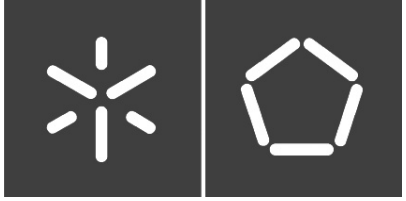
Universidade do Minho
Escola de Engenharia

Pedro Jorge Barros Vasconcelos Guimarães

Rede automatizada de spamtraps

Outubro de 2011



Universidade do Minho
Escola de Engenharia

Pedro Jorge Barros Vasconcelos Guimarães

Rede automatizada de spamtraps

Dissertação de Mestrado
Mestrado em Engenharia Informática

Trabalho efectuado sob a orientação de
Prof. Doutor António Luís Sousa

Outubro de 2011

Declaração

Nome: Pedro Jorge Barros Vasconcelos Guimarães

Endereço Electrónico: pg15398@alunos.uminho.pt

Telefone: 914486608

Bilhete de identidade: 13261663

Título da Tese: Rede automatizada de spamtraps

Orientador: Professor Doutor António Luís Sousa

Ano de conclusão: 2011

Designação do Mestrado: Mestrado em Engenharia Informática

Universidade do Minho, Maio de 2011

Assinatura: _____

Who has time to manually spam web sites? That can't be very cost effective.

Eric Cheng

Agradecimentos

Em primeiro lugar devo agradecer pelos preciosos conhecimentos técnicos que me foram transmitidos pelo Nuno Pais Fernandes ao longo da fase de concepção e desenvolvimento. As suas ideias e sugestões foram de um inestimável valor. Bastará realçar que o projecto não o seria sem a sua participação.

Ao Professor António Luís Sousa que sempre me apontou o caminho indicado em termos académicos, contribuindo com ideias valiosas para o projecto.

À equipa de suporte técnico da Eurotux, em especial ao Duarte Rocha, Paulo Silva e Fernando Gomes, pelo tempo que dedicaram a elucidar-me sobre as mais variadas questões técnicas relacionadas com este projecto.

À administração da Eurotux Informática S.A. pela facilidade e flexibilidade horárias oferecidas, bem como pelas excelentes condições de trabalho.

Ao Manuel Dias que embarcou no mesmo desafio de realizar a dissertação de mestrado na indústria e que sempre me ajudou naquilo que estava ao seu alcance.

À família, amigos e em especial a José Durães e Cláudia Matosa que sempre revelaram grande interesse e curiosidade, contribuindo com as preciosas sugestões.

Resumo

Spam é o termo que define o envio de mensagens não solicitadas por meios electrónicos. A vertente do fenómeno associada ao email é um problema que afecta a Internet desde os seus primórdios. São muitas e sofisticadas as técnicas adoptadas pelos spammers para espalhar mensagens não solicitadas. Apesar disso, os mecanismos anti-spam têm vindo a tornar-se cada vez mais eficazes.

Uma técnica de combate ao spam que se tem tornado cada vez mais popular é a utilização de spamtraps. Tratam-se de normais endereços de email que não são utilizados por humanos, tendo apenas como objectivo a captura de mensagens não solicitadas e por isso classificadas como spam. Estes endereços são colocados em locais estratégicos na Internet para que os spammers os recolham e os utilizem para enviar spam.

Neste trabalho é proposto um novo modo de combater o spam tendo por base a distribuição automática de spamtraps por diversos locais. O projecto de software baseado neste princípio será responsável pela recepção e encaminhamento das mensagens recebidas pelas spamtraps. Estas mensagens de spam, futuramente, serão utilizadas para alimentar um sistema de filtragem de spam já existente, de modo a que este esteja sempre actualizado com os conteúdos mais recentes enviados pelos spammers, bem como das mais diversas informações relativas aos remetentes destas.

Palavras-chave: administração de sistemas, email, segurança, redes, internet, spam, spamtrap

Abstract

Spam is the term that defines the sending of unsolicited messages through electronic means. Since the beginning of the Internet, email spam is considered a serious problem. Nowadays, spammers are using increasingly sophisticated techniques in order to deliver unsolicited mail. Nevertheless, anti-spam mechanisms are becoming progressively more effective.

The use of spamtraps is a technique that is becoming more popular. Spamtraps are unused, but yet valid, email addresses used as traps in order to seize unsolicited email, spam. Such addresses are placed in strategic locations over the Internet so spammers can collect and use them to send spam.

The project based on this approach will be able to disseminate email addresses by various sites and then receive and route the generated messages. These messages will be used to feed an existing content based filtering system in a way that it will continuously learn about the most recent spam contents, as well as information concerning the senders.

Keywords: systems administration, email, security, networks, internet, spam, spamtrap

Conteúdo

Lista de Algoritmos

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Desde o inicio da massificação da Internet que o envio de mensagens indesejáveis ou não solicitadas, spam, se tem vindo a revelar um problema que afecta o regular funcionamento da rede global. Este fenómeno possui diversas vertentes que têm incidência na web, no email, nos *newsgroups*, nos fóruns de conversação e em praticamente todos os locais interactivos e susceptíveis de serem visualizados por humanos. Estas vertentes do spam, apesar de possuírem dinâmicas de funcionamento diferentes, partilham entre si o mesmo objectivo final: o lucro económico. No entanto, o termo "spam"é maioritariamente utilizado no contexto do correio electrónico e é sobre este aspecto que este trabalho se debruça.

São propostas diferentes definições para caracterizar o spam que divergem, essencialmente, no que diz respeito à classificação dos conteúdos das mensagens. Não obstante, as várias definições de spam convergem num ponto essencial - o spam não tem apenas que ver com o conteúdo mas sim com consentimento do destinatário [8]. É nesse sentido que se considera que uma mensagem de email é considerada como spam quando esta se revela não solicitada, sendo que, normalmente, é enviada em conjunto com um grande número de missivas com conteúdos semelhantes e para diferentes endereços [47].

Ao longo dos últimos anos o spam tem-se revelado um fenómeno crescente, utilizando métodos cada vez mais complexos que tentam, a todo o custo, tornar obsoletos os mecanismos que o tentam controlar. Apesar dos sofisticados métodos de combate actualmente existentes e da generalizada hostilidade por partes dos utilizadores da Internet, o fenómeno do spam tende a resistir e a tornar-se cada vez mais forte [55]. Esta resistência relaciona-se com a lucrativa indústria que actua na sua retaguarda e que tem como um dos seus pilares de sustentabilidade o envio de mensagens não solicitadas [40].

Este flagelo será o principal foco deste estudo, que terá como principal

objectivo o de entender as dinâmicas de funcionamento deste fenómeno. Em virtude deste estudo, será apresentado um método que tem como principal objectivo o de aumentar a eficácia de um sistema de filtragem de spam já existente, o ETMX.

1.1 Contexto

O marketing directo é uma prática que já faz parte das sociedades desde o século XIX, altura em que surgiram os primeiros catálogos de produtos enviados por correio. Esta é uma actividade que sempre se revelou aliciante para os vendedores pelo facto de ser possível fazer correlações directas entre o investimento em publicidade e as receitas com as vendas [36].

Assim, com o aparecimento e a massificação da Internet, cedo os *marketers* se aperceberam que estavam perante um terreno fértil para as suas actividades. Neste sentido, rapidamente se começaram a fazer sentir abusos por parte de quem enviava a publicidade. Este tipo de abusos, em pouco tempo, ganhou uma denominação: spam. Consequentemente, e seguindo a lógica das regras sintácticas/semânticas da língua inglesa, os indivíduos ou entidades que praticam actividades relacionadas com spam passaram a ser conhecidos como *spammers*.

Actualmente, a lógica que sustenta os spammers é simples: o custo de enviar um email é praticamente nulo [36] e basta que, em milhões de mensagens, um número reduzido de utilizadores aceda ao que é pedido na missiva para que o pequeno investimento efectuado tenha retorno imediato e com lucros associados. Dado que os *spammers* não publicam relatórios de contabilidade é difícil saber-se ao certo o dinheiro que estes conseguem obter com as campanhas de spam. No entanto, num estudo onde os investigadores se infiltraram numa *botnet*[1] responsável pela maior parte do envio de spam a nível mundial - *storm botnet* - conseguiu-se obter uma estimativa dos custos e lucros marginais destas actividades. Assim, e no caso em que o spam se baseia em marketing simples, concluiu-se que a percentagem de vendas por emails enviados é de 0,00001%, o que significa que se obtém uma venda por cada 100 mil mensagens [36].

Dado que, no estudo referido, cada venda tinha um valor médio de 100$ podemos inferir que, em média, se obtêm receitas de mil dólares por cada milhão de emails enviados. De realçar que, actualmente, a maioria dos *spammers* não controla toda a indústria onde actua, ficando apenas, na generali-

[1]Rede distribuída de máquinas, comprometidas por vírus/trojans, que é utilizada para lançar ataques DDOS (*Distributed Denial Of Service*), enviar spam entre outras funções de cariz ilícito [45].

	Custo total	**N.º de destinatários**	**Custo por destinatário**
Correio normal	$ 9.700	7.000	$ 1,3900
Telemarketing	$ 160	240	$ 0,6600
Panfletos com público-alvo	$ 7.500	100.000	$ 0,0750
Panfletos sem público-alvo	$ 30.000	442.000	$ 0,0670
Fax	$ 30	600	$ 0,0500
Anúncios online	$ 35	1.000	$ 0,0350
Spam	$250	500.000	$ 0,0005

Tabela 1.1: Comparação de custos entre métodos de marketing comuns e spam [9]

.

dade dos casos, responsáveis pelo "marketing". Em troca recebem comissões das vendas que, tipicamente, oscilam entre os 30% e os 50% [40].

O fraco *ratio* de vendas por emails enviados é compensado através do envio de um número cada vez superior de mensagens. Neste sentido, estima-se que a percentagem de email enviado que é considerado spam oscila entre os 88% e os 91%, o que corresponde a cerca de 120 mil milhões de mensagens não solicitadas enviadas diariamente [41, 36]. Este tipo de práticas tem um grande impacto na utilização dos recursos das máquinas, nomeadamente, ao nível da capacidade de processamento, utilização de disco e largura de banda [14]. Consequentemente, as perdas financeiras também se fazem sentir ao nível da factura energética. A ocupação de recursos nos servidores provocada pelo envio de spam torna igualmente mais lenta a entrega de email legítimo (i.e.: *ham*) [61].

Os custos económicos relacionados com o spam são apenas uma parte do problema - a crescente desconfiança nas comunicações digitais e a perda de produtividade das pessoas causam prejuízos difíceis de contabilizar. Por outro lado, o spam é responsável por alimentar uma indústria multimilionária de soluções anti-spam. Esta indústria tem vindo a criar soluções cada vez mais sofisticadas para fazer face aos métodos igualmente cada vez mais complexos utilizados pelos spammers [15], desenvolvendo-se uma espiral de fuga e perseguição que aparenta não ter término.

Os conteúdos enviados pelos spammers diferem na forma e nos objectivos. O mais vulgar é o marketing simples que é o equivalente cibernético aos panfletos não endereçados colocados nas caixas de correio. No entanto, a prática relacionada com o envio de emails que têm como objectivo burlar as vítimas ou mesmo que tentam infectar os seus computadores com software malicioso tem vindo a ganhar cada vez mais espaço. No capítulo das tenta-

tivas de burla é o *phishing*[2] que ultimamente tem merecido maior destaque por parte da comunicação social e das próprias instituições afectadas.

Por estes motivos, as práticas relacionadas com spam estão frequentemente relacionadas com redes criminosas que, apesar de encetarem as suas actividades a partir de um número limitado de países, actuam em todo o Mundo [52]. No entanto, por ser um fenómeno que ocorre no mundo cibernético, muito caracterizado pelo anonimato dos seus utilizadores, a detecção e a punição dos responsáveis torna-se uma tarefa difícil para as autoridades.

1.2 Motivação

É certo que o combate ao spam só se revela eficaz quando se conhece o fenómeno em detalhe. Consequentemente, torna-se essencial captar e analisar grandes quantidades de spam para que se possam extrair importantes informações que revelarão proveniências e tendências dos causadores do problema. Por forma a concretizar essa tarefa, é necessário entender qual o processo mais comum utilizado pelos *spammers* para a recolha dos endereços electrónicos das suas vítimas. Com efeito, estudos sobre o tema apontam para três métodos utilizados com mais frequência [49]:

- Recolha de endereços em vários locais da Internet com o auxílio de software que, de forma automática e metódica, visita um conjunto vasto de páginas web, denominado por *crawler* [11]. O objectivo de um *crawler* pode ser desde o de criar índices de páginas Web (eg.: *Google bot*) até o de procurar endereços de email por forma a criar uma lista. Um *crawler* com este intuito é denominado *harvester* [18].

- Aquisição de listas de endereços;

- Utilização de dicionários de palavras sobre um domínio numa estratégia do tipo *bruteforce*;

Destes métodos, a recolha de endereços com o auxílio de *harvesters* é o principal modo de obtenção de emails [52]. Não obstante, os locais mais comuns que são alvo de buscas por parte dos *harvesters* são páginas web, *blogs*, *newsgroups*, redes sociais e *mailing lists* [61]. Neste sentido, a utilização

[2]Forma recorrente de fraude que, através do envio de emails falsos em nome de entidades confiáveis, tenta persuadir os destinatários a ceder dados pessoais sensíveis. As vítimas mais frequentes deste tipo de crime são os utilizadores mais ingénuos e/ou inexperientes da Internet [17].

de spamtraps (ou *"spam honeypots"*) nos locais enunciados para a obtenção de spam torna-se apetecível por forma a reterem-se quantidades bastante satisfatórias de spam. No entanto, a colocação deste tipo de *honeypots* é frequentemente efectuada com a ajuda de humanos, o que torna o processo potencialmente penoso, demorado e custoso. Assim, e sabendo-se que a distribuição de *honeypots* para efeitos de captação de spam é uma estratégia que já se revelou um sucesso em projectos como o *Project Honeypot*, decidiu-se investigar a viabilidade de se desenvolver um sistema capaz de automatizar este processo.

O mecanismo automático deverá ter em conta os locais mais prováveis para a recolha de endereços por parte dos *harvesters* por forma a maximizar-se a quantidade e variedade do spam a receber. Neste contexto, a utilização de *newsgroups* será vantajosa por, além de serem locais tradicionalmente "varridos"pelos *harvesters*, existirem vários serviços dedicados à sua replicação para páginas Web[3]. Não obstante, a filosofia de funcionamento descentralizado e de constante replicação da *Usenet* permitirá que a difusão das spamtraps se faça de modo mais intenso. Consequentemente, esta reprodução de conteúdos aumentará significativamente as probabilidades de determinada spamtrap ser recolhida por um *harvester*.

Do mesmo modo, destaca-se a opção pela utilização do *twitter* [33] que será vantajosa na medida em que as spamtraps, além de acompanhadas com conteúdos relevantes, poderão ser associadas a *hashtags*. Com base nisto, será de esperar que as spamtraps sejam replicadas através de sites que utilizam *widgets* do *twitter* [32]. Estes têm como função mostrar os conteúdos que estão a ser publicados no *twitter* sobre determinado assunto em cada momento. É comum os *webmasters* adicionarem este tipo de *add-ons* aos seus sites para aumentarem a quantidade (e por vezes, a qualidade) dos conteúdos dos seus sites por forma a que os motores de pesquisa os destaquem nas listas de resultados, optimizando-os para um conjunto restrito de palavras-chave [69].

Após recolha das spamtraps pelos *harvesters*, prevê-se que estas sejam incluídas em listas de emails que, futuramente, poderão ser comercializadas entre os spammers. Por este motivo, e para que o projecto seja bem sucedido, os *spammers* não deverão descobrir o intuito das spamtraps. Se o sigilo não se mantiver, os *spammers* poderão usufruir das fragilidades inerentes à utilização de spamtraps, inscrevendo-as em *newsletters* legítimas ou enviando conteúdo "envenenado"(i.e.: conteúdo que não é spam e que pode ser semelhante ao enviado por entidades legítimas) comprometendo os resultados e os objectivos finais do projecto. No entanto, se o objectivo das spamtraps

[3]Exemplo: *Google Groups* (`http://groups.google.com`)

nunca for descoberto, os *spammers* nunca terão forma de saber se estarão a enviar spam para utilizadores comuns ou para *honeypots*.

A informação disponibilizada pelas spamtraps poderá ser utilizada de inúmeras formas e com diferentes objectivos. Numa primeira fase as informações serão agrupadas e comparadas, criando-se relatórios detalhados para efeitos de reconhecimento do fenómeno. Posteriormente, o corpo e os cabeçalhos das mensagens de spam poderão ser utilizados para alimentar filtros anti-spam baseados em conteúdos. Os endereços IP presentes nos cabeçalhos poderão ser utilizados para enriquecer *blacklists*, utilizadas pelos servidores de email por forma a não permitirem que determinada mensagem, enviada por determinado endereço IP, chegue ao utilizador final.

1.3 Objectivos

Este trabalho tem como objectivo principal a concepção de um sistema de software capaz de colocar spamtraps em vários locais da Internet de forma automática e numa escala de dimensões consideráveis. Os locais a utilizar serão diversos *newsgroups*, fóruns de conversação, websites e *twitter*.

Para que as spamtraps se mantenham disfarçadas e integradas no meio onde são colocadas, juntar-se-ão a estas conteúdos inócuos - i.e.: conteúdos com o mínimo de relevância mas que não suscitem grande interesse de réplica em humanos. Desta forma, evitar-se-á que a colocação das spamtraps seja considerada spam e que humanos as utilizem para enviar correio electrónico legítimo (i.e.: *ham*). Este objectivo deverá ser alcançado através da colocação de conteúdos contextualizados de alguma forma com cada local - eg.: spamtraps acompanhadas de anedotas deverão ser colocadas em grupos como *alt.jokes* ou *alt.misc.funny*.

A frequência da colocação das spamtraps estará directamente relacionada com a sensibilidade do local. Neste sentido, em determinado momento, um fórum nunca receberá a mesma quantidade de spamtraps que receberá uma conta no *twitter* ou um website sob controlo do sistema. Inicialmente deverá observar-se cuidadosamente os locais que receberam spamtraps no sentido de se verificar a indiferença desejada por parte de eventuais utilizadores humanos. Para além destes aspectos, o sistema deverá funcionar de forma discreta para que nunca se levante suspeitas quanto ao intuito das spamtraps.

De forma a prever os métodos *bruteforce* baseados em dicionários e utilizados pelos spammers, configurar-se-á o servidor de email para aceitar qualquer email dirigido aos domínios que tiver sob controlo. As mensagens dirigidas a um destinatário não existente serão colocadas numa caixa de correio especial criada em cada domínio, *catch-all*. Estas mensagens serão alvo de

uma análise extra de modo a ser possível inferir quais os destinatários mais comuns baseados em dicionários.

Numa fase inicial, o sistema, na sua totalidade, deverá ficar alojado numa máquina preparada especialmente para este efeito. Apesar de se pretender uma total autonomia do sistema, é desejável que seja possível controlar as funcionalidades deste remotamente através de um sistema de *webservices*. Apenas desta forma será possível aumentar o grau de autonomia do sistema, fazendo-se pequenos ajustes à frequência e locais de colocação das spamtraps conforme a sensibilidade destes.

No final deste projecto deverá ser possível perceber quem costuma enviar mensagens não solicitadas e quais os conteúdos mais comuns. A partir dos dados recebidos espera-se identificar quais os IPs e quais os países onde a prática do envio de spam é mais comum. Para além de efeitos de estudo, é desejável que o spam seja submetido a um sistema de filtragem por conteúdo, de modo a que este se torne progressivamente mais eficaz. Desta forma, pretende-se que a rede automática de spamtraps esteja constantemente em funcionamento por forma a alimentar, continuamente, os mecanismos anti-spam que se desejam permanentemente actualizados.

Pretende-se com este projecto complementar um produto anti-spam, desenvolvido pela empresa Eurotux Informática S.A., denominado ETMX [20]. O passo seguinte, em caso de sucesso do projecto, será o de aumentar a escala em termos de recursos, domínios e de locais de distribuição de spamtraps.

1.4 Estrutura do documento

O capítulo 2 descreve as dinâmicas de funcionamento do spam tendo por base diversos trabalhos publicados sobre o tema. Ainda neste capítulo, são descritos os métodos actuais de mitigação do fenómeno do spam.

No capítulo 3 descreve-se detalhadamente a arquitectura de software, revelando-se o modo de concepção e de funcionamento do sistema desenvolvido.

O capítulo 4 detalha as características da efectiva distribuição das spamtraps pelos mais diversos locais.

No capítulo 5 revelam-se os resultados obtidos em relação ao spam recebido, quer através das spamtraps quer através de outros endereços.

O capítulo 6 descreve as diversas formas de aplicação do sistema concebido, segundo perspectivas comerciais e académicas. Neste capítulo são igualmente apresentadas as conclusões sobre os resultados obtidos, comparando-se estes com as expectativas iniciais.

Capítulo 2

Spam, spamtraps e factos relacionados

Desde o início da massificação da Internet que os spammers se têm mantido frequentemente um passo à frente em relação aos métodos e tecnologias utilizadas pela indústria anti-spam. É por este motivo que, apesar dos inúmeros filtros e restrições por onde passa um email antes de chegar ao seu destinatário, se continua a receber spam nas caixas de correio electrónico de utilizadores comuns. Neste capítulo são descritos os métodos mais comuns utilizados pelos spammers, bem como as melhores práticas e tecnologias existentes para deter as suas missivas.

No contexto do projecto associado a este estudo, é descrito o modo de funcionamento de algumas redes de spamtraps actualmente existentes. Neste sentido, são feitas comparações entre estas e a rede de spamtraps que se pretende desenvolver.

2.1 Métodos de spam

Os métodos utilizados pelos *spammers* têm vindo a sofrer mutações ao longo do tempo por forma a continuarem com as suas práticas lucrativas [15]. No entanto, a lógica que sustenta os envios de email em massa continua a mesma desde o início do fenómeno, sendo que o processo de envio de spam pode ser desdobrado em três fases:

1. **Aquisição de endereços de email;**
2. **Preparação de conteúdos;**
3. **Envio em massa.**

As estratégias anti-spam devem passar pelo combate nestas três frentes, em simultâneo. No entanto, para este efeito, é necessário conhecer em detalhe as estratégias utilizadas em cada uma destas etapas.

2.1.1 Aquisição de endereços

O método mais comum para a aquisição de endereços de correio electrónico continua a ser através de *harvesting* [18]. O *harvesting* caracteriza-se pela procura de endereços em vários locais da Internet utilizando, para esse efeito, *crawlers*. Frequentemente, estes *crawlers* são apenas pequenos pedaços de software rudimentar que somente processam conteúdos de texto, com a finalidade de verificar se alguma parte destes combina com uma expressão regular simples [52, 18]. Por este motivo, é fácil evitar a recolha de um endereço de email colocado na Internet, bastando para isso aplicar algumas regras simples que se encontram descritas na secção seguinte.

Segundo estatísticas publicadas pelo *Project Honeypot*, é em cinco países que se efectuam mais de 50% do total de recolhas de endereços de email, conforme detalhado na Tabela 2.1.

