MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA

TEN DIEGO MILHOMEM SCHMITT
TEN JOÃO LIMA SANTANELLI

DETECÇÃO DE BOTNETS

Rio de Janeiro
2012

INSTITUTO MILITAR DE ENGENHARIA

TEN DIEGO MILHOMEM SCHMITT

TEN JOÃO LIMA SANTANELLI

DETECÇÃO DE BOTNETS

Projeto de Final de Curso do Curso de Engenharia de Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Engenheiro de Computação.

Orientador: Sergio dos Santos Cardoso Silva–M.Sc.

Rio de Janeiro

2012

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 – Praia Vermelha
Rio de Janeiro - RJ CEP: 22290-270

000.000 Schmitt, Diego Milhomem. Santanelli, João Lima.
S000m Detecção de Botnets/ Diego Milhomem Schmitt, João Lima Santanelli. - Rio de janeiro: Instituto Militar de Engenharia, 2011.

p.: il.

Projeto de Final de Curso – Instituto Militar de Engenharia, 2012.

1. . 2. .3.. 4. . I. Schmitt, Diego Milhomem. II. Santanelli, João Lima. III. Instituto Militar de Engenharia.

CDD 000.000

INSTITUTO MILITAR DE ENGENHARIA

TEN DIEGO MILHOMEM SCHMITT
TEN JOÃO LIMA SANTANELLI

DETECÇÃO DE BOTNETS

Projeto de Final de Curso do Curso de Engenharia de Computação do Instituto Militar de Engenharia, como requisito para colação de grau no curso de Engenharia de Computação.

Orientador: Sergio dos Santos Cardoso Silva – M.Sc.

Aprovada em 25 de maio de 2012 pela seguinte Banca Examinadora:

_____

Sergio dos Santos Cardoso Silva– M.Sc. do IME

_____

Anderson Fernandes P. dos Santos– D.Sc. do IME

_____

Julio Cesar Duarte – D.Sc. do IME

Rio de Janeiro

2012

AGRADECIMENTOS

Dedicamos este trabalho às nossas famílias que, sempre ao nosso lado, nos auxiliam de forma incondicional, dando a nossas vidas um sentido maior. Aos nossos colegas, que nos apóiam tanto nos bons momentos quanto nos mais difíceis. E, principalmente, ao nosso orientador Maj Cardoso, por guiar nossos passos durante toda esta pesquisa, abrindo para nós as portas do conhecimento.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

Bit	Binary Digit
CPF	Cadastro de Pessoa Física
CPU	Central Processing Unit
C&C	Comando e Controle
DDoS	Distributed Denial-of-Service
DNS	Domain Name System
HTML	HyperText Markup Language
IP	Internet Protocol
IRC	Internet Relay Chat
OSI	Open Systems Interconnection
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
Web	World Wide Web

RESUMO

Botnets são redes de máquinas infectadas por softwares maliciosos que atendem a instruções de um servidor de comando e controle. Estas redes podem ser usadas para realizar diversos tipos de ataques a servidores e a máquinas terminais. O principal dentre estes ataques é o de negação de serviço, que consiste na "inundação" de um servidor alvo pelos acessos de um número muito grande de máquinas clientes, o que acarreta um grande prejuízo na prestação de determinado serviço ou até mesmo a queda do servidor.

Para conter o avanço destas comunidades de máquinas "zumbis", faz-se necessário o estudo de suas características de forma a desenvolver uma metodologia para detectar a existência das mesmas para que, posteriormente, se possa combatê-las.

Esse projeto tem por finalidade desenvolver uma técnica de detectar *botnets* a partir da análise de um tráfego da internet através de propriedades de grafos de dispersão de tráfego (TDGs) e estabelecer uma metodologia para identificar elementos como *bots* e centros de C&C, levando em consideração características especificas apresentadas por essas redes e dados estatísticos sobre as mesmas.

ABSTRACT

Botnets are networks of machines infected with malicious software that meet the instructions of a command and control server. These networks can be used to perform various types of attacks on machines and terminal servers. Chief among these attacks is the Denial of Service, which is the "flood" of a target server for access by a large number of client machines, which causes great harm in providing a particular service or even the fall of server.

To contain the spread of these communities machinery "zombie", it is necessary to study their characteristics in order to develop a methodology to detect the existence of the same so that, later, you can fight them.

This project aims to develop a technique to detect botnets from the analysis of internet traffic through the properties of graphs of dispersion of traffic (TDGs) and establish a methodology for identifying elements such as bots and centers of C & C, taking into account characteristics presented by these specific networks and statistical data on them.

1 INTRODUÇÃO

O espaço cibernético mundial sofre, constantemente, com a ameaça dos mais diversos tipos de ataque por parte de *hackers*. Ao passo que os mecanismos de defesa de redes evoluem, nota-se, também, uma diversificação dos tipos de ataque, gerando um ciclo que, provavelmente nunca terá fim.

Um tipo de ataque que vem sendo bastante utilizado ultimamente é o *ataque de negação de serviço*, que consiste em sobrecarregar um servidor alvo impedindo ou dificultando sua comunicação. Para isso, são utilizadas redes de computadores infectados com softwares maliciosos, conhecidas como *botnets*.

Este projeto consiste em desenvolver uma ferramenta capaz de detectar de forma eficiente este tipo de comunidade. Para isso, serão abordados aspectos sobre grafos, cujas propriedades permitem um estudo detalhado sobre o comportamento das máquinas dentro de uma rede. Também serão estudadas as características gerais das *botnets*, bem como seus tipos e protocolos utilizados.

1.1 MOTIVAÇÃO

Inicialmente, o ato de "*hackear*" sistemas era motivado principalmente pelo vandalismo e pelo reconhecimento dentro da comunidade de "*hackers*". Entretanto, com o passar do tempo, é possível notar que tal prática tem sido utilizada para os mais diversos fins, dentre os quais podemos citar a guerra cibernética, que se utiliza de ataques aos servidores do inimigo para obter informações confidenciais, bem como para dificultar a utilização dos mesmos. Esta última finalidade pode ser alcançada através de um ataque de negação de serviço, cujo emprego vem crescendo cada vez mais.

Nos tempos atuais, uma das maneiras mais fáceis de obter acesso aos dados de um indivíduo, é através de seu computador. Nele, são informados desde dados pessoais, como nome,

endereço, CPF e data de nascimento, até senhas dos mais variados tipos de serviço (e-mail, conta bancária, etc.).

Para tal, existem inúmeros programas maliciosos que coletam esses dados e os enviam à fonte interessada, violando a segurança do usuário sem que este sequer tome conhecimento.

Um tipo de ataque muito comum nos dias de hoje é o ataque de negação de serviço (*DDoS*), que utiliza o acesso mútuo de várias máquinas clientes para congestionar o acesso a um servidor alvo, fazendo com que seja impossível que este ofereça seus serviços normalmente.

Assim, surgem as *botnets*, que são redes formadas por máquinas infectadas que se comunicam com um servidor de comando e controle enviando-lhes informações e recebendo suas instruções. Estas máquinas "zumbis" podem ser usadas para atacar outros alvos, sendo assim uma ferramenta importantíssima na disseminação de *spams*.

É evidente a necessidade do estudo de uma ferramenta capaz de investigar a existência dessas redes de softwares nocivos, já que não seria possível eliminá-las sem antes detectá-las. Para tal deve ser feito o monitoramento do tráfego de pacotes dentro de uma rede.

1.2 OBJETIVO DO TRABALHO

Este trabalho tem por objetivo o desenvolvimento de uma ferramenta capaz de realizar a detecção de *botnets* a partir da análise de um registro de tráfego (trace) aplicando, para tal, propriedades de grafos e empregando medidas estatísticas para levantar características de um comportamento *bot*.

1.3 METODOLOGIA

Na primeira etapa do trabalho, foi realizado um estudo sobre *botnets*. Nesta fase, que funcionou como uma ambientação ao projeto, foram estudadas as características gerais deste tipo

de rede. Foi realizada, também, uma revisão sobre os protocolos de comunicação HTTP, IRC e *peer-to-peer*, que são mais comumente utilizados pelas *botnets*.

Em seguida, foram extraídas de um *trace* informações sobre o tráfego de pacotes em uma rede. Estas informações foram filtradas de acordo com os protocolos de comunicação utilizados e, através do IP de origem e do IP de destino de cada pacote, foram estabelecidas as relações de contato entre os membros da rede.

Com base nesses dados, foram construídos os grafos de dispersão de tráfego para representar a comunicação entre os nós da rede. No capítulo 3, serão mostrados alguns conceitos importantes sobre grafos, bem como algumas de suas propriedades. Inicialmente, foram utilizados grafos de contato, que são grafos orientados de arestas ponderadas, nos quais a direção de cada aresta representa a intenção de contato inicial, enquanto seu peso representa o número de interações entre os nós envolvidos. Estes grafos foram analisados com base nas características topológicas das *botnets* centralizadas.

A partir dos grafos de contato, foram obtidos grafos de relacionamento, que facilitam a detecção de grupos de nós que apresentam contatos em comum.

Com base nas interações entre os nós da rede, geraram-se dados estatísticos, que possibilitaram a busca por elementos que apresentassem comportamento suspeito.

Foi criado um aplicativo que, a partir dos dados de um *trace*, consegue, empregando a metodologia acima descrita, informar quais elementos dentro da rede são suspeitos de pertencerem a uma *botnet*.

1.4 ESTRUTURA DO TEXTO

Este relatório é composto pelos seguintes capítulos:

a) O Capítulo 1 faz uma introdução, contendo a motivação, os objetivos e a metodologia utilizada para a concretização do projeto.

b) O Capítulo 2 apresenta um estudo sobre *botnets*, abordando suas principais características, como arquitetura e protocolos de comunicação utilizados. Descreve também os principais tipos de ataques realizados e os principais mecanismos utilizados na detecção dessas redes maliciosas.

c) O Capítulo 3 apresenta os conceitos relativos à Teoria dos Grafos que se fazem necessários para o entendimento da representação das interações entre os nós de uma rede como grafos de contato e de relacionamento.

d) O Capítulo 4 mostra como os dados relativos ao tráfego de pacotes em uma rede podem ser transformados em um grafo para facilitar a compreensão do modo como os nós se comunicam entre eles.

e) O Capítulo 5 descreve as diversas etapas do desenvolvimento do projeto, desde a aquisição do *trace* até o desenvolvimento da ferramenta de detecção.

f) O Capítulo 6 apresenta as conclusões sobre o projeto, bem como sugestões para o aperfeiçoamento da ferramenta desenvolvida.

