilização do mesmo.

2.3.3 *Flash Drivers*

A memória *USB Flash Drive*, popularmente conhecido por *pen drive* é um dispositivo de memória constituído por memória *flash* que oferece uma série de vantagens em comparação com os demais dispositivos de armazenamento portáveis, como por exemplo os disquetes, dispositivo que entrou em desuso após maciça utilização dos *pen drives*; em grande parte por esses serem mais compactos, velozes e possuírem maior capacidade de armazenamento, além de serem mais resistentes devido à ausência de peças móveis. Os *drives flash* são compatíveis praticamente com todos os sistemas operacionais, como o Windows, Mac OS X, Linux, etc.

Da mesma forma que as mídias ópticas, o *pen drive* é uma boa opção para *backups* pessoais, por outro lado, em ambientes corporativos seu uso para armazenamento redundante não é aconselhável; sugere-se utilização de uma mídia cujo o volume suportado atenda maiores quantidades do que encontradas em *flash*

drivers e seja mais resistente e confiável, tendo em vista que o *pen drive* é projetado para usuários domésticos e produzido em larga escala, o que significa que em geral a sua fragilidade é maior e a confiabilidade menor ao ser comparado a um HD ou uma fita magnética.

2.3.4 *Solid-State Drive*

Morimoto (2010) descreve o SSD (*Solid-State Drive*) como um "HD" que utiliza *chips* de memória *flash* no lugar dos discos magnéticos, conforme mostra o autor, eles foram projetados para substituírem o HD, sendo conectados pelos barramentos SATA (*Serial Advanced Technology Attachment*) ou IDE (*Integrated Drive Eletronics*), sendo compatíveis com os dispositivos atuais, o que facilita e acelera sua utilização. O autor apresenta que apesar das taxas de transferência serem similares à de um HD modesto, os SSDs atingem tempos de acesso aos dados extremamente baixos, e isso melhora seu desempenho consideravelmente e reduz bastante o tempo de inicialização *boot* dos computadores. Os SSDs oferecem também a vantagem de consumirem menos energia elétrica, serem mais resistentes por não possuírem partes móveis, além de serem silenciosos.

Em compensação, eles possuem uma desvantagem fatal, que é a questão do custo, Morimoto (2010) apresenta que no ano de 2007, a aquisição desse equipamento por instituições com poucos recursos ficaria inviável; atualmente, no ano de 2015, o acesso desses equipamentos ainda é custoso, apesar de uma grande queda de preço se comparada com a análise feita em 2007. Se essa tendência repetir-se, como ocorreu com diversas tecnologias até então, em breve SSDs também serão uma alternativa para *backups* pessoais e empresariais, sejam grandes, médios ou de pequeno porte.

2.3.5 Fitas Magnéticas

As fitas magnéticas estão disponíveis em rolos, cartuchos ou cassetes; para o armazenamento de dados as fitas são uma das principais representantes dos modos de armazenamento, sendo o mais antigo ainda usado para *backups*.

Quando comparadas a outras mídias, as vantagens das fitas são a grande capacidade de armazenamento, o baixo custo relativo por unidade armazenada, a alta expectativa de vida e a confiabilidade do armazenamento ao longo de sua vida útil, por outro lado suas desvantagens são o acesso sequencial e o elevado custo dos dispositivos de leitura/gravação e também a sua maior fragilidade.

Como esse trabalho destina-se especialmente para empresas com pouco volume de dados, recomenda-se o armazenamento dos dados em um HD específico para isso, e se for desejável, replicação do *backup* em outro HD ou em outra mídia.

2.3.6 *Backup* em Nuvem

Não diferenciando entre usuários domésticos ou corporativos e tampouco o tamanho da instituição, desastres e acidentes podem ocorrer a qualquer um e em qualquer lugar. Por isso o *backup* é tão essencial, assim como manter cópias em locais distintos, umas das diversas formas de evitar inacessibilidade dos dados é manter a mídia fisicamente distante de onde o *backup* é realizado. Atualmente, para atender essa especificidade, as cópias de segurança são também realizadas no que comercialmente é chamado de *backup* na nuvem, o que basicamente consiste em enviar seus dados para uma empresa terceirizada através da Internet, no caso da perda de acesso aos dados, uma cópia estará disponível nesse serviço.

Empresas com grandes volumes de dados tem dificuldade de utilizar esse meio de *backup*, pois suas desvantagens são sobrecarregar o uso da rede de dados e ter custo adicional contratando a empresa que disponibilizará o serviço; por outro lado, existem empresas que disponibilizam esse serviço de forma gratuita em tamanho atrativo e suficiente para realizar a cópia integral dos *backups* de pequenas organizações, essas podem através do *software* gerenciador dos *backups*, programar o envio dos dados em horário de ociosidade da rede de dados mitigando as desvantagens dessa prática.

Caso a instituição tenha mais de uma sede, é possível utilizar o conceito de *backup* em nuvem entre as próprias filiais, por exemplo, uma empresa com duas sedes, pode realizar o *backup* localmente e enviar seus dados de uma sede para a outra, mantendo assim acesso a informação, mesmo no caso da destruição física

completa de uma das sedes, essa forma é atrativa para empresas de qualquer tamanho, pois não necessitam do serviço de uma empresa terceirizada.

O *backup*, para ser efetivamente redundante, demanda que as informações sejam armazenadas em localidades diferentes dos dados originais, por exemplo, no caso do *backup* de um servidor e um outro dispositivo alocado fisicamente ao lado desse: no caso do HD do servidor parar de operar, ainda se terá acesso aos dados, devido a cópia redundante na mídia que está nas proximidades. Já no caso de um incêndio na sala, ambas as mídias, original e *backup,* serão danificados e a inacessibilidade das informações será um problema agravante. Algumas ameaças inerentes à segurança física dos dados são: desastres naturais, vandalismo, incêndios e outros. Por essas e outras situações é de extrema necessidade que exista um *backup* externo situado fora da localidade original (BARROS, 2007).

2.4 Bacula

Segundo Sibbald (2015), detentor da licença do nome do *software* Bacula (o qual é formado pelas palavras "*backup"* e "Drácula", uma junção que remete ao funcionamento do Bacula, que geralmente é agendado para operar no período noturno, para não concorrer com a utilização de recursos com os usuários, e com o vampiro Drácula da literatura, que ataca apenas a noite) e fundador do projeto, explica no manual oficial do *software* Bacula que a aplicação em questão é um conjunto de serviços que permitem ao administrador gerir *backups*, restaurações e verificar dados dos computadores através da rede em diversos sistemas operacionais, possuindo também suporte a vários tipos de mídias, desde fitas magnéticas até discos rígidos.

Ainda conforme Sibbald (2015), o Bacula pode ser utilizado em ambientes com apenas um computador até centenas de dispositivos espalhados em uma grande rede. Na mesma linha de pensamento, Faria (2014), afirma que o Bacula é um conjunto de programas que permite administrar *backups*, restauração e verificação dos dados de computadores em uma rede com sistemas variados, tendo como grande vantagem ser uma solução sob a licença *software* livre, ou seja, possui código fonte aberto para análise de qualquer pessoa que esteja interessada em seu funcionamento.

Os *softwares* apresentados nesse trabalho, o Bacula e o Zabbix, não exigem a necessidade de adquirir licenças para sua utilização, evitando então, gastos que em *softwares* proprietários seriam necessários, o que faz esse tipo de solução ser mais facilmente aplicável em empresas de pequeno e médio porte pela inexistência desse custo.

Complementar ao exposto, Barros (2007) define o Bacula como solução completa para realização de *backups* em rede, com grande quantidade recursos e funcionalidades, o autor afirma que o funcionamento do Bacula é dividido nos seguintes módulos apresentados na sequência.

O Bacula é um conjunto de 5 serviços: *Director*, *Console*, *File*, *Storage* e o *Monitor*, como se pode observar na Figura 5.

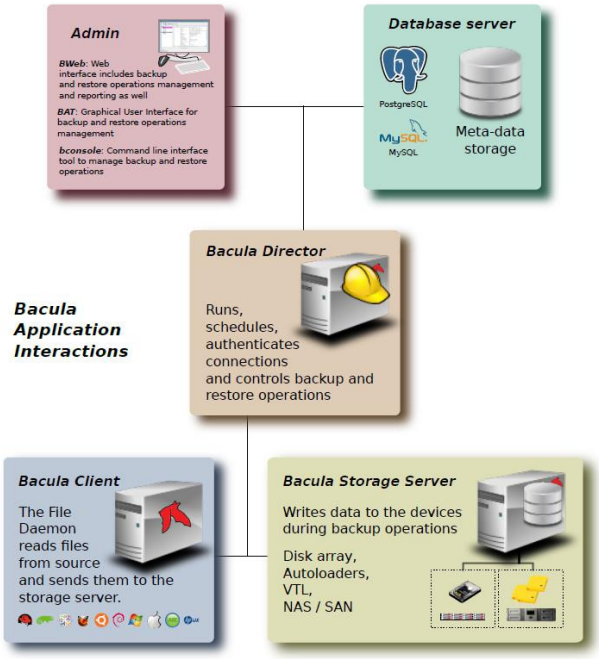

Figura 5 -Componentes bacula
Fonte: (SIBBALD, 2015)

O Bacula *Director* é o serviço que supervisiona todos os *backups*, restaurações verificações e os arquivos. Conforme indicado em Kern (2015), o administrador do Sistema utiliza o Bacula *Director* para agendar *backups* e recuperações de arquivos. Esse serviço opera em segundo plano no servidor. Faria (2014) reforça essa ideia afirmando que o *Director* consiste em administrar todos os processos de *backup*, restauração, verificação e arquivamento.

O serviço de console do Bacula, o *console manager*, permite que o administrador se comunique com o Bacula *Director* diretamente. SiBBALD (2015)

ilustra que esse serviço é disponibilizado em 3 versões: console com interface baseada em texto, *QTbased* e a *wxWidgets*.

A primeira e mais simples é a console baseada em texto, segundo Kern, essa interface é adequada para a maioria dos administradores. A segunda versão é baseada na interface *gnome*, sendo bem mais completa que a primeira interface, mas com praticamente as mesmas funcionalidades que a primeira oferece. Já a terceira versão, como a segunda, não traz novas funcionalidades que a interface pela console não disponha, mas facilita o gerenciamento, pois possui recursos que promovem sua utilização, tais como completar o comando usando a tabulação e ajuda instantânea a respeito do comando que o administrador está digitando.

Faria (2014) resume o *Console Manager* como o meio entre o usuário ou administrador com o *Director* podendo ser executado em qualquer computador da rede e em diversos sistemas operacionais.