Posição	**País**	**Percentagem**
1	Espanha	15,9
2	China	14,6
3	Estados Unidos	11,3
4	Roménia	6,0
5	Alemanha	4,5
6	Países Baixos	3,8
7	Reino Unido	3,5
8	Itália	3,2
9	Costa do Marfim	3,2
10	França	3,0
11	Senegal	2,5
12	Canadá	2,4
13	Brasil	2,2
14	Japão	2,1
15	Malásia	1,6

Tabela 2.1: Países de origem de práticas de *harvesting* [65]

No entanto, quando se analisa os dados mais recentes relativos aos países onde a incidência de envio de spam é maior, verifica-se que os países com

Posição	País	Percentagem
1	China	10,0
2	Brasil	9,0
3	Estados Unidos	7,2
4	Alemanha	6,5
5	Rússia	6,0
6	Turquia	5,4
7	Índia	4,7
8	Itália	4,2
9	Reino Unido	3,8
10	Coreia do Sul	3,8
11	Polónia	3,3
12	Espanha	3,2
13	Vietname	2,4
14	Argentina	2,3
15	Colômbia	2,1

Tabela 2.2: Países com maior incidência de envio de spam [66]

mais actividade a nível de *harvesting* não são necessariamente aqueles que enviam mais spam, Tabela 2.2. Neste sentido, acredita-se que o ciclo do spam não é sempre protagonizado pelos mesmos indivíduos. Assim, os *harvesters* colocam-se numa posição de topo na cadeia do spam, compilando e vendendo listas de endereços que são, posteriormente, adquiridas por outros *spammers* [34, 52].

Actualmente, é possível encontrar vários *websites* dedicados à comercialização de listas que praticam preços que variam conforme a quantidade e o nicho de mercado que se pretende explorar [63]. Estas listas têm origem, na sua maioria, em práticas de *harvesting*, no entanto, existem vários modos de obter endereços de email válidos, como o ataque através de dicionários de palavras. Esta técnica consiste em "adivinhar"endereços de email, gerando automaticamente, a partir de dicionários de palavras e de algarismos aleatórios, nomes e combinações frequentemente utilizados para definir *usernames* [27]. Para verificarem a validade de um dado endereço revelado por via desta técnica, os *spammers*, frequentemente, enviam simplesmente um email em branco e assumem a validade do email se não for devolvido nenhum erro por parte dos servidores de email [27, 31]. As probabilidade de sucesso são directamente proporcionais à quantidade de combinações e à qualidade dos dicionários utilizados.

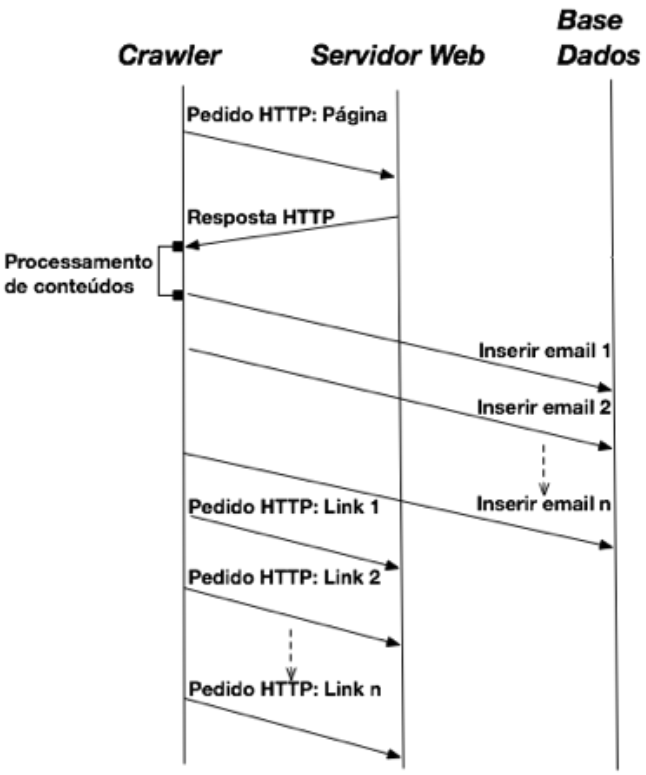


Figura 2.1: Processo típico de *harvesting* em páginas Web [5]

As listas de endereços obtidas por esta via são frequentemente valorizadas por, muitas vezes, possuírem emails para os quais nunca tinha sido enviado qualquer tipo de spam [27]. Desta forma, acredita-se que os seus utilizadores poderão ser mais susceptíveis de aceder ao que é publicitado nas missivas por não estarem familiarizados com o fenómeno do spam.

No entanto, os endereços de email podem ser obtidos de forma ética através da promoção de *newsletters* ou através de registos obrigatórios em sites que fornecem determinados serviços. Naturalmente, os utilizadores esperam que os seus dados, nomeadamente os endereços de email, sejam protegidos por essas entidades. Apesar disso, é possível que estas empresas ou entidades legítimas cedam listas de endereços em troca de benefícios monetários.

Métodos menos comuns passam pelo ataque a bases de dados, obtendo endereços de email e outras informações relevantes dos utilizadores de determinada entidade ou serviço [31]. Verifica-se igualmente a utilização de *trojans* que "varrem"os computadores das vítimas por forma a encontrarem endereços de email que são posteriormente enviados aos *spammers*.

2.1.2 Preparação de conteúdos

Actualmente, um dos principais métodos de combate ao spam passa pela filtragem de conteúdos com base nas frequências de palavras mais utilizadas nas missivas de spam [14]. De forma a contornar este mecanismo, os spammers frequentemente limitam-se a aplicar pequenos truques no texto para disfarçar determinadas palavras. Assim, palavras como "Spam"são escritas sob a forma "S_P_A_M", tornando-se perfeitamente perceptíveis por humanos mas que poderão escapar aos filtros anti-spam menos sofisticados. Além das reescritas destas palavras, os spammers podem também usufruir da flexibilidade do HTML (*HyperText Markup Language*), ofuscando o texto-fonte de forma mais agressiva e utilizando folhas de estilo para colocar a mensagem perceptível para os humanos [29, 28].

Ainda no campo da ofuscação de texto, é de referir a forte popularidade da utilização de imagens ao invés de texto. Através desta prática relativamente simples, os spammers conseguem iludir uma boa parte dos filtros anti-spam baseados em conteúdo [68]. Como resposta às implementações OCR (*Optical Character Recognition*) e de algoritmos de comparação de imagens nos filtros anti-spam, os spammers começaram a aplicar técnicas CAPTCHA (*Completely Automated Public Turing test to tell Computers and Humans Apart*). Estas consistem em distorcer e adicionar ruído a imagens por forma a que apenas humanos sejam capazes de percepcionar o que está implícito na imagem [68]. Além deste facto, os programas de reconhecimento de imagens necessitam de recursos computacionais significativos, o que coloca problemas de eficiência ao nível dos servidores de email [28].

De referir um mecanismo de criação de conteúdos descrito em [39] que utiliza um sofisticado sistema de *templates*, por forma a que cada mensagem gerada seja diferente de todas as outras, sem que a mensagem final seja fortemente distorcida. Em cada *template* apenas são definidos os locais das variáveis de texto e imagem, sendo a atribuição de valores a estas efectuada de forma completamente aleatória a partir de dicionários de palavras minimamente relevantes no contexto da mensagem [50]. Este tipo de sofisticação, apenas ao alcance de *spammers* com acesso a *botnets*, é frequentemente eficaz na fuga aos filtros de spam.

Enquanto que o formato das mensagens de spam varia conforme os avanços nas técnicas anti-spam, o conteúdo está dependente da indústria que suporta o spam. Por este motivo, o conteúdo destas mensagens varia consoante as campanhas de spam activas em cada momento, frequentemente associadas a eventos altamente difundidos pelos meios de comunicação social (eg.: morte de artistas conhecidos, desastres naturais, entre outros). Outros temas populares, relacionam-se com produtos farmacêuticos ou com ofertas de trabalho

[39]. Não obstante, em [3] conclui-se que uma grande percentagem de emails de spam não são mais do que burlas. Ainda neste estudo, conclui-se que cerca de 30% das mensagens aliciam as vítimas a visitar sites externos, através da colocação de URLs (*Uniform Resource Locator*) nos seus conteúdos.

Um nicho de spam que tem vindo a ganhar popularidade entre os *spammers* nos últimos anos é o *stock spam*. Trata-se de um tipo de spam que actua no âmbito dos mercados bolsistas e que tenta beneficiar da ingenuidade ou da irracionalidade das decisões dos seus destinatários. As mensagens enviadas neste contexto tendem a lançar especulações sobre valorizações/desvalorizações que poderão ocorrer sobre determinadas acções. Neste caso, os spammers obtêm rendimentos conforme a sua participação nas empresas sobre as quais lançam especulações. Frequentemente, os emails são escritos de forma a levarem o destinatário a crer que está perante uma informação privilegiada, enviada por engano para o seu endereço [9].

2.1.3 Envio de spam

O nível de sofisticação dos métodos de envio de spam tem vindo a crescer vertiginosamente nos últimos anos. Inicialmente, a distribuição de spam tinha por base a utilização de *open-relays*, *open proxies* ou mesmo clientes individuais. No entanto, com a evolução dos mecanismos de filtragem com base nas informações de rede, a utilização destes recursos foi preterida em relação às *botnets*. Uma *botnet* é uma rede em larga escala de máquinas comprometidas com software malicioso, denominadas *zombies*, controladas por um número limitado de servidores C&C (*Command and Control*). Estimativas recentes sugerem que existem mais de um milhão de computadores comprometidos com *botnets*, sendo os sistemas *Windows* os mais vulneráveis aos *trojans* [60].

Em [56], numa pesquisa que incidiu sobre os sistemas operativos dos clientes que enviaram spam no contexto da investigação, concluiu-se que era o *Windows* que prevalecia neste campo, Tabela 2.3.

Inicialmente, as *botnets* eram comandadas através de simples comandos enviados através de IRC (*Internet Relay Chat*) mas, com o apertar das tecnologias anti-spam, as ordens passaram a ser enviadas através de protocolos mais elaborados, recorrendo frequentemente a aplicações de cifras [35]. É comum que estas redes cresçam através da auto-propagação com recurso a emails de spam, engenharia social ou através da exploração de vulnerabilidades de determinados sistemas operativos [35]. Estas rede são comandadas por indivíduos que as utilizam para vários fins, sendo os mais comuns os que se relacionam com ataques DDoS, envio de spam ou com a própria propagação.

Em relação ao envio de spam por *botnets*, estudos revelam que, já no ano

Sistema Operativo	N.º de clientes	Quantidade de spam
Windows (Total)	854.404 (70,0%)	5.863.112 (58,0%)
W̃indows (2000 ou XP)	604.252 (49,0%)	4.060.290 (40,2%)
W̃indows 98	13.727 (1,1%)	54.856 (0,5%)
W̃indows 95	559 (0,1%)	2.797 (0,1%)
W̃indows (outros)	235.866 (19,0%)	1.745.169 (17,2%)
Linux	28.132 (2,3%)	557.377 (5,5%)
FreeBSD	6.584 (0,5%)	152.456 (1,5%)
MacOS	2.944 (0,2%)	46.151 (0,4%)

Tabela 2.3: Principais sistemas operativos de *relays* de spam [56]

de 2004, 70% de todo o spam era enviado através destas redes [60], sendo que, actualmente, apenas uma pequena fracção não é enviada através destes meios [50]. Tipicamente, os spammers não são os operadores destas redes e apenas fazem uso delas quando necessitam de enviar spam. O acesso a estas *botnets* é normalmente concedido em regime de aluguer, o que torna ainda mais difícil a sua mitigação [61, 45].

O envio de spam através de *botnets* possibilita enganar os mecanismos anti-spam baseados em informações de rede [35]. Neste sentido, um *spammer* que distribua mensagens através de uma *botnet* não vê, facilmente, a sua actividade impedida por estes filtros, dado que, estará a enviar spam a partir de diferentes IPs, de diferentes ISPs (*Internet Service Providers*), distribuídos por vários países. A utilização destas redes aumenta exponencialmente a capacidade de envio de spam por unidade de tempo, dado que uma botnet tem à sua disposição uma quantidade muito superior de largura de banda do que um servidor comum.

Apesar da desvantagem em relação aos métodos de filtragem baseados em *blacklists*, os *spammers* que não têm acesso a *botnets* recorrem frequentemente a *open relays*. Estes consistem em servidores de envio de email que, por estarem mal configurados, por opção ou ingenuidade do administrador do sistema ou por possuírem software malicioso instalado, aceitam receber e reencaminhar emails que não dirigidos para si ou para a sua infraestrutura [27]. Trata-se de um modo utilizado pelos *spammers* para ofuscarem o local a partir do qual desencadeiam as suas actividades e de não serem penalizados pelo seu próprio ISP, dado que, o envio de spam configura, frequentemente, uma violação dos termos de contrato do serviço de acesso à Internet. Por este motivo, é frequente que os *spammers* efectuem uma procura automática e agressiva por *open relays* [61].

Destaca-se igualmente outro método popular de distribuição de spam que se baseia em *open-proxies*. Estes *proxies* são normalmente utilizados para servirem de intermediários de pedidos HTTP (*Hypertext Transfer Protocol*), recebendo os pedidos do cliente e encaminhado-os para o servidor como se fossem seus, efectuando um processo análogo no processamento das respostas. Utilizam-se quando, por algum motivo, existirem *firewalls* a bloquear o acesso a determinados sítios ou serviços na Web. No entanto, *open proxies* mal configurados, podem facilmente ser utilizados para enviar spam. Ao contrário dos *open relays*, o spam é entregue ao *proxy* pela própria porta HTTP através da utilização do comando "HTTP CONNECT"[27]. Tipicamente utilizado para estabelecer conexões HTTPS (*Hypertext Transfer Protocol Secure*) através de HTTP, este comando pode permitir ao atacante ordenar ao *proxy* que estabeleça uma conexão TCP (*Transmission Control Protocol*) entre o cliente (e eventual *spammer*) e uma qualquer porta de um qualquer servidor. Por representar uma falha de segurança muito grave, é comum os *proxies* terem a opção de utilização do "HTTP CONNECT"altamente restringida.

Tal como acontece com os *zombies* das *botnets*, é provável que uma grande parte dos *open relays* e dos *open proxies* existentes não tenham intenção de o ser. Esta situação é causada pela crescente propagação de software malicioso que tem a capacidade de transformar simples clientes de acesso à Internet em máquinas de envio de spam. No entanto, estes dois métodos são muito mais susceptíveis de serem travados através de filtros anti-spam baseados nas informações de rede.

2.2 Mitigação

Os métodos anti-spam têm vindo a tornar-se cada vez mais eficazes e complexos com o passar dos anos. Ao longo deste tempo foram publicados inúmeros estudos e investigações neste sentido. Desenvolveram-se diversos sistemas baseados nestas propostas, no entanto, apenas vingaram alguns: os que melhor se adaptavam às dinâmicas de transformação dos *spammers* e os que eram mais viáveis de implementar em larga escala.

Actualmente, o combate ao spam é sustentado nos seguintes três paradigmas:

- **Prevenção de recolha de endereços;**
- **Filtragem baseada em conteúdos;**
- **Filtragem não-baseada em conteúdos.**

A prevenção é uma forma de combate desconexa das restantes por implicar a sensibilização dos utilizadores de email, bem como a dos administradores de sistemas dos vários pontos de recolha de endereços de email. As aproximações baseadas em filtros diferenciam-se por serem reactivas, podendo ser combinadas por forma a aumentarem as taxas de eficácia na retenção de spam.

2.2.1 Prevenção de recolha de endereços

É axiomático afirmar-se que a melhor maneira de impedir que determinado endereço de email receba spam é evitando a sua propagação. Neste sentido, deve-se evitar a sua publicação em qualquer local de fácil acesso aos *harvesters*. No entanto, se a divulgação for inevitável, existem algumas boas práticas que se poderão ter em conta:

- **Disfarçar o endereço;**

  Dado o perfil rudimentar da maioria dos *harvesting crawlers*, é possível evitar que um endereço seja recolhido fazendo apenas algumas alterações aos caracteres que o compõem. Em [18] conclui-se que o simples acto de substituir todos os caracteres de um endereço pelos códigos equivalentes em HTML[1], reduzia para praticamente zero a probabilidade de receber spam. É igualmente eficaz substituir alguns caracteres por símbolos ou palavras facilmente perceptíveis por humanos[2]. Dada a eficácia destes métodos, em [18] é proposto um modo de automatizar este processo ao nível do servidor HTTP.

- **Publicar o endereço através de uma imagem;**

  Dada a grande quantidade de recursos computacionais que as técnicas OCR necessitam, é muito improvável que *harvesters* tentem interpretar as imagens dos locais que visitam em busca de endereços de email. Além desse facto, a OCR não é uma técnica completamente fiável pelo que os endereços descodificados poderiam não corresponder aos que estão explicitados nas imagens [18]. A desvantagem deste método relaciona-se com o facto do utilizador comum não poder seleccionar e copiar o texto directamente, sendo necessário efectuar uma transcrição manual.

[1] Exemplo: O carácter 'a' é substituído pelo código '&#97'; que é correctamente interpretado pelo *browser* mas não pelos *harvesters* comuns.

[2] Exemplo: "foo@example.com"passa a "foo 'arroba' example 'ponto' com.

- **Utilização de endereços descartáveis;**

  Trata-se de uma técnica que deve ser utilizada sempre que, para obter algum serviço ou informação, seja obrigatório fornecer um endereço de email a entidades que não inspirem total confiança ao utilizador. Utiliza-se quando se pretende receber alguns emails num curto espaço de tempo (exemplo: registo e obtenção de credenciais). Existem vários locais na Internet que fornecem este tipo de serviços gratuitamente[3], pelo que é uma técnica que pode ser usada pelo utilizador comum.

- **Evitar a utilização de nomes comuns.**

  Quando se cria um endereço de email, deverá ser dada atenção ao nome de utilizador para que este não seja facilmente "adivinhado"pelos *spammers* que recorrem a dicionários de palavras. Neste sentido, considera-se boa prática juntar ao nome caracteres menos comuns e números (eg.: foo1213@example.com).

Na área de influência dos administradores de sistemas dos vários sítios na Internet, é possível fazer um combate preventivo com base nas informações fornecidas sobre os *harvesters* por projectos como o *Project Honeypot*. Através da utilização de informações permanentemente actualizadas, é possível bloquear à partida a acção dos *harvesters*. Os detalhes utilizados para barrar estes *bots* poderão ser o próprio IP ou o *User Agent* [19].

Alternativamente, em [19] é sugerido que se poderá mitigar a actividade dos *harvesters* criando *websites* fictícios com *links* dinâmicos para eles próprios num esquema de ciclo infinito. Através desta solução, pretende-se que os *harvesters* fiquem "aprisionados"e que os responsáveis por estas actividades fiquem com uma quantidade substancial de recursos computacionais ocupados, diminuindo, na generalidade, a sua eficiência. Ao mesmo tempo, pretende-se que as bases de dados de endereços de email recolhidos pelos *harvesters* sejam "envenenadas"com milhares de emails falsos. Neste sentido, foi desenvolvido um *script* em formato CGI (*Common Gateway Interface*) denominado *Wpoison* [44] que poderá ser facilmente instalado por qualquer administrador de qualquer *website*.

Por fim, existem propostas para a criação de leis que impeçam e punam a actividade de recolha ilegal de endereços de email. Em [6] propõe-se que o administrador de qualquer *site* tenha ao seu alcance um mecanismo legal que proteja o seu site de actividades de *harvesting*. Neste sentido, bastaria incluir uma norma, denominada *"no-email-collection"*, nos termos e condições de utilização do site e uma referência numa *Meta Tag* para leitura automática

[3]Exemplo: `http://10minutemail.com`

dos *crawlers*. De forma a punir quem não respeitasse a norma, seriam sempre incluídas spamtraps únicas para que as autoridades pudessem seguir o rasto dos *harvesters*.

2.2.2 Filtragem baseada em conteúdos

O combate ao spam tem na filtragem de conteúdos um dos maiores aliados, sendo uma das técnicas mais populares implementadas pelos administradores de servidores de email. Estes filtros funcionam com base na comparação do assunto e do corpo dos emails com listas de palavras ou frases que se sabem ser comuns em mensagens de spam. Para este efeito, verifica-se a frequência de certas palavras na mensagem e comparam-se os conteúdos com outros já classificados como spam [27].

No entanto, trata-se de uma solução que necessita de ser constantemente avaliada no contexto em que está inserida [19]. Neste sentido, o servidor de recepção de email de uma empresa farmacêutica não deverá reter mensagens com palavras relacionadas com medicamentos, mesmo que estas sejam bastante comuns em mensagens de spam. Por este motivo, foram implementados mecanismos que permitem que os operadores dos servidores atribuam um determinado "peso" a cada palavra/expressão que possa ser considerada spam [19]. Esta informação é introduzida no software de filtragem sob a forma de regras estáticas que deverão ser correctamente ajustadas, i.e., não se pode introduzir uma regra demasiado generalista nem demasiado específica sob pena de se comprometer a eficácia da filtragem.

Apesar dos esforços para manter esta abordagem eficaz, os *spammers* têm vindo a conseguir iludir os filtros baseados em listas de palavras/expressões através da introdução de combinações aleatórias de palavras e de erros ortográficos propositados [55]. Adicionalmente, estes filtros não possuem capacidade para reter o spam baseado em imagens [8]. Para colmatar essa lacuna, existem várias técnicas em estudo que permitem inferir se determinada imagem é susceptível de ser considerada spam, recolhendo-se várias características da mesma sem recorrer à sua interpretação total via OCR [8].

Por forma a contornar os problemas relacionados com os filtros estáticos, é recorrente combinar-se estas técnicas com algoritmos de auto-aprendizagem com base em estatística - eg.: *Naive Bayes*, SVM (*Support Vector Machines*) e *LogitBoost*) [14, 55]. A utilização de técnicas de auto-aprendizagem permite concluir, com elevado grau de eficácia, se determinada mensagem é spam, mesmo que o sistema nunca tenha processado o conteúdo da mesma.

Neste campo, o software *open-source SpamAssassin* [25] é um dos mais populares, combinando os melhores métodos de filtragem de conteúdos. Para esse efeito, aplica vários filtros baseados em expressões regulares e uma imple-

mentação do modelo probabilístico *Naive Bayes*[4], previamente enriquecida com mensagens de spam e *ham* [57]. A acção deste software é complementada com comparações das informações de rede dos servidores de envio com *blacklists* públicas.

Tipicamente, este tipo de sistemas de filtragem produzem apenas classificações conforme a probabilidade de determinada mensagem ser ou não spam. Desta forma, é possível que o administrador de sistemas ajuste as acções a tomar conforme a classificação dada pelo sistema de filtragem [27]. Assim, uma mensagem com probabilidade de ser spam de 90% pode ser imediatamente apagada; uma mensagem com probabilidade de ser spam de 40% pode ser colocada numa pasta especial para que o utilizador ou o operador possam decidir mais tarde sobre a acção a tomar.

A eficiência de sistemas baseados no modelo *Naive Bayes* está directamente relacionada com a quantidade e variedade de mensagens submetidas para aprendizagem [27]. Por este motivo, é possível obterem-se taxas de eficácia na detecção de spam na ordem dos 99%, no entanto, é necessário submeter milhares, idealmente milhões, de mensagens previamente classificadas como spam ou *ham*. Este processo é potencialmente moroso, penoso e custoso para poder ser efectuado apenas por humanos, sendo este um dos principais pontos negativos desta abordagem.