2 BOTNETS

2.1 DEFINIÇÃO

O termo *bot*, derivado de "*robot*" (que significa "trabalhador" no idioma tcheco), é usado para descrever um processo computacional que funciona de forma automática.

Na área de jogos, tem-se um exemplo clássico do emprego de *bots* quando se pensa nos oponentes simulados pela máquina, que possuem uma inteligência artificial que procura emular o comportamento humano.

No entanto, os *bots* nem sempre são usados de forma positiva. Através deles é possível, por exemplo, realizar ataques distribuídos a um servidor. Às redes compostas por máquinas que abrigam *bots* maliciosos que normalmente exploram falhas de segurança num sistema operacional para permitir que sejam controlados remotamente, recebendo e respondendo a mensagens de um ou mais centros de comando e controle, onde se pode identificar a figura do *botmaster* que é o individuo que detém o controle da rede e administra e define o que ela deve fazer. Assim, surgiu o termo *botnet* que resume em sua idéia uma rede de máquinas zumbi.

O software mal-intencionado, quando instalado em um computador, deixa-o sob o comando de um controlador central, que é capaz de enviar instruções a todas as máquinas infectadas, coordenando, assim, o ataque.

Uma das formas de propagação das *botnets* é o envio de e-mails maliciosos contendo algum tipo de software malicioso. Em geral, o e-mail contém algo que desperte o interesse de quem o recebe. Contudo, o conteúdo em anexo possui um arquivo executável disfarçado que, ao ser ativado, instala o *bot* no computador, tornando-o uma máquina zumbi.

As *botnets* possuem diversas aplicações. Uma delas é a realização de ataques de negação de serviço. Estes consistem no acesso simultâneo de inúmeros *bots* ao servidor alvo, que acaba ficando sobrecarregado face ao grande número de acessos. Dessa forma, o servidor fica impossibilitado de responder às solicitações de outros clientes, acarretando a falha na prestação do serviço.

Outra aplicação é o roubo de informações confidenciais. Neste caso, ocorre a disseminação de programas espiões através dos *bots*. Estes programas são responsáveis por armazenar todo o tipo de informação que o usuário digita no computador e enviá-las ao servidor de comando e controle.

Um aspecto que pode ser facilmente observado é que sempre haverá a comunicação entre os *bots* e o servidor de comando e controle. Dessa forma, todos os *bots* devem trocar mensagens direta ou indiretamente com esta máquina. Tal fato motiva a abordagem do problema através da confecção de um grafo de relacionamento entre as máquinas que fazem parte da rede.

A análise do grafo pode fornecer evidências de qual das máquinas atua como centro de comando e controle. Este, normalmente, acaba sendo representado por um nó com várias arestas, já que tem de se comunicar com um grande número de *bots*. Além disso, é possível tentar identificar os *bots* a partir das comunidades.

2.2 CLASSIFICAÇÃO DAS BOTNETS

As *botnets* podem ser caracterizadas de acordo com o tipo de comunicação entre os *bots* pertencentes à mesma rede e entre *bots* e o centro de comando e controle. Com isso, podem ser identificados três principais tipos de topologia para representar uma determinada *botnet*.

2.2.1 MODELO CENTRALIZADO

É o modelo predominante entres as *botnets*, dentre as quais se pode citar o *AgoBot,* o *SDbot*, *RBotI*e o *EvilBo*t, que usam predominantemente a arquitetura centralizada, segundo SAHA (2005, p. 5). Esse tipo de *botnet* utiliza um nó central para realizar a comunicação com os demais *bots*. Geralmente, quando computador de uma vítima é infectado, o *bot* realiza contato com o C&C para aguardar comandos e instruções.

Podem ser destacados três principais aspectos que viabilizam a utilização de *botnet*s centralizadas. Por possuir uma grande quantidade ferramentas de software para dar suporte à implementação, esse modelo apresenta uma codificação bastante simples o que facilita alterações e aprimoramentos. Pelo tipo de sua arquitetura, *botnet*s assim construídas podem coordenar um número de *bots* o que certamente reflete num potencial de lucro financeiro maior em atividades ilegais, fazendo com que esse modelo apresente uma aceitação alta.

Um segundo aspecto é que ainda poucas ações para combater *botnet* são tomadas oferecendo certa liberdade para a utilização desse modelo. Por último, a latência na troca de mensagens é bem pequena o que facilita o controle dos *bots* e possíveis ataques.

Em contrapartida, *botnets* centralizadas apresentam problemas clássicos herdados desse tipo de arquitetura. A presença de um único nó para realizar a comunicação poder comprometer toda a *botnet* em caso de falhas ou na desativação do mesmo. Vale ressaltar que uma possível sobrecarga do mesmo pode impedir o bom funcionamento de uma *botnet*. A FIG.3.1 exemplifica um modelo de *botnet* centralizado baseado numa arquitetura IRC representando um ataque de negação de serviço.

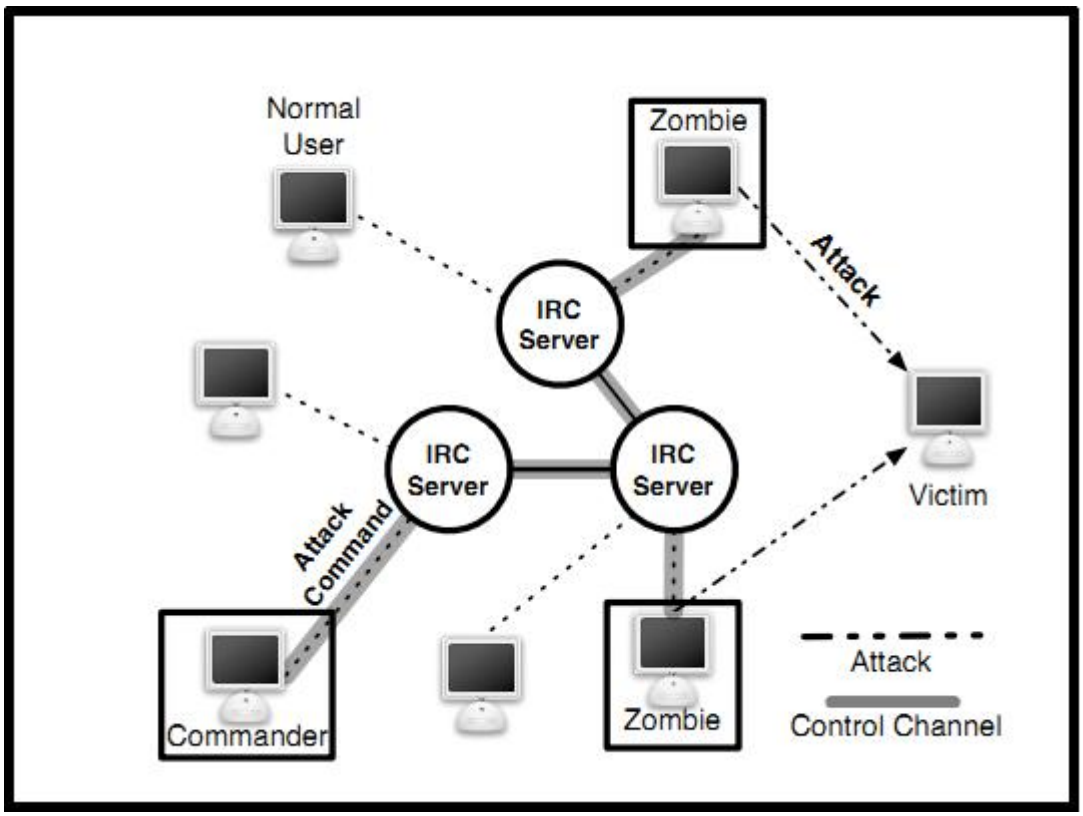

FIG.2.1. *Botnet* centralizada utilizando servidores IRC para realizar um ataque de *DDoS*, COOKE (2005, p.4).

2.2.2 BASEADA NO MODELO PEER-TO-PEER

O modelo P2P surgiu como uma alternativa ao modelo anteriormente apresentado, partindo do princípio de que não há um servidor central que possa ser desativado comprometendo toda rede. Esse modelo tem como paradigma a comunicação P2P que herda as características de uma rede menos suscetível a falhas. Esse tipo de *botnet* se apresenta em número menor comparada com as centralizadas.

Entretanto, por ter um sistema menos dependente de um ou poucos servidores de comando e controle, esse modelo é bem mais difícil de ser detectado e também destruído, visto que mesmo eliminando um ou mais servidores de C&C não se pode garantir que a *botnet* foi

destruída totalmente. Por outro lado, nessa arquitetura não se pode ter garantias de latência nem de confiabilidade na troca das mensagens.

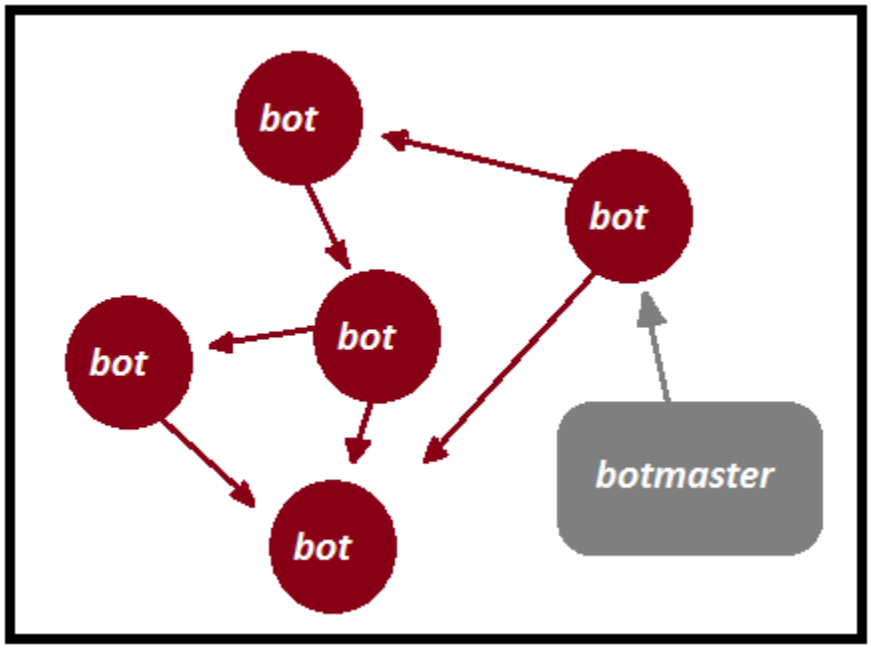

FIG.2.2. Ilustração de uma *botnet* descentralizada.

