UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SISTEMAS DE DETECÇÃO DE INTRUSÃO VIRTUALIZADOS PARA COMPUTAÇÃO EM NUVEM

LEONARDO SILVA DE MELO

São Luís
2013

LEONARDO SILVA DE MELO

SISTEMAS DE DETECÇÃO DE INTRUSÃO VIRTUALIZADOS PARA COMPUTAÇÃO EM NUVEM

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Zair Abdelouahab, Ph.D.
Coorientador: Josenilson Dias Araújo, M. Sc.

São Luís
2013

Melo, Leonardo Silva de

Sistemas de detecção de intrusão virtualizados para computação em nuvem / Leonardo Silva de Melo. – 2013.

75 f.

Orientador: Zair Abdelouahab
Monografia (Graduação) – Universidade Federal do Maranhão, Curso de Ciência da Computação, 2013.

1. Computação em nuvem. 2. Sistemas de detecção de intrusão 3. Virtualização. I. Título.

CDU 004.738.2

LEONARDO SILVA DE MELO

SISTEMAS DE DETECÇÃO DE INTRUSÃO VIRTUALIZADOS PARA COMPUTAÇÃO EM NUVEM

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Aprovada em: 10 de Dezembro de 2013.

Ph.D. Zair Abdelouahab
Universidade Federal do Maranhão

Dr. Francisco José da Silva e Silva
Universidade Federal do Maranhão

M. Sc. Carlos Eduardo Portela Serra de Castro
Universidade Federal do Maranhão

Ao meu pai, Luzardo, por ser meu referencial de pai, que mesmo estando ausente em alguns momentos, sempre teve seus ensinamentos empíricos atrelados à minha forma de ser e de pensar.

AGRADECIMENTOS

Ao Alfa e o Ômega, o Princípio e o Fim, a Deus em quem eu confio. Porque d'Ele, por Ele e para Ele são todas as coisas; glória, pois, a Ele eternamente.

A minha mãe, Jakeline, por ser uma mãe exemplar que, com seu amor incondicional, deixou muitos sonhos em suspenso para que eu pudesse sonhar também.

Aos meus irmãos, Lohany Hélen, Fernando, Marcos e Lilian, por serem referencial na minha caminhada, me dando apoio e incentivo nos momentos tanto fáceis quanto difíceis.

A minha tia, Jakenes, minha segunda mãe, por todo apoio material e psicológico oferecido desde o início da minha existência.

A todos os meus familiares, por terem contribuído com o apoio e progresso das minhas faculdades mentais subjacentes.

A minha namorada, Fernanda, minha linda, que com seu apoio, carinho, respeito e paciência nos faz melhores. *Forever together.*

Ao meu amigo e coorientador Josenilson, pelo apoio acadêmico e moral oferecido durante toda a concretização deste trabalho.

Ao meu diletíssimo amigo e irmão Fernando Melo pelo exemplo, apoio e motivação, especialmente durante todo este processo de término de curso.

Ao meu irmão Fernando e a meus amigos Daniel Hossoe e Steve Ataky, que me disponibilizaram seus respectivos notebooks para que eu pudesse concluir a escrita deste trabalho.

Ao eminente jovem, exemplo de cidadão internacional, Steve Ataky pelo provimento de diversão enquanto nos deslocávamos de volta às nossas respectivas residências, no ônibus.

Aos meus amigos Jonathan, Artur, Jean, Mário, Steve e Glécio, pelo apoio laboratorial oferecido na elaboração deste trabalho.

Ao amigo Luiz Aurélio, por ter me ajudado na preparação da apresentação e pelas caronas disponibilizadas.

Aos meus confrades do Laboratório de Arquiteturas em Sistemas Computacionais (LABSAC): Profa. Maria da Guia, Mário, Jean, Carol, Josenilson, Mauro, Eduardo, Luiz, Dhully, Gleison, Renato, Cláudio, Wendell, William, Higo, Bruno, Steve e Nilde.

A Catedral de Louvor Maranata em Vinhais e em Maracanã, representados pela Pra. Rutiane e pelo Pr. José Fernando, respectivamente, pelo apoio espiritual e pelas orações à minha família direcionadas.

Ao professor PhD. Zair Abdelouahab, que me acolheu em seu laboratório para participar das pesquisas, mesmo sem o ganho de bolsa, e que pela sua grande experiência acadêmica, sendo um referencial tanto nacional quanto internacionalmente, concedeu-me sua orientação, apoio e paciência.

Aos caros membros da banca, além do meu caríssimo orientador PhD. Zair, os professores Dr. Francisco e M. Sc. Portela, pela contribuição no exame e na disponibilidade de participação na magnífica banca.

A Universidade Federal do Maranhão representada pelos professores do Departamento de Informática que, com suas qualidades e experiências, contribuíram para a minha formação acadêmica.

Por fim, a todos que direta ou indiretamente contribuíram para a realização deste objetivo.

"But seek ye first the kingdom of God, and his righteousness; and all these things shall be added unto you."

(Matthew 6.33)

RESUMO

O contexto da segurança na computação em nuvem tem recebido muito investimento e atenção por parte das empresas e organizações interessadas na utilização desta tecnologia. Sobretudo no que concerne à confiabilidade na virtualização das infraestruturas, na correta utilização dos recursos por políticas de segurança eficazes e na garantia de um armazenamento de dados seguro, muitas técnicas de segurança sobrepujam o esperado para a manutenção de um sistema de nuvem. Não obstante, a utilização indevida das máquinas virtuais disponibilizadas pelos ambientes de nuvem é um fator a ser considerado na adoção de estratégias para a manutenção da disponibilidade dos serviços de nuvem. Refletindo neste aspecto, a adoção de técnicas de detecção de intrusão já exaustivamente adotadas pelas redes de computadores foi adaptada, neste trabalho, para servir de uma ferramenta de segurança adequada para a detecção de atividades maliciosas e tráfegos suspeitos em redes de máquinas virtuais que executam em infraestruturas de computação em nuvem. Como contribuição fundamental, este trabalho apresenta a concretização de um módulo de contramedidas adaptado a um IDS que opera em uma infraestrutura de computação em nuvem, proporcionando a suspensão de máquinas virtuais suspeitas, o que possibilita um tempo hábil para que o administrador da infraestrutura tome as medidas necessárias para a manutenção do ambiente virtual, contatando o usuário, migrando ou reiniciando a VM.

Palavras-Chave: Computação em Nuvem. Sistemas de Detecção de Intrusão. Virtualização.

ABSTRACT

The cloud computing security context has received large investments and attention by enterprises and organizations that are interested in utilization of this technology. Especially with regard to the reliability of infrastructures virtualization, of right resources utilization by effective security policies and of ensuring of secure data storage, many security techniques surpasses the expected for maintaining a cloud system. However, misuse of the available virtual machines by the cloud environment is a factor to be considered in the adoption of strategies to the maintenance of the cloud services availability. Reflecting on this point, adoption of intrusion detection techniques already adopted extensively by computer networks was adapted, in this work, to serve like an adequate security tool to the malicious activity detection and suspicious traffics in networks of virtual machines that execute in cloud computing infrastructures. As a key contribution, this work presents the concretization of a countermeasure module adapted for an IDS that operates on a cloud computing infrastructure, providing the suspension of suspicious virtual machines, which enables timely manner so that the administrator of the infrastructure takes appropriate measures for the maintenance of the virtual environment, contacting the user, migrating or restarting the VM.

Keywords: Cloud Computing. Intrusion Detection. Virtualization.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AaaS	*Attacks as a Service*
API	*Application Programming Interface*
CSA	*Cloud Security Alliance*
DMZ	*Demilitarized Zone*
FN	Falsos Negativos
FP	Falsos Positivos
HIDS	*Host-based Intrusion Detection Systems*
IaaS	*Infrastructure as a Service*
IDS	*Intrusion Detection System*
IPS	*Intrusion Prevention System*
NBAD	*Network Behaviour Intrusion Detection Systems*
NIDS	*Network Intrusion Detection Systems*
NIST	*National Institute of Standards and Technology*
OS	*Operating System*
PaaS	*Platform as a Service*
RFC	*Request for Comments*
SaaS	*Software as a Service*
SLA	*Service Level Agreement*
VM	*Virtual Machine*
VMM	*Virtual Machine Monitor*

SUMÁRIO

1. INTRODUÇÃO

Mainframes, computadores pessoais, arquiteturas cliente-servidor e servidores Web foram precursores do que hoje se denomina computação em nuvem. A possibilidade de utilizar recursos de computação remotos pagando somente pelo uso, podendo ser acessados a qualquer hora e de qualquer lugar de forma segura, são características fundamentais para que a computação em nuvem fosse adotada por grandes empresas.

Contudo, assim como todo sistema computacional, as infraestruturas de computação em nuvem sofrem diversos tipos de ataques por parte de *crackers* e usuários maliciosos, que se aproveitam das vulnerabilidades existentes para executar seus intentos ilegais. Por conta disso, os administradores de servidores de computação em nuvem devem estar atentos às vulnerabilidades reportadas e se prevenirem de possíveis ataques, adotando políticas de segurança tanto a nível infraestrutural quanto a nível de uso, para a correta utilização dos serviços disponibilizados pela nuvem.

De fato, cresce em demasia a quantidade de pesquisas que se tem desenvolvido para o progresso da segurança no contexto da computação em nuvem. Muitas soluções tanto de mercado quanto acadêmicas têm se mostrado eficazes para a prevenção de ataques, utilizando-se de regras de firewall para a implementação de políticas de segurança, *e.g.*, o *Arno's Iptables Firewall* [1]; para a detecção de intrusos, capturando informações de tráfego da rede para posterior análise, *e.g.*, o sistema de detecção de intrusão (IDS) proposto por Araújo [2]; e mesmo para pós-ataques, como os sistemas de prevenção de intrusão (IPS), que armazenam em uma base de dados as características de ameaças previamente determinadas, *e.g.*, o *Virtela's Managed Intrusion Prevention System (IPS) service* [3].

Neste contexto, como base para a concretização do trabalho aqui desenvolvido, o paradigma da computação em nuvem pode ser definido como um modelo de acesso remoto, através de redes de computadores, que possibilita uma utilização ubíqua, conveniente e sob demanda a um vetor de recursos de computação configuráveis, *e.g.* servidores, armazenamento, aplicações e serviços, disponíveis através da internet como serviços, por uma infraestrutura de processamento de dados, permitindo um rápido fornecimento e liberação destes recursos com o mínimo de esforço administrativo ou interação com o provedor de serviços [4].

A característica chave para o conceito de computação em nuvem é a virtualização, que oferece diversos benefícios como a flexibilidade, o isolamento e a alta taxa de utilização dos recursos, a fim de evitar o desperdício na capacidade de hardware [5]. Desta forma, a

virtualização é utilizada na concretização dos serviços de computação em nuvem, pois, além dos benefícios já descritos, oferece proteção e sigilo aos usuários da nuvem.

Portanto, partindo destes fundamentos, pode-se alicerçar um conhecimento sólido a respeito das tecnologias que dão suporte à computação em nuvem e, desta forma, implantar uma tecnologia de segurança em tempo real para a proteção de seus usuários.

Para se concretizar este elemento de segurança, este trabalho utiliza um sistema de detecção de intrusão (IDS) implantado em ambiente virtual, e adiciona-se um componente de reação como parte do IDS, a fim de efetuar contramedidas em caso da existência de atividades maliciosas, para que assim sejam realizados testes e respectivas avaliações deste componente no ambiente virtualizado.

1. 1 Revisão Bibliográfica

Atualmente os trabalhos que tratam diretamente da implantação de sistemas de detecção baseados em ambientes de computação em nuvem se mostram recentes, e consequentemente, os trabalhos que servem de base para a proposta desenvolvida se tornam referenciais de direcionamento para a consecução e implementação do sistema proposto no estudo.

Neste ínterim, o trabalho de Araújo [2] se manifesta como uma contribuição valiosa para o progresso da tecnologia de detecção de intrusão para ambientes virtualizados como os das infraestruturas de computação em nuvem. No trabalho, o IDS é ajustado de tal forma a proteger o ambiente das máquinas virtuais, a fim de que se detectem atividades maliciosas ou comportamentos suspeitos porventura executados pelos usuários internos ao ambiente. Sensores de IDS são implantados nas máquinas virtuais para realizar a detecção, enviando alertas para o administrador do sistema de nuvem, provendo elasticidade e expansão de acordo com o provisionamento dos recursos da nuvem. Destarte, o principal objetivo desta abordagem é evitar que usuários maliciosos se utilizem do alto poder de processamento das infraestruturas de nuvem para realizar ataques tanto à própria quanto a outras máquinas virtuais. No entanto, o trabalho não executa contramedidas para a proteção em tempo real do ambiente, dependendo, assim, da atuação imediata do administrador da rede.

Na referência bibliográfica [6], um ambiente virtualizado de computação em nuvem foi utilizado para implantar um sistema de detecção de intrusão específico para redes de computadores, a fim de observar todo o tráfego da rede virtual, tendo em vista diminuir a perda de pacotes e manipular um grande fluxo de pacotes de dados. Nesta abordagem, o IDS

se utilizou de uma combinação entre redes neurais e clusterização utilizando a lógica *fuzzy* para detectar ataques de acordo com os dados previamente armazenados dos atacantes.

A referência bibliográfica [7] trata de um sistema de detecção de intrusão extensível para um ambiente virtualizado, com um módulo de IDS gerenciador e várias instâncias de IDS, denominados sensores de IDS, em cada máquina virtual. Tais sensores se encarregam de identificar comportamentos maliciosos e gerar alertas através de um componente de reporte para o gerenciador do IDS.

A revisão bibliográfica a respeito dos sistemas de detecção de intrusão para ambientes virtualizados evidenciou que a implantação de ferramentas de segurança já utilizadas para ambientes comuns de redes de computadores está se tornando realidade também em sistemas computacionais virtualizados, tendo em vista a progressiva utilização dos serviços disponibilizados pela computação em nuvem e consequente necessidade de segurança às infraestruturas que a suportam. Sob tal aspecto, este é um tema que necessita ser levado em consideração por analistas de segurança de redes de computadores e administradores de servidores de computação em nuvem, no intuito de que estejam prontos para aprimorar seus ambientes de trabalho por meio do conhecimento de pesquisas que adaptam as tecnologias de segurança já existentes.

1. 2 Motivação

A partir da revisão bibliográfica foi possível observar que os sistemas computacionais que servem de base para a infraestrutura da computação em nuvem são constantemente vulneráveis aos ataques de usuários tanto externos quanto internos a tais sistemas. Por conta disto, muitas estratégias de segurança têm sido implantadas, aprimorando a confiabilidade e a disponibilidade dos serviços.

Neste contexto, os sistemas de detecção, tão amplamente utilizados na segurança de redes de computadores, se tornam úteis para a consecução da detecção de intrusões em ambientes virtualizados de computação em nuvem. Assim, o aprimoramento de tais sistemas de detecção reforça a segurança dos servidores de nuvem, pois aumenta o âmbito de prevenção de ataques e a janela de detecção de intrusão.

Dessa forma, em virtude da realidade da computação em nuvem em aperfeiçoar a sua segurança, torna-se atrativo e necessário a realização de estudos de ferramentas de segurança, tanto passivas quanto ativas, no sentido de detecção e prevenção, respectivamente.

Logo, como uma das ferramentas de segurança, no aspecto da detecção, se faz oportuno o estudo dos sistemas de detecção de intrusão específicos para infraestruturas de nuvem, tendo em vista que a virtualização concretiza o cerne de funcionamento destas, destacando os aspectos fundamentais dos IDS, classificando-os e apontando seus possíveis benefícios para o contexto em tela.

1. 3 Objetivos

O propósito desta pesquisa está centrado no desenvolvimento de um estudo detalhado sobre sistemas de detecção de intrusão virtualizados, para servir de base à detecção de intrusos em ambientes de computação em nuvem, sustentando a escalabilidade e flexibilidade existentes nas infraestruturas de computação em nuvem.

Para alcançar tal objetivo, serão analisados livros e artigos que relacionam os principais atributos da computação em nuvem, das estratégias de virtualização e dos sistemas de detecção de intrusão, almejando os seguintes objetivos:

- Estudar conceitos e aplicações de computação em nuvem, virtualização e sistemas de detecção de intrusão;
- Estabelecer um conhecimento em torno dos principais sistemas de detecção de intrusão que serão utilizados como base para o desenvolvimento dos estudos, principalmente aqueles que se integrem a sistemas virtualizados;
- Realizar o levantamento dos dados relacionados aos principais problemas da segurança no contexto da computação em nuvem;
- Projetar e implementar um estudo de caso consistente na elaboração de um componente de reação no IDS para um ambiente virtualizado;
- Aplicar testes e metodologias de avaliação sobre o componente desenvolvido;

Visando orientar a leitura deste trabalho, a seguir é apresentada a estrutura da monografia que relata os assuntos abordados por cada capítulo.

1. 4 Organização do trabalho

Esta monografia está estruturada da seguinte maneira:

- **Capítulo 1** – INTRODUÇÃO: Apresenta-se a introdução geral do trabalho. Evidencia-se o contexto da computação em nuvem, a problemática da segurança neste contexto e a elucidação da ferramenta de detecção de intrusão para alertar aos administradores de tais sistemas sobre atividades suspeitas.
- **Capítulo 2** – FUNDAMENTOS DA COMPUTAÇÃO EM NUVEM: Neste capítulo são apresentadas as principais características das infraestruturas de computação em nuvem. A computação em nuvem apresentada na forma de suas categorias de serviços e aplicações e modelos de implantação, descrevendo as nuvens privadas, comunitárias, públicas e híbridas, além de elucidar o problema da padronização e portabilidade das nuvens.
- **Capítulo 3** – VIRTUALIZAÇÃO: São destacadas neste capítulo as principais estratégias de virtualização. Tais estratégias são úteis para a configuração dos servidores de computação em nuvem e entendimento dos processos de virtualização intrínsecos a tais infraestruturas.
- **Capítulo 4** – SISTEMAS DE DETECÇÃO DE INTRUSÃO (IDS): Aqui são apresentados os principais pontos relativos aos sistemas de detecção de intrusão, com enfoque em aplicações para os servidores de computação em nuvem. Apresentam-se classificações tanto em relação à forma de detecção, quanto à localização da captura dos dados.
- **Capítulo 5** – SEGURANÇA EM COMPUTAÇÃO EM NUVEM: Neste capitulo é destacada a problemática da segurança da computação em nuvem no contexto hodierno. Apresentam-se conceitos e especificações de vulnerabilidade em ambientes de nuvem e as mais notórias ameaças do âmbito da computação em nuvem.
- **Capítulo 6** – ESTUDO DE CASO: Neste capítulo é concretizado um componente adicional ao sistema de detecção de intrusão, projetado especificamente para sistemas virtualizados que abarcam servidores de computação em nuvem. Realizam-se testes e extraem-se resultados específicos em prol da ratificação da eficiência do componente concretizado.
- **Capítulo 7** – CONCLUSÕES: As conclusões são apresentadas a partir dos capítulos desenvolvidos. Também são apresentadas sugestões para trabalhos futuros relacionados ao tema desta monografia.

