Estêvão Samuel Procópio

Detecção e bloqueio de malwares
Estudo de caso na UFVJM

Monografia de Pós-Graduação "*Lato Sensu*" apresentada ao Departamento de Ciência da Computação para obtenção do título de Especialista em "Administração em Redes Linux"

Orientador
Prof. Arlindo Follador Neto

Lavras
Minas Gerais - Brasil
2011

Estêvão Samuel Procópio

Detecção e bloqueio de malwares
Estudo de caso na UFVJM

Monografia de Pós-Graduação "*Lato Sensu*" apresentada ao Departamento de Ciência da Computação para obtenção do título de Especialista em "Administração em Redes Linux"

Aprovada em *01/05/2010*

Prof. Dr. Joaquim Quinteiro Uchôa

Prof. Eric

Prof. Arlindo Follador Neto
(Orientador)

Lavras
Minas Gerais - Brasil
2011

Dedico o presente trabalho primeiramente a Deus que, pela sua imensa graça e misericórdia, me deu força, saúde e capacidade para enfrentar esse desafio.

Dedico-o também à minha amada noiva e futura esposa Tatiana Magalhães Amaral que me ajudou e permaneceu ao meu lado durante todo o desenvolvimento desse trabalho com paciência, amor, atenção e dedicação, trazendo-me palavras de apoio e incentivo.

Agradecimentos

Ao meu orientador Arlindo Follador Neto por aceitar o desafio, acreditar na possibilidade e incentivar a conclusão desse trabalho.

Sumário

Lista de Figuras

Resumo

No cenário atual de redes corporativas um dos grandes problemas são os programas maliciosos (*malwares*). Vírus e *worms* são alguns deles e sua principal característica é se multiplicar utilizando a *Internet*. Os riscos trazidos vão além da exposição de dados confidenciais, o aumento de tráfego na rede ou a possibilidade de multiplicação. Este projeto propõe um sistema de detecção e bloqueio de computadores suspeitos de infecção com o intuito de reduzir os riscos causados por *malwares* na UFVJM e consequentemente a sua proliferação.

Palavras-Chave: *Malwares*; Análise de tráfego; Controle de acesso.

Capítulo 1

Introdução

No cenário de redes corporativas um dos grandes problemas são os programas maliciosos (*malwares*) e a sua proliferação. Vírus e vermes (*worms*) são alguns deles e sua principal característica é a de se espalharem pela máquina[1] e até mesmo pela rede interna ou *Internet*.

O termo vírus de computador foi primeiramente utilizado na década de 80, por Fred Cohen, em seu trabalho intitulado *Computer Viruses - Theory and Experiments*. Cohen (1984) define vírus de computador como "um programa que pode infectar outros programas modificando-os para incluir neles uma cópia possivelmente evoluída de si mesmo". Dessa forma, todo programa infectado agirá também como vírus, podendo infectar outros programas na mesma máquina em que for executado.

Com a popularização das redes locais e suas interconexões com a *Internet* os programas maliciosos não somente aumentaram em quantidade como também evoluíram. Esta evolução começou a partir da descoberta de falhas de segurança que podem ser exploradas remotamente (via rede local ou *Internet*), permitindo execução de códigos no computador que as possui. Assim, esses programas maliciosos começaram a ser implementados para se replicarem valendo-se dessas falhas de segurança descobertas.

Enquanto os vírus necessitam que o usuário execute o programa malicioso para que a infecção ocorra, os vermes são capazes de se replicarem sem intervenção humana, buscando na rede (local ou *Internet*) máquinas com falhas de

[1] Este trabalho utiliza o termo *máquina* para identificar computadores pessoais, notebooks e estações de trabalho de uma determinada instituição.

segurança para serem exploradas. Cada máquina conectada à rede passa a ser um possível hospedeiro para o verme.

1.1 Objetivos

O presente projeto visa o desenvolvimento de um sistema livre para detecção e bloqueio de máquinas suspeitas de infecção por programas maliciosos com o intuito de reduzir as infecções por *malware* na instituição e também diminuir a possibilidade de proliferação da infecção para outras redes, isolando o tráfego das máquinas infectadas internamente à instituição. Para tanto, torna-se necessário investigar o comportamento desses programas maliciosos e analisar o tráfego gerado por eles na rede ao se replicarem ou atualizarem a fim de serem definidos os processos de identificação das infecções.

Tal sistema será desenvolvido, testado e implantado na UFVJM[2] visando reduzir as infecções por *malware* das estações de trabalho da instituição. Essas infecções foram detectadas primeiramente pelo CAIS - RNP[3], que reportou à UFVJM enviando alertas por e-mail para informar da existência da propagação de programas maliciosos partindo da instituição. Esse sistema fará a análise do tráfego na rede da UFVJM e, a partir das anomalias detectadas, efetivará o bloqueio das máquinas, impedindo que a infecção se alastre por outras redes de computadores.

1.2 Metodologia

As bases metodológicas utilizadas para o desenvolvimento deste trabalho foram a pesquisa bibliográfica e a pesquisa de campo. A pesquisa bibliográfica foi utilizada com o objetivo de embasar teoricamente o projeto e dar domínio instrumental e teórico sobre *malwares* e seus mecanismos de replicação e atualização. Paralelamente ao desenvolvimento da pesquisa bibliográfica deu-se a pesquisa de campo, onde foram utilizadas ferramentas livres para detecção do tráfego, bem como pesquisas e implementações de programas para bloqueio dessas máquinas, limitando o tráfego gerado pelos programas maliciosos à rede na qual a máquina infectada se encontra.

[2]Universidade Federal dos Vales do Jequitinhonha e Mucuri

[3]Centro de Atendimento e Incidentes de Segurança da Rede Nacional de Pesquisa, à qual a UFVJM é filiada.

1.3 Organização do Trabalho

O **Capítulo 1** apresenta uma breve introdução e histórico sobre programas maliciosos, enquanto o **Capítulo 2** apresenta um pouco mais detalhadamente o funcionamento desses programas, seu impacto em redes corporativas e as formas utilizadas atualmente para a detecção dessas pragas. O **Capítulo 3** apresenta a instituição onde o presente trabalho foi desenvolvido e as ferramentas que foram utilizadas, adaptadas e desenvolvidas para identificar e isolar da *Internet* as máquinas infectadas por programas maliciosos. O **Capítulo 4** apresenta os experimentos e também os resultados obtidos da execução deste projeto. O **Capítulo 5** apresenta as conclusões do autor e o **Capítulo 6** mostra os trabalhos que serão desenvolvidos posteriormente relacionados ao presente estudo.

Capítulo 2

Malwares

O presente capítulo apresenta o conceito de *malwares*, seus principais tipos, os problemas causados quando acontecem infecções em redes corporativas e quais formas podem ser utilizadas para realizar a detecção dessas pragas virtuais.

2.1 Apresentação

O termo *Malware* é utilizado para referenciar genericamente a programas maliciosos. Esses programas são classificados em categorias baseadas em suas características. Algumas dessas classificações são:

***Adwares*:** programas que geram *popups* de propaganda quando se conectam à *Internet*;

***Betrayware*:** programas que dizem remover *malwares*, mas que, na verdade, instalam mais pestes na máquina;

Cavalos de Tróia: programas que parecem fazer bem, porém contém código malicioso. *Betraywares* são bons exemplos de Cavalos de Tróia.

***Loggers*:** programas que capturam as teclas digitadas pelo usuário e também imagens do estado atual da tela e enviam as enviam para um e-mail previamente configurado.

***Spyware*:** programas que coletam informações sobre o uso do computador e navegação pela *Internet* e disponibilizam em um banco de dados para que posteriormente essas informações sejam usadas para *marketing* e *spam*;

Vírus: programas desenvolvidos para se replicarem, infectando arquivos executáveis, documentos ou outros tipos de arquivos em uma máquina;

Vermes: são vírus que se replicam e infectam outras máquinas valendo-se de falhas de segurança conhecidas e exploráveis via rede.

Um breve histórico mostra que em 1982 John Shoch e Jon Hupp, pesquisadores da Xerox de Palo Alto, fizeram seus primeiros experimentos com sistemas distribuídos. Os experimentos se basearam em um romance escrito por John Brunner intitulado *The Shockwave Ride*, a partir da ideia de um verme que percorre uma rede de computadores. Nesse estudo, Shoch e Hupp (1982) definem verme como sendo um sistema de computação distribuída em uma ou mais máquinas. Cada máquina específica é chamada de segmento do verme e esses segmentos permanecem em comunicação entre si. Caso haja falha em algum segmento, o verme automaticamente busca uma nova máquina, a inicializa e a adiciona ao sistema de computação. Desse modo, como máquinas entram e saem desse sistema, o verme parece se mover através da rede. A esse estudo foi dado o nome de *The "Worm" Programs*.

O principal problema encontrado na concepção de Shoch e Hupp (1982) foi de como controlar o crescimento de um verme e, ao mesmo tempo, manter a estabilidade do sistema. Os primeiros experimentos foram um pouco caóticos, porém o sistema foi melhorado e foi possível desenvolver aplicações distribuídas como testes de tráfego *Ethernet* utilizando o verme. Esse Programa da Xerox foi responsável pela disseminação do termo verme para programas de computador que infectavam máquinas pela rede sem a necessidade de intervenção do usuário.

A primeira grande infecção por programa malicioso que foi noticiada ficou conhecida como a helmintíase da *Internet*. O termo helmintíase remete a uma infecção por helmintos (vermes parasitários). Esse evento, ocorrido em novembro de 1988, foi causado por um verme, posteriormente denominado de *Morris* em homenagem ao seu autor. Esse verme infectava máquinas VAX e SUN-3[1] com sistema operacional BSD UNIX[2] nas versões 4.2 ou 4.3, explorando vulnerabili-

[1] Computadores de 32 bits utilizados nas décadas de 70 e 80. As máquinas VAX era fabricadas pela DEC (Digital Equipment Corporation) e as SUN-3, conforme o nome indica, pela Sun Microsystems.