2.2.3 MODELO ALEATÓRIO (RANDOM MODEL)

Esse modelo parte do principio que nenhum *bot* conhece mais que outro *bot* pertencente à rede. Para se comunicarem, faz-se necessário uma procura na rede para detectar a presença de *bots* que ficam a espera de uma conexão com o seu *botmaster*. Somente assim é possível a troca de mensagens que são feitas geralmente utilizando algum algoritmo de criptografia. Este tipo de *botnet* é bastante simples e apresenta altas chances de sobrevivência, pois se algum *bot* for identificado, isso não comprometeria a rede. Entretanto o modelo aleatório apresenta problemas como a falta de garantia na troca de mensagens e latência na rede.

A tabela abaixo mostra um comparativo entre as topologias das *botnet*s levando em conta alguns aspectos como facilidade de detecção, capacidade de sobrevivência, latência e complexidade de implementação de acordo com COOKE (2005, p.5).

TAB.2.1. Comparação entre os padrões de C&C de *botnet*s, segundo COOKE (2005, p.5).

Topologia	**Complexidade**	**Detecção**	**Latência**	**Sobrevivência**
Centralizada	Baixa	Média	baixa	baixa
P2P	Média	Baixa	média	média
Aleatória	Baixa	Alta	alta	alta

2.3 PRINCIPAIS PROTOCOLOS DE COMUNICAÇÃO EMPREGADOS EM BOTNETS

Entender os protocolos de comunicação utilizados pelas *botnet*s é importante para identificar possíveis ferramentas de software utilizadas pelo *malware* e ajudar na decodificação das conversas entre *bots* e seus mestres. Usualmente *botnet*s não se comunicam por protocolos próprios, mas sim por protocolos públicos bem conhecidos e com uma vasta disponibilidade de software que os implementam.

2.3.1 PROTOCOLO IRC (INTERNET RELAY CHAT)

Foi um protocolo criado para dar suporte a um sistema de teletexo comunitário com recursos avançados como conversas públicas de usuários separadas por canais e troca de mensagens privadas entre eles.

É um dos protocolos mais utilizados na maioria das *botnet*s encontradas. Sua estrutura foi desenvolvida utilizando o protocolo TCP/IP como base e é utilizado para comunicação entre

cliente-servidor e servidor-servidor. O protocolo IRC é divido em três principais conceitos de comunicação. O primeiro deles é o um-para-um (*one-to-one*) utilizado para troca de mensagens privadas o que é útil numa *botnet* quando um *botmaster* quiser enviar um comando para um *bot* em especial.

O segundo é o muitos-para-um onde um usuário (ou servidor) pode enviar mensagens para uma determinada lista de usuários, ou também para um determinado grupo de usuários (ou servidores) que estão em um mesmo canal (é equivalente a mensagens em multicast) que é muito utilizado para enviar comandos para uma grande quantidade de *bots*.

O terceiro é o todos-para-um. Similar ao muitos-para-um, entretanto é mais bem definido como uma espécie de mensagem em broadcast enviada para todos os clientes, servidores ou os dois. O funcionamento de uma *botnet* rodando em servidor IRC é bastante simples e similar a qualquer outro servidor IRC. Um *botmaster* cria um canal no qual os *bots* irão se conectar ao *botmaster* que poderá enviar comandos e instruções.

2.3.2 PROTOCOLO HTTP

O *HyperText Transfer Protocol* é um protocolo da camada de aplicação do modelo *OSI* que se destina ao tratamento requisições e respostas entre clientes e servidores. Normalmente esse protocolo utiliza a porta 80 para a comunicação de sites Web, que são programados em na linguagem HTML (*HyperText Markup Language*).

Um dos meios de comunicação também bastante popular entre as *botnet*s por oferecer vantagens frente ao IRC. Uma delas é que o próprio IRC se tornou conhecido por ser utilizados por *botnets* e, consequentemente, monitorado nas redes dificultando a comunicação. Basta que se configure o Firewall para bloquear portas que incluam as portas IRC (geralmente a porta 6667). Assim, pode-se utilizar o protocolo HTTP para burlar essas políticas de segurança uma vez que fica mais difícil detectar o tráfego de uma *botnet* entre o enorme tráfego HTTP normal.

2.3.3 PROTOCOLO PEER-TO-PEER

É caracterizado pelo fato de um nó pertencente a rede poder ser considerado tanto como cliente como servidor e marcada por uma arquitetura de sistemas distribuídos.

Algumas variações mais avançadas de *botnet*s utilizam o protocolo P2P (peer-to-peer) para se comunicarem e realizarem a troca de arquivos, em que algumas chegam até a utilizar dados com algum tipo de criptografia. O número de *botnet*s baseadas nesse protocolo são menores, contudo, por oferecer maior dificuldade de detecção, estima-se que seja o rumo das *botnet*s para o futuro.

O sistema de comando e controle (C&C) é uma parte importantíssima de uma *botnet* que concentra a responsabilidade do bom funcionamento operacional de sua rede e, também, acredita-se ser o elo mais fraco do sistema, pois uma interrupção na comunicação entre C&C e seus *bots* ou então encerramento de uma central de comando e controle ativa impede que o *botmaster*, que é o individuo que controla a *botnet*, perca a capacidade de comunicação e de coordenação.

Além disso, o entendimento de como uma *botnet* funciona é fundamental para que seja possível o desenvolvimento de ferramentas de combate eficientes, explorando suas características e principais fraquezas. A seguir serão abordados três principais modelos que podem ser utilizadas por uma *botnet*.

2.3.4 DNS

O Sistema de Domínio de Nomes (DNS) é um sistema que tem por objetivo proporcionar um mecanismo de nomeação tal forma que esses nomes possam ser utilizáveis em diferentes hospedeiros, redes, protocolos, internets e organizações. É caracterizado por sua estrutura hierárquica e distribuída a qual fornece um serviço transparente para o usuário.

Na rede mundial de computadores, podem-se identificar dois tipos principais de espaço de nomes: endereços IP e os nomes de domínio. O serviço prestado por um servidor de DNS é responsável por guardar a relação entre esses dois espaços de nomes.

O serviço DNS é composto por uma base de dados distribuída numa hierarquia de servidores de nomes organizada em uma hierarquia de domínios. É um protocolo da camada de aplicação que permite traduzir nomes em endereços IP e também o contrário. Utiliza como padrão a porta 53 e o protocolo de transporte UDP para consultas.

Esse protocolo é particularmente importante, pois devido à necessidade de comunicação entre bots e C&C, no caso de botnets centralizadas, faz com que os bots tenham alguma informação sobre de como se conectar ao mesmo. Isso pode ser feito de duas maneiras, segundo PLOHMANN (2011):

- Um IP fixo pré-definido no código fonte do *bot*.
- Um nome de domínio ou um conjunto de nomes a ser consultado quando uma máquina é infectada.

Quando um *bot* utiliza um nome de domínio é necessário para estabelecer a comunicação resolver o nome em questão para se descobrir o *IP* do servidor de comando. Assim, ao capturar um tráfego de rede, pode-se analisar os pacotes DNS para coletar informações sobre os nomes de domínio que uma determinada máquina está tentando resolver e identificar um possível C&C. Para isso, deve-se inspecionar o *payload* do pacote DNS. Esse *payload* segue um formato padrão subdividido em cinco partes, de acordo com a FIG.2.2, das quais algumas podem ser vazias dependendo da mensagem, conforme em MOCKAPETRIS (1987).

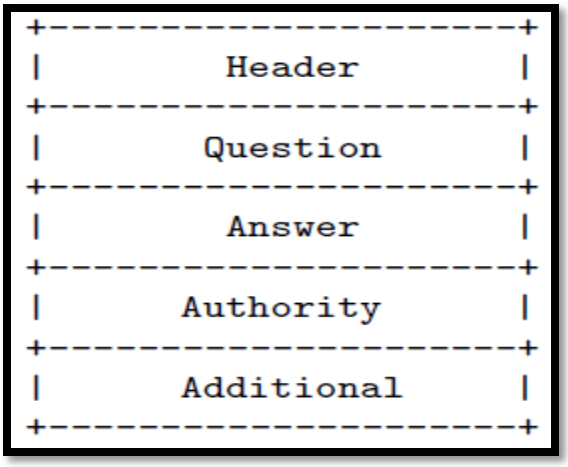

FIG.2.3. Estrutura de um *payload* de pacote DNS, MOCKAPETRIS (1987).

O campo *Header* é divido em seis partes de 16 bits cada, conforme a FIG.2.3. Ele especifica como os outros campos são apresentados, indica se uma mensagem é uma resposta ou uma pergunta, indica o tipo de consulta se é a padrão ou alguma outra. A tabela TAB.2.2 detalha cada um dos campos presente no Header.