2. FUNDAMENTOS DA COMPUTAÇÃO EM NUVEM

Neste capítulo são apresentados conceitos gerais relativos à computação em nuvem e à forma como ela se apresenta no contexto atual, a fim de facilitar e servir de base para a compreensão dos capítulos posteriores.

2.1 Definições

Os mainframes iniciaram a era dos computadores de grande capacidade de processamento. Esta primeira geração, como classificada e descrita por Rajan [8], é marcada por estes grandes computadores, utilizados por organizações de larga escala, a fim de realizar o processamento de uma grande quantidade de dados. Entretanto, eram financeiramente inviáveis de serem adquiridos para uso individual.

Após a era dos mainframes, na segunda geração, os computadores pessoais apareceram como uma solução para o uso individual, porque eram dispositivos de tamanho pequeno e seus usuários não necessitavam de um conhecimento muito especializado para operá-los. Porém, se uma organização resolvesse criar uma aplicação, para que fosse constantemente atualizada para todos os usuários, seria necessário ter uma base de dados e uma interface para cada PC. Isto não somente aumentaria o custo de implementação da aplicação, mas também ficaria complexa e de difícil configuração.

Na terceira geração, a arquitetura cliente-servidor surgiu como uma solução para problema de implementação das aplicações oriundas da segunda geração. Neste tipo de arquitetura, os servidores e os clientes são diferentes entidades, que se conectam por uma rede de computadores. O banco de dados é implantado no servidor e a interface é implantada em cada cliente. Esta estratégia não somente diminui os custos de computação, como também incrementa o desempenho, a escalabilidade e a alta disponibilidade, diminuindo os esforços necessários para a implementação. Contudo, nesta arquitetura, os recursos de infraestrutura são limitados, pois existe uma quantidade máxima de clientes que podem obter acesso ao servidor ao mesmo tempo, além de não prover informações de uma maneira efetiva e distribuída.

A quarta geração, evidenciada pela disponibilização de servidores web, possibilitou o provimento de informações de maneira distribuída. As informações e os dados são armazenados em diferentes servidores, provendo serviços de acordo com a demanda dos

usuários. Contudo, tais sistemas apresentam limitações de recurso, como armazenamento, memória e capacidade de processamento [8].

Em seguida, chegamos à quinta-geração, representada pela infraestrutura de computação em nuvem, disponibilizando um vetor de recursos e aliando os benefícios da computação em grade, paralela e distribuída.

Neste ínterim, o paradigma da computação em nuvem pode ser definido como um modelo de acesso remoto, através de redes de computadores, que possibilita uma utilização ubíqua, conveniente e sob demanda a um vetor de recursos de computação configuráveis, *e.g.* servidores, armazenamento, aplicações e serviços, disponíveis através da internet como serviços, por uma infraestrutura de processamento de dados, permitindo um rápido fornecimento e liberação destes recursos com o mínimo de esforço administrativo ou interação com o provedor de serviços [4].

Neste paradigma, os usuários precisam apenas ter acesso à internet via *browser* e pagam somente de acordo com a demanda dos recursos disponibilizados, que se demonstram infinitos do ponto de vista do usuário. Ou seja, define-se como modelo de negócios da computação em nuvem o pague pelo uso (*pay-as-you-go*). Acrescentado a esta característica, o acesso a qualquer hora e em qualquer lugar, a possibilidade de compartilhamento e colaboração facilitada e o armazenamento seguro dos dados na infraestrutura se apresentam como as principais vantagens na adoção da computação em nuvem [9].

Do ponto de vista dos provedores da nuvem, a elasticidade, propriedade que confere celeridade ao fornecimento e liberação de serviços, a mensuração detalhada do pagamento dos serviços e a capacidade de configuração dinâmica, otimizando a utilização dos recursos, se demonstram como as principais vantagens na manutenção da infraestrutura de computação em nuvem.

Conforme o exposto, apesar das semelhanças entre a computação em nuvem e em grade, tais tecnologias se diferenciam nos seguintes aspectos [10]:

- **Modelo de negócios:** na computação em grade os acordos são bilaterais, realizados entre instituições acadêmicas, enquanto os recursos de nuvem requerem um modelo de negócios diferenciado, permitindo a disponibilização de recursos computacionais a diferentes usuários, a partir de um provedor de serviços de nuvem;

- **Gerenciamento dos recursos:** enquanto a computação em grade depende de sistemas em lote, a computação em nuvem se utiliza das tecnologias de virtualização para o gerenciamento de recursos;
- **Modelos de provisão de recursos:** na computação em grade o modelo de provisionamento de recursos é baseado em organizações virtuais, onde os relacionamentos são mantidos *off-line*, enquanto na computação em nuvem o uso de SLAs[1] se mostra suficiente para a modelagem do provisionamento de recursos.
- **Disponibilidade dos recursos:** em grades, muitas vezes os recursos não estão disponíveis e até mesmo ficam ociosos, enquanto na computação em nuvem busca-se uma massiva elasticidade, tentando encontrar um balanceamento entre o desperdício de recursos, devido à sobrecarga de virtualização e o modo *standby* dos dispositivos, e o agrupamento de recursos, a fim de facilitar a eficiência de consumo dos recursos e reduzir o consumo de energia elétrica.

2.2 Características da Computação em Nuvem

O NIST[2], além trazer uma definição mais específica ao paradigma exposto, também descreve as características e modelos que abrangem a computação em nuvem, *i.e.*, o autosserviço sob demanda, o amplo acesso à rede, o agrupamento dos recursos, a rápida elasticidade e a mensuração do serviço [4].

- **Autosserviço sob demanda:** capacidade do usuário de consumir os recursos da nuvem sem necessitar de um intermediário humano, ou seja, automaticamente. Esse consumo é feito de modo que o usuário tenha facilidade em interagir com a interface e que toda a alocação e liberação dos recursos sejam transparentes ao usuário;
- **Amplo acesso à rede:** os serviços são disponibilizados na rede, de tal forma que os clientes possam acessá-los a partir de quaisquer tipos de plataforma, bastando ter acesso à internet;

[1] *Service Level Agreement*. Contrato em Nível de Prestação de Serviços. Documento formal utilizado para a contratação de um serviço de TI. Nele especificam-se os requisitos mínimos aceitáveis para o serviço proposto.
[2] *National Institute of Standards and Technology*. Agência Federal de Tecnologia dos EUA. Trabalha com a indústria para desenvolver e aplicar tecnologias, medidas e padrões. Disponível em http://www.nist.gov/.

- **Agrupamento de recursos:** capacidade do provedor de serviços em apresentar um modelo de múltiplos serviços, organizados e alocados sob demanda, de tal forma a suportar o fornecimento a múltiplos consumidores. Estes recursos são alocados de forma que o usuário não se preocupe em saber onde eles se localizam fisicamente. Além disso, estes recursos são apresentados hierarquicamente de acordo com o modelo de serviços abordado pela computação em nuvem. A virtualização é a base para o desenvolvimento e a implantação de tais recursos, pois os hipervisores gerenciam a alocação de CPU e memória para os usuários de acordo com a necessidade;
- **Elasticidade:** capacidade de fornecer ou liberar recursos de acordo com a demanda, o que pode dar ao usuário uma ilusão de uma infinidade de recursos;
- **Mensuração do serviço:** possibilidade de medição dos serviços, a fim de cobrar dos usuários a quantia devida de acordo com uma determinada medida específica para cada espécie de recurso consumido.

2.3 Categorias de serviços e aplicações

Como mencionado, a computação em nuvem possibilita largos vetores de recursos disponibilizados na forma de serviços, definindo um padrão arquitetural orientado a serviços, como exibido na Figura 1, adaptada de [11].

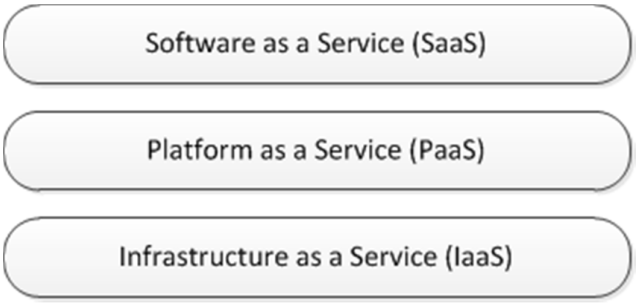

Figura 1. Padrão arquitetural de computação em nuvem.

Tais modelos, definidos pelo NIST [4], são apresentados a seguir.

2.3.1 SaaS

Software as a Service é um modelo de entrega que fornece aos consumidores a capacidade de utilizar as aplicações, com propósitos específicos, executando sobre uma infraestrutura de nuvem. O consumidor não gerencia ou controla a infraestrutura de nuvem subjacente, tais como rede, servidores, sistemas operacionais, armazenamento ou mesmo configurações da aplicação atual, mas apenas utiliza o software que se encontra concretizado como um serviço web.

A Figura 2, adaptada de [12], mostra o funcionamento de uma aplicação SaaS fornecida pelo provedor de serviços, que entrega o software como um serviço web, assumindo a responsabilidade de atualizar a aplicação e mantê-la disponível aos consumidores.

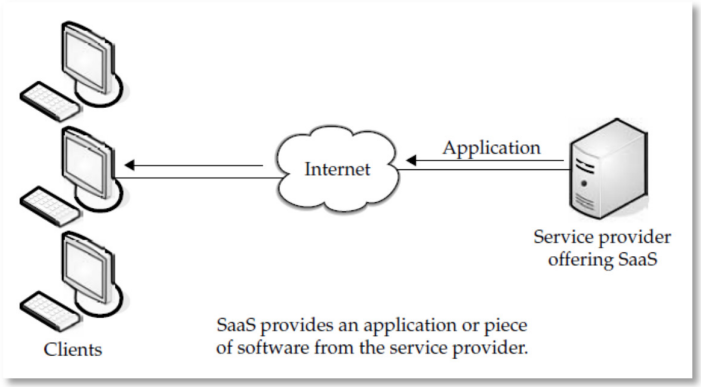

Figura 2. Fornecimento de SaaS por um provedor de serviços [12].

O SaaS possibilita que os provedores de nuvem se beneficiem de uma instalação e manutenção simplificada de software, além de ter um controle centralizado sobre a versão [9]. Desta maneira, novos recursos podem ser unificados aos sistemas de software, sendo transparentes para o usuário, além de se ter a redução dos custos com licenças e com instalações de novas atualizações de software.

Google Apps [13], Basecamp [14] e Dropbox [15] são exemplos de SaaS.

2.3.2 PaaS

Platform as a Service, também denominado *cloudware* [12], é outro modelo de entrega que fornece aos consumidores uma plataforma para que possam criar seus próprios

softwares tendo a própria infraestrutura da nuvem como subjacente, usando linguagens de programação, bibliotecas, serviços e ferramentas suportadas pelo provedor. Desta forma, o usuário não precisa fazer *download* ou instalação das ferramentas necessárias ao desenvolvimento.

Assim como em SaaS, o consumidor não gerencia ou controla a infraestrutura de nuvem, mas tem controle sobre as aplicações implantadas e possivelmente sobre as configurações do ambiente respectivo a partir do qual o software está sendo executado.

A Figura 3, adaptada de [12], mostra o funcionamento de uma plataforma fornecida pelo provedor de serviços PaaS, que entrega a plataforma de desenvolvimento como um serviço web. Os serviços PaaS fornecem a possibilidade de projetar, desenvolver, testar, implantar e disponibilizar a aplicação na web por meio de um serviço de *host*.

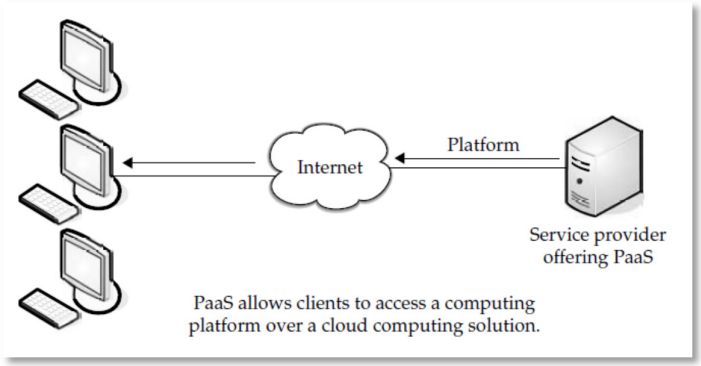

Figura 3. Fornecimento de PaaS por um provedor de serviços [12].

OpenShift [16], Google App Engine [17] e Postgres Plus Cloud Database [18] são exemplos de PaaS.

2.3.3 IaaS

Infrastructure as a Service, também denominada de HaaS (*Hardware as a Service*), é outro modelo de entrega de serviços que disponibiliza aos consumidores a capacidade de processamento, armazenamento, rede e outros recursos fundamentais de computação, como recursos virtualizados, onde o consumidor é apto a implantar e executar quaisquer softwares do seu interesse, incluindo sistemas operacionais.

Assim como nos modelos anteriores, o consumidor não é habilitado a gerenciar ou controlar a infraestrutura da nuvem subjacente. Contudo, pode controlar o sistema operacional implantado, a estrutura de armazenamento e as aplicações implantadas, e possivelmente pode ter controle limitado em selecionar os componentes de rede porventura necessários para a atividade requerida.

A Figura 4, adaptada de [12], demonstra a disponibilização dos recursos entregues pelo IaaS, como processamento de dados, ciclos de CPU, memória e armazenamento.

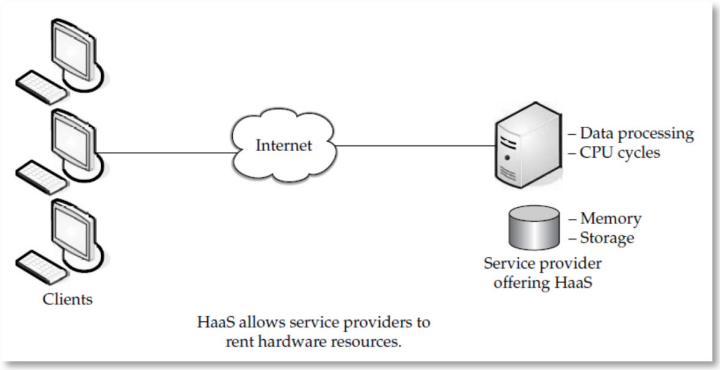

Figura 4. Fornecimento de IaaS por um provedor de serviços [12].

Logo, tal modelo de serviços é o cerne do que se denomina “nuvem” em sentido estrito, pois se trata da concretização da infraestrutura física deste paradigma, a partir da virtualização dos recursos computacionais. Tal virtualização é utilizada como uma forma de otimizar a utilização dos recursos dos *datacenters* e permitir a existência da escalabilidade do modelo sobre exame.

Eucalyptus [19], OpenStack [20] e Windows Azure [21] são exemplos de IaaS.

2.4 Modelos de Implantação

No que tange à maneira com que a infraestrutura da nuvem pode ser estabelecida, determinando e limitando o acesso e a utilização a determinado grupo de usuários, o NIST realizou uma classificação com relação ao modelo de implantação, pelo qual as infraestruturas de nuvem podem apresentar. Compõe tal classificação as nuvens privadas, comunitárias, públicas e híbridas [4].

2.4.1 Nuvens Privadas

Nuvens privadas são infraestruturas proprietárias de nuvem, que fornecem serviços baseados em computação em nuvem a múltiplos usuários de uma única organização. A posse, o gerenciamento, a manutenção e a operação da infraestrutura podem tanto ser realizadas pela própria organização (*on-premisse*) quanto por empresas terceirizadas (*off-premisse*), ou uma combinação destas.

Tais nuvens privadas apresentam maior segurança e controle, pois são implantadas em um ambiente de rede de computadores privado. Além de prover um ambiente bem gerenciado, otimiza o uso dos recursos computacionais, suportando específicas cargas de trabalho e provendo um autosserviço de recursos de hardware e software para a organização. Entretanto, o gasto com a manutenção e o pagamento de funcionários para manter a infraestrutura funcionando acabam reduzindo o lucro da organização [22].

2.4.2 Nuvens Comunitárias

Nuvens comunitárias são infraestruturas de nuvem utilizadas por um determinado grupo de usuários de organizações que possuem atividades compartilhadas. São gerenciadas, mantidas e administradas por uma ou mais organizações que fazem parte da comunidade.

2.4.3 Nuvens Públicas

Nuvens públicas são infraestruturas de nuvem disponibilizadas para uso ao público em geral. São mantidas, administradas e gerenciadas por empresas, universidades ou organizações governamentais, ou uma combinação destas. Os equipamentos ficam localizados fisicamente no próprio local onde se situa o provedor da nuvem.

Tais nuvens públicas são mais vulneráveis a ataques que as nuvens privadas, pois os *datacenters* que suportam a infraestrutura da nuvem não se encontram nos domínios geográficos e administrativos da organização que a adotar. Porém, é a melhor escolha quando se trata de disponibilizar serviços para vários usuários, para desenvolver e testar aplicações, quando se deseja capacidade de computação adicional ou para elaborar projetos colaborativos [22].

2.4.4 Nuvens Híbridas

Nuvens híbridas são infraestruturas de nuvem provenientes da combinação de dois ou mais modelos de implantação outrora mencionados (privada, comunitária ou pública). Bastante utilizadas por tecnologias padronizadas ou proprietárias, que suportam a portabilidade de dados e aplicações entre nuvens distintas.