[2] Berkeley Software Distribution UNIX é um sistema operacional UNIX desenvolvido pela Universidade de Berkeley, na Califórnia, durante os anos 70 e 80.

dades do *sendmail*[3] e do *fingerd*[4]. Após infectar a máquina vítima, o verme tentava se conectar em máquinas conhecidas da máquina infectada, buscando informações nos arquivos /etc/hosts.equiv, .rhosts[5], .forward[6], etc. Além disso, tentava quebrar as senhas utilizadas pelos usuários. Como consequência eram percebidas quedas de desempenho nas máquinas infectadas. Falta de espaço em disco também foram notadas, visto que os processos iniciados pelo verme criavam vários arquivos temporários (REYNOLDS, 1989).

A cura somente foi possível após análise e entendimento das formas de propagação do verme. Foram criadas correções para as falhas de segurança do *sendmail* e do *fingerd* que ele explorava. Com as máquinas atualizadas, o verme não poderia mais se espalhar pela rede.

Os programas maliciosos em geral estão evoluindo cada vez mais e trazendo grandes riscos às redes corporativas. Um desses riscos é a utilização desses programas para a criação de *botnets*. Tem-se uma *botnet* quando um conjunto de máquinas infectadas por determinado programa malicioso tem a capacidade de receber comandos do seu criador pela *Internet*. Desse modo, o criador do programa malicioso tem a possibilidade de controlar uma quantidade relativamente alta de máquinas espalhadas pela rede para fazer o que bem entender.

Em um estudo sobre a evolução dos *malwares*, Chen e Robert (2004) listaram as principais características dos programas maliciosos modernos, dentre as quais se destacam a utilização de diversos meios para se proliferarem: exploração de falhas que permitem execução remota de código e também pelo envio do programa malicioso por e-mails, transferência de arquivos em redes ponto-a-ponto e mensagens instantâneas. Outra característica que merece destaque é a atualização dinâmica pela *Internet*. Através dessas atualizações, um verme pode evoluir, aprender novas formas de infecção e se proliferar cada vez mais. Alguns desses vermes foram capazes de infectar mais de 300.000 máquinas apenas nas primeiras doze horas de atividade. Chen e Robert (2004) concluem o trabalho apontando para necessidade de um sistema com capacidade de detectar rapidamente e sem erros os sinais desses vermes visando isolar os computadores infectados do resto da *Internet*, a fim de contar a infecção.

[3] Agente de transferência de correio eletrônio (*MTA*).

[4] Serviço utilizado para fornecer informações sobre os usuários de um determinado computador.

[5] /etc/hosts.equiv e .rhosts são arquivos de configuração do *remote shell* (*rsh*) que criam relações de confiança entre máquinas, permitindo que usuários de outras máquinas conectem sem necessidade de autenticação por senha.

[6] Arquivo de configuração de encaminhamento de e-mail, informando o novo endereço de e-mail a ser utilizado para encaminhas as mensagens que chegarem para determinado usuário.

2.2 *Malwares* em Redes Corporativas

Ao infectarem máquinas de redes corporativas, os programas maliciosos trazem diversos riscos, como a exposição de dados confidenciais e a possibilidade da infecção se espalhar pela rede interna causando aumento do tráfego na rede. Além desses riscos, há ainda a possibilidade de recrutamento de máquinas para *botnets*, normalmente utilizadas para cometer crimes *online*, como efetuar ataques de negação de serviço a grandes empresas. Nesses ataques, as máquinas recrutadas pelo *malware* são utilizadas para fazer requisições a um determinado servidor a fim de esgotar sua capacidade de processar requisições. Dependendo da quantidade de máquinas recrutadas e do poder de processamento do servidor alvo, esse ataque pode prejudicar a prestação do serviço ou até mesmo tornar o serviço inacessível.

Os problemas previamente citados ficam em segundo plano quando são abordados os prejuízos financeiros que que podem ocorrer para a instituição afetada. Uma associação bancária do Reino Unido estimou que as perdas diretas causadas por *malwares* aos seus membros foram de 12,2 milhões de libras em 2004, aumentando em 90% (23,2 milhões de libras) em 95 e ainda subindo 44% (33,5 milhões de libras) em 2006 (WHITTAKER, 2007).

Consequências piores do que essas também podem ser causadas em casos de programas maliciosos espalhando-se em redes internas de hospitais. Segundo Pompon (2010), uma infecção por *malware* em janeiro de 2005 levou o Northwest Hospital de Seattle a tomar medidas de emergência. O tráfego gerado pela praga foi tão intenso que os serviços de laboratório foram impedidos de operar, terminais da UTI foram desconectados e até mesmo as portas das salas de cirurgia, que eram controladas por computador, ficaram travadas, visto que os equipamentos não conseguia se comunicar corretamente pela rede em consequência do alto tráfego de dados gerado pela infecção. O plano de contingência e recuperação de desastres foi posto em prática e salvou o hospital do pior. Entretanto, as ações não impediram os transtornos causados tanto para a equipe técnica quanto para os pacientes que tiveram suas cirurgias reagendadas por causa do incidente.

Os casos citados anteriormente mostram a importância de um sistema de detecção de *malwares* em uma rede corporativa. A possibilidade de identificar as maquinas infectadas leva à tomada de decisões e criação de planos de ação práticos visando a remoção dessas pragas e a restauração do funcionamento da instituição.

Como no caso do Northwest Hospital, o tráfego gerado na rede pode se tornar evidente, pois para infectar novas máquinas, eles necessitam fazer alguns acessos pela rede, buscando máquinas vulneráveis. Outro tipo de acesso à rede gerado

por programas maliciosos em geral é para buscar atualizações a serem instaladas ou comandos a serem executados. Tais acessos normalmente são feitos via HTTP para evitar possíveis bloqueios por *firewall*[7].

Um estudo feito por Cooke, Jahanian e McPherson (2005) mostra que o tráfego de propagação de programas maliciosos podem ocorrer utilizando diferentes mecanismos, como compartilhamentos de arquivo que não utilizem senhas ou utilizem senhas fracas, transmissão por redes ponto a ponto, utilizando-se de *backdoors*[8] deixadas por outros programas maliciosos, ou mesmo explorando falhas conhecidas, divulgadas em boletins de segurança. Além disso, o estudo mostra também que os programas maliciosos atuais estão utilizando formas de ataque e comunicação extremamente avançadas. Em alguns casos, o mesmo *malware* é capazes de executar ataques de negação de serviço, captura de senhas, envio de *spam* etc.

Bacher *et al.* (2005) utilizaram *honeynets*[9] para verificar o funcionamento e disseminação de alguns programas maliciosos que recrutam máquinas para montar *botnets*. Nesse estudo, foi detectado que mais de 80% do tráfego capturado estava relacionado a requisições de serviços NetBIOS (portas 137/UDP e 129/TCP), Microsoft-DS (porta 445/TCP) e Microsoft RPC (porta 135/TCP). Uma parte menor do tráfego estudado por Bacher *et al.* (2005) mostra que os programas maliciosos exploram vulnerabilidades conhecidas nos serviços IIS/Apache, Microsoft SQL Server, Microsoft UPnP, dentre outros.

2.3 Formas de Detecção

Atulmente as principais ferramentas utilizadas na detecção de *malwares* são desenvolvidas para serem utilizadas individualmente em cada computador. Alguns exemplos são os antivírus e *antispywares*. Os estudos de Kolbitsch *et al.* (2009) mostram que tais ferramentas possuem modelos de detecção ineficientes, pois se concentram em um *malware* específico ou em uma determinada família de *malwares* com o comportamento similar. A detecção é baseada em assinaturas, onde uma sequência de bytes que identifica unicamente um determinado *malware* ou

[7] Dispositivo normalmente utilizado para proteger redes de acesso não autorizado pela sua capacidade de permitir ou bloquear transmissões de dados em uma rede baseando-se em regras.

[8] Um método obter acesso a um computador burlando os mecanismos normais de autenticação.

[9] Uma *honeynet* é uma rede projetada especificamente para ser comprometida e que contém mecanismos de controle para previnir que seja utilizada como base para ataques contra outras redes. Uma *honeynet* é uma ferramenta útil para se estudar a forma como estão sendo feitos os ataques a redes de computadores.

família são armazenados em um bando de dados e comparados com os arquivos executáveis do computador. Além de não conseguirem detectar novas pragas sem a atualização do banco de dados de assinaturas, essa abordagem pode ser facilmente burlada por técnicas como ofuscamento e polimorfismo (KOLBITSCH *et al.*, 2009).

Uma abordagem diferente que vem sendo explorada é baseada na análise do tráfego gerado por computadores infectados. Normalmente essa análise é feita através de sistemas de detecção de intrusão (*intrusion detection systems* ou *IDSs*). Esses sistemas monitoram o tráfego da rede buscando por atividades maliciosas, anomalias ou violações da política de segurança e produzem relatórios de alertas.

Uma técnica bastante utilizada consiste em detectar as atividades maliciosas através da comparação do conteúdo do pacote trafegado na rede com padrões predefinidos, também conhecidos como assinaturas. Entretanto essas assinaturas não são dos prórprios vermes, mas sim de falhas de segurança que tais vermes utilizam para se espalharem. Desse modo, a detecção via análise de tráfego, mesmo por assinatura, pode ser considerada mais eficiente do que a busca local por executáveis infectados, visto que vários vermes utilizam uma mesma falha de segurança para se replicarem.

Outra técnica utilizada na detecção por análise de tráfego é a classificação de tráfego e detecção de anomalias por análise estatística. Essas análises consistem na utilização de métodos probabilísticos de comparação de conjuntos de tráfegos identificados como um conjunto normal. Se um determinado tráfego tiver uma baixa probabilidade de fazer parte desse conjunto normal, um alerta é emitido indicando tal anomalia (TAYLOR; ALVES-FOSS, 2001).

