UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

SORAYA SYBELE HOSSAIN

Detecção de aplicações envio de SPAM através da mineração de tráfego DNS.

Trabalho de Conclusão apresentado como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Prof. Dr. Raul Fernando Weber
Orientador

Porto Alegre, dezembro de 2010

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Hossain, Soraya Sybele

Detecção de aplicações envio de SPAM através da mineração de tráfego DNS. / Soraya Sybele Hossain. – Porto Alegre: Graduação em Ciência da Computação da UFRGS, 2010.

56 f.: il.

Trabalho de Conclusão (bacharelado) – Universidade Federal do Rio Grande do Sul. Curso de Bacharelado em Ciência da Computação, Porto Alegre, BR–RS, 2010. Orientador: Raul Fernando Weber.

1. UFRGS. 2. Segurança. 3. DNS. 4. SPAM. 5. IDS. I. Weber, Raul Fernando. II. Título.

”Rule #1: Spammers lie.
”Rule #2: If a spammer seems to be telling the truth, see Rule #1.”
— THE RULES OF SPAM

AGRADECIMENTOS

Primeiramente, agradeço à minha mãe, Eyde Hossain, por ter me incentivado durante a realização deste trabalho e durante toda a minha vida acadêmica. Por estar sempre presente, por sempre me oferecer auxílio quando precisei e por ser meu exemplo de coragem e determinação na vida. Minha mãe, minha amiga, minha ídola. Muito obrigada por ser a pessoa mais importante em minha vida.

Agradeço ao meu pai, Mohammed Akhtar Hossain, que indiretamente foi o responsável por este momento, ao me permitir o acesso à um 386 a 20 anos atrás, e causando a minha paixão por aquela estranha máquina que me garantiu uma infância tremendamente divertida. Infelizmente não pode acompanhar toda a minha trajetória acadêmica, pois a vida nos toma pessoas importantes cedo demais, nos fazendo sentir saudades, e um imenso desejo de que pudessem compartilhar a alegria deste momento conosco.

Agradeço à minha avó, Albertina Augênia Rigotti, por ter sido mais que uma avó, ter sido uma mãe, uma amiga. Por sempre ter me feito rir quando eu estava estressada com o excesso de atividades da universidade, apesar de ter me feito chorar ao me deixar no início desse ano. Também gostaria que estivesses aqui compartilhando esse momento.

Agradeço aos meus amigos, por me fazerem sorrir todo dia ao me lembrar da existência de vocês. Por estarem ao meu lado, por me ajudarem a superar um dos anos mais difíceis de minha vida, por me fazerem seguir em frente.

Dentre esse grupo de pessoas tão especial, devo meus agradecimentos especiais à três pessoas que tiveram um papel fundamental na realização deste trabalho: Ewerton Miglioranza, Joseane Barrios Coelho e Liesbet Olaerts. Muito obrigada pelo apoio técnico e psicológico. Obrigada por aguentarem meus momentos de nervosismo e insegurança e sempre acreditarem em mim. Eu não teria conseguido sem vocês.

E a duas muito especiais, que tiveram um papel fundamental em toda a minha vida acadêmica: Carlos Eduardo Ramisch e Jean Felipe Patikowski Cheiran. Sem vocês eu não teria chegado até aqui.

Agradeço muito o meu orientador Raul Weber, por ter me aceitado como sua orientanda, por ter acreditado e confiado em mim para escolher e desenvolver o tema deste trabalho, mesmo eu tendo pouquíssimo conhecimento a respeito do assunto no início da realização deste, movida apenas pela curiosidade. Obrigada por todo o apoio, todo o auxílio e toda a compreensão, e peço desculpas por quaisquer excesso de preocupação que eu tenha causado. Obrigada também por ter proporcionado alguns dos momentos mais divertidos da graduação nos laboratórios do prédio 67 com seu skin de Lara Croft.

Agradeço a professora Taisy Weber, pelo apoio que me foi dado no momento que pensei em desistir do trabalho por pensar não estar bom o suficiente.

Agradeço ao pessoal da comunidade FLOSS, por ter me mostrado uma causa em que vale a pena acreditar, e por ter proporcionado todo o ferramental para a realização deste

trabalho.

Agradeço à UFRGS, à todos os mestres, colegas e amigos que tive a oportunidade de conhecer no tempo de realização deste curso. Por todo o conhecimento transmitido, e todas as experiências compartilhadas. Mais do que fornecer um título, vocês foram responsáveis por grande parte do que sou hoje.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

C&C	Command & Control
DDoS	Distributed Denial-of-Service
DNS	Domain Name System
DoS	Denial-of-Service
GSoC	Google Summer of Code
IDS	Intrusion Detection System
IDPS	Intrusion Detection and Prevention System
IPS	Intrusion Prevention System
IRC	Internet Relay Chat
ISP	Internet Service Provider
SMTP	Simple Mail Transfer Protocol
SPF	Sender Policy Framework
VoIP	Voice over Internet Protocol

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Novas ameaças de rede surgem a cada dia, assim como novas idéias para combatê-las. O Sistema de Nomes de Domínio (DNS) é uma infraestrutura crítica no funcionamento da Internet, e por isso é importante monitorar e proteger o seu tráfego de atividades maliciosas, colaborando com a segurança do ciberespaço.

Por ser uma estrutura crucial no funcionamento da rede, também pode ser utilizado na disseminação de *malware*, como o envio de e-mails em massa (*spam*) e o surgimento de redes zumbis (*botnets*). Esse tipo de atividade tem crescido cada vez mais, o que acaba por impulsionar a pesquisa por novas idéias e abordagens para combatê-la.

Estudos recentes tem mostrado que é possível detectar aplicações maliciosas através do comportamento do tráfego DNS emitido por uma máquina infectada. Então por que não utilizar essa metodologia para tentar detectar e combater *spammers*?

Neste trabalho foram analisadas diversas técnicas de detecção de atividades maliciosas através da monitoração do tráfego DNS. Dentre elas, um método de detecção de atividades de máquinas infectadas por vermes de envio de e-mail em massa foi escolhido para ser analisado, implementado e testado no ambiente de rede do Instituto de Informática da UFRGS.

Através dos resultados obtidos, foi verificada a eficácia da técnica, assim como características do ambiente testado.

Palavras-chave: UFRGS, segurança, DNS, SPAM, IDS.

Detection of Mass-Mailing Malware by Mining DNS Traffic Data

ABSTRACT

New network threats appear every day, as well as new ideas to combat them. The Domain Name System (DNS) is a critical infrastructure in the Internet , thus it's important to monitor and protect it's traffic against malicious activities, collaborating with the security of cyberspace.

Because it is a crucial structure in the network operation, it's services can also be used to disseminate malware, such as spamming and helping with the rise of new botnets. This type of activity has grown increasingly, which ultimately drive the search for new ideas and approaches to combat it.

Recent studies have shown that it is possible to detect malicious activities through the behavior of DNS traffic sent by an infected host. So why not use this method to try to detect, and combat, spammers?

In this publication, various techniques for detecting malicious activity by monitoring DNS traffic were studied. Among them, a method for detecting activities of hosts infected by mass-mailing worms was chosen to be analyzed, tested and implemented in the network environment of Instituto de Informática from UFRGS.

Through the results obtained, we verified the effectiveness of this technique, as well as some characteristics of the tested environment.

Keywords: UFRGS, Internet Security, DNS, SPAM, IDS.

1 INTRODUÇÃO

O *Sistema de Domínio de Nomes* (DNS) é um dos sistemas mais críticos da Internet. Não há como negar sua importância, e o seu vasto uso, visto que a utilização da rede sem o sistema de resolução de nomes seria impraticável atualmente. Por essa razão, a monitoração de seu tráfego e a proteção do mesmo contra atividades maliciosas são extremamente importantes no ciberespaço.

Além dos seres humanos, diversas aplicações utilizam a resolução de nomes para os mais diversos fins ao acessar a rede, e ao monitorar o tráfego DNS dessas aplicações, podemos facilmente monitorar suas atividades[WIL 2003], incluíndo atividades de *malwares*, e com isso proteger outros recursos da rede, como servidores de e-mail, por exemplo.

Existem estudos mostrando que é possível detectar a presença de aplicações maliciosas através de comportamentos específicos na realização de consultas DNS. Por exemplo, máquinas infectadas pelo verme **MyDoom.A** realizam a tentativa de obter o endereço IP atrelado ao nome de domínio "www.sco.com"antes de atacar o *website* da SCO[SYM 2010]. A divisão da organização *HoneyNet* localizada na Malásia[1] divulgou alguns estudos sobre a detecção do vírus **Conficker** através das consultas DNS que as máquinas infectadas realizavam para um determinado endereço, que provavelmente se referia ao centro de comando (C&C) desse vírus[CON 2010, CON 2010a]. Assim como também foi possível verificar que, enquanto o foco das preocupações se concentrava na variação C do Conficker (também conhecida como "o vírus do 1º de abril", pois espalharam-se boatos que ele iria ser ativado apenas nesse dia, permanecendo invisível até então), a variação A buscava atualizações através do endereço "trafficconverter.biz"
[CON 2010a].

No entanto, para a maioria das atividades maliciosas, normalmente é bastante difícil de encontrar um tipo de consulta DNS característico, que possa ser utilizada como uma *signature query*[ISH 2005] para identificar atividades maliciosas através da análise do tráfego DNS. Podemos tomar o caso das *botnets* como exemplo. *Botnets são redes formadas por computadores infectados com bots. Estas redes podem ser compostas por centenas ou milhares de computadores. Um invasor que tenha controle sobre uma botnet pode utilizá-la para aumentar a potência de seus ataques, por exemplo, para enviar centenas de milhares de e-mails de phishing ou spam, desferir ataques de negação de serviço, etc.*[CER 2010] A aparição de *botnets* vem crescendo em um ritmo assustador, chegando ao aparecimento de 5 milhões de novos *bots* por mês[CHO 2007], e tem se tornado uma das ameaças de segurança mais críticas da rede[CHO 2007, HON 2010]. Existem publicações que comprovam que atividades de *botnets* podem ser descobertas e rastreadas através da monitoração cuidadosa do tráfefo DNS realizado pelos *bots*[CHO 2007,

[1] http://blog.honeynet.org.my/

VIL 2009]. No entanto, há uma grande gama de atividades que podem ser desempenhadas por uma *botnet*, tal como o acesso aos seus servidores de comando e controle, ataque à *websites* (mais comumente ataques de DDoS), até o envio de *spam*, e a ordem de execução dessas atividades pode não estar diretamente presente na aplicação que a realiza, mas sim em um controlador externo(o mais comum é que essas ordens sejam transmitidas através de canais de controle como o IRC). Assim, não é possível identificar todas as *queries* DNS que possam caracterizar o funcionamento de uma *botnet*. Mesmo que essas *queries* possam ser identificadas, ainda assim elas corresponderiam a apenas uma parte de todas as consultas que possam indicar um comportamento suspeito, e não é possível detectar os *bots* com uma alta probabilidade de acerto[ISH 2005].

Mas, por outro lado, vermes de envio de *e-mails* em massa, tais como o **Netsky** se propagam através do envio de e-mails contendo aplicações maliciosas através dos endereços de e-mail que são encontrados na máquina infectada. Esses vermes enviam consultas DNS para encontrar o servidor de e-mail relacionado ao endereço de e-mail encontrado, e com isso deixam rastros de sua atividade no tráfego DNS. Mais especificamente, foi observado que o número de consultas DNS enviadas à diversos ISPs obtiveram um aumento significativo após a propagação de vermes de envio de *spam* na rede[ISH 2005].

Esse trabalho se propõe a apresentar uma implementação e um estudo de caso da técnica de detecção de *hosts* infectados por vermes de envio de e-mails em massa através da análise do tráfego DNS proposta por [ISH 2005].

1.1 Motivação

A utilização da análise do comportamento de tráfego DNS para detecção de atividades maliciosas é bastante recente, sendo que as primeiras idéias acerca do tema começaram a aparecer em publicações datadas de 2003.

A idéia para este trabalho se deu através de uma das propostas apresentadas pela organização *The Honeynet Project*, para o projeto *Google Summer of Code*, na edição realizada no ano de 2010.

O *The HoneyNet Project*[2] é uma organização sem fins lucrativos, focada em pesquisa e desenvolvimento, que visa a melhoria da segurança da Internet como um todo, sem custos para o público, como eles mesmos definem. Essa organização possui sub-divisões (chamadas por eles de *Chapters*) por todo o mundo, incluíndo o Brasil[3], e é firmemente comprometida com os ideais de desenvolvimento *Open Source*[HON 2010a]. Esses ideais basicamente descrevem práticas em produção e desenvolvimento que promovem acesso ao código-fonte do *software* desenvolvido, facilitando a cooperação no desenvolvimento, assim como o compartilhamento de informações e conhecimento [OPE 2010, FOS 2010, FLO 2010, FSF 2010].