Um exemplo de utilização desta espécie de modelo de implantação é o denominado *Cloudbursting*, o qual permite a ampliação da capacidade computacional da nuvem combinando seus recursos com o de outra, permitindo o balanceamento de carga entre nuvens [23].

2.5 *Cloud lock-in* e padronização

Uma desvantagem nas categorias de serviços e aplicações está na falta de interoperabilidade e portabilidade entre provedores de nuvem, de tal forma que um programa que é criado em uma determinada plataforma não se torna possível de ser executado em outra plataforma, ou para isso é exigido um determinado preço [12].

Este alto risco de aprisionamento forçado, denominado *cloud lock-in*, e a falta de padronização, são desvantagens para que as empresas possam confiar a disponibilização dos seus serviços por meio de infraestruturas de nuvem.

O problema do *lock-in* está em que cada nuvem possui sua própria implementação de APIs[3] e de banco de dados, se utilizando de diferentes linguagens de programação e de recursos proprietários, dificultando a interoperabilidade e a migração de dados entre nuvens diversas [24].

A interoperabilidade requer a translação de aplicações específicas e de funcionalidade de serviços entre nuvens, e para que esta translação seja facilitada, é necessária a padronização da computação em nuvem. Tal padronização depende principalmente das tecnologias de virtualização, pois nuvens que se utilizam de hipervisores diferentes apresentam uma interoperabilidade dificultada, em parte porque eles não se utilizam de um mesmo formato de dados [25].

[3] *Application Programming Interface.*

As plataformas de nuvem também não se tornam interoperáveis porque suas VMs[4] não interagem de forma padrão com diferentes arquiteturas de redes e armazenamento, APIs, conexões de rede, bancos de dados e outros elementos [25].

Tendo em vista estas questões, surgiram as OpenAPIs, APIs de código aberto, que são camadas de abstração que proporcionam interoperabilidade, protegendo as aplicações de incompatibilidades entre provedores de nuvem. As OpenAPIs funcionam de tal forma a realizar a tradução automática de aplicação entre nuvens diferentes a fim de possibilitar a compatibilidade. Temos exemplos de OpenAPIs a Deltacloud [26], a Apache Libcloud [27] e a Simple Cloud [28].

A preocupação com o mercado a respeito do *cloud lock-in* levou à escrita do Open Cloud Manifesto [29], que estabelece uma especificação de um padrão aberto para nuvens. Tal manifesto estabelece desafios, objetivos e princípios necessários para nuvens abertas, possibilitando uma liberdade de escolha para os usuários, de tal forma que estes não fiquem restritos a recursos proprietários e ao aprisionamento de nuvens.

Muitas empresas já aderiram ao Open Cloud Manifesto, como a Adobe [30], a Hewlett Packard [31] e a IBM [32].

[4] *Virtual Machines.* Máquinas Virtuais.

3. VIRTUALIZAÇÃO

A característica chave para o conceito de nuvem é a virtualização, que oferece diversos benefícios como a flexibilidade, o isolamento e a alta taxa de utilização dos recursos [5]. Por conta da relevância das tecnologias de virtualização, e após descrevermos a abordagem realizada sobre computação em nuvem no capítulo anterior, explana-se neste capítulo uma breve exposição dos conceitos relacionados à virtualização.

3.1 Definições

O conceito de virtualização é antigo, originando-se na década de 1960, com a criação do OS experimental M44/44X, pela IBM©. Na década de 1980, o PC tornou-se acessível pelo barateamento do hardware, de tal forma que a virtualização ficou esquecida. Já no início dos anos 90, com o crescimento da capacidade de processamento e armazenamento do hardware, a virtualização se tornou viável novamente [33].

A base para a implementação da virtualização é o hipervisor, denominado monitor de máquinas virtuais (VMM), que tem a funcionalidade de criar uma camada de virtualização, permitindo a uma única máquina física o suporte ao funcionamento de múltiplas máquinas virtuais [5].

Existem três técnicas de virtualização, que são a virtualização total, a para-virtualização e a virtualização em nível de sistema operacional, conforme descrito a seguir.

3.2 Estratégias de virtualização

A virtualização total (*Full Virtualizatio*n), ilustrada pela Figura 5, adaptada de [5], oferece uma simulação fiel de hardware correspondente exatamente ao hardware real, de tal forma que o sistema operacional convidado (G*uest OS*) não precisa ser modificado para que execute sobre a camada de virtualização, que se encontra sobre o sistema hospedeiro (*Host OS*). A desvantagem está em que os sistemas virtualizados, a partir desta técnica, executam mais lentamente do que se fossem executados em máquinas reais, pois toda chamada ao hardware real é intermediada pelo hipervisor [33].

Alguns autores, como em [34], introduzem uma classificação da virtualização completa, distinguindo a implementação da camada de virtualização (VMM) direto no hardware da instalação, do hipervisor sobre o sistema operacional hospedeiro.

Em suma, este tipo de virtualização fornece isolamento e segurança para as VMs e simplifica a migração e a portabilidade entre as mesmas [5]. Exemplos de hipervisores que adotam esta estratégia são QEmu [35], VMWare [36] e VirtualBox [37].

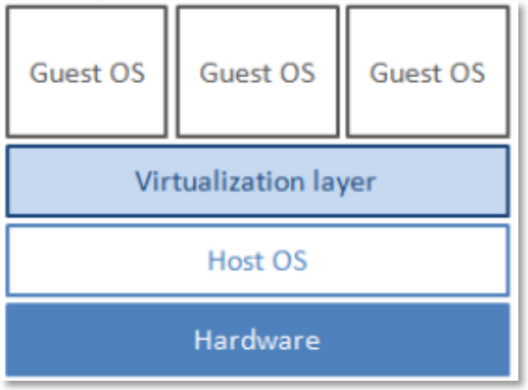

Figura 5. Arquitetura da Virtualização Completa [5].

A para-virtualização (*Para-Virtualization*), ilustrada pela Figura 6, adaptada de [5], providencia a simulação de VM semelhante, mas não idêntica ao hardware subjacente [5]. Nesta estratégia de virtualização é inserido um hardware virtual, que funciona como um acoplamento otimizado entre o sistema convidado e o hipervisor, fazendo com que o *Guest OS* execute seus processos com um desempenho mais elevado do que o fornecido pela virtualização total. Contudo, para isso se faz necessário que haja modificações no sistema convidado para que ele execute na plataforma virtual [33].

Apesar da necessidade dessa adaptação afetar na portabilidade, os sistemas convidados ficam habilitados a acessar recursos diretamente do hardware, sendo apenas monitorados pelo hipervisor, que gerencia a alocação dos recursos como memória e discos. Neste caso, o sistema convidado também tem acesso direto aos dispositivos, como mouse e teclado, sendo apenas gerenciado pelo hipervisor, no caso de múltiplas máquinas virtuais, para a prioridade e ordem de acesso.

Xen [38] e Hyper-V [39] são exemplos de sistemas que fornecem a para-virtualização.

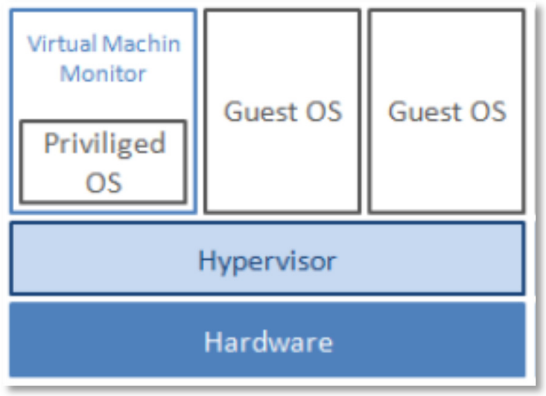

Figura 6. Arquitetura da Para-Virtualização [5].

A virtualização em nível de sistema operacional (*OS-level Virtualization*) ilustrada pela Figura 7, adaptada de [5], oferece um ambiente de isolamento de recursos entre usuários. Em vez de instanciar múltiplas máquinas virtuais que são cópias do mesmo *kernel* do OS, ele oferece um contexto de processo isolado dentro de um único *kernel*. Tal estratégia existe apenas em sistemas Linux.

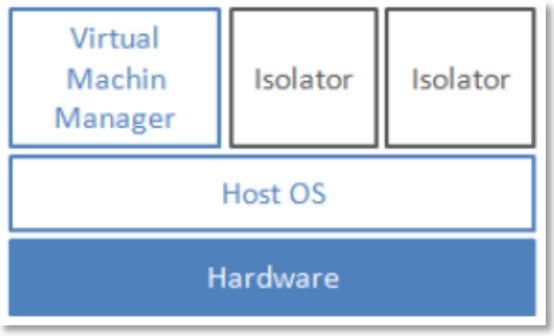

Figura 7. Arquitetura da Virtualização em Nível de Sistema Operacional [5].

Portanto, a conceituação das várias estratégias de virtualização permite fornecer uma visão geral da execução das propriedades dos hipervisores em diversos ambientes, podendo tais estratégias ser amplamente utilizadas para aplicações de segurança em arquiteturas de sistemas computacionais. E neste âmbito, o paradigma de computação em nuvem se utiliza das máquinas virtuais para oferecer, além da escalabilidade, o isolamento para com as outras máquinas virtuais do mesmo hipervisor, sendo um dos fatores utilizados para a segurança da infraestrutura.

No capítulo posterior descreveremos os sistemas de detecção de intrusão, evidenciando as definições e classificações, focando em sua aplicação no contexto de virtualização, a fim de evidenciar seus benefícios às infraestruturas da computação em nuvem.

4. SISTEMAS DE DETECÇÃO DE INTRUSÃO (IDS)

Baseado nas características da computação em nuvem e na definição de virtualização já abordadas, introduzimos neste capítulo uma das tecnologias bastante utilizadas na área de segurança de redes de computadores, os sistemas de detecção de intrusão. Igualmente denominados *Intrusion Detection System* (IDS), tais sistemas se mostram muito úteis na detecção de intrusão em ambientes que suportam a infraestrutura da computação em nuvem, elaborando relatórios de ataques e reportando-os aos seus respectivos administradores.

Considerando tais aspectos, este capítulo delineia o conceito de IDS, abordando suas características, classificações e aplicações. Não obstante, tal capítulo tem a função de introduzir os conceitos desta tecnologia para que possam ser implantados em ambiente virtualizados de computação em nuvem.

4.1 Definições

De acordo com o NIST [40], o IDS é um software que automatiza o processo de detecção de intrusão, monitorando os eventos que ocorrem em um sistema computacional ou em uma rede e analisa-os a fim de alertar sobre possíveis incidentes, *i.e.*, violações ou ameaças iminentes de violação de políticas de segurança computacionais, políticas aceitáveis de uso, ou práticas padronizadas de segurança.

Desta forma, os IDS são partes de software ou dispositivos físicos que se encontram implantados em um ambiente controlado de redes de computadores, mais especificamente nos *hosts*, na própria rede ou em ambos, funcionando na detecção de atividades indesejáveis provenientes tanto de usuários maliciosos internos quanto externos. As atividades consideradas indesejadas passíveis de serem detectadas são, *e.g.*, eventos ilícitos ou tráfegos maliciosos de dados, que violam as políticas de segurança ou que violam as políticas aceitáveis de uso [41].

Os dados oriundos da detecção podem ser armazenados em bancos de dados ou na forma de *logs* de eventos, para que possam ser comparados com outros dados de auditorias anteriores, a fim de validar a detecção. Tal validação tem as funções de reconsiderar as políticas de segurança ou de emitir um índice de severidade da intrusão, conforme o propósito do ambiente o qual o IDS está situado.

Outrossim, tais sistemas podem até mesmo servir como sistemas reacionais, executando contramedidas, de acordo com o tipo, a severidade e a relevância do ataque, desde que assim habilitados e na medida em que o administrador do ambiente da rede permitir. Ao ser configurado desta forma, o NIST, em [40], classifica o IDS como sendo um *Intrusion Prevention System* (IPS), identificando-o como um sistema que realiza a detecção de intrusão que atenta a interromper incidentes possivelmente detectados. Cumpre salientar que neste trabalho será feito um estudo de caso, expondo a execução de contramedidas, de acordo com determinado ataque, utilizando-se do ambiente proposto em [2].

A detecção pode ser feita por meio de *signature matching* ou *misuse detection*, *i.e.*, detecção por assinatura, ou por meio de detecção de anomalia, que pode ser realizada tanto pela auditoria de tráfego anômalo de dados quanto pela verificação do comportamento malicioso de usuários internos.

4.2 Classificações

No que diz respeito à forma de detecção de intrusão, aos ataques advindos desta, e à localização do IDS no seu âmbito de atuação, podemos descrever uma categorização suficiente para o esclarecimento da sua funcionalidade.

Em [42] é apresentada uma extensa taxonomia de IDS, elaborando classificações com relação à fonte dos dados, ao processamento, ao tempo de detecção, ao ambiente, à arquitetura, à reação e aos alertas. Contudo, para este trabalho, nos ateremos somente à fonte dos dados e à forma pela qual a detecção se processa.

Tais classificações decorrem das formas de configuração do IDS na rede, e para cada classificação existem vantagens e desvantagens, *i.e.*, o custo de configuração e manutenção, a utilização dos recursos no qual está inserido, a limitação fixada pelas políticas de segurança, a privacidade dos dados do usuário e a eficiência da detecção em si.

4.2.1 Quanto à forma de detecção

No que diz respeito à maneira com que o IDS identificará os ataques, temos a detecção por assinatura e por anomalia.

A detecção de intrusão baseada em assinatura é o método mais eficiente no propósito de identificar e reconhecer ataques previamente definidos, disponíveis em uma base de dados de assinaturas. Uma assinatura é um padrão de ataque em redes de computadores,

que pode ser utilizada para detectar ataques que se encaixam nesse mesmo padrão, que porventura possam ocorrer dentro de um ambiente de rede.

Assim, as regras são um conjunto ordenado de assinaturas, que funcionam de forma a expandir a capacidade de detecção de ataques ou ameaças dentro de um sistema computacional ou de uma rede. Podem ser concretizadas pelo uso de *honeypots*, obtendo-se conhecimento sobre ferramentas e técnicas as quais o invasor pode utilizar na intrusão [43].

Na Figura 8, adaptada de [43], tem-se um exemplo de uma regra de ataque utilizada pelo Snort [43]. No caso em destaque, a regra permite a emissão de alertas para todos os pacotes TCP com TTL[5] = 100, que se direcionam ao servidor 192.168.1.10, à porta 80, a partir de qualquer fonte, ou seja, de qualquer endereço IP e de qualquer porta.

```
alert tcp any any -> 192.168.1.10/32 80 (msg: "TTL=100"; \ttl: 100;)
```

Figura 8. Exemplo de regra de detecção de ataque utilizada no Snort [43].

O método de detecção por assinatura utiliza a comparação de assinaturas (*signature matching*) contidas em uma base de dados, com os dados que trafegam no ambiente de rede, a fim de reconhecer a ocorrência de um ataque recente. Com o surgimento de novos ataques, é necessário que a base de dados seja atualizada constantemente, pois, se isso não ocorrer, a taxa de falsos negativos pode aumentar com o passar do tempo e, consequentemente, comprometer a segurança do ambiente. Além disso, uma pequena modificação na assinatura de determinado ataque pode prejudicar a detecção, pois o IDS não a reconhecerá como sendo de um ataque.

Como nenhum sistema é perfeito, e o IDS não é exceção, podemos observar alguns casos de erros na detecção e, a partir destes casos, classificá-los em falsos positivos (FP) e falsos negativos (FN). Os FP acontecem quando o IDS considera erroneamente como ataque um tráfego benigno, enquanto os FN são considerados um caso pior do que os FP, pois, neste caso, o tráfego maligno não é detectado [41].

De acordo com [44], citado por [45], recentemente foi realizada uma coleta de vários FP e FN em um ambiente de tráfego de rede real e se obtiveram três conclusões a respeito destas falhas. Primeiramente, grande parte dos casos de falso foram FN, porque o comportamento das aplicações e seus conteúdos apresentam um formato próprio, não

[5] *Time To Live*. "Tempo de vida" de um datagrama, ou seja, a quantidade máxima de nós (comutador de pacotes) que o datagrama caminhará até chegar ao seu destino, sendo descartado ao chegar em zero.

especificado pelos padrões RFC[6]. Segundo, muitos dos casos de FP são oriundos de políticas de gerenciamento e não por motivo de segurança. Terceiro, existe uma grande porcentagem de FN de ataques antigos que ainda não são devidamente reconhecidos.

Como descrito inicialmente, a técnica de detecção de intrusão por assinatura se mostra como a de maior acurácia, pois apresenta pouca porcentagem de erros; de maior celeridade, pois é realizada apenas uma operação de comparação; e com pouca geração de FP, porque as assinaturas são bem definidas para cada tipo de ataque.

Exemplos de IDS que utilizam a abordagem de detecção por assinaturas são o Snort [46], que possui uma grande base de assinaturas disponível na internet, o NetShield [47], que alcança uma maior acurácia pela utilização de operadores de expressões regulares em suas assinaturas e o Kargus [48], que se denomina um IDS baseado em assinatura altamente escalável, apresentando uma performance comparável a um IDS baseado em hardware, aproveitando as inovações de CPU *multicore* e GPU[7] heterogênea, sendo capaz de apresentar uma alta taxa de monitoramento, chegando a ser em tempo real.

Enquanto isso, a detecção por anomalia observa o tráfego da rede, a formação dos pacotes IP e o comportamento do usuário na utilização dos recursos. A anomalia, neste contexto, deve ser entendida como um distúrbio comportamental, observada com relação tanto ao tráfego da rede quanto à forma de utilização dos recursos pelo usuário [41].

Este método de detecção tanto verifica se um pacote IP foi malformado com base nas regras de formação especificadas, quanto utiliza tecnologias de reconhecimento de padrões ou adota limiares específicos, para indicar com determinada precisão a probabilidade de um tráfego analisado ser considerado um ataque. O reconhecimento de padrões também é utilizado para detectar comportamento suspeito de determinado usuário com base em um perfil previamente estabelecido para o mesmo.

A elevada taxa de FP é uma desvantagem da detecção baseada em anomalia, pois o IDS pode considerar o tráfego benigno ou o comportamento não nocivo do usuário, como um novo ataque. Sem embargo, este método de detecção tem a vantagem de criar novas assinaturas de ataques e não precisa de atualização do banco de dados de assinaturas.