Entretanto, algumas considerações precisam ser feitas ao utilizar um *IDS* para monitoramento de tráfego e detecção de *malwares*. Em redes de computadores baseadas em *switches*, os tráfegos são direcionais, indo da origem para o destino sem que seja feita difusão dos pacotes para toda a rede. Esse fato impede o *IDS* de analizar a parte mais significativa do tráfego gerado, que é a comunicação ponto a ponto entre computadores da rede interna.

Existem algumas técnicas que podem ser utilizadas para solucionar esse problema. A mais confiável seria a utilização de *switches* com suporte a *port mirroring*, que consiste na configuração do *switch* para que todo tráfego que passar por ele, independentemente de origem e destino, seja espelhado em uma determinada porta onde o *IDS* estaria conectado.

Caso os *switches* da rede não possuam o recurso de *port mirroring*, existe ainda a possiblidade de se instalar um *bridge*[10] entre eles, conforme a Figura 2.1. Dessa forma, todo o tráfego que passar do *Switch 1* para o *Switch 2* será capturado e analizado pelo *IDS* instalado na *bridge*. Entretanto, o tráfego entre máquinas conectadas a um mesmo *switch* não poderá ser capturado, caracterizando a limitação dessa abordagem.

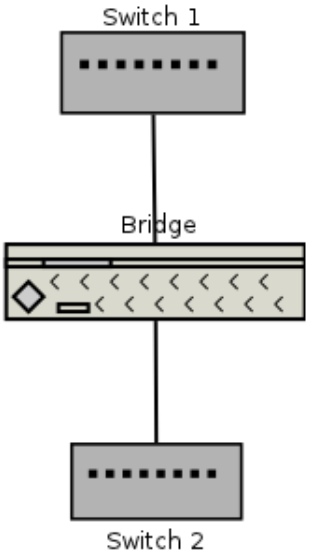

Figura 2.1: Utilização de *bridge* entre os *switches*

[10]Componente de rede utilizado para interligar dois ou mais segmentos de rede.

Capítulo 3

Metodologia

O presente capítulo apresenta o uma breve descrição do ambiente onde o trabalho foi desenvolvido e os detalhes da implementação da detecção das infecções, a visualização das informações e o bloqueio das máquinas infectadas. Por fim, as experiências feitas antes da implantação do sistema são apresentadas.

3.1 Ambiente

A Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM) é constituída de três *campi*. O Campus I e o Campus JK estão localizados na cidade de Diamantina (MG), abrigando seis faculdades e vinte e três cursos de graduação; e o Campus Avançado do Mucuri, que se localiza em Teófilo Otoni (MG) e abriga três faculdades com nove cursos de graduação.

O Campus JK está em fase de expansão. Novos prédios estão sendo construídos para abrigar tanto a parte administrativa quanto acadêmica da Universidade. Alguns setores já foram transferidos do Campus I para o Campus JK e a previsão é que toda a parte administrativa seja transferida para este Campus até o final do ano de 2011. Para comportar esse crescimento foram elaborados projetos de aquisição de novos ativos de rede e também da substituição dos ativos atuais por equipamentos gerenciáveis. Existem também projetos para melhoria da infraestrutura lógica da rede da UFVJM, como a segmentação em subredes e *vlans*[1] visando uma maior segurança e isolamento das máquinas, otimização do tráfego etc. A atual infraes-

[1] subredes virtuais configuradas em *switches* gerenciaveis de camada 2

trutura não possui nenhuma segmentação, tornando o ambiente mais propício para a multiplicação de programas maliciosos.

A UFVJM conta com aproximadamente 500 funcionários, entre professores e técnico-administrativos, e com uma infraestrutura de aproximadamente 2200 estações de trabalho. O acesso à *Internet* da UFVJM se dá através de um *link* com a RNP (Rede Nacional de Pesquisa). Também via RNP se dá a interligação entre os *campi* de Diamantina e de Teófilo Otoni. A interligação entre os *campi* localizados na cidade de Diamantina é feita via rede *wireless* da própria instituição.

A instituição possui ainda um *gateway*[2] que utiliza Debian GNU/Linux[3] como sistema operacional. O *gateway* da instituição é responsável por alguns serviços como *DHCP*[4], *firewall* e *web proxy*[5], utilizando, respectivamente, *ISC DHCP server*[6], *netfilter/iptables*[7], *Squid*[8] e *squidGuard*[9]. O *squidGuard* não é necessário para o serviço de *web proxy*, porém é utilizado na instituição para fazer redirecionamento de URLs (endereços *web*) como, por exemplo, para baixar as atualizações do Ubuntu Linux de um repositório local da instituição ao invés da internet, aumentando a velocidade das atualizações desses sistemas operacionais e também otimizando a utilização da banda da instituição.

Os servidores *web* e de banco de dados da instituição são virtualizados utilizando *Xen*[10] e também utilizam o sistema operacional Debian GNU/Linux. O servidor *web* possui instalado o *Apache*[11] com módulo de execução de páginas

[2] máquina responsável por interligar duas ou mais redes diferentes

[3] Maiores informações em `http://debian.org`

[4] Serviço que automatiza a configuração de endereçamento para máquinas da rede

[5] Serviço que permite a otimização do uso da banda fazendo *cache* de requisições previamente feitas por outros usuários. Permite também o controle do acesso aos conteúdos por usuário, por origem ou mesmo por horário.

[6] http://www.isc.org/software/dhcp

[7] http://www.netfilter.org/

[8] http://www.squid-cache.org/

[9] http://squidguard.org/

[10] Xen é uma camada de software que permite a criação de máquinas virtuais, tornando possível a execução de vários sistemas operacionais concorrentemente em uma mesma máquina. Maiores informações em http://xen.org/

[11] servidor de páginas *web* com suporte a módulos que permitem a criação de páginas dinâmicas, autenticação etc. Maiores informações em http://apache.org/

escritas em *PHP*[12]. O servidor de banco de dados possui instalado os servidores *MySQL* e *PostgreSQL*[13] para servir os *sites* hospedados no servidor *web*.

A UFVJM já possuía um sistema para cadastro de estações de trabalho (desenvolvido pelo autor) com o objetivo de identificar quais estações de trabalho pertencem a cada setor da instituição e, caso haja, identificar o usuário responsável por aquela estação de trabalho. Visando o reaproveitamento de dados de pessoas e setores da instituição, o sistema de cadastro de estações de trabalho foi integrado ao sistema de informação utilizado pela UFVJM para controle acadêmico e administrativo. A modelagem do sistema de cadastro de estações de trabalho, incluindo os dados que tal sistema utiliza do sistema de controle acadêmico e administrativo, se encontram na Figura 3.1.

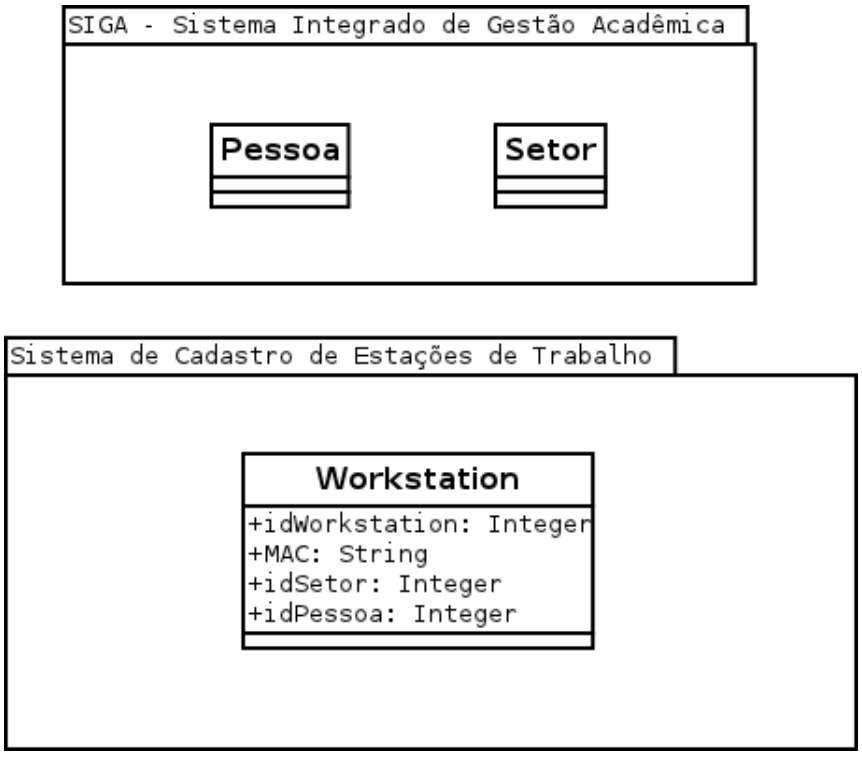

Figura 3.1: Modelagem do sistema de cadastro de estações de trabalho

[12]Linguagem de programação que permite a criação de páginas dinâmicas, acesso a bancos de dados etc. Maiores informações em http://php.net/

[13]Sistemas gerenciadores de bancos de dados relacionais, sendo que o *PostgreSQL* possibilita também a utilização do modelo Objeto-Relacional. Maiores informações em http://mysql.com/ e http://www.postgresql.org/

3.2 Implementação

Para a detecção dos programas maliciosos foi utilizada a abordagem de análise de tráfego. Essa escolha foi embasada nas características e vantagens apresentadas na Seção 2.3. Desse modo, foi utilizado um *IDS* para efetuar a análise do tráfego e detecção de *malwares*. Além disso, os servidores de banco de dados, *web* e o *gateway* da instituição foram aproveitados para a implementação das partes necessárias para o funcionamento do sistema de detecção e bloqueio de máquinas infectadas.

3.2.1 Detecção

O *IDS* escolhido para a implementação de detecção de *malwares* por análise de tráfego foi o *Snort*. O *Snort* é um *IDS* livre e bastante flexível quanto as formas de detecção. Além da sua eficiente detecção por assinatura, onde são analisados os conteúdos dos pacotes para verificar anomalias ou exploração de vulnerabilidades de determinados serviços, o *Snort* possui também um sistema de preprocessadores que permitem uma análise mais detalhada do tráfego, tornando possível a implementação de outros tipos de análise e detecção (KOZIOL, 2003).