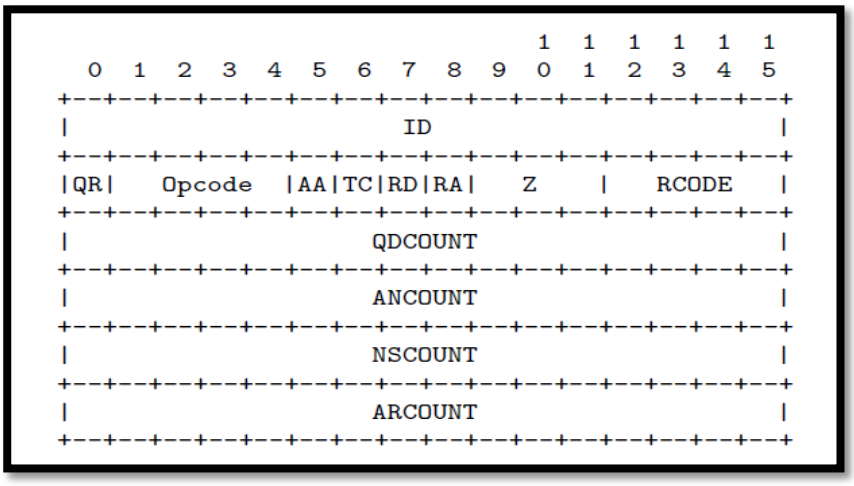

FIG.2.4. Formato do cabeçalho (*Header*) do *payload* de um pacote DNS, MOCKAPETRIS (1987).

TAB.2.2. Descrição do cabeçalho.

Campo	Significado
ID	Identificador de 16 bits.
QR	Especifica se é uma pergunta (0) ou resposta (1)
OPCODE	Especifica o tipo de consulta: 0-padrão, 1-reversa, 2-complementar, 3-15 reservado para uso futuro.
AA	O bit valido indica uma resposta autoritária.
TC	Indica truncamento.
RD	Indica o pedido de uso de recursão.
RA	Indica se consultas recursivas estão disponíveis ou não.
Z	Campo reservado.
RCODE	Tipo de resposta: 0-sem erro, 1- erro no formato, 2-falha no servidor, 3-nome inexistente, 5-recusado.
QDCOUNT	Indica o número de perguntas no campo de pergunta.
ANCOUNT	Indica o número de respostas no campo de respostas.
NSCOUNT	Indica o número de NS no campo de autoridades.
ARCOUNT	Indica o número de informações adicionais presente no campo *Additional.*

O campo de pergunta (*Question*) é dividido em três seções conforme a FIG.2.4. A primeira (*QNAME*) seção é de tamanho variável e contem o nome de domínio a ser consultado. A segunda seção (*QTYPE*) de tamanho fixo de 16 bits especifica o tipo de consulta e a terceira seção (*QCLASS*) especifica o tipo de classe, normalmente é IN (internet).

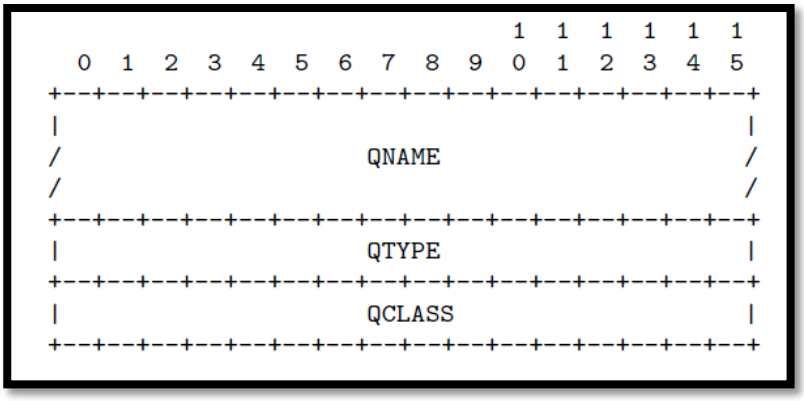

FIG.2.5. Formato do campo pergunta (*Question*) do *payload* de um pacote, MOCKAPETRIS (1987).

Os demais campos: resposta (*Answer*), autoridades (*Authority*) e informações adicionais (*Additional*) que compõem a parte de dados do pacote DNS, possuem um mesmo formato apresentado pela TAB.2.3.

TAB.2. 3. Descrição dos campos

Campo	Significado
Name	Nome de domínio aos quais as informações pertencem (tamanho variável).
Type	Indica o tipo da informação (16 bits).
CLASS	Especifica a classe das informações no campo RDATA (16 bits).
TTL	Time to Live, dado em segundos (32bits).
RDLENGTH	Tamanho do campo RDATA (16 bits).
RDATA	Campo de tamanho variável que descreve as informações de acordo com o tipo e a classe.

2.4 ATAQUES REALIZADOS POR BOTNETS

Serão abordadas nesta seção algumas atividades peculiares que podem ser realizadas por uma *botnet*. A primeira delas consiste no recrutamento de novas "máquinas zumbis". A infecção de novos hospedeiros pode ocorrer de diversas formas. Uma delas, talvez a mais comum, se dá através da incorporação de um software malicioso em uma mensagem eletrônica. Normalmente, o conteúdo da mensagem induz o usuário a abrir um arquivo executável capaz de instalar o *bot* em seu computador ou a acessar a um *link* que levará à instalação do *malware*.

Ao ser executado, o arquivo instala, na máquina, um programa que será capaz de abrir uma *backdoor* para estabelecer a comunicação com o servidor de comando e controle. Dessa forma, a máquina infectada passa a fazer parte da *botnet*.

De acordo com seu objetivo, as *botnets* podem realizar diferentes atividades. Um objetivo bastante frequente é o roubo de informações confidenciais do usuário. Geralmente, isto é feito através de *keyloggers*, que são programas capazes de identificar as sequências de caracteres digitados pelo usuário. Existem também os *screenloggers*, que, de tempos em tempos (geralmente a cada clique do *mouse*), capturam e armazenam a imagem da tela do computador. Estas informações são enviadas periodicamente ao *botmaster* (indivíduo que opera a *botnet*) através de canais IRC ou endereços de e-mail específicos.

É importante ressaltar que esta atividade pode ser extremamente danosa ao usuário, uma vez que é comum a digitação de senhas e *logins* de contas dos mais variados serviços, que podem variar desde *e-mails* até operações bancárias. O acesso de pessoas mal-intencionadas a tais informações pode acarretar grandes prejuízos a seus proprietários.

Outro tipo de atividade a ser abordado neste estudo é o envio de *spams*. A utilização de *botnets* para a disseminação dos *spams* apresenta algumas vantagens para o *hacker* que deseja realizar o ataque. Primeiramente, o volume de mensagens enviadas por uma *botnet* é muito maior do que se o serviço fosse feito por uma única máquina. Dessa forma, é possível uma propagação muito mais eficaz. Além disso, como o ataque não é realizado diretamente pela máquina de comando e controle, acaba sendo impossível localizá-la através da utilização de meios legais.

Por último, tem-se o ataque de negação de serviço (*DDoS - Distributed Denial of Service*). Este tipo de ataque é o mais antigo dentre os realizados através de *botnets*. Consiste em

coordenar o acesso simultâneo de várias máquinas a um determinado servidor, ocupando grande parte dos recursos de suas CPUs e retardando o tempo de resposta a consultas de outros clientes, ou, até mesmo, provocando a queda do sistema. Vários servidores de grande porte, como o da Microsoft e o da Yahoo! Já foram vítimas desses ataques.

2.5 DETECÇÃO DE BOTNETS

Nesta seção serão abordados alguns tópicos sobre técnicas mais frequentes utilizadas para detectar tráfego *bot* e também se prevenir contra a propagação de programas maliciosos. Pode-se dividi-las em categorias onde se destacam a detecção pelo tráfego de rede, baseada no host e por padrão de comportamento.

2.5.1 DETECÇÃO PELO TRÁFEGO DE REDE

Partindo do princípio que deve haver uma comunicação entre os *bots* e seus *botmasters*, analisar o tráfego de rede em busca de determinados protocolos utilizados como IRC e HTTP empregados de forma não usual poderia indicar a presença de uma *botnet*. Por exemplo, detectar tráfego IRC em lugares onde é proibido, como em meios corporativos, como também tráfego IRC com sintaxe a qual não é usual para um ser humano, e sim para máquina. Observar tráfego SMTP em grande quantidade oriundo de máquinas não destinadas para tal pode indicar a presença de *malwares* realizando spams que são muito comuns em *botnets*.

Outro aspecto também é o fato de muitos *botmaster* utilizarem DNS dinâmico para evitar que sejam descobertos, levando a uma possível suspeita de que DNSs que mudam com alta frequência e contem um padrão anormal de nome podem estar comprometidos. Requisições DNS iguais em um mesmo intervalo de tempo também podem ser um forte indício da existência de máquinas comprometidas tentando se comunicar com um servidor central.

Interceptando um determinado tráfego suspeito e realizando uma análise dos pacotes capturados pode ser possível conseguir informações que levem a descobrir um centro de comando e controle e também a identificar características sobre funcionalidades dessa *botnet*. Entretanto, o uso dessas técnicas vem apresentando uma eficiência limitada pelo uso de técnicas de evasão como criptografia para impedir que se descubra o que se está transmitindo e ofuscação de tráfego para burlar Firewalls e IDS.

Serão descritas, a seguir, algumas técnicas de detecção de *botnets* baseadas na análise do tráfego de rede.

2.5.1.1 INSPEÇÃO DE PACOTES

A inspeção de pacotes consiste na análise dos dados de cada pacote enviado na rede confrontando-os com padrões pré-definidos de atividades maliciosas.

Segundo PLOHMANN (2011, p.41), esta técnica apresenta algumas desvantagens. A primeira é que a análise de todos os pacotes enviados em uma rede apresentaria um custo computacional muito grande, tornando o processo inviável. Além disso, há uma chance elevada de ocorrer falso-positivos, o que compromete a eficiência da rede. Por fim, a detecção é baseada em padrões, que podem ser, eventualmente, burlados. Dessa forma, a segurança do método não é absolutamente confiável.

2.5.1.2 ANÁLISE DOS REGISTROS DE FLUXO

Esta técnica consiste em inspecionar o fluxo da rede com base não nos dados de cada pacote, mas nas características de cada contato. São levadas em conta informações como endereços de origem e destino, porta utilizada para a comunicação, número de pacotes trocados e o protocolo utilizado.