File Daemon é o *software* que é instalado na máquina que vai ter os arquivos salvos, ou seja o *Client*, esse *Daemon* vai enviar os arquivos solicitados pelo *Director Daemon*.

*Storage Daemon a*dministrar a gravação e restauração dos dados e atributos dos *backups* fisicamente em mídias apropriadas.

Catalog mantém a indexação de todos os arquivos que são armazenados no *backup* e gera uma base de dados dos volumes gerenciados pelo *Director Daemon. O Catalog* agiliza a busca de um arquivo no *backup* na hora de uma restauração, pois torna-se mais rápido encontrá-lo.

2.5 Gerenciamento e Monitoramento

Simples e eficazes medidas que englobam a aplicação de ferramentas de gestão e monitoramento do ambiente são a resposta para incidentes, a qual a solução se torna mais ágil e justifica o investimento. Nos tópicos a seguir, os conceitos de gerência e monitoramento são apresentados, assim como o *software* Zabbix.

2.5.1 Modelo de Gerenciamento de Rede

A ISO (*Internacional Organization for Standardization)*, projetou um modelo de gerenciamento de rede com o intuito de padronizar e aperfeiçoar o processo de monitoramento da rede e seus serviços, esse modelo é dividido em cinco segmentos (GOBATO, 2012), (KUROSE e ROSS, 2010)

- Gerenciamento de falhas: tem a incumbência de identificar, registrar e reagir às falhas da rede, fazendo o tratamento em tempo real dessas.
- Gerenciamento de configuração: permite que o administrador da rede conheça os equipamentos que fazem parte da rede administrada e quais são as configurações de hardware e *software* que esses possuem (KUROSE e ROSS, 2010).
- Gerenciamento de desempenho: é aplicado no gerenciamento da Internet, pois é responsável por gerir, analisar, informar e controlar o desempenho de diversos equipamentos distintos da rede.
- Gerenciamento de segurança: controla o acesso dos dispositivos da rede, controlando e identificando violações de políticas de segurança definidas, sendo responsável pela criação, monitoração e manutenção de logs e serviços de segurança.
- Gerenciamento de contabilização: Possibilita o controle de acesso dos usuários aos recursos da rede, como por exemplo, quotas de utilização para cobrança por utilização e reserva de acesso privilegiado a recursos especificados (KUROSE e ROSS, 2010).

2.5.2 A Infraestrutura do Gerenciamento de Rede

Conforme Kurose e Ross, (2010) explicam, existem três agentes principais em uma arquitetura de gerenciamento de rede: o agente gerenciador, os dispositivos gerenciados e o protocolo de gerenciamento de rede.

O agente gerenciador é a aplicação em qual o administrador pode interagir com os dispositivos de rede, controlando a coleta, processamento, análise e/ou a apresentação dos dados providos do gerenciamento da rede.

Já os dispositivos gerenciados são os equipamentos inseridos na rede e seus *softwares*. A gama de itens que podem ser gerenciadas é extensa, um dispositivo gerenciado pode ser um host, um roteador, um hub, um *no-break*, etc, e cada dispositivo gerenciado pode ter diversos itens a serem monitorados.

Por fim, o protocolo de gerenciamento de rede é o meio de comunicação entre gerenciador e gerenciado, e possibilita que se analise as informações dos dispositivos gerenciados e também execute ações sobre eles. É importante ressaltar que o protocolo de gerenciamento por si próprio não gerencia a rede, apenas fornece um meio no qual o gestor de rede pode gerenciar, monitorar, testar, consultar, configurar, analisar, avaliar e controlar a rede. O protocolo mais utilizado atualmente com esse propósito é o SNMP, exposto a seguir (GOBATO, 2012).

2.6 SNMP

O *Simple Network Management Protocol* ou Protocolo Simples de Gestão da Rede trata-se de um protocolo, definido pelo IETF (*Internet Enginnering Task Force*) usado para gerenciar e monitorar redes que utilizam a pilha de protocolos TCP/IP (*Transmission Control Protocol / Internet Protocol*). Com o SNMP, os gestores podem monitorar os equipamentos e serviços da rede de forma centralizada, também é possível usar o SNMP para analisar o desempenho de uma rede, detectar problemas, acusar falhas e acompanhar como os recursos estão sendo utilizados.

Segundo Mauro e Schmidt (2005), no livro "*Essential SNMP, Second Edition*", o SNMP é constituído por um conjunto simples de operações que permitem a leitura de informações ou a escrita de parâmetros em equipamentos que implementem o SNMP. Conforme o autor explana, o SNMP pode ser utilizado para o gerenciamento de diversos dispositivos distintos, como equipamentos de rede, fontes de energia e também *softwares*, análise do fluxo de aplicações web e a disponibilidade de um serviço de banco de dados sãos exemplos de um possível monitoramento.

Ainda conforme os autores, são dois os componentes principais no gerenciamento SNMP, o agente e o gerente. O gerente forma-se de um servidor que execute algum *software* responsável pelas tarefas de análise e gerenciamento de uma

rede e são chamados NMS (*Network Management Stations*); os agentes são os dispositivos que enviam a informação ao gerente.

Um serviço NMS pode solicitar informações dos agentes pelo meio de *poolings*, caso tenha permissão de leitura e gravação, modificar a configuração dos agentes, receber *traps* dos agentes, ou seja, receber informações do agente sem uma solicitação prévia. O agente SNMP é executado nos dispositivos de rede monitorados, podendo ser um aplicativo separado ou incorporado ao *firmware* ou sistema operacional do dispositivo. Em síntese, os *polls* utilizam-se de consultas dos agentes iniciadas pelo NMS e as *traps* são um método utilizado pelo agente para informar o NMS a ocorrência de algum evento importante (MAURO e SCHMIDT, 2005) .

As informações consultadas pela NMS são definidas em uma lista de objetos, a MIB (*Management Information Base*), que se compara com um banco de dados de objetos gerenciados que o agente monitora. Todo tipo de informações sobre o estado consultado pela NMS é definido em uma MIB, sendo possível que muitas MIBs sejam implementadas em um agente, porém uma MIB específica, denominada MIB-II, definida pela RFC (*Request for Comments*) 1213, é implementada por todos os agentes e engloba informações essenciais do gerenciamento TCP/IP.

Até a publicação deste trabalho, o SNMP já evoluiu para sua terceira versão, o SNMPv3, passando pelo SNMPv1 e pelo SNMPv2. O SNMPv1, foi desenvolvido no ano 1988 pela IETF, teve sua origem no protocolo de monitoramento de gateways IP, o SGMP (*Simple Gateway Management Protocol*), suas RFCs de lançamento foram as RFC 1065, 1066 e 1067, posteriormente substituídas pelas RFCs 1155, 1156 e 1157.

Ainda de acordo com Mauro e Schmidt (2005), o método de segurança desenvolvido para o SNMPv1 é fraco, podendo ser facilmente burlado, pois o SNMPv1 utiliza *Strings* de comunidade**,** que funciona com todos os dispositivos de uma rede sendo considerados como integrantes de uma comunidade, toda troca de mensagens SNMPv1 entre os dispositivos membros são identificadas por uma *string* que é apresentada no campo do cabeçalho da mensagem, essa *string* funciona como uma senha, toda mensagem recebida com a string de comunidade errada é rejeitada pelo receptor. O uso de *strings* protege contra ataques simples de um atacante com conhecimentos limitados, como as *strings* são enviadas em texto puro, elas podem ser facilmente descobertas. Pode-se inferir que para um usuário doméstico a

segurança fornecida pelo SNMPv1 é suficiente. Por outro lado, para grandes ambientes de rede, o SNMPv1 não fornece o nível suficiente de segurança necessário.

O SNMPv2 definido pelas RFCs 1901 até a 1908 surgiu com a necessidade de aprimorar o SNMPv1 em algumas áreas, introduzindo um pequeno conjunto de novas funcionalidades, mas manteve os mesmos problemas em relação a segurança. A principal diferença entre a versão 1 e a versão 2 do SNMP é que esta possibilita a criação de um conjunto hierárquico, sendo então, escalável para um grande número de agentes.

Na versão SNMPv3, conforme Mauro e Schmidt (2005), não se teve mudanças na estrutura do protocolo, a não ser a implementação da criptografia que resolveu o problema da segurança e mudança na nomenclatura conforme os autores detalham no excerto a seguir:

> Apesar do SNMPv3 não fazer alterações ao protocolo, além do acréscimo de segurança por criptográfa, seus desenvolvedores têm conseguido fazer as coisas parecem muito diferentes através da introdução de novas convenções, conceitos e terminologia. As alterações terminológicas são tão radicais que é difícil acreditar que os novos termos essencialmente descrevem o mesmo *software* assim como o antigo, mas elas fazem. No entanto, elas diferem na forma como se relacionam entre si, e elas especificam com muito mais precisão as peças que uma implementação SNMP necessita. A mudança mais importante é que a versão 3 abandona a noção de gestores e agentes. Ambos os gestores e agentes agora são chamados de entidades SNMP. Cada entidade consiste de um motor SNMP e uma ou mais aplicações SNMP, (...) Esses novos conceitos são importantes porque definem uma arquitetura ao invés de simplesmente um conjunto de mensagens; a arquitectura ajuda a separar diferentes peças do sistema SNMP de uma maneira que torna possível uma aplicação segura. (MAURO e SCHMIDT, 2005, p. 74-75).

2.7 MIB

A *Management Information Base* (MIB) é o total dos objetos que podem ser gerenciados, ele procura definir todas as informações necessárias para gerenciamento da rede e é utilizado como guia de referência pelo protocolo SNMP.

O processo foi elaborado dessa maneira, pois o gerente e agente para poderem comunicar-se devem estabelecer nos nomes os significados das operações. Ressalta-se que não é papel do protocolo SNMP definir uma MIB, ele atua somente no formato da mensagem e descreve como elas são codificadas.

Os objetos de uma MIB são definidos pelo padrão ASN.1, uma MIB pode ser ilustrada como uma árvore abstrata com um *root* anônimo; os níveis da árvore são compostos pelos itens de dados individuais, identificadores de objetos que identificam ou nomeiam unicamente os objetos da MIB na árvore e identificadores que possuem uma organização hierárquica com um dígito específico referente para diferentes organizações.

Segundo (KUROSE e ROSS, 2010), existe uma linguagem de definição de dados definida como SMI (*Structure of Management Information*), que especifica quais os tipos de dados, modelo de objeto e as regras usadas para escrever e revisar informações de gerenciamento; os objetos MIB são projetados nessa linguagem de definição de dados.