O *Snort* também possui um mecanismo de resposta à tentativa de intrusão, podendo ser considerado um sistema de prevenção de intrusão (*IPS*). Entretanto, esse mecanismo é limitado ao bloqueio da conexão ou ao envio de pacotes para a origem e o destino do tráfego anômalo detectado. Como o objetivo do sistema é, além de bloquear o usuário, informá-lo sobre a situação e ainda dar a possibilidade de que ele preencha um cadastro da localização da máquina infectada, o mecanismo de resposta do *Snort* não pode ser utilizado.

A escolha desse sistema de detecção de intrusão se deu, além da experiência do autor com essa ferramente, também pela flexibilidade do seu sistema de detecção baseado em regras e também pela possibilidade de futuras implementações e testes de preprocessadores de tráfego com o objetivo de realizar análises estatísticas e classificação e agregação de tráfego para detecção de *malwares*, como proposto por Taylor e Alves-Foss (2001), Gu *et al.* (2007) e Gil *et al.* (2009). Cabe ressaltar que as soluções de detecção citadas anteriormente não foram utilizadas pelo presente projeto por não serem disponibilizadas ou serem disponibilizadas com restrições de licença, não atendendo ao objetivo do projeto de construir uma ferramenta livre para detecção e bloqueio de máquinas infectadas por *malwares*.

Em sua configuração padrão, no Debian, os alertas do *Snort* são registrados em um arquivo de *log*. Desse modo, foi desenvolvido um *script* para analisar os *logs* do *Snort* em busca de registros de tráfego gerado por *malwares* (Figura 3.2). Esse *script* é executado via *cron*[14] a cada minuto buscando máquinas infectadas detectadas no minuto anterior à sua execução. Ao encontrar algum registro, o *script* atualiza o banco de dados do sistema de cadastro de estações de trabalho com a data e hora do tráfego mais recentemente detectado.

Para armazenar o horário em que o tráfego de uma determinada máquina infectada por *malware* foi identificado pelo *Snort*, o sistema de cadastro de estações de trabalho foi estendido. Além do tráfego, visando facilitar a localização das máquinas, foram também adicionados campos no cadastro de estações de trabalho para conter a localização física dessas máquinas. As alterações no modelo de dados estão representadas na Figura 3.3.

3.2.2 Gerenciamento via *Web*

Um sistema *web*[15] foi desenvolvido para facilitar a visualização das máquinas que foram detectadas como infectadas e também para informar ao usuário da máquina sobre a infecção. O sistema ainda permite que o usuário da máquina infectada informe ou corrija a localização física da máquina para facilitar o acesso à Seção de Suporte e Apoio Técnico da UFVJM e solucionar o problema da infecção de forma mais rápida e eficiente.

A linguagem escolhida para o desenvolvimento desse sistema foi o *PHP*, visto que a infraestrutura dos servidores da universidade já possuia todos os recursos instalados e configurados para a execução de sistemas *web* desenvolvidos nessa linguagem.

Nessa interface, apenas são listadas as máquinas cujo tráfego identificado como *malware* ocorreu nos últimos 15 minutos. A Figura 3.4 mostra a tela em que são visualizadas as máquinas infectadas. Caso já estejam cadastradas as informações de setor, sala e pessoa responsável pela estação de trabalho, a Seção de Suporte e Apoio Técnico da UFVJM pode agir proativamente no sentido de isolar e desinfectar essas estações de trabalho.

[14] Agendador de tarefas que executa determinados programas em determinados intervalos de tempo (por exemplo, a cada minuto, dia, semana ou mês). Disponível em http://ftp.isc.org/isc/cron/

[15] O *design* gráfico do sistema foi desenvolvido por Paulo Henrique Mota Andrade, acadêmico e estagiário da UFVJM.

```
#!/bin/bash
. /etc/antimalware-tools/antimalware-tools.conf
PATH=$PATH:/bin:/usr/bin
# Busca nos alertas do Snort os IPs de origem
ips=$(get_virus_traffic_source)
for ip in $ips; do
    echo "Found $ip"
    # Para cada IP encontrado, busque o MAC
    mac=$(get_mac $ip)
    if [ ! -z "$mac" ]; then
    # Verifica, pelo MAC, se a maquina ja foi cadastrado
    # no banco de dados
        ws=$(get_workstation $mac)
        if [ -z "$ws" ]; then
        # Caso negativo, cadastre a maquina no banco de
        # dados usando o MAC
            echo 'inserindo...'
            insert_workstation $mac
            ws=$(get_workstation $mac)
        fi
    # Atualiza a data do registro de trafego de malware
    # da maquina no banco de dados
        up=$(update_traffic $ws $ip | grep '0');
        if [ ! -z "$up" ]; then
        # Ou, caso nao exista trafego cadastrado,
        # insere um novo registro de trafego
            insert_traffic $ws $ip > /dev/null;
        fi
    fi;
done
```

Figura 3.2: Script para informar possíveis tráfegos gerados por *malware*

Como as máquinas infectadas são bloqueadas automaticamente, o tempo de 15 minutos para o desbloqueio foi definido na tentativa de minimizar problemas com falsos positivos e também e também a insatisfação do usuário da máquina bloqueada. Esse tempo foi definido como um parâmetro de configuração do sistema, podendo ser alterado de acordo com as necessidades.

Foram desenvolvidas também páginas com o objetivo de informar ao usuário que sua estação de trabalho foi infectada por programas maliciosos e também de atualizar os dados de localização das máquinas, instruindo o responsável pela máquina a preenchê-los caso o sistema não os conheça (Figura 3.5). Após o cadastro dos dados referentes à localização da máquina (setor e sala em que se encontra), o sistema redireciona o usuário para uma outra página que o informa que a Seção de

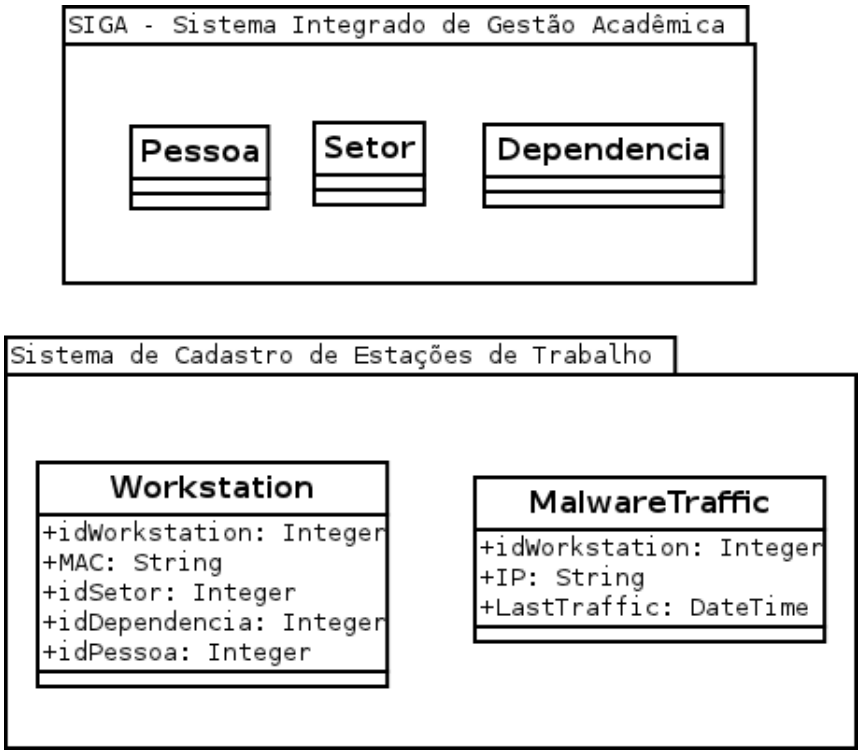

Figura 3.3: Modelagem do sistema de cadastro de estações de trabalho

Suporte e Apoio Técnico já está ciente do problema e que o usuário deve aguardar atendimento ou entrar em contato por telefone com a Seção de Suporte e Apoio Técnico para maiores esclarecimentos (Figura 3.6).

3.2.3 Bloqueio

Para efetuar o bloqueio o sistema configura um redirecionamento no *squidGuard* utilizando um filtro a partir dos *IPs* de origem que foram detectados pelo *Snort* (Figura 3.7) e também uma regra de acesso (*ACL*) que redireciona qualquer acesso *HTTP* vindo dos *IPs* infectados para o sistema desenvolvido pelo autor (Figura 3.8). Tanto o filtro quando a regra de acesso gerados pelo sistema *web* são substituídos em um arquivo de configuração previamente criado pelo autor, conforme Figura 3.9.

Figura 3.4: Tela do sistema que mostra quais máquinas estão infectadas com *malware*

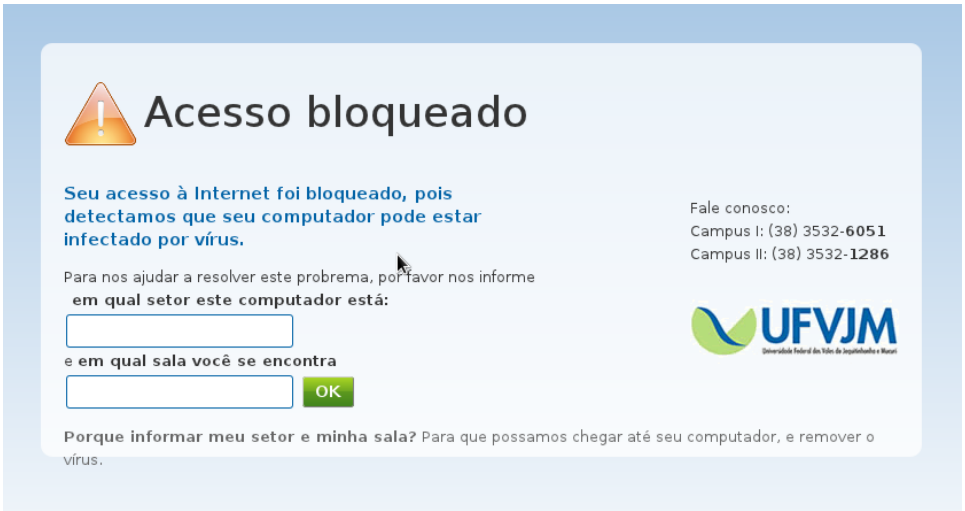

Figura 3.5: Tela do sistema que informa ao usuário sobre a infecção

Além do redirecionamento do *squidGuard*, o sistema também cria um arquivo de regras para o *iptables* (Figura 3.11) impedindo que qualquer tráfego vindo daquela máquina seja repassado para a *Internet*, com exceção do servidor *web* da UFVJM, onde o sistema que informa sobre a infecção está hospedado.