2.5.1.3 ABORDAGEM BASEADA EM *DNS*

Nem sempre uma *botnet* utiliza IP fixo para a comunicação entre seus *hosts*. Muitas vezes, os contatos se dão através de servidores DNS. Isto pode oferecer grandes vantagens à *botnet*. Dentre elas, pode-se citar o fato de que um domínio de nome pode estar associado a múltiplos endereços de IP. Esses endereços não precisam, necessariamente, ser estáticos, podendo variar conforme a demanda, o que torna mais difícil a detecção e a derrubada dos *hosts*.

Além disso, a utilização de servidores DNS, permite a criação de domínios maliciosos, de nomes semelhantes aos de sites conhecidos, fazendo com que novos hosts possam ser infectados em consequência, por exemplo, de um erro de digitação do usuário.

2.5.1.4 RASTREAMENTO DE REDES DE FLUXO RÁPIDO

São chamadas de redes de fluxo rápido aquelas redes que modificam rapidamente seus registros de endereços de IP relacionados a um determinado domínio DNS. Dessa forma, torna-se mais difícil detectar e derrubar seus nós. Geralmente, os nós dessas redes são máquinas infectadas por *bots*, e o *botmaster* é quem tem o controle do servidor DNS. Assim, a descoberta dessas redes, muitas vezes leva a uma *botnet*.

2.5.2 DETECÇÃO PELO HOST

Uma característica importante e que pode contribuir para detecção de *botnets* é a análise com base na máquina do usuário, uma vez que a descoberta de um *bot* pode fornecer informações sobre essa rede de *bots.*

Certas características que são fáceis de verificar como máquina muito lenta, *popups* no browser de navegação surgindo sem explicação, outras como criar conexões de internet,

modificar registros, desativar antivírus, tentativa de compartilhar arquivos, não tão óbvias para um usuário pouco experiente, podem sugerir que esta máquina esta comprometida com algum software malicioso. A partir de momento em que se descobre o *malware* é possível realizar uma análise do mesmo de forma a descobrir algumas informações sobre a *botnet* como algum tipo de chave publica de criptografia, o domínio ou IP o qual a máquina deve se conectar. Esses dados podem ajudar na descoberta de pontos sensíveis como um servidor de C&C e acarretar na desativação da mesma.

Esses métodos, embora sejam eficientes, demandam um grande esforço para identificar e analisar o *malware* devido a técnicas de ofuscação de código empregadas para dificultar o entendimento e também pelo fato de não ser flexível, pois exige, a cada novo *malware*, que o processo seja repetido.

Serão descritas, a seguir, algumas técnicas de detecção de *botnets* baseadas na análise do host.

2.5.2.1 UTILIZAÇÃO DE *HONEYPOTS*

Honeypots são máquinas desprotegidas instaladas em uma rede com o intuito de serem infectadas, para que seja possível estudar as novas práticas e estratégias utilizadas pelos hackers e desenvolvedores de *malware*.

No caso das *botnets*, os *honeypots* podem ser aplicados para descobrir os padrões comportamentais dos membros daquela rede, permitindo que estes sejam, posteriormente, identificados com facilidade.

2.5.2.2 ANÁLISE DO *FEEDBACK* DE *SOFTWARE* ANTIVÍRUS

Uma característica positiva dos antivírus modernos é a comunicação bidirecional entre o cliente e o provedor do antivírus. O *feedback* enviado pelo cliente à empresa fornecedora do

software permite que esta faça uma análise detalhada das condições da máquina, possibilitando a tomada de decisões mais confiáveis. Essas informações também são de grande importância para a empresa, que, com base nos dados obtidos de vários clientes, é capaz de traçar um perfil com as características de grande parte das atividades maliciosas presentes na rede. Dessa forma, é possível detectar as possíveis ameaças e, assim, proteger-se contra elas.

2.5.3 DETECÇÃO PELO PADRÃO DE COMPORTAMENTO

Uma forma diferente de se pensar para detectar *botnet* é compor um conjunto de características comuns às estruturas das *botnet*s. Pode-se citar como exemplo que como a grande parte das *botnet*s utiliza DNS dinâmico, cada vez que for trocado o seu IP para evitar detecção, vai ser notável um número elevado de requisições para resolver o endereço atual do servidor de C&C o que pode indicar que esse domínio esta sendo utilizado para uma *botnet*.

Pode-se também realizar uma análise conjunta onde se emprega técnicas de análise do *malware* junto com inspeção do trafego de rede, formas de propagação e topologia para se identificar *bots* e centros de C&C. Um exemplo de sucesso que ilustra essa técnica foi a desativação da Tequila Botnet uma *botnet* mexicana que continha em torno de dois mil *bots*, utilizada para crimes cibernéticos e ataque *DDoS*, entre outras atividades, segundo ROMERA (2010).

A análise pelo comportamento revela que é mais proveitoso investigar padrões de comportamento que certamente não variam com as técnicas evasivas mencionadas anteriormente e que podem ser aplicadas para qualquer tipo de *botnet* que se encaixe no perfil, permitindo descobrir e eliminar esses tipos de redes.

3 CONCEITOS IMPORTANTES SOBRE GRAFOS

Para a análise do comportamento dos nós de uma rede, devem ser aplicadas ferramentas da Teoria dos Grafos. Serão apresentadas, a seguir, algumas definições importantes para o entendimento do método empregado na detecção de *botnets*.

3.1 MULTIGRAFO

Um multigrafo $G = (V, E)$ é um par ordenado onde V é um conjunto finito não vazio de vértices e E é um conjunto de pares não ordenados de elementos do conjunto V ou arestas.

Também se pode usar a notação $G = (V(G), E(G))$.

3.2 GRAFO

Laço é toda aresta de um grafo $G = (V, E)$ que é do tipo $e = (v, v), v \in V, e \in E$.

Um grafo é um multigrafo que não admite nem arestas repetidas nem laços.

3.3 GRAFO DIRECIONAL OU ORIENTADO

Grafo direcional ou orientado é um grafo $G = (V, E)$ cujas arestas (elementos de E) são pares ordenados. Dessa forma, sejam $e = (u, v)$ e $e' = (v, u)$ arestas de um grafo ordenado G. Então, tem-se que $e \neq e'$.

3.4 SUBGRAFO

Sejam dois grafos $G = (V, E)$ e $G' = (V', E')$. Diz-se que G' é um subgrafo de G se, e somente se, $V' \supseteq V$ e $E' \supseteq E$.

3.5 GRAFO COMPLETO

Um grafo $G(V, E)$ é dito completo se, e somente se, $\forall\, u, v \in V, \exists\, e \in$ E tal que $e = (u, v)$. Ou seja, cada dois vértices do grafo são, necessariamente, ligados por uma aresta.

3.6 CLIQUE

Sejam dois grafos G e G', tais que G' é um subgrafo de G. Se G' é completo, então se diz que G' é um clique de G.

3.7 SUCESSOR

Seja $G = (E, V)$ um grafo orientado. Sejam $u, v \in V$ vértices de G. Diz-se que v é sucessor de u se, e somente se, $\exists\, e \in E$ tal que $e = (u, v)$.

3.8 GRAU DE UM VÉRTICE

Diz-se que uma aresta *e* incide sobre um vértice *v* se, e somente se, $e = (u, v)$, $u \in V$. Seja um vértice *V*, sobre o qual incidem exatamente *n* arestas. Diz-se que *V* tem grau *n*.

3.9 CENTRALIDADE

A centralidade de um vértice está relacionada à sua importância dentro do grafo. Existem cinco medidas de centralidade que podem ser utilizadas na análise de grafos: centralidade de grau *(degree centrality),* centralidade de intermediação *(betweenness centrality),* centralidade de proximidade *(closeness centrality),* centralidade de autovetor *(eigenvector centrality) e PageRank.*

3.9.1 CENTRALIDADE DE GRAU

Relaciona a importância do grafo à quantidade de arestas que incidem sobre ele, ou seja, à quantidade de relacionamentos diretos.

Seja $G = (V, E)$ um grafo com n vértices e $v \in V$ um vértice. A centralidade de grau do vértice *v* na rede, notada por $C_D(v)$, é definida como:

$$C_D(v) = \frac{grau(v)}{n - 1}$$

Onde *n* é o número de vértices do grafo *G* e $grau(v)$ é o grau do vértice *v*.

3.9.2 CENTRALIDADE DE INTERMEDIAÇÃO

Relaciona a importância do vértice à frequência de suas ocorrências nos caminhos mínimos entre cada par de vértices do grafo. Seja $v \in V$ um vértice do grafo G. A centralidade de intermediação de v é dada por:

$$C_B(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

Onde σ_{st} é o número de caminhos mínimos entre s e t, e $\sigma_{st}(v)$ é o número mínimo de caminhos entre s e t que passam por v.

3.9.3 CENTRALIDADE DE PROXIMIDADE

Relaciona a importância do vértice à média de suas distâncias aos demais vértices do grafo. Quanto menor a média, maior a importância. Seja $v \in V$ um vértice de G. A centralidade de proximidade de v é dada por:

$$C_C(v) = \frac{1}{\sum_{t \in V, t \neq v} d_G(v,t)}$$

Onde $d_G(v,t)$ é o comprimento do menor caminho do vértice v ao vértice t.

3.9.4 CENTRALIDADE DE AUTOVETOR

Atribui uma pontuação a cada vértice. Esta pontuação está, também, relacionada às pontuações dos demais vértices. Assim, a centralidade de autovetor mostra a capacidade de um nó da rede de transmitir informações de forma indireta aos demais nós.

Seja $1 \leq i \leq n$, x_i a pontuação do *i*-ésimo vértice do grafo $G = (V, E)$ com n vértices e A, a matriz de adjacência do grafo. Os elementos da matriz representam a importância das relações entre os vértices. Estas relações são representadas pelas arestas do grafo.