Muitos trabalhos relacionados demonstram como um IDS pode ser utilizado com base em diversas técnicas de detecção. A partir da experiência obtida na utilização dos IDS

---

[6] *Request For Comments*. Série de documentos contendo notas técnicas e organizacionais sobre a internet e seus protocolos. Disponível em http://www.ietf.org/rfc.html.

[7] *Graphics Processing Unit*. Unidade de Processamento Gráfico. Microprocessador com uma estrutura altamente paralela. Realiza o processamento de computação gráfica com maior efetividade que as CPUs genéricas.

em ambientes tanto acadêmicos quanto industriais ou empresariais, surgiram tendências que vieram confirmar a alta eficiência da abordagem do sistema.

Tais tendências afirmam que a detecção baseada em assinatura está sendo considerada obsoleta. Os trabalhos [49] e [50] demonstram essa desvantagem em relação às outras técnicas de detecção, pois ela pode abrir brechas para várias formas de evasão de IDS, além do fato de as assinaturas existentes não servirem mais para a descoberta de novos ataques.

Um dos diversos motivos para a obsolescência da detecção por assinatura está na crescente quantidade de smartphones que estão se conectando às redes de computadores, tornando os recursos de armazenamento e de bateria insuficientes para suportar a detecção por assinatura em tais dispositivos. Logo, a detecção por anomalia está tomando espaço frente às novas tecnologias, apoiada nas técnicas de reconhecimento de padrões ou baseadas em dados estatísticos [50].

Além destes aspectos, o crescimento dos alvos de ataque e a comunicação criptografada não permitem que a detecção por assinatura atue de maneira adequada. Neste aspecto os servidores se incluem, pois a grande quantidade de dados que trafegam nas redes atuais não torna viável a detecção por assinatura, porque muitos dados devem ser analisados de acordo com milhões de assinaturas, o que causa um *delay* na detecção, dificultando a política da detecção em tempo real [50].

4.2.2 Quanto à localização da captura dos dados

Quanto à localização do IDS no ambiente de rede, podemos classificar em NIDS (*Network Intrusion Detection Systems*), HIDS (*Host-based Intrusion Detection Systems*) e Hybrid IDS.

O NIDS tem como objetivo detectar eventos maliciosos e indesejáveis presentes no tráfego de dados da rede. Tem como componentes [41]:

- **Sensores:** instalados em locais estratégicos da rede, realizando a captura dos pacotes de dados;
- **Banco de dados:** armazenando os pacotes capturados considerados maliciosos ou suspeitos;
- **Central de gerenciamento:** recebendo pacotes provenientes de todos os sensores, analisando tais pacotes de acordo com determinada técnica estudada

na seção anterior, e enviando pacotes oriundos de tráfego malicioso ao banco de dados;

- **Console:** permitindo ao administrador da rede obter o controle sobre a configuração e a manutenção do NIDS.

O HIDS se localiza no *host*, ou seja, na máquina física ou virtual conectada à rede, a fim de analisar seus *logs* de dados e perfis de usuário. Tem como principais funções analisar código malicioso, detectar sobrecarga de *buffer*, monitorar o sistema de arquivos, analisar *logs* e supervisionar a execução do monitor de configuração da rede, a fim de detectar mudanças ou configurações impróprias da interface de rede [41].

O Hybrid IDS é uma junção das vantagens do NIDS e do HIDS, permitindo uma maior capacidade de detecção, pois tanto capturam dados que trafegam na rede quanto analisam *logs* de dados em *hosts*. Geralmente os sensores são distribuídos em locais estratégicos da rede, a fim de que a capacidade de detecção seja aumentada e que toda a rede esteja protegida contra ataques oriundos de eventos tanto internos quanto externos ao ambiente o qual se encontra a rede.

A partir destas classificações, os tipos de IDS podem ser combinados de acordo com um propósito específico. À guisa de exemplo, temos o NBAD (*Network Behaviour Intrusion Detection Systems*), combinação do NIDS com a técnica de detecção por anomalia. Distribui os sensores em pontos estratégicos da rede, como em *firewalls* e switches de DMZ[8], a fim de analisar o tráfego dos dados com base em determinado padrão de tráfego para aquela rede [41]. Assim, os pacotes não são analisados individualmente, mas sim com base em determinado padrão, permitindo a detecção de ataques como DoS (*Denial of Service*), DDoS (*Distributed Denial of Service*) e SynFlood.

4.3 Síntese

Apresentadas suas definições e classificações, pode-se afirmar que o IDS é uma tecnologia bastante útil e eficiente para a detecção de intrusos e de outras ameaças que podem comprometer sistemas em redes de computadores. O foco deste trabalho é implantar tal tecnologia em redes de computadores virtualizadas, especificamente oferecendo suporte para

[8] *Demilitarized Zone*. Zona Desmilitarizada. É uma rede situada entre uma rede privada e a internet, tendo a função de manter os serviços externos em funcionamento, isolando a rede privada caso os servidores de tais serviços sejam invadidos.

a segurança da infraestrutura de computação em nuvem, em termos de detecção de intrusos e geração de alertas para os administradores.

Sendo assim, no capítulo posterior elucida-se uma abordagem diferenciada da segurança no contexto da computação em nuvem, a fim de possibilitar a demonstração da utilização de um IDS propriamente virtualizado em prol da solução de ameaças de segurança em infraestruturas de nuvem.

5. SEGURANÇA EM COMPUTAÇÃO EM NUVEM

Uma recente publicação relatada em [51], mostra que os clientes da Gartner[9] frequentemente se sentem desapontados com a segurança e a confiabilidade dos serviços de computação em nuvem oferecidos pela companhia, na forma de contratos. Isto se dá pelo fato de que a companhia não deixa claro, nos termos do contrato, sobre a continuidade do negócio e a recuperação dos serviços e dados após a ocorrência de algum imprevisto que possa desestabilizar a infraestrutura de nuvem. Incidentes de segurança, como vazamento de dados, ataques de negação de serviço e mesmo usuários maliciosos podem ser imprevistos relacionados, que retiram a credibilidade da solidez das infraestruturas de nuvem das empresas.

Entre estas publicações e a repercussão que a computação em nuvem atingiu na mídia, muitos crackers visualizam as infraestruturas de computação em nuvem como um alvo em potencial. Tais invasores geralmente se utilizam de técnicas semelhantes para efetuar seus ataques. Logo, quando uma vulnerabilidade é descoberta, a proteção deve ser rapidamente efetivada para que outros ataques não continuem obtendo êxito.

Neste contexto, se faz necessário que se defina o conceito de vulnerabilidade como um fator proeminente de risco [52]. Mais especificamente, de acordo com a taxonomia do Open Groups [53], a vulnerabilidade é a probabilidade de que um patrimônio se encontre apto a resistir às ações de um agente ameaçador. Tal conceito se concretiza quando existe uma diferença entre a força aplicada pelo agente ameaçador e a habilidade do objeto a resistir a esta força. Logo, a vulnerabilidade sempre é definida em termos de risco a certo tipo de ataque.

As vulnerabilidades classificadas como específicas de computação em nuvem, são baseadas nas seguintes características:

- Intrínsecas ou prevalentes ao núcleo da computação em nuvem;
- Têm sua razão principal em uma das características essenciais de nuvem do NIST;
- Causadas quando inovações de nuvem fazem com que controles de segurança se tornem difíceis ou impossíveis de implementar;

[9] Gartner, Inc. é uma companhia de pesquisa e consultoria que lida com tecnologia da informação, com sede em Stanford, EUA [60].

- Predominantes nos oferecimentos estabelecidos no estado da arte da computação em nuvem.

A possibilidade de um atacante obter um escape de máquina virtual, dominação e *hijacking* de sessão, e criptografias inseguras ou obsoletas são exemplos de vulnerabilidades do núcleo tecnológico da computação em nuvem. No que tange a vulnerabilidades nas características essenciais de nuvem, são exemplos que correlacionam uma ou mais características como causas principais, as seguintes:

- Acesso não autorizado à interface de gerenciamento;
- Vulnerabilidades de protocolo de internet;
- Vulnerabilidade na recuperação de dados;
- Evasão de métrica e compra (*billing*).

Logo, tem-se conhecimento de que os principais controles de segurança conhecidos na proteção das infraestruturas de computação em nuvem ainda apresentam defeitos na sua implementação e execução, de tal forma que, *e.g.*, redes virtualizadas oferecem controle insuficiente baseados em rede, os procedimentos de gerenciamento de chave são pobres e métricas de segurança não são adaptadas a infraestruturas de nuvem.

Portanto, existem vulnerabilidades específicas de computação em nuvem que devem ser conhecidas para que se contextualizem as principais características de nuvem dentro do contexto da segurança de redes.

Em virtude disto, a seguir são expostas as principais ameaças de segurança associadas à utilização das infraestruturas de computação em nuvem. Algumas ameaças são oriundas da utilização e gerenciamento de uma determinada categoria de serviços e aplicações, enquanto outras são independentes destes aspectos.

5.1 Principais Ameaças da Computação em Nuvem

A identificação de ameaças em computação em nuvem é um requisito para a elaboração de políticas de segurança em empresas que adotam esse paradigma na disponibilização de serviços.

O relatório "*The Notorious Nine Cloud Computing Top Threats in 2013*" [54], elaborado pela CSA[10], destaca nove ameaças que podem ser encontradas em computação em nuvem no ano de 2013. A seguir são apresentadas tais ameaças.

5.1.1 Violação de Dados

A ameaça mais notória relatada pelo CSA no ano de 2013 é a violação de dados nos servidores de nuvem. Tal violação está diretamente relacionada tanto à perda quanto ao vazamento de dados. A possibilidade de perda dos dados está relacionada à perda física propriamente dita dos dados ou da perda da permissão ou chave criptográfica de proteção de acesso dos dados. A possibilidade de vazamento de dados é caracterizada pela invasão por intrusos nos sistemas de nuvem ou pela utilização inadequada de usuários leigos na operação das interfaces dos servidores.

A partir de Novembro de 2013, pesquisadores da Universidade do Norte da Califórnia, da Universidade de Wisconsin e da Corporação RSA produziram um artigo descrevendo como uma máquina virtual poderia usar um canal alternativo de informações para extrair chaves de criptografia privada utilizadas em outras máquinas virtuais no mesmo servidor físico. Entretanto, em muitos casos, os atacantes geralmente não se utilizam deste mecanismo. Da mesma forma, se um banco de dados de serviço de nuvem que serve a vários locatários não é projetado adequadamente, uma falha em uma aplicação cliente permitirá ao atacante acesso não somente aos dados do cliente, mas a qualquer outro cliente também.

As implicações decorrentes da violação de dados são de que, enquanto a perda e o vazamento de dados são ambas sérias ameaças à computação em nuvem, as medidas que são adotadas para mitigar uma dessas ameaças pode exacerbar a outra. Pode-se habilitar a encriptação de dados para evitar o impacto de um vazamento de dados, mas se tal chave de encriptação é perdida, seus dados também serão perdidos. Alternativamente, deve-se decidir pela manutenção de backups *off-line* dos dados para reduzir o impacto de uma catastrófica perda de dados, entretanto isto aumenta sua exposição à violação de dados.

[10] *Cloud Security Alliance*. Organização que tem como foco de pesquisa o estudo das características de proteção e manutenção da segurança no contexto de computação em nuvem.

5.1.2 Perda de dados

Dados armazenados no ambiente de nuvem podem ser perdidos devido a razões outras além de atacantes maliciosos. Qualquer exclusão acidental pelo provedor de serviços de nuvem, ou pior, uma catástrofe física como incêndio ou terremoto, pode direcionar para uma perda permanente de dados de usuários, a menos que o provedor possa adequar medidas de backup de dados. E até mesmo se um usuário criptografa seus dados antes de armazená-los na nuvem, e perda sua chave de criptografia, os dados também poderão ser perdidos.

5.1.3 *Hijacking* de contas e serviços

Hijacking de contas e serviços não são novos. Métodos de ataque como *pishing*, fraude, e exploração de softwares vulneráveis ainda alcançam resultados. Credenciais e senhas são frequentemente reusadas, o que amplifica o impacto de tais ataques. Soluções de computação em nuvem adicionam novas ameaças para o contexto.

Desta forma, se um atacante tem acesso a credenciais de um usuário, eles podem observar sem autorização suas atividades e transações, manipular dados, retornar informações falsificadas, e redirecionar seus clientes para sites ilegítimos. Sua conta e instâncias de serviços podem se tornar uma nova base para o atacante, se utilizando do poder de sua reputação para executar ataques subsequentes.

A implicação da ameaça de sequestro de contas e serviços está em que a organização deve ressaltar a proibição do compartilhamento de credenciais de contas entre usuários e serviços, e introduzir uma forte técnica de autenticação dupla quando possível.

5.1.4 Interfaces e APIs inseguras

Os provedores de computação em nuvem expõem um conjunto de interfaces de software ou APIs que usuários se utilizam para gerenciar e interagir com os serviços de nuvem. Provisionamento, gerenciamento e monitoramento são todas as formas de utilização destas interfaces. A segurança e a disponibilidade de serviços de nuvem em geral são dependentes da segurança dessas APIs básicas. Da autenticação e controle de acesso para a criptografia e o monitoramento de atividades, estas interfaces devem ser projetadas para proteger contra tentativas acidentais e maliciosas de ação ludibriante.

A negligência sobre um conjunto fraco de interfaces e APIs expõe as organizações a uma variedade de incidentes de segurança relacionada a confidencialidade, integridade, disponibilidade e responsabilidade.

5.1.5 Negação de serviço (*DoS* e *DDoS*)

Ataques de negação de serviço são ataques que previnem os usuários de um serviço de nuvem a acessarem seus dados e suas aplicações. Forçando a vítima do serviço de nuvem a consumir quantidades desordenadas de recursos finitos do sistema, tais como poder de processamento, memória, espaço em disco ou largura de banda de rede, o atacante causa uma lentidão de serviços intolerável. A implicação deste ataque está em que não existem condições de identificar a origem de tal ataque.

5.1.6 Intrusos maliciosos

Uma ameaça de intrusão maliciosa para uma organização pode ser um empregado atual ou aposentado, contratado ou outro parceiro de negócios que tem ou tinha acesso autorizado a uma rede, sistema ou dados da organização. Intencionalmente, o desafeto institucional excede ou abusa de tal acesso, de modo a afetar negativamente a confidencialidade, integridade ou disponibilidade das informações da organização ou sistemas de informação da empresa.

5.1.7 Abuso de serviços de nuvem

Um dos grandes benefícios da computação em nuvem é permitir que mesmo pequenas organizações obtenham acesso à vasta quantidade de poder de processamento. Entretanto, nem todos os usuários pretendem se utilizar destes benefícios para o bem da sociedade. Um atacante precisa levar anos para quebrar uma chave criptografada usando seus recursos de hardware limitados, mas usando um conjunto de servidores de nuvem, ele pode quebrá-la em minutos. Alternativamente, ele pode usar tal conjunto de servidores para elaborar um ataque DoS, disseminar *malware*[11] ou mesmo distribuir um software pirata.

[11] *Malware*. *Malicious Software* ou Software Malicioso são todos os tipos de software que obtém acesso a sistemas sem o consentimento deste, com o objetivo de causar algum tipo de dano, roubo de dados ou acesso a informações confidenciais ou não. Vírus, cavalos de Tróia e *rootkits* são exemplos de *malwares*.

No contexto hodierno dos ataques computacionais originários de usuários internos aos servidores de nuvem, muitos *crackers* estão disponibilizando seus serviços por meio de tais servidores. Eles comercializam ataques e vendem *botnets*[12], que podem ser usados para executar ataques DDoS massivos. Atualmente tal serviço se denomina "*Attacks as a Service*" (AaaS).

O exemplo mais recente de um AaaS tem sido um grupo chinês chamado IM DDODS, permitindo que clientes se cadastrem e ordenem ataques DDoS para quaisquer alvos que eles desejarem. O site do IM DDODS está escrito somente em Mandarim e usa um modelo de *self-service*, onde os usuários podem criar contas, escolher alvos e executar os ataques por si mesmos.

Mesmo se utilizando de grandes infraestruturas de nuvem, os proprietários de AaaS disponibilizam seus serviços por um preço não tão elevado, sendo acessível mesmo a usuários que dispõem de um ignóbil poder aquisitivo. O próprio IM DDODS oferece alguns serviços gratuitos aos seus clientes pagantes.

5.1.8 Diligência prévia

A computação em nuvem trouxe uma espécie de “corrida do ouro”, com muitas organizações que se apressam na promessa de redução de custos, eficiência operacional e aprimoramento da segurança. Enquanto estes podem ser os reais objetivos para organizações que possuem recursos para adotar propriamente as tecnologias de nuvem, muitas empresas se utilizam do contexto da nuvem sem compreender o escopo completo do empreendimento.

Sem o entendimento integral do ambiente de infraestrutura de computação em nuvem, aplicações ou serviços implantados na nuvem, e responsabilidades operacionais tais como resposta a incidentes, criptografia, e monitoramento de segurança, organizações estão assumindo níveis desconhecidos de risco.

O ponto de partida para as empresas e organizações que se movem para um modelo de tecnologia de nuvem está em que eles devem ter recursos suficientes, e executar diligências prévias internas e extensas em provedores de serviço de nuvem para entender os riscos que elas assumem ao adotar este novo modelo de tecnologia.

[12] *Botnet*. É um conjunto de agentes de softwares implantados em diversos computadores interconectados por uma rede, a fim de concretizar um software distribuído a ser utilizado para ataques cibernéticos.

5.1.9 Vulnerabilidades da tecnologia compartilhada

Provedores de serviços de nuvem disponibilizam seus serviços de uma maneira escalável, compartilhando infraestrutura, plataformas e aplicações. Se são os componentes subjacentes que constroem esta infraestrutura, e eles não estão projetados para oferecer propriedades de isolamento fortes para uma arquitetura de muitos locatários (IaaS), plataformas reimplantadas (PaaS), ou aplicações multiusuário (SaaS), a ameaça de vulnerabilidades compartilhada pode existir em todos os modelos de entrega.