2.8 Zabbix

O Zabbix é um *software* livre, por isso, conforme já explanado, possui seu código fonte à disposição de consultas e alterações no caso dessas serem necessárias. Ele é um sistema de monitoramento distribuído capaz de monitorar a disponibilidade de diversos itens da infraestrutura da rede, serviços de rede e as aplicações, também proporciona a possibilidade de coletar informações dos dispositivos conectados na rede, recebendo as informações através de *scripts*, agentes nativos do Zabbix, agentes SNMP e até mesmo agentes IPMI (*Intelligent Platform Management Interface*), que é uma interface padronizada para gerência de *hardware*, esse padrão é utilizado pela Intel, Dell, HP e NEC.

Um dos principais colaboradores do Zabbix é Alexei Vladishev, sendo criador e principal desenvolvedor do projeto que teve desenvolvimento iniciado em 2001 na cidade de Riga, na Letônia, utilizando a linguagem de programação PHP (*PHP: Hypertext Preprocessor*), após finalizado, o programa foi disponibilizado aos usuários através de uma interface web com suporte a banco de dados (GOBATO, 2012).

Com o *Zabbix* é possível analisar diversos tipos de dados a respeito da rede, pode-se monitorar servidores, dispositivos virtuais, serviços da rede, etc. Adicional a manter os dados coletados, o programa dispõe de muitos modos para visualiza-los, seja através de gráficos, telas ou mapas. O *software* é flexível na análise das

informações coletadas, podendo-se criar alertas para apenas o que interessar ao administrador.

O *Zabbix* pode ser utilizado em ambientes de diversas complexidades, pois é flexível para trabalhar tanto com grandes redes com diversos itens a serem monitorados, quanto para pequenas organizações com poucos.

Grandes instituições brasileiras são referência no uso do Zabbix como sistema de auxílio a gerência, o que fornece indícios de sua utilidade; a aceitação maciça se deve a robustez e flexibilidade do *software*, pois é adaptável a diversos cenários distintos. Algumas das empresas brasileiras utilizadoras do Zabbix são: Petrobras, Dataprev e Serpro.

2.8.1 Estrutura do Zabbix

O Zabbix possui suporte a variados sistemas operacionais, como Linux, Solaris, Mac OS X, Windows e vários outros, mas sua aplicação servidor deve necessariamente ser hospedada em uma máquina com o sistema operacional Linux ou Mac OS, pois existe uma dependência em relação à estrutura do Zabbix, visto que o mesmo foi projetado com o intuito de ser uma ferramenta de código aberto, por isso não existe um pacote do servidor disponível para as versões do Windows (GOBATO, 2012).

O sistema divide-se em três componentes distintos: agente, servidor e interface, que são elucidados nos tópicos a seguir.

2.8.2 Agente Zabbix

O agente *Zabbix* tem a finalidade de monitorar ativamente os serviços e equipamentos localmente, como por exemplo, *hard drivers* e demais componentes, estáticas de utilização, temperatura e inumeráveis outros itens. Os sistemas podem executar o agente que irá agrupar as informações operacionais do equipamento monitorado e enviar essa coleta ao servidor, no caso de falhas, tanto de hardware (atolamento de papel em uma impressora, por exemplo) quanto de *software* (um

serviço que parou de operar), o agente *Zabbix* enviará o ocorrido ao servidor *Zabbix* que pode prontamente alertar os administradores do equipamento, relatando o ocorrido.

Segundo Pinto (2014), os desenvolvedores do *Zabbix* criaram o agente de monitoramento em paralelo com o sistema, esse agente foi codificado na linguagem de programação C, sendo suportado por diversas plataformas (inclusive Windows, Linux e Mac OS), sendo capaz de realizar coletas de dados do processador, memória, disco, informações da interface de rede e demais informações a respeito do dispositivo.

Ainda conforme Pinto (2014), no princípio, uma das preocupações dos desenvolvedores e dos administradores com interesse em utilizar o *Zabbix* era se instalar um *software* não essencial (o agente) em um servidor poderia levá-lo a consumir processamento excessivo, diminuindo assim o seu desempenho.

Nos testes realizados e divulgados pelos mantedores do site oficial do *Zabbix*, o mesmo constatou um uso de CPU (*Central Processing Unit*) não significativo e utilização de memória por volta de 4MB, conforme pode-se ser visto na Figura 6.

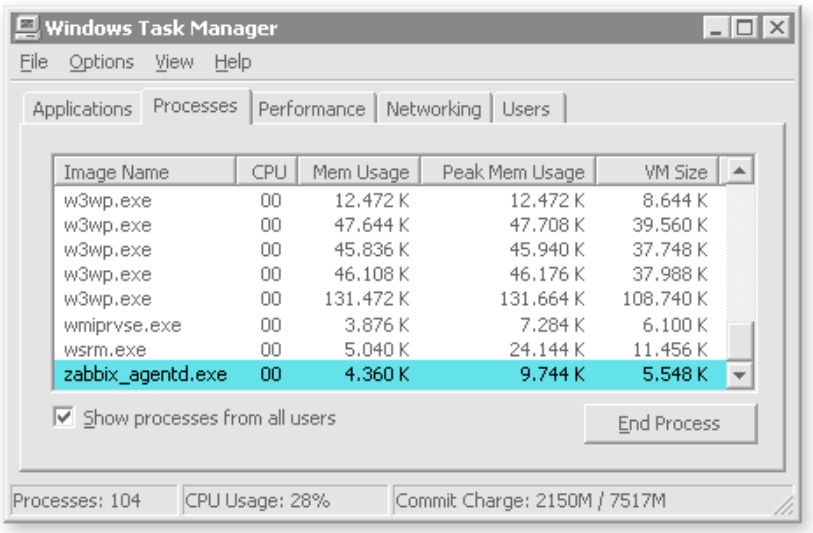

Figura 6 - Consumo Memória do agente.
Fonte: (MEMÓRIA, 2015)

Com base nos dados apresentados, pode-se concluir que a instalação do agente não irá afetar o desempenho do servidor (ou estação) monitorado; em um ambiente real, onde a utilização dos recursos de hardware pelo agente Zabbix esteja prejudicando o funcionamento ideal do equipamento analisado ou de seus serviços,

recomenda-se upgrade do mesmo, pois é provável que ele esteja operando em sua capacidade máxima (mesmo sem a instalação do agente).

Quando instalado, o agente utiliza-se de checagens ativas (*trapping*) e passivas (*polling*) para obter as informações a serem repassadas ao servidor; nas verificações ativas, o agente faz uma requisição ao servidor Zabbix e envia os resultados periodicamente e nas verificações passivas a requisição é enviada pelo servidor Zabbix para a máquina monitorada, que responde com os dados solicitados pelo servidor. Ambas as formas são ilustradas nas Figuras 7 e 8 respectivamente.

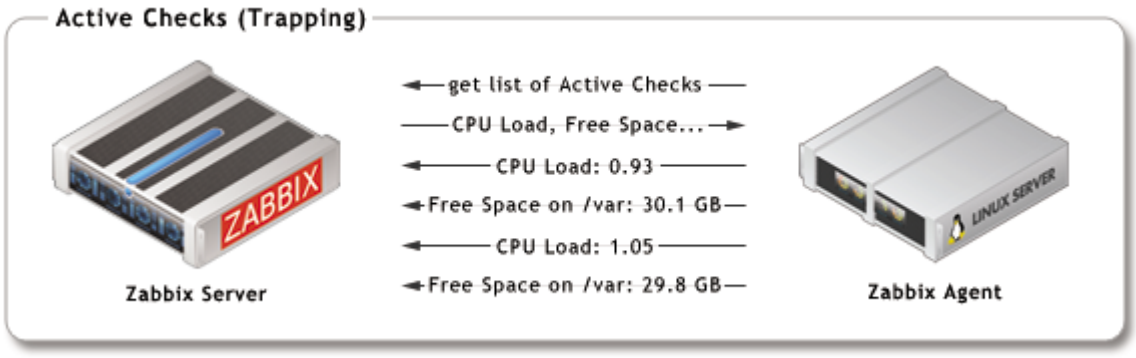

Figura 7 - Verificação ativa.
Fonte: (ZABBIX, 2015).

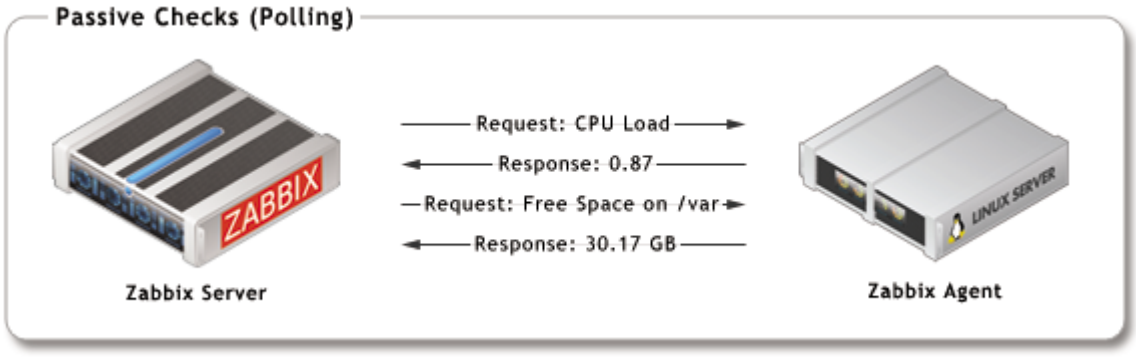

Figura 8 - Verificação passiva.
Fonte: (ZABBIX, 2015a).

2.8.3 Agente SNMP

O monitoramento por SNMP possibilita verificar diversos tipos de dispositivos de rede que não suportam a instalação do agente Zabbix, mencionado no tópico anterior. Na prática isso significa que é possível obter dados de roteadores, *switches*, impressoras, *nobreaks*, etc, não sendo necessário instalar e configurar qualquer

software adicional nesses dispositivos, pois a grande maioria dos equipamentos já vem com suporte nativo ao protocolo SNMP.

O servidor Zabbix por conseguir coletar dados de dispositivos que suportam o protocolo SNMP, esse, amplamente difundido e utilizado por diversos equipamentos distintos, torna-se robusto e completo por incorpora-lo ao seu funcionamento, sendo um intermediário ante as informações que o SNMP e o administrador da rede dispõem. A apresentação das informações através de uma interface gráfica web, com a possibilidade de gerar gráficos em tempo real e alertas configuráveis para que atuação para resolver o problema, em sua origem informada pelo protocolo SNMP, possa ser resolvida o mais rápido possível faz com que a utilidade das informações do SNMP aumente exponencialmente.