Definimos a pontuação do *i*-ésimo vértice do grafo como:

$$x_i = \frac{\sum_{j \in M(i)} x_j}{\lambda}$$

Onde $M(i)$ é o conjunto dos vértices adjacentes ao *i*-ésimo vértice de G e λ é uma constante não nula. Dessa forma, pode-se reescrever a equação acima como:

$$x_i = \frac{\sum_{j=1}^{n} a_{ij} x_j}{\lambda}$$

Escrevendo na forma vetorial, obtém-se:

$$\boldsymbol{x} = \frac{1}{\lambda} A\boldsymbol{x}$$

Este resultado equivale à equação que define o autovetor de uma matriz quando $x \neq 0$:

$$\lambda \boldsymbol{x} = A\boldsymbol{x}$$

A solução da equação anterior determina autovetores diferentes para cada valor de λ possível. Como todas as entradas do autovetor devem ser positivas, tem-se, pelo teorema de

Perron-Frobenius (PRASOLOV, 1994), que somente o maior autovalor da matriz *A* atende à condição dada pela equação. Dessa forma, o autovetor de Perron é usado para definir a medida de centralidade de autovetor, ou seja, a *i*-ésima componente do autovetor de Perron, é a pontuação do *i*-ésimo vértice do grafo *G*.

3.9.5 *PAGERANK*

É uma forma generalizada do valor de centralidade de autovetor. O *pagerank* de um vértice de um grafo *G* é dado pela equação:

$$x = D(D - \alpha A)^{-1}L$$

Onde $A = (a_{ij})$ é a matriz de adjacência de *G*, $D = (d_{ij})$ a matriz diagonal dos vértices de *G*, *L* é um vetor com todos os componentes iguais a um e α é uma constante a ser ajustada pelo usuário (0,85 é um valor frequentemente utilizado).

O *pagerank* é a base do algoritmo de buscas da empresa Google.

3.10 DETECÇÃO DE COMUNIDADES EM GRAFOS

Segundo Fortunato (2010), uma comunidade é um grupo de nós que apresentam propriedades comuns ou desempenham funções similares dentro de uma rede. Uma *botnet*, portanto pode ser classificada como uma comunidade. Desta forma, a análise de propriedades dos grafos para a detecção de comunidades pode ser muito útil para a identificação das redes de *bots*.

Como ferramentas para a detecção de comunidades, cabe destacar a utilização dos algoritmos de Girvan-Newman e de Clauset-Newman-Moore, ambos presentes na biblioteca SNAP (2010), bem como dos algoritmos de Newman e de Fiedler.

4. DETECÇÃO DE BOTNETS COM BASE NA ANÁLISE DE GRAFOS

Conforme abordado na seção de detecção de *botnets*, pode-se entender que frequentemente se procura observar uma determinada característica que levante a suspeita de se estar tratando de uma de uma rede *bot*.

Muitas *botnets* hoje em dia vêm empregando técnicas para esconder seu tráfego, tais como migrar de protocolos já bem conhecidos por serem empregados em *botnets* como IRC, para protocolos mais comuns como HTTP de forma a esconder seu tráfego e também de fugir de políticas de segurança que bloqueiam suas formas mais tradicionais de se comunicar. Conforme em TREND MICRO (2006), pode-se citar a utilização de criptografia na troca de mensagens e também de técnicas de tunelamento como forma de dificultar a inspeção do conteúdo. Com isso, apenas a análise do tráfego de rede, vem se tornando uma ferramenta pouco eficaz diante das técnicas evasivas e da sofisticação emprega nessas redes de *bots*.

Outro aspecto que também serve para identificar *botnets* que apresenta certas limitações é analisar um determinado *malware* e estabelecer padrões de detecção baseados nele. A grande diversidade de programas existentes e a atualização dos mesmos ou o lançamento de outros geraria uma demanda alta de recursos humanos capacitados bem como tempo e investimentos para manter uma base de informações sobre cada um.

A análise pelo padrão comportamental representa uma maneira de não se fixar somente num determinado *malware* ou um conjunto deles e nem numa funcionalidade que a *botnet* possa apresentar. Dessa maneira, esse projeto propõe uma forma de detecção de *botnets* se baseando no relacionamento entre cada membro dessa rede, propondo a utilização de grafos de dispersão de tráfego (*Traffic Dispersion Graphs* - TDGs) para analisar e visualizar o tráfego de uma rede.

4.1 GERAÇÃO E ANÁLISE DE GRAFOS A PARTIR DE UM TRACE

4.1.1 OBTENÇÃO DE UM TRACE

Primeiramente, faz-se necessário capturar um tráfego da internet (*trace*) que possivelmente contenha tráfego oriundo de máquinas pertencentes a uma *botnet*. Isso pode ser feito realizando uma captura de pacotes de um determinado enlace de dados, por exemplo, de um *backbone*, onde se espera encontrar uma grande quantidade de pacotes de diversas máquinas diferentes. Outra característica que pode ser levada em consideração são estatísticas de localidades onde se tem uma grade probabilidade de se encontrar tráfego proveniente de *botnets*, por exemplo, o Brasil é um dos países que lidera o ranking mundial como fonte de Spam, segundo SYMANTEC (2012). Como não há a necessidade de que se seja em tempo real, pode ser utilizado um arquivo que o salve num formato conveniente, para que possa ser posteriormente manipulado.

4.1.2 ANÁLISE DE UM *TRACE*

Para a análise de um *trace*, é necessário que se estabeleça uma maneira de se filtrar o tráfego capturado de modo a salvar tudo o que possivelmente possa conter comunicação relacionada a *botnet*s e descartar o que não desperta interesse de imediato. Assim, pode-se empregar os principais protocolos de comunicação utilizados em redes *bot* como critério para separar as informações a serem analisadas. Como o importante a princípio é somente o relacionamento entre os membros de uma *botnet*, basta que se guarde somente a informação do IP da máquina de origem e o IP da máquina de destino ou o domínio que a represente.

Cada protocolo tem uma determinada especificidade que deve ser observada. Para o protocolo HTTP, a maneira mais simples de se identificar esse tipo de tráfego é observar a porta

em que um determinado pacote trafega. Conforme IANA (2012), a porta padrão de comunicação desse protocolo que é associada a um serviço Web é a porta 80 nos protocolos de transporte TCP e UDP. Para o tráfego IRC a lógica é a mesma mudando somente as portas que vão da 6665 até a 6669, somente no protocolo de transporte TCP. Dessa maneira, para separar o tráfego HTPP e IRC basta analisar o pacote e verificar as porta em que se está comunicando. Para o tráfego DNS, do qual se pode inferir algumas informações sobre a intenção de contato, é feita uma análise parecida também com base na porta que é a de número 53. Sabe-se que existem *botnets* que não utilizam um IP *hard-coded*[1] o que faz com que seus *bot* tenham que buscar saber a qual IP devem se conectar. O motivo de analisar o DNS vem do fato de se poder saber a qual domínio uma determinada máquina está querendo saber o IP. Para isso é necessário que se analise o *payload*[2] do pacote DNS.

Para fins desse projeto, serão usados apenas os campos de cabeçalho (*Header*) e pergunta (*Question*), descritos na seção 2.3.4, que são suficientes para determinar o domínio a ser consultado e o IP que originou a consulta. O campo de cabeçalho, como descrito na FIG.2.3, fornece, através do bit representado por *QR*, se o pacote está carregando uma pergunta (valor zero) ou uma resposta (valor um). Com isso, sabe-se que caso se tenha um pergunta, que o IP de origem representa quem teve a intenção de resolver o domínio, caso contrário o IP de destino. Pelos quatro bits representados pelo campo interno *OPCODE*, pode-se determinar qual tipo de pergunta foi realizada. Esse campo deve assumir o valor um, pois se deseja saber apenas os tipos de perguntas padrão (*standard querry*). Já no campo pergunta como ilustra a FIG.2.4, é possível saber qual foi o domínio que se desejava descobrir o IP. Para tal basta analisar o campo interno *QNAME* e verificar se este realmente representa um endereço da internet, que é indicado pelo valor um nos dois octetos do campo *QCLASS*. Essa análise é necessária, pois caso fizesse a associação entre IP de origem e destino do pacote DNS levaria num mapeamento de um Servidor de Nomes a uma máquina o que não acrescenta nenhuma informação útil sobre relacionamento entre membros de uma rede *bot.*

[1] *Hard-coded*: escrito diretamente no código
[2] *Payload*: carga útil do pacote, equivalente ao dado real sendo transmitido.

4.1.3 GERAÇÃO E ANÁLISE DE GRAFOS

A partir desse ponto, já se tem a informação de quem se comunica com quem, que é representada pelos endereços IPs das máquinas de origem e destino.

Pode-se definir um multigrafo direcional $G_1 = (V,A)$ para representar a comunicação entre elementos de uma rede, onde cada vértice *v* pertencente a V representa um endereço IP ou nome de domínio e cada aresta *a* $=(v_1,v_2)$ pertencente a A com $v_1 \in V$ e $v_2 \in V$, representa a informação de que houve a troca de pacotes entre esses vértices. A direção é dada do primeiro vértice para o segundo, representando a origem e o destino do pacote. A FIG.4.4 exemplifica a interação entre dois endereços IPs onde um pacote é enviado de 202.116.207.75 para 209.119.182.64.

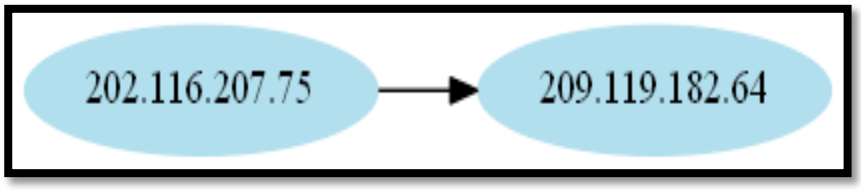

FIG.4.1. Representação gráfica de uma comunicação entre duas máquinas.