Desde 2005, o projeto *Google Summer of Code*[4], também conhecido como *GSoC*, idealizado pela companhia Google, visa promover o desenvolvimento de software *open source*, através do envolvimento de estudantes universitários no processo de pesquisa e desenvolvimento dos mesmos, fazendo com que estes adquiram conhecimento e experiência em um projeto real de grande porte. Este trabalho é realizado durante o período das férias escolares no hemisfério norte do planeta, onde o estudante (em alguns projetos são escolhidos mais de um) é orientado por um mentor durante o período de desenvolvimento,

[2] http://www.honeynet.org/about

[3] http://www.honeynet.org.br/

[4] http://code.google.com/intl/pt-BR/soc/

e ao final do projeto escolhido, o estudante recebe uma quantia em dinheiro como premiação pelo trabalho realizado, assim como o mentor, pelo tempo e auxílio despendido ao estudante[GSO 2010].

Na edição de 2010 do GSoC, uma das organizações mentoras foi a The HoneyNet Project, e um dos projetos apresentados tratava da detecção de *hosts* infectados por aplicações maliciosas através da análise do comportamento do tráfego DNS[5]. Nenhum tipo de aplicação maliciosa era especificamente explicado, sendo somente citadas algumas referências sobre a rastreamento de máquinas infectadas pelo verme *Conficker*, realizadas pela divisão da HoneyNet localizada na Malásia, já previamente citados[CON 2010, CON 2010a].

Ao realizar uma pesquisa focada nessa proposta de projeto, na busca por uma maior especificidade quanto ao tipo de aplicação maliciosa que seria analisada, foram encontrados estudos interessantes sobre a detecção de aplicações de *spam*, através da mineração dos dados obtidos em consultas DNS, e a descrição de alguns experimentos realizados em diversos tipos de ambientes. Detalhes sobre estes estudos serão mostrados no capítulo 3.

Com isso, o método proposto por [ISH 2005] foi escolhido por ser a única publicação propondo uma maneira de detectar máquinas infectadas por vermes de envio de e-mails em massa sem estar atrelado a uma organização específica de rede, que era passível de ser reproduzida no ambiente de rede do Instituto de Informática da UFRGS.

1.2 Organização do Trabalho

Primeiramente, será apresentada uma breve descrição e histórico das ameaças de rede relacionadas com a realização deste trabalho. No capítulo 2, uma análise um pouco mais detalhada a respeito de *spam* será apresentada na sessão 2.1, com a mostra de algumas métricas atuais fornecidas por centros de coleta de estatíticas da rede, assim como uma explicação a respeito de vermes e sistemas de detecção de intrusão será apresentada nas sessões 2.2 e 2.3 respectivamante.

Após, no capítulo 3 será mostrada a pesquisa realizada sobre o estado-da-arte na detecção de aplicações maliciosas através da análise de tráfego DNS, assim como trabalhos relacionados acerca ao tema.

Posteriormente, a técnica publicada em [ISH 2005] será explicada no capítulo 4. A experimentação realizada será descrita em detalhes, como a coleta de dados e o ambiente utilizado no experimento em 4.1, assim como a implementação da técnica em 4.3. Os resultados obtidos serão apresentados em 4.4, junto com a sua explanação.

Por fim, as conclusões obtidas neste trabalho podem ser vistas no capítulo 5 seguidas de sugestões de trabalhos futuros.

[5] http://www.honeynet.org/gsoc/ideas

2 AMEAÇAS DE REDE

"Informação é poder.- quem esteve acompanhando diversos meios de comunicação durante os meses de novembro e dezembro de 2010., principalmente meios difundidos na Internet, certamente leu ou ouviu esta frase diversas vezes. Estamos vivendo na era da informação[1]. Era em que o conhecimento de um determinado conjunto de informações possui um enorme poder, como pode-se perceber através do recente caso de vazamento de informações confidenciais de missões diplomáticas do Governo dos Estados Unidos da América[2], através da organização *Wikileaks*[3].

Segundo [SIL 2009]: *"Existe um consenso na nossa sociedade de que o bem mais importante do século XXI é a informação. Os últimos anos foram marcados pela popularização crescente de equipamentos de comunicação, variando de telefones celulares a computadores, passando por vários tipos de dispositivos móveis. Em consequência, há uma grande disseminação de informações estratégicas e importantes circulando em uma infraestrutura pública de comunicação: a Internet. Cresce, portanto, a necessidade de proteção, de forma a evitar que as informações sejam roubadas e impropriamente modificadas."*

De acordo com a Internet World Stats[4], 1,96 bilhão de pessoas tinham acesso à Internet em junho de 2010, o que representa aproximadamente 28,7% da população mundial. Destes, aproximadamente 40,5 milhões são brasileiros[CET 2010]. A quantidade de dados trafegadas na rede mundial de computadores já chega à quantia de 8 *exabytes*[5] de informações mensais[WOR 2010].

Com essa quantidade considerável de dados sendo trafegada, o cuidado com a área denominada *segurança da informação* se tornou um dos pontos principais dos estudos envolvendo redes de computadores na atualidade. Área esta que deve garantir quatro propriedades[SIL 2009, TAN 2002]: *confidencialidade*, *integridade*, *disponibilidade* e *autenticidade*. A confidencialidade implica que uma informação só pode ser acessada por quem detém um determinado nível de privilégio ou autorização. A propriedade da integridade significa que uma informação só pode ser modificada por quem possui a devida autorização para fazê-lo. Disponibilidade é uma propriedade que oferece a garantia de uma informação estar disponível sempre que uma parte autorizada deseje consultá-la. Por fim, a propriedade da autenticidade visa garantir que as partes envolvidas sejam realmente quem elas dizem ser, ou seja, é preciso garantir uma forma de identificá-las.

Neste capítulo, será mostrada uma abordagem sobre algumas ameaças de rede exis-

[1] http://pt.wikipedia.org/wiki/Era_da_Informação
[2] http://en.wikipedia.org/wiki/Wikileaks_cables
[3] http://en.wikipedia.org/wiki/WikiLeaks
[4] http://www.internetworldstats.com/stats.htm
[5] http://en.wikipedia.org/wiki/Exabyte

tentes, suas causas e formas de detectá-las. Na seção 2.1 será apresentado o histórico de surgimento de *spams*, algumas métricas relacionadas e alguns estudos realizados sobre os mesmos. Na seção 2.2 será mostrada uma explicação sobre *worms*, ou vermes, em geral, um breve histórico e a situação atual de vermes focados em *spam*. Na seção 2.3, uma breve explicação sobre os sistemas de detecção de intrusão será apresentada.

Maiores informações sobre outros tipos de ameaças de rede não serão apresentados pois fogem do escopo deste trabalho.

2.1 SPAM

Propagandas, correntes, boatos, fraudes e lendas urbanas são apenas alguns itens que figuram na lista de mensagens eletrônicas não-solicitadas que tem causado muita dor de cabeça aos usuários de serviços de mensagem eletrônica, assim como dos responsáveis pela administração de servidores de envio destas. O grande volume de envio indiscriminado de mensagens em massa, através de meios eletrônicos, ou *spam*, como são chamadas essas mensagens, é atualmente um dos principais problemas da comunicação eletrônica.

Segundo [DEF 2010], uma mensagem eletrônica é considerada "spam"se (A) a identidade e o contexto dos destinatário são irrelevantes, já que a mensagem é igualmente aplicável a muitos outros potenciais destinatários e (B) o destinatário não concedeu uma permissão verficável deliberada, explícita, e passível de ser revogada para que esta mensagem seja à ele enviada.

Em 2009, o Brasil tornou-se o maior emissor de *spams* do mundo, figurando como o primeiro colocado na listagem divulgada pelo [IDG 2010], com 7.7 trilhões de mensagens por ano. Segundo [CGI 2010], o principal problema para esse aumento de envio de *spams* se deve ao grande número de máquinas de usuários finais desprotegidas.

Tabela 2.1: Diferença de volume de SPAM entre 2008 e 2009 (medidas em trilhões)anunciado pela Cisco.

País	**Volume SPAM 2009**	**Volume SPAM 2008**	**Diferença de Volume**
Brasil	7.7	2.7	192.6%
Estados Unidos	6.6	8.3	-20.3%
Índia	3.6	1.6	130.4%
Coréia do Sul	3.1	1.7	81.2%
Turquia	2.6	3.8	-31.3%
Vietnã	2.5	0.5	367.7%
China	2.4	3.2	-24.3%
Polônia	2.4	1.6	43.4%
Rússia	2.3	3.7	-38.2%
Argentina	1.5	1.3	16.0%

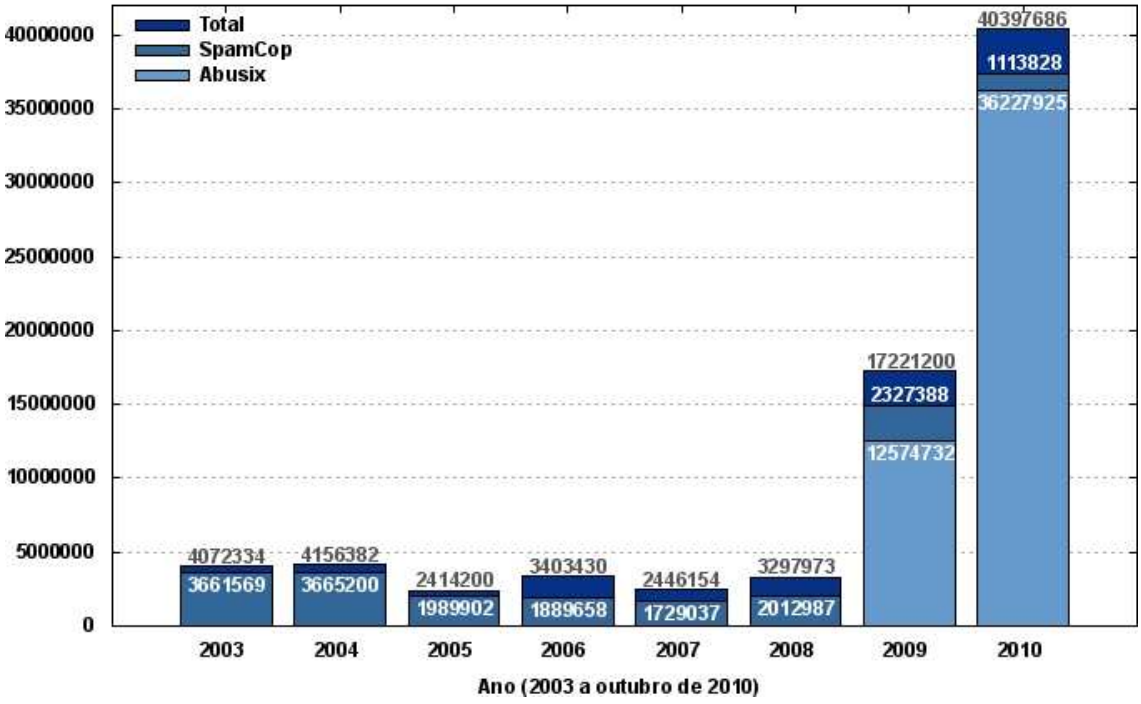

Figura 2.1: SPAMs reportados ao CERT.br por ano, referente ao mês de Outubro.

2.1.1 Tipos de SPAM

Atualmente, existem diversos tipos de spam[ANT 2010], como os que disseminam correntes, boatos, lendas urbanas, propagandas, pornografia, programas maliciosos, fraudes e golpes. Segue uma breve explicação de cada um deles:

- **Correntes:** um texto característico de uma corrente geralmente pede para que o usuário (destinatário) repasse a mensagem um determinado número de vezes ou, ainda, para pessoas conhecidas de seu círculo de contatos. O texto pode contar uma história antiga, descrever uma simpatia (superstição) ou, simplesmente, desejar sorte.

- **Propagandas:** os *spams* com conteúdo de propaganda são conhecidos como UCE (*Unsolicited Comercial E-mail*). A publicidade pode envolver produtos, serviços, pessoas, sites etc. Esse tipo de spam é motivo de discussão e polêmica, afinal, é possível fazer marketing na Internet sem fazer spam. No entanto, aqueles que insistem em divulgar sua imagem ou negócio por meio de mensagens não solicitadas, acabam comprometendo sua credibilidade.

- **Calúnia e difamação:** existem casos de envio de grande quantidade de e-mails ou mensagens eletrônicas contendo ameaças, brincadeiras inconvenientes ou difamação de amigos ou pessoas envolvidas em relacionamentos amorosos passados. O ato de enviar uma grande quantidade de mensagens, por si, já caracteriza o *spam*. Caso a entidade difamada neste tipo de mensagem sentir-se lesada, pode registrar Boletim de Ocorrência na Polícia e, eventualmente, conduzir processo por calúnia e difamação.

- **Pornografia:** o envio de material de pornografia por meio de mensagens não solicitadas é uma das modalidades mais antigas de *spam*.

- **Spit:** o **spit** refere-se ao *"spam via Internet Telephony"*. Assim, as mensagens não solicitadas também se propagam por outros meios, atingindo os usuários dos "telefones IP"(VoIP).

- **Spim:** o **spim** é o termo empregado para os *"spams via Instant Messenge"*, ou seja, o envio de mensagens eletrônicas não solicitadas por meio dos aplicativos de troca de mensagens instantâneas.