2.8.4 Monitoramento sem Agente

O Zabbix possibilita nativamente realizar alguns testes e monitorar recursos da rede utilizando ferramentas próprias ou do sistema operacional, sem precisar então, de qualquer tipo de agente, esse modo de operação é prático quando não for possível alterar a configuração de um servidor, por exemplo. Apesar da disponibilidade dessa forma de monitoramento, ela não é aconselhada pois as informações que podem ser obtidas com essa forma de operação são limitadas. A tabela a seguir apresenta quais os itens que podem ser analisados sem utilização de agentes:

Serviços da Rede	Disponibilidade da porta TCP Tempo de resposta da porta TCP Serviço de checagem
ICMP Ping	Disponibilidade do Servidor Tempo de resposta do ICMP response Perda de Pacotes
Remote Check	Executar comandos via SSH ou Telnet

Tabela 01 - Utilização de agentes

Fonte: (ZABBIZ, 2015b).

2.9 Servidor

O servidor é o centralizador da aplicação *Zabbix*, com ele, pode-se verificar remotamente o andamento dos serviços de rede do mais diversos, conforme apresentado nos tópicos anteriores, sendo ele, o componente central para o qual os agentes encaminharem as informações e estatísticas da disponibilidade dos equipamentos e serviços analisados.

O servidor armazena toda a configuração, dados estatísticos e dados operacionais recebidos, e é também ele que irá ativar e alertar os administradores quando surgirem problemas em qualquer um dos sistemas monitorados, seguindo as configurações que lhe forem definidas (VLADISHEV, 2015).

2.10 Interface

A interface é a estrutura que possibilita ao gestor o acesso para interagir e administrar o sistema, com o intuito de proporcionar um fácil acesso ao monitoramento dos dados e configurações através do *Zabbix*, a interface dele foi projetada para ser utilizada via web, o que possibilita ao administrador ou usuário acessar através do próprio navegador instalado em seu equipamento, para alguns navegadores, o Zabbix indica em seu site, alguns *plugins* que facilitam o acesso ao servidor e aos dados apresentados, conforme ilustrado na figura X.

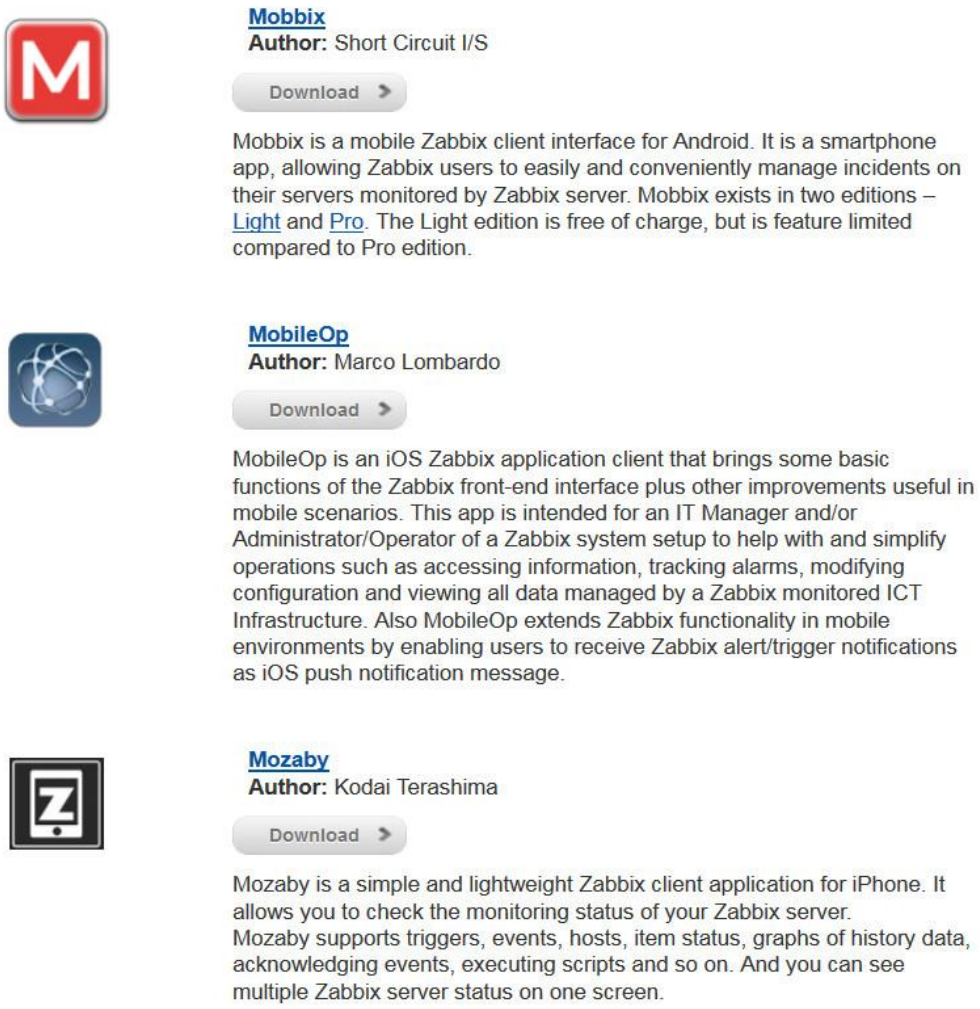

Figura 9 – Plugins.
Fonte: http://www.zabbix.com/third_party_tools.php

2.11 Alertas

Um grande diferencial do *Zabbix* em comparação com as demais ferramentas similares e que esse possui uma interface intuitiva que apresenta um modo fácil de definir prioridades para problemas e encaminhar avisos aos responsáveis de várias formas distintas.

Para entender melhor esse conceito, ilustremos a seguir o *Zabbix* sendo utilizado em uma empresa com uma filial na qual precisa ter comunicação ininterrupta, nesse ambiente, caso aconteça a perca de comunicação, o serviço foi configurado para disparar um alerta de gravidade baixa na interface web do *Zabbix*; passados 5 minutos o alerta aumenta para gravidade média e um e-mail é enviado para o grupo

de atendimento; após 30 minutos de permanência do problema o nível do problema eleva-se para grave e um e-mail e SMS são enviados para o gerente da equipe de atendimento. Como pode-se notar, os alertas e avisos são altamente maleáveis e configuráveis, cabendo ao gestor definir quais são os possíveis problemas que podem ocorrer em sua instituição e qual a gravidade desses, também é preciso configurar quais indivíduos devem ser alertados e através de qual meio.

A gama de modos de alertas disponíveis no *Zabbix* é grande, por exemplo, atualmente é possível configurar o *Zabbix* para realizar envio do alerta através pelo aplicativo *Whatsapp*, além dos modos mais tradicionais como e-mail e SMS. No Capítulo 4 a aplicação dessa funcionalidade é apresentada.

2.12 Appliances

Os desenvolvedores do *Zabbix* fornecem em seu site um modo simples para que os interessados no *software* possam testá-lo; com o download da a*plliance* fornecida não é preciso instalar nem mesmo o sistema operacional e nem o próprio *Zabbix*, pois eles já vêm pré-instalados e configurados, para usar esse recurso basta ter instalado um programa de virtualização como VMware ou VirtualBox.

Conforme evidencia (PINTO, 2014), esse modo de utilização deve ser utilizado apenas como teste, por 4 motivos principais:

- A imagem não vem otimizada para ambiente empresarial.
- O sistema operacional embutido tem as senhas padrões ou com baixa segurança.
- Dificuldade de atualizar através da a*plliance.*
- Não incluem sistemas operacionais de 64 bits.

3 MATERIAIS E MÉTODOS

Este capítulo contém os materiais utilizados para a instalação e configuração dos *softwares* analisados, esses referem-se às ferramentas e as tecnologias, utilizadas.

3.1 Ferramentas

Os materiais utilizados para o desenvolvimento deste trabalho foram as referências bibliográficas consultadas e citadas, assim como os diversos *softwares* e sistemas operacionais utilizados.

Para a implementação e utilização dos serviços de monitoramento e *backup* utilizou-se alguns sistemas operacionais e os seguintes *softwares:*

- Ubuntu 14.04 LTS (*Long Time Support*) como sistema operacional desempenhando o papel de servidor na instalação do Bacula e Zabbix.
- Sistemas operacionais da família Windows, Ubuntu, Debian e para instalação dos softwares em modo cliente;
- Bacula, como *software* da gestão do serviço de *backup*;
- Zabbix como *software* de monitoramento;
- Bancos de dados PostgreSQL e MySQL;
- Virtual Box;
- Google Calendar.

3.1.1 Ubuntu

O Ubuntu (CANONICAL, 2015) é uma distribuição GNU/Linux baseada na distribuição Debian, mantida pela empresa Canonical. O Ubuntu tem por grande diferencial do Debian por ser lançado a cada semestre, disponibilizando suporte 18 meses seguintes.

O Ubuntu surgiu com a ideia de ser um sistema operacional de simples utilização, assim qualquer pessoa poderá ter fácil interação com ele, não importando sua nacionalidade, nível de conhecimento e/ou limitações físicas.

A cada 2 anos uma versão com longo tempo de suporte é lançada, sendo esta a versão ideal para utilização em servidores e ambientes corporativos, pois, uma vez que o suporte é garantindo para os próximos 2 anos.

A primeira versão disponibilizada do Ubuntu foi a 4.10 , sua nomenclatura se dá fato de ter sido lançada no ano de 2004, no mês de outubro, após isso, semestralmente novas versões foram liberadas, a versão atual do Ubuntu é a versão 15.04, que leva o codinome de "*The Vivid Vervet*".

3.1.2 PostgreSQL

PostgreSQL é um SGBD (Sistema de Gerenciamento de Banco de Dados) utilizado para armazenar dados e informações de todas as áreas de atuação existentes, bem como gerir o acesso a essas. Conforme MILANI (2008), Um SGBD deve controlar o armazenamento dos dados e seu acesso, ou seja, quem pode ler ou alterar cada informação. Seu nome inicial era Postgre, designado em um projeto da Universidade Berkeley na Califórnia (EUA). Alguns alunos, em 1986 orientados pelo professor Michael Stonebraker deram início a um projeto para a elaboração de um modelo e as regras para um novo *software* de armazenamento de dados, obtendo apoio de órgãos do governo americano.

O SGBD mantém-se como *software* de código-fonte aberto, por isso sua escolha para esse trabalho, tendo em vista que desejasse apresentar uma alternativa de *backup* e monitoramento para pequenas empresas.

O PostgreSQL foi utilizado para instalação e configuração do *software* Bacula.

3.1.3 MySQL

O MySQL é um sistema de gerenciamento de banco de dados relacional de código fonte aberto de nível corporativo, sendo não apenas um banco de dados, mas sim, um gerenciador de banco de dados.