Figura 3.6: Tela do sistema que o usuário visualiza quando os dados de localização da máquina já foram preenchidos

```
src virus {
  ip 10.0.253.239 10.0.234.248 10.0.251.177 10.0.243.49
  log virus-redirect
}
```

Figura 3.7: Exemplo de bloqueio por origem retornado pelo sistema *web*

```
virus {
  pass void none
  redirect http://void.ufvjm.edu.br/antivirus/index.php
}
```

Figura 3.8: Exemplo de *ACL* retornada pelo sistema *web*

Além da geração do filtro de *IPs* e da *ACL* pelo sistema *web*, foi criado também um *script* que busca essas configurações nele e as aplica ao *squidGuard* e ao *iptables* (Figura 3.10). Esse *script* é executado no *gateway* da instituição via *cron* a cada minuto, visando um bloqueio imediato das máquinas infectadas.

```
dbhome /var/lib/squidguard/db
logdir /var/log/squid3

rew ubunturep {
  s@br.archive.ubuntu.com/ubuntu@repo.ufvjm.edu.br/ubuntu@i
  s@security.ubuntu.com/ubuntu@repo.ufvjm.edu.br/ubuntu-sec@i
}

# @@@VIRUS SRC@@@

dest void {
  domainlist void
}

acl {
  default {
    pass all
    rewrite  ubunturep
  }

# @@@VIRUS ACL@@@

}
```

Figura 3.9: *Template* para o arquivo de configuração do *squidguard*

3.3 Metodologia de experimentos

Para a detecção do tráfego gerado pelos programas maliciosos foi utilizada uma máquina com sistema operacional Debian GNU/Linux 6 (*Squeeze*). Como a rede da UFVJM ainda é basicamente composta por *switches* que não possuem o recurso de *port mirroring*, foi escolhida a configuração de uma *bridge* para que a análise do tráfego fosse possível. Foram instaladas duas interfaces de rede na máquina em que o sistema de detecção seria configurado. Foi criada uma interface virtual (*br0*) a partir dessas duas interfaces reais, de modo que qualquer pacote que fosse destinado a cada uma das interfaces reais seria reencaminhada à outra interface real. Assim, todo o tráfego que chegasse a qualquer interface de rede poderia ser analizado pela máquina e também seria encaminhada à outra interface, não causando prejuízo na comunicação entre os segmentos de rede que estariam interconectados pela máquina. Essa máquina foi colocada estrategicamente entre o Campus I e o Campus JK para que uma maior parte do tráfego pudesse ser analisada.

```
#!/bin/bash

PATH=/bin:/usr/bin:/usr/sbin

CONF_PATH="/etc/squid";
CONF="$CONF_PATH/squidGuard.conf"

BASEURL="http://void.ufvjm.edu.br/antivirus/index.php";

# Baixando configuracoes do sistema web
src=$(wget -o /dev/null -O - "$BASEURL/squidguard/src")
acl=$(wget -o /dev/null -O - "$BASEURL/squidguard/acl")
virus=$(wget -o /dev/null -O - "$BASEURL/iptables/virus")

# Substituindo configuracoes a partir do template de configuracao
# do squidguard
cat $CONF.tpl | perl -pi~ -e "s|# \@\@\@VIRUS SRC\@\@\@|$src|" \
> $CONF.tmp
cat $CONF.tmp | perl -pi~ -e "s|# \@\@\@VIRUS ACL@\@\@|$acl|" \
> $CONF.tmp2

# Substituindo o arquivo de configuracao do squidguard
mv /etc/squid/squidGuard.conf{.tmp2,}

# Substituindo o arquivo de bloqueio das maquinas via iptables
echo "$virus" > /root/firewall/rules/S07virus

# Aplicando as novas configuracoes
squid -k reconfigure
invoke-rc.d iptables restart
```

Figura 3.10: *Script* que baixa e aplica as configurações geradas pelo sistema *web*

```
# IPs com trafego gerado por malwares
$IPTABLES -I tcp_forward_packets -s 10.0.241.155 \
-d 200.131.252.28 -p tcp --dport 80 -j allowed
$IPTABLES -I tcp_forward_packets -s 10.0.241.155 -p tcp -j DROP
$IPTABLES -I udp_forward_packets -s 10.0.241.155 -p udp -j DROP
```

Figura 3.11: Exemplo de arquivo para o *firewall* retornado pelo sistema *web*

Para a detecção em si, foram utilizadas algumas assinaturas de tráfego que identificam a proliferação ou exploração de falhas de segurança conhecidas e utilizadas por *malwares*. A princípio, foram selecionadas as regras publicadas por

Leder e Werner (2009) com o objetivo de detectar a proliferação do *Conficker*[16] (Figura 3.12).

```
# Alerta de proliferacao - Conficker.A na rede local
# Fonte: http://www.honeynet.org/files/KYE-Conficker.pdf
alert tcp any any -> $HOME_NET 445 (msg: "Possible Malware Traffic
 - Conficker.A shellcode"; content: "|e8 ff ff ff ff c1|^|8d|N|
10 80|1|c4|Af|81|9EPu|f5 ae c6 9d a0|O|85 ea|O|84 c8|O|84 d8|O|
c4|O|9c cc|IrX|c4 c4 c4|,|ed c4 c4 c4 94|&<O8|92|\;|d3|WG|02 c3|,|
dc c4 c4 c4 f7 16 96 96|O|08 a2 03 c5 bc ea 95|\;|b3 c0 96 96 95
 92 96|\;|f3|\;|24|i| 95 92|QO|8f f8|O|88 cf bc c7 0f f7|2I|d0|w|
c7 95 e4|O|d6 c7 17 f7 04 05 04 c3 f6 c6 86|D|fe c4 b1|1|ff 01 b0
 c2 82 ff b5 dc b6 1b|O|95 e0 c7 17 cb|s|d0 b6|O|85 d8 c7 07|O|c0|
T|c7 07 9a 9d 07 a4|fN|b2 e2|Dh|0c b1 b6 a8 a9 ab aa c4|]|e7 99 1d
 ac b0 b0 b4 fe eb eb|"; sid: 2000001; rev: 1;)

# Alerta de proliferacao - Conficker.B na rede local
# Fonte: http://www.honeynet.org/files/KYE-Conficker.pdf
alert tcp any any -> $HOME_NET 445 (msg: "Possible Malware Traffic
 - Conficker.B shellcode"; content: "|e8 ff ff ff ff c2|_|8d|O|
10 80|1|c4|Af|81|9MSu|f5|8|ae c6 9d a0|O|85 ea|O|84 c8|O|84 d8|O|
c4|O|9c cc|Ise|c4 c4 c4|,|ed c4 c4 c4 94|&<O8|92|\;|d3|WG|02 c3|,|
dc c4 c4 c4 f7 16 96 96|O|08 a2 03 c5 bc ea 95|\;|b3 c0 96 96 95
 92 96|\;|f3|\;|24 |i|95 92|QO|8f f8|O|88 cf bc c7 0f f7|2I|d0|w|
c7 95 e4|O|d6 c7 17 cb c4 04 cb|{|04 05 04 c3 f6 c6 86|D|fe c4 b1|
1|ff 01 b0 c2 82 ff b5 dc b6 1f|O|95 e0 c7 17 cb|s|d0 b6|O|85 d8
 c7 07|O|c0|T|c7 07 9a 9d 07 a4|fN|b2 e2|Dh|0c b1 b6 a8 a9 ab aa
 c4|]|e7 99 1d ac b0 b0 b4 fe eb eb|"; sid: 2000002;rev: 1;)
```

Figura 3.12: Regras para detecção do *Conficker*

As assinaturas que identificam exploração de falhas de segurança utilizadas no sistema de detecção e bloqueio de *malwares* foram definidas pelo *Cyber-Threat Analytics* (*Cyber-TA*)[17]. O *Cyber-TA* é um projeto de pesquisa construído atravéz de parcerias entre universidades norteamericanas como a *University of California*, *North Carolina State University*, *Stanford University*, *University of Texas* e *Yale University*. O projeto foi criado com o objetivo de melhorar a capacidade das instituições de se defenderem de programas maliciosos que se espalham pela *Internet*.

[16] *Conficker* é um programa malicioso que infectou milhões de computadores desde o início de suas atividades em 2008. A grande inovação do *Conficker* foi a possibilidade do programa malicioso se atualizar via *Internet*, modificando seu comportamento e dificultando ainda mais a sua detecção e remoção.

[17] Maiores informações em `http://www.cyber-ta.org/`

O projeto *Cyber-TA* possui uma *honeynet* infectada com diversos tipos de *malwares*. O tráfego gerado por essas infecções são analisados a fim de serem construídas assinaturas para o *Snort* com o objetivo de detectar o tráfego dos programas maliciosos naquela rede. Essas assinaturas são então testadas e sua eficiência é medida de acordo com a quantidade de infecções que ela é capaz de detectar dentro da *honeynet* do projeto.