2.2.3 Filtragem não-baseada em conteúdos

A filtragem não-baseada em conteúdos engloba todas as técnicas que não recorrem à análise directa daquilo que está contido no assunto e no corpo da mensagem. Neste âmbito, as filtragens com base em *blacklists*, *graylists* e *whitelists* continuam a ser as mais populares. As *blacklists* reúnem informações, nomeadamente IP e respectivo país de origem, de *relays* que se sabem ser responsáveis por envio de spam. De modo oposto, as *whitelists* revelam endereços IP de *relays* que não enviam spam e que muito dificilmente algum dia enviarão.

As *graylists* são listas criadas pelos MTAs (*Mail Transfer Agents*) que servem para colocar IPs de *relays* e respectivas mensagens em quarentena. Estes são adicionados a *graylists* quando não se conhece o IP e por esse motivo existirem dúvidas sobre se este poderá estar a enviar spam ou não. Neste caso, os MTAs utilizam uma técnica denominada *Tempfailing* que assume que, caso o email recebido seja legítimo, o servidor de envio tentará enviar novamente o email num intervalo de tempo na ordem dos minutos, horas ou

[4]Algoritmo baseado no modelo probabilístico de Bayes que não considera a possível relação entre as características de determinada amostra no cálculo da sua classe [58].

dias. Caso seja spam, não deverão ocorrer novas tentativas pelo facto de os spammers não preverem essa regra nos seus sistemas de envio de email [31, 71]. A desvantagem da utilização das *graylists* está relacionada com o atraso que provoca na entrega de mensagens legítimas; desde alguns minutos a vários dias.

Actualmente existem diversos serviços *online* que fornecem informações permanentemente actualizadas sob a forma de *blacklists*[5]. Através da informação fornecida pelas DNSBLs (*DNS-based Blackhole Lists*), os MTAs podem reter imediatamente um email que foi enviado a partir de um *relay* com IP suspeito, salvaguardando desta forma uma série de recursos computacionais que seriam necessários para efectuar uma filtragem por conteúdo. A informação necessária para a comparação de informações com listas públicas é extraída dos cabeçalhos "*Received*"do email. Trata-se de um cabeçalho obrigatório que é adicionado pelos próprios *gateways* [38]. Desta forma, o destinatário final tem acesso ao caminho percorrido por determinado email - o *SMTP (Simple Mail Transfer Protocol) Path*.

```
Received: from mout9.freenet.de (mout9.freenet.de [195.4.92.99])
    (using TLSv1 with cipher DHE-RSA-AES256-SHA (256/256 bits))
    (No client certificate requested)
    by trp.eurotux.com (Postfix) with ESMTP id A8888890EC6
    for <news-h@linhalinux.com>; Sun, 24 Jul 2011 21:26:09 +0400 (MSD)
Received: from [195.4.92.14] (helo=4.mx.freenet.de)
    by mout9.freenet.de with esmtpa (ID opyuhghtg@freenet.de) (port 25) (Exim 4.76 #5)
    id 1Ql2R1-0001Z8-5K; Sun, 24 Jul 2011 19:25:47 +0200
Received: from 184-106-79-169.static.cloud-ips.com ([184.106.79.169]:51122 helo=User)
    by 4.mx.freenet.de with esmtpa (ID opyuhghtg@freenet.de) (port 587) (Exim 4.72 #5)
    id 1Ql2R0-000148-L0; Sun, 24 Jul 2011 19:25:46 +0200
```

Figura 2.2: Cabeçalhos *Received* de um email considerado spam, recebido no âmbito do projecto *Automated Spamtrap Network*

Ainda no âmbito da filtragem por listas, existem MTAs que simplesmente bloqueiam ou colocam em quarentena todos os emails enviados a partir de gamas de IPs de determinados países que, estatisticamente, se sabe que enviam grandes quantidades de spam. Apesar de polémica, esta é uma medida que tem vindo a revelar-se eficaz [31]. Pelas vantagens decorrentes da utilização de listas, podemos considerar este tipo de filtragem como uma primeira linha de combate ao spam e, embora a sua eficácia seja comprometida pela emergência das *botnets* [31], assume um papel bastante relevante em combinação com outros sistemas.

[5]Exemplos:
Barracuda (`http://www.barracudacentral.org`) e
Spamhaus (`http://www.spamhaus.org/sbl`)

Um dos mecanismos que tem vindo a ganhar popularidade é o que corresponde a acções colaborativas. Estas consistem na utilização da informação fornecida pelos utilizadores acerca da caracterização de determinada mensagem como spam ou *ham*. Nos mecanismos colaborativos, o modo mais comum de obter essa informação é através da utilização de redes P2P ou do próprio *feedback* directo dos utilizadores, tipicamente enviado através de um botão concebido para o efeito [8].

Apesar de serem técnicas potencialmente eficazes à primeira vista, a colaboração dos utilizadores tem a desvantagem de ser relativamente pequena em relação a outras técnicas. Neste sentido, não é possível "treinar"eficazmente um sistema de auto-aprendizagem apenas com mecanismos colaborativos. Além deste facto, acresce a subjectividade que está inerente ao conceito de spam [7]. Assim, aquilo que uns utilizadores consideram spam, outros poderão considerar legítimo e com utilidade.

Na área da autenticação dos utilizadores de email, com vista a evitar técnicas de *spoofing*[6], têm sido propostas várias soluções que recorrem frequentemente a algoritmos criptográficos como PGP (*Pretty Good Privacy*) ou o S/MIME (*Secure/Multipurpose Internet Mail Extensions*). Tipicamente, estas soluções tentam provar a identidade da própria pessoa ao invés da autenticidade do endereço de email, pelo que é necessário recorrer-se a certificados digitais. Estas abordagens têm-se revelado bastante eficazes em testes de segurança, no entanto, a sua implementação a um nível global parece estar condenada ao eterno adiamento por razões de usabilidade [28].

As abordagens que têm como objectivo provar a identidade do remetente ao nível do domínio, nomeadamente DKIM (*Domain Keys Identified Mail*) e *SenderID*, têm vindo a registar uma adopção superior em relação às soluções baseadas na identidade do próprio remetente [28]. Através destas soluções, é possível que uma organização assuma a responsabilidade do email e que o receptor a verifique. Apesar de possuírem o mesmo objectivo, estas técnicas possuem filosofias de funcionamento diferentes.

Na técnica DKIM é utilizado o conceito dos pares de chaves pública e privada para autenticar determinado domínio de envio. Em cada mensagem, é adicionado um cabeçalho *DKIM-Signature* que contém uma assinatura digital, um código *hash* cifrado decorrente do corpo da mensagem entre outras informações relevantes [2]. O destinatário, ao receber uma mensagem com o cabeçalho *DKIM-Signature*, efectua um pedido DNS (*Domain Name System*), com base no domínio do remetente com vista a obter a chave pública

[6]Prática muito utilizada por *spammers* (nomeadamente, em contextos de *phishing*) que consiste em explorar a vulnerabilidade do protocolo SMTP que permite que qualquer remetente envie emails em nome de outro através da simples manipulação do cabeçalho *"From"* [70]

correspondente. Com esta informação, o destinatário poderá decifrar o valor da *hash* que está no cabeçalho e recalcular ele próprio a *hash* do email. Se os dois códigos coincidirem, fica provado que o remetente enviou o email a partir do domínio explicitado no campo *From* e que o email não foi modificado pelo caminho.

O sistema *SenderID* é idêntico ao SPF (*Sender Policy Framework*), embora introduza alguns melhoramentos. Este método verifica a autenticidade do campo remetente através da comparação dos IPs do remetente com uma entrada específica (tipicamente, TXT - *Text record*) nos servidores DNS. Nestes, o administrador de sistemas especifica quais os *hosts* que estão autorizados a enviar emails em nome do domínio que representam e desejam proteger [70]. Nenhum dos últimos dois mecanismos descritos foi concebido especificamente para combater o spam. No entanto, são técnicas que, quando integradas com outras, permitem aumentar as taxas de retenção de emails não solicitados.

Alternativamente, uma das sugestões anti-spam mais radicais é a de introduzir um sistema de recepção e envio de emails semelhante aos correios físicos, na vertente de custos. As soluções neste sentido sugerem a criação de um modelo de "selo digital", em que para o obter seria necessário comprá-lo a um custo o mais irrisório possível. Desta forma, utilizadores legítimos não teriam grandes dificuldades em suportar esse pequeno custo, mas os *spammers* teriam custos incomportáveis dada a quantidade exorbitante de emails que enviam, pelo que se acredita que a sua actividade deixaria de ser rentável [71].

Em sequência das investigações efectuadas na área, têm sido sugeridas imensas soluções anti-spam que nunca foram implementadas devido a questões de usabilidade e de viabilidade técnica ou económica. As abordagens actualmente em utilização não se revelam eficazes quando implementadas no singular, sendo sempre necessário a integração com outros mecanismos de funcionamento diferente. Só desta forma é possível diminuir as probabilidades de os *spammers* contornarem os mecanismos implementados.

2.3 Redes de *honeypots*

A utilização de *honeypots* permite alcançar variados objectivos no campo da segurança informática. Através dos *honeypots* é possível mitigar ou prevenir problemas utilizando abordagens diferentes do comum. No contexto do spam, a utilização de *honeypots* como spamtraps permite que a mitigação do problema seja potenciada, conseguindo-se travar grandes campanhas antes que utilizadores legítimos se apercebam.

2.3.1 Conceito

A maioria dos sistemas anti-spam está concebido para funcionar de forma reactiva, actuando apenas quando a missiva de spam está a caminho do receptor final. Por forma a que se possa actuar a um nível cada vez mais longínquo do receptor e mais próximo dos *spammers*, torna-se essencial obter grandes quantidades de informações sobre as suas campanhas e infraestruturas utilizadas no envio de spam. Deste modo, é possível estabelecer padrões e correlações entre vários tipos de informações que poderão servir para aprimorar os sistemas anti-spam reactivos e preventivos.

Dada a dificuldade em obter toda a informação necessária de forma manual e dada a globalização do fenómeno do spam, torna-se necessário criar mecanismos que recolham dados abrangentes e fiáveis. As redes de *honeypots* podem ser constituídas apenas por spamtraps ou poderão ser mais complexas e incluir conjuntos de falsos *open-relays* e *open proxies*.

Neste contexto, o *daemon HoneyD* [54] é uma ferramenta de grande utilidade que facilita a criação de um conjunto vasto de *honeypots*. Este instrumento permite criar facilmente postos virtuais numa rede que aparentam correr diversos serviços e sistemas operativos que emulam características reais que, por este motivo, são denominados *honeypots* de baixa interactividade. Com estes mecanismos pretende-se que os atacantes (spammers ou *crackers*) sejam iludidos e levados a tentar atacar os sistemas virtuais, protegendo-se os reais.

2.3.2 Exemplos de redes de *honeypots*

Os projectos que se encontram descritos nesta secção possuem diferentes objectivos; destinam-se a fins comerciais, académicos ou simplesmente altruístas. No entanto, é difícil encontrar descrições técnicas detalhadas sobre o modo como funcionam algumas das redes devido às suas características de cariz confidencial que necessitam de manter, sob pena de perderem eficácia. Dada a agressividade e a capacidade de rápida adaptação reconhecidas aos *spammers*, qualquer fuga de informação neste sentido poderia comprometer a eficácia do projecto, bem como, a fiabilidade da informação obtida.

Project Honey Pot

Actualmente, é o projecto sem fins lucrativos com mais destaque e sucessos acumulados na área das redes de *honeypots* de spam. A distribuição das spamtraps é efectuada com o auxílio de milhares de *webmasters* provenientes de várias regiões a nível global. Para participarem no projecto, estes colocam

nos seus *websites* um pequeno pedaço de software que servirá para descarregar spamtraps de um servidor central e para as colocar no site de forma automática. Estas spamtraps são únicas e são geradas conforme alguns dados do visitante (IP, *User-agent* e cabeçalho *HTTP referrer*) [52].

As informações sobre o visitante são criteriosamente registadas e posteriormente analisadas. Desta forma, além da informação decorrente do spam enviado para a spamtrap, o servidor central do projecto tem acesso às informações de rede de cada *harvester*. No *site* do projecto são regularmente publicadas as informações sobre os *harvesters* e *spammers* que se conseguiram detectar no âmbito do projecto.

De referir que, além das spamtraps, é igualmente colocado conteúdo estático que indica uma norma que avisa os utilizadores que o *site* não permite a recolha de endereços de email. Para evitar que os *harvesters* reconheçam que o local possui *honeypots*, o texto da norma é modificado de forma aleatória antes de ser colocado em cada *site* [52].

HoneySpam

A ideia subjacente a este projecto, descrita em [5], é a de utilizar a ferramenta *HoneyD* para criar uma rede de *honeypots* dirigida especificamente para o estudo do fenómeno do spam. Trata-se de uma *framework* que deve ser instalada na DMZ (*De Militarized Zone*) das redes que se pretende proteger e estudar. Inclui diversos serviços, sendo cada um vocacionado para um aspecto do funcionamento do spam. Neste sentido, possui falsos servidores HTTP para permitir o *harvesting*; falsos *open-proxies* e *open relays* para permitir o registo de algumas informações sobre os *spammers* e ainda servidores SMTP dedicados para recolher todo o spam recebido.

O objectivo desta *framework* é o de dificultar o trabalho dos *spammers* de diversas formas. Assim, utilizam os falsos servidores HTTP para colocar falsos *sites* que utilizam mecanismos semelhantes aos do software *Wpoison*, de modo a "envenenar"listas de endereços e a ocupar recursos computacionais dos *spammers*. Os falsos *open relays* e os *open proxies* são utilizados para levar os *spammers* a acreditar que estão a conseguir enviar spam através destes. No entanto, o spam enviado por esta via é bloqueado ao início e todos os registos são guardados para uso posterior. Os servidores SMTP decorrem da utilização de spamtraps verdadeiras, e têm como objectivo recolher o spam enviado e os dados adjacentes.

Projecto SpamPots

Coordenado pelo Grupo de Respostas a Incidentes de Segurança para a Internet Brasileira, este projecto tem como objectivo a obtenção de métricas sobre o abuso de redes de banda larga e envio de spam. Para este fim, foram utilizados dez *honeypots* de baixa interactividade [10].

A filosofia de funcionamento utilizada é semelhante ao do projecto *HoneySpam* no contexto dos *open relays* e *open proxies*, contudo não são utilizados *honeypots* HTTP. Ao contrário do *Honeyspam*, os investigadores decidiram apostar na distribuição dos *honeypots* por cinco máquinas ligadas a diferentes operadoras de cabo e ADSL, distribuídas por residências e empresas [10].

Os responsáveis publicaram diversos trabalhos sobre as métricas recolhidas durante um determinado período de observação. Em sequência dos resultados, manteve-se e expandiu-se a rede de *honeypots* para nove países.

2.3.3 Inovações e diferenças

O projecto "Rede automatizada de spamtraps"possui algumas semelhanças em relação aos exemplos de sistemas descritos na secção anterior. É utilizado um servidor HTTP que serve para alojar sites que possuem centenas de spamtraps. No entanto, contrariamente ao que é feito no projecto *Honeyspam*, os sites encontram-se ligados através de hiperligações estáticas, dado que não é objectivo do projecto envenenar e armadilhar o *crawler* dos *spammers*. Cada site possui algum texto, pelo menos uma imagem, e centenas de spamtraps reais, i.e., correspondem a caixas de correio verdadeiras que são monitorizadas no âmbito do projecto.

Ao contrário dos exemplos descritos, o projecto de software desenvolvido foca-se unicamente na distribuição de spamtraps e na recepção do respectivo spam, pelo que não são criados *honeypots* correspondentes a *open relays* ou *open proxies*. Dado o ênfase dado à componente das spamtraps, a distribuição destas é efectuada de um modo mais complexo, apostando-se na variedade de locais onde estas serão colocadas e nos conteúdos que as acompanham.

Neste sentido, considera-se que é na forma como as spamtraps são distribuídas que foi introduzida a maior inovação em relação a projectos semelhantes. Estas são distribuídas automaticamente, de forma discreta e inócua, pelos mais variados locais, sendo a frequência e os intervalos de tempo ajustáveis conforme necessário. Desta forma, pretende-se aumentar as probabilidades dos *harvesters* recolherem os endereços de email que correspondem a *honeypots* e por sua vez estes serem alvo de envio de spam. Através desta estratégia, espera-se captar o maior número possível de campanhas de spam

por forma a que futuramente o spam recolhido seja utilizado para aumentar a eficácia e abrangência de filtros anti-spam.

Capítulo 3

Rede Automatizada de Spamtraps

O projecto de software apresentado neste capítulo, e objecto principal desta dissertação, assenta numa arquitectura modular e de fácil integração. Neste sentido, foram desenvolvidos módulos de software que possuem diferentes objectivos e que funcionam de forma independente entre si. Assim, tornar-se-á mais fácil adicionar ou remover funcionalidades no futuro, bastando para isso manter a coerência das APIs (*Application Programming Interface*).

De referir a adopção de tecnologias *open-source* no decurso do desenvolvimento deste projecto de software. A opção efectuada neste sentido permitiu que, além da poupança a nível financeiro, se usufruísse da flexibilidade que caracterizam estas ferramentas, tendo sido efectuadas adaptações conforme as características e necessidades específicas do projecto. A comunidade que suporta este tipo de ferramentas foi outras das mais-valias que levou à escolha deste tipo de tecnologias.

Por forma a que o software desenvolvido fosse facilmente integrado noutras soluções, desenvolveu-se uma API generalista, acessível via *webservices* e que se encontra descrita neste capítulo. Foi igualmente desenvolvido um cliente *webservices* específico de modo a que se pudesse testar o sistema e iniciar a respectiva distribuição de spamtraps sem a dependência de outras aplicações que eventualmente integrarão a API. O cliente poderá ser executado a partir de qualquer ponto de rede, bastando apenas ter acesso ao servidor HTTP que fornece acesso à interface *webservices*. Neste sentido, o operador do sistema poderá coordená-lo manualmente ou através de mecanismos à sua escolha que automatizem o funcionamento do cliente *webservices*[1].

[1]Exemplo: Serviço *crond* (Unix/Linux)

3.1 Arquitectura do sistema

A concepção prática deste projecto pode ser resumida em dois principais componentes: configuração de serviços e desenvolvimento de módulos de software. Os serviços foram configurados tendo em conta a especificidade dos objectivos propostos, tendo sido desactivadas funcionalidades que não revelavam especial utilidade neste contexto. No entanto, cada um dos serviços desempenha um papel único e fundamental no contexto global do projecto.

Os módulos de software foram concebidos de raiz, utilizando a linguagem *Perl* [26], e em consonância com a configuração dos serviços *Unix* utilizados. De referir que todos integram bibliotecas pré-existentes, de cariz *open-source*, nas mais variadas funcionalidades. A utilização destes recursos é detalhada nas secções correspondentes deste capítulo.

Exceptuando a distribuição de spamtraps, todas as interacções desta aplicação com o exterior são efectuadas com a intermediação dos serviços configurados, conforme Figura 3.1.

3.1.1 Módulos de distribuição de spamtraps

Por forma a facilitar futuras expansões e integrações destes módulos, foi convencionado que cada módulo de distribuição activa teria uma API mínima a desenvolver:

```
spread_spamtraps(number, topic_ID, domain_traps);
```

A variável *number* corresponde ao número de spamtraps que se deseja distribuir e a *topic_ID* à identificação numérica do tópico com que se pretende integrar a spamtrap. A *domain_traps* é uma variável facultativa que permite ao administrador do sistema indicar o domínio das spamtraps que deseja distribuir. Caso a última variável não seja explicitada, o sistema escolherá as spamtraps aleatoriamente, sem restrição de domínio.

No exemplo seguinte, demonstra-se como se pode utilizar o módulo *newsgroups* para distribuir cinquenta spamtraps com o domínio "example.com", disfarçadas com conteúdos do tópico "4":

```
newsgroups::spread_spamtraps(50, 4, "example.com");
```

Fóruns

Os fóruns de Internet são uma plataforma de comunicação que se tem revelado bastante popular e útil. Por este motivo, foram criados diversas plataformas que permitem que qualquer utilizador, sem conhecimentos profundos na área de informática, consiga criar e publicar o seu próprio fórum. Com efeito,

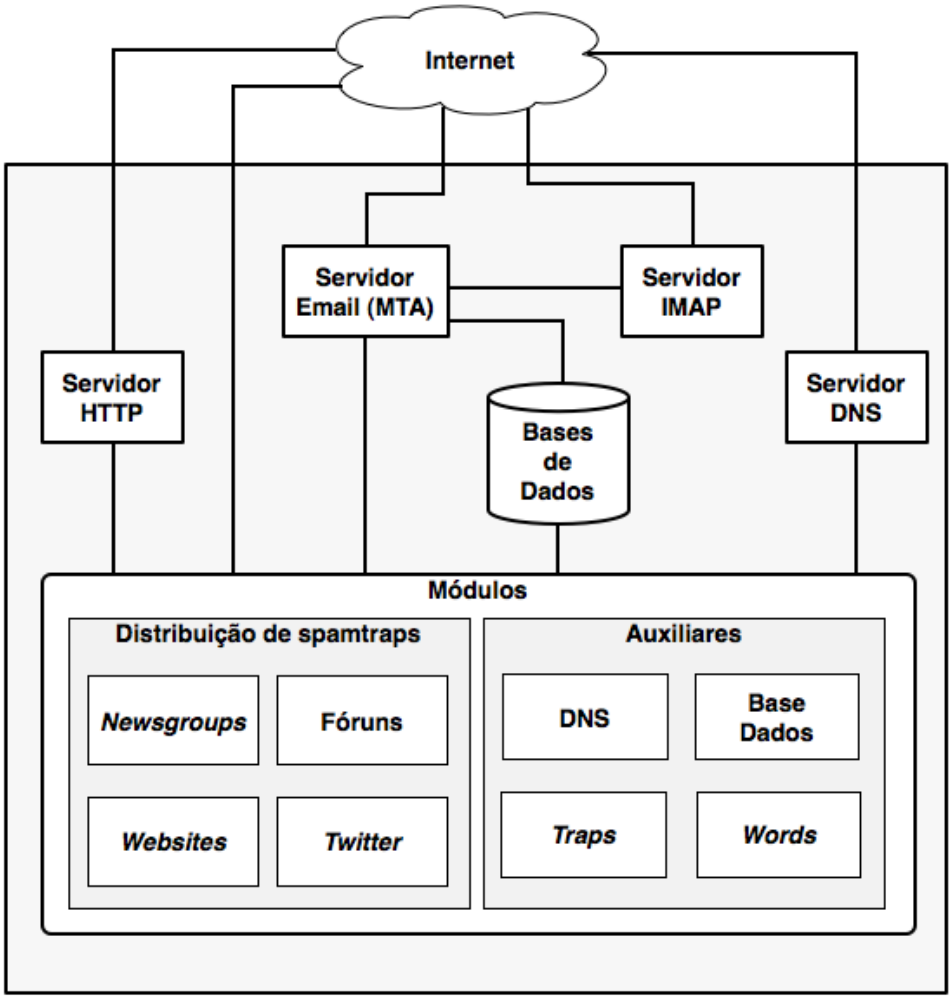


Figura 3.1: Arquitectura do sistema

duas plataformas baseadas na linguagem PHP (*Hypertext Preprocessor*) [30] têm vindo a assumir protagonismo neste campo: *vBulletin* e *phpBB*.