O Servidor MySQL foi originalmente desenvolvido para tratar com bancos de dados grandes de maneira mais eficiente que as soluções existentes, ele oferece hoje um diverso conjunto de operações. Conectividade, segurança e velocidade fazem com que o MySQL seja altamente adaptável para utilização de bancos de dados de dados na Internet.

Nesse trabalho o MySQL foi utilizado na instalação e configuração do *software* Zabbix.

3.1.4 VirtualBox

O VirtualBox é um *software* de propriedade da empresa Oracle, possui código fonte aberto, sob os termos da GPL (General Public License), pode ser instalado em sistemas operacionais Windows, Linux, Solaris Macintosh, tendo suporte a um grande número de sistemas convidados (VIRTUALBOX, 2015).

O VirtualBox utiliza partes de outros sistemas virtualizadores, como o QEMU e o BOCHS, para base de seu funcionamento. O sistema pode ser adquirido gratuitamente no site do fabricante e atualmente está na versão 5.0. Por ter interface simplificada e ser de fácil operação, nesse trabalho foi utilizado para a instalação dos sistemas operacionais utilizados para configurar e testar os serviços analisados.

3.1.5 Google Calendar

O Google possui diversos serviços disponibilizados gratuitamente para os seus usuários, como e-mail, redes sociais, etc. Nesse trabalho foi utilizado a Agenda do Google, o Google Calendar, pois essa possibilita o envio de SMS para o celular do usuário quando um evento na agenda for criado, como é possível através de eventos do servidor Zabbix criar eventos no Google Calendar, o mesmo foi utilizado para demostrar a possibilidade de utilizar esse recurso para enviar SMSs ao administrador da rede sem gerar custos para isso (GOOGLE, 2015).

4 RESULTADOS E DISCUSSÕES

Neste capítulo serão apresentados os resultados obtidos e como os mesmos foram elaborados.

4.1 Ambiente virtual Bacula

Para exemplificar o funcionamento do *software* Bacula, o mesmo foi instalado e configurado em uma máquina virtual Linux e o cliente utilizado foi uma a estação Windows hospedeira.

O processo da instalação do servidor é simples, podendo ser instalado pelos repositórios, se a distribuição disponibilizar o pacote, ou então, através da compilação do código fonte que pode ser obtida através do site do *software*.

No ambiente de testes foram instalados os componentes que compõem o servidor Bacula e configurados para funcionamento com intuito de estudar e explanar os conceitos apresentados no Capítulo 2.

4.1.1 Director Daemon

Na instalação do *Director Daemon* do Bacula, por padrão, o arquivo de configuração Bacula-dir.conf é criado no diretório "/etc/Bacula/", nesse arquivo são definidas as principais configurações do Bacula como um todo e onde estão definidas as localizações dos demais componentes.

Tarefas como agendamento de *backups*, quem são os clientes, quais são as informações e onde salvá-las são definidas nesse arquivo.

Barros (2007) afirma que a função do campo *Director* dentro desse arquivo de configuração é definir os atributos de nome e senha para a autenticação da console, também é nesse arquivo onde define-se em quais diretórios os arquivos binários e demais componentes necessários para o funcionamento do programa encontram-se no sistema operacional. É importante notar que a configuração do acesso ao Bacula pela console, para funcionar, deve ter a senha de acesso definida nos arquivos de

configuração igual a senha definida na variável *Password*, pois caso contrário não haverá comunicação entre os *Daemons*.

Ainda conforme Barros (2007), é nesse arquivo que se definem os trabalhos de *backup*, os *Jobs,* que são uma ação, seja de *backup* ou *restore*, a ser realizada, e também os clientes, que são os equipamentos incluídos para terem os dados salvos.

No ambiente de testes, optou-se por remover as entradas de *jobs* e clientes do arquivo principal e adicioná-las em um arquivo de configuração separado, para isso foi criado o arquivo Bacula-dir-client-and-*jobs*.conf no diretório "/etc/Bacula/".

Ainda nesse arquivo de configuração, deve-se definir o *Fileset*, que é um conjunto de regras usados para definir quais arquivos copiar de cada cliente, seguindo a mesma lógica dos *jobs* e clientes, optou-se por criar um novo arquivo, o Bacula-dir-filesets.conf no diretório "/etc/Bacula/" para definir essas configurações. Esses arquivos de configuração separados do *Director Daemon* não afetam o desempenho e funcionamento do serviço, sendo que, a ligação entre esses dois arquivos é realizada no arquivo principal, ressalta-se que esse modo de dividir as configurações em mais de um arquivo não altera o funcionamento e o desempenho do Bacula*,* mas facilita ao administrador prestar suporte ao sistema. As configurações de *jobs*, clientes e *filesets* serão tratadas na sequência.

Ainda nesse arquivo encontra-se o campo *Schedule*, onde as regras de agendamento dos *jobs* são criadas, neste cenário virtual, um *backup* completo será realizado no primeiro domingo de cada mês, a cada dia um *backup* incremental será feito, e nos domingos subsequentes, um *backup* diferencial será realizado. As configurações explanadas podem ser visualizadas no Anexo 1 desse trabalho.

Na sequência do arquivo, deve-se configurar os parâmetros do *Storage*, que define o local físico onde os dados serão salvos, sendo que esse não precisa necessariamente estar fisicamente conectado ao servidor Bacula. O *Storage Daemon* possui a variável *Device Type*, que suporta 3 parâmetros: *File*, *Tape* e DVD, que permitem a gravação de dados em arquivos no HD ou similares, fitas e DVDs.

Seguindo o arquivo, as configurações do catálogo são definidas, o *Director Daemon* precisa salvar as informações referentes a cada *backup* ou *restore* realizado, para isso, ele utiliza um banco de dados; neste simulacro, o banco escolhido foi o PostgreSQL, mas também existe suporte para MySQL e SqLite.

Na sequência temos as configurações de mensagens, que podem ser enviadas por e-mail, nesse exemplo, os e-mails são enviados apenas localmente, sendo acessíveis apenas pelo sistema operacional onde o Bacula está instalado.

Após isso temos o *Pool*, que define um conjunto de volumes para o armazenamento dos dados, algumas das variáveis que são definidas aqui são: número máximo de volumes contidos no volume, número máximo de cópias que podem ser feitas, tempo de armazenamento do *backup*, entre diversas outras opções referentes a administração do armazenamento.

Por fim, temos a configuração do console, nesse exemplo, o modo utilizado para acessar o *Director Daemon* foi a console de texto instalada no próprio servidor Bacula, sendo acessada diretamente pelo sistema operacional, esse modo não é o único, também pode-se utilizar a console em um dispositivo separado; atualmente já existem interfaces gráficas para realizar as configurações feitas pela console, esse modo não traz funcionalidades extras, mas facilita seu uso. Neste trabalho, que tem fins acadêmicos, optou-se por utilizar a console, que trata o funcionamento do serviço mais de perto, possibilitando melhor entendimento técnico.

4.1.2 Clientes e Jobs

As configurações de clientes e *jobs,* que são instanciados no arquivo Bacula-dir.conf e definidos através de outro arquivo, o Bacula-dir-clients-and-*jobs*.conf.

Nesse arquivo, é iniciado um *Jobdef,* que segundo Barros (2007) é recurso opcional para instanciar *jobs*, caso uma empresa possua 30 máquinas, sendo elas divididas entre estações Linux e Windows, não será preciso criar um *job* novo para cada cliente, pode-se criar um *jobdef* para cada necessidade e replicá-los para todas as estações que utilizem aquela regra.

Um *Jobdef*, pode ser definido com 4 tipos de operações, as quais são: *Backup*, *Restore*, *Verify* e *Admin*. A função *Verify* compara o conteúdo do catálogo com os arquivos do *backup*, a opção *Admin* faz o *pruning* do catálogo.

Nesse arquivo também definimos configurações como o nível do *backup*, (podendo ele ser completo, incremental ou diferencial), a opção *schedule* vai importar o agendamento definido para este trabalho, o campo *storage* vai definir o dispositivo

físico para gravar os dados e em *mensages* especifica-se onde as mensagens serão enviadas, o campo *Pool* define as regras do *backup*, como tamanho, rótulo, etc.

4.1.3 Fileset

O arquivo de configuração do *fileset* é onde define-se a configuração dos *backups* e *restores* e como esses serão realizados, ou seja, estipula-se campos, como destino dos *backups*, métodos de compressão, etc.

4.2.1 Ambiente virtual Zabbix

Para os testes das funcionalidades do sistema Zabbix em ambiente virtual foram utilizados 3 sistemas operacionais, um sistema Windows, utilizado como cliente, sendo monitorado através do protocolo SNMP; uma máquina virtual com Ubuntu, sendo utilizada como cliente para instalação do agente Zabbix e o sistema Ubuntu sendo utilizado para a instalação do servidor Zabbix.

A instalação do servidor Zabbix no sistema operacional Ubuntu é bastante simples, a própria Canonical, empresa que mantém a distribuição Ubuntu, possui em seus repositórios o pacote da aplicação Zabbix, sendo possível instalá-lo com apenas um comando no terminal do sistema. Também existe a possibilidade de baixar os códigos fonte e compilá-lo, ambos os modos são válidos, tendo como única diferença que através do repositório o sistema já vem pré-configurado e a instalação de *software* de terceiros também é feita automaticamente, além de já importar *templates* para posterior configuração do Zabbix.

Já a instalação através de compilação é mais complexa, mas apresenta a vantagem de ser possível escolher qual a versão exata instalar (algumas vezes os repositórios das distribuições demoram para disponibilizar a versão mais atual do *software*) e quais recursos serão disponíveis, podendo assim, apesar de ser mais trabalhoso, ter uma instalação mais limpa e, portanto, mais eficiente.

Como esse trabalho preocupa-se mais em analisar a ferramenta do que com o desempenho da mesma, a instalação foi realizada através do repositório.

Na parte dos clientes monitorados, foram criados 2 hosts, para o primeiro host foram criados 2 itens analisados, o primeiro analisa se existe conexão entre o cliente e servidor e o segundo a velocidade deles conforme podem ser vistos nas Figuras 10 e 11.