A questão é que uma simples vulnerabilidade ou configuração incorreta pode desencadear no compromisso de proteger toda a estrutura em torno de um provedor de nuvem inteiro. Logo, a implicação desta ameaça está na periculosidade causada pela vulnerabilidade, porque potencialmente pode afetar uma nuvem inteira de uma vez.

5.2 Síntese

Em suma, a segurança da computação em nuvem em si, com suas diversas conjunturas e ramificações, é de fundamental importância para que os serviços disponibilizados aos seus demandantes sejam de confiável utilização e eficiência. Não obstante, constatamos que algumas vulnerabilidades são específicas no contexto da computação em nuvem. E como formas específicas de ameaças, suas respectivas contramedidas devem ser especializadas para as infraestruturas peculiares de nuvem.

Sem embargo, diversas contramedidas podem ser utilizadas em nuvens, sendo que cada uma aborda um aspecto da segurança distinto e ao mesmo tempo complementar aos outros aspectos. Desta forma, apresentamos no capítulo subsequente a descrição da abordagem de contramedida de um sistema de detecção de intrusão aplicado para computação em nuvem.

6. ESTUDO DE CASO

Neste capítulo, evidenciamos a concretização e o funcionamento do componente de contramedidas do trabalho de Araújo, J. D. [2], configurando o IDS adaptado à configuração de rede presente no ambiente virtual. Tal sistema de contramedidas foi implementado utilizando-se a linguagem de programação Java [55], servindo de suporte para que o administrador possa resolver os problemas da rede em tempo hábil e de maneira a minimizar os danos ao sistema porventura provenientes de um ataque.

Os componentes da arquitetura-base serão demonstrados, juntamente com o ambiente de testes corretamente estruturado, coletando resultados e dados a respeito da eficiência do componente de reação, ao se deparar com ataques provenientes de usuários internos.

6.1 Arquitetura do IDS

A Figura 9, adaptada de [2], demonstra os componentes do IDS e suas respectivas interações entre si. Tais componentes se distribuem pela infraestrutura da nuvem, de tal forma que para cada *Node Controller* da IaaS, existe um conjunto composto pelo IDS_Node, juntamente com Netsensor, IDS_Analyser, IDS_Pool e IDS_Reaction. Sendo que cada IDS_Node centraliza as informações daquele respectivo *Node Controller* e os envia para o IDS_Admin, localizado no *Cloud Controller* da IaaS.

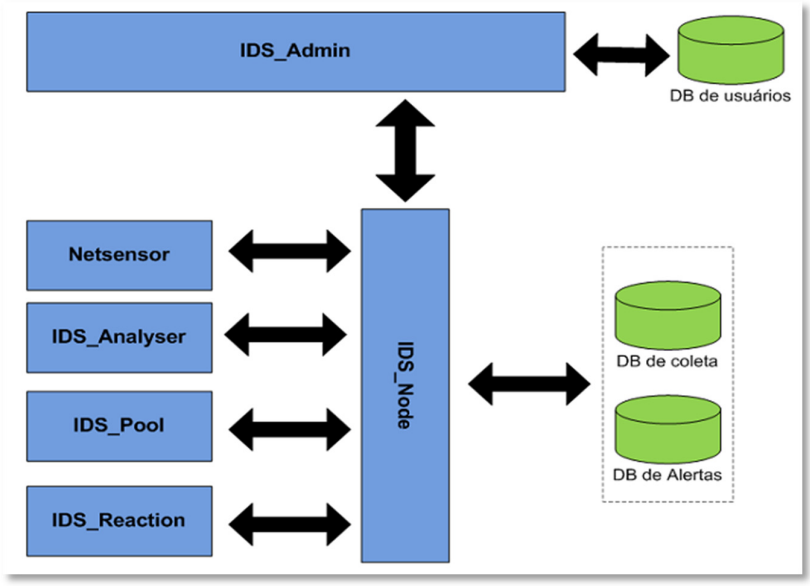

Figura 9. Arquitetura do IDS [2].

O IDS_Admin funciona como o administrador de todas as informações de detecção de intrusão oriundas de cada Node Controller da infraestrutura, recebendo tais informações a partir de cada IDS_Node presente em seu respectivo Node. A Figura 10, adaptada de [2], mostra a execução do IDS_Admin como um administrador de todos os módulos de IDS, que se encontram em cada nó, tendo a função de informar aos administradores da nuvem acerca dos eventos ocorridos e dos status do sistema no que tange à manutenção da segurança.

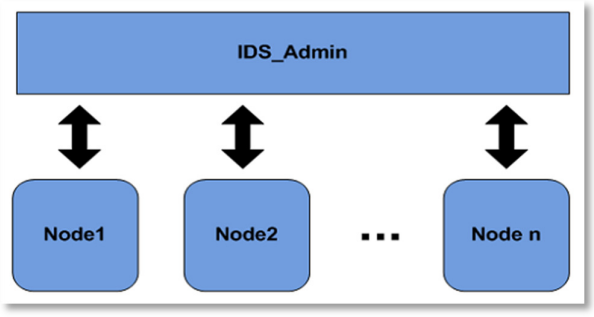

Figura 10. Arquitetura do IDS com vários nodes [2].

Na Figura 11, adaptada de [2], temos o digrama de classes do IDS, evidenciando o nome das classes, seus respectivos métodos e atributos, caracterizando a distribuição dos componentes do IDS no ambiente de computação em nuvem no qual for implantado. A

classes IDS_Admin funcionando como um monitor de alertas, provenientes de cada IDS_Node implantado em cada *Node Controller*, para que o administrador possa tomar as providências necessárias para a manutenção da disponibilidade dos serviços da nuvem. A classe IDS_Node funciona como um centralizador das ações do IDS dentro de um *Node Controller*, inicializando o Start_IDS, recebendo os alertas provenientes do IDS_Analyser, ativando o IDS_Reaction caso ocorra alguma intrusão e encaminhando os respectivos alertas ao IDS_Admin.

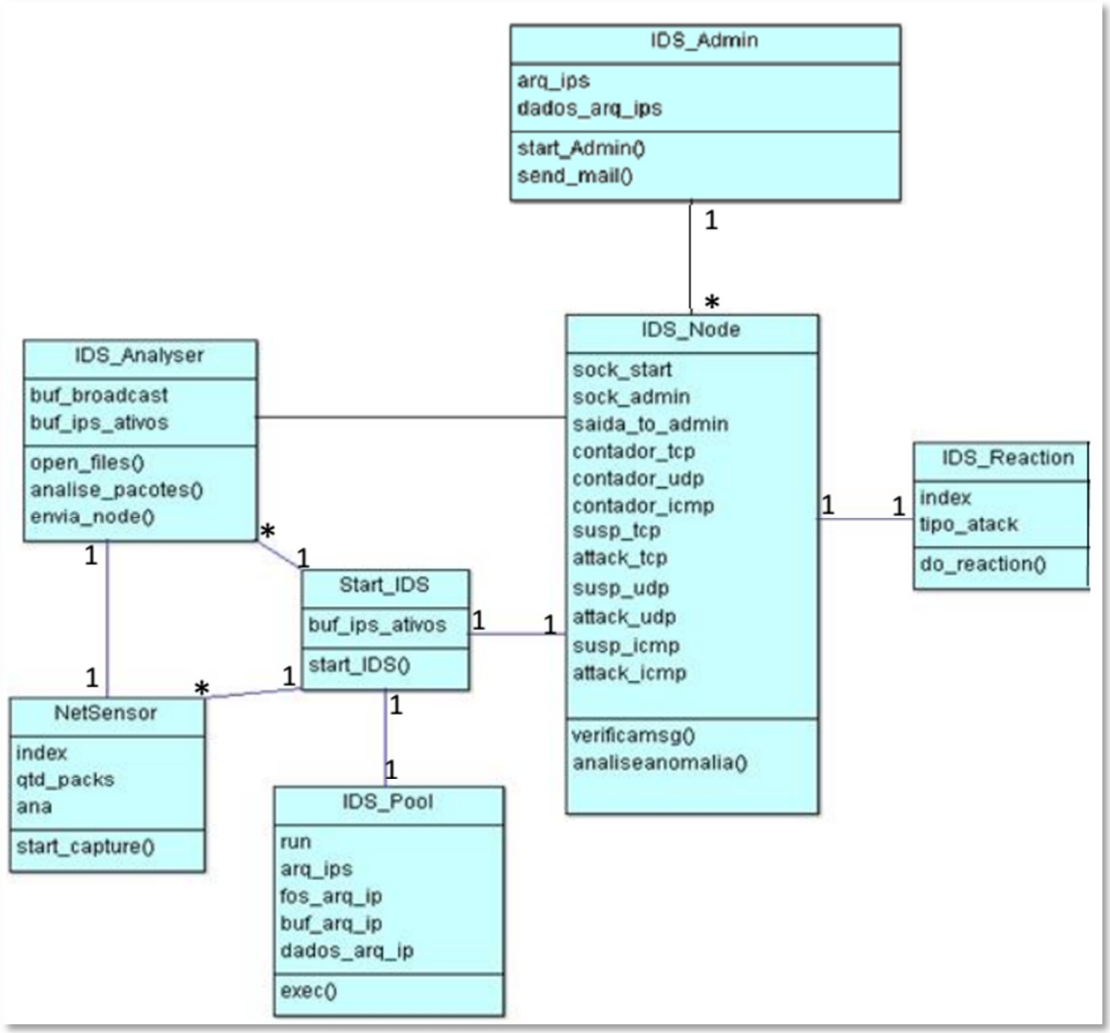

Figura 11. Diagrama de classes do IDS [2].

Em seguida, temos a representação do diagrama de atividades, pela Figura 12, extraída de [2], demonstrando a execução da detecção de intrusão pelo IDS. Inicialmente, o Netsensor realiza a captura dos pacotes da interface de rede do ambiente das máquinas

virtuais, disponibilizando tais pacotes ao IDS_Analyser, que realizará a análise do cabeçalho IP de cada pacote, comparando-o com padrões de ataque pré-determinados, evidenciando a detecção por assinatura.

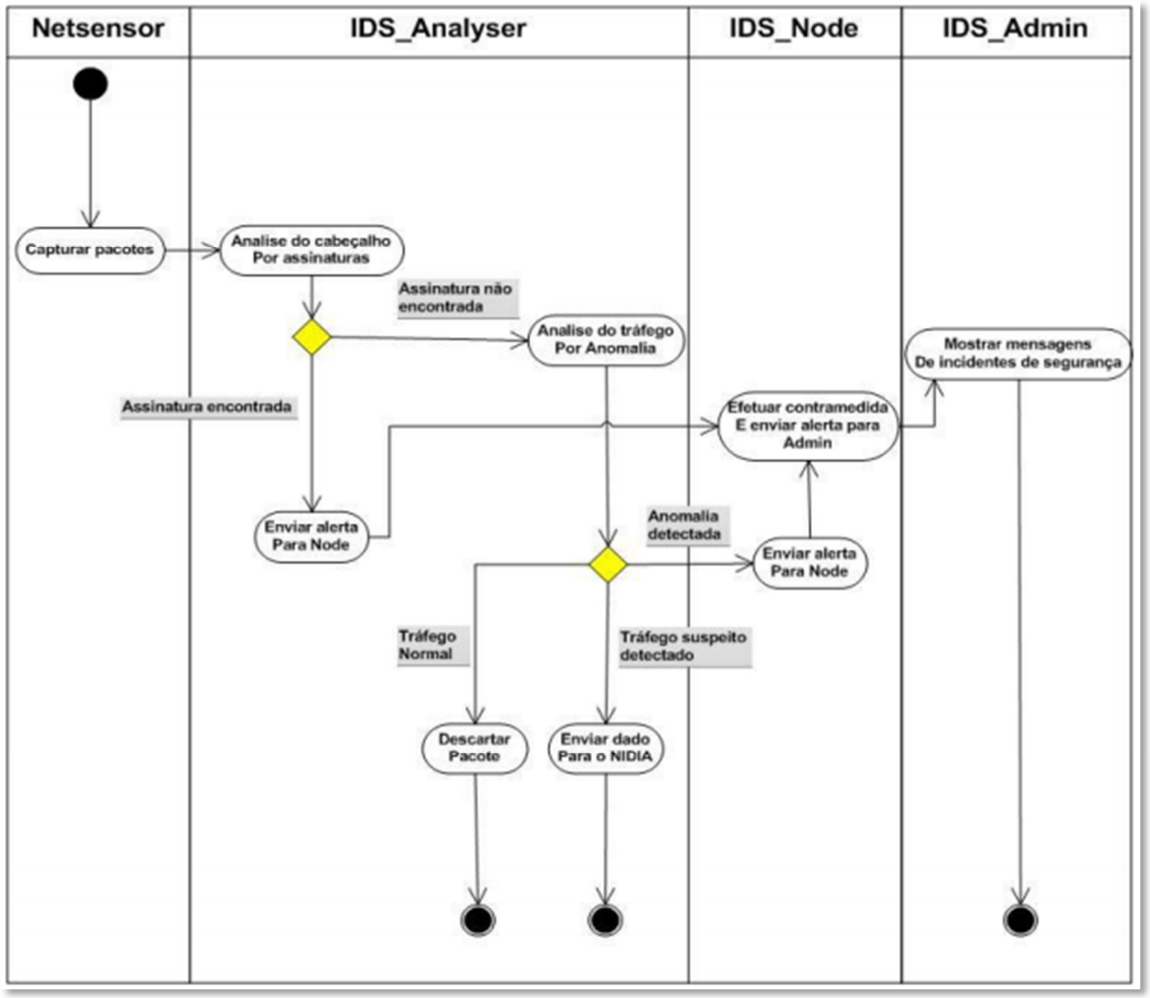

Figura 12. Diagrama de Atividades do IDS [2].

Se um padrão de assinatura é encontrado em determinado cabeçalho de pacote, é enviado um alerta ao IDS_Node, indicando os dados do ataque, entre estes o endereço IP de origem e de destino, para que o IDS_Node inicialize o Reaction, para que seja efetuada a contramedida necessária, e enviando um alerta ao Admin sobre os eventos ocorridos.

Caso não seja encontrado um padrão de ataque no cabeçalho do pacote, será realizada a análise do tráfego por anomalia. Tal análise se encontra fora do escopo deste trabalho, pois o componente de reação não lidará com os procedimentos de execução de contramedidas caso sejam encontradas anomalias no tráfego da rede das VMs da

infraestrutura de computação em nuvem, mas somente com ataques detectados a partir dos métodos de detecção por assinatura.

Apresentadas as características de modelagem e funcionamento do IDS utilizado para o ambiente de testes, demonstremos tal ambiente, detalhando a arquitetura utilizada e focando na descrição do módulo Reaction, objeto do estudo de caso, que executará as contramedidas necessárias conforme requisições do IDS_Node.

6.2 Ambiente de testes

O ambiente de testes utilizado para implantar o IDS encontrou guarida no LABSAC[13], tendo como base a utilização de duas máquinas conectadas em rede, como mostra a Figura 13. Tal ambiente tem o objetivo de simular um ambiente de infraestrutura de computação em nuvem (IaaS).

Tais máquinas que mantém o ambiente em funcionamento são os hosts **LABCLOUD** e **AGATHA**. O host **LABCLOUD** funciona como o sistema de virtualização principal, tendo a função de *Node Controller* no contexto do ambiente simulado de computação em nuvem. Nele foi instalado o Linux Ubuntu Server 12.04.3 [56], juntamente com o monitor de máquinas virtuais KVM [57] e a biblioteca de gerenciamento de virtualização libvirt [58], a fim de permitir o funcionamento de um ambiente de virtualização semelhante à de uma IaaS. Enquanto o host **AGATHA** se mostrou útil para a virtualização de um VM administrativa, denominada **bt**, executando sobre o gerenciador de virtualização VirtualBox, versão 4.2 [37].

[13] Laboratório de Sistemas em Arquiteturas Computacionais do Departamento de Engenharia de Eletricidade da Universidade Federal do Maranhão.

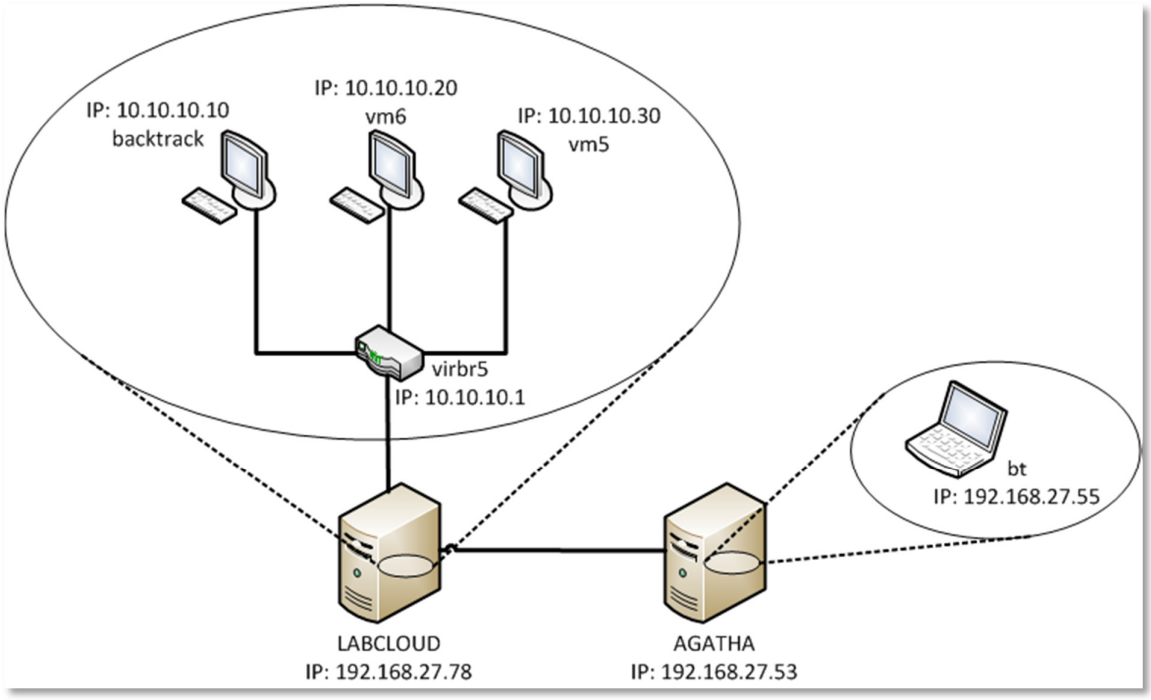

Figura 13. Ambiente de testes do IDS.