- **Através de redes de relacionamento:** a maioria dos sites de redes de relacionamento possuem opções que permitem aos usuários enviar mensagens em massa para os seus contatos, ou um conjunto de pessoas pertencente à um determinado grupo ou comunidade. Esses sites propiciam um terreno fértil para a propagação de *spam*, principalmente, de boatos e propagandas.

Independente do tipo de *spam*, e para viabilizar o envio de grandes quantidades de e-mail os *spammers* tem utilizado computadores infectados por vermes ou *bots* pertencentes à *botnets*. Esses computadores infectados são controlados por *spammers* e transformados em servidores de e-mail para envio de *spam*. Na maioria dos casos, o dono do computador demora a perceber o problema, em geral notando apenas lentidão na máquina ou na conexão com a internet. Os *bots*, ou computadores zumbis, são muito explorados pelos *spammers*, pois fornecem anonimato no envio de *spam*.

2.1.2 Origem e Curiosidades

As controvérsias acompanham o *spam* desde seu "nascimento", cuja data "oficial"pode ser considerada como 5 de março 1994[ANT 2010a]. Neste dia, dois advogados, Canter e Siegel, enviaram uma mensagem sobre uma loteria de Green Cards americanos para um grupo de discussão da USENET. O ato de enviar uma mensagem de propaganda para um fórum sem foco no assunto causou espanto e revolta em muitos assinantes do grupo.

No entanto, o pior aconteceria no dia 12 de abril de 1994, quando os advogados enviaram a mesma mensagem para diversos grupos de discussão da USENET. Foi utilizado um programa capaz de automatizar o envio em massa da mensagem de propaganda. As reações foram imediatas e negativas, gerando apelos sobre a violação da Netiqueta - um conjunto de regras de boas maneiras para os usuários da rede. O grande número de mensagens trocadas sobre o assunto comprometeu o desempenho da rede, causando um dos conhecidos efeitos colaterais do spam[6].

Durante as inflamadas discussões sobre o ocorrido, surgiu a referência ao termo *spam*, relembrando uma cena do programa de TV do grupo inglês **Monty Python**, onde vikings inconvenientes estavam em uma lanchonete, repetindo diversas vezes a palavra "spam", referindo-se a um conhecido enlatado americano composto de presunto condimentado[7].

[6] As mensagens históricas podem ser encontradas em http://web.archive.org/web/20011214024742/math-www.uni-paderborn.de/ axel/BL/CS941211.txt

[7] http://www.youtube.com/watch?v=nwy2MPT5RE

Figura 2.2: Lata de carne pré-cozida SPAM (Spiced Ham).

2.2 Vermes

Segundo a definição de [CGI 2010a], *worm*, ou verme, é um programa capaz de se propagar automaticamente através de redes, enviando cópias de si mesmo de computador para computador.

Diferente do vírus, o verme não embute cópias de si mesmo em outros programas ou arquivos e não necessita ser explicitamente executado para se propagar. Sua propagação se dá através da exploração de vulnerabilidades existentes ou falhas na configuração de softwares instalados em computadores.

Geralmente o verme não tem como conseqüência os mesmos danos gerados por um vírus, como por exemplo a infecção de programas e arquivos ou a destruição de informações. Isto não quer dizer que não represente uma ameaça à segurança de um computador, ou que não cause qualquer tipo de dano.

Vermes são notadamente responsáveis por consumir muitos recursos. Degradam sensivelmente o desempenho de redes e podem lotar o disco rígido de computadores, devido à grande quantidade de cópias de si mesmo que costumam propagar. Além disso, podem gerar grandes transtornos para aqueles que estão recebendo tais cópias.

Detectar a presença de um verme em um computador não é uma tarefa fácil. Muitas vezes os vermes realizam uma série de atividades, incluindo sua propagação, sem que o usuário tenha conhecimento. Embora alguns programas antivírus permitam detectar a presença de vermes e até mesmo evitar que eles se propaguem, isto nem sempre é possível.

Atualmente, os worms correspondem à 8.17% dos incidentes reportados ao CERT.br, entre os meses de julho e setembro, como mostra o gráfico abaixo[CGI 2010b]:

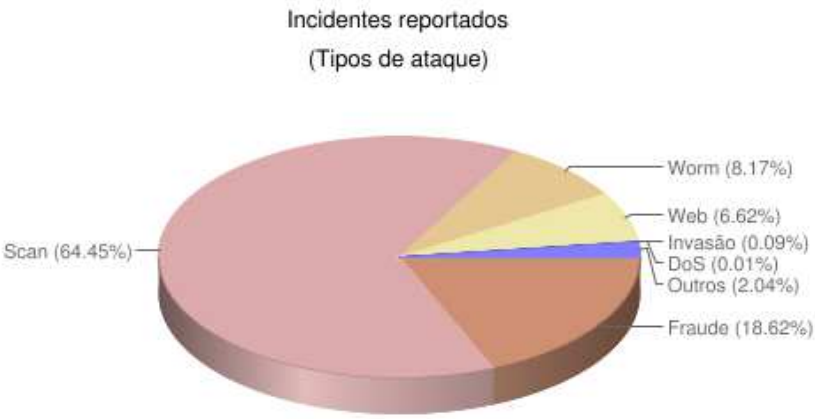

Figura 2.3: Incidentes Reportados ao CERT.br - Julho a Setembro de 2010.

Legenda do gráfico mostrado em 2.3[CGI 2010b]:

- **worm:** notificações de atividades maliciosas relacionadas com o processo automatizado de propagação de códigos maliciosos na rede.

- **DoS:** notificações de ataques de negação de serviço, onde o atacante utiliza um computador ou um conjunto de computadores para tirar de operação um serviço, computador ou rede.

- **invasão:** um ataque bem sucedido que resulte no acesso não autorizado a um computador ou rede.

- **web:** um caso particular de ataque visando especificamente o comprometimento de servidores Web ou desfigurações de páginas na Internet.

- **scan:** notificações de varreduras em redes de computadores, com o intuito de identificar quais computadores estão ativos e quais serviços estão sendo disponibilizados por eles. é amplamente utilizado por atacantes para identificar potenciais alvos, pois permite associar possíveis vulnerabilidades aos serviços habilitados em um computador.

- **fraude:** segundo o dicionário Houaiss, é "qualquer ato ardiloso, enganoso, de máfé, com intuito de lesar ou ludibriar outrem, ou de não cumprir determinado dever; logro". Esta categoria engloba as notificações de tentativas de fraudes, ou seja, de incidentes em que ocorre uma tentativa de obter vantagem.

- **outros:** notificações de incidentes que não se enquadram nas categorias anteriores.

2.2.1 Origem

A primeira aparição do termo *worm* ocorreu no romance "The Shockwave Rider", escrito por John Brunner em 1975. Neste romance, o personagem Nichlas Haflinger projeta e realiza o disparo de um *worm* coletor de dados como um ato de vingança contra as pessoas que estão no poder de uma rede nacional de informações eletrônicas que induz a conformidade em massa. Segue um dos trechos que aparecem no livro: "Vocês possuem o maior verme já solto na rede, e ele automaticamente sabota qualquer tentativa de monitorá-lo... Nunca houve um verme com uma cabeça tão dura ou um rabo tão comprido!"

Mas o primeiro verme real surgiu em 1988, mais precisamente no dia 2 de novembro. Neste dia, o estudante de graduação em ciência da computação da Cornell University, Robert Tappan Morris, disparou o verme que ficou conhecido como Morris Worm. Existe registro que esse verme infectou cerca de 10% de todos os computadores até então existentes na Internet[8], se propagando através de uma série de erros no BSD Unix e seus similares. Morris foi condenado a prestar 400 horas de serviços à comunidade e pagar uma multa de US$10.000, e foi a primeira pessoa a ser julgada e condenada sob o "Computer Fraud and Abuse Act"de 1986.[WIK 2010].

[8]http://www.bs2.com/cvirus.htm#anchor111400

2.2.2 Um breve histórico dos *Mass-mailing Worms*

Vermes de envio de *spam* se propagam através do envio de e-mails de conteúdo malicioso para os endereços encontrados na máquina infectada. A maioria destes possui um próprio mecanismo SMTP para envio de e-mails. Antes de realizarem o envio dessas mensagens, são enviadas consultas procurando por registros MX para os servidores de DNS locais, pra que o servidor de e-mail relacionado com os determinados endereços de e-mail possam ser encontrados.

Como os e-mails enviados por esses vermes possuem uma forma clara de serem identificados através dos seus arquivos em anexo (podendo ser considerada uma "assinatura"dos mesmos), esses vermes necessitam de seu próprio mecanismo SMTP para que seus e-mails não dependam dos servidores de e-mail dos provedores de internet das máquinas infectadas. No entanto, essas máquinas não podem ser identificadas através da monitoração do tráfego de e-mail, já que ele não apresenta nenhum comportamento anormal.

Dentre os vermes focados em envio de e-mails, os primeiros de que se tem notícia são o **Sobig** e o **Mydoom**[STA 2004], que surgiram em 2003 e 2004 respectivamente. Nos anos seguintes, houve um grande número de estudos publicados a respeito destes vermes.

Outros bastante conhecidos que surgiram após estes, são o **Netsky** e o **Conficker**. O Netsky teve a sua primeira aparição no dia 16 de fevereiro de 2004, e uma variação sua, chamada de "Netsky B", surgiu dois dias depois. O criador do Netsky é o mesmo responsável pela criação de outros vermes, como o Sasser. Estima-se que durante a seua existência, o Netsky teve aproximadamente 29 diferentes variações, sendo que a variação P se manteve como o verme predominante no envio de e-mails em massa por mais de dois anos e meio[SYM 2010a].

O Conficker é um verme cujo alvo é o sistema operacional Microsoft Windows. Sua primeira aparição conhecida foi em novembro de 2008[9]. Ele se aproveitava de falhas no Windows e utilizava ataques de dicionário para obter a senha de administrador do sistema, para então criar um canal de comunicação com uma centro de controle que possa comandar a máquina infectada, transformando-a em uma máquina zumbi. Desde então, o Conficker se espalhou rapidamente, e acredita-se que ele tenha causado a maior infecção de máquinas por vermes conhecida desde o SQL Slammer (2003), com mais de 7 milhões de máquinas espalhadas por mais de 200 países sob o seu controle. Tem-se o conhecimento da existência de 5 variações do Conficker atualmente.[CON 2010b]

2.2.3 Botnets

Uma *botnet* é uma coleção de recursos (*bots*) controlados por um centro de controle, e que executam tarefas a mando deste. O termo é geralmente associado com o uso de software malicioso, mas também pode se referir a uma rede de computadores utilizando software de computação distribuída. O controle dos *bots* pertencentes à uma *botnet* se dá principalmente através da utilização de canais de IRC (*Internet Relay Chat*)[CHO 2007].

Como citado anteriormente, uma das práticas mais comuns de vermes é transformarem a máquina infectada em um computador zumbi, ou *bot*, para que ela possa ser controlada remotamente e utilizada na execução de atividades maliciosas, como o envio de *spams* ou para atacar endereços de sites da Internet.

A figura 2.5 exemplifica o ciclo de vida do *spam* originado numa *botnet*: no passo **(1)**, temos o *spammer* , que infecta o receptor **(2)** com a aplicacação maliciosa **(3)**, transformando as máquinas infectadas em máquinas zumbis **(4)**. Conforme as ordens recebidas

[9]http://www.microsoft.com/protect/computer/viruses/worms/conficker.mspx

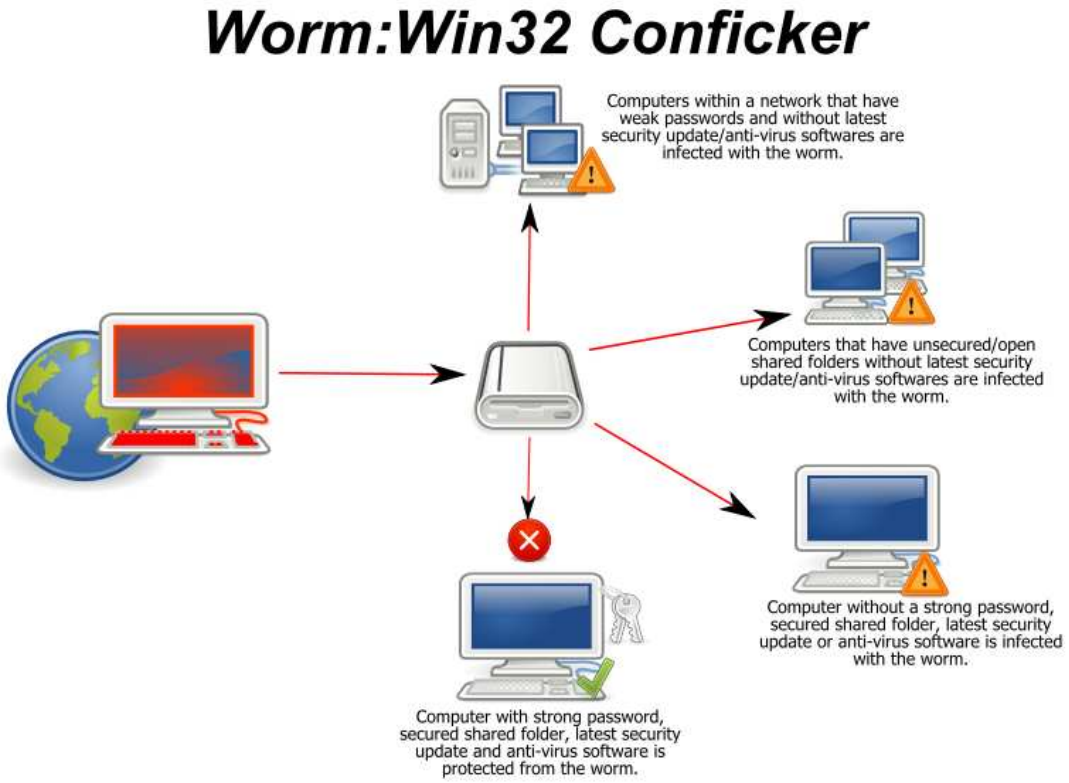

Figura 2.4: Método de disseminação do verme Conficker.

pelo controlador, essas máquinas se transformam em disseminadoras de vírus e *trojans* **(6)** ou viram emissoras de *spam* **(7)**, fazendo com que outros usuários tenham acesso à essas ferramentas maliciosas através do seu recebimento na rede e se transformem em emissoras também **(8)**.