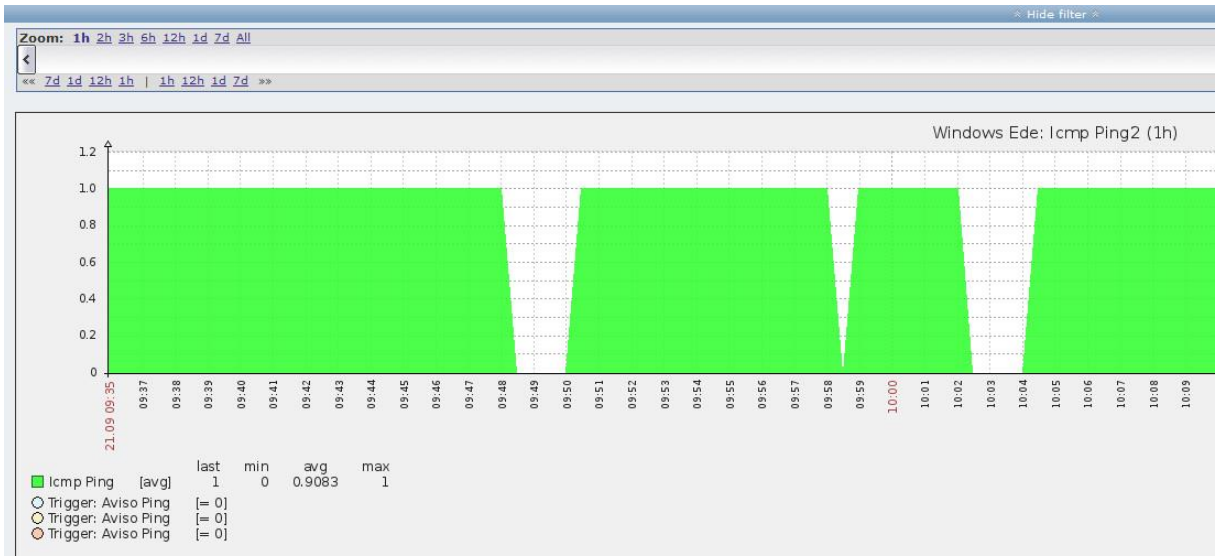

Figura 10 – Ping.
Fonte: Autoria própria.

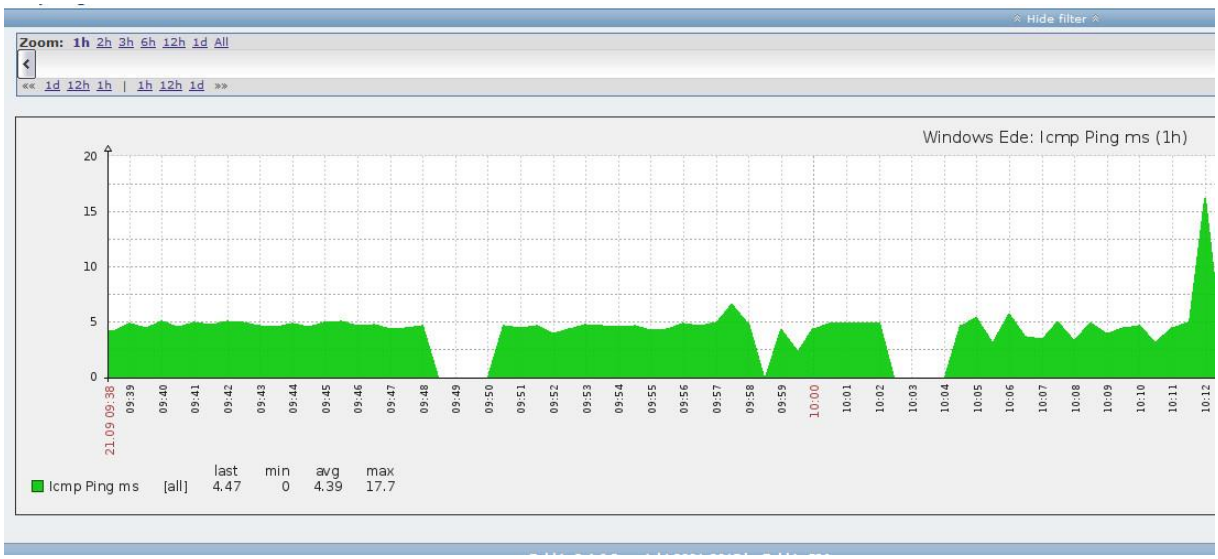

Figura 11 - Ping MS.
Fonte: Autoria própria.

O Zabbix permite uma maleabilidade dos gráficos e alertas muito grandes, pode-se customizá-los para que fiquem da maneira desejada; praticamente qualquer situação pode ser mensurada e alertas de avisos podem ser configurados. Por exemplo, em uma rede interna, a velocidade do tráfego da rede tende a sempre estar

em um nível estável, então faz mais sentido monitorar apenas a disponibilidade de uma determinada estação, conforme Figura 10.

Já no caso de um serviço que utilize da internet, a demora na comunicação pode ser tão grave quanto a sua indisponibilidade, monitorar a velocidade e criar alertas para situações de velocidades críticas é uma solução para esse cenário, conforme Figura 11, por exemplo.

4.2.2 ALERTAS

Para exemplificar o funcionamento dos avisos, foi realizado um monitoramento da comunicação do servidor com uma estação, foram criadas 5 *triggers*, cada uma como uma gravidade superior a anterior e com conforme o tempo sem comunicação aumenta, sobe-se o nível de criticidade do problema, conforme pode-se analisar na Figura 12.

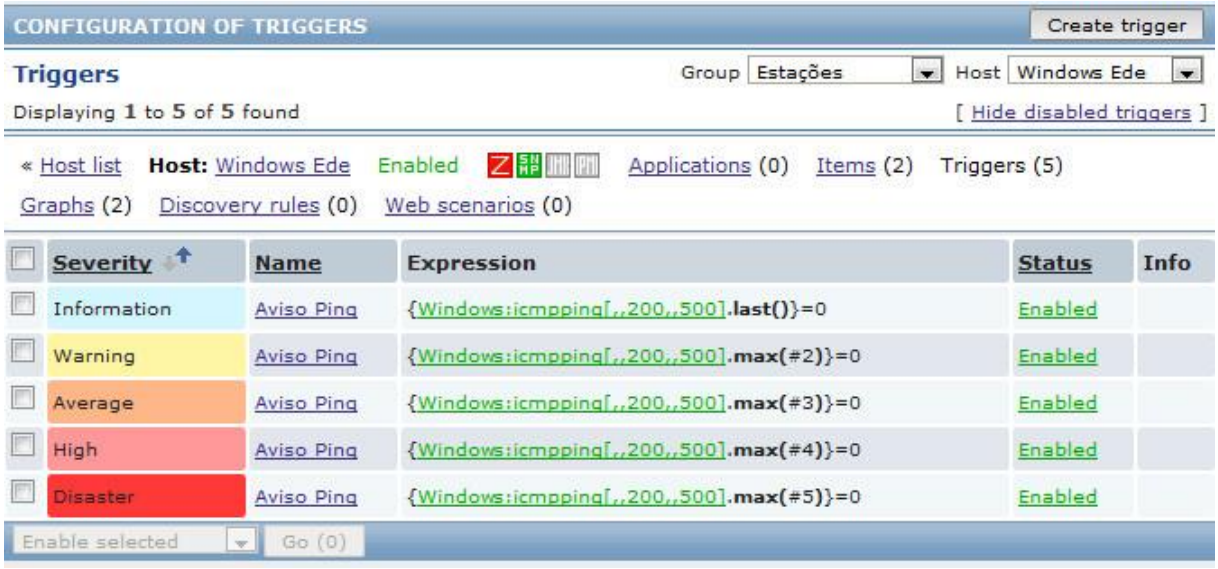

Figura 12 - Avisos
Fonte: Autoria própria.

Caso ocorra a falha de comunicação entre as estações, a primeira *trigger* com classificação de informação será acionada, após isso, a cada período de tempo de permanência do problema a próxima *trigger* com grau maior de severidade será acionada, passando pelas classificações *warning*, *average*, *high* e *disaster*, conforme exemplifica a Figura 13

Figura 13 - Alertas.
Fonte: Autoria própria.

Em uma *trigger* com essa configuração, com a severidade do problema aumentando conforme o tempo em um ambiente real, ela provavelmente não será configurada conforme está apresentado na Figura 13. Neste exemplo, foi optado por manter todos os avisos na tela para fins elucidativos, entretanto, em um ambiente real, a interface fica com entendimento mais claro se o aviso de menor severidade for substituído pelo aviso subsequente de grau maior.

Com as *triggers* definidas, pode-se definir como avisar os indivíduos responsáveis para tratar o problema, neste simulacro, foi escolhido seguir da seguinte maneira:

1	Aviso na dashboard do Zabbix

2	Aviso Sonoro
3	Aviso E-mail
4	Aviso WhatsAPP

No modo 1, o aviso fica conforme já exemplificado na Figura 11. O modo 2, que trata do aviso sonoro, pode ser configurado nas configurações do *profile*, conforme pode ser visto na Figura 12

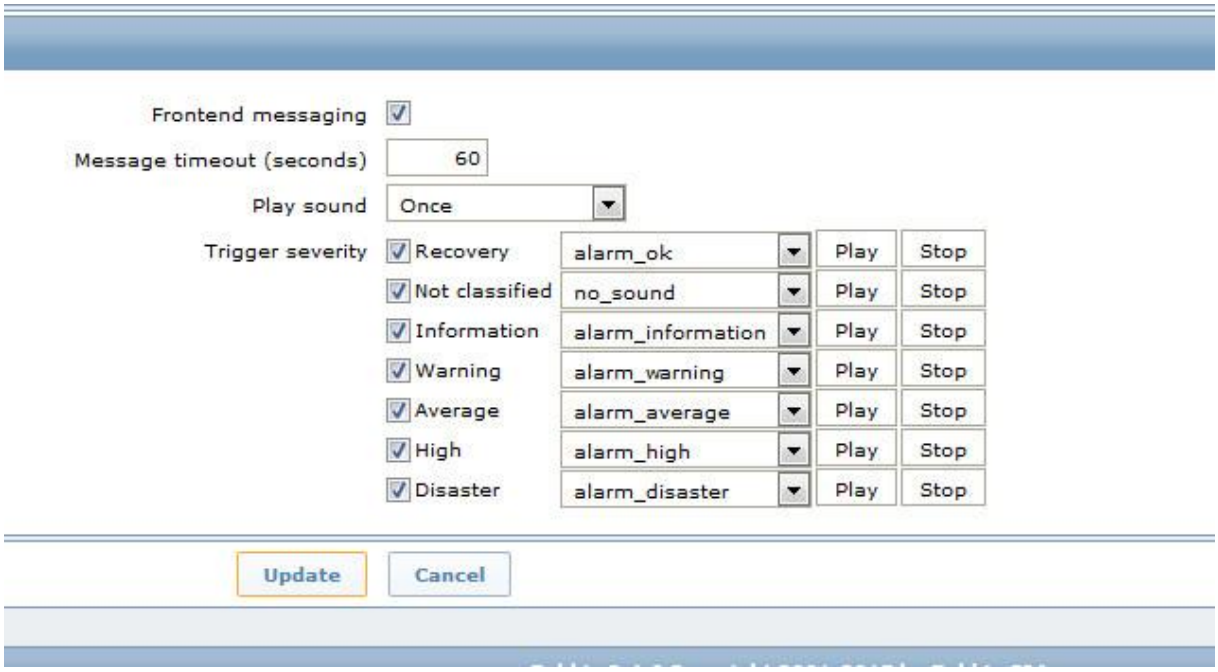

Figura 14 - Sons.
Fonte: Autoria própria.