A VM **bt** funciona como o módulo Admin, contida no host **AGATHA** que representa o *Cloud Controller*, que fornece uma interface para o administrador da IaaS, exibindo todos os eventos oriundos de cada nó da nuvem. O IDS se adequa a qualquer quantidade de nós disponíveis na nuvem, ou seja, trata-se de um IDS distribuído, sendo que no caso abordado, temos somente um nó, representado pela rede de endereçamento IPv4 10.10.10.0 e gateway 10.10.10.1, concretizado pela interface de rede virtual **virbr5**. Tal VM **bt** conecta-se diretamente com o IDS_Node contido na máquina **LABCLOUD**, recebendo os alertas necessários. No caso de existirem mais nós no ambiente, todos os IDS_Node de cada *Node Controller* serão conectados à VM **bt**, para que se obtenham informações de todos os nós.

No host **LABCLOUD**, representando o *Node Controller*, foi criada uma rede virtual gerenciada pelo KVM, segmentada pelo roteador **virbr5**, suportando as VMs **backtrack**, **vm6** e **vm5**, com os respectivos IPs contidos na Tabela 1.

Tabela 1. Endereçamento IP das máquinas virtuais.

Máquina Virtual	Endereço IP
bt	192.168.27.55
backtrack	10.10.10.10
vm6	10.10.10.20
vm5	10.10.10.30

Para nossos testes, a VM **backtrack** será a máquina virtual que sofreu um ataque *masquerading* e que estará executando sob a administração de um usuário malicioso que se passa por usuário legítimo, que realizará ataques, a fim de testar a vulnerabilidade do ambiente. A partir dela serão executados comandos de testes de vulnerabilidade, como varredura de portas, utilizando a metodologia de envio de pacotes ICMP com requisição de resposta para todos os hosts da rede ou enviando para o endereço broadcast da rede, e executando ataques do contexto de negação de serviços, como o Synflood e o Smurf.

Como "vítimas" dos ataques, foram instanciadas as máquinas virtuais **vm6** e **vm5**, que estão na mesma rede da VM maliciosa, porém não representam perigo à rede, pelo fato de serem administradas por usuários legítimos. Contudo, tais VMs estão propositadamente em execução e apresentando vulnerabilidade a ataques, a fim de que se demonstre a dinâmica de reação e contramedidas do módulo Reaction.

6.3 Implementação do componente IDS_Reaction

Tendo como base a ideia de elaborar um módulo de proteção das máquinas virtuais dentro do contexto de infraestrutura de computação em nuvem, concretizamos o Reaction. O módulo Reaction, componente de contramedidas do IDS, que integra funções específicas para cada *Node Controller* por meio do seu respectivo IDS_Node, tem como função intrínseca a proteção do sistema no caso de ataques internos ao sistema, efetuados por usuários maliciosos que se passam por legítimos, suspendendo a execução da VM maliciosa até que o administrador tome a decisão de continuar a execução de tal VM, migrá-la ou desligá-la, conforme se faça necessário.

Logo, mesmo que o IDS seja capaz de realizar a proteção automática do sistema, os sistemas responsivos, que realizam contramedidas em sistemas, não substituem os

administradores de sistemas, mas somente realizam atividades preventivas, a fim de que protejam o sistema até que o administrador possa tomar suas decisões.

Na implementação do componente Reaction, toma-se como base uma das mais notórias ameaças para a computação em nuvem [54], como descrita no item 5.1.5 deste trabalho: a negação de serviço, abordando tanto a negação de serviço ordinária (DoS[14]) quanto a negação de serviço distribuída (DDoS[15]), que se utiliza de vários computadores "zumbis" para realizar o ataque com maior efetividade.

Contudo, ataques DoS e DDoS são bastante difíceis de se combater num ambiente virtualizado, pois tais tipos de ataques são baseados no IP Spoofing, técnica que manipula o cabeçalho dos pacotes IP, a fim de mascarar ou forjar a origem dos pacotes, dificultando a detecção e localização do usuário malicioso.

No estudo de caso em questão, serão realizados comandos básicos de rede pelo usuário malicioso, como "ping" para outra VM, "ping" para o endereço de broadcast da rede e o ataque Smurf, que se trata de um ataque de negação de serviço distribuído, que forja o endereço de origem do cabeçalho IP e é enviado para o endereço de broadcast, a fim de que todas as máquinas presentes na rede respondam para o endereço da vítima, que é o endereço de origem. Nos dois primeiros casos, a VM maliciosa será suspensa, já que se tem conhecimento exato do IP de origem, enquanto que no ataque Smurf a VM da vítima será suspensa, no intuito de que o administrador da nuvem migre tal VM para outro *Node Controller* mais seguro, ou alerte o usuário para que ele reinicie o sistema.

No caso da suspensão de VM maliciosa, o nome do usuário malicioso que a está administrando será colocado na lista negra (*blacklist*), que ficará disponível para o servidor de autenticação, para que este possa confirmar os dados do usuário que foi considerado malicioso em seu último acesso e assim ratificar seus dados pessoais na nuvem, retirando seu nome da *blacklist*, caso haja confirmação da sua identidade legítima.

Desta forma, as contramedidas serão realizadas usando comandos da biblioteca libvirt [58] e do próprio conjunto de comandos nativos do Linux. Ao ser chamado pelo IDS_Node, o componente Reaction obtém o nome do usuário da VM maliciosa e resolve o nome da VM a partir do endereço IP, a fim de que suspenda tal VM pelo comando: "**# virsh suspend <vm name>**". Onde "virsh suspend" é o comando da biblioteca libvirt que suspende uma máquina virtual do ambiente de máquinas virtuais e "vm name" é a denominação do host da VM referenciada.

---

[14] *DoS*: *Denial of Service*.
[15] *DDoS*: *Distributed Denial of Service.*

O Código 1 demonstra um trecho de código em Java do componente Reaction para suspender uma VM. Sendo que no apêndice A encontra-se todo o código-fonte devidamente implementado e comentado.

Código 1. Reaction suspendendo VM.

```
73  // suspend VM
74  Process process_suspend = run.exec("virsh suspend "
75          + name_malicious_vm);
76  if (process_suspend == null) {
77      System.out.println("Erro na suspensão da VM maliciosa.");
78  } else {
79      System.out.println("Suspending malicious (victim) VM...");
80  }
```

Após a suspensão da VM, o nome do usuário da VM maliciosa será colocado na *blacklist*, alertando o administrador de que tal usuário está praticando atos que vão de encontro às políticas de segurança de utilização da infraestrutura de nuvem, se forem acordados no SLA. A suspensão da VM protege o sistema de avarias supervenientes, enquanto o administrador toma as providências necessárias para o reforço da segurança do ambiente de nuvem, tais como informar ao usuário da VM vítima a respeito do ataque, migrá-la para um ambiente de quarentena ou, no caso da VM maliciosa, desligá-la.

Assim que haja evidências da ocorrência de ataques no ambiente, dados podem ser coletados para avaliação e comparação de padrões, propiciando um ambiente para a atuação da computação forense, no intuito de direcionar investigações para a descoberta de autoria e materialidade dos crimes cibernéticos praticados em infraestruturas de computação em nuvem.

A seguir, temos o detalhamento dos testes realizados, descrevendo os ataques concretizados, a forma com que o sistema de contramedidas reagiu e os resultados obtidos.

6.4 Testes e resultados

Primeiramente inicializamos a VM administrativa (**bt**), executando no host AGATHA (*Cloud* Controller) e, em seguida, os componentes do IDS no host LABCLOUD (*Node Controller*), que são o IDS_Node, que inicializa o Start_IDS, e as VMs **backtrack, vm5** e **vm6**.

Utilizamos a VM **backtrack** para realizar os ataques de rede, dentro do ambiente virtual, tendo **vm5** e **vm6** como máquinas virtuais vítimas. O ambiente foi organizado da forma ilustrada pela Figura 14, que possibilita uma visão global da interface de administração do LABCLOUD, contendo o terminal de comandos de administração do LABCLOUD ("TERMINAL DO LABCLOUD") para visualizar o status do ambiente das máquinas virtuais, a interface de usuário da **vm5** ("VM5"), a interface do componente de inicialização do IDS no respectivo *Node Controller* Start_IDS ("Start_IDS"), a interface de usuário da **backtrack** ("BACKTRACK"), a interface de usuário da **vm6** ("VM6") e a interface de execução do componente IDS_Node ("IDS_Node"), que realiza a administração local dos componentes do IDS.

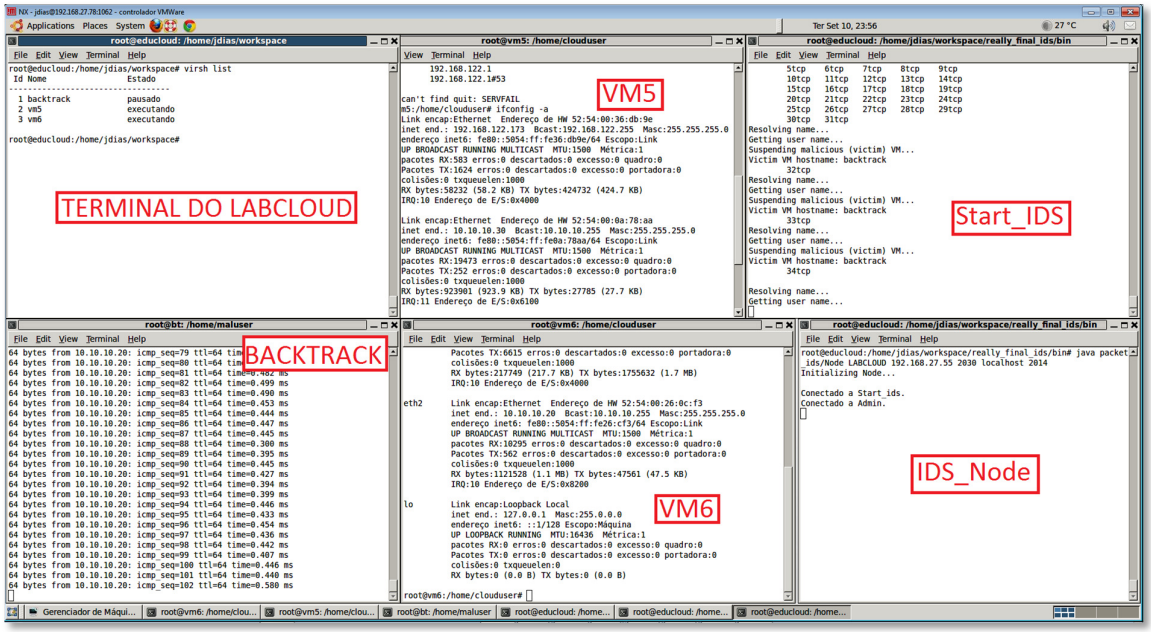

Figura 14. Visão ampla da interface do LABCLOUD.

6.4.1 Teste 1: "probattack"

A Figura 14 também representa o primeiro teste realizado, indicando uma varredura de host ou mesmo um ataque Ping of Death[16], que denominamos de "probattack", tendo a seguinte sequência de eventos:

[16] Ping of Death. Tipo de ataque de negação de serviço realizado por um computador ao enviar pacotes ICMP com tamanho superior ao permitido, gerando assim, sobrecarga de *buffer* ou *kernel dump*.

1. Emissão do comando "**# ping 10.10.10.20**" de **backtrack** para **vm6** (Figura 15);

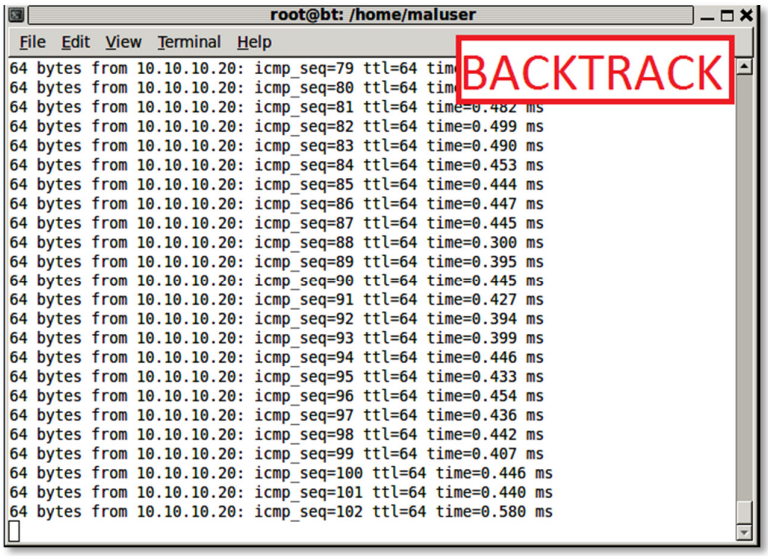

Figura 15. Interface da backtrack ao emitir o comando ping para a vm6 (10.10.10.20).

2. Captura e análise dos pacotes ICMP, requisição de contramedidas, paralização da **backtrack** e respectiva inclusão do nome do usuário atual (maluser) na *blacklist* (Figura 16 e Figura 17);

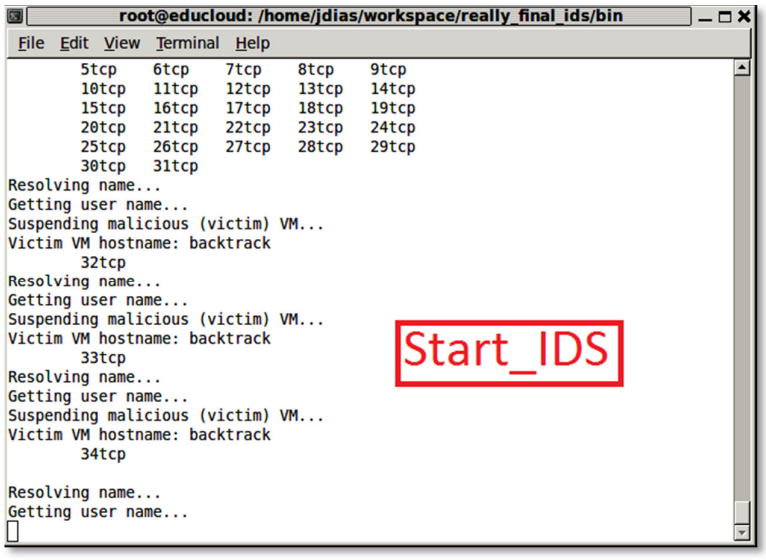

Figura 16. Interface de Start_IDS, exibindo o status do componente Reaction ao executar as contramedidas.

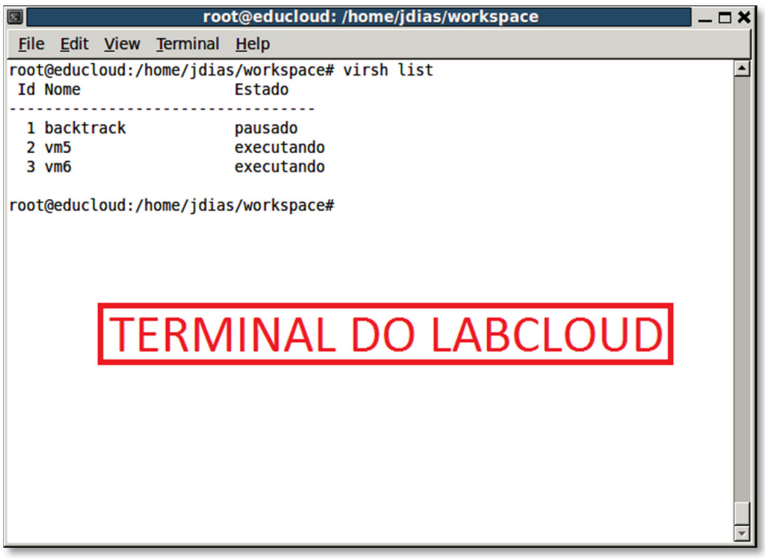

Figura 17. Interface do terminal LABCLOUD, evidenciando o status "pausado" na backtrack, após a execução da contramedida.

3. Emissão de um alerta para a **bt**, que representa o administrador do sistema. A Figura 18 representa esta situação, evidenciando o *hostname* da VM maliciosa suspensa, o nome do usuário considerado malicioso (maluser) e sua respectiva inserção na *blacklist*.

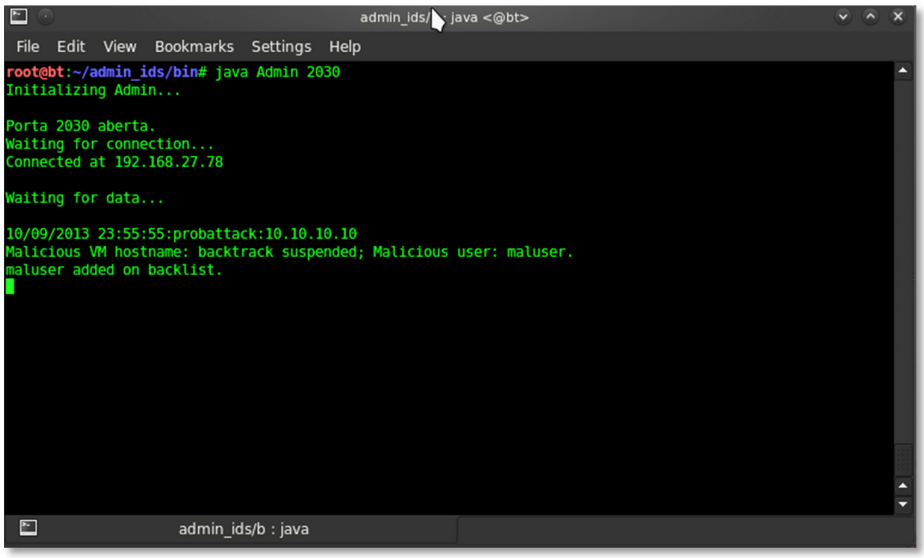

Figura 18. Interface da VM administrativa bt no momento do alerta da suspeita de ataque de varredura.

Neste ponto, a **backtrack** está suspensa e o alerta foi emitido ao administrador, podendo qualquer medida ser tomada em benefício da maximização da segurança do sistema.