Um dos vermes mais conhecidos na infecção de máquinas para a criação de *bots* é o *Storm Worm*. Este verme ganhou notoriedade no ano de 2007, onde era enviado através de um e-mail que anunciava desastres climáticos, possivelmente envolvendo uma grande quantidade de vítimas. Registra-se que ele foi responsável por 8% das infecções por *malware* no mundo em 22 de janeiro de 2007.[WIK 2010a]. Estima-se que a *botnet* formada por *bots* infectados pelo Storm Worm, chamada de *Storn Botnet* é uma das maiores existentes, podendo conter entre 1 a 50 milhões de máquinas infectadas pelo verme[WIK 2010b].

Conforme [CHO 2007], cerca de 172.000 *bots* são criados por dia, o que significa um aparecimento de mais de 5 milhões de *bots* por mês. Segundo publicações recentes realizadas pela Symantec, 30.000 *bots* são observados por dia. Uma única *botnet* pode chegar a ter mais de 140.000 *bots* pertencentes à ela, e já foram reportados ataques de DDoS com mais de 10Gbps de capacidade realizados por *botnets*.

Na imagem 2.6, o funcionamento de uma botnet é melhor exemplificado: no passo **(1)**, o *botmaster* dissemina aplicações maliciosas para transformar as máquinas infectadas em máquinas zumbis que ficarão sob seu controle **(2)**. Assim, através de alguma proposta que envolva ganhos financeiros, ou algum outro benefício **(3)** ele utiliza a sua rede de zumbis para espalhar alguma informação, como mensagens não-solicitadas, por exemplo **(4)**.

Apesar dos principais usos de *botnets* ser a execução de atividades maliciosas, como o envio de *spams*, roubo de informações das máquinas infectadas, efetuar ataques de ne-

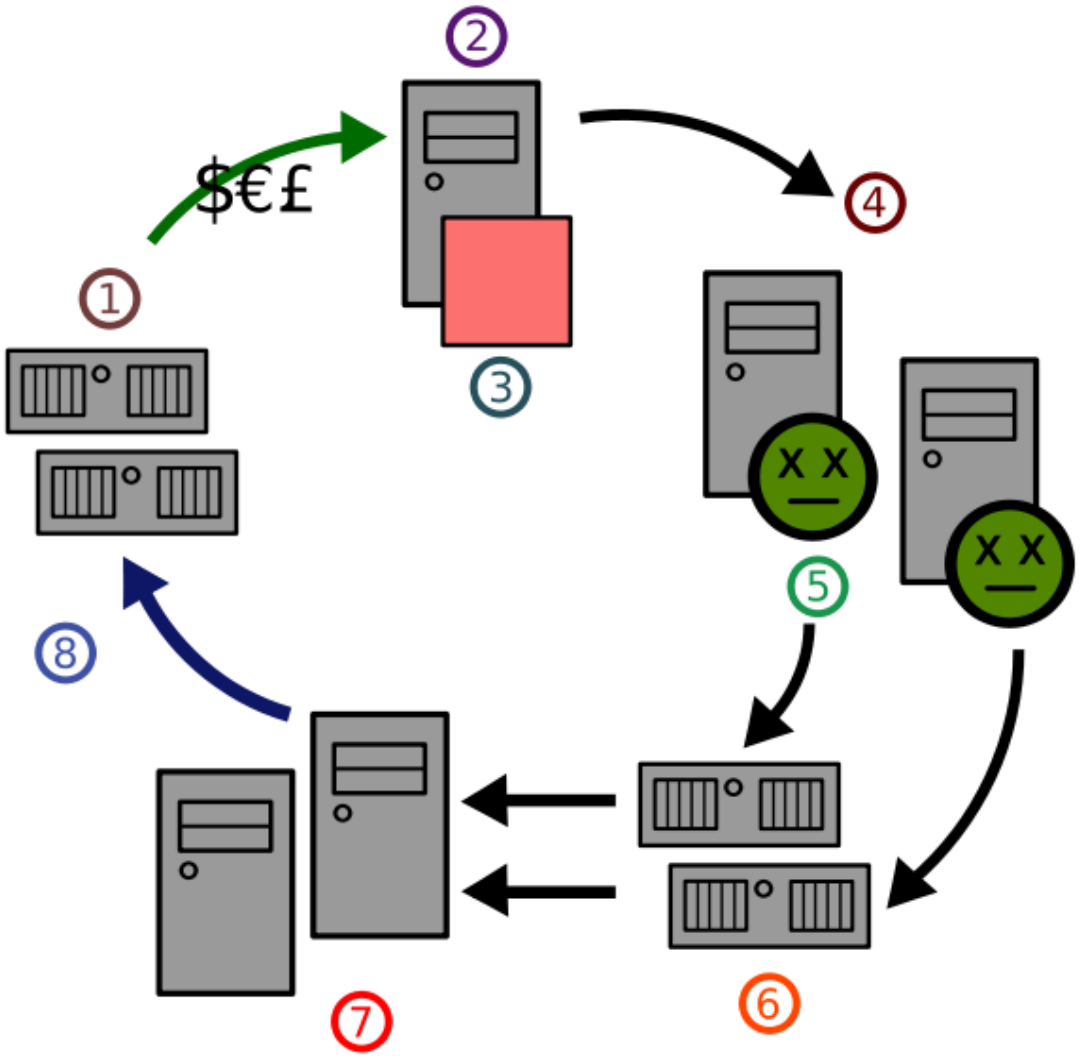

Figura 2.5: Ciclo de vida do spam em uma botnet

gação de serviço visando o lucro financeiro, as *botnets* também podem ser utilizadas para processamento distribuído, utilizando os recursos dos *bots* na resolução de problemas que auxiliam estudos sobre astronomia ou sobre o aquecimento global, como é realizado pela rede BOINC[10] ou até mesmo em protestos de cunho político, como a recente mobilização de mais de 30.000 máquinas (voluntárias) na realização de ataques de DDoS em organizações que prejudicaram de alguma forma a organização Wikileaks[TEC 2010].

Poucos estudos foram realizados sobre *botnets* até hoje, sendo esta considerada uma área bastante promissora para pesquisa.

[10]http://boinc.berkeley.edu/index.php

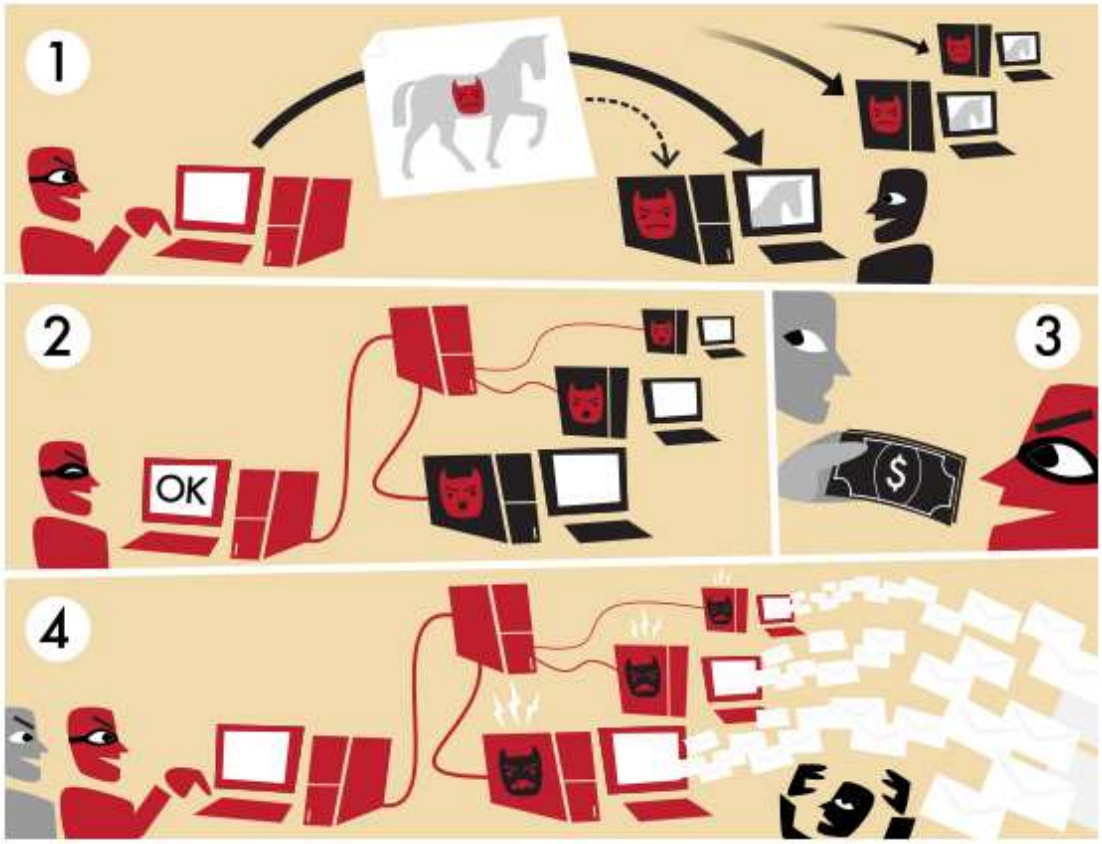

Figura 2.6: Funcionamento de uma botnet.

2.3 Sistemas de Detecção de Intrusão

Sistemas de Detecção de Intrusão (IDS) são dispositivos em hardware ou aplicações de software que monitoram a atividade de redes ou sistemas em busca de atividades maliciosas ou violações de políticas pré-estabelecidas, e produz relatórios para uma estação de gerenciamento, para que se possa ter um acompanhamento e efetuar análises posteriores dessas atividades.

Existe também o conceito de Sistemas de Prevenção de Intrusão (IPS), que é o processo de detecção de intrusão visando evitar a ocorrência de incidentes, antes que eles possam causar algum dano. Sistemas de Detecção e Prevenção de Intrusão (IDPS) são primariamente focados em identificar possíveis incidentes, armazenando informações a respeito dos mesmos (nos chamados "arquivos de *logs*"), na tentativa de pará-los e reportar a sua ocorrência para os administradores responsáveis pela segurança do sistema ou da rede.

Além disso, organizações também utilizam IDPSs para outros propósitos, como a identificação de problemas com políticas de segurança, documentação de ameças já existentes, além de dissuadir indivíduos a violares políticas de segurança. IDPSs têm se tornado indispensáveis em toda infraestrutura de segurança de redes e sistemas de praticamente qualquer organização.

IDPSs tipicamente registram informações de eventos observados, notificam administradores responsáveis pela segurança dos sistemas ou da rede monitoradas, e produzem relatórios a respeito dos dados coletados. Muitos IDPs também podem responder à uma ameaça previnindo que a mesma ocorra com sucesso. Eles costumam utilizar diversas

técnicas de resposta, que envolvem a parada do ataque, mudança do ambiente de segurança (reconfiguração de um *firewall* durante a ocorrência de um ataque, por exemplo) ou modificando o conteúdo do ataque.

Como a técnica de análise de consultas DNS para detectar aplicações de envio de *spam* se baseia no mesmo conceito de IDSs, que é realizar a monitoração de um determinado evento, guardar informações sobre o mesmo para posteriormente analisá-los e chegar à uma conclusão a respeito dos mesmos, podemos considerar essa técnica como um Sistema de Detecção de Intrusão.

3 DETECÇÃO DE APLICAÇÕES MALICIOSAS ATRAVÉS DE DNS

O sistema, ou serviço de nomes de domínios (DNS) foi bastante citado nos capítulos e sessões anteriores, mas até então não foi fornecida uma explicação mais aprofundada sobre o mesmo. Neste capítulo será apresentada uma breve descrição deste sistema e algumas características de seu protocolo, necessárias para o entendimento da técnica mostrada no capítulo 4. Também serão citados alguns trabalhos relacionados à detecção de aplicações maliciosas através de análise de DNS em 3.2.