O aviso por e-mail é o que requer mais trabalho, pois envolve configurar outras ferramentas que serão utilizadas pelo Zabbix para enviar o e-mail. Ainda analisando esse modo, percebe-se que com criatividade pode-se monitorar praticamente qualquer item.

4.3 Ambiente Real

Os conceitos e serviços aqui apresentados foram aplicados em um pequeno consultório odontológico, neste ambiente não existia nenhum tipo de monitoramento e nem política de *backups*.

Ao observar e analisar o cenário, constatou-se que o mesmo possuía 2 estações e um sistema de monitoramento de câmeras, além de acesso à internet através de uma operadora.

Os dados críticos que poderiam afetar o negócio do escritório em questão, no caso de inacessibilidade desses, são os dados dos pacientes e consultas que são mantidas em formato digital em uma das estações e as gravações realizadas pelo sistema de monitoramento que são mantidos na outra estação. Antes de qualquer intervenção, o escritório já realizava uma forma de backup da agenda de consultas, quando ocorre uma modificação na planilha virtual, ocorre também, em uma agenda física, para que se possa atender os pacientes na inacessibilidade do computador que mantém a planilha.

4.3.1 Bacula no Ambiente Real

Nesse ambiente, o Bacula foi configurado de maneira bastante sucinta, sendo um servidor Ubuntu e duas estações Windows. O servidor foi configurado com dois HD’s, um interno e outro externo, o *backup* realizado em ambos é idêntico, tendo como única diferença que o HD interno fica sempre no escritório e realiza *backups* todos os dias (inclusive sábados e domingos), e o HD externo deve ser levado sempre que se fechar o escritório para ser ter uma cópia dos dados externa ao local físico.

A primeira estação, que mantém os dados dos pacientes das consultas teve o agente *Bacula* instalado e no servidor foi configurado a realização do *backup* para ser realizado todos os dias, com exceção de sábado e domingo. Como os dados salvos neste ambiente real de experimento tem tamanhos irrisórios, não são necessários realizar *backups* diferencias ou incrementais. Pela conveniência na restauração, optou-se em realizar o *backup* completo.

O sistema de monitoramento possui um HD próprio onde mantém-se as gravações, e através do *software* instalado na segunda estação, diariamente são feitas as cópias das novas gravações para o computador. Nessa máquina, o agente Bacula também foi instalado e o servidor Bacula foi configurado para realizar o *backup* diariamente. Tem-se assim, 4 locais distintos aonde esses dados são salvos: no HD do sistema de monitoramento, na estação do escritório, e nos HDs interno e externo do servidor Bacula, a Figura 15 exemplifica o fluxo de dados entre os equipamentos.

Figura 15 - Equipamentos.
Fonte: Autoria própria.

4.3.2 Zabbix no Ambiente Real

O monitoramento das estações no ambiente real de experimento foi realizado através dos agentes Zabbix, entretanto nenhum alerta das estações foi criado, dessa forma ficam salvos os registros, por exemplo, da quantidade de trafego de entrada e saída, tempo de utilização, utilização dos periféricos e demais dados da máquina que pode ser consultado pelo proprietário do estabelecimento.

O sistema de câmeras foi monitorado com a função *simple check* do Zabbix e alertas foram criados, caso se passe10 minutos sem conexão, um aviso sonoro é emitido, se em 30 minutos não houver volta da conexão um SMS é enviado utilizando de ferramentas do Google. Durante os primeiros testes esse procedimento funcionou perfeitamente, mas após algum tempo não foi possível mais utilizá-lo, pois o Google não mais permitiu a utilização do Google Calendar para envios de SMS.

O HD externo do *Zabbix* também é monitorado, caso o HD ultrapasse 90 por cento da sua capacidade, um aviso sonoro é emitido.

5 CONCLUSÃO

Nesse capitulo serão apresentadas as dificuldades encontradas, trabalhos futuros e a conclusão do trabalho.

5.1 Dificuldades encontradas

No transcorrer deste trabalho, algumas dificuldades ocorreram quando estudava-se as ferramentas; a princípio, o sistema operacional a ser utilizado como servidor das soluções Zabbix e Bacula seria o Debian, o mesmo chegou a ser utilizado e testado com as soluções, pois ele suporta ambos os *softwares*, apesar disso, durante a instalação e testes dos serviços nesse sistema, distintas dificuldades foram encontradas, tanto na instalação quanto na manutenção e configuração dos serviços, além de certa dificuldade técnica para utilização das ferramentas e recursos do próprio sistema operacional, devido ao exposto, decidiu-se testar o *software* no sistema operacional Ubuntu, no qual a instalação foi mais fácil e sem maiores dificuldades, sendo então, esse adotado para os testes nos ambientes, tanto o real quanto o virtual.

Quanto aos serviços em si, o *software* Zabbix mostrou-se um desafio, em parte devido a sua alta complexidade e maleabilidade; o mesmo possibilita que a configuração seja realizada do zero, definindo-se tudo que deseja ser monitorado e conforme deseja-se, para isso, é preciso configurar muitos passos, como grupo, itens, gráficos, triggers, etc. A princípio foi tentado configurar todos os itens analisados do zero, isso mostrou-se bastante trabalhoso, entretanto, ao utilizar os *templates* e demais configurações já pré-definidas que o próprio programa oferece, ficou mais fácil a utilização, para quase qualquer item que queira-se monitorar em qualquer sistema operacional, seja por SMNP ou pelo agente, é possível encontrar um modelo pré-configurado, com gráficos e alertas criados, esse modo de utilização agiliza bastante a sua utilização, pequenas empresas que venham a utilizar a ferramenta podem usar a configuração dessa forma, pois assim ganharam tempo na implementação do serviço.

Ainda sobre o Zabbix, durante os testes dos avisos e alertas, foi utilizado a ferramenta Calendar do Google para criar um evento na agenda pelo *software* Zabbix,

dessa forma, o Google mandaria um SMS para o celular cadastrado em seu serviço, durante os testes essa forma de enviou funcionou, mas a partir de um determinado momento, o serviço do Google não aceitou mais inclusão de registros na agenda pelo *software* Zabbix, apesar disso, ficou demostrada a possibilidade de através de serviços de terceiros enviar SMS pelo Zabbix. Pode-se utilizar outra empresa ou ferramenta, que forneça o serviço de envio de SMS, gratuitamente ou não, para que seja utilizável pelo Zabbix.

5.2 Conclusão

Com a configuração dos serviços em ambiente real, é possível deduzir que, não apenas o Zabbix e o Bacula, mas também outras soluções em *software* livre já amplamente utilizadas por grandes empresas, podem ser aplicadas também em pequenas instituições. Apesar das ferramentas serem relativamente simples e de intuitiva instalação, usuários não familiarizados com os conceitos de informática teriam dificuldades para configura-la por conta própria em seus negócios.

Uma alternativa para contornar esse problema seria se profissionais com conhecimento prévio das ferramentas se dedicassem a criar empresas para fornecer esses serviços focados nesses pequenos, porém numerosos, clientes.

Acredita-se que com isso pode-se aumentar o valor das instituições como um todo, pois as tornariam mais seguras em relação aos seus dados e como menor tempo de ação para problemas que até então não eram monitorados.

5.3 Trabalhos futuros

Esse trabalho focou-se em dois serviços que em geral, grandes instituições já implementam e pequenas organizações, pelos motivos apresentados, não conseguem implementar ou não conhecem. Complementar a isso, existem outros serviços, já bem difundidos e utilizados por grandes instituições que são ignorados por pequenos negócios, como por exemplo, serviços de proxy, que são importantes para fins de auditoria e controle da internet.

Um novo trabalho complementar a esse poderia focar nesse e em outros serviços ou ferramentas.

Pode-se também, desenvolver uma forma de disponibilizar os serviços apresentados (ou outros) de forma pré-configurada, para que possa ser utilizada da forma mais fácil e rápida possível por pessoas com pouco conhecimento em informática.

REFERÊNCIAS BIBLIOGRÁFICAS

BACKUP. Disponível em http://fernandopsalmeida.blogspot.com.br/2011/11/backup-diferencial-e-backup-incremental.html. 2015. Acesso em 26/07/2015

BARROS, Eurian. **Entendendo os Conceitos de Backup.** ISBN 9788573936292. Editora Ciência Moderna. 2007.

CANONICAL. http://www.canonical.com/products. 2015. Acesso em 07/07/2015.

FERREIRA, E. **Análise e confiabilidade de sistemas redundantes de armazenamento em discos magnéticos**. Dissertação (Mestrado em Engenharia de Computação e Sistemas Digitais) – Escola Politécnica da Universidade de São Paulo, São Paulo, 2003.

FSF. http://www.fsf.org/. 2015. Acesso em 09 de setembro de 2015.

GOBATO, Rudinei Luiz. **Gerência De Redes – Análise Da Ferramenta Zabbix.** Monografia (Aperfeiçoamento/Especialização em Gerenciamento e Configuração de Equip. de Redes) - Universidade Tecnológica Federal do Paraná. 2012.

GOOGLE. https://www.google.com/intx/pt-BR/work/apps/business/products/calendar/. 2015. Acesso em 13/10/2015.

GUISE, Preston D. **Enterprise systems backup and restore: a corporate insurance policy**. Nova York: Auerbach Publications. ISBN 978-1-4200-7639-4. 2009.

JUNIOR, P. D.: **Gerenciamento Centralizado De Backups Distribuídos.** Trabalho de conclusão de Curso – Universidade de Itajaí – Curso de Ciência da Computação. 2010.

KUROSE, J. F., ROSS, K. W.: **Redes de computadores e a internet - uma nova abordagem**, 5a Edição, Addison Wesley. 2010.

MAURO, Douglas R; SCHMIDT, Kevin J. **Essential SNMP, Second Edition.** ISBN:0596008406. O'Reilly Media, 2005.

FARIA, Medrado de; HEITOR. **Bacula - Ferramenta Livre de Backup. Segunda Edição.** Editora: Brasport. ISBN 8574526894. 208 Pgs. 2014.

MEMÓRIA. Disponível em http://www.zabbix.com/zabbix_agent.php. 2015. Acesso 29/07/2015.

MILANI, André. **PostgreSQL – O guia do programador.** Editora Novatec. ISBN 9788575221570. 382 pags. 2008.