Por serem plataformas que funcionam de modo diferente, não é possível desenvolver um módulo único que preveja todas as suas especificidades. Neste sentido, decidiu-se desenvolver sub-módulos referentes a cada tipo de fórum: *vBulletin.pm* e *phpBB.pm*. Estes são controlados por um módulo que abstrai o administrador do facto de existem diferentes plataformas. Assim, sempre que do lado do cliente *webservices* for enviada uma ordem para distribuir spamtraps em fóruns, o sistema encarregar-se-á de o fazer de modo aleatório pelos dois modelos de fóruns.

Inicialmente, o desenvolvimento da componente de software correspon-

Algoritmo 3.1 Distribuição de spamtraps em fóruns

```
Require: num_spamtraps;
Require: topic;
  spamtraps = get_spamtraps(num_spamtraps);
  forums = get_forums();
  contents = get_contents(topic);
  counter = 0;
  while counter < num_spamtraps do
    forum = get_random_forum(forums);
    forum_threads = get_threads(forum);
    for forum_threads as thread do
      if is_thread_suitable(thread, topic) then
        content = get_random_content(contents, topic);
        spamtrap = get_random_spamtrap(spamtraps);
        post_topic(thread, content, spamtrap);
        counter = counter +1
      end if
    end for
  end while
```

dente aos fóruns *phpBB* teve como base o módulo *"WWW :: Mechanize :: Plugin :: phpBB"* [59] do *Perl*. No entanto, com o avançar do desenvolvimento, rapidamente se chegou à conclusão que o mesmo não correspondia às expectativas e não era suficientemente genérico para suportar os objectivos do projecto.

Por este motivo, decidiu-se alterar o papel relativo a este módulo, passando-se a integrar apenas algumas das suas funcionalidades, tal como a que permite efectuar um login em qualquer fórum *phpBB*:

```
phpbb_login("username", "password");
```

As restantes funcionalidades foram implementadas utilizando o módulo *"WWW :: Mechanize"* [67] que permite recriar a utilização de um *browser* a partir de um *script Perl*. A componente de software correspondente à distribuição de spamtraps em fóruns *vBulletin* foi igualmente desenvolvida através da integração do módulo *Perl* referido anteriormente.

Apesar das diferenças ao nível da organização e concepção dos dois tipos de fóruns, o algoritmo que realiza a distribuição de spamtraps é idêntico em ambos, Algoritmo 3.1.

Inicialmente, o módulo deverá obter, através de uma consulta à base de dados, uma lista de spamtraps que ainda não foram utilizadas, uma lista de

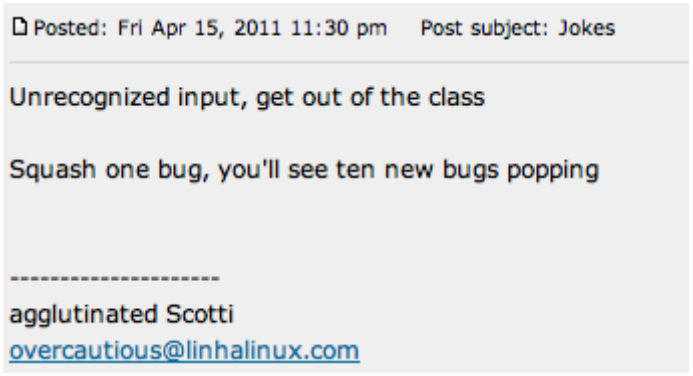


Figura 3.2: Exemplo de colocação de uma spamtrap num fórum *phpBB*

fóruns (com as respectivas credenciais) e uma lista de conteúdos relacionados com o tópico especificado pelo administrador. Enquanto o número de spamtraps especificado não for distribuído, o sistema visitará diferentes fóruns, escolhidos aleatoriamente da lista obtida inicialmente, em busca de *threads* compatíveis com os conteúdos especificados.

A compatibilidade referida é deduzida através da comparação de um conjunto de palavras-chave associadas ao tópico com o título da *thread*. Caso a *thread* se revele relacionada com o conteúdo a colocar, é enviada ordem para se colocar um novo tópico. Se a *thread* não estiver relacionada o programa continuará a percorrer as restantes. Na eventualidade de determinado fórum não possuir nenhuma *thread* satisfatória, o sistema encarregar-se-á de percorrer as dos restantes fóruns.

A mensagem colocada nos fóruns é constituída por um pequeno texto, com um número restrito de frases e por uma assinatura, Figura 3.2. A última é formada por nomes aleatórios retirados de um dicionário de palavras e por uma spamtrap. Quando a colocação é bem sucedida, é registado o local e a data onde foi colocada a spamtrap, impossibilitando deste modo a sua reutilização futura.

Newsgroups

O módulo referente à distribuição de spamtraps em *newsgroups* foi desenvolvido através da integração da biblioteca *News :: NNTPClient* [4] do *Perl*. Este módulo permite interagir, de forma simples e directa, com os servidores de *newsgroups*, através do protocolo NNTP (*Network News Transfer Protocol*).

Tal como nos restantes módulos, toda a acção é desencadeada através da invocação do método *spread_spamtraps* com os respectivos parâmetros.

Algoritmo 3.2 Distribuição de spamtraps em *newsgroups*

```
Require: num_spamtraps;
Require: topic;
  spamtraps = get_spamtraps(num_spamtraps);
  servers = get_newsgroup_servers();
  contents = get_contents(topic);
  counter = 0;
  while counter < num_spamtraps do
    server = get_random_server(servers);
    groups = get_groups(server);
    for groups as group do
      if is_group_suitable(group, topic) then
        content = get_random_content(contents, topic);
        spamtrap = get_random_spamtrap(spamtraps);
        post(server, group, content, spamtrap);
        counter = counter +1
      end if
    end for
  end while
```

O algoritmo de distribuição de spamtraps em *newsgroups*, conceptualmente, funciona de forma semelhante aos dos módulos referentes aos fóruns. Neste sentido, existe uma lista de servidores de *newsgroups* que é percorrida em busca de grupos compatíveis com o conteúdo com o qual se pretende acompanhar as spamtraps, Algoritmo 4.3.

A compatibilidade entre o tema dos conteúdos a utilizar e o grupo é deduzida através da comparação do nome do grupo com as palavras-chave associadas ao tópico escolhido. Assim, caso o administrador opte pela utilização do tópico *"jokes"*, o sistema tentará encontrar grupos cujo nome tenha palavras-chave como: "jokes", "joke", "piadas", "funny"ou "fun". Através deste mecanismo evita-se que a colocação das spamtraps em *newsgroups* tenha impactos negativos nos grupos, mantendo-se o esforço de assegurar a inocuidade do sistema. Neste sentido, às palavras-chave relacionadas com o tópico, juntaram-se termos como "test"ou "tests"de forma a abranger grupos como "alt.jokes.test"que, normalmente, são muito pouco visualizados por utilizadores humanos.

Algoritmo 3.3 Distribuição de spamtraps em contas do *Twitter*

```
Require: num_spamtraps;
Require: topic;
  spamtraps = get_spamtraps(num_spamtraps);
  twitter_accounts = get_twitter_accounts();
  contents = get_contents(topic);
  counter = 0;
  twitter_account = get_random_account(twitter_accounts);
  for 0 to num_spamtraps do
    content = get_random_content(contents, topic);
    spamtrap = get_random_spamtrap(spamtraps);
    update(account, content, spamtrap);
    seconds = rand(5,10);
    sleep(seconds);
    counter = counter +1
  end for
```

Twitter

A propagação de spamtraps em contas da rede social *Twitter* é o principal objectivo deste módulo. Novamente, a acção é desencadeada pela invocação do método *twitter::spread_spamtraps*. No entanto, por forma a facilitar a interacção com o *Twitter*, é necessário utilizar a API desta rede social. Para este efeito, recorreu-se ao módulo *Net :: Twitter* [43] do *Perl*. Este módulo permite que se coloquem actualizações em determinada conta do *Twitter* com uma simples invocação, desde que lhe sejam passadas as informações necessárias, Figura 3.3

```
$nt = Net::Twitter->new(
        traits              => @traits,
        consumer_key        => $consumer_key,
        consumer_secret     => $consumer_secret,
        access_token        => $token,
        access_token_secret => $token_secret,
);
$nt->update("Exemplo");
```

Figura 3.3: Exemplo de invocação da API do *twitter*

A variável "traits" especifica uma série de parâmetros relacionados com o funcionamento do objecto e da respectiva API, como por exemplo, o tipo

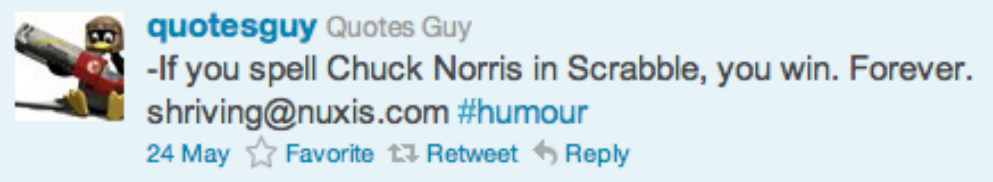


Figura 3.4: Exemplo de colocação de uma spamtrap numa conta do *twitter*

de API que é suposto o objecto suportar ou o modo como serão tratados os erros. As variáveis correspondentes ao "consumer_key", "consumer_secret", "access_token"e "access_token_secret"são obtidas no site "Twitter Developers" [64] após a associação de determinada conta com a aplicação. No âmbito deste projecto, foi decidido que sempre que fossem criadas contas do *Twitter* para efeitos de distribuição de spamtraps, guardar-se-iam estes dados na tabela "twitter_accounts"na base de dados.

O processo de distribuição de spamtraps funciona de modo mais simples do que os restantes módulos dado que qualquer publicação é apenas colocada na página referente ao próprio utilizador. Assim, por forma a associar os conteúdos e respectivas spamtraps a um determinado tópico, é colocada, pelo menos uma *hashtag* no fim de cada actualização. Por este motivo, o algoritmo de propagação de spamtraps em contas de *Twitter* possui menos instruções do que os anteriormente apresentados, Algoritmo 3.3.

De referir que, ao contrário dos módulos apresentados anteriormente, todas as spamtraps são colocadas numa única conta escolhida aleatoriamente no início do programa. Este facto permite que o número de tentativas de conexão ao *twitter* seja reduzido, evitando-se desta forma que a rede social associe o sistema automático de distribuição de spamtraps a actividades relacionadas com spam ou com tentativas de provocar uma negação de serviço. A colocação de spamtraps no *twitter* é por este motivo efectuada em blocos com um máximo de dez actualizações, sendo estas separadas por um intervalo aleatório entre cinco e quinze segundos.

O resultado obtido será sempre um número de actualizações coincidente com o número de spamtraps que se desejam colocar, com diferentes conteúdos dentro do mesmo tópico. A Figura 3.4 mostra um exemplo de uma das actualizações efectuadas automaticamente pelo sistema.

De referir que, dada a limitação de 140 (cento e quarenta) caracteres imposta pela rede social em cada actualização, configurou-se o módulo para apenas utilizar conteúdos cujo tamanho seja reduzido e que permitam ainda adicionar uma spamtrap.

Algoritmo 3.4 Criação de *websites* com spamtraps

Require: num_spamtraps;
Require: topic;
Require: subdomain;
Require: domain;
website_directory = create_website_directory(subdomain, domain);
create_website_html_file(website_directory, topic, num_spamtraps);
add_virtual_host_configuration(subdomain, domain, website_directory);
dns::add_subdomain(subdomain, domain);
reload_http_server();

Websites

O modo de propagação de spamtraps via *websites* estáticos diferencia-se dos restantes pelo facto de esta acção assumir um carácter passivo. Neste sentido, bastará adicionar um *website* para que um número pré-determinado de spamtraps fique imediatamente disponível aos *crawlers* dos *harvesters*, sem necessidade de qualquer intervenção posterior. Por este motivo, o processo que desencadeia a publicação das spamtraps em *websites* diferencia-se significativamente dos restantes.

A acção é iniciada assim que se invoca o método *"add_website"* que possui quatro parâmetros obrigatórios e um facultativo: número de spamtraps, subdomínio, domínio, tópico e domínio das spamtraps. Como forma de exemplo, o administrador que pretender adicionar o *website teste.example.com*, utilizando conteúdos relacionados com o tópico identificado pelo número "1"e de modo a distribuir duzentas spamtraps, deverá invocar o seguinte método:

```
websites::add_website("1", "teste", "example.com", 200);
```

De referir que o administrador deverá adicionar o domínio *example.com* previamente ao sistema, utilizando para esse efeito o respectivo método de *web-services*, que por sua vez invocará uma função do módulo *dns.pm*, detalhado na secção seguinte. A invocação do método *add_website* iniciará uma série de procedimentos automáticos que culminarão na criação e disponibilização do *website* no servidor HTTP local, Algoritmo 3.4.

O resultado final deste algoritmo é a criação de um *website* simples, que possui um título (relacionado com o tópico escolhido), uma lista de spamtraps apresentadas sob a forma de uma *unordered list* do HTML, alguns parágrafos de texto, uma imagem aleatória, porém relacionada com o tópico, uma secção com *links* e um rodapé com palavras aleatórias. A Figura 3.5 destaca uma amostra simplificada de código HTML gerado a partir do processo descrito.

Por se tratarem de *websites* que ficarão sob o controlo do sistema, o pro-

```
<html>
<head><title>Cars</title></head>
<body>
<h1>Cars</h1>
<ul>
 <li>Aletha Cassius cibol misclassified</li>
 <li><a href="mailto:Cole@linhalinux.com">
              Morty Betty_Lou Janos Peter_Hugo</a>
 </li>
 <li><a href="mailto:Yorke@linhalinux.com">
               Wen_Hsuch eruditional Olivia Gwyneth</a>
 </li>
</ul>
<div id='contents'>(...)was a manufacturing
company founded as a former British car Starley
& Sutton Co of Coventry in 1878. After developing(...)
</div>
<img src=".." alt="Pierre_Philippe" />
<div id="links">
 <a href="http://svn.nuxis.com">http://svn.nuxis.com</a>
</div>
<div id="footer">Giedre j Mary_Ann Fionnuala</div>
</body>
</html>
```

Figura 3.5: Amostra de código HTML gerado pelo módulo *websites*

cesso terá de abranger todos os aspectos inerentes a este facto. Assim, após gerar o conteúdo do novo *website*, o módulo cria um novo registo no servidor de DNS do sistema e configura um novo *Virtual Host*[2] no servidor HTTP. Após estas duas acções, os servidores DNS e HTTP são automaticamente recarregados com as novas configurações.

Devido ao facto de se possuir total controlo sobre os *websites* criados no âmbito deste sistema, adicionou-se a possibilidade de se colocar nestes, de forma automática, *links* para outros *websites*, fóruns ou contas de *twitter* associadas ao sistema de distribuição de spamtraps. Esta prática pretende provocar o redireccionamento de *crawlers* para locais que possuam outras

[2]Trata-se de uma configuração que permite que um único servidor HTTP responda por diferentes *websites*: `http://httpd.apache.org/docs/2.0/vhosts`

spamtraps, de forma a maximizar as probabilidades de recolha e envio de spam para estes endereços de email. No seguimento desta lógica, e dado que a criação de *websites* não implica a intervenção de elementos externos, é possível colocar um elevado número de spamtraps em cada uma das páginas, mantendo a discrição e inocuidade desejadas.

3.1.2 Módulos auxiliares

Os componentes de software descritos nesta subsecção, apesar de não intervirem directamente na distribuição de spamtraps, possuem um papel fulcral no funcionamento deste sistema. As funções que desempenham não permitem que sejam agrupados num grupo de módulos com um objectivo comum. Neste sentido, tanto possuem o propósito de introduzir uma camada de abstracção de software (módulo da base de dados) como de geração de palavras/frases aleatórios para vários efeitos (módulo *words.pm*). Nesta subsecção, serão alvo de descrição os objectivos globais de cada módulo e serão dados alguns exemplos sobre o seu modo de funcionamento.

Base de dados

Em qualquer projecto de software, a flexibilidade e a capacidade de adaptação a diferentes tecnologias sem grande esforço são características que se deverão ter sempre em conta. Neste sentido, e com o objectivo de possibilitar a utilização de uma base de dados diferente ou mesmo de qualquer outra tecnologia de armazenamento de informações, foi desenvolvido um módulo que disponibiliza uma API relativamente simplificada que permite abstrair o administrador do paradigma utilizado para a retenção de dados.

Dado que a escolha para o armazenamento de dados incidiu na tecnologia *MySQL* [13], todas as especificidades relativas a esta tecnologia foram reunidas neste módulo. Assim, todas as instruções SQL (*Structured Query Language*) utilizadas em todo o sistema poderão ser encontradas neste ficheiro. Deste modo, se no futuro for necessário alterar a tecnologia associada ao armazenamento de dados, bastará criar um módulo idêntico a nível de API.

Por forma a facilitar a comunicação entre este componente e a base de dados, optou-se pela utilização do módulo DBI [48] (*Database Interface*) do *Perl*. A simplificação introduzida, resultado da inclusão deste módulo, pode ser traduzida através da amostra de código que permite a introdução de um novo domínio na base de dados, Figura 3.6.

A utilização externa do módulo *database.pm* é igualmente simples, bastando invocar o método correspondente à interacção que se deseja efectuar com a base de dados. Assim, a título de exemplo, se for necessário introdu-

```
$dbh = DBI->connect( $dsn, $user, $pass )
my $query = "INSERT INTO domains (`desc`)
             VALUES ('$address');";
my $dbs   = $dbh->prepare(qq{$query});
$dbs->execute();
```

Figura 3.6: Amostra de código com integração do módulo DBI.

zir uma nova spamtrap (*spamtrap@example.com*) bastará invocar o método *insert_spamtrap* da seguinte forma:

```
database::insert_spamtrap("spamtrap@example.com");
```

Automaticamente, o software validará se a spamtrap já existe na base de dados e se o domínio é válido e está sob o controlo do sistema. Em todos os métodos é devolvido um inteiro negativo caso a operação não se tenha efectuado com sucesso, sendo que mais detalhes acerca de cada erro poderão ser encontrados nos *logs* do sistema.

DNS

Dada a necessidade de se possuir um sistema de resolução de nomes e de interagir com o mesmo, desenvolveu-se um módulo (*dns.pm*) que reúne todas as definições das interacções do sistema com o servidor DNS. Desta forma, além das vantagens a nível de organização do código-fonte, cria-se uma abstracção que permitirá que no futuro se possam utilizar outros serviços de resolução de nomes. Neste sentido, e à semelhança da componente anterior, bastará respeitar a API desenvolvida se for necessário criar um novo módulo com o mesmo objectivo.

O módulo foi concebido para interagir com o serviço BIND [12] (*Berkeley Internet Name Domain*), utilizando-se por este motivo ficheiros de zonas (*zone files*) para efeitos de configuração dos vários domínios e respectivos registos. A gestão destes ficheiros é o principal alvo deste módulo, sendo que foi concebido tanto para editar, como para apagar ou criar novos ficheiros. A sua API simplificada permite adicionar, editar e remover domínios/registos com invocações de fácil interpretação:

```
dns::add_domain("example.com");
dns::add_subdomain("subdomain","example.com");
dns::rem_subdomain("subdomain","example.com");
dns::rem_domain("example.com");
```

Sempre que exista uma invocação deste tipo, o módulo encarregar-se-á de, além de alterar todos os ficheiros necessários, de recarregar o serviço com as novas definições, assegurando-se antes que as configurações estão efectivamente correctas antes de as tentar aplicar.

Words

A utilização constante de palavras e frases aleatórias para diversos efeitos criou a necessidade de se desenvolver um módulo que desempenhasse apenas essas tarefas. Trata-se de um módulo relativamente simples cujo principal objectivo é o de fornecer palavras ou listas de palavras aleatórias, com base num dicionário criado num ficheiro à parte. Assim, e a título de exemplo, se houver necessidade de obter uma lista de vinte palavras, poderá ser efectuada a seguinte invocação a partir do exterior do módulo:

```
words::get_random_words(20);
```

A lista de palavras criada pelo método acima exemplificado é devolvida sob a forma de um *array* de *strings*. A qualidade da aleatoriedade das palavras devolvidas está directamente relacionada com o número de entradas no ficheiro do dicionário, não sendo registadas quaisquer palavras utilizadas anteriormente.

3.1.3 Ficheiros de configuração

Em todo o sistema desenvolvido existem variáveis comuns a vários módulos relacionadas com diferentes aspectos. Neste sentido, e por forma a centralizar informações vitais para o funcionamento do sistema, foram criados três ficheiros de configuração. De modo a facilitar a leitura e edição de entradas, optou-se pelo formato de ficheiros *.ini* que são facilmente interpretados pelas mais diversas bibliotecas, como a *Config :: Tiny* [37] do *Perl*. O exemplo seguinte revela o formato geral destes ficheiros de configuração:

```
[section]
var1=value1
var2=value2
```

Para aceder ao valor da *var2* a partir de um ficheiro exterior, assumindo que o nome do ficheiro é "config.ini"e utilizando o módulo *Config :: Tiny* do *Perl*, basta utilizar as seguintes linhas de código:

```
$conf = Config::Tiny->read('config.ini');
$var2 = $conf->{section}->{var2};
```

A opção pela separação dos ficheiros relaciona-se com uma evidente distinção lógica dos seus propósitos. Neste sentido, informações relativa a directorias e variáveis relacionadas com o serviço DNS foram colocadas no ficheiro *asn.conf*, Figura 3.7.

```
[asn]
dir=/var/www/cgi-bin/asn
log_dir=/var/log/asn

[apache]
htdocs_path=/var/www/html
conf_d_path=/etc/httpd/conf.d
```

Figura 3.7: Amostra de entradas no ficheiro *asn.conf*

Por sua vez, o ficheiro *mech.conf* possui informação relativa aos cabeçalhos HTTP a utilizar pelo módulo *WWW :: Mechanize* aquando da distribuição de spamtraps, Figura 3.8.

```
[headers]
REQUEST_METHOD='GET'
HTTP_ACCEPT_ENCODING='gzip, deflate'
HTTP_ACCEPT_LANGUAGE='pt-PT'
HTTP_ACCEPT_CHARSET='utf-8'
HTTP_CONNECTION='Keep-Alive'
HTTP_USER_AGENT='Mozilla/4.0
(compatible; MSIE 6.0; Windows NT 5.0;
 ie6; Q312461; .NET CLR 1.0.3705;
  .NET CLR 1.1.4322)'
```

Figura 3.8: Cabeçalhos HTTP utilizados pelo módulo *WWW :: Mechanize*

Por fim, o ficheiro *db.conf* possui informação de acesso às bases de dados do sistema, que poderá ser alterada a qualquer altura, conforme necessário.