3.1 Sistema de Domínio de Nomes

Ao falarmos de DNS, muitas vezes não é claro se estamos nos referindo ao **Sistema** de Domínio de Nomes ou ao **Serviço** de Domínio de Nomes. Apesar de estarem fortemente relacionados e serem interdependentes, suas definições são distintas.

O DNS (domain name system) forma um sistema sofisticado que define um protocolo de aplicação responsável por administrar nomes de máquinas e endereços IP na Internet. O DNS é um exemplo de um sistema distribuído de proporções mundiais, já que é sob o seu sistema de resolução de nomes que a Internet inteira opera.[SIL 2009]

O Sistema de Nomes de Domínio (DNS) se constitui em um sistema hierárquico de nomes, construído em uma base distribuída de máquinas, serviços, ou qualquer recurso conectado na Internet ou em uma rede privada. Ele associa várias informações com nomes de domínios designados para cada uma das entidades participantes. Mais importante que isso, o DNS provém um serviço que realiza a tradução de nomes de domínios (facilmente lembráveis por seres humanos) para identificadores numéricos associados com equipamentos de rede, com o propósito de localizar e endereçar esses equipamentos ao redor do mundo. A vantagem de se utilizar tal sistema se deve ao fato de que nomes são mais fáceis de serem lembrados por pessoas, além de que, caso o endereço físico mude, o serviço ou a máquina ainda poderá ser encontrada pelo seu nome lógico, sem que o usuário precise decorar um novo endereço. Uma analogia bastante utilizada para explicar o funcionamento do DNS, é como se ele funcionasse como uma "agenda telefônica"da Internet, traduzindo nomes que são lembrados facilmente (nomes de domínio), por números (endereços IP).

O Sistema de Domínio de Nomes também especifica o funcionamento técnico deste serviço de dados. Ele define o protocolo DNS, uma definição detalhada das estruturas de dados utilizadas e as trocas de informação utilizadas no DNS, como parte da suíte de protocolos utilizados na Internet.

De acordo com [TAN 2002], o DNS é um sistema de nomes de domínios hierárquico

e distribuído, ou seja, um domínio (como .org) pode ser particionado em diversos sub-domínios (.org.acm) e assim por diante. Esses domínios podem ser representados por uma estrutura de árvore 3.1, onde no topo se encontram os domínios de alto nível (*top-level domains*) que representam entidades genéricas (comerciais, governamentais, educacionais, etc.) e países, e na base ou folhas, os sub-domínios que ligam os respectivos hosts ao nome hierárquico.

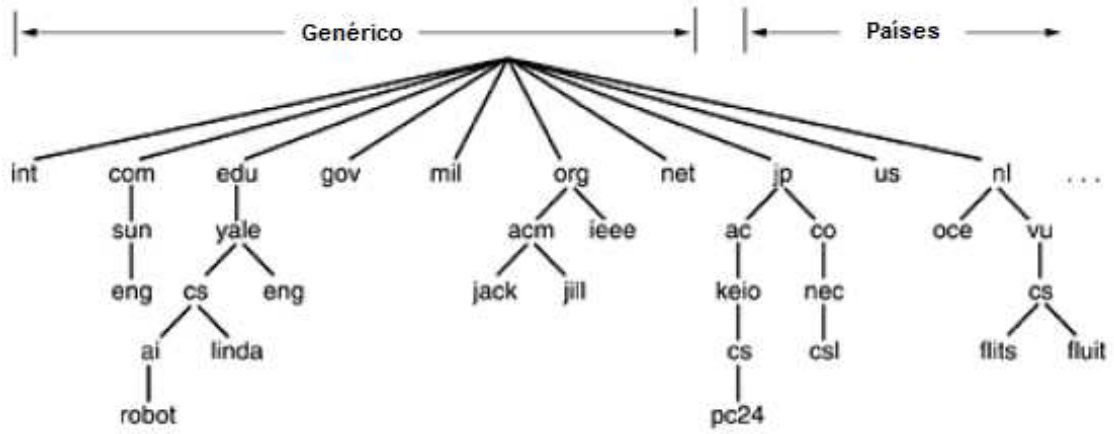

Figura 3.1: Estrutura hierárquica do DNS.

3.1.1 A mensagem DNS

Mensagens DNS de solicitação e resposta compartilham um mesmo formato, que é mostrado na figura 3.2.

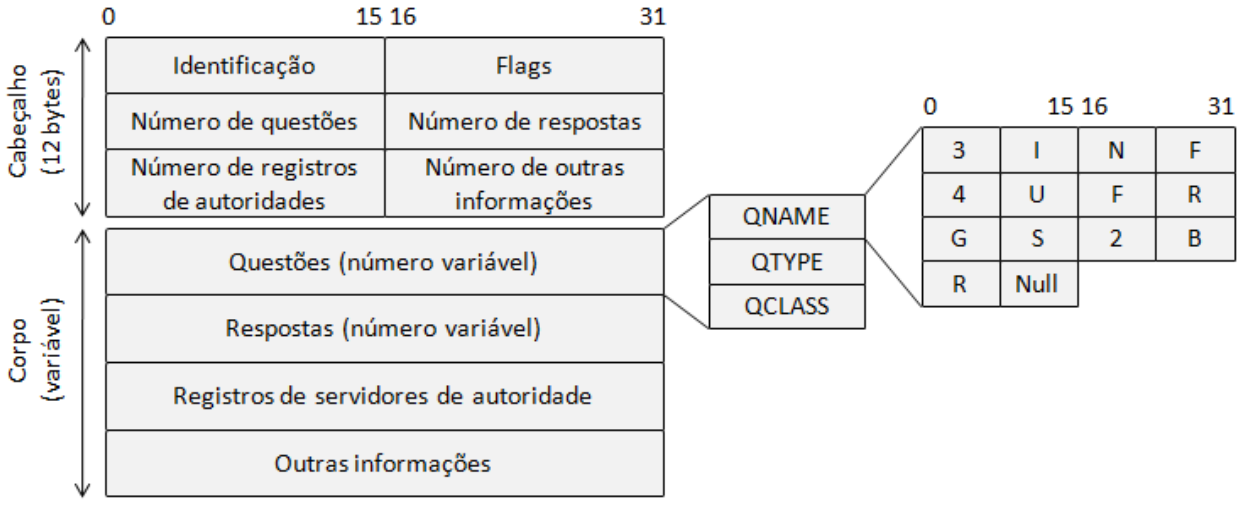

Figura 3.2: Formato das mensagens DNS.

O cabeçalho de uma mensagem DNS é formado pelos primeiros seis campos mostrados na imagem 3.2, que ao todo possuem o tamanho fixo de *12 bytes*.

Dentro do cabeçalho podemos encontrar os seguintes campos:

- **Identificação:** o valor deste campo é definido pela máquina que realiza a solicitação, de forma a identificá-la. Esse valor é copiado na mensagem de resposta fazendo com que esta chegue ao seu destinatário correto.

- **Flags:** esse campo é divido entre outros oito campos. O campo *QR* indica se a mensagem é uma solicitação ou resposta. O campo *código* indica se a solicitação é tradicional (valor 0), reversa (valor 1) ou de verificação do estado interno do servidor DNS (valor 2). O campo *AA* indica se o servidor em questão é um servidor autoritativo, ou seja, é o responsável final pelo domínio em questão. O campo *TC* indica se a mensagem foi truncada por ultrapassar o limite de *512 bytes*. O campo *RD* indica se a resolução será realizada de forma recursiva ou iterativa. Como nem todos servidores possuem suporte a resoluções recursivas, essa capacidade é indicada pelo campo *RA* que estará habilitado caso o servidor em questão ofereça suporte à resoluções recursivas. E, por fim, o campo *CResp* apresenta o código de resposta associado à solicitação DNS, sendo os códigos mais comuns de valor 0 (indica ausência de erro) e de valor 3 (indica erro de nome).
- **Número de questões:** número de perguntas realizadas. Esse número normalmente é 1.
- **Número de respostas:** número de respostas que serão enviadas. Este valor é, pelo menos, 1.
- **Número de registros de autoridades:** número de servidores autoritativos para o domínio em questão.
- **Número de outras informações:** informações extras que possam ser relevantes na mensagem.

Assim, para acessar um host que possui um endereço de rede através de um nome, uma aplicação deve fazer uma chamada ao procedimento de resolução (*resolver*), passando o nome como parâmetro. O *resolver*, por sua vez, se comunica com o servidor DNS local, o qual irá traduzir o nome fornecido para um endereço de rede (IP). Essa tradução é feita através da associação do nome do domínio à registros de recursos (*resource records*). Um *resource record* é identificado pelos campos demonstrados na imagem 3.3.

DNS Resource Record (RR)

NAME	sequence of labels, variable length
TYPE	integer, 16 bits
CLASS	integer, 16 bits
TTL	integer, 32 bits
RDLENGTH	unsigned integer, 16 bits
RDATA	string of octets, variable length

Figura 3.3: Estrutura de um Resource Record.

O campo **NAME** identifica a que domínio o recurso se aplica, e é utilizado como a chave primária na busca através do nome fornecido. O campo **TYPE** informa o tipo do

registro, definindo o significado da informação que ele carrega. Os principais tipos estão listados na tabela 3.1. O campo **CLASS** informa a família de protocolos utilizados, onde comumente é utilizado IN (Internet). O campo **TTL** representa o tempo de validade do registro e indica quando um registro em cache deve ser atualizado. O campo **RDLENGTH** representa o tamanho do registro de recurso, o qual depende do valor atribuído ao campo TYPE. Por fim, o campo **RDATA** representa os dados do registro, sendo que sua semântica é definida pelo tipo do registro. Por exemplo, para um *resource record* do tipo A, o campo RDATA informará o endereço IP de um *host*. Como exemplo, o seguinte registro inf.ufrgs.br 86400 IN A 143.54.11.16 indica que o nome de domínio inf.ufrgs.br está ligado ao endereço de rede 143.54.11.16. No ambiente de rede, esses registros transitam em forma binária, encapsulados em pacotes DNS sobre a camada de transporte[TAN 2002].

Tabela 3.1: Principais tipos de registro DNS utilizados.

Tipo de Registro	RFC Relacionada	Descrição
A	1035	Registro de endereços.
AAAA	3596	Registro de endereços IPv6.
CNAME	1035	Registro de nome canônico.
MX	1035	Registro de troca de mensagens eletrônicas (*mail exchange*).
NS	1035	Registro de nomes (*name server*).
PTR	1035	Registro de recurso de ponteiro (*pointer record*).
SOA	1035	Registro de início de autoridade.
SRV	2782	Localizador de serviço.
TXT	1035	Registro de texto.

3.2 Trabalhos Relacionados

Em [WHY 2005] os autores propõem analisar a correlação de *queries* DNS com os pedidos de conexões efetuados. Visto que a grande maioria dos vermes existentes tentam infectar novos hosts atribuindo IPs de forma randômica, uma conexão que não possua atividade DNS pode ser considerada anômala. Contudo, como existem aplicações que legitimamente não necessitam de consultas DNS (pois atribuem o endereço IP diretamente), é proposto também a criação de uma "whitelist", ou seja, uma lista de hosts que podem estabelecer conexões sem realizar consultas DNS. Como podem haver diversos aplicativos legítimos em uma rede aberta, a criação desta *whitelist* pode se tornar algo custoso, porém, para redes mais restritas, como é o caso de redes corporativas, a solução se torna viável.

Em [JUN 2004] os autores desenvolvem um algoritmo chamado Threshold Random Walk (TRW) para identificar hosts maliciosos. Foi observado que, ao tentar rastrear novos alvos, hosts infectados acabam tentando acessar diversos nomes e serviços que não existem. Assim, o algoritmo se baseia neste comportamento para determinar a probabilidade de um host estar infectado, onde conexões legítimas diminuem essa probabilidade, enquanto conexões falhas a aumentam.

Em [BRO 2007] é proposta a implementação de um método de detecção de *spam* distribuído, focado no combate a redes zumbis. A principal dificuldade em conter *spam*

através de *botnets* reside no fato de que cada nodo infectado envia uma pequena quantidade de e-mails para um mesmo servidor, o que torna o uso de *blacklists* ineficiente. Baseado nisso, os autores sugerem a implementação de uma rede de troca de informações distribuída, chamada de Trinity, onde cada nodo é acoplado em um MTA (Mail Transfer Agent). Ao receber uma requisição de recebimento de email, este nodo é capaz de se comunicar com os outros através de uma base de dados distribuída, a qual informa se o endereço de origem do e-mail já possui registro de atividades recentes em outros MTAs. Contudo, as principais dificuldades encontradas são, detectar mensagens de SPAMs que ainda não possuem um registro de atividades recentes (o que ocorre no início da atividade de SPAM), e depender da colaboração dos MTAs, pois a eficácia do método depende da quantidade de nodos presentes na rede Trinity.