Embora essa representação deixe bem clara o relacionamento entre duas máquinas de uma rede, pode-se ainda estender a definição do grafo acima de forma a representar de uma maneira conveniente em função de propriedades de *botnets*. Assim, propõe-se o uso de gráficos de dispersão de tráfego (*Traffic Dispersion Graph* - TDG) como uma forma de representar a interação entre os membros de uma rede. Para grafos estáticos, como é o utilizado nesse projeto, existe uma característica temporal associada, pois o grafo representa um determinado estado de comunicação em um intervalo de tempo especifico, conforme ILIOFOTOU (2007). O comportamento entre *bots* de uma *botnet* revela uma característica que permite redefinir o conceito de interação entre dois nós num grafo TDG. Sabe-se que a princípio quando uma

máquina é infectada por um *malware*, este deve procurar se conectar com um *botmaster* ou um centro de C&C de forma que a intenção de quem inicia a comunicação parte da máquina *bot*. Essa característica permite estender o conceito de interação entre dois nós do multigrafo G_1. Assim propõe-se um novo grafo direcional ponderado $G_2 = (V,A_2)$ onde V é o mesmo conjunto de nós de G_1 e A_2 é o conjunto novo de arestas definidos da seguinte forma:

- A primeira intenção de contato entre dois nós v_1 e v_2 define a direção da aresta como sendo o sentido de origem e destino entre os nós v_1 e v_2.
- Caso haja mais pacotes torçados entre v_1 e v_2 e arestas do tipo (v_1,v_2) e (v_2,v_1) apareçam, é apenas incrementado o peso da aresta de forma a guardar a informação da quantidade de pacotes trocados entre esses nós. Assim, define-se $w(a) \geq 0$ para todo $a \in A$, $w: A \rightarrow R$ como sendo a quantidade de pacotes trocas entre os vértices pertencentes a *a.*

Esse novo grafo G_2 embora contenha o mesmo conjunto de vértices de G_1 possui um quantidade muito menor de arestas o que reduz o esforço computacional para grafos onde o número de vértices é grande e viabiliza a visualização gráfica do mesmo. Tomando como base a FIG.4.4 e supondo que os IPs ali representados tenham trocado numa comunicação um total de seis pacotes, chega-se a seguinte representação dada pela FIG.4.5 onde o peso w(*a*) da aresta é apresentado como o número associado a mesma.

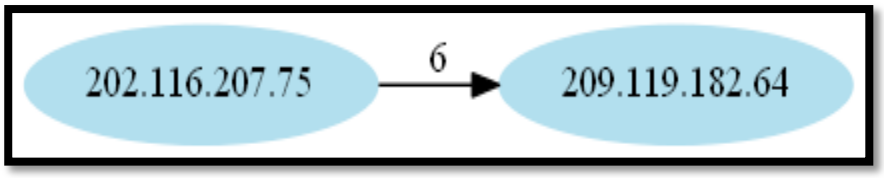

FIG.4.2. Representação gráfica considerando o número de pacotes trocados como o peso da aresta.

Para se detectar uma *botnet* é importante observar características que a diferenciem de um comportamento normal em uma rede. Os TDGs do tipo G_2 podem ser uma ferramenta útil para

se observar determinadas propriedades e identificar a presença de um tráfego *bot*. O grau de um nó representa o número de arestas ligadas a ele. Num grafo de dispersão de trafego pode-se relacionar o grau de um nó com o sentido das arestas. Um determinado nó que tenha um grau elevado aonde as arestas chegam pode representar, por exemplo, um servidor Web. O contrário pode ser exemplificado por um usuário comum realizando acesso a vários sites distintos. A FIG.4.6 abaixo exemplifica a situação de como um esquema cliente-servidor pode ser identificado. Nesse trabalho esse tipo de grafo G_2 será chamado de grafo de contato, pois se representa somente a intenção de contato sem explorar nenhuma propriedade a princípio.

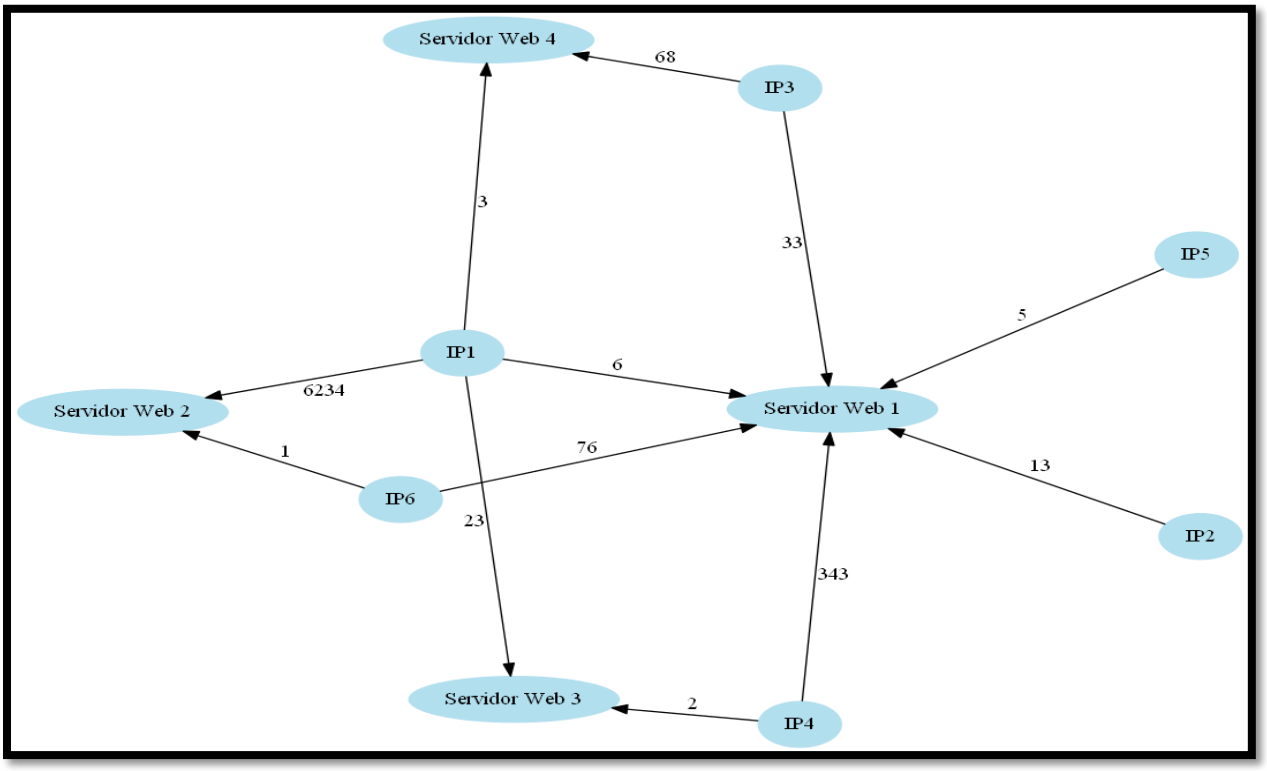

FIG.4.3. Exemplo de uma comunicação cliente-servidor.

Para *botnets* com uma topologia centralizada é possível inferir algumas propriedades a partir da análise do TDG. Sabe-se que os nós que representam centros de comando e controle normalmente têm como característica ser um nó de chegada e se existe mais de um centro de

C&C é provável que os *bot* tenham algum tipo de comunicação com eles estabelecendo certos padrões associados, pois possuem em comum contato com os mesmos nós. A FIG.4.7 mostra uma possível configuração de uma *botnet* centralizada onde há dois centros de C&C.

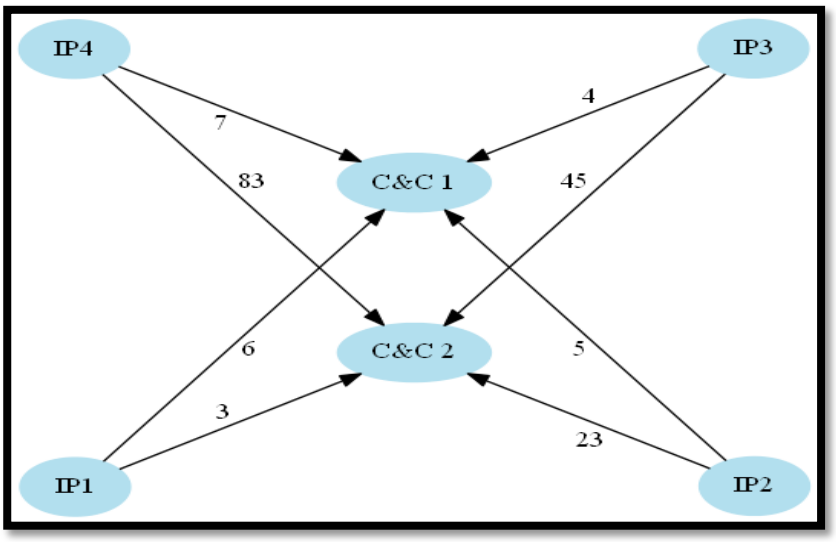

FIG.4.4. Topologia centralizada com dois centros de C&C.

Com base nessa propriedade é possível estabelecer um tipo de relacionamento característico entre esses nós. Pode-se assumir que nós origem que tenham em comum um mesmo nó destino tem a propriedade de se relacionarem. A representação gráfica disso é um grafo não direcionado ponderado completo $G_3 = (V,A_3)$ onde V é o mesmo conjunto de vértices de G_2 e A_3 é um novo conjunto de arestas onde cada vértice v_1 e v_2 pertencentes a V possuem uma aresta (v_1,v_2) se e somente se eles possuem um aresta (v_1,v_3) e (v_2,v_3) pertencentes a A_2. Isto quer dizer que v_1 e v_2 iniciaram comunicação com o mesmo vértice em comum de destino v_3. Caso v_1 e v_1 possuam mais de um vértice me comum de destino, essa característica é representada por um peso na aresta (v_1,v_2) pertencente a A_3. Pode-se ainda reduzir o tamanho do conjunto de vertes V de G_3 como sendo um novo conjunto V_3 subconjunto de V onde os vértices v pertence a V_3 se e somente se existe uma aresta *a* pertencente a A_3 tal que v pertence a *a*. Isso evita a representação de nós isolados, que não se caracterizam por apresentarem esse tipo de

relacionamento. Esse tipo de grafo $G_3 = (V_3, A_3)$ é chamado de grafo de relacionamento. Pela FIG.4.8 pode ser observado o relacionamento entre os nós montado a partir de exemplo da FIG.4.7.