3.1.4 Integração

Dado que é objectivo do sistema desenvolvido assumir um carácter genérico, foi decidido que, numa primeira fase, não se desenvolveria uma interface gráfica para a utilização das suas funcionalidades. Neste sentido, decidiu-se

desenvolver uma interface *webservices* REST (*Representational State Transfer*) que permitisse facilitar uma futura integração noutras aplicações. Não obstante, além do servidor, desenvolveu-se um pequeno cliente de *webservices* que facilita os testes e a sua integração em rotinas automáticas.

Servidor *webservices*

Por forma a colocar o sistema em "escuta" de ordens por parte do cliente, foi criado um ficheiro CGI denominado *et-traps.cgi*. Neste ficheiro reúnem-se as definições de todos os métodos disponíveis para utilização do cliente *webservices*, estando a API organizada em seis classes:

- *Domains*;
- *Forums*;
- *Newsgroups*;
- *Spamtraps*;
- *twitter*;
- *Websites*.

Sobre cada uma desta classes, é possível invocar determinados métodos conforme os objectivos previstos para cada uma. A Figura 3.9 revela a API que foi prevista para cada uma das classes referidas, destacando somente os métodos mais importantes:

Os argumentos "domain_traps"são de cariz facultativo; quando não são especificados o sistema escolherá as spamtraps aleatoriamente e sem observar qualquer domínio.

O processamento de cada pedido não está apenas dependente do módulo que lhe está directamente associado; existe a possibilidade de ser necessário invocar vários métodos de vários módulos para responder a determinados pedidos. O exemplo seguinte demonstra o pedido que é necessário efectuar para distribuir spamtraps em *forums*:

```
et-traps.cgi/forums/spread_spamtraps
```

O servidor ao receber a ordem descrita, automaticamente encarregar-se-á de invocar os métodos *spread_spamtraps* dos módulos *phpBB.pm* e *vBulletin* de forma aleatória. As invocações deverão ser feitas utilizando pedidos HTTP POST, onde se deverão incluir os argumentos necessários para cada método.

Domains:

```
add_domain(domain);
rem_domain(domain);
```

Forums:

```
spread_spamtraps(number, topic_id, domain_traps);
```

Newsgroups:

```
spread_spamtraps(number, topic_id, domain_traps);
```

Spamtraps:

```
create_random_spamtraps(number, domain);
rem_spamtrap(spamtrap);
```

twitter:

```
spread_spamtraps(number, topic_id, domain_traps);
```

Websites:

```
add_website(topic_id, subdomain, domain,
            num_traps, domain_traps);
```

Figura 3.9: API simplificada das classes desenvolvidas

Cliente *webservices*

Por forma a efectuar uma comunicação com o servidor *webservices*, foi desenvolvido um pequeno cliente utilizando a linguagem *Perl*. Este cliente possui diversos métodos, sendo que cada um corresponde a uma funcionalidade prevista no servidor *webservices*. A integração do módulo *LWP :: UserAgent* [1] do *Perl* permite facilitar a criação e envio de pedidos POST HTTP ao servidor. A Figura 3.10 revela a descrição do método que permite enviar a ordem para distribuir spamtraps em *newsgroups*.

O módulo criado pode ser utilizado em qualquer local com acesso HTTP à máquina que aloja o sistema, quer de forma directa quer através da integração noutra aplicação, funcionando por este motivo de forma completamente independente dos restantes módulos do lado do servidor.

3.1.5 *Spam collector*

O facto de a cada spamtrap estar associada uma conta de email provoca uma grande dispersão de *Maildirs* e de ficheiros relativos a mensagens de spam

```
sub spread_spamtraps_newsgroups {
 my ($number, $topicID, $domain_traps) = @_;
 my $browser = LWP::UserAgent->new;
 my $url=
 "http://$ip/cgi-bin/asn/et-traps.cgi/
     =/newsgroups/spread_spamtraps";
 my $response = $browser->post($url,
           'Content-Type' => 'text',
           'Content'  => $number.",".$topicID.
                 ",".$domain_traps);
  die 'Error getting $url' unless $response->is_success;
  print $response->content . "\n";
}
```

Figura 3.10: Amostra de código fonte do cliente de *webservices*

recebidas. Por forma a centralizar todas as mensagens numa plataforma que facilitasse a posterior pesquisa e investigação, criou-se um *script* autónomo em relação ao restante sistema que percorre todas as caixas de email em busca de novas mensagens de spam. Para este efeito, foram incluídos dois módulos do *Perl* que facilitam as tarefas relativas às conexões IMAP e de *parsing* dos cabeçalhos das mensagens: *"Net :: IMAP :: Simple"* [21] e *"Email :: Simple"* [62].

Por ser uma operação que implica um relativo custo a nível de recursos computacionais, convencionou-se que este *script* apenas seria executado uma vez por dia. Antes de iniciar a operação que permite reunir o spam, o *script* carrega as informações relativas às spamtraps activas e coloca-as numa lista. Para cada elemento desta lista, é iniciada uma conexão IMAP e são procuradas as mensagens cuja *flag* indica que é uma mensagem nova.

Em cada mensagem nova encontrada, são separadas diversas informações que constam nos cabeçalhos, como o email do remetente, o destinatário, a data, as informações do *relay*, o assunto e o próprio corpo da mensagem. Esta separação permitirá colocar estas informações em diferentes colunas da tabela *Spam* da base de dados, de modo a que a pesquisa envolvendo estes dados seja facilitada.

O paradigma escolhido para a recolha e reunião das mensagens de spam, apesar de ser menos eficaz do que uma recolha directa dos ficheiros, permite que o script possa ser executado em qualquer máquina, bastando para isso possuir acesso às bases de dados e consequentemente às credenciais IMAP relativas a cada spamtrap.

3.2 Base de dados

O correcto funcionamento do sistema automatizado de distribuição de spamtraps está dependente de duas bases de dados. Estas armazenam informação que deve ser mantida de forma completamente independente por forma a assegurar modularidade e flexibilidade do sistema. A base de dados "asn"possui todos os dados relativos às spamtraps, locais de distribuição, spam recebido e outras informações auxiliares, sendo que, o modelo conceptual desta base de dados foi inteiramente idealizado no âmbito deste projecto.

De forma a salvaguardar-se toda a informação sobre as caixas de correio correspondentes às spamtraps, optou-se pela utilização de uma outra base de dados denominada "postfix". Esta, criada inteiramente através de processos automáticos aquando da instalação e configuração do MTA *postfix* [51], armazena as credenciais de acesso, *alias*, algumas configurações e variados dados relativos ao funcionamento do servidor de email.

Nesta secção, a base de dados alvo de maior detalhe é a que corresponde directamente ao funcionamento do sistema de distribuição de spamtraps, "asn".

3.2.1 Esquema relacional

A base de dados "ASN"é composta por onze tabelas sendo o número de colunas variável entre dois e onze, Figura 3.11. Nesta subsecção encontra-se detalhado propósito de cada uma das tabelas, bem como o seu papel no sistema e na própria base de dados.

Topics

Os campos desta tabela armazenam a informação relativa aos tópicos dos conteúdos que acompanham as spamtraps. Assim, a cada tópico está associado, além de um código de inequívoco de identificação, uma breve descrição e um conjunto de palavras chave associadas, separadas por vírgulas. Durante a distribuição de spamtraps, o sistema utilizará esta lista de palavras chave para verificar se o local onde está a colocar as spamtraps se compatibiliza com os conteúdos que as acompanham. A Figura 3.12 revela uma entrada típica na tabela em questão.

Images

O papel desta tabela é relativamente pequeno, sendo apenas utilizada aquando da criação de *websites* que possuem algum texto e uma imagem relacionada.

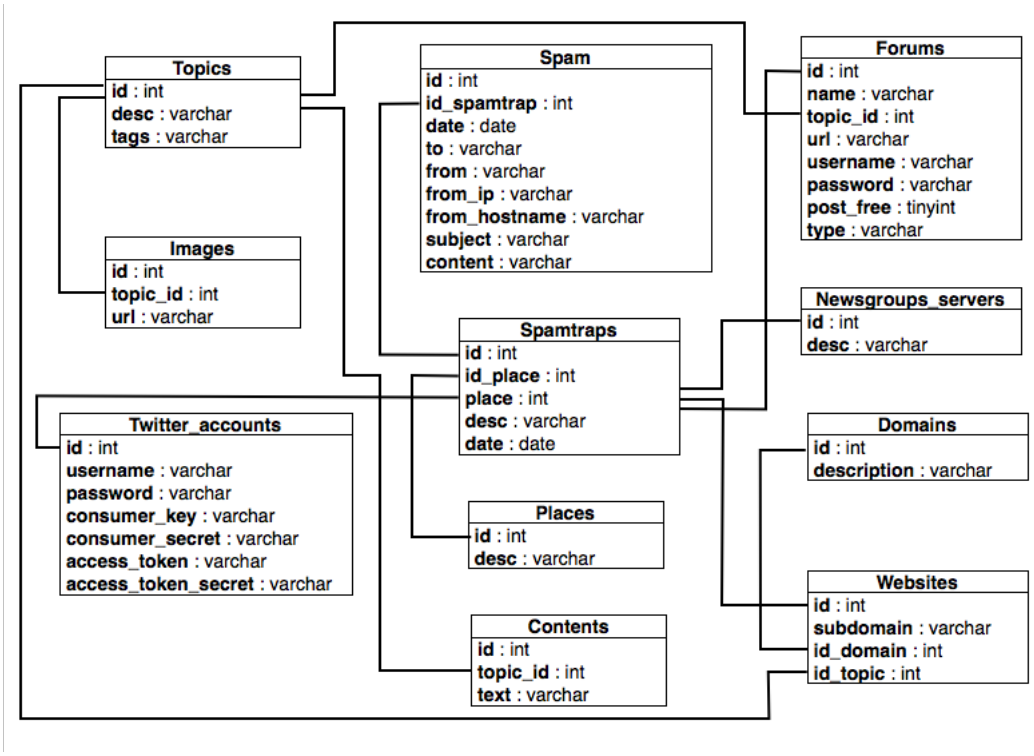


Figura 3.11: Esquema relacional da base de dados

Na tabela em questão é possível armazenar várias URLs de imagens e associa-las ao tópico respectivo. O exemplo seguinte revela uma entrada em que a imagem de um computador é associada ao tópico identificado pelo número 5, "Informatics".

```
ID: 5
topic_id: 1
url: http://www.hr.armstrong.edu/sac/imgs/computer.jpg
```

twitter_accounts

Toda a informação relativa às contas de *twitter* utilizadas na distribuição de spamtraps encontra-se nesta tabela. Além dos campos habituais relativos a credenciais, possui chaves geradas automaticamente após o registo da aplicação no sistema de API do *twitter* e que são de uso obrigatório para efeitos de autenticação e transferência de dados. Cada conta de *twitter* registada nesta tabela possui um identificador único que deverá ser utilizado durante a distribuição de spamtraps.

```
ID: 4
Desc: "Jokes"
Tags: "humor,humour,funny,joke,jokes,trash,Miscellaneous,
test,cancel,off-topic,offtopic,testes,teste,lixo,
anedotas,anedota, piada,piadas,random,
comic,comico,comics,comicos,comedia,fun"
```

Figura 3.12: Entrada típica na tabela *Topics*

Spam

Embora não represente um papel directo no processo de distribuição de spamtraps, esta tabela assume especial importância na análise de resultados. Nesta tabela são armazenados todos os emails de spam enviados para o servidor de email, sendo que as informações são separadas por várias colunas por forma a facilitar a pesquisa e o reconhecimento de resultados. Não obstante, a correspondência das mensagens de spam com o respectivo identificador da spamtrap assume grande importância no contexto global do projecto. É através desta ligação que se poderão estabelecer raciocínios que permitirão inferir quais as spamtraps que têm um maior grau de sucesso na recepção de spam e, por conseguinte, deduzir quais os locais onde a distribuição de endereços é mais eficiente.

Spamtraps

O papel desta tabela é fulcral no funcionamento do sistema. Reúne informação acerca de cada spamtrap existente, quer já tenha sido colocada quer ainda esteja de reserva. Durante a distribuição das spamtraps, é colocada informação nesta tabela acerca do local onde foi deixada, bem como a respectiva data. Esta informação servirá para, posteriormente, se relacionar o local de distribuição com o spam recebido.

O exemplo seguinte demonstra o registo efectuado para uma spamtrap que foi colocada na conta de *twitter* identificada pelo valor "1".

```
ID: 38041
Desc: Inez@linhalinux.com
Place: 5
ID_place: 1
Date: 2011-05-06 14:29:25
```

Places

Perante a variedade dos locais de distribuição de spamtraps, surgiu a necessidade de se classificar e agrupar estes sítios. Neste sentido, criou-se a tabela *Places* por forma a que fossem registados os diferentes tipos de locais, conforme o exemplo da entrada abaixo explicitada:

ID: 2
Desc: newsgroups

A finalidade desta tabela relaciona-se essencialmente com a necessidade de se facilitar a pesquisa de spam e spamtraps por tipos de locais, bem como de reduzir a redundância de informação na tabela *spamtraps*.

Contents

Por forma a garantir a inocuidade do sistema, foi criada a tabela *contents* que deve registar diversos conteúdos de texto. Estes, agrupados por tópico (campo *topic_id*), servirão para acompanhar as spamtraps. Por este motivo, deverão inserir-se grandes quantidades de conteúdos nesta tabela por forma a que o sistema não crie um padrão facilmente revelado.

Forums

Tal como a tabela *twitter_accounts*, regista os dados relativos a contas de utilizador nos mais diversos *forums* da Internet. A tabela *Forums* regista igualmente o identificador do tópico do principal assunto do fórum a considerar. A utilidade deste campo relaciona-se essencialmente com a primazia que o sistema deverá dar em relação ao fórum, conforme o tópico dos conteúdos que acompanharão a spamtrap. No entanto, dado que muitos fóruns possuem diversos temas, a escolha do local onde se colocará a spamtrap é essencialmente efectuada conforme o título de cada *thread*.

Além das credenciais e do tópico principal, são retidas algumas informações adicionais relativas a cada fórum, destacando-se o campo *post_free*. Este campo, do género booleano, regista se é possível colocar *posts* sem que seja necessário observar os temas relativos a cada *thread*. Assim, sempre que um fórum possuir o valor "1" neste campo, o sistema colocará a spamtrap acompanhada de conteúdos num local aleatório.

Esta opção relaciona-se essencialmente com o facto de se prever serem utilizados alguns fóruns que já deixaram de ter qualquer tipo de actividade por utilizadores humanos, restando apenas spam e conteúdos antigos. Por outro lado, é também prevista a possibilidade de o sistema utilizar fóruns criados

propositadamente para efeitos de distribuição de spamtraps por elementos ligados à administração deste software.

De referir o campo *type* que regista o tipo de sistema em que o fórum foi concebido. Desta forma, e conforme convencionado na fase inicial, está previsto o suporte a fóruns *phpBB* e *vBulletin*.

Newsgroups_servers

Nesta tabela são registados todos os servidores de *newsgroups* que não requerem autenticação para escrita de mensagens. Durante a distribuição de spamtraps, são escolhidos servidores aleatórios a partir desta tabela sendo a conexão com cada um destes efectuada directamente através do protocolo NNTP. O exemplo seguinte demonstra uma entrada típica desta tabela:

```
ID: 6
Desc: news.aioe.org
```

Domains

Na sequência da integração da gestão de domínios, surge a necessidade de se criar uma tabela simples para efeitos de registo e por forma a facilitar diversas operações internas do sistema. No entanto, é na vertente de gestão de *websites* com spamtraps que se regista uma maior necessidade na utilização de uma tabela deste tipo. Assim, e por forma a reduzir a redundância de informação na tabela *Websites*, criou-se uma tabela com apenas dois campos, correspondentes a um identificador e ao próprio domínio, conforme o exemplo que se segue:

```
ID: 171
Desc: linhalinux.com
```

Websites

Além das configurações necessárias a efectuar no servidor HTTP, o sistema regista igualmente na base de dados todos os *websites* que são criados. Por cada *website* criado, são armazenados dados relativos ao seu domínio (sob a forma do identificador constante da tabela *domains*), subdomínio e tópico dos conteúdos. O identificador de cada *website* servirá para, à semelhança das restantes tabelas que representam locais de distribuição, associar aos registos das spamtraps colocadas em cada um. O exemplo seguinte revela o registo do *website www.linhalinux.com*, associado ao tópico *Jokes*:

```
ID: 958
Subdomain: www
ID_domain: 171
ID_topic: 4
```

3.3 Serviços

Os objectivos do sistema de distribuição de spamtraps implicam a utilização e gestão de vários serviços de rede. Neste sentido, procedeu-se à instalação e configuração de quatro serviços diferentes, que assegurarão a resolução de nomes, a recepção, acesso ao email e a disponibilização de *websites*.

3.3.1 Servidor DNS

De modo a assegurar a resolução de nomes dos domínios sob alçada do sistema, configurou-se um servidor DNS em ambiente Linux. Dadas as opções disponíveis actualmente, a escolha para o serviço DNS recaiu no software BIND, versão 9.3.6, dada a sua simplicidade e vasta documentação. Acresce o facto de ser um dos serviços de DNS mais utilizados a nível global, assegurando-se por este motivo a sua robustez e fiabilidade. Não obstante, e por ser um serviço completamente configurável a partir de ficheiros de texto simples, as definições dos registos são facilmente alteradas a partir do módulo *dns.pm*.

3.3.2 Servidor Email

A escolha do software *Postfix*, na versão 2.3.3, para a gestão de email decorreu da necessidade de se assegurar a recepção de grandes quantidades de mensagens em curtos espaços de tempo, tendo-se em conta as reconhecidas características de robustez e flexibilidade desta plataforma.

Como forma de armazenar as credenciais e as mensagens referentes às caixas de correio das spamtraps, optou-se pela utilização de *virtual mailboxes* ao invés de as associar com contas *Unix*. Dada a grande quantidade de caixas de correio a criar, a última opção seria pouco viável. Neste sentido, foi associada uma base de dados *MySQL*, completamente independente da base de dados do sistema de distribuição de spamtraps, que armazena todas as informações referentes a cada caixa de correio. No entanto, as mensagens são guardadas em formato *Maildir* numa pasta do sistema (/usr/local/virtual), hierarquizada por domínios e nomes de utilizador.

3.3.3 Servidor IMAP

Por forma a que fosse possível aceder organizadamente às mensagens de spam nas diferentes caixas de correio de spamtraps, optou-se pela instalação e configuração de um servidor IMAP [16]. Para desempenhar este papel, foi escolhido o software *Dovecot* na versão 1.0.7.

A sua configuração foi personalizada conforme o tipo de *mailboxes* utilizadas. Neste sentido, e dada a opção pela utilização de *virtual mailboxes* no servidor de email, configurou-se o serviço *Dovecot* para que procurasse as credenciais na base de dados do *Postfix* e fornecesse as mensagens presentes na directoria de sistema indicada na sua configuração, Figura 3.13.

```
user_query =
SELECT '/usr/local/virtual/\%d/\%u' AS home,
CONCAT('*:bytes=', quota) AS quota_rule FROM mailbox
WHERE username = CONCAT('\%u', '@\%d') AND active = 1

password_query =
SELECT CONCAT('/var/mail/', maildir)
AS userdb_home, username as user, password,
CONCAT('*:bytes=', quota) AS userdb_quota_rule
FROM mailbox WHERE username = '\%u' AND active = 1
```

Figura 3.13: Amostra da configuração do serviço *dovecot*

Tal como revelado no exemplo anterior, configurou-se o *Dovecot* para utilizar as *queries* descritas por forma a obter e a comparar credenciais. As variáveis "%u" e "%d" correspondem, respectivamente, ao nome de utilizador e domínio requisitados pelo cliente IMAP.

3.3.4 Servidor Web

Por forma a disponibilizar os sites que possuem spamtraps associadas, configurou-se e instalou-se o software HTTP da Apache [22], versão 2.2.3. A sua configuração encontra-se dividida em vários ficheiros por questões de organização e flexibilidade. Além do ficheiro principal, que possui as configurações genéricas do serviço HTTP (ficheiro "/etc/httpd/conf/httpd.conf"), existem vários outros que definem cada *Virtual Host* e que se encontram na pasta do sistema "/etc/httpd/conf.d". Estes são criados e configurados automaticamente aquando da execução do respectivo método do módulo *websites.pm*,

tendo-se optado, por razões de organização, pela criação de um ficheiro por cada *Virtual Host*.

3.4 Máquina virtual

Por forma a assegurar o funcionamento de todo o sistema desenvolvido, optou-se pela criação de uma máquina virtual dedicada em ambiente OpenVZ [46]. Esta opção permitirá que, no futuro, conforme as necessidades, se ajustem rapidamente as mais diversas características da máquina. Estes ajustes, em ambiente virtual, são frequentemente efectuados sem provocar qualquer tipo de indisponibilidade. Não obstante as vantagens a nível de gestão de recursos decorrentes do ambiente de virtualização, esta opção facilitará a migração da máquina para outro local, caso seja necessário.

3.4.1 Características gerais

Apesar da sua especificidade, o sistema desenvolvido não requer recursos computacionais extraordinários na fase inicial. Neste sentido, criou-se uma máquina virtual com um processador virtual de 2.33GHz de frequência e 4096 KB de memória *cache*.

A nível de memória principal, configurou-se a máquina para que possuísse 512MB. Quanto ao armazenamento secundário, utilizou-se um disco virtual com 15 GB de espaço e 500.000 (quinhentos mil) *inodes*.

3.4.2 Características de rede

Apesar do esforço de inocuidade efectuado, algumas acções efectuadas pelo sistema poderão ser conotadas como spam por algumas entidades. Neste sentido, e por forma a evitar sanções com consequências noutros sistemas, decidiu-se atribuir um IP público dedicado exclusivamente para o presente projecto. Este IP encontra-se associado à única interface de rede do sistema, sendo que, no futuro, é possível adicionar-se mais dispositivos virtuais de rede sem grande constrangimento.

3.4.3 Sistema Operativo

Na sequência da opção pela utilização de tecnologias *open-source*, foi decidido que o seria utilizado o sistema operativo Linux. A distribuição escolhida foi a *CentOS 5.4* [53], sendo a versão do *kernel* a 2.6.18-164, para processadores de 32 bit.

A instalação do sistema operativo foi efectuada utilizando uma única partição - *root (/)* - que ocupa a totalidade do espaço disponível em disco.

Capítulo 4

Distribuição automática de spamtraps

O fim do desenvolvimento e configuração do sistema culminou numa fase intensiva de testes. Estes, efectuados em ambiente controlado, permitiram resolver pequenos problemas e aprimorar a aplicação por forma a que se tornasse progressivamente mais autónoma. Verificou-se igualmente a correcta funcionalidade dos vários serviços configurados e testou-se a capacidade da máquina para as tarefas que nesta se iriam processar. As fases seguintes relacionaram-se com o reconhecimento e criação de contas de utilizador nos mais diversos locais, bem como a criação de conteúdos minimamente relevantes para serem colocados juntamente com as spamtraps. Após o término destas fases, iniciou-se a distribuição efectiva e autónoma de spamtraps pelos mais diversos locais, fora do ambiente controlado.

Neste capítulo é descrito o procedimento que culminou na colocação de milhares de spamtraps em locais distribuídos pela Internet. Não obstante, é objecto de detalhe o procedimento de preparação da acção bem como o da própria distribuição de spamtraps, numa perspectiva que permite efectuar um enquadramento ao nível de quantidades e de datas.