Em [CHO 2007] os autores propõem a detecção de *bots* baseado em duas características. A primeira consiste em analisar o tráfego DNS. Considerando que esses bots são controlados por um servidor único (C&C ou command-and-control server) e que constantemente o endereço desse servidor é alterado para dificultar a detecção da *botnet*, é esperado que as máquinas infectadas façam uma requisição DNS em massa para descobrir o novo endereço do servidor C&C, gerando um pico de consultas DNS para um mesmo nome. A segunda consiste em detectar a migração dos bots de um servidor C&C para outro. Como são necessários dois domínios (o antigo e o novo), é possível estimar essa migração através da análise do tamanho da lista de IPs dos diferentes domínios. Contudo, esse tipo de análise se mostra custosa para um servidor real, devido ao grande número de dados e a complexidade de se casar essas informações com todas as requisições efetuadas.

4 EXPERIMENTAÇÃO

Conforme descrito no capítulo 3, a mineração do tráfego DNS pode apresentar resultados bastante interessantes para obter informações sobre o comportamento dos dados transmitidos em uma rede, assim como de seus respectivos transmissores.

O objetivo deste capítulo é detalhar o entendimento, a implementação e os resultados obtidos através da técnica de mineração de tráfego DNS apresentada em [ISH 2005] para detectar máquinas infectadas por vermes de envio de e-mails em massa.

4.1 Coleta de Dados

A coleta dos dados utilizados no experimento se deu através da ferramenta **DNSCAP**[1], no ambiente do Instituto de Informática da Universidade Federal do Rio Grande do Sul[2]. Foram coletadas todas as consultas DNS realizadas no dia 17/11/2010 do Instituto de Informática, no horário das 8:00 às 17:00, já que a maior utilização da rede ocorre durante esse período.

A ferramenta DNSCAP consiste em um utilitário *Open Source* de captura de tráfego especializada para tráfego DNS. Ela produz dados no formato binário PCAP[3] (**Pa**cket **Ca**pture), um formato relativamente comum utilizado por ferramentas bastantes conhecidas, tais como o TCPDUMP[4]. Foi preferível utilizar a ferramenta DNSCAP ao invés do TCPDUMP por ser um utilitário mais poderoso e focado em tráfego DNS, que era o objetivo da captura de dados.

Para a captura, foi executada a seguinte linha de comando, em ambiente Unix

```
dnscap -w diretorio_de_destino
```

Com isso, todas as consultas e respostas DNS serão armazenadas em um arquivo de log, que será armazenado dentro do diretório definido pelo parâmetro **diretorio_de_destino**.

Para o processamento dos dados, o arquivo contendo o tráfego DNS foi dividido em arquivos menores, contendo o tráfego realizado por hora (o que resultou em 10 arquivos de dados em formato *pcap*), pois o volume de dados obtidos era bastante considerável para ser processado de uma única vez. Para a divisão do arquivo de dados, foi utilizada a ferramenta **editcap**[5], um utilitário da ferramenta **Wireshark**[6] para edição de arquivos

[1] https://www.dns-oarc.net/tools/dnscap
[2] http://www.inf.ufrgs.br/
[3] http://www.tcpdump.org/pcap.html
[4] http://www.tcpdump.org/
[5] http://www.wireshark.org/docs/man-pages/editcap.html
[6] http://www.wireshark.org/

em formato *pcap*. A seguinte linha de comando foi utilizada, substituíndo **<seconds per file>** por **3600**

```
editcap -i <seconds per file> <infile> <outfile>
```

Após isto, foi feita a transformação do arquivos binários em arquivos contendo informações em formato textual, utilizando a ferramenta **tshark**[7], também um utilitário do Wireshark, que permite a conversão de formato de arquivos *pcap*. A seguinte linha de comando foi utilizada para realizar essa conversão, sendo aplicada à todos os arquivos previamente gerados

```
tshark -V -r file_to_convert.pcap > file_to_convert.txt
```

Ao todo, foram realizadas 531.298 consultas nesse período. O número de estações com IPs de origem distintos é 10.770, e o número de consultas distintas realizadas é 87.562. Uma consulta distinta é definida pela tupla (*query content, query type*).

A porcentagem de cada tipo de consulta realizada está listada na tabela 4.1.

Diferentemente do ambiente utilizado em [ISH 2005], que se tratava de um grande ISP do Japão, o ambiente utilizado para a realização deste experimento é um ambiente controlado, e que utiliza algumas políticas e aplicações de segurança no combate ao *spam*, sendo utilizadas as seguintes aplicações: filtro de conteudo (SpamAssassim[8]+ClamAv[9]), RBL[10], graylist[11] e listas negras locais, que trabalham no nível de IP e envelope SMTP. Também não são aceitas mensagens oriundas de servidores que não tenham DNS reverso cadastrado ou que correspondam a máquinas domésticas (dsl).

Atualmente isto está localizado na servidora SMTP (puma.inf.ufrgs.br) do Instituto de Informática, mas segundo o que foi informado pela administração de rede do mesmo, está sendo realizada uma migração dessa organização centralizada para uma distribuida que funcionará em camadas. Como parte da migração, o sistema de filtragem de *spam* será modificado do SpamAssassim para o DSPAM[12].

Segundo [ISH 2005], ao observador o tráfego DNS, pode-se perceber algumas consultas características de estações que enviam muitas consultas para recursos MX. O tipo de consulta mais comum é aquele cujo tipo da consulta é indefinido. Ao monitorar a sequência de *bytes* das consultas realizadas, foi descoberto que isso ocorre devido à um erro de formatação causado por um *bug* de vermes que enviam consultas de tipo MX. Em uma consulta DNS, no campo contendo a requisição para o *nameserver*, o final do QNAME (*query name*) é indicado por um *byte* nulo, seguido pelo QTYPE (*query type*) [MOC 87]. No entanto, nessas consultas em que o erro de formatação ocorre, o *byte* nulo é acidentalmente inserido dentro do QNAME. Portanto, os *bytes* após o *byte* nulo são lidos como sendo o QTYPE, que não está corretamente definido.

Sendo assim, estações que enviaram uma consulta DNS com esse conteúdo podem claramente ser consideradas infectadas por vermes de envio de SPAM. Além do mais, pode-se utilizar esse tipo de consulta como uma *signature query*[ISH 2005], ou seja, uma assinatura que indica quais são as estações previamente infectadas. 126 estações possuem registros de envio desse tipo de consulta, o que corresponde à 1.17% do total de estações analisadas.

[7] http://www.wireshark.org/docs/man-pages/tshark.html
[8] http://spamassassin.apache.org/
[9] http://www.clamav.net/
[10] http://en.wikipedia.org/wiki/DNSBL
[11] http://www.greylisting.org/
[12] http://www.nuclearelephant.com/

Tabela 4.1: Taxa de ocorrência de cada tipo de consulta DNS realizado (%)

Tipo de Consulta	Número de Consultas	Taxa de ocorrência
A	363441	68.41
MX	42384	7.98
PTR	38269	7.20
NS	423	0.08
SOA	2058	0.39
CNAME	32	0.00
AAAA	68150	12.83
ANY	454	0.08
SRV	88	0.02
Outras	15999	3.01

4.2 Método Proposto

O objetivo do método proposto em [ISH 2005] consiste em detectar uma estação infectada por vermes utilizando as consultas DNS (que de agora em diante serão citadas como *queries*, ou *query* ao se referir a uma única consulta, para melhor aproximação com os termos utilizados em [ISH 2005]) enviadas pelo hospedeiro. Definindo de uma forma melhor, para cada estação (que de agora em diante será chamada de *host*) h, dado que o conteúdo das *queries* enviadas pelo *host* h seja Q_h, queremos calcular a probabilidade de um *host* estar infectado, que pode ser definida como

$$Pr(\textit{host h está infectado}|Q_h). \tag{4.1}$$

Para calcular essa probabilidade, é proposto um procedimento composto de três passos, iniciando com a *signature query* encontrada no tráfego DNS capturado, como mostrado na figura 4.1 (as setas mostradas nos três passos significam que os procedimentos são realizados na ordem em que elas sugerem).

As equações a seguir foram propostas por [ISH 2005], e foram utilizadas as idéias apresentadas em [GRA 2010, LOU 2003, ROB 2003] sobre filtragem de *spam* utilizando inferências Bayesianas.

O primeiro passo é classificar cada *host* baseado no envio da *signature query*, caso o *host* tenha ou não enviado uma *query* classificada pela *signature query*.

Consideramos $H = h$ o conjunto de total de *hosts* que tenham enviado mais de uma *query* durante o tempo de captura do tráfego DNS, e consideramos que o conjunto de todos os *hosts* que enviaram a *signature query* é definido como I.

Ou seja,

$$I := \{h \in H | h \textit{ realizou o enviou da signature query}\}, \tag{4.2}$$

e definimos os *hosts h* pertencentes à *I* como *hosts* inicialmente infectados.

O segundo passo é calcular a pontuação do conteúdo de uma *query* que expresse a probabilidade de um *host h* estar infectado, dado que o *host* tenha realizado o envio daquela *query*. Somente um dos métodos apresentados em Ish05 será mostrado, pois foi

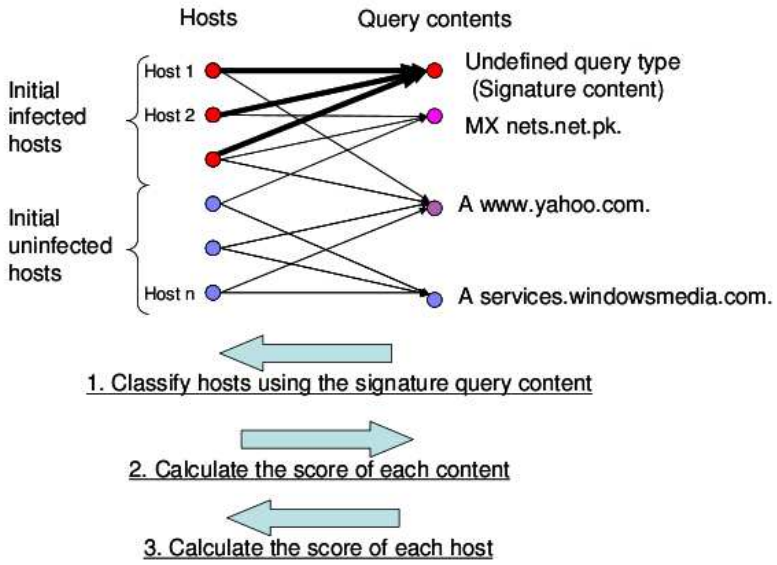

Figura 4.1: Ligação dos pares *host* e *query* utilizados na definição da probabilidade de infecção de um determinado *host*. A imagem foi retirada da publicação [ISH 2005].

o que se provou mais efetivo e o mesmo foi utilizado no desenvolvimento da aplicação que realizou a análise e classificação dos dados colhidos.

Este método consiste em calcular a pontuação de uma *query* baseada no número de hosts inicialmente infectados que enviaram essa mesma *query*. Uma *query* é definida pela tupla (*query content*, *query type*).

Primeiramente é calculada a taxa de ocorrência de cada *query* q pela seguinte equação

$$I_H(q) = \frac{\#h \in I | q \in Q_h}{\#I} \tag{4.3}$$

Esse valor indica a taxa que o conteúdo de q é consultado por *hosts* inicialmente infectados.

Como descrito anteriormente, porque mesmo *hosts* infectados podem enviar *queries* para nomes de domínio populares (tal como www.google.com), essas *queries* podem também obter um alto índice $I_H(q)$. Portanto, o índice $I_H(q)$ não é apropriado para estimar a suspiciosidade do conteúdo da *query* q. Para evitar isso, também é calculada uma taxa que mostra a possibilidade desta *query* ter sido enviada por *hosts* não infectados inicialmente.

$$N_H(q) = \frac{\#h \in \bar{I} | q \in Q_h}{\#\bar{I}} \tag{4.4}$$

Em seguida, a pontuação de uma *query* q, definida como $S_H(q)$, é calculada da seguinte forma:

$$S_H(q) = \frac{I_H(q)}{I_H(q) + N_H(q)} \tag{4.5}$$

$S_H(q)$ é uma estimação grosseira da probabilidade de um *host* h que envie a *query* q esteja infectado com um verme de *spam*. Na realidade, para que a probabilidade real seja calculada, é preciso conhecer a probabilidade de um *host* h estar infectado, mas por essa medida não ser conhecida de antemão, esse número é definido como 0.5[ISH 2005].

Com esse cálculo, uma *query* enviada por apenas um *host* é pontuada como 0 ou 1, dependendo se o *host* estava marcado como inicialmente infectado ou não. No entando, não podemos esperar que essas *queries* realmente tenham uma probabilidade extrema como as citadas. Mas, como foi mostrado em [ROB 2003], pode-se modificar essa probabilidade utilizando as constantes P_{init} e N_{init}, como segue

$$S'_H(q) = \frac{P_{init} + N_q * S_H(q)}{N_{init} + N_q} \tag{4.6}$$

onde N_q é o número total de *queries* com aquele determinado conteúdo.

Como pode ser observado, a pontuação inicial de uma determinada *query* é dada por P_{init}/N_{init}, o que indica que a probabilidade *a priori* de um *host* estar infectado dado que esse *host* envia essa determinada *query*. Assim que o número de *queries* iguais a q aumenta, o índice $S_H(q)$ é ponderado no cálculo de $S'_H(q)$. Por não se ter nenhum conhecimento *a priori* além da *signature content*, pode-se definir o parâmetro P_{init} para 0.5 e N_{init} para 1.