MORIMOTO, Carlos E.: **Hardware, o Guia Definitivo.** Editora Sul. ISBN 9788599593165. 1086 pags. 2010.

PINTO, André Viera: **Uma Solução de Monitoramento para Infraestrutura Corporativa de Serviços de TI**. Trabalho final do curso Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro – Unirio. 70 pgs. 2014.

SIBBALD Kern - **Bacula Main Reference**. Disponível em http://www.bacula.org/5.2.x-anuals/en/main/main/Bacula_Main_Reference.html. Acesso em 27/09/2015. 2015.

TOIGO, Jon William. **Recuperação de Sistemas de Informação - riscos, desastres e catástrofes -** Rio de Janeiro, Editora LTC – Livros Técnicos e Científicos. 1990.

VIRTUALBOX. https://www.virtualbox.org/. 2015. Acesso em 02/05/2015

VLADISHEV, A. **Manual Zabbix, Vs 1.6,** 17a Edição. Disp. em http://www.zabbix.com/downloads/ZABBIX%20Manual%20v1.6.pdf. Acesso em 13/08/2015. 2015.

ZABBIX. Disponível em http://www.zabbix.com/zabbix_agent.php. 2015. Acesso em 07/06/2015.

ZABBIX. Disponível em http://www.zabbix.com/zabbix_agent.php. 2015ª. Acesso em 07/06/2015.

ZABBIX. Disponível em http://www.zabbix.com/agentless_monitoring.php. 2015b. Acesso em 07/06/2015.

ZANATTI, Pedro Diego. **Recuperação de dados: estudo de viabilidade de implantação de clínica de recuperação de dados**. Disponível em http://bibliodigital.unijui.edu.br:8080/xmlui/handle/123456789/2380. 52 fls. 2014.

APÊNDICE 1.

bacula-dir-filesets.conf

```
FileSet {
  Name = "Windows"
  Include {
    Options {
      signature = MD5
      compression = GZIP
    }
   File = "C:/Program Files/Bacula"
  }

  Exclude {
  }
}

FileSet {
  Name = "Linux"
  Include {
    Options {
      signature = MD5
      compression = GZIP
      #onefs = no
    }
   File = "/home"
  }

  Exclude {
  }
}
```

```
FileSet {
  Name = "Catalog"
  Include {
    Options {
      signature = MD5
    }
    File = "/var/lib/bacula/bacula.sql"
  }
}
```

APÊNDICE 2.

basic-clients.conf

```
FDPort = 9102
  Catalog = MyCatalog
  File Retention = 30 days            # 30 days
  Job Retention = 6 months            # six months
  AutoPrune = yes                     # Prune expired Jobs/Files
```

APÊNDICE 3.

bacula-dir-clients-and-jobs.conf

```
JobDefs {
  Name = "DefaultJob"
  Type = Backup
  Level = Incremental
  #FileSet = "Full Set"
  Schedule = "WeeklyCycle"
  Storage = File
  Messages = Standard
  Pool = File
  Priority = 10
  Write Bootstrap = "/var/lib/bacula/%c.bsr"
  Allow Duplicate Jobs = no
  Cancel Lower Level Duplicates = yes
}

# 'Job' e Cliente do 'Director'
Job {
  Name = "BackupDirector"
  Client = ubuntu-server-fd
  JobDefs = "DefaultJob"
  Fileset = "Catalog"
}

Client {
  Name = ubuntu-server-fd
  Address = localhost
  Password = "kKHkc5ek-HalhjrNuUeOeTS72_UOqkTyJ"  # password for
FileDaemon
```

```
  @/etc/bacula/basic-client.conf
}

# 'Job' e Cliente do 'Windows'
Job {
  Name = "BackupWindows"
  Client = ede-nb-fd
  JobDefs = "DefaultJob"
  Fileset = "Windows"
}

Client {
  Name = ede-nb-fd
  Address = 192.168.25.12
  Password = "mTLXm26H/FkSPaN7rD/xLRtRE2to4TD94EbF4hYLeSFE"
  @/etc/bacula/basic-client.conf
}

# 'Job' e Cliente do 'Debian'
Job {
  Name = "BackupDebian"
  Client = debian
  JobDefs = "DefaultJob"
  Fileset = "Linux"
}

Client {
  Name = debian
  Address = 192.168.25.83
  Password = "iYFLXDCo0nrpMn7hn3fhIdVnXoQU0WAzJ"
  @/etc/bacula/basic-client.conf
}
# Backup the catalog database (after the nightly save)
```

```
Job {
  Name = "BackupCatalog"
  Client = ubuntu-server-fd
  JobDefs = "DefaultJob"
  Level = Full
  FileSet="Catalog"
  Schedule = "WeeklyCycleAfterBackup"
  RunBeforeJob = "/etc/bacula/scripts/make_catalog_backup.pl MyCatalog"
  RunAfterJob  = "/etc/bacula/scripts/delete_catalog_backup"
  Write Bootstrap = "/var/lib/bacula/%n.bsr"
  Priority = 11                   # run after main backup
}
```

APÊNDICE 4.

```
# Job padrão para a restauração
Job {
  Name = "RestoreFiles"
  Type = Restore
  Client=ubuntu-server-fd
  FileSet="Catalog"
  Storage = File
  Pool = Default
  Messages = Standard
  Where = /nonexistant/path/to/file/archive/dir/bacula-restores
```

APÊNDICE 5.

bacula-dir.conf

```
Director {
Name = ubuntu-server-dir
QueryFile = "/etc/Bacula/scripts/query.sql"
WorkingDirectory = "/var/lib/Bacula"
PidDirectory = "/var/run/Bacula"
Maximum Concurrent Jobs = 1
Password = "SENHA DO CONSOLE"
Messages = Daemon
DirAddresses = { ip = { addr = 127.0.0.1; port=9101;} ip = { addr = 192.168.25.97;
port=9101;}}
}
# Configurações de Clientes e 'Jobs'
@/etc/Bacula/Bacula-dir-clients-and-jobs.conf
# Configurações dos Filesets
@/etc/Bacula/Bacula-dir-filesets.conf
# Agendamento dos Backups
Schedule {
= "WeeklyCycle"
Run = Full 1st sun at 23:05
Run = Differential 2nd-5th sun at 23:05
Run = Incremental mon-sat at 23:05
}
# /Definition of file storage device
Storage {
Name = File
Address = 192.168.25.97
SDPort = 9103
Password = "SENHA DO STORAGE"
Device = FileStorage
Media Type = File
```

```
}
# Generic catalog service
Catalog {
Name = MyCatalog
dbname = "Bacula"; DB Address = ""; dbuser = "Bacula"; dbpassword = "senha1"
43}

# //Reasonable message delivery -- send most everything to email address
Messages {
Name = Standard
mailcommand = "/usr/sbin/bsmtp -h localhost -f \"\(Bacula\) \<%r\>\" -s \"Bacula:
%t %e of %c %l\" %r"
operatorcommand = "/usr/sbin/bsmtp -h localhost -f \"\(Bacula\) \<%r\>\" -s
\"Bacula: Intervention needed for %j\" %r"
mail = root = all, !skipped
operator = root = mount
console = all, !skipped, !saved
append = "/var/log/Bacula/Bacula.log" = all, !skipped
catalog = all
}
Messages {
Name = Daemon
mailcommand = "/usr/sbin/bsmtp -h localhost -f \"\(Bacula\) \<%r\>\" -s \"Bacula
daemon message\" %r"
mail = root = all, !skipped
console = all, !skipped, !saved
append = "/var/log/Bacula/Bacula.log" = all, !skipped
}
# File Pool definition
Pool {
Name = File
Pool Type = Backup
Recycle = yes                    # Bacula can automatically recycle Volumes
```

```
AutoPrune = yes                    # Prune expired volumes
Volume Retention = 365 days        # one year
Maximum Volume Bytes = 1G          # Limit Volume size to something reasonable
Maximum Volumes = 5                # Limit number of Volumes in Pool
Label Format = "vol"
}
# Scratch pool definition
Pool {
Name = Scratch
Pool Type = Backup
}

Console {
Name = ubuntu-server-mon
Password = "SENHA DA CONSOLE"
CommandACL = status, .status
}
```

APÊNDICE 6.

Bacula-dir-clients-and-jobs.conf

```
JobDefs {
Name = "DefaultJob"
Type = Backup
Level = Incremental
Schedule = "WeeklyCycle"
Storage = File
Messages = Standard
Pool = File
Priority = 10
Write Bootstrap = "/var/lib/Bacula/%c.bsr"
Allow Duplicate Jobs = no
Cancel Lower Level Duplicates = yes
}
# 'Job' e Cliente do 'Director'
Job {
Name = "BackupDirector"
Client = ubuntu-server-fd
JobDefs = "DefaultJob"
Fileset = "Catalog"
}
Client {
Name = ubuntu-server-fd
Address = localhost
Password = "kKHkc5ek-HalhjrNuUeOeTS72_UOqkTyJ" # password for FileDaemon
@/etc/Bacula/basic-client.conf
}
# 'Job' e Cliente do 'Windows'
Job {
Name = "BackupWindows"
```

```
Client = ede-nb-fd
JobDefs = "DefaultJob"
Fileset = "Windows"
}
Client {
Name = ede-nb-fd
Address = 192.168.25.12
Password = "mTLXm26H/FkSPaN7rD/xLRtRE2to4TD94EbF4hYLeSFE"
@/etc/Bacula/basic-client.conf
}
# 'Job' e Cliente do 'Debian'
Job {
Name = "BackupDebian"
Client = debian
JobDefs = "DefaultJob"
Fileset = "Linux"
}
Client {
Name = debian
Address = 192.168.25.83
Password = "iYFLXDCo0nrpMn7hn3fhIdVnXoQU0WAzJ"
@/etc/Bacula/basic-client.conf
}
# Backup the catalog database (after the nightly save)
Job {
Name = "BackupCatalog"
Client = ubuntu-server-fd
JobDefs = "DefaultJob"
Level = Full
FileSet="Catalog"
Schedule = "WeeklyCycleAfterBackup"
RunBeforeJob = "/etc/Bacula/scripts/make_catalog_backup.pl MyCatalog"
RunAfterJob  = "/etc/Bacula/scripts/delete_catalog_backup"
```

```
Write Bootstrap = "/var/lib/Bacula/%n.bsr"
Priority = 11                   # run after main backup
}
# Job padrão para a restauração
Job {
Name = "RestoreFiles"
Type = Restore
Client=ubuntu-server-fd
FileSet="Catalog"
Storage = File
Pool = Default
Messages = Standard
Where = /nonexistant/path/to/file/archive/dir/Bacula-restores
}
```