4.1 Preparação

A preparação para a início da distribuição de spamtraps desenvolveu-se em várias frentes. Foi necessário criar os conteúdos que acompanhariam as spamtraps, procurar locais adequados de distribuição e de preparar os serviços necessários para suportar o que era pretendido. Nesta fase, foram criadas, automaticamente, as caixas de correio correspondentes às spamtraps e foram criadas diversas contas em *forums* da *Web* e no *twitter*. Em relação aos *news-*

groups optou-se pela distribuição de conteúdos apenas em servidores que não exigissem registo prévio.

4.1.1 Criação de conteúdos

Os conteúdos que acompanham as spamtraps desempenham um papel fulcral no esforço de descrição e inocuidade que se pretende para o sistema. Neste sentido, procuram-se tópicos pouco susceptíveis de gerar discussão ou grande interesse em humanos. No entanto, o tamanho dos conteúdos teria de ser adaptado aos locais onde iriam ser colocados, ie., não faria sentido colocar um artigo grande sobre informática num *update* do *twitter*. Por este motivo, criaram-se três tópicos para serem utilizados em *websites* ("Informatics", "Cars" e "Clothes") e dois tópicos para serem utilizados nos restantes locais ("Jokes" e "Famous Quotes"). Foram igualmente associadas imagens a cada um dos tópicos, através da colocação de entradas na tabela *Images* da base de dados.

Aos tópicos direccionados para os *websites* foram associados artigos de tamanho considerável, enquanto que aos restantes associaram-se conteúdos de tamanho reduzido. Abrangendo a totalidade de tópicos, foram adicionadas 803 (oitocentos e três) entradas à tabela respeitante a conteúdos, sendo a maioria destes de tamanho reduzido.

4.1.2 Levantamento de locais

Por forma a definir os locais susceptíveis de serem visitados por *harvesters*, foram efectuadas pesquisas exaustivas na Internet. Visitaram-se dezenas de fóruns com vista a encontrar os que já possuíam pouca actividade de humanos e os que possuíam secções potencialmente compatíveis com os conteúdos preparados para acompanhar as spamtraps. Os fóruns que possuíam estas características à data eram alvo de uma tentativa de registo de utilizador, utilizando-se para isso credenciais fictícias e um email exclusivamente dedicado para o efeito. Naturalmente, as mensagens enviadas para o referido endereço de email não foram consideradas nos resultados.

No total, foram criadas com sucesso vinte e duas contas em fóruns de conversação, de forma manual, tendo as credenciais sido reunidas na tabela *Forums* da base de dados. A não opção pela automatização deste processo deveu-se ao princípio definido inicialmente de manter o sistema inócuo. Adicionalmente, a generalização da utilização de mecanismos *CAPTCHA* durante os registos tornaria o processo pouco fiável, trabalhoso e pouco eficiente em termos de gastos de recursos computacionais.

Com vista a efectuar uma distribuição de spamtraps em servidores de *newsgroups*, foi efectuada uma pesquisa que culminou com a elaboração de uma lista de 197 (cento e noventa e sete) servidores. Esta lista foi resultado de uma refinação de outra com um número muito superior de entradas que, no entanto, possuía servidores que não aceitavam *posts* de utilizadores não registados ou que já não se encontravam em pleno funcionamento. Para este efeito, foi elaborado e utilizado um pequeno *script* que, através de conexões directas via NNTP, visitou cada um dos servidores da lista obtida inicialmente. Os elementos da lista final foram adicionados à tabela *newsgroups* da base de dados.

Por fim, foram criadas quatro contas na rede social *twitter* de forma manual. A cada uma foi associada uma aplicação, por forma a que fosse possível utilizar a sua API conforme as regras impostas pela rede social. Os dados das contas criadas foram registados na tabela *twitter_accounts* da base de dados.

4.1.3 Sistema

Ao nível do sistema testou-se a conectividade dos serviços com o exterior e assegurou-se que apenas as funções necessárias para o funcionamento do sistema estavam activadas. Preparou-se igualmente o servidor HTTP para receber *Virtual Hosts* e configuraram-se os seguintes domínios no servidor DNS:

- *nuxis.com*;
- *linhalinux.com*.

Ambos os domínios já existiam previamente, no entanto, no momento inicial do estudo já não estavam a ser utilizados activamente. O servidor DNS foi configurado para que respondesse a *queries* DNS do tipo A e MX para ambos domínios.

De modo a preparar as caixas de correio associadas às spamtraps, foram criadas, de forma automática, 209.825 (duzentas e nove mil, oitocentas e vinte cinco) contas distribuídas de forma praticamente equitativa por ambos os domínios (*nuxis.com*: 104.920 e *linhalinux.com*: 104.905). Para este efeito, foi utilizado um dicionário de 161.780 (cento e sessenta e um mil, setecentas e oitenta) palavras.

Prevendo-se o envio de spam para endereços não existentes em cada domínio, num contexto de tácticas *bruteforce*, foram criados dois endereços de email especiais:

- *catch-all@nuxis.com*;
- *catch-all@linhalinux.com*;

Neste sentido, configurou-se um *alias* em cada domínio que apenas é aplicado quando o destinatário do email não existe, redireccionando a missiva para a respectiva caixa *catch-all*.

4.2 Distribuição

Após a verificação e preparação final do sistema, iniciou-se a distribuição efectiva de spamtraps fora do ambiente controlado. Esta actividade teve início no dia 25 de Março de 2011 e término a 24 de Maio de 2011, conforme Figura 4.1.

A não uniformidade na distribuição das spamtraps em relação ao tempo está relacionada com decisões tomadas ao longo deste período, com a aleatoriedade introduzida e com questões de carácter técnico que foram progressivamente resolvidas. Além deste facto, a observação do comportamento do sistema através dos *logs* permitiu que se efectuassem ajustes às quantidades e aos locais de distribuição das spamtraps.

No final da distribuição, verificou-se que, na totalidade, foram colocadas 27.061 (vinte e sete mil e sessenta e uma) spamtraps ao longo dos sessenta dias. Em média, foram colocadas 451 (quatrocentas e cinquenta e uma) spamtraps por dia, o que equivale a cerca de 3.157 (três mil cento e cinquenta e sete) por semana.

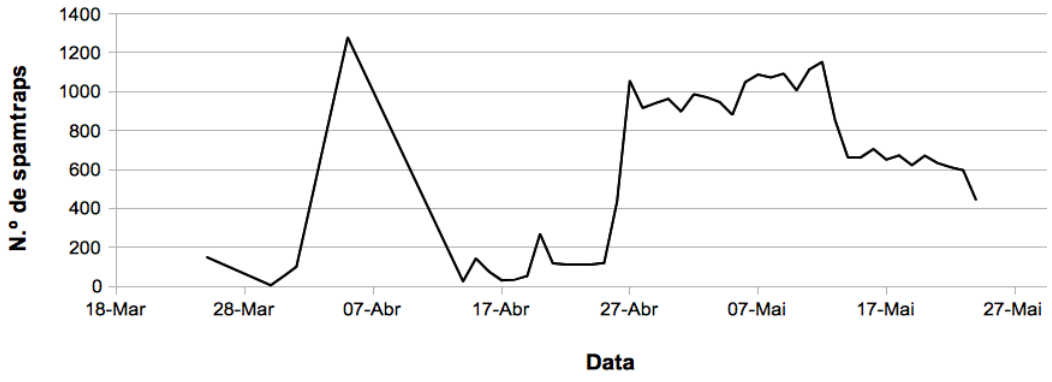


Figura 4.1: Distribuição diária de spamtraps

O pico de distribuição de spamtraps registado a 5 de Abril, evidenciado na Figura 4.1, relaciona-se com o facto de ter sido o dia em que foram disponibi-

lizados todos os *websites* com spamtraps. No entanto, a subida registada a 26 de Abril deriva do facto de ter sido tomada a decisão de aumentar o número de spamtraps distribuídas diariamente, mantendo-se o patamar nesta ordem até 15 de Maio. Nesta altura decidiu-se moderar a distribuição de spamtraps para cerca de seis centenas por dia.

Conforme explicitado na Tabela 4.1, pode-se verificar que a distribuição de spamtraps pelos locais não foi equitativa. Este facto relaciona-se com a sensibilidade associada a cada local e com o número de contas disponíveis. Neste sentido, fez-se uma análise da correlação de risco de incómodo para utilizadores humanos e da probabilidade de sucesso de recolha das spamtraps. Não obstante, rapidamente se concluiu que o local que deveria ser alvo de uma maior quantidade de spamtraps e de forma mais frequente eram as quatro contas de *twitter*.

De referir que a distribuição das spamtraps foi comandada a partir do cliente *webservices*, sendo a execução dos seus métodos programada a partir de um serviço de rotinas de sistema, o *crontab* do *Unix*. Assim, todos os ajustes relacionados com frequências de distribuição ou com quantidades de spamtraps poderiam ser feitos através da edição de um único ficheiro (*/etc/-crontab*).

Local	**Número de spamtraps**	**Percentagem total**
twitter	15.990	59,0
Newsgroups	9.467	35,0
Websites	1.212	4,5
Forums	392	1,5

Tabela 4.1: Número de spamtraps colocadas por local

.

4.2.1 *Forums*

A distribuição de spamtraps em fóruns foi a que se processou com mais moderação, dada a grande sensibilidade dos sistemas em questão. Em média foram colocadas diariamente entre seis e sete spamtraps, o que equivale a, aproximadamente, 45 (quarenta e cinco) por semana. No geral, a distribuição de spamtraps em fóruns ao longo do tempo esteve sujeita às mesmas restrições e decisões aplicadas a nível global, conforme anteriormente descrito e demonstrado pela Figura 3.1. Dos vinte e dois fóruns preparados inicialmente, apenas foram distribuídas spamtraps por dezoito. Este facto está

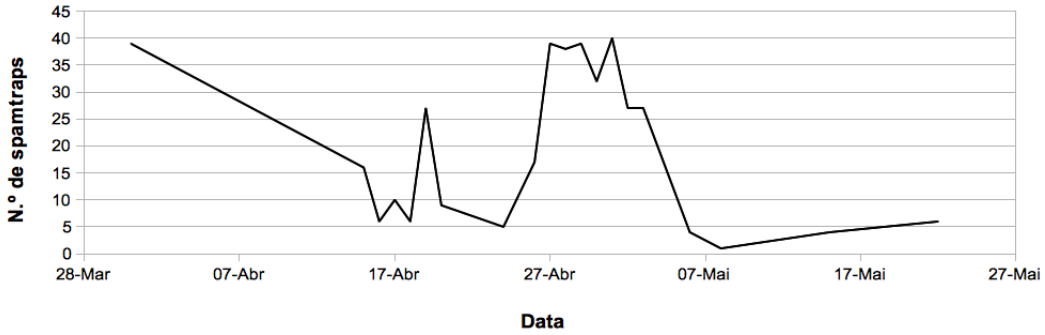


Figura 4.2: Distribuição diária de spamtraps em fóruns

relacionado com os registos de contas que não chegaram a ser autorizados pelos moderadores de alguns dos fóruns em tempo útil.

O número máximo de spamtraps colocadas num único fórum ao longo do período de distribuição foi de 80 (oitenta). De modo oposto, foi colocada uma única spamtrap em quatro fóruns. Apesar dos módulos de distribuição de spamtraps se encontrarem configurados para escolherem fóruns de forma aleatória, dos dezoito efectivamente utilizados, cinco receberam cerca de 70% da totalidade das spamtraps distribuídas. Esta discrepância está relacionada com o facto de alguns fóruns escolhidos possuírem regras apertadas de controlo, banindo o utilizador criado após uma, duas ou mesmo três mensagens.

Não obstante os conteúdos minimamente relevantes que acompanham a spamtrap e a escolha da *thread* apropriada, os moderadores provavelmente consideraram que a publicação de um endereço de email poderia estar relacionado com algum tipo de actividade associada a spam, tendo esta situação acontecido em oito fóruns. Ocorreram diferentes desfechos para as mensagens colocadas; alguns fóruns moveram-nas simplesmente para *threads* secundárias (normalmente denominadas por "lixo", "reciclagem" ou "spam") enquanto que outros eliminaram-nas definitivamente.

Nos restantes cinco fóruns não se verificou qualquer tipo de restrição e por esse motivo a discrepância observada, apesar de menor, pode ser explicada pela eliminação de *threads* relevantes para o sistema, por indisponibilidades pontuais ou por tempos de resposta demasiado elevados.

4.2.2 *Newsgroups*

O plano de distribuição de spamtraps em *newsgroups* foi mais ambicioso devido ao facto de se possuir uma lista de 197 (cento e noventa e sete) servidores disponíveis para receber *posts* anónimos. Acresce o facto de muitos

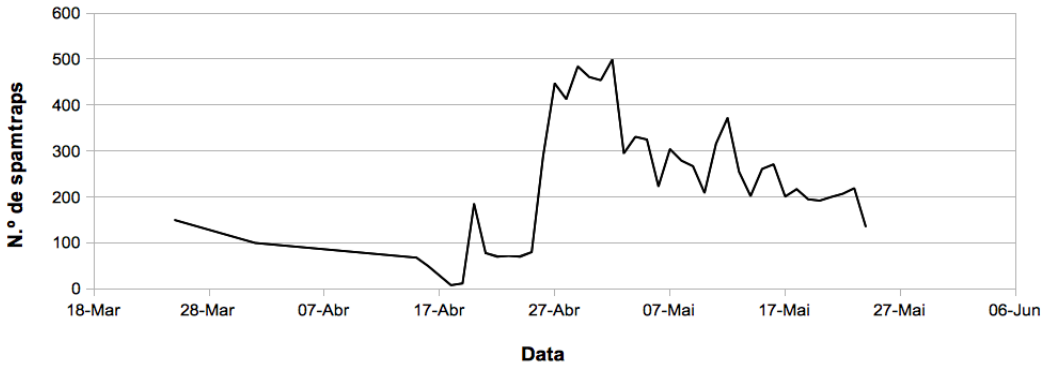


Figura 4.3: Distribuição diária de spamtraps em *newsgroups*

dos servidores albergarem centenas de grupos potencialmente compatíveis com os conteúdos com que se pretendiam acompanhar as spamtraps. Neste sentido, foram colocadas 9.467 (nove mil, quatrocentos e sessenta e sete) spamtraps ao longo do período de distribuição.

A média diária de colocação de spamtraps foi de cerca de 158 (cento e cinquenta e oito), o que corresponde a aproximadamente 1.104 (mil cento e quatro) por semana.

O número máximo de spamtraps colocadas num único servidor foi de 121 (cento e vinte e um) enquanto que o mínimo foi de apenas um. Dos 197 (cento e noventa e sete) servidores, apenas 170 (cento e setenta) receberam efectivamente spamtraps nos grupos que alojam. Este facto relaciona-se com a incompatibilidade de tópicos dos grupos alojados em cada servidor e com indisponibilidades de serviço. Dado que o email colocado no cabeçalho "*From*" poderia ser utilizado para confirmação de colocação de conteúdos ou para correspondência relacionada com o grupo, foi decidido criar um email dedicado para esta matéria (*news-here@linhalinux.com*). Consequentemente, este email não foi incluído na lista de spamtraps e por este motivo o spam recebido por este endereço não será objecto de estudo.

No decurso do período de distribuição, observou-se que muitos dos *posts* colocados acabaram por ser removidos ao fim de algum tempo, facto que se relaciona com as políticas associadas a cada grupo e com a necessidade de confirmação de submissão de conteúdos, via email. Não obstante, foi possível verificar que ocorreu uma elevada taxa de replicação de conteúdos para a *Web*, o que provavelmente poderá compensar o facto anterior[1].

[1]Exemplo: `http://devnet.jetbrains.net/thread/304429` (25/08/2011)

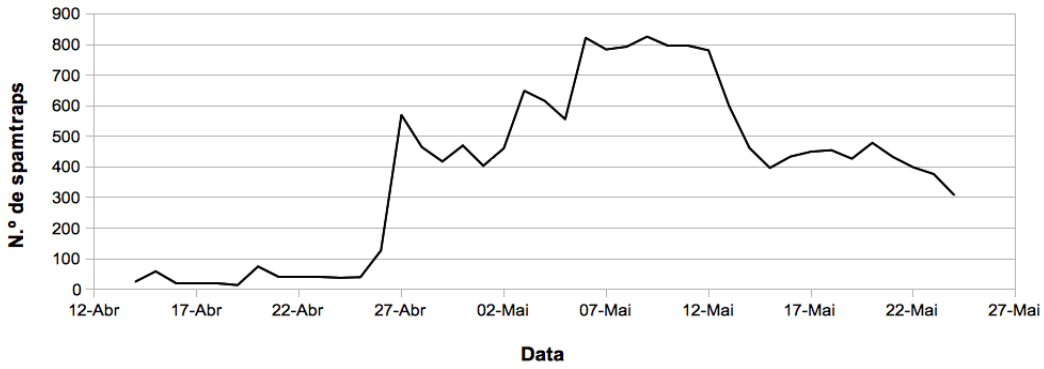


Figura 4.4: Distribuição diária de spamtraps em contas do *twitter*

4.2.3 *Twitter*

A distribuição de spamtraps em contas do *twitter*, apesar de se ter iniciado ligeiramente mais tarde, foi a que se processou de modo mais agressivo e ambicioso dado que as restrições ao nível da sensibilidade do local são muito inferiores. Neste sentido, foram colocadas 15.990 (quinze mil, novecentas e noventa e nove) spamtraps, numa média diária de 267 (duzentas e sessenta e sete) no contexto global da acção. Este número corresponde a cerca de 1.869 (mil oitocentas e sessenta e nove) de spamtraps enviadas semanalmente.

A replicação dos conteúdos colocados para vários locais da *Web* foi verificada pouco tempo depois do inicio da distribuição das spamtraps nas contas do *twitter*. Este facto foi observado através da colocação de palavras-chave específicas nos motores de pesquisa comuns. No entanto, alguns dos *sites* que apareciam nos resultados das pesquisas, já não possuíam quaisquer conteúdos. Este facto está relacionado com a associação dos conteúdos a *Trending Widgets*, ie., *widgets* que vão mostrando o que está a ser escrito no *twitter* em determinado momento e sobre determinado assunto, sendo registados pelos *crawlers* dos motores de pesquisa[2].

Observou-se igualmente que diversos *websites* integraram de forma permanente diversos conteúdos recolhidos no *twitter*. Esta é uma táctica comum praticada por *webmasters* com vista ao enriquecimento de conteúdos dos seus *sites* e consequente vantagem nos resultados dos motores de pesquisa para determinadas palavras-chave. É de esperar que esta prática traga vantagens ao nível da propagação das spamtraps pela *Web* e consequente recepção de spam.

[2]Exemplo: `http://spidername.com`

4.2.4 *Websites*

A criação e disponibilização de forma automática de todos os *websites* no âmbito deste sistema foi efectuada a 5 de Abril. Consequentemente, todas as spamtraps associadas a estes *websites* foram publicadas na mesma altura. Neste sentido, foram criados e disponibilizados doze sites, sendo que os domínios utilizados coincidem com os das spamtraps distribuídas. Para cada domínio, foram criados sites com os seguintes subdomínios:

- *svn*;
- *groups*;
- *blog*;
- *ftp*;
- *www*;
- *news*.

Em cada *website* foram colocadas 101 (cento e uma) spamtraps, perfazendo um total de 1.212 (mil duzentas e doze).

A escolha dos subdomínios teve em linha de conta a popularidade de cada um dos nomes referidos, por forma a aumentar as probabilidades de *harvesting*. Neste sentido, em cada um dos *Websites* criados, foram colocados *links*, de forma aleatória, para outros sítios ou para fóruns onde foram colocadas spamtraps. Além deste facto, configuraram-se os domínios para responderem aos pedidos que não indicam um subdomínio (ex.: "nuxis.com"), servindo um *website* já criado.

De forma a complementar o trabalho de *link building*, foram criados diversos *links* pela *Web*, nomeadamente ao nível de directórios e de plataformas de estatísticas e de *feedback* acerca de *Websites*[3].

[3]Exemplo: `http://www.aboutus.org/LinhaLinux.com`

Capítulo 5

Resultados

Neste capítulo detalham-se os resultados obtidos sob a forma de mensagens de spam recebidas. Inicia-se esta análise numa perspectiva geral com especial enfoque na correlação entre quantidades e datas, seguindo-se um relato acerca do próprio conteúdo das mensagens, bem como acerca de informação relacionada com os remetentes. Por fim, procede-se à análise mais relevante decorrente do projecto de distribuição automática de spamtraps, onde são estabelecidas correlações entre o spam recebido e os *honeypots* colocados.

Os dados utilizados na análise de resultados foram inteiramente captados a partir da base de dados "asn", sendo a tabela *spam* a que possui mais relevância neste contexto.

5.1 Enquadramento geral

Numa perspectiva de preceder uma análise profunda de resultados são explicitados, nesta secção, alguns factos gerais que permitem efectuar um enquadramento ao nível da escala de quantidade de mensagens e de tempo. O período de recolha e análise de spam teve início a 30 de Março e término a 27 de Agosto de 2011, totalizando 151 (cento e cinquenta e um) dias. Ao longo deste hiato foram recebidas 13.555 (treze mil, quinhentas e cinquenta e cinco) mensagens de spam. Este número corresponde a uma média diária de, aproximadamente, 90 (noventa) mensagens por dia.

Durante o referido período não se registaram quaisquer interrupções significativas na recepção de email. Não obstante, e apesar da fraca probabilidade, a ausência de registo não implica que na realidade não tenha ocorrido qualquer tipo de suspensão temporária. No entanto, caso se tenha verificado alguma interrupção por motivos externos, é provável que as mensagens enviadas para o sistema nunca tenham chegado, dado que não é comum que os

relays de spam repitam tentativas de envio, tal como relatado no Capítulo 2. Desta forma, é improvável que uma eventual falha temporária na recepção de emails implicasse um aumento de actividade em períodos de tempo posteriores.

5.1.1 Análise do espaço temporal

Durante o período correspondente à recepção de spam, verificou-se que, apesar da instabilidade do número de mensagens recebidas diariamente, houve um crescimento progressivo, conforme Figura 5.1. Este aumento coincidiu com a progressão da distribuição de spamtraps e consequente divulgação dos domínios utilizados.

A 27 de Julho verificou-se o maior número de mensagens recebidas em todo o período de análise, observando-se a entrada de 321 (trezentas e vinte e uma) missivas. Não obstante, o dia em que menos mensagens foram recebidas foi o de 7 de Maio, com apenas duas missivas recebidas. De um modo geral, os dias seguintes aos maiores picos revelados pela Figura 5.1 revelaram um decréscimo abrupto no número de mensagens recebidas. Este facto teve especial relevância a 6 e 7 de Maio, no intervalo entre 26 e 29 de Julho e a 8 e 9 de Agosto.

Dia da semana	**Número de mensagens**	**Percentagem**
Domingo	1.605	12
Segunda-feira	1.749	13
Terça-feira	2.132	16
Quarta-feira	2.191	16
Quinta-feira	1.960	14
Sexta-feira	2.076	15
Sábado	1.842	14

Tabela 5.1: Número de mensagens recebidas por dia da semana

.