O terceiro passo é o cálculo da pontuação de um determinado *host* que indica a probabilidade que ele esteja infectado, baseando-se nas *queries* enviadas por ele e a pontuação destas.

Existem diversas maneiras heurísticas de calcular a pontuação de um *host*. É conhecido que utilizando apenas *queries* com pontuações de valores extremos (próximo de 1 ou 0), e calculando a média geométrica das pontuações, pode-se chegar à um bom resultado para os filtros de *spam* Bayesianos[GRA 2010, LOU 2003]. Podemos, portanto, definir dois limiares, T_H e T_L, para altos e baixos valores, respectivamente, e calcular a pontuação do *host* como é explicado a seguir.

Seja

$$m = \#\{q \in Q_h | S'_H(q) > T_H\} \tag{4.7}$$

$$I(h) = \begin{cases} 1 & \text{se } m = 0 \\ 1 - (\prod_{\{q \in Q_h | S'_H(q) > T_H\}} S'_H(q))^{1/m} & \text{caso contrário} \end{cases} \tag{4.8}$$

$$k = \#\{q \in Q_h | S'_H(q) < T_L\} \tag{4.9}$$

$$N(h) = \begin{cases} 1 & \text{se } k = 0 \\ 1 - (\prod_{\{q \in Q_h | S'_H(q) < T_L\}} (1 - S'_H(q)))^{1/k} & \text{caso contrário} \end{cases} \tag{4.10}$$

Finalmente, o nível de confiança de que o *host* h está infectado é calculado da seguinte forma[LOU 2003]:

$$P(h) = \frac{1 + N(h) - I(h)}{2(N(h) + I(h))} \tag{4.11}$$

Assim como a pontuação das *queries*, a pontuação acima também tem como resultado um valor entre 0 e 1, e 1 significa que o *host* está infectado com a probabilidade de 1.

Finalmente, é dito que um *host* h está infectado se o parâmetro $P(h)$ é maior que um limiar pré-definido.

4.3 Implementação

Nesta sessão, serão apresentados detalhes de uma implementação da técnica descrita na sessão sec:metodoproposto, para gerar as medidas apresentadas nas seções sec:analisetrafego e os resultados apresentados na sessão sec:resultados.

A aplicacação criada para este trabalho consiste em um conjunto de *scripts*, criados utilizando as tecnologias *Python*[13] (v.2.6.6) e *Bash/Shell Script*[14]. Também foi utilizado o SGBD *MySQL*[15] (v.14.14, distribuição 5.1.49 para debian-linux-gnu) para armazenamento e consultas dos dados do tráfego de rede capturado. Estes scripts estão sob a licença *GNU General Public License v3*[16], sendo hospedado no repositório público de código *Google Code*[17].

A publicação [ISH 2005] não cita detalhes sobre como a implementação do método proposto foi realizada, então esta implementação foi baseada em decisões de projeto tomadas pela própria autora deste trabalho, conforme o desenvolvimento da aplicação e a realização dos testes necessários.

A abordagem de armazenamento dos dados em uma base de dados foi escolhida por se tratar de uma quantidade de dados considerável, cujo processamento em memória não era possível devido a limitações de *hardware*. Além disso, as informações estariam disponíveis para anaálises futuras, sem necessidade de reprocessamento de informações.

Após a conversão dos dados do tráfego capturado em formato *pcap* explicados na sessão sec:coletadados para o formato texto, foi criado um script que realiza o *parsing* dos dados de todos os arquivos texto encontrados em um determinado diretório e armazena esses dados em uma tabela do banco de dados, com os seguintes campos:

Tabela 4.2: Tabela de armazenamento dos dados das consultas DNS

Campo da Tabela	Tipo do Campo	Descrição
source_host	VARCHAR(20)	IP de origem da requisição efetuada.
destination_host	VARCHAR(20)	IP de destino da requisição efetuada.
query_type	VARCHAR(10)	Tipo da consulta efetuada.
query_content	VARCHAR(255)	Conteúdo da *query* efetuada, ou seja, o *hostname* buscado.

As respostas DNS recebidas contidas nos arquivos de *log* do tráfego capturado foram descartadas por serem irrelevantes para a análise da metodologia apresentada.

[13] http://www.python.org/
[14] http://en.wikipedia.org/wiki/Shell_script
[15] http://www.mysql.com/
[16] http://www.gnu.org/licenses/gpl.html
[17] http://code.google.com/

Logo em seguida, foram efetuados as classificações e cálculos descritos em 4.2, 4.3, 4.4, 4.5 e 4.6. Esses dados foram armazenados em uma tabela com os seguintes campos:

Tabela 4.3: Tabela de armazenamento dos índices $I_H(q)$, $N_H(q)$, $S_H(q)$ e $S'_H(q)$

Campo da Tabela	Tipo do Campo	Descrição
query_content	VARCHAR(255)	Conteúdo da *query* efetuada, ou seja, o *hostname* buscado.
query_type	VARCHAR(10)	Tipo da consulta efetuada.
n_queries_sent	INT(10)	Número de *queries* com um determinado conteúdo e tipo enviadas.
i_index	VARCHAR(10)	Índice que indica a proporção que uma determinada consulta é enviada por *hosts* inicialmente infectados
n_index	VARCHAR(10)	Índice que indica a proporção que uma determinada consulta é enviada por *hosts* inicialmente não-infectados
s_index	VARCHAR(10)	Estimativa inicial da probabilidade que um *host* que tenha enviado a *query* esteja infectado por um verme de envio de *spam*
s_final_index	VARCHAR(15)	Probabilidade final da *query* ter sido enviada por uma aplicacação maliciosa.

Por fim, são calculados os dados necessários para a classificação efetiva dos *hosts*, conforme explicado em 4.2. Os cálculos descritos em 4.7, 4.8, 4.9 e 4.10 sáo realizados, e os dados obtidos são armazenados em uma tabela com a seguinte estrutura:

Tabela 4.4: Tabela de armazenamento dos índices $I(h)$, $N(h)$ e $P(h)$

Campo da Tabela	Tipo do Campo	Descrição
host	VARCHAR(255)	*host* do qual é calculada a probabilidade de estar infectado
m_index	INT(10)	Índice necessário para os cálculos do índice $I(h)$
ih_index	VARCHAR(15)	Índice utilizado nos cálculos da probabilidade final de um *host* estar infectado
k_index	INT(10)	Índice necessário para os cálculos do índice $N(h)$
nh_index	VARCHAR(15)	Índice utilizado nos cálculos da probabilidade de um *host* estar infectado
ph_index	VARCHAR(15)	Probabilidade final do *host* estar infectado

4.4 Resultados Obtidos

Após a execução da implementação da metodologia descrita na sessão anterior, foram obtidos as seguintes probabilidades de infecção para os hosts listados (representadas pelo índice P(h), ordenados a partir do mais significativo):

Tabela 4.5: Ranking de pontuação dos *hosts*

Posição	Host	Descrição[18]	P(h)
1	187.32.41.67	CIA DE TELECOM DO BRASIL CENTRAL	0.998
2	141.52.27.36	Forschungszentrum Karlsruhe (FZK)	0.997
3	150.164.255.131	Universidade Federal de Minas Gerais	0.997
4	202.96.209.16	CHINANET Shanghai province network	0.997
5	74.220.198.174	BLUEHOST-NETWORK-2	0.997
6	193.190.2.129	Hasselt University	0.997
7	202.96.209.23	CHINANET Shanghai province network	0.997
8	203.143.172.14	National ICT Australia	0.995
9	216.144.187.199	PENTELEDATA-4BLK	0.995
10	74.125.44.85	GOOGLE	0.995
11	200.164.110.2	Processamento de Dados do Pará	0.995
12	213.180.147.158	Grupa Onet.pl SA - ODC1-2 Services	0.995
13	72.69.227.122	Verizon Online LLC	0.995
14	204.194.238.22	m12.dfw.opendns.com	0.995
15	71.243.0.38	VIS-BLOCK	0.995
16	216.136.82.101	TWTC-NETBLK-7	0.995
17	213.92.5.192	INET-NET	0.995
18	217.20.63.140	NORDIT-NET	0.995
19	62.220.18.18	Versatel Deutschland	0.995
20	209.94.103.253	BESTWEB-BLK-1	0.995

Na tabela 4.5, onde são mostrados os 20 *hosts* que obtiveram a maior probabilidade de estarem infectados por uma aplicação maliciosa de envio de *spam*. Como foi descrito em 4.1, o ambiente utilizado para a experimentação é um ambiente controlado, com diversas políticas de controle de *spam* e isso se reflete nos resultados encontrados, onde nenhuma máquina pertencente ao Instituto de Informática da UFRGS figura na listagem.

Um resultado interessante é que diversos *hosts* que aparecem na listagem são pertencentes à provedores de serviços na Internet, empresas de telefonia e universidades, sendo que alguns deles, como o "209.94.103.253"está associado a uma máquina que está identificada como "spam filter".

Dentre as máquinas listadas, 35% delas figuram em conhecidas *blacklists*[19] de *spammers*.

Analisando a pontuação das *queries*, obtemos os dados apresentados na tabela 4.6.

Apesar do primeiro resultado do *ranking* ser o mais alto, por definição, já que ele corresponde à *signature query* utilizada no experimento, os outros resultados foram bem diferentes do que os apresentados em [ISH 2005], onde a grande maioria das *queries* que figuravam na listagem eram *queries* de tipo MX. Novamente, isso se deve à diferença dos ambientes utilizados, e ao analisar os resultados obtidos percebe-se que condizem com as políticas de segurança descritas em 4.1.

[19] Verificado em: http://www.mxtoolbox.com/blacklists.aspx

Tabela 4.6: Ranking de pontuação das *queries*

Posição	Nome Consultado	Tipo da Consulta	$S'_H(q)$
1	(99)	Unknown	0.996
2	puma.inf.ufrgs.br	TXT	0.980
3	163.5.54.143.in-addr.arpa	PTR	0.961
4	63.5.54.143.in-addr.arpa	PTR	0.948
5	_policy._domainkey.inf.ufrgs.br	TXT	0.945
6	208.5.54.143.in-addr.arpa	PTR	0.945
7	sbmicro.org.br	TXT	0.943
8	230.13.54.143.in-addr.arpa	PTR	0.938
9	inf.ufrgs.br	TXT	0.938
10	177.5.54.143.in-addr.arpa	PTR	0.937
11	12.6.54.143.in-addr.arpa	PTR	0.934
12	71.5.54.143.in-addr.arpa	PTR	0.929
13	149.13.54.143.in-addr.arpa	PTR	0.926
14	_adsp._domainkey.inf.ufrgs.br	TXT	0.926
15	183.5.54.143.in-addr.arpa	PTR	0.923
16	sbmicro.org.br	MX	0.922
17	169.5.54.143.IN-ADDR.ARPA	PTR	0.922
18	148.83.54.143.in-addr.arpa	PTR	0.922
19	cf._dns-sd._udp.128.7.54.143.in-addr.arpa	TXT	0.919
20	226.13.54.143.in-addr.arpa	PTR	0.919
21	www.inf.ufrgs.br	MX	0.919
22	47.6.54.143.in-addr.arpa	PTR	0.916

A maior parte das *queries* obtidas é do tipo PTR, que se deve à checagem do servidor de DNS reverso para que a identidade do servidor possa ser validada. Isso se deve à política de não-aceitação de mensagens oriundas de servidores que não possuam DNS reverso cadastrado. Esta é uma prática bastante comum utilizada atualmente no combate ao *spam*. Com ela, é possível validar a autenticidade do IP de origem, evitando assim que alguém utilize um domínio que não lhe pertence para o envio de e-mails em massa.

Quanto às consultas de tipo TXT que também figuram em grande número na listagem, provavelmente estão relacionadas com a utilização do Sender Policy Framework (SPF). Conforme [SCH 2006], a infra-estrutura de envio de e-mails existente possui a propriedade que define que qualquer *host* que injetar mensagens dentro do sistema de envio de e-mails pode identificar a si próprio com o nome de domínio que quiser. Isso pode ser realizado em uma grande variedade de níveis: em particular na sessão, no envelope e nos *headers* de e-mails. Apesar deste recurso ser desejável em algumas situações, ele é um dos maiores obstáculos na redução de *spam*. Além disso, muitos detentores de nomes de domínio estão bastante preocupados, com razão, com a facilidade com que outras entidades podem fazer uso de seus nomes de domínio, muitas vezes com intenções maliciosas. A RFC 4408 define um protocolo em que proprietários de domínios devem autorizar o uso de seus nomes de domínio a determinados *hosts*, nas identidades "MAIL FROM"ou "HELO". Um benefício adicional para os receptores de mensagens de correio eletrônico é que após o uso de uma identidade ser verificada, decisões sobre a mensagem podem ser tomadas baseadas na política local do nome de domínio do remetente, ao invés de seu endereço IP. Isto é vantajoso porque a reputação de nomes de domínio poderá ser

mais precisa do que a reputação de um determinado endereço IP. Além do mais, se uma identidade reivindicada falhar na verificação, podem ser tomadas medidas mais enérgicas contra tais tipos de mensagens, como por exemplo, rejeitá-las.