No segundo e terceiros testes, temos a denominação de "smurf" tanto para ping para broadcast ordinário, evidenciando uma varredura dos hosts presentes na rede, quanto para ping para broadcast em conjunção com IP Spoofing, que é o ataque Smurf propriamente dito. Como o IP de origem pertence à rede virtual, mas não existem garantias de que tal IP de origem é o mesmo da máquina que o originou, todo ping para broadcast é classificado como suspeito e rotulado como "smurf", para fins de orientação ao administrador.

6.4.2 Teste 2: "smurf" em ping broadcast

O segundo teste tem o propósito de verificar a proveniência de um host que emite um ping para broadcast, demonstrando que tal comando é considerado suspeito mesmo que não seja propriamente um ataque Smurf. Temos a seguinte sequência de eventos para este teste:

1. Emissão do comando "**# ping 10.10.10.255 -b**" de **backtrack** para **broadcast** (Figura 19), sendo que o endereço "10.10.10.255" é o endereço de broadcast da rede, e a opção "-b" do comando representa a emissão de pacotes ICMP para broadcast;

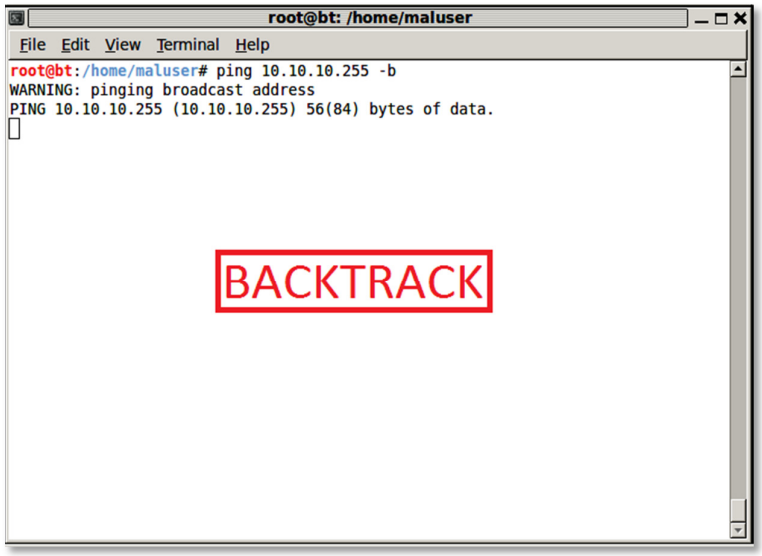

Figura 19. Interface da backtrack, emitindo ping para broadcast.

2. Captura e análise dos pacotes ICMP, requisição de contramedidas, paralização da **backtrack** e respectiva inclusão do nome do usuário atual (maluser) na *blacklist* (Figura 20).

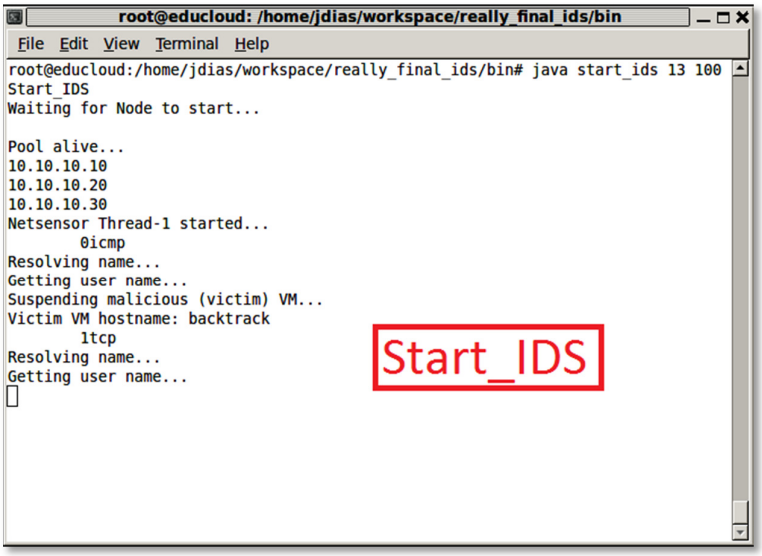

Figura 20. Interface de Start_IDS, exibindo os alertas de status de Reaction ao executar as contramedidas no caso de suspeita de ataque pelo endereço broadcast.

3. Emissão de um alerta para a **bt**, que representa o administrador do sistema. A Figura 21 representa esta situação, evidenciando o *hostname* da VM maliciosa suspensa, que no caso é sempre considerado vítima, por não ser possível, neste sistema, confirmar se o IP da VM suspeita é o mesmo do IP de origem do cabeçalho do pacote ICMP, e o nome do usuário malicioso.

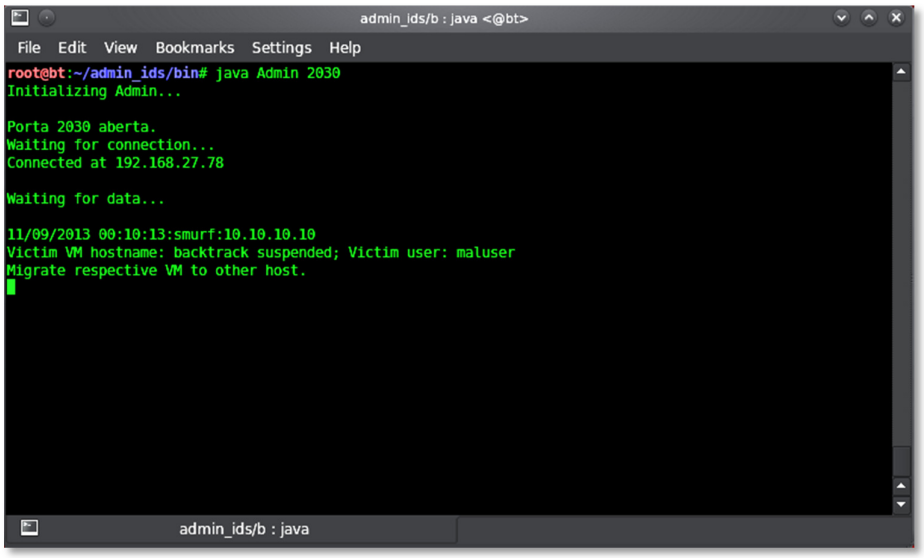

Figura 21. Interface da VM administrativa bt no momento do recebimento do alerta da suspeita de ataque smurf (ping para broadcast trivial).

6.4.3 Teste 3: "smurf" propriamente dito

O terceiro teste se refere à característica do componente de reação do IDS em detectar a vítima do ataque Smurf e possibilitar ao administrador da rede a opção de migrar tal VM para uma zona de quarentena para possíveis análises forenses, contatar-se com a vítima ou mesmo reiniciar tal VM.

Contudo, assim como evidenciado no Teste 2, o IDS não tem condições de determinar se a VM relacionada à ameaça está apenas realizando uma varredura dos endereços IP dentro da rede virtual, realizando um ping broadcast, ou se ela está sendo vítima de um ataque Smurf. Por conta disso, é necessário que a VM seja deslocada para uma área considerada segura e ponha a responsabilidade sobre tal VM diretamente nas mãos do administrador da rede.

Sem embargo, temos a seguinte sequência de eventos para este teste:

1. Emissão do comando "**# hping3 –a 10.10.10.20 -1 10.10.10.255 –i 1**" de **backtrack** para broadcast, para ordenar a resposta para a **vm6** (Figura 22). Este comando se utiliza da ferramenta hping3 [59], que permite a edição do cabeçalho e o controle de envio de pacotes ICMP. A **backtrack** envia vários pacotes ICMP *echo request* para o endereço broadcast (10.10.10.255), falsificando o endereço de

origem para a **vm6** (10.10.10.20), que será a vítima do ataque, recebendo uma enxurrada (*flooding*) de pacotes ICMP *echo reply* como forma de resposta.

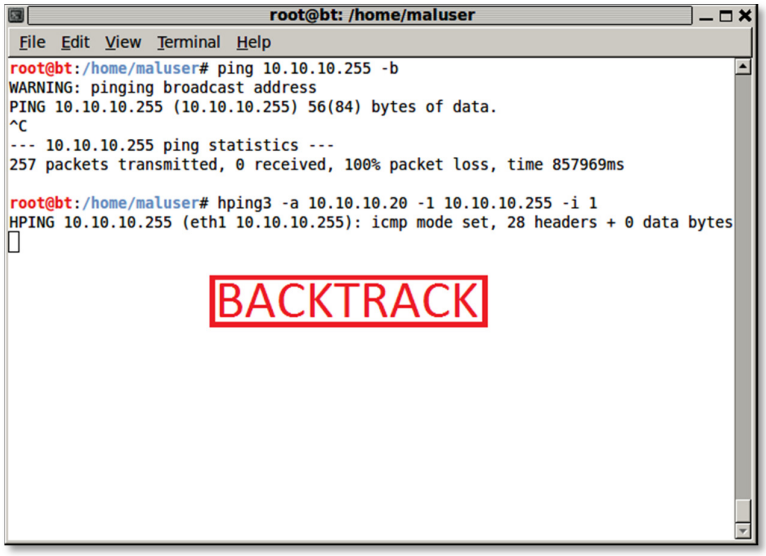

Figura 22. Interface da backtrack, emitindo uma enxurrada de pacotes ICMP para broadcast, com IP de origem falsificado.

2. Captura e análise dos pacotes ICMP, requisição de contramedidas, paralização da **vm6** (Figura 23) e respectiva inclusão do nome do usuário vítima (clouduser) na *blacklist* (Figura 24).

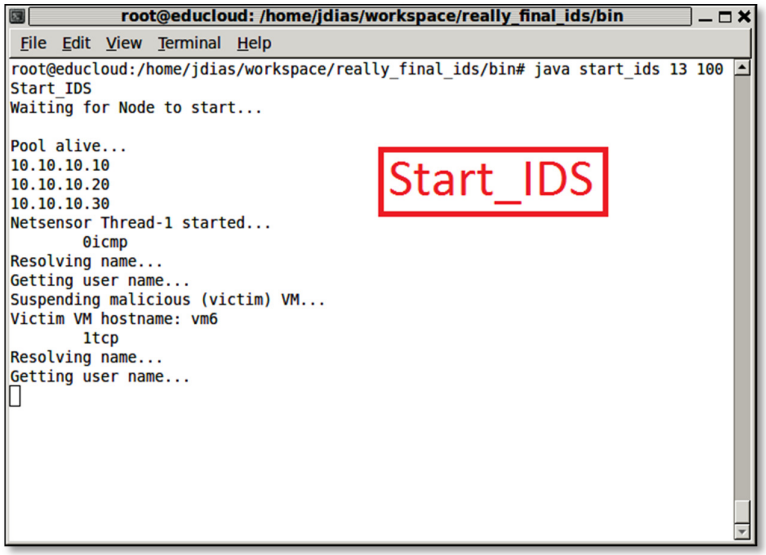

Figura 23. Interface de Start_IDS, identificando o nome da VM vitima (vm6) e exibindo o status do componente Reaction ao executar as contramedidas.

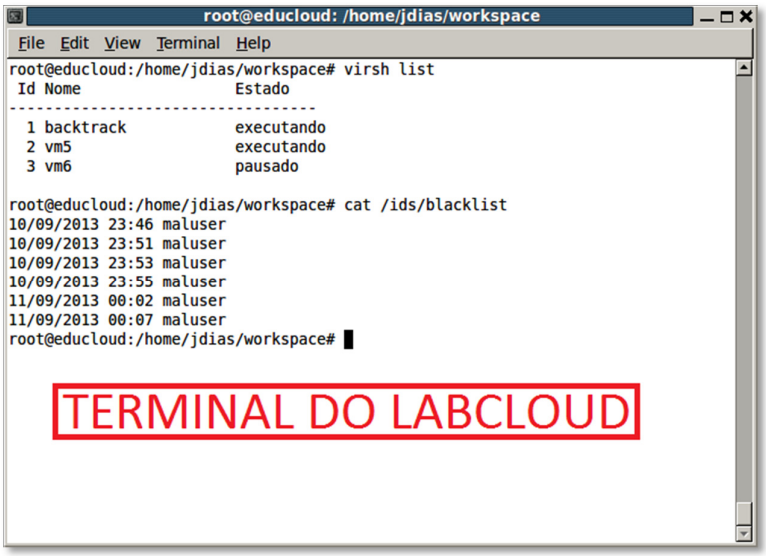

Figura 24. Interface do terminal LABCLOUD, evidenciando o status "pausado" da vm6, após a execução da contramedida.

3. Emissão de um alerta para a VM **bt** (Figura 25), evidenciando o hostname da VM vítima suspensa (**vm6**) e identificando o nome do usuário desta VM (clouduser).

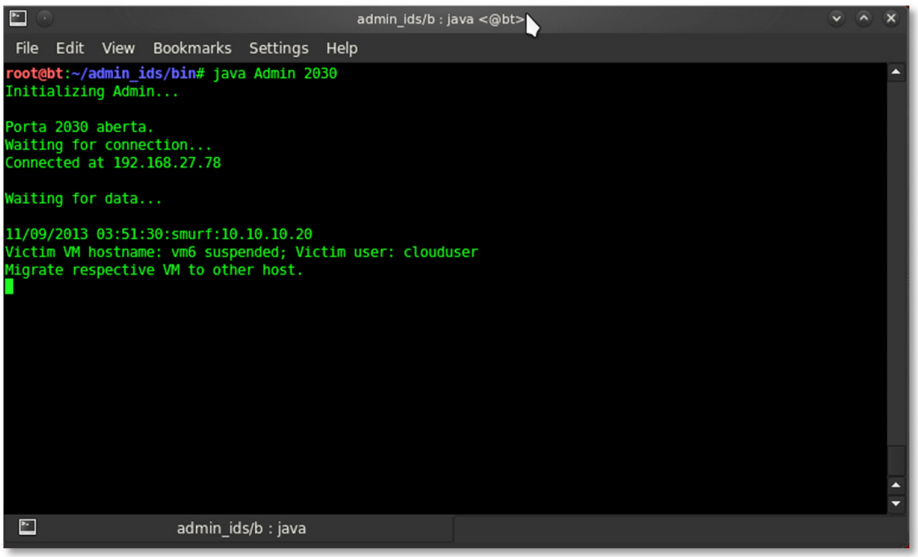

Figura 25. Interface da VM administrativa bt no momento do recebimento do alerta de ataque smurf.

6.5 Avaliação dos resultados

Realizados os testes correspondentes à implementação e a implantação do componente responsivo no ambiente de detecção de intrusão, ajustado a um ambiente virtual de uma infraestrutura de computação em nuvem, podemos constatar a funcionalidade de tal componente, reagindo na forma de contramedidas a todos os ataques a VMs testados.

No Teste 1 foi simulado o ataque de Ping of Death, de tal forma que o IDS detectou o ataque e executou o componente de reação, paralisando a VM atacante (**backtrack**) e enviando um alerta ao administrador (**bt**). Neste contexto simulado, a estratégia se mostrou eficiente, e mesmo em contexto real, no qual a VM realizará uma varredura de portas em contrassenso às políticas de segurança da infraestrutura, a reação se mostrará eficiente, pois, tão logo o IDS detecte a ocorrência do envio do primeiro pacote ICMP para a realização do ataque, o sistema suspenderá a VM, evitando a progressão do ataque.

No Teste 2, a tentativa de envio de pacotes ICMP para todas as VMs por meio do endereço de broadcast foi logo suspensa pelo componente de contramedidas, estando a VM atacante (**backtrack**) disponível à intervenção do administrador da rede.

Por fim, no Teste 3, o ataque Smurf foi constatado e o sistema de detecção reagiu suspendendo a VM vítima (**vm6**), deixando-a disponível à decisão do administrador. Todavia, neste tipo de ataque, o atacante é dificilmente encontrado, pois o endereço IP de origem é substituído pelo endereço IP da VM vítima.

6.6 Síntese

Diante ao exposto, neste capítulo foi apresentada uma solução para a concretização de um módulo de contramedidas para um sistema de detecção elaborado especificamente para possibilitar sua implantação em infraestruturas de computação em nuvem.

Esclarecemos a arquitetura do referido IDS, seus diagramas de sequência, de classe e a forma com que cada componente atua na detecção, além da interação existente entre o IDS e o administrador da rede. Em seguida, apresentamos a concretização do componente de reação, suas respectivas funcionalidades e sua integração com o IDS exposto.

Ao final, os testes e resultados demonstraram a eficiência do sistema de detecção em determinados casos de ataques, evidenciando a elevada capacidade do componente Reaction na realização de contramedidas.

7. CONCLUSÃO

Este trabalho demonstrou que o contexto da segurança na computação em nuvem tem recebido muito investimento e atenção por parte das empresas e organizações interessadas na utilização desta tecnologia. Sobretudo no que concerne à confiabilidade na virtualização das infraestruturas, na correta utilização dos recursos por políticas de segurança eficazes e na garantia de um armazenamento de dados seguro, muitas técnicas de segurança sobrepujam o esperado para a manutenção de um sistema de nuvem.

Não obstante, a utilização indevida das máquinas virtuais disponibilizadas pelos ambientes de nuvem é um fator a ser considerado na adoção de estratégias para a manutenção da disponibilidade dos serviços de nuvem. Refletindo neste aspecto, a adoção de técnicas de detecção de intrusão já exaustivamente adotadas pelas redes de computadores foi adaptada, neste trabalho, para servir de uma ferramenta de segurança adequada para a detecção de atividades de maliciosas e tráfegos suspeitos em redes de máquinas virtuais que executam em infraestruturas de computação em nuvem.

E como contribuição fundamental, este trabalho apresenta a concretização de um módulo de contramedidas adaptado a um IDS que opera em uma infraestrutura de computação em nuvem. Tal módulo proporcionou a suspensão de máquinas virtuais suspeitas, possibilitando um tempo hábil para que o administrador da infraestrutura tome as medidas necessárias para a manutenção do ambiente virtual, contatando o usuário, migrando ou reiniciando a VM.