A distribuição do número de mensagens pelos dias da semana ocorreu com uma diferença máxima de 4%. Os dias em que ocorreu maior recepção de spam foram as Quartas-feiras, conforme Tabela 5.1. Do lado oposto, os Domingos foram os dias em que menos mensagens de spam foram recebidas.

Numa perspectiva mensal, a recepção de spam foi relativamente constante ao longo dos dias, não se observando nenhum padrão característico neste âmbito. Não obstante, os dias em que houve uma maior recepção de spam

foram o 6 e 7, enquanto que os dias 1, 9 e 11 registaram um nível mínimo de mensagens. Ao nível da recepção de spam ao longo do dia, não foram identificados quaisquer padrões ou picos susceptíveis de ênfase.

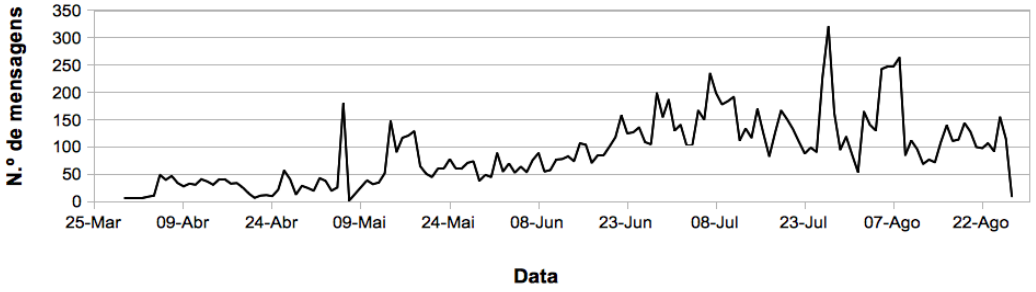


Figura 5.1: Recepção diária de spam

5.1.2 Destinatários

Na fase de planeamento do projecto ficou decidido prever que o sistema recebesse qualquer mensagem não solicitada dirigida aos domínios utilizados, numa perspectiva de receber o máximo de spam possível. Por este motivo, decidiu-se distinguir o spam recebido através das spamtraps e o que foi resultado de técnicas relacionadas com dicionários, utilizados numa tentativa de adivinhar potenciais destinatários em determinado domínio. Neste contexto registou-se um equilíbrio no número de mensagens recebidas: 6.826 (seis mil, oitocentas e vinte seis) para destinatários baseados em dicionários e 6.729 (seis mil, setecentas e vinte e nove) dirigidas a spamtraps.

O número de mensagens recebidas relativas ao início do período de análise (ie.: primeiros quinze dias) corresponde, quase na totalidade, a missivas dirigidas a destinatários não existentes. Este valor foi sendo reequilibrado à medida que se processava a distribuição automática de spamtraps no sentido em que um número crescente de mensagens se dirigia a estes *honeypots*.

Relativamente às mensagens recebidas nas caixas *catch-all*, registou-se um grande desequilíbrio entre os domínios analisados. Assim, contabilizaram-se 6.655 (seis mil, seiscentas e cinquenta e cinco) mensagens para o email *catch-all@nuxis.com* e apenas 171 (cento e setenta e uma) para o endereço *catch-all@linhalinux.com*. O endereço de email mais vezes tentado para o domínio *nuxis.com* foi *infraplanta@nuxis.com*, contabilizando-se 1.540 (mil quinhentos e quarenta), seguindo-se *gabriela@nuxis.com*, *abralapalabra@nuxis.com* e *nurwhiw@nuxis.com* com 1.489 (mil quatrocentas e oitenta e nove), 1.488 (mil

quatrocentas e oitenta e oito) e 1.280 (mil duzentas e oitenta) mensagens, respectivamente.

Os destinatários originais das mensagens captadas pelo endereço *catch-all@linhalinux.com* diferiram de forma vincada em relação ao outro domínio. Desta forma, o destinatário mais comum não existente e dirigido a este domínio foi *"news-h..."@linhalinux.com*, com 41 mensagens recebidas. Neste âmbito evidencia-se igualmente o endereço *news-h@linhalinux.com* que foi o destinatário original de 40 mensagens. De referir que se tratam de endereços de email captados na web, decorrentes da distribuição de spamtraps, que resultam de uma ofuscação de emails *news-here@linhalinux.com* praticada por parte dos *sites* que se dedicam à replicação de conteúdos de *newsgroups*. Não obstante, o endereço *info@linhalinux.com* obteve a segunda maior quantidade de mensagens neste âmbito, contabilizando -se 40. Este foi o email mais relevante no contexto de *bruteforce* de destinatários contra o domínio *linhalinux.com*.

5.2 Spam e spamtraps distribuídas

A análise presente nesta secção possui os resultados mais relevantes de todo o estudo, no sentido em que serão efectuadas correlações entre a distribuição automática de spamtraps e o spam recebido por estes *honeypots*. Recorde-se que o número de spamtraps colocadas neste âmbito foi de 27.061 (vinte e sete mil e sessenta e um). Assim, até ao fim do período de análise, a 27 de Agosto, 575 (quinhentas e setenta e cinco) spamtraps receberam efectivamente spam. Neste sentido, a taxa de sucesso registada na globalidade foi de cerca de 2%.

O intervalo de tempo mais curto registado entre a colocação de uma spamtrap e o início da recepção de spam foi de apenas 6 minutos e 54 segundos. Este registo foi obtido por uma spamtrap colocada numa conta do *twitter* a 2 de Maio. O registo do intervalo mais longo corresponde a 119 dias, 7 horas e 12 minutos, tendo sido obtido por uma spamtrap colocada num *website*. Em média, o intervalo de tempo entre a colocação da spamtrap e a recepção do primeiro email de spam foi de 29 dias, 8 horas e 10 minutos. A Tabela 5.2 revela a quantidade de spam recebido por local de colocação de spamtraps, demonstrando a percentagem em relação à totalidade de mensagens dirigidas a estas.

Nas subsecções seguintes é feita a análise de cada uma das categorias de locais onde foram colocadas spamtraps, verificando-se as respectivas taxas de sucesso bem como outras métricas. Considera-se que uma spamtrap foi bem sucedida quando esta recebeu, pelo menos, uma mensagem de spam.

Local	Número de mensagens	Percentagem
Twitter	2.994	45
Newsgroups	2.099	31
Websites	828	12
Forums	808	12

Tabela 5.2: Mensagens de spam recebidas por local de colocação da spamtrap

.

5.2.1 *Forums*

O número de spamtraps colocadas em fóruns da web que efectivamente receberam spam foi de treze, num total de 392 (trezentas e noventa e duas). Este valor corresponde a uma taxa de sucesso de 3,3%, sendo ligeiramente superior à média total registada. A recepção da primeira mensagem de spam por cada spamtrap demorou, em média, 7 dias, 23 horas e 17 minutos. O intervalo de tempo mais curto entre a colocação de uma spamtrap e a efectiva recepção de spam foi de 25 minutos e 16 segundos. Do lado oposto, o intervalo mais longo correspondeu a 52 dias, 2 horas e 10 minutos.

Em média, cada spamtrap pertencente ao conjunto de spamtraps bem sucedidas, recebeu 62 mensagens de spam. O valor máximo de missivas recebidas foi alcançado pela spamtrap *entoutcas@nuxis.com*, obtendo 148 (cento e quarenta e oito). Esta spamtrap foi colocada a 2 de Maio de 2011 na secção *testing* do *forum SkyscraperCity*[1]. No que diz respeito aos restantes elementos do conjunto das spamtraps que receberam spam, apenas duas não foram colocadas neste fórum:

- *khaki@nuxis.com*;
- *neutralists@linhalinux.com*.

As spamtraps listadas receberam, respectivamente, seis e três mensagens, tendo sido colocadas em dois fóruns diferentes: *The Joke Forum*[2] e *The VB Geek*[3].

5.2.2 *Newsgroups*

Das 9.467 (nove mil, quatrocentas e sessenta e sete) spamtraps colocadas nos mais diversos *newsgroups*, receberam spam 205 (duzentas e cinco). Neste

[1] `http://www.skyscrapercity.com`
[2] `http://www.thejokeforum.com`
[3] `http://www.thevbgeek.com`

sentido, a taxa de sucesso da distribuição de spamtraps nestes locais cifrou-se nos 2,2%. A spamtrap que mais rapidamente começou a receber spam foi a *momentary@nuxis.com*, registando-se um intervalo de 15 horas, 20 minutos e 47 segundos. No conjunto das spamtraps que receberam spam, a primeira mensagem demorou, em média, 33 dias, 15 horas e 36 minutos a ser recebida.

A quantidade média de mensagens de spam que cada spamtrap registou foi de cerca de dez, num intervalo de valores que variou entre 1 e 129 (cento e vinte e nove). O valor máximo foi alcançado pela spamtrap *nexus@nuxis.com*, colocado a 29 de Abril no grupo *microsoft.public.opsmgr.general* alojado no servidor `aioe.org`. Apesar deste valor, observou-se que 86% das spamtraps bem sucedidas receberam uma quantidade inferior a 10 mensagens.

O servidor de *newsgroups* que se revelou mais eficaz foi o *news.mozilla.org*, tendo as spamtraps lá colocadas recebido 539 (quinhentas e trinta e nove) mensagens de spam. De referir que, apesar deste valor, apenas 5% das spamtraps foram distribuídas por grupos alojados por este servidor.

5.2.3 *Twitter*

Apesar do grande volume de spamtraps colocadas em contas do *twitter*, apenas receberam spam 161 (cento e sessenta e uma), obtendo-se uma taxa de sucesso de 1,0%. O intervalo médio de tempo que decorreu entre a publicação das spamtraps e a recepção das primeiras mensagens de spam foi de 12 dias, 7 horas e 55 minutos. Tal como referido anteriormente, foi numa spamtrap colocada numa conta do *twitter* que se verificou a recepção mais rápida de spam, registando-se 6 minutos e 54 segundos. Este intervalo foi obtido pela spamtrap *loiter@linhalinux.com*, colocada às 20:33 do dia 2 de Maio. Apesar deste registo, a referida spamtrap apenas recebeu uma mensagem durante a totalidade do período de observação.

Em termos de quantidades de mensagens, o valor médio rondou as dezoito por spamtrap bem sucedida. Neste contexto, o valor máximo obtido foi de 632 (seiscentos e trinta e duas) mensagens, tendo sido registado pela spamtrap *despoil@linhalinux.com*. Tal como evidenciado pela Tabela 5.3, de entre trinta utilizadas, a *hashtag* mais bem sucedida foi a *#trash*.

Globalmente, 84% das spamtraps bem sucedidas, receberam menos do que dez mensagens cada. Não obstante, as cinco spamtraps que mais receberam spam, reuniram 61% da totalidade de mensagens.

5.2.4 *Websites*

A colocação de spamtraps em *websites* registou a maior taxa de sucesso, cifrando-se nos 16,2%. Neste sentido, registou-se a recepção de spam em 196

Spamtrap	**Número de mensagens**	***Hashtag***
despoil@linhalinux.com	632	*#trash*
inquiring@linhalinux.com	537	*#testing*
catchers@nuxis.com	293	*#trash*
socage@linhalinux.com	240	*#trash*
undermines@linhalinux.com	138	*#random*

Tabela 5.3: *Hashtags* que acompanharam spamtraps mais bem sucedidas

.

(cento e noventa e seis) das 1.212 (mil duzentas e doze) spamtraps colocadas. O tempo médio que decorreu entre a publicação das spamtraps e a respectiva chegada de spam foi de 40 dias, 5 horas e 31 minutos. Este intervalo é, no conjunto dos locais de distribuição de spamtraps, o mais longo.

Cada spamtrap recebeu, em média, quatro mensagens não solicitadas, tendo este número oscilado entre um e doze. O *website* onde foram colocadas as spamtraps que receberam mais spam foi o *blog.linhalinux.com*, reunindo 52% das spamtraps bem sucedidas e 80% da totalidade de mensagens. Apesar das spamtraps terem sido colocadas ao mesmo tempo, verificou-se que nem todas receberam a mesma quantidade de mensagens.

Não obstante, e embora com datas de recepção diferentes, pôde-se identificar um conjunto constante de mensagens de spam, coincidindo no conteúdo, no assunto mas não no IP do *relay*. No entanto, além do facto de as mensagens não terem sido enviadas ao mesmo tempo, verificou-se que não foi seguida qualquer ordem específica. Neste sentido, é possível deduzir o processo que culminou no envio da maioria do spam para as spamtraps colocadas nos *websites*:

1. Escolha aleatória de um endereço de email a partir de um conjunto pré-definido;

2. Escolha aleatória de uma mensagem a partir de um conjunto de missivas pré-definida;

3. Escolha e envio da mensagem a partir de um *relay* pertencente a conjunto de 129 (cento e vinte e nove) IPs identificados.

Este raciocínio permite prever que, embora determinadas spamtraps colocadas juntamente com outras não tenham recebido qualquer mensagem de spam durante o período de análise, venham a receber no futuro no sentido em que estas aparentam terem sido todas colocadas numa bolsa de endereços.

Verificou-se igualmente que a posição das spamtraps no *website* não teve qualquer influência na quantidade de spam recebida, na data de recepção da primeira mensagem ou no tipo de spam.

5.3 Spam

A análise efectuada até ao momento permite apenas deduzir resultados sob o ponto de vista quantitativo. Neste sentido, dedica-se esta secção a uma perspectiva direccionada para os conteúdos das mensagens de spam recebidas, observando-se igualmente os cabeçalhos presentes nas mensagens. Através destes é possível deduzir informações acerca de algumas dinâmicas de funcionamento da amostra de spam que foi possível reunir, bem como identificar as mais diversas campanhas captadas pelas spamtraps. Por fim, é feita uma análise às informações relativas aos remetentes, identificando-se gamas de IPs e respectivos países de origem.

5.3.1 Conteúdo

No conjunto das 13.555 (treze mil, quinhentas e cinquenta e cinco) mensagens de spam recebidas, pode-se identificar as mais diversas formas que o spam assume. Neste sentido, após observação de uma grande quantidade de mensagens de spam, foi possível verificar que é possível agrupá-las em oito categorias:

- Fraudes financeiras;
- Envio de vírus/*trojans*;
- *Phishing*
- Venda de medicamentos;
- Venda de produtos contrafeitos;
- Venda de serviços *online*;
- Ofertas de emprego falsas;
- Outros.

Na categoria relativa às fraudes financeiras, incluem-se todas as propostas para parcerias financeiras, recepções de doações milionárias e falsos anúncios de premiados em concursos. A segunda categoria diz respeito às mensagens

sem qualquer conteúdo, embora frequentemente com um assunto apelativo, com anexos com ficheiros binários. A venda de produtos farmacêuticos e contrafeitos formam duas categorias dado o grande número de mensagens que é possível incluir nestes dois grupos. As vendas de serviços *online* incluem todo o tipo de propostas para comercialização de listas de endereços de email, *link building*, SEO e venda de anúncios. A categoria relativa às ofertas de emprego falsas engloba todas as propostas de trabalho, desde os *part-times* que consistem no preenchimento de questionários aos anúncios para empregos milionários nos mais diversos países europeus e americanos. Por fim, a última categoria inclui todas as mensagens que não são passíveis de ser incluídas nas anteriores ou cujo objectivo final se mostrou imperceptível.

Para efeitos estatísticos, recolheu-se uma amostra de 575 (quinhentas e setenta e cinco) mensagens de spam, obtendo-se um grau de confiança de 95% e uma margem de erro de 4%. Através desta amostra foi possível averiguar que 92% do spam recebido vinha escrito em Inglês, 5% em Português, 2% em Italiano e 1% em Alemão ou Espanhol. Em termos de conteúdos, observou-se que 28% das mensagens de spam publicitavam a venda de produtos farmacêuticos de baixo custo de diversos tipos, sendo os produtos mais anunciados os das marcas *Viagra*® e *Cialis*®. Neste âmbito, verificou-se igualmente uma forte tentativa de vender produtos naturais que reclamam propriedades curativas fora do comum. A distribuição da amostra pelas categorias de spam identificadas encontra-se evidenciada no gráfico da Figura 5.2.

No que concerne aos assuntos das mensagens, verificou-se que, na sua maioria, estavam de alguma forma relacionados com o corpo do email. No entanto, observou-se que alguns destes possuíam apenas sequências de caracteres aparentemente aleatórios, tal como evidenciado pelo seguinte exemplo:

=?gb2312?B?yKu5+rj3tdjTqs/6yv2+3dK7wMCx7Q==?=

Na sequência de uma análise a estes emails, concluiu-se que na sua maioria transportavam apenas anexos binários com vírus ou *trojans*. Com efeito, verificou-se que 43 das mensagens recebidas incluíam a palavra "spam" no assunto, de um modo semelhante ao que é adicionado por diversos filtros anti-spam:

"[SPAM] [NOT A SPAM] – YOU ARE AN 'El Gordo' WINNER"

"*****SPAM***** RE:TOYOTA AUTOMOBILE COMPANY AWARD.."

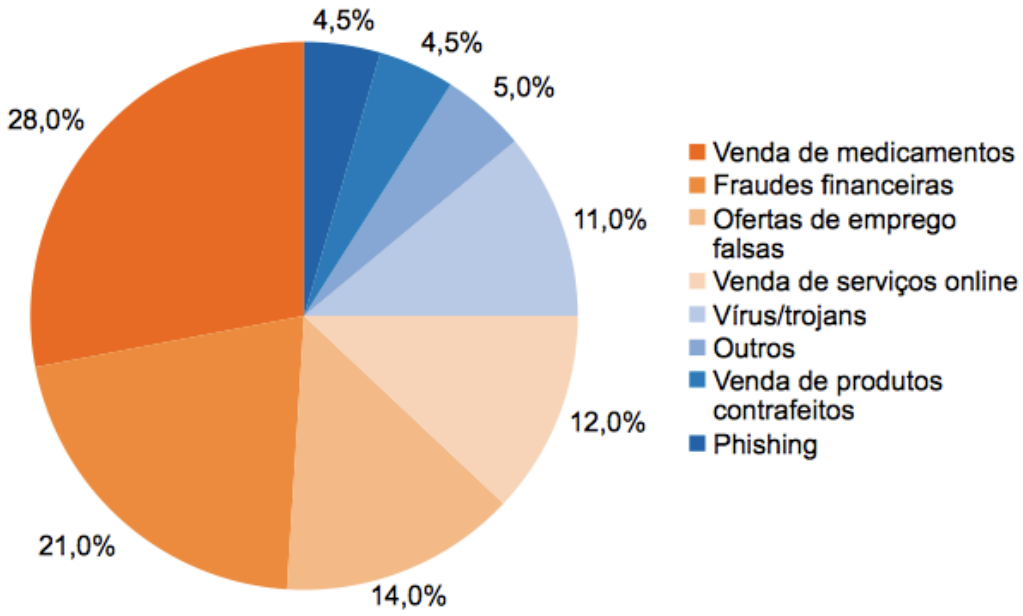


Figura 5.2: Percentagem de mensagens por categoria de spam

5.3.2 Proveniência

As informações relativas à proveniência do spam assumem um carácter especialmente útil no contexto de filtros anti-spam baseados nestes dados.

Gama de IPs	N.º de mensagens	Percentagem
111.224.250.0 - 111.224.250.255	1.851	14,0%
184.22.0.0 - 184.22.255.255	319	2,3%
209.85.128.0 - 209.85.255.255	265	2,0%
173.212.201.0 - 173.212.201.255	159	1,2%

Tabela 5.4: Spam recebido por gama de IPs

.

Neste sentido e por forma a complementar a informação relativa ao spam recebido no contexto do presente projecto, efectuou-se uma pesquisa que permitiu averiguar, para cada um dos IPs registados, o país de origem. Esta informação foi baseada numa base de dados, disponibilizada pela empresa MaxMind [42], que, para cada gama de IPs, especifica o seu país de origem.

Através da referida análise, foi possível verificar que cerca de 76% do spam recebido teve origem em 15 países diferentes. Tal como evidenciado pela

País	Número de mensagens	Percentagem
Estados Unidos	2.484	18,3%
China	2.413	17,8%
Rússia	1.019	7,5%
Coreia do Sul	918	6,8%
Brasil	731	5,4%
Ucrânia	543	4,0%
Índia	538	4,0%
Taiwan	297	2,2%
Roménia	285	2,1%
Alemanha	229	1,7%
Itália	227	1,7%
Polónia	175	1,3%
Reino Unido	167	1,2%
Vietname	155	1,1%
Bielorrússia	150	1,1%

Tabela 5.5: Spam recebido por país

.

Tabela 5.5, os Estados Unidos, China e Rússia foram as nações a partir das quais se enviou mais spam, somando 43% do total das mensagens recebidas.

No que concerne às gamas de IPs registadas verificou-se uma grande dispersão. No entanto, foi possível identificar os quatro conjuntos de onde partiram mais mensagens de spam, Tabela 5.4.

Capítulo 6

Conclusões

A análise dos resultados da distribuição automática de spamtraps permite enquadrar o projecto numa perspectiva bem sucedida, na medida em que foi possível reunir quantidades consideráveis de spam sem necessidade de intervenções prolongadas por humanos. Apesar de se ter verificado que apenas 2% das spamtraps colocadas receberam efectivamente spam durante o período de análise, foi possível averiguar quais os métodos mais eficazes que permitem obter quantidades superiores de mensagens não solicitadas.

6.1 Objectivos e resultados

Através da utilização de diversas tecnologias foi possível concretizar o principal objectivo do projecto: a distribuição automatizada de spamtraps em *newsgroups*, *forums*, *websites* e *twitter*. Às spamtraps fizeram-se acompanhar diversos conteúdos relacionados com cada local, sendo esta acção particularmente importante em *newsgroups* e *forums*. A correlação entre a sensibilidade do local de colocação das spamtraps e o spam recebido foi um dos aspectos que mais se teve em consideração. Verificou-se que o risco de prejudicar o normal funcionamento de determinados locais não era directamente proporcional à quantidade e/ou variedade de spam recebido pelas respectivas spamtraps. Num contexto de risco nulo, destacam-se os *websites* sob controlo do sistema cujas spamtraps colocadas apresentaram a taxa de sucesso (ie.: percentagem de spamtraps que receberam spam) mais elevada de todos os locais utilizados. Do lado oposto, os locais mais sensíveis que foram utilizados (fóruns de conversação) registaram uma taxa de sucesso de 3,3% que, apesar de ser ligeiramente superior à média, não compensa em termos de risco.

A comparação entre a percentagem total de spamtraps colocadas por local e o respectivo spam recebido permitiu verificar que estes valores não foram

directamente proporcionais. Neste âmbito, os locais mais eficientes foram os *forums* e os *websites*. O gráfico revelado na Figura 6.1 permite comprovar este facto através da comparação do total de spamtraps distribuídas e percentagem de spam recebido dirigido a estas. Este gráfico apenas contabiliza o spam dirigido às spamtraps.