Por fim, as consultas de tipo MX que figuram no *ranking* são poucas, mas é interessante observar os nomes de domínios que aparecem junto à elas. Um deles pertence ao próprio Instituto de Informática, enquanto que o outro pertence à Sociedade Brasileira de Microeletrônica, cujo domínio é bastante acessado dentro do ambiente analisado. Existe uma chance alta de que estes domínios tenham sido rastreados por aplicações maliciosas de envio de *spam* e as mesmas estejam buscando pelos *mail-servers* correspondente à eles.

Os resultados mostraram-se bastante satisfatórios, e coerentes com os dados e o ambiente analisado.

5 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho foi desenvolvido com o intuito de estudar maneiras de detectar aplicações maliciosas através da análise de tráfego DNS. Esta detecção acabou tomando um foco mais específico posteriormente, se voltando para o rastreamento de máquinas infectadas por vermes responsáveis pelo envio de e-mails em massa. Metodologias existentes foram pesquisadas e a técnica de detecção baseada no método de inferência Bayesiana proposta por [ISH 2005], com o intuito de detectar *hosts* através de heurísticas e um conhecimento parcial do tráfego de rede realizado por esses vermes foi escolhida para ser analisada e implementada. O motivo desta escolha se deu por este trabalho ser o único conhecido com o foco em detecção de envio de *spam*, além de ser passível de ser reprodutível em um ambiente real para que sua eficiência pudesse ser testada.

Com isso, iniciou-se o a pesquisa para a realização do processo de captura dos dados necessários para o experimento e as ferramentas utilizadas para tal, assim como o projeto e desenvolvimento da aplicação utilizada para executar a metodologia proposta. A implementação consistiu em um conjunto de pequenas aplicações, podendo ser denominadas de *scripts*, que realizavam a separação dos dados necessários para o estudo, assim como a execução da metodologia proposta. Essa aplicação está sob a licença *GNU General Public License v3*[1], e foi hospedada no repositório público de código *Google Code*[2], permitindo assim que qualquer pessoa interessada em contribuir com o projeto tenha acesso ao código desenvolvido. Como ambiente de estudo, foi escolhido o Instituto de Informática da UFRGS.

Após a coleta e o processamento dos dados referentes à um dia útil de tráfego DNS, os resultados obtidos foram bastante satisfatórios, apesar de não mostrarem nenhum problema realmente preocupante no ambiente de estudo. Isso se deve à este ser um ambiente controlado, utilizando diversas técnicas de combate à proliferação de *spam* e execução de atividades maliciosas na rede. No entanto, os resultados obtidos foram bastante coerentes com o objetivo da pesquisa, já que os três tipos de consultas que obtiveram altas pontuações no *ranking* apresentado são relacionadas à uma possível tentativa de descoberta dos servidores de e-mail relacionados ao nome de domínio buscado.

Para a execução da técnica proposta, foi utilizada uma assinatura que indica o comportamento anômalo de máquinas infectadas por *worms*, mas isso não significa que esse método está limitado somente à essa assinatura. Novas assinaturas podem ser descobertas através de outros métodos, como a aplicação de engenharia reversa em vermes como é realizada por fabricantes de aplicativos *anti-virus* nos dias atuais. Assim como também podem ser efetuados testes com diferentes valores de alguns parâmetros utilizados nos

[1] http://www.gnu.org/licenses/gpl.html
[2] http://code.google.com/

cálculos, de forma que o resultado acabe convergindo para algo coerente com o cenário avaliado.

Para trabalhos futuros, é sugerido o estudo de aplicações maliciosas de uma forma mais abrangente, como a detecção de *botnets* através da análise do tráfego DNS, como pode ser visto ser um assunto bastante discutido em trabalhos publicados atualmente, apesar de poucas metodologias para isso existirem.

Também pode ser realizada uma melhoria no método estudado neste trabalho, incorporando as novas metodologias aplicadas atualmente para evitar a disseminação de *spam*, como a validação do IP de origem de uma solicitação através da verificação de DNS reverso, ou até mesmo a utilização do padrão *Sender Policy Framework* para permitir a utilização de nomes de domínio no envio de mensagens em massa.

REFERÊNCIAS

[ANT 2010] ANTISPAMBR. **Antispam.br** :: tipos de spam. 2010. Disponível em: http://www.antispam.br/tipos/. Acesso em: novembro 2010.

[ANT 2010a] ANTISPAMBR. **Antispam.br** :: origem do spam. 2010. Disponível em: http://www.antispam.br/historia/. Acesso em: novembro 2010.

[BRO 2007] BRODSKY, A. **A distributed content independent method for spam detection**. 2007.

[CER 2010] CERT.BR. **Cartilha de segurança – parte viii**: códigos maliciosos (malware). 2010. Disponível em: http://cartilha.cert.br/malware/sec7.html. Acesso em: novembro 2010.

[CET 2010] CETIC. **Cetic.br - painel ibope/netratings**. 2010. Disponível em: http://www.cetic.br/usuarios/ibope/w-tab02-01-2010.htm. Acesso em: dezembro 2010.

[CGI 2010] CGIBR. **Cgi.br - o brasil no cenário do envio de spam**. 2010. Disponível em: http://www.cgi.br/publicacoes/documentacao/spam.htm. Acesso em: novembro 2010.

[CGI 2010a] CGIBR. **Cartilha de segurança - parte viii**: códigos maliciosos (malware). 2010. http://cartilha.cert.br/malware/sec6.html#subsec6.1. Acesso em: novembro 2010.

[CGI 2010b] CGIBR. **Cert.br stats (julho a setembro de 2010)**. 2010. http://www.cert.br/stats/incidentes/2010-jul-sep/tipos-ataque.html. Acesso em: novembro 2010.

[CHO 2007] CHOI, H. et al. Botnet detection by monitoring group activities in dns traffic. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER AND INFORMATION TECHNOLOGY, 7., 2007, Washington, DC, USA. **Proceedings. . .** IEEE Computer Society, 2007. p.715–720.

[CON 2010b] CONFICKER. **Conficker**. 2010. Disponível em: http://pt.wikipedia.org/wiki/Conficker. Acesso em: novembro 2010.

[CON 2010] CONFICKERDNS. **Conficker.c and dns**. 2010. Disponível em: https://blog.honeynet.org.my/?p=84. Acesso em: novembro 2010.

[CON 2010a] CONFICKERVARIANTA. **Conficker**: the other not so famous variant a. 2010. Disponível em: https://blog.honeynet.org.my/?p=64. Acesso em: novembro 2010.

[DEF 2010] DEFINITIONSPAM. **The spamhaus project - the definition of spam**. 2010. Disponível em: http://www.spamhaus.org/definition.html. Acesso em: novembro 2010.

[FLO 2010] FLOSSPROJECT. **Floss project**. 2010. Disponível em: http://www.flossproject.org/. Acesso em: novembro 2010.

[FOS 2010] FOSS. **Free and open source software**. 2010. Disponível em: http://en.wikipedia.org/wiki/Free_and_open_source_software. Acesso em: novembro 2010.

[FSF 2010] FSF. **Free software foundation**. 2010. Disponível em: http://www.fsf.org/. Acesso em: novembro 2010.

[GRA 2010] GRAHAM, P. **A plan for spam**. 2010. http://www.paulgraham.com/spam.html. Acesso em: novembro 2010.

[GSO 2010] GSOC. **Google summer of code**. 2010. Disponível em: http://code.google.com/intl/pt-BR/soc/. Acesso em: novembro 2010.

[HON 2010] HONEYNETPROJECT. **Know your enemy**: tracking botnets | the honeynet project. 2010. Disponível em: http://www.honeynet.org/papers/bots. Acesso em: novembro 2010.

[HON 2010a] HONEYNETPROJECT. **About the honeynet project**. 2010. Disponível em: http://www.honeynet.org/about. Acesso em: novembro 2010.

[IDG 2010] IDGCISCO. **Brasil assume a liderança do spam mundial em 2009, diz cisco**. 2010. Disponível em: http://idgnow.uol.com.br/seguranca/2009/12/08/brasil-assume-a-lideranca-do-spam-mundial-em-2009-diz-cisco/. Acesso em: novembro 2010.

[ISH 2005] ISHIBASHI, K.; TOYONO, T.; TOYAMA, K. Detecting mass-mailing worm infected hosts by mining dns traffic data. In: IN: PROCEEDINGS OF THE SPECIAL INTEREST GROUP ON DATA COMMUNICATIONS (SIGCOMM, 2005. **Anais...** [S.l.: s.n.], 2005.

[JUN 2004] JUNG, J. et al. Fast portscan detection using sequential hypothesis testing. In: IN PROCEEDINGS OF THE IEEE SYMPOSIUM ON SECURITY AND PRIVACY, 2004. **Anais...** [S.l.: s.n.], 2004.

[LOU 2003] LOUIS, G. **Bogofilter calculations**: comparing geometric mean with fisher's method for combining probabilities. 2003. http://www.bgl.nu/bogofilter/fisher.html. Acesso em: novembro 2010.

[MOC 87] MOCKAPETRIS, P. **Domain names - implementation and specification**. 1987. [S.l.]: IETF, 1987. n.1035. (Request for Comments). Updated

by RFCs 1101, 1183, 1348, 1876, 1982, 1995, 1996, 2065, 2136, 2181, 2137, 2308, 2535, 2845, 3425, 3658, 4033, 4034, 4035, 4343, 5936, 5966, RFC 1035 (Standard).

[OPE 2010] OPENSOURCE. **Opensource**. 2010. Disponível em: http://en.wikipedia.org/wiki/Open_source. Acesso em: novembro 2010.

[ROB 2003] ROBINSON, G. A statistical approach to the spam problem. **Linux J.**, Seattle, WA, USA, v.2003, p.3–, March 2003.

[SCH 2006] SCHLITT, M. W. W. **Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1**. 2006. [S.l.]: IETF, 2006. n.4408. (Experimental). RFC 4408.

[SIL 2009] SILVA CARISSIMI JUERGEN ROCHOL, L. Z. G. Alexandre da. **Redes de computadores**. [S.l.]: Bookman, 2009.

[STA 2004] STAN, C. W. et al. A study of mass-mailing worms. In: IN THE 2ND ACM WORKSHOP ON RAPID MALCODE, 2004. **Anais...** ACM Press, 2004. p.1–10.

[SYM 2010] SYMANTEC. **W32.mydoom.a@mm | symantec corp.** 2010. Disponível em: http://securityresponse.symantec.com/avcenter/venc/data/w32.novarg.a@mm.html. Acesso em: novembro 2010.

[SYM 2010a] SYMANTEC. **W32.netsky.p@mm | symantec corp.** 2010. Disponível em: http://securityresponse.symantec.com/avcenter/venc/data/w32.netsky.p@mm.html. Acesso em: novembro 2010.

[TAN 2002] TANENBAUM, A. S. **Computer networks**. [S.l.]: Prentice Hall, 2002.

[TEC 2010] TECHWORLD. **Anonymous uses 30,000 pc strong botnet in wikileaks campaign - techworld.com**. 2010. http://news.techworld.com/security/3252663/anonymous-uses-30000-pc-strong-botnet-in-wikileaks-campaign/?olo=rss Acesso em: dezembro 2010.

[VIL 2009] VILLAMARIN-SALOMON R.; BRUSTOLONI, J. **Identifying botnets using anomaly detection techniques applied to dns traffic**. 2009.

[WHY 2005] WHYTE, D.; KRANAKIS, E.; OORSCHOT, P. C. van. Dns-based detection of scanning worms in an enterprise network. In: IN PROC. OF THE 12TH ANNUAL NETWORK AND DISTRIBUTED SYSTEM SECURITY SYMPOSIUM, 2005. **Anais...** [S.l.: s.n.], 2005. p.181–195.

[WIK 2010] WIKIPEDIA. **Computer worm - wikipedia, the free encyclopedia**. 2010. http://en.wikipedia.org/wiki/Computer_worm. Acesso em: novembro 2010.

[WIK 2010a] WIKIPEDIA. **Storm worm - wikipedia, the free encyclopedia**. 2010. http://en.wikipedia.org/wiki/Storm_Worm Acesso em: novembro 2010.

[WIK 2010b] WIKIPEDIA. **Storm botnet - wikipedia, the free encyclopedia**. 2010. http://en.wikipedia.org/wiki/Storm_botnet Acesso em: novembro 2010.

[WIL 2003] WILLS, C. E.; MIKHAILOV, M.; SHANG, H. Inferring relative popularity of internet applications by actively querying dns caches. In: IN PROCEEDINGS OF THE ACM SIGCOMM INTERNET MEASUREMENT CONFERENCE, 2003. **Anais...** [S.l.: s.n.], 2003. p.78–90.

[WOR 2010] WORLDWIDETRAFFIC. **Beware surfers**: cyberspace is filling up - times online. 2010. Disponível em: http://technology.timesonline.co.uk/tol/news/tech_and_web/the_web/article6169488.ece. Acesso em: dezembro 2010.