Foi lançado então o *Malware Threat Center*[18], um sistema *web* que atualiza diariamente as informações sobre as assinaturas desenvolvidas pelo projeto *Cyber-TA* com base nas informações disponibilizadas pela *honeynet*. Essas assinaturas estão disponíveis em `http://mtc.sri.com/live_data/signatures/`.

Dentre as assinaturas disponibilizadas pelo *Cyber-TA* atravéz do *Malware Threat Center*, foram utilizadas apenas as cujo nível de detecção de infecção identificado pelo próprio projeto fosse maior do que 50%. Nos testes iniciais, alguns dos alertas que foram gerados, ao serem analisados, foram identificados como falsos positivos. As regras que estavam gerando esses alertas detectavam qualquer tipo de transferência de arquivos executáveis do *Windoes* (arquivos *PE - Portable Executable*) pela rede como sendo uma infecção por *malware*. Por serem genéricas o suficiente, verificando qualquer requisição *HTTP* que resultasse no *download* de um arquivo executável, essas regras foram consideradas pelo autor como desnecessárias e sua remoção fez com que os falsos positivos fossem eliminados.

Vale ressaltar que tanto as assinaturas de detecção de programas maliciosos como também de vulnerabilidades exploradas por tais programas foram adaptadas para serem utilizadas na rede interna da UFVJM, de acordo com a Figura 3.13. Futuramente serão desenvolvidas regras próprias para detectarem pragas específicas na instituição. Entretanto, para que isso seja feito, será necessário identificar tráfegos e analisá-lo em busca de padrões e assinaturas de possíveis *malwares*.

Para os testes iniciais das regras do *Snort* foi desenvolvido um *script* (Figura 3.14) utilizando a biblioteca *scapy*[19], que permite a manipulação de pacotes em *Python*[20] de maneira simples e prática. Um exemplo de teste pode ser acompanhado na Figura 3.15. A primeira figura mostra o *script* enviando um pacote que deve ser identificado como *Conficker* pelo *Snort*. A segunda, mostra a identificação do tráfego no *log* de alertas do *Snort*.

[18] *Malware Threat Center* é um serviço construído utilizando os dados produzidos pelo projeto *Cyber-TA*.

[19] Maiores informações em `http://www.secdev.org/projects/scapy/`

[20] Linguagem de programação e *script* de alto nível, orientada a objetos e que possui diversas bibliotecas disponíveis. Maiores informações eh `http://www.python.org`

```
# Alerta de Exploit NOOP
# Fonte: http://mtc.sri.com/live_data/signatures/ (adaptada)
alert tcp any any -> $HOME_NET [135:139,445,1025] (
msg:"Possible Malware Traffic - E2[rb] SHELLCODE x86 0x90
unicode NOOP";
content:"|90 90 90 90 90 90 90 90 90 90|";
classtype:shellcode-detect;
sid:299913; rev:1;)

# Alerta para exploit MS-DS
# Fonte: http://mtc.sri.com/live_data/signatures/ (adaptada)
alert tcp any any -> $HOME_NET 445 (
msg:"Possible Malware Traffic - E2[rb] NETBIOS SMB-DS IPC$
unicode share access";
flow:established,to_server;
content:"|00|"; depth:1; content:"|FF|SMBu"; within:5;
distance:3; byte_test:1,&,128,6,relative; pcre:"/^.{27}/R";
byte_jump:2,7,little,relative;
content:"I|00|P|00|C|00 24 00 00 00|"; distance:2; nocase;
flowbits:set,smb.tree.connect.ipc;
classtype:protocol-command-decode; sid:22466;
rev:7;)

# Alerta para exploit LSA
# Fonte: http://mtc.sri.com/live_data/signatures/ (adaptada)
alert tcp any any -> $HOME_NET 445 (
msg: "Possible Malware Traffic - E2[rb] BotHunter EXPLOIT
 LSA exploit";
content:"|3131313131313131313131313131313131313131313131|";
classtype: misc-activity;
sid:292000032; rev:99; )
```

Figura 3.13: Regras para detecção de *exploits* de vulnerabilidade do *Windows*

Após os testes iniciais feitos pelo *script*, o servidor foi deixado identificando o tráfego e atualizando o banco de dados com o horário do último tráfego detectado (utilizando o *Script* 3.2). O *gateway* também foi configurado para efetuar o bloqueio das máquinas infectadas (utilizando o *Script* 3.10).

```
#! /usr/bin/env python

# Set log level to benefit from Scapy warnings
import logging
logging.getLogger("scapy").setLevel(1)

from scapy.all import *

def test_send (dstip, p):
    sr(IP(dst=dstip)/TCP(dport=445)/p)

def test_conficker(dstip):
    p = 'e8ffffffffc25f8d4f108031c4416681394d5375f538aec69da04f' +
    '85ea4f84c84f84d84fc44f9ccc497365c4c4c42cedc4c4c494263c4f38' +
    '923bd3574702c32cdcc4c4c4f71696964f08a203c5bcea953bb3c09696' +
    '9592963bf33b24699592514f8ff84f88cfbcc70ff73249d077c795e44f' +
    'd6c717cbc404cb7b040504c3f6c68644fec4b131ff01b0c282ffb5dcb6' +
    '1f4f95e0c717cb73d0b64f85d8c7074fc054c7079a9d07a4664eb2e244' +
    '680cb1b6a8a9abaac45de7991dacb0b0b4feebeb'.decode ('hex')
    test_send (dstip, p)

def test_noop_exploit (dstip):
    p = '909090909090909090'.decode ('hex')
    test_send (dstip, p)

def test_lsa_exploit (dstip):
    p = '313131313131313131313131313131313131313131313131313131313131313131313131313131' +
    '131313131313131313131313131313131313131313131313131313131313131313131313131313131' +
    '313131313131313131313131313131313131313131313131313131313131313131313131313131313131' +
    '1313131313131313131313131313131' +
    '3131313131313131313131313131'.decode ('hex');
    test_send (dstip, p)

if __name__ == "__main__":
    interact(mydict=globals(), mybanner="Snort Rule Tester v0.1")
```

Figura 3.14: *Script* para testar regras do *Snort*

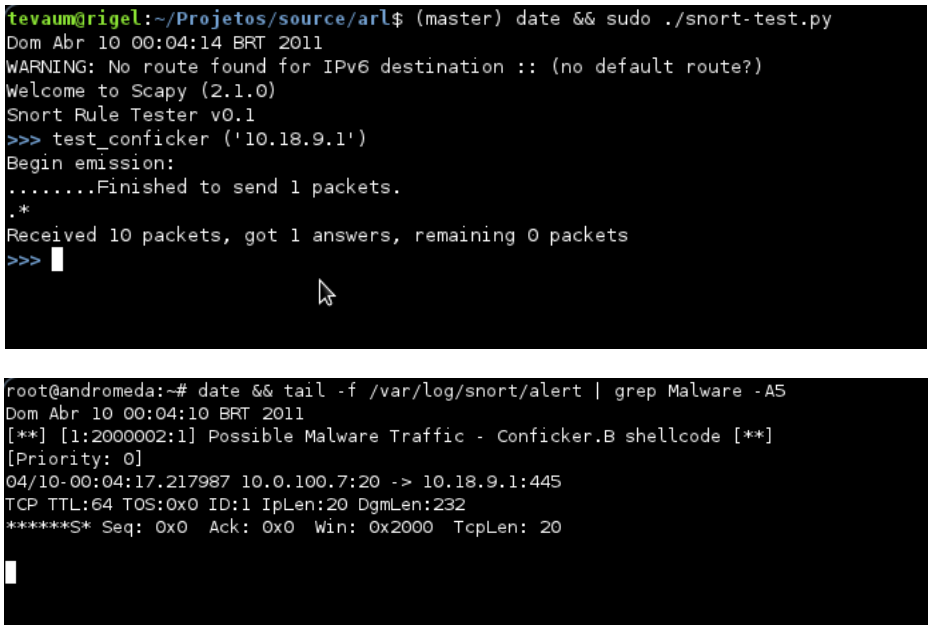

Figura 3.15: Envio do pacote de testes e detecção do tráfego pelo *Snort*

Capítulo 4

Resultados e Discussão

Até a execução do presente trabalho, a UFVJM não possuía nenhuma solução corporativa para detecção de programas maliciosos e era necessário ser alertada por órgãos externos sobre os problemas relacionados à infecção por *malwares* na instituição. Dessa forma, o desenvolvimento e implantação do sistema de detecção e bloqueio de máquinas infectadas por *malwares* foi considerado um grande avanço pela Diretoria de Tecnologia da Informação da UFVJM. Como consequência da implantação desse sistema vieram vários benefícios, diretos e indiretos, tanto de ordem técnica quanto administrativa.

Foram consideradas benefícios indiretos trazidos pelo sistema as implementações feitas a partir das atualização das bases de dados de estações de trabalho por setor e identificação dos responsáveis pelas máquinas. Com os dados atualizados, foi possível definir faixas de endereçamento IP por setor, o que permitiu novas possibilidades e implementações como a medição do tráfego por setor, garantia de qualidade de serviço através de controle de banda e priorização de tráfego, restrição de acesso a determinados conteúdos na *web* também por setor, dentre outras.

Como benefícios diretos são consideradas, principalmente, as informações geradas pelo sistema de detecção e bloqueio de *malwares*, dando à instituição autonomia no planejamento de ações para eliminar as infecções. A conscientização da equipe de Suporte e Apoio Técnico com relação às atualizações de segurança do *Windows* é um exemplo dessas ações. Anteriormente à experiência de detecção e bloqueio das máquinas infectadas por programas maliciosos essas atualizações não eram instaladas. Também estão sendo planejadas campanhas buscando conscientizar os usuários e treiná-los a identificar possíveis fraudes, com o objetivo de reduzir as possibilidades de novas infecções por programas maliciosos.