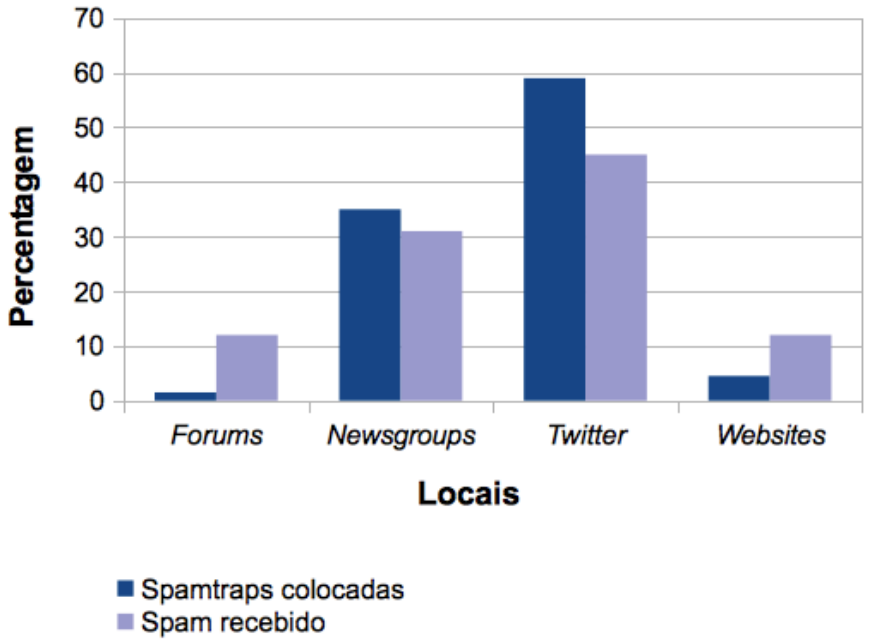


Figura 6.1: Spamtraps colocadas e respectivo spam recebido por local

A replicação dos conteúdos e das respectivas spamtraps assumiu um importante papel na definição do valor das taxas de sucesso e das respectivas quantidades de spam recebidas. Apesar da impossibilidade de se efectuar estas observações com o rigor científico pretendido, verificou-se que os conteúdos colocados nos fóruns aparentam ter tido a menor taxa de replicação. Não obstante, simples procuras nos motores de pesquisa mais comuns, permitiram concluir que a taxa de replicação das spamtraps colocadas nos *newsgroups* e *twitter* foi muito superior, observando-se uma grande diversidade de locais com os mesmos conteúdos.

Na amostra de mensagens utilizada para fins estatísticos descrita no Capítulo 5 não foram contabilizados quaisquer mensagens de *ham*. Adicionalmente, além da amostra referida, foi analisado um grande número de mensagens durante o período de análise, não se tendo verificado a existência

de *ham*. Neste sentido, é possível concluir que o objectivo proposto que previa que não fosse provocado interesse de resposta em utilizadores humanos foi cumprido.

A configuração efectuada no servidor de email que permitiu a recepção de email não dirigido a caixas de correio já existentes foi um aspecto bem sucedido, no sentido em que cerca de metade das mensagens de spam foram recebidas desta forma. Neste âmbito verificou-se que um do domínios utilizados - *nuxis.com* - começou a receber spam assim que se terminou a configuração do servidor de email. Desta forma, é possível concluir que a utilização de domínios antigos e sem utilidade actual potencia a recepção de spam com destinatários baseados em dicionários. Por outro lado, esta configuração no servidor de email permitiu contornar a ofuscação das spamtraps aplicada por alguns serviços de replicação de conteúdos (eg.: *Google Groups*), que têm como objectivo proteger os endereços de email dos utilizadores. Desta forma, o spam que seria à partida enviado para uma caixa de correio de uma spamtrap acaba por ser recolhido na caixa *catch-all*, não se perdendo a informação presente na mensagem.

6.2 Aplicação

Os resultados obtidos permitem conjecturar diversas aplicações para todo o spam que é possível receber como proveito da utilização do sistema desenvolvido. Neste sentido, além do próprio estudo académico e do estabelecimento de correlações com outras variáveis, é possível utilizar as mensagens de spam recebidas para efeitos de combate ao próprio fenómeno. Assim, poderão utilizar-se as informações que as próprias mensagens não solicitadas fornecem para fins de melhoramento de filtros anti-spam baseados nas mais variadas filosofias de funcionamento.

Nesta secção descrevem-se diversos métodos que permitem tirar o melhor proveito destas mensagens, podendo ser alvo de integração com o sistema de distribuição automática de spamtraps desenvolvido. Pretende-se, neste sentido, estudar a viabilidade de automatizar todo o processo, que tem uma abrangência que vai desde a colocação das spamtraps até ao processamento das mensagens por estas recebidas e consequente afinação dos sistemas anti-spam. Por fim, é efectuada uma breve análise à possibilidade de se enquadrar o sistema desenvolvido em soluções comerciais.

6.2.1 Integração com ETMX

O produto ETMX, desenvolvido pela empresa Eurotux Informática S.A., providencia um conjunto de serviços relacionados com email que permitem a sua filtragem, *backups* e protecção contra determinados ataques. Trata-se de um produto que transfere toda a carga relacionada com estes processos para as plataformas da empresa, libertando e/ou poupando determinados recursos dos clientes. Desta forma, o serviço actua como um intermediário do email, entregando apenas as mensagens que não consideradas spam pelo serviço de filtragem.

A tecnologia em que o serviço ETMX se baseia para a filtragem de email é o *SpamAssassin* [20]. Por este motivo é necessário que a integração do sistema de distribuição automática de spamtraps com o ETMX seja efectuada através desta plataforma. Esta conexão entre os sistemas será feita sob a forma de submissão automática de spam por forma a que o *SpamAssassin* aprimore a filtragem de spam.

O processo referido poderá ser efectuado através da utilização de um conjunto específico de ferramentas para o efeito, *masses* [23]. Estas ferramentas permitem submeter grandes quantidades de mensagens, das quais já se conhece previamente o seu estatuto de spam ou *ham*, melhorando desta forma o sistema de classificações da plataforma. No entanto, será necessário ter em conta a melhor forma de conduzir este processo de modo a que a submissão de numerosas mensagens de spam não contribua para uma eventual perda de performance consequente necessidade de analisar um número superior de regras [24]. Neste âmbito, deverão ser efectuados testes em ambiente controlado de modo a seja possível averiguar quais as melhores combinações a nível de número de mensagens submetidas, frequência e ferramentas a utilizar para o efeito.

6.2.2 Aplicações comerciais

Dada a crescente utilização do correio electrónico e a forte popularidade a este associada, torna-se cada vez mais necessário que as entidades responsáveis forneçam um serviço o mais fiável possível. Neste sentido, surgiu um mercado com uma oferta imensa na área do combate ao spam que tem vindo a tornar os produtos e serviços cada vez mas eficazes e sofisticados. No entanto, apesar da forte competitividade que está sempre associada ao ramo tecnológico, é possível introduzir no mercado serviços focalizados que proporcionem melhor qualidade em determinados nichos.

É precisamente neste ponto que se pode enquadrar uma possível saída comercial para o projecto desenvolvido, no sentido em que é possível controlar

os locais onde são colocadas as spamtraps, bem como os conteúdos que as acompanham. Consequentemente é possível efectuar um esforço de captação de spam no contexto de um país, assunto ou mesmo de locais na Internet, por forma a monitorizar determinadas campanhas de spam que poderão passar ao lado dos filtros comuns, nomeadamente mensagens em português.

A perspectiva de utilidade comercial do sistema desenvolvido pode ser encarada quer de uma forma directa quer indirecta. Na vertente indirecta, poderá utilizar-se todo o spam que é possível recolher com o único objectivo de melhorar sistemas anti-spam pré-existentes, enriquecendo-os com spam numa perspectiva de afinação e criação de regras de filtragem de email. Desta forma, o benefício desta abordagem decorreria do aumento da qualidade do serviço anti-spam e da consequente valorização do mesmo.

A aplicação comercial directa poderá ser concebida através da criação de um serviço responsável pelo fornecimento de informações constantes acerca do spam monitorizado. Este poderá ser utilizado pelas mais diversas entidades interessadas em obter spam para os mais diversos fins.

No entanto, para que qualquer um dos cenários a nível comercial se concretize será necessário reforçar o estatuto de confidencialidade que deve caracterizar o sistema de distribuição automática de spamtraps. Caso os *spammers* descobrissem que estão a enviar spam para *honeypots*, imediatamente poderiam adicionar os IPs associados ao projecto à sua própria *blacklist*. Acresce ainda a possibilidade de todo o sistema ser comprometido devido a um possível "envenenamento"do mesmo, através do envio de mensagens para as spamtraps com conteúdos normalmente associados a emails lícitos. Por este motivo, no futuro será necessário utilizar um conjunto de IPs de diversas gamas, bem como um conjunto alargado de domínios sem características que os unam.

6.3 Trabalho futuro

O sucesso deste projecto fora do âmbito académico está dependente do aumento do número de diversas variáveis intervenientes. Neste âmbito, a adição de domínios assume-se como a necessidade mais prioritária, por forma a que seja possível aumentar o leque de spamtraps disponíveis, e assim captar spam o mais abrangente possível.

Na sequência da replicação que foi verificada na publicação de determinadas spamtraps, nomeadamente no *twitter* e em *newsgroups*, deverão aumentar-se os conteúdos disponíveis utilizados para acompanhar os endereços de email. Deste modo, assegurar-se-á que o sistema poderá tirar o máximo partido possível dos mecanismos de replicação existentes na web.

Neste âmbito, será necessário aumentar a abrangência de locais por onde serão distribuídas as spamtraps, sendo que este aumento deverá ser acompanhado de uma diminuição na frequência de colocação de *honeypots* em cada local. Esta diminuição contribuirá para reduzir a probabilidade de as actividades realizadas no âmbito deste sistema serem consideradas como spam ou como tendo quaisquer objectivos maliciosos.

A utilização deste sistema em ambiente de produção, e tendo em conta o objectivo de recepção de um número cada vez superior de mensagens de spam, deverá criar a necessidade de aumentar alguns recursos em termos de armazenamento e de rede. Neste sentido, será necessário conceber uma arquitectura de complexidade superior que englobe várias máquinas com diferentes IPs, para onde os diferentes domínios deverão resolver. Desta forma elimina-se a possibilidade de todos os resultados obtidos serem postos em causa devido ao facto do único IP público utilizado poder ser reconhecido pelos spammers. Por outro lado, deverá ser feito um esforço no sentido de aumentar a eficiência do sistema em termos de armazenamento, comprimindo ou apagando spam mais antigo que já não se revele útil, entre outras medidas que se revelem adequadas.

Futuramente, os problemas que poderão surgir relacionam-se com a possibilidade de *reverse-blacklist* por parte dos spammers. Esta consiste na criação e divulgação de *blacklists* do lado dos spammers por forma a evitarem o envio de spam para os IPs ou domínios que forem reconhecidos como tendo *honeypots*. Adicionalmente, poderão ocorrer práticas de envenenamento das spamtraps, consistindo no envio conteúdos que normalmente se encontram em emails legítimos para as spamtraps, de modo a que estes sejam incorrectamente considerados spam. Para evitar estas situações, deverão ser tomadas diversas medidas que ofusquem os reais objectivos do sistema.

Bibliografia

[1] Gisle Aas. libwww-perl-6.02. `http://search.cpan.org/~gaas/libwww-perl-6.02/`, July 2011.

[2] E. Allman, J. Callas, M. Delany, M. Libbey, J. Fenton, and M. Thomas. DomainKeys Identified Mail (DKIM) Signatures. RFC 4871 (Proposed Standard), May 2007. URL `http://www.ietf.org/rfc/rfc4871.txt`. Updated by RFC 5672.

[3] David S. Anderson, Chris Fleizach, Stefan Savage, and Geoffrey M. Voelker. Spamscatter: Characterizing internet scam hosting infrastructure. In *Proceedings of the USENIX Security Symposium*, volume 4515 of *LNCS*, Boston, MA, August 2007. Springer-Verlag, Springer-Verlag.

[4] Rodger V. Anderson. Nntpclient-0.37. `http://search.cpan.org/~rva/NNTPClient-0.37/`, July 2011.

[5] Mauro Andreolini, Alessandro Bulgarelli, Michele Colajanni, and Francesca Mazzoni. Honeyspam: honeypots fighting spam at the source. In *Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop*, pages 11–11, Berkeley, CA, USA, 2005. USENIX Association. URL `http://portal.acm.org/citation.cfm?id=1251282.1251293`.

[6] Matthew Prince Arthur, Arthur M. Keller, and Ben Dahl. No-email-collection flag, 2004. URL `http://research.microsoft.com/en-us/um/people/joshuago/conference/papers-2004/163.pdf`.

[7] Joshua Attenberg, Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Mar tin Zinkevich. Collaborative email-spam filtering with the hashing-trick. 2009.

[8] Enrico Blanzieri and Anton Bryl. A survey of learning-based techniques of email spam filtering - technical report # dit-06-056. Technical report,

University of Trento - Information Enginnering and Computer Science Department, January 2008.

[9] Rainer Böhme and Thorsten Holz. The effect of stock spam on financial markets. 2006. URL http://ssrn.com/abstract=897431.

[10] Marcelo H. P. C. Chaves. Resultados preliminares do projeto spampots: Uso de honeypots de baixa interatividade na obtenção de métricas sobre o abuso de redes de banda larga para o envio de spam, 2007. URL http://www.cert.br/docs/whitepapers/spampots/.

[11] Junghoo Cho and Hector Garcia-Molina. The evolution of the web and implications for an incremental crawler. 1999.

[12] Internet Systems Consortium. Bind. http://www.isc.org/software/bind, July 2011.

[13] Oracle Corporation. Mysql :: The world's most popular open source database. http://www.mysql.com, July 2011.

[14] Paulo Cortez, André Correia, Pedro Sousa, Miguel Rocha, and Miguel Rio. Spam email filtering using network-level properties. In *Advances in Data Mining. Applications and Theoretical Aspects*, volume 6171/2010, pages 476–489, 2010.

[15] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. An open digest-based technique for spam detection. In *ISCA PDCS'2004*, pages 559–564, 2004.

[16] dovecot.org. Secure imap server. http://www.dovecot.org, July 2011.

[17] Patrick Dwyer and Zhenhai Duan. Mdmap: Assisting users in identifying phishing emails. 2010.

[18] Tobias Eggendorfer and Jörg Keller. Preventing spam by dynamically obfuscating email- addresses. 2005.

[19] Tobias Eggendorfer and Jörg Keller. Dynamically blocking access to web pages for spammers' harvesters. In Sanguthevar Rajasekaran, editor, *Communication, Network, and Information Security*, pages 205–210. IASTED/ACTA Press, 2006. ISBN 0-88986-636-8. URL http://dblp.uni-trier.de/db/conf/cnis/cnis2006.html#EggendorferK06.

[20] SA. Eurotux Informática. Eurotux mail exchange. http://eurotux.com/empresa/marketing/eurotux-folheto-etmx-04.pdf, July 2011.

[21] João Fonseca. Net-imap-simple-0.93. `http://search.cpan.org/~jpaf/Net-IMAP-Simple-0.93/`, July 2011.

[22] Apache Software Foundation. Apache http server project. `http://httpd.apache.org/`, July 2011.

[23] Apache Software Foundation. Masses overview. `http://wiki.apache.org/spamassassin/MassesOverview`, October 2011.

[24] Apache Software Foundation. Faster perfomance. `http://wiki.apache.org/spamassassin/FasterPerformance`, October 2011.

[25] Apache Software Foundation. The apache spamassassin project. `http://spamassassin.apache.org`, July 2011.

[26] The Perl Foundation. The perl programming language. `www.perl.org`, July 2011.

[27] Joshua Goodman. Spam: Technologies and policies. 2003.

[28] Joshua Goodman, Gordon V. Cormack, and David Heckerman. Spam and the ongoing battle for the inbox. *Commun. ACM*, 50:24–33, February 2007. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/1216016.1216017. URL `http://doi.acm.org/10.1145/1216016.1216017`.

[29] John Graham-Cumming. Tricks of the spammer's trade. *Hakin9*, 2004.

[30] The PHP Group. Php: Hypertext preprocessor. `www.php.net`, July 2011.

[31] Marek Handl. Spam identification independent of email body contents. Master's thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, Department of Computer Science and Engineering, 2010.

[32] Twitter Inc. Twitter: Widgets. `http://twitter.com/about/resources/widgets`, October 2011.

[33] Twitter Inc. Twitter. `http://twitter.com`, October 2011.

[34] John Ioannidis. Fighting spam by encapsulating policy in email addresses. In *NDSS*. The Internet Society, 2003. ISBN 1-891562-16-9, 1-891562-15-0.

[35] John P. John, Alexander Moshchuk, Steven D. Gribble, and Arvind Krishnamurthy. Studying spamming botnets using botlab. In *Proceedings of the 6th USENIX symposium on Networked systems design and implementation*, pages 291–306, Berkeley, CA, USA, 2009. USENIX Association. URL `http://portal.acm.org/citation.cfm?id=1558977.1558997`.

[36] Chris Kanich, Christian Kreibich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. Spamalytics: an empirical analysis of spam marketing conversion. 2008.

[37] Adam Kennedy. Config-tiny-2.14. `http://search.cpan.org/~adamk/Config-Tiny-2.14/`, July 2011.

[38] J. Klensin. Simple Mail Transfer Protocol. RFC 2821 (Proposed Standard), April 2001. URL `http://www.ietf.org/rfc/rfc2821.txt`. Obsoleted by RFC 5321, updated by RFC 5336.

[39] Christian Kreibich, Chris Kanich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. On the spam campaign trail. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, pages 1:1–1:9, Berkeley, CA, USA, 2008. USENIX Association. URL `http://portal.acm.org/citation.cfm?id=1387709.1387710`.

[40] Kirill Levchenko, Andreas Pitsillidis, Neha Chachra, Brandon Enright, Márk Félegyházi, Chris Grier, Tristan Halvorson, Chris Kanich, Christian Kreibich, He Liu, Damon McCoy, Nocholas Weaver, Vern Paxson, Geoffrey Voelker, and Stefan Savage. Click trajectories: End-to-end analysis of the spam value chain. In *IEE Symposium on Security and Privacy*, 2011.

[41] MAAWG. Email metrics program: The network operators' perspective - report #14 – third and fourth quarter 2010. Technical report, Messaging Anti-Abuse Working Group, March 2011.

[42] Inc MaxMind. Geolocation and online fraud prevention from maxmind. `http://www.maxmind.com/`, October 2011.

[43] Marc Mims. Net-twitter-3.18001. `http://search.cpan.org/~mmims/Net-Twitter-3.18001/`, July 2011.

[44] Infinite Monkeys. Wpoison. `http://www.monkeys.com/wpoison`, July 2011.

[45] NISCC. Botnets - the threat to the critical national infrastructure. Technical report, National Infrastructure Security Co-ordination Centre, October 2005.

[46] OpenVZ. Openvz linux containers. `http://wiki.openvz.org`, October 2011.

[47] Patrick Pantel and Dekang Lin. Spamcop: A spam classification and organization program. In *AAAI Technical Report WS-98-05*, page 1, 1998.

[48] Perl.org. Perl dbi. `http://dbi.perl.org`, July 2011.

[49] S.L. Pfleeger and G. Bloom. Canning spam: Proposed solutions to unwanted email. *Security Privacy, IEEE*, 3(2):40 – 47, march-april 2005. ISSN 1540-7993. doi: 10.1109/MSP.2005.38.

[50] Andreas Pitsillidis, Kirill Levchenko, Christian Kreibich, Chris Kanich, Geoffrey M. Voelker, Vern Paxson, Nicholas Weaver, and Stefan Savage. Botnet judo: Fighting spam with itself. 2010.

[51] Postfix.org. The postfix home page. `http://www.postfix.org`, July 2011.

[52] Matthew B. Prince, Lee Holloway, Eric Langheinrich, Benjamin M. Dahl, and Arthur M. Keller. Understanding how spammers steal your e-mail address: An analysis of the first six months of data from project honey pot. In *Second Conference on Email and Anti-Spam (CEAS 2005)*, 2005.

[53] CentOS Project. Centos: The community enterprise operating system. `http://www.centos.org/`, July 2011.

[54] Niels Provos. Developments of the honeyd virtual honeypot. `www.honeyd.org`, July 2011.

[55] Calton Pu and Steve Webb. Observed trends in spam construction techniques: A case study of spam evolution. 2006.

[56] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '06, pages 291–302, New York, NY, USA, 2006. ACM. ISBN 1-59593-308-5. doi: http://doi.acm.org/10.

1145/1159913.1159947. URL http://doi.acm.org/10.1145/1159913.1159947.

[57] Frank Rietta. Spam and the ongoing battle for safe communications, 2007.

[58] Irina Rish. An empirical study of the naive Bayes classifier. In *IJCAI-01 workshop on Empirical Methods in AI*, 2001. URL http://www.intellektik.informatik.tu-darmstadt.de/\~{}tom/IJCAI01/Rish.pdf.

[59] Michael Schilli. Www-mechanize-plugin-phpbb. http://search.cpan.org/~mschilli/WWW-Mechanize-Plugin-phpBB/, July 2011.

[60] Alexander K. Seewald and Wilfried N. Gansterer. On the detection and identification of botnets. *Computers & Security*, 29(1): 45 – 58, 2010. ISSN 0167-4048. doi: DOI:10.1016/j.cose.2009.07.007. URL http://www.sciencedirect.com/science/article/pii/S0167404809000820.

[61] Craig A. Shue, Minaxi Gupta, Chin Hua Kong, John T. Lubia, and Asim S. Yuksel. Spamology: A study of spam origins. In *Sixth Conference on Email and Anti-Spam (CEAS 2009)*, 2009.

[62] Ricardo Signes. Email-simple-2.100. http://search.cpan.org/~rjbs/Email-Simple-2.100/, July 2011.

[63] Emarketing Solutions. Buy email lists. http://www.americaint.com/bulk-email-lists/buy-email-lists.html, July 2011.

[64] Inc. Twitter. Twitter developers. https://dev.twitter.com, July 2011.

[65] Inc Unspam Technologies. Project honey pot - top harvester countries. http://www.projecthoneypot.org/harvester_top_countries.php, July 2011.

[66] Inc Unspam Technologies. Project honey pot - top spam server countries. http://www.projecthoneypot.org/spam_server_top_countries.php, July 2011.

[67] Jesse Vincent. Www-mechanize-1.70. http://search.cpan.org/~jesse/WWW-Mechanize-1.70/, July 2011.

[68] Zhe Wang, William Josephson, Qin Lv, Moses Charikar, and Kai Li. Filtering image spam with near-duplicate detection. In *Fourth Conference on Email and Anti-Spam, CEAS 2007*, 2007.

[69] Tom Wilde, Kyle Morton, Yuliya Lobacheva, Nina Zinovieva, and Marie Meteer. Search engine optimization, September 2009. URL `http://www.freepatentsonline.com/y2009/0240674.html`.

[70] M. Wong and W. Schlitt. Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1. RFC 4408 (Experimental), April 2006. URL `http://www.ietf.org/rfc/rfc4408.txt`.

[71] Mengjun Xie, Heng Yin, and Haining Wang. An effective defense against email spam laundering. In *Proceedings of the 13th ACM conference on Computer and communications security*, CCS '06, pages 179–190, New York, NY, USA, 2006. ACM. ISBN 1-59593-518-5. doi: http://doi.acm.org/10.1145/1180405.1180428. URL `http://doi.acm.org/10.1145/1180405.1180428`.