7.1 Retrospectiva do trabalho

A computação em nuvem foi amplamente discutida ao longo de todo texto deste trabalho, mas no primeiro capítulo tivemos o primeiro contato com a tecnologia, abordando conceitos gerais e a forma com que ela se apresenta no contexto atual. Foram apresentadas suas características principais, as categorias de serviços e aplicações e os modelos de implantação, de acordo com o NIST. Além de ter sido apresentada, com a devida ênfase, a questão do *lock-in* das plataformas de nuvem e as principais tentativas de padronização.

Como conceito chave de concretização da computação em nuvem, no segundo capítulo, a virtualização se mostra evidente para oferecer escalabilidade e isolamento necessários para a efetivação da segurança das infraestruturas de nuvem. Foram apresentadas

as principais estratégias de virtualização, como formas de adaptação da virtualização para vários contextos e objetivos.

Sistemas de Detecção de Intrusão é o cerne para a concretização das estratégias de detecção de intrusão em todos os contextos em que envolvam redes de computadores, tanto físicos quando virtuais. No quarto capítulo foram apresentadas as definições e algumas classificações de acordo com o NIST para o conceito de IDS, tendo diversas outras classificações formuladas por acadêmicos, sendo combinações tanto do local em que são implantados quanto da forma como se realiza a detecção.

A segurança da computação em nuvem com suas diversas conjunturas e ramificações é de fundamental importância para que os serviços disponibilizados aos seus demandantes sejam de confiável utilização e eficiência. Destarte, no quinto capítulo é apresentada a problemática da segurança no contexto da computação em nuvem, sendo apresentados os conceitos específicos de vulnerabilidade para computação em nuvem e as ameaças mais notórias de computação em nuvem no ano de 2013.

No sexto e último capítulo é apresentado o estudo de caso, que demonstra uma solução para a concretização de um módulo de contramedidas para um sistema de detecção elaborado especificamente para possibilitar sua implantação em infraestruturas de computação em nuvem. Foi esclarecida a arquitetura do referido IDS, seus diagramas de sequência, de classe e a forma com que cada componente atua na detecção, além da interação existente entre o IDS e o administrador da rede. Em seguida, foi apresentada a concretização do componente de reação, suas respectivas funcionalidades e sua integração com o IDS exposto. Ao final, os testes e resultados demonstraram a eficiência do sistema de detecção em determinados casos de ataques, evidenciando a elevada capacidade do componente Reaction na realização de contramedidas.

Neste trabalho surgiram algumas dificuldades que se mostraram intrínsecas à concretização do ambiente de testes. No que tange à adoção do QEmu [35] como o hipervisor para a realização dos testes, tal hipervisor se mostrou de grande valia para a realização deste trabalho, pois grande parte do que foi implementado se baseou em sua plataforma de virtualização oferecida e pelo estudo dos comandos realizados na biblioteca libvirt [58].

Os testes não foram realizados em plataforma isoladamente de nuvem, se utilizando de IaaS populares como o Eucalyptus [19], entretanto a utilização da abordagem proposta neste trabalho serviu de alicerce para trabalhos futuros que envolvam a utilização de módulos de contramedidas de IDS para infraestruturas de computação em nuvem.

7.2 Trabalhos futuros

O tema desta monografia foi sistemas de detecção de intrusão virtualizados para computação em nuvem. Contribuições futuras nessa área incluem:

- Análise forense das máquinas virtuais suspensas pelo componente Reaction, a fim de se verificar a procedência da intrusão;
- Utilização de *honeypots* juntamente com o IDS, para facilitar a análise forense, indicando possíveis suspeitos ou procedimentos de intrusão;
- Expansão da capacidade de detecção do IDS, utilizando-se de técnicas sofisticadas de detecção por anomalia, como reconhecimento de padrões ou padronização de comportamento dos usuários;
- Expansão da capacidade do IDS, implementando um módulo de previsão de ataques, utilizando métodos de aprendizagem de máquina para detectar possíveis intrusões;
- Adequação do componente Reaction à expansão do IDS, suspendendo VMs nos casos necessários, a depender dos novos tipos de ataques.

BIBLIOGRAFIA

[1] "Arno's Projects," [Online]. Available: http://rocky.eld.leidenuniv.nl/joomla/. [Acesso em 20 Agosto 2013].

[2] J. D. ARAÚJO e Z. ABDELOUAHAB, "Um modelo de detecção de intrusão para ambientes de computação em nuvem," São Luís, 2013.

[3] "Virtela," Virtela Technology Services Incorporated, 2012. [Online]. Available: http://www.virtela.net/services/managed-security-services/ips/. [Acesso em 20 Agosto 2013].

[4] P. Mell e T. Grance, "The NIST Definition of Cloud Computing," National Institute of Standards and Technology, Gaithersburg, 2011.

[5] M. MAHJOUB , A. MDHAFFAR , R. B. HALIMA e M. JMAIEL, "A comparative study of the current Cloud Computing technologies and offers," *First International Symposium on Network Cloud Computing and Applications,* pp. 131-134, 2011.

[6] V. G. Deshmukh, A. G. Borkut e N. A. Agam, "Intrusion Detection System For Cloud Computing," *International Journal of Engineering Research & Technology (IJERT),* vol. 2, n. 4, pp. 2149-2153, April 2013.

[7] S. Roschke, F. Cheng e C. Meinel, "An Extensible and Virtualization-Compatible IDS Management Architecture," *Fifth International Conference On Information Assurance and Security,* pp. 130-134, 2009.

[8] S. Rajan e A. Jairath, "Cloud Computing: The Fifth Generation of Computing," *2011 International Conference on Communication Systems and Network Technologies (CSNT),* pp. 665-667, 3-5 June 2011.

[9] M. Armbrust e e. al., "Above the Clouds: A Berkeley View of Cloud Computing," Berkeley, 2009.

[10] I. Brandic e S. Dustdar, "Grid vs Cloud – A Technology Comparison," *Information Technology 4,* pp. 173-179, 2011.

[11] F. R. C. Sousa, L. O. Moreira e J. C. Machado, "Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios," [Online]. Available: http://www.es.ufc.br/~flavio/papers/ercemapi2009.pdf. [Acesso em 22 Agosto 2013].

[12] T. J. Velte, . A. T. Velte e R. Elsenpeter, Cloud Computing, a Practical Approach, 1ª ed., New York, NY: McGraw-Hill, Inc., 2010.

[13] "Google Apps for Business," Google, 2013. [Online]. Available: http://www.google.com/intx/pt-BR/enterprise/apps/business/. [Acesso em 23 Agosto 2013].

[14] "Basecamp," 37signals, LLC, 1999-2013. [Online]. Available: https://basecamp.com/. [Acesso em 23 Agosto 2013].

[15] "Dropbox," Dropbox, Inc., [Online]. Available: https://www.dropbox.com/. [Acesso em 23 Agosto 2013].

[16] "OpenShift," Red Hat, Inc., [Online]. Available: https://www.openshift.com/. [Acesso em 26 Agosto 2013].

[17] "Google App Engine," Google, [Online]. Available: https://developers.google.com/appengine/. [Acesso em 26 Agosto 2013].

[18] "Postgres Plus Cloud Database," [Online]. Available: http://www.enterprisedb.com/cloud-database. [Acesso em 26 Agosto 2013].

[19] “Eucalyptus,” Eucalyptus Systems, Inc., [Online]. Available: http://www.eucalyptus.com/. [Acesso em 26 Agosto 2013].

[20] “OpenStack,” [Online]. Available: http://www.openstack.org/. [Acesso em 26 Agosto 2013].

[21] “Windows Azure,” Microsoft©, [Online]. Available: http://www.windowsazure.com/. [Acesso em 26 Agosto 2013].

[22] J. Hurwitz, R. Bloor, M. Kaufman e F. Halper, “Discovering Private and Hybrid Clouds,” em *Cloud Computing for Dummies*, Hoboken, New Jersey: Wiley Publishing, Inc., 2009, pp. 87-104.

[23] D. SMITH and e. all, "Hype Cicle for Cloud Computing," *Gartner’s 2009 Hype Cycle Special Report Evaluates Maturity of 1,650 Technologies,* pp. 1-56, jul 2009.

[24] C. Chede, “Cuidados com Lock-in em Cloud Computing,” IBM®, 27 April 2010. [Online]. Available: https://www.ibm.com/developerworks/community/blogs/ctaurion/entry/cuidados_com_lock_in_em_cloud_computing1?lang=en. [Acesso em 26 Agosto 2013].

[25] S. Ortiz, “The Problem with Cloud-Computing Standardization,” *Computer,* vol. 44, n. 7, pp. 13-16, July 2011.

[26] “Deltacloud API,” Apache Software Foundation, [Online]. Available: http://deltacloud.apache.org/. [Acesso em 26 Agosto 2013].

[27] “Apache Libcloud,” The Apache Software Foundation, [Online]. Available: http://libcloud.apache.org/. [Acesso em 26 Agosto 2013].

[28] “Simple Cloud API,” Zend Technologies Inc., [Online]. Available: http://www.simplecloud.org/. [Acesso em 26 Agosto 2013].

[29] “Open Cloud Manifesto,” 2009. [Online]. Available: http://www.opencloudmanifesto.org/Open%20Cloud%20Manifesto.pdf. [Acesso em 26 Agosto 2013].

[30] “Adobe,” [Online]. Available: http://www.adobe.com/. [Acesso em 26 Agosto 2013].

[31] “HP Converged Cloud,” [Online]. Available: http://www8.hp.com/us/en/business-solutions/cloud-computing.html. [Acesso em 26 Agosto 2013].

[32] “IBM,” [Online]. Available: http://www.ibm.com/cloud-computing/us/en/. [Acesso em 26 Agosto 2013].

[33] M. A. P. Laureano e C. A. Maziero, “Virtualização: Conceitos e Aplicações em Segurança,” *Minicursos do VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais,* pp. 1-49, 2008.

[34] L. DUFLOT, O. GRUMELARD, O. LEVILLAIN e B. MORIN, “On the Limits of Hypervisor- and Virtual Machine Monitor-Based Isolation,” *Towards Hardware-Intrinsic Security, Information Security and Cryptography,* 2010.

[35] “QEMU open source processor emulator,” 2013. [Online]. Available: http://wiki.qemu.org/. [Acesso em 28 Outubro 2013].

[36] “VMware,” VMware, Inc. ©, 2013. [Online]. Available: http://www.vmware.com/. [Acesso em 28 Outubro 2013].

[37] “VirtualBox,” Oracle ©, 2013. [Online]. Available: https://www.virtualbox.org/. [Acesso em 11 Setembro 2013].

[38] “Xen Project,” Xen Project, A Linux Foundation Collaborative Project ©, 2013. [Online]. Available: http://www.xenproject.org/. [Acesso em 28 Outubro 2013].

[39] "Visão geral do Hyper-V," Microsoft ©, 2013. [Online]. Available: http://technet.microsoft.com/pt-BR/library/hh831531.aspx. [Acesso em 28 Outubro 2013].

[40] K. Scarfone e P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," Gaithersburg, 2007.

[41] T. M. Wu, "Intrusion Detection Systems," Fort Belvoir, 2009.

[42] F. SABAHI e A. MOVAGHAR, "Intrusion Detection: A Survey," *The Third International Conference on Systems and Networks Communications,* pp. 23-26, Outubro 2008.

[43] R. U. REHMAN, Intrusion Detection Systems with Snort, New Jersey: Prentice Hall PTR, 2003, pp. 75-129.

[44] S.-J. Horng, M.-Y. Su, Y.-H. Chen, T.-W. Kao, R.-J. Chen, J.-L. Lai e C. D. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Expert Systems with Applications,* vol. 38, pp. 306-313, Janeiro 2011.

[45] H.-J. Liao, C.-H. R. Lin, Y.-c. Lin e K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications,* vol. 36, pp. 16-24, Janeiro 2013.

[46] "SNORT," SNORT, 2010. [Online]. Available: http://www.snort.org/. [Acesso em 16 Julho 2013].

[47] Z. Li, Y. Chen e B. Liu, "NetShield," Northwestern University, [Online]. Available: http://www.nshield.org/. [Acesso em 16 Julho 2013].

[48] M. Jamshed, J. Lee, S. Moon, I. Yun, D. Kim, S. Lee, Y. Yi e K. Park, "Kargus: A Highly-scalable Software-based NIDS," Korea Advanced Institute of Science and Technology, 30 Novembro 2012. [Online]. Available: http://shader.kaist.edu/kargus/. [Acesso em 16 Julho 2013].

[49] C. Endorf, J. Mellander e E. Schultz, "Intrusion detection & prevention," em *Intrusion detection & prevention*, New York, McGraw Hill Professional, 2004, pp. 345-359.

[50] R. Koch, "Towards Next-Generation Intrusion Detection," *Proceedings of the 3rd International Conference on Cyber Conflict (ICCC),* pp. 151-168, Junho 2011.

[51] E. Messmer, "Gartner: Long hard climb to high level of cloud computing security," Network World, 10 Abril 2013. [Online]. Available: http://www.networkworld.com/news/2013/041013-gartner-cloud-security-268587.html. [Acesso em 18 Julho 2013].

[52] B. Grobauer, T. Walloschek e S. E. , "Understanding Cloud Computing Vulnerabilities," *Security Privacy, IEEE,* vol. 9, n. 2, pp. 50-57, Março - Abril 2011.

[53] "Risk Taxonomy," The Open Group, Berkshire, 2009.

[54] Cloud Security Alliance, "The Notorious Nine Cloud Computing Top Threats in 2013," 2013.

[55] "Obtenha Informações sobre a Tecnologia Java," Oracle ©, 2013. [Online]. Available: http://www.java.com/pt_BR/about/. [Acesso em 11 Setembro 2013].

[56] "Ubuntu 12.04.3 LTS (Precise Pangolin)," Canonical Ltd. ©, 2013. [Online]. Available: http://releases.ubuntu.com/precise/. [Acesso em 11 Setembro 2013].

[57] "Kernel Based Virtual Machine," Red Hat, 2013. [Online]. Available: http://www.linux-kvm.org/page/Main_Page. [Acesso em 11 Setembro 2013].

[58] "libvirt Virtualization API," Red Hat, 2013. [Online]. Available: http://libvirt.org/. [Acesso em 11 Setembro 2013].

[59] “hping,” 2013. [Online]. Available: http://www.hping.org/. [Acesso em 30 Outubro 2013].

[60] Gartner Inc., “Gartner,” Gartner, 2013. [Online]. Available: http://www.gartner.com/technology/home.jsp. [Acesso em 18 Julho 2013].

APÊNDICE A – Código Fonte do Módulo de contramedidas Reaction

A.1 Introdução

Este apêndice apresenta o código fonte do módulo de contramedidas Reaction, utilizado na suspensão das máquinas virtuais suspeitas do ambiente virtualizado de computação em nuvem. O código foi implementado utilizando a linguagem de programação Java [55], sendo devidamente comentado.

A.2 Código Fonte

```
package packet_ids;

import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.text.SimpleDateFormat;

public class Reaction {
	String attack; // attack name
	String ip_malicious; // endereço IPv4 da VM suspeita (vítima ou atacante)
	String name_malicious_vm; // nome de host da VM suspeita (ou vítima)
	String malicious_user; // nome do usuário (ou masquerading) online na VM suspeita (ou
vítima)

	public Reaction(String a, String ip) {
		this.attack = a;
		this.ip_malicious = ip;
	}

	/**
	 * do_reaction realiza a contramedida necessária.
	 */
	public String do_reaction() {
		String data = "dd/MM/yyyy";
		String hora = "HH:mm";
		String data1, hora1, date;
		java.util.Date agora = new java.util.Date();
		SimpleDateFormat formata = new SimpleDateFormat(data);
		data1 = formata.format(agora);
		formata = new SimpleDateFormat(hora);
		hora1 = formata.format(agora);
		date = String.format(data1 + " " + hora1);

		String line = null, display = null, alert = "";
		try {
			Runtime run = Runtime.getRuntime();

			// resolve vm host name
			Process process_name = run.exec("resolveip -s " + ip_malicious);
			if (process_name == null) {
				System.out.println("\nError in resolve name.");
			} else {
				System.out.println("\nResolving name...");
			}
			BufferedReader in_resolve = new BufferedReader(
					new InputStreamReader(process_name.getInputStream()));
			while ((line = in_resolve.readLine()) != null) {
				if (line != null) {
					name_malicious_vm = line;
```

```
                        }
                    }
                    line = "";

                    // get user name
                    Process process_username = run.exec("ssh " + ip_malicious
                            + " 'users'");
                    if (process_username == null) {
                        System.out.println("Error in getting user name.");
                    } else {
                        System.out.println("Getting user name...");
                    }
                    BufferedReader in_user = new BufferedReader(new InputStreamReader(
                            process_username.getInputStream()));
                    while ((line = in_user.readLine()) != null) {
                        if (line != null) {
                            malicious_user = line;
                        }
                    }

                    // suspend VM
                    Process process_suspend = run.exec("virsh suspend "
                            + name_malicious_vm);
                    if (process_suspend == null) {
                        System.out.println("Erro na suspensão da VM maliciosa.");
                    } else {
                        System.out.println("Suspending malicious (victim) VM...");
                    }

                    // ping para outra vm ou broadcast
                    if (attack.equals("probattack")) {
                        display = String.format("Malicious VM hostname: "
                                + name_malicious_vm);
                        // colocar o usuário na blacklist
                        PrintStream p = new PrintStream(new BufferedOutputStream(
                                new FileOutputStream("/ids/blacklist", true)));
                        p.println(date + " " + malicious_user);
                        p.close();

                        alert = String.format(display + " suspended; Malicious user: "
                                + malicious_user + "." + "\n" + malicious_user
                                + " added on backlist.");
                        // ataque synflood ou smurf
                    } else if (attack.equals("synflood") || attack.equals("smurf")) {
                        display = String.format("Victim VM hostname: "
                                + name_malicious_vm);
                        alert = String.format(display + " suspended; Victim user: "
                                + malicious_user
                                + "\nMigrate respective VM to other host.");
                    // alertar ao admin sobre a suspeita de ataque indicando a possibilidade
                    // de migrar a respectiva VM, tomando o cuidado para que seja realizada
                    // uma auditoria no ambiente de rede a fim de que providências sejam
                    // tomadas para paralizar o respectivo ataque.
                    }

                    System.out.println(display);
                } catch (IOException e) {
                    e.printStackTrace();
                }
                return alert; // retorna para Analyser
        }

}
```