Vale ressaltar que a detecção e bloqueio das estações de trabalho infectadas juntamente com a informação de sua localização auxiliou na identificação de máquinas com falhas de segurança e sistemas operacionais desatualizados. A instalação das atualizações do sistema operacional nas máquinas detectadas fez com que a proliferação dos programas malicioso na rede diminuisse, consequentemente reduzindo de forma gradativa a detecção de máquinas infectadas, conforme o grárico apresentado na Figura 4.1. No gráfico, o valor do eixo *infecções* indica a quantidade de máquinas detectadas durante todo o mês, contando somente a primeira detecção de cada máquina naquele mês.

É possível notar pela Figura 4.1 que no primeiro mês a redução das infecções foi menor que nos meses seguintes. Isso se dá pelo fato de que, no início, a equipe responsável pela desinfecção das máquinas estava somente executando o antivírus, não estavam instalando as atualizações de segurança necessárias para a correção das vulnerabilidades utilizadas pelos *malwares*. Desse modo, ao ser reiniciada, a máquina era novamente infectada. A equipe foi instruída a proceder com a atualização do sistema operacional das máquinas na metade da terceira semana do mês de novembro. A redução geral das infecções somente não foi mais efetiva pelo fato de que a equipe disponibilizada para trabalhar na remoção dos *malwares* era pequena (apenas dois funcionário por campus). Houve uma redução nas infecções de aproximadamente 83% nos cinco meses em que aplicação foi utilizada. Vale ressaltar ainda que a equipe responsável pela desinfecção das máquinas não era exclusiva e tinha que realizar outros trabalhos durante esse período.

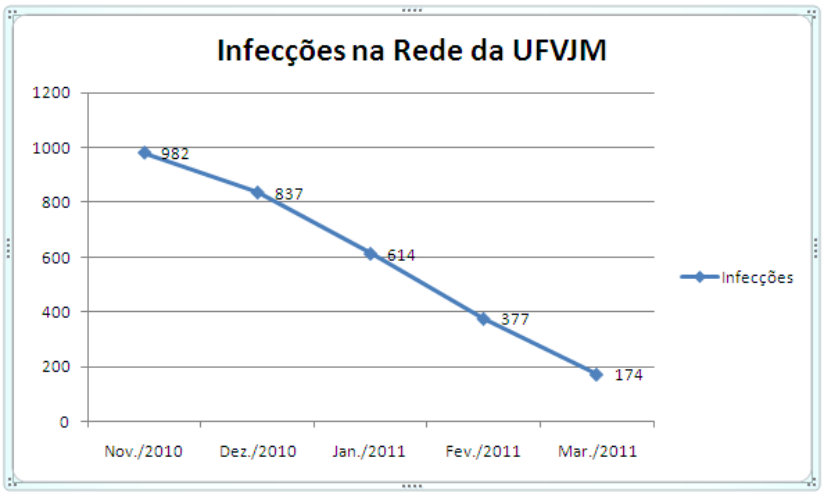

Figura 4.1: Gráfico da redução de infecção

O tráfego gerado pelas máquinas infectadas também foi medido. O sétimo dia útil de cada mês foi utilizado como referência para as medições. Nesse dia, deixou-se executando, na máquina utilizada para a detecção dos *malwares*, uma captura de todos os pacotes trafegados nas portas 135, 139 e 445 (portas utilizadas pelo protocolo SMB e que alguns *malwares* também utilizam para se proliferarem). A captura foi iniciada às 06:00 e finalizada às 18:00, considerando as seis primeiras horas como turno matutino e as outras, turno vespertino. As reduções tanto do tráfego quanto da quantidade de pacotes gerados durante os meses de execução do sistema de detecção e bloqueio de máquinas infectadas por *malwares* pode ser acompanhado nas Figuras 4.2 e 4.3.

Pode-se perceber que desde a primeira medição houve uma redução de 85% no tráfego medido. Vale relembrar que a rede da UFVJM não possui segmentação, porém é composta por *switches*, impeddindo que determinados tráfegos sejam medidos. O tráfego medido foi somente o de troca de pacotes entre os *campi* de Diamantina, sendo que o tráfego inter-campus (tentativa de infecção de máquinas do Campus JK para máquinas do próprio Campus JK por exemplo) não entraram nessa medição. Desse modo, a redução das infecções e do tráfego gerado pelas máquinas infectadas podem ser consideradas maiores do que a redução apontada pelos gráficos das Figuras 4.1, 4.2 e 4.3.

Houveram também problemas e limitações enfrentados pela equipe da DTI durante a execução das atividades de detecção e correção das máquinas identificadas e bloqueadas pelo sistema. Algumas das máquinas detectadas não faziam parte do patrimônio da instituição, de modo que, teoricamente, a Seção de Suporte e Apoio ao Usuário não poderia atuar na correção dessas máquinas, mesmo tendo consciência de que elas poderiam vir a infectar novamente as máquinas da instituição. A falta de segmentação da rede permite que máquinas pessoais de alunos ou professores infectem as máquinas da instituição. Para resolver esse problema, a Diretoria de Tecnologia da Informação já iniciou a substituição de alguns ativos da rede por novos ativos gerenciáveis com suporte a *vlans*. Após a substituição, a segmentação será configurada, visando limitar a proliferação dos *malwares*, aumentando a segurança na rede da instituição.

Outro problema enfrentado foi com relação ao preenchimento das informações de localização física das máquinas. Por medo de serem punidos, alguns usuários não preencheram as informações de localização. Essa atitude dificultou a ação da Seção de Suporte e Apoio Técnico para eliminar as infecções das máquinas da instituição. Algumas dessas máquinas puderam ser identificadas com o auxílio dos novos ativos gerenciáveis adquiridos pela instituição, porém a atitude mais efetiva na solução desse problema foi enviar um comunicado oficial via e-mail

informando que as informações requisitadas pelo sistema de detecção e bloqueio de *malwares* somente seriam utilizadas internamente à Seção de Suporte e Apoio Técnico para a localização, desinfecção e liberação do acesso à máquina e não implicariam na punição do usuário responsável pela máquina.

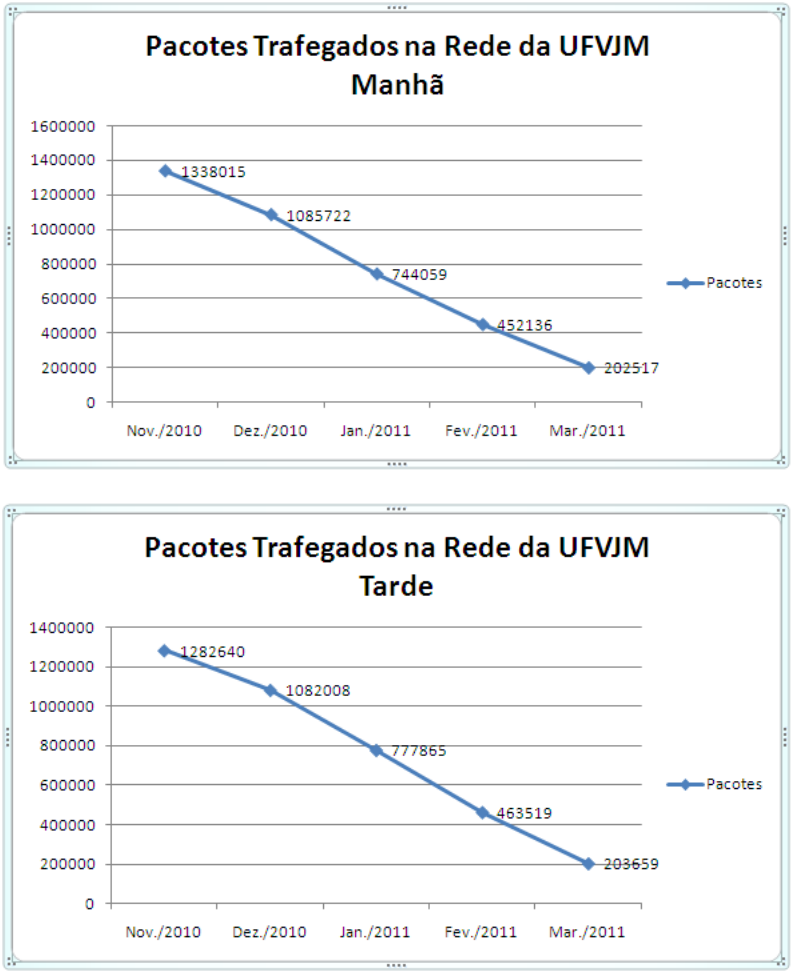

Figura 4.2: Redução do tráfego (pacotes) na rede da UFVJM

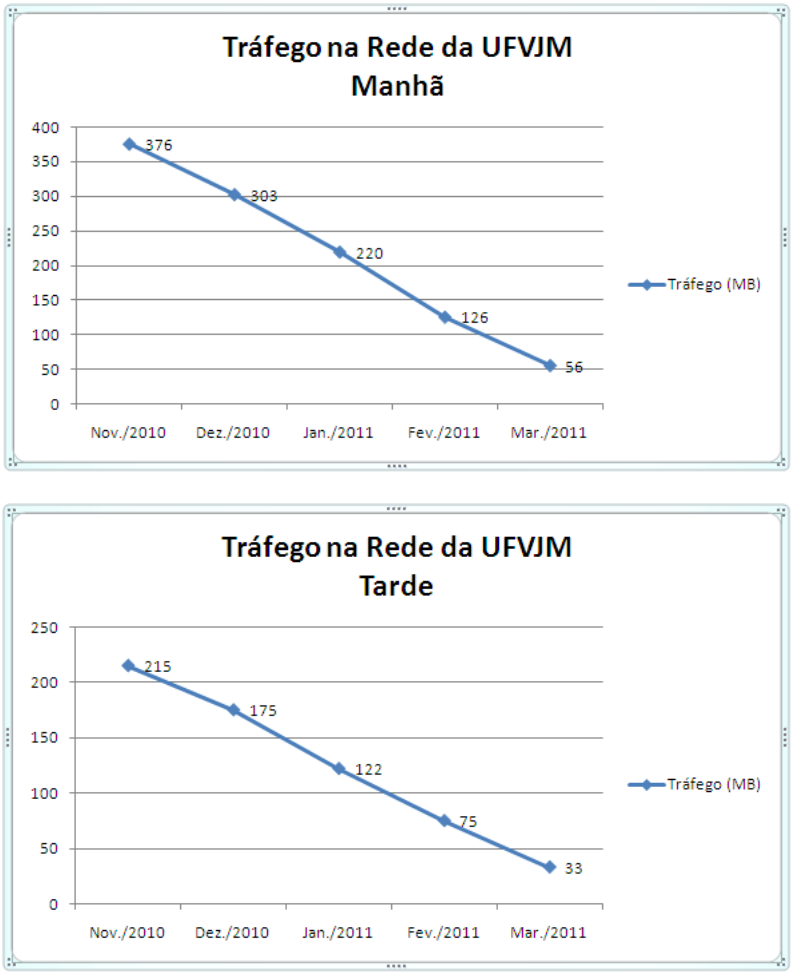

Figura 4.3: Redução do tráfego (MB) na rede da UFVJM

Capítulo 5

Conclusão

5.1 Considerações Finais

Este trabalho evidonciou a importância de manter os sistemas operacionais atualizados, visto que falhas de segurança são utilizadas não somente por programas maliciosos, mas também por pessoas mal-intencionadas e podem, inclusive, trazer problemas graves de segurança da informação e no funcionamento de uma rede corporativa. Torna-se necessária a conscientização das equipes técnicas responsáveis pela manutenção das máquinas para mantê-las atualizadas e evitar esses problemas. A não atualização das máquinas deixa brechas para programas maliciosos e o tráfego gerado por máquinas infectadas podem atrapalhar o bom funcionamento da rede local, ocasionando perda de pacotes ou mesmo sobrecarga em alguns segmentos da rede que possuem conexão de baixa velocidade.

Foi notado também que a falta de um sistema de detecção de programas maliciosos em uma rede corporativa pode levar a problemas sérios com relação aos dados da instituição. Como visto na Figura 4.1, no início da implantação do sistema foram detectadas 982 máquinas infectadas. Se considerarmos a possibilidade de que essas máquinas parassem de funcionar ou tivessem seus dados perdidos por causa dessa infecção, as consequências para a instituição teriam sido catastróficas. Atrelando uma grande quantidade de máquinas infectadas em uma rede corporativa com a falta de atualização das máquinas e de um sistema de detecção de *malwares* a fim de que as infecções sejam tratadas leva a um ambiente completamente inseguro e instável.

Os resultados evidenciados pela solução proposta mostraram a importância de manter as estações de trabalho de uma rede corporativa atualizadas. A experiência adquirida no desenvolvimento deste trabalho torna clara também a importância da utilização de um sistema de detecção de programas maliciosos em redes corporativas. Entretanto para que esse sistema funcione de modo aceitável, é necessário que as assinaturas de detecção dos *malwares* estejam sempre atualizadas, em vista do rápido aparecimento de novas pragas. É importante também que a rede corporativa esteja segmentada, organizada em subredes e *vlans*, limitando os domínios de *broadcast* com o objetivo de impedir que uma determinada infecção possa atingir toda a rede. Tal medida auxilia ainda mais na contenção da infecção.

Finalmente, conclui-se que estre trabalhou cumpriu com seu objetivo de construir um sistema para identificação de bloqueio de máquinas infectadas por *malwares*. O sistema foi capaz de identificar corretamente as infecções e de ser proativo no sentido de impedir que o tráfego originado pela máquina alcançasse a *Internet* e também de munir o usuário com a possibilidade de cadastrar sua localização para que a equipe técnica responsável pudesse solucionar o problema. Vale ressaltar que para conter infecções de *malwares* é necessário buscar atualizar-se sempre, pois a cada dia novas falhas de segurança são descobertas e novas pragas surgem explorando essas vulnerabilidades.

5.2 Trabalhos Futuros

Um problema local que precisará ser resolvido futuramente é a mudança dos setores administrativos da UFVJM do Campus I para o Campus JK. Isso diminuirá bastante o tráfego gerado entre os *campi* de Diamantina e fará com que o sistema de detecção retrate menos fielmente a realidade. A solução para esse problema dependerá da instalação dos *switches* que serão adquiridos. Já foi verificado o suporte a *port mirroring* de modo que será possível o espelhamento do tráfego do *switch* para uma determinada porta onde o sistema de detecção será conectado.

Atualmente, a detecção dos alertas é feita via *script* executando de minuto em minuto no *cron*. Seria interessante uma abordagem que gerasse menos processamento por parte do sistema e que, ainda assim, informasse dos alertas em tempo real para as outras máquinas envolvidas no funcionamento do sistema (servidor de banco de dados e *gateway*).

Dentre as possíveis técnicas pesquisadas para alertar essas outras máquinas sobre os problemas identificados, as consideradas mais interessantes foram a utili-

zação de um *plugin* de saída que permite o envio de alertas via *SNMP*. Utilizando esse plugin, a atualização do banco de dados de máquinas infectadas seria menos custosa não seria necessário ter o *Script* 3.2 executando a cada minuto para coletar os dados.

Através dessa abordagem seria possível também, além de informar ao servidor de banco de dados sobre a detecção do tráfego, informar também o *gateway* para fazer o bloqueio da máquina que gerou o tráfego, eliminando o *overhead* de processamento trazido pela execução dos scripts a cada minuto no *cron*.

Um fator extremamente importante que deve ser analisado antes de partir para a implementação desse envio de alerta em tempo real para o servidor de banco de dados e para o *gateway* é a questão da segurança. Como o sistema trata de bloqueio de máquinas, deve-se ter cuidado para não tornar possível o bloqueio de máquinas indevidas por usuários mal intencionados. Para evitar esse tipo de falha, os servidores de banco de dados e *gateway* precisam garantir que a origem da informação dos alertas seja confiável. Uma forma de se alcançar esse objetivo é utilizando autenticação via certificado, porém isso aumentaria a complexidade do sistema.

Um outro ponto importante a ser levado em consideração é a implementação de exceções no mecanismo de bloqueio. Essa implementação é útil em caso de máquinas de hospitais ou indústrias que não podem perder o acesso à rede por controlarem instrumentos de precisão. Nesses casos acidentes graves podem ser causados caso a máquina perca acesso à rede.

Por fim, um outro trabalho interessante seria a comparação da técnica utilizada na detecção de *malwares* do presente trabalho com outras técnicas estudadas por Taylor e Alves-Foss (2001), Gu *et al.* (2007) e Gil *et al.* (2009), que utilizam análise estatística do tráfego para detecção de anomalias.

Referências Bibliográficas

BACHER, P.; HOLZ, T.; KOTTER, M.; WICHERSK, G. *Know your Enemy*: Tracking botnets. The Honeynet Project, 2005. Disponível em: <http://www.honeynet.org/papers/bots>.

CHEN, T.; ROBERT, J.-M. The evolution of viruses and worms. *Statistical Methods in Computer Security*, 2004. Disponível em: <http://vx.netlux.org/lib-/atc01.html>.

COHEN, F. Computer viruses: Theory and experiments. *Computers & Security*, v. 6, p. 22–35, 1984. Disponível em: <http://vx.netlux.org/lib/afc01.html>.

COOKE, E.; JAHANIAN, F.; MCPHERSON, D. The zombie roundup: Understanding, detecting, and disrupting botnets. 2005.

GIL, C.; GÓMEZ, J.; PADILLA, N.; BAÑOS, R.; JIMÉNEZ, C. Design of a snort-based hybrid intrusion detection system. In: *Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part II: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*. Berlin, Heidelberg: Springer-Verlag, 2009. (IWANN '09), p. 515–522. ISBN 978-3-642-02480-1. Disponível em: <http://dx.doi.org/10.1007/978-3-642-02481-8_75>.

GU, G.; PORRAS, P.; YEGNESWARAN, V.; FONG, M.; LEE, W. Bothunter: detecting malware infection through ids-driven dialog correlation. In: *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2007. p. 12:1–12:16. ISBN 111-333-5555-77-9. Disponível em: <http://portal.acm.org/citation.cfm?id=1362903.1362915>.

KOLBITSCH, C.; COMPARETTI, P. M.; KRUEGEL, C.; KIRDA, E.; ZHOU, X.; WANG, X. Effective and efficient malware detection at the end host. In: *Proceedings of the 18th conference on USENIX security symposium*. Berkeley,

CA, USA: USENIX Association, 2009. (SSYM'09), p. 351–366. Disponível em: <http://portal.acm.org/citation.cfm?id=1855768.1855790>.

KOZIOL, J. *Intrusion detection with Snort*. 1. ed. [S.l.]: Sams, 2003.

LEDER, F.; WERNER, T. *Know your Enemy*: Containing conficker. The Honeynet Project, abr. 2009. Disponível em: <http://honeynet.org/files/KYE-Conficker.pdf>.

POMPON, R. A. Successes and failures apprehending malware authors. In: VIRUS BULLETIN CONFERENCE (VB2010). [S.l.]: Virus Bulletin, 2010.

REYNOLDS, J. *Helminthiasis of the Internet*. IETF, dez. 1989. RFC 1135 (Informational). (Request for Comments, 1135). Disponível em: <http://www.ietf.org/rfc/rfc1135.txt>.

SHOCH, J. F.; HUPP, J. A. The "worm"programs: early experience with a distributed computation. *Commun. ACM*, ACM, New York, NY, USA, v. 25, p. 172–180, March 1982. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/358453.358455>.

TAYLOR, C.; ALVES-FOSS, J. Nate: Network analysis of anomalous traffic events, a low-cost approach. In: *Proceedings of the 2001 workshop on New security paradigms*. New York, NY, USA: ACM, 2001. (NSPW '01), p. 89–96. ISBN 1-58113-457-6. Disponível em: <http://doi.acm.org/10.1145/508171-.508186>.

WHITTAKER, C. *The impact of malware on UK financial intitutions*. 2007. Disponível em: <http://www.oecd.org/dataoecd/33/53/38652807.pdf>.