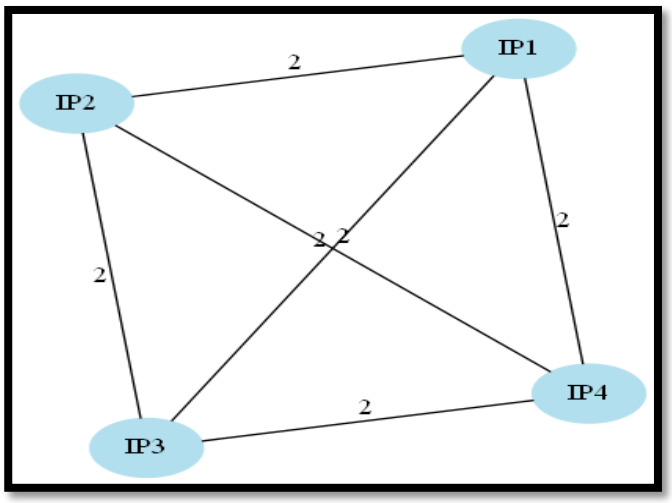

FIG.4.5. Relacionamento entre nós de uma *botnet*.

É possível afirmar que para uma topologia centralizada de uma rede de *bots* essa configuração pode ser sempre alcançada para os membros da rede, sendo possível apenas com uma inspeção visual identificar uma possível rede de *bots*. Além disso, pelo peso das arestas pode-se ainda inferir a quantidade de centros de C&C presentes na rede.

Entretanto não se pode afirmar que todos os nós que apresentam esse relacionamento são de fato pertencentes a uma *botnet*. Um exemplo que ilustraria isso é um servidor do Google, em que existiriam muitos nós apontando para o mesmo centro caracterizando um relacionamento entre eles, o que seria um erro. Assim, deve-se fazer uma busca para se obter algum tipo de informação sobre o nó central antes de poder fazer qualquer tipo de afirmação.

4.1.4 GERAÇÃO DE DADOS ESTATÍSTICOS

Para gerar estatísticas sobre o tráfego analisado com base em características de uma rede *bot*, será adotada algumas premissas para se levantar informações sobre os dados gerados e separar o que possivelmente pode indicar uma especificidade de uma *bonet*. Essas premissas são:

- Comportamento Robótico.
- Tráfego *Stealth*.
- Interesse em comum.

O comportamento robótico advém do fato de que um *malware* é um programa de computador e que é programado para agir da mesma forma frete a mesmas condições. Assim, como um *malware* é replicado para agir em muitas máquinas, espera-se que eles apresentem o mesmo comportamento de forma que o tráfego gerado na comunicação seja semelhante aos demais.

Botnets procuram apresentar tráfego *stealth* que significa que o comportamento na comunicação entre membros de uma rede *bot* tende a ser discreto. A FIG.4.9 a seguir exemplifica um possível envio de comandos de um C&C para seus *bots* através de um canal IRC, de forma se pode perceber que é anormal, uniforme e também baixa a quantidade de pacotes trocados entre esses nós.

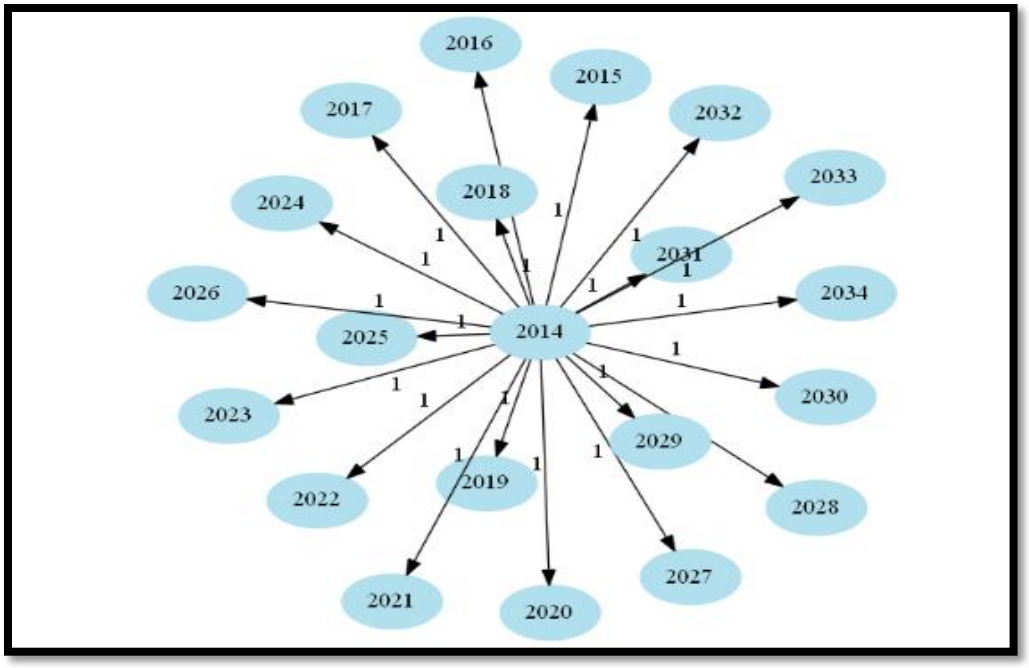

FIG.4.6. Representa um possível envio de comando de um C&C utilizando um servidor IRC. A uniformidade e a baixa quantidade de pacotes trocados tornam esse tipo de característica anormal dentre o que se pressupõem para uma comunicação IRC. Essa imagem é resultante da analise de um *trace* no protocolo IRC do dia 03/04/2011disponibilizado pela MAWI (2012).

O interesse comum é uma característica que faz com que se analise nós que contenham um grau de chegada maior que dois, pois admite-se que para formarem um rede seria necessário pelo menos três elementos, isto se traduz na presença de pelo menos dois *bots* e um C&C e faz com que se descarte a comunicação um-para-um numa rede, uma vez que é praticamente impossível inferir algo sobre ela.

A partir dessas características podem ser levantadas estatísticas analisando o grafo de contato. Cada nó que possua grau de chegada maior que dois, deve ser analisado de forma a se obter a média de pacotes trocados e o desvio padrão. Essas medidas podem estabelecer uma forma de se diferenciar o comportamento robótico de um *malware* de um usuário comum. A princípio, espera-se que por apresentar o mesmo comportamento os *bots* venham a apresentar certa uniformidade o poderia se identificado como um desvio padrão baixo ou nulo. A

informação sobre a média total de pacotes trocados por nós com grau de chegada superior a dois, pode ser um indicador para se estabelecer o que poder ser enquadrado com tráfego *stealth*.

Assim, pode-se analisar o grafo de contato de um ou mais *traces* para se obter esses dados estatísticos e poder definir um conjunto de elementos pertencentes ao grafo que possa ser indicado como suspeito.

5 TRABALHOS REALIZADOS

5.1 AQUISIÇÃO E ANÁLISE DE TRACES

Primeiramente, foi desenvolvido um programa com base nas características descritas na seção 4.1 que pudesse receber como fonte de dados arquivos contendo pacotes da internet e analisa-los segundo seu protocolo para gerar os grafos de contato, relacionamento e as estatísticas de média, desvio padrão e grau de chegada para cada nó dos grafos de contato.

Desenvolvido na linguagem Java, esse programa utiliza a *API* JPCAP(2007) que encapsula as funcionalidades da WINPCAP (2012) que é uma ferramenta baseada na biblioteca PCAP (2003) que, entre outras funções, permite interceptar e mostrar o tráfego de pacotes por uma determinada interface de rede, podendo também salvar em um arquivo para uma futura análise. Além disso, é gerado um arquivo separado na linguagem *dot* de forma a possibilitar a visualização dos grafos pelo GRAPHVIZ (2012). A FIG.5.1 mostra os passos aplicados na análise de um trace pelo programa.

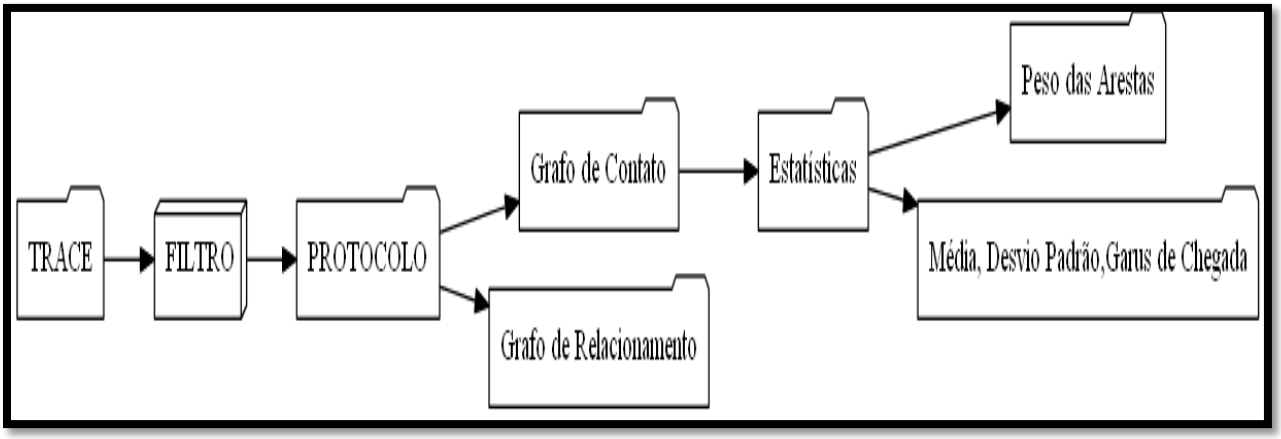

FIG.5.1. Esquema do programa.

Como fonte inicial de dados para análise, foram adquiridos sete *traces* no formato tcpdump, disponibilizados pela empresa MAWI (2012). Esses *traces* são referentes aos sete primeiros dias de Abril de 2011 onde foram capturados pacotes num intervalo de quinze minutos. Estes *traces* são oriundos de um link transoceânico de 150mb/s.

As FIG.5.2 e FIG.5.3 são um exemplo dos grafos de contato para os protocolos HTTP e IRC visualizados no GRAPHVIZ (2012) resultantes da análise do trace do dia 03. Para diminuir o tamanho dos traces, eles são disponibilizados sem o *payload*, fato que inviabiliza a análise para o protocolo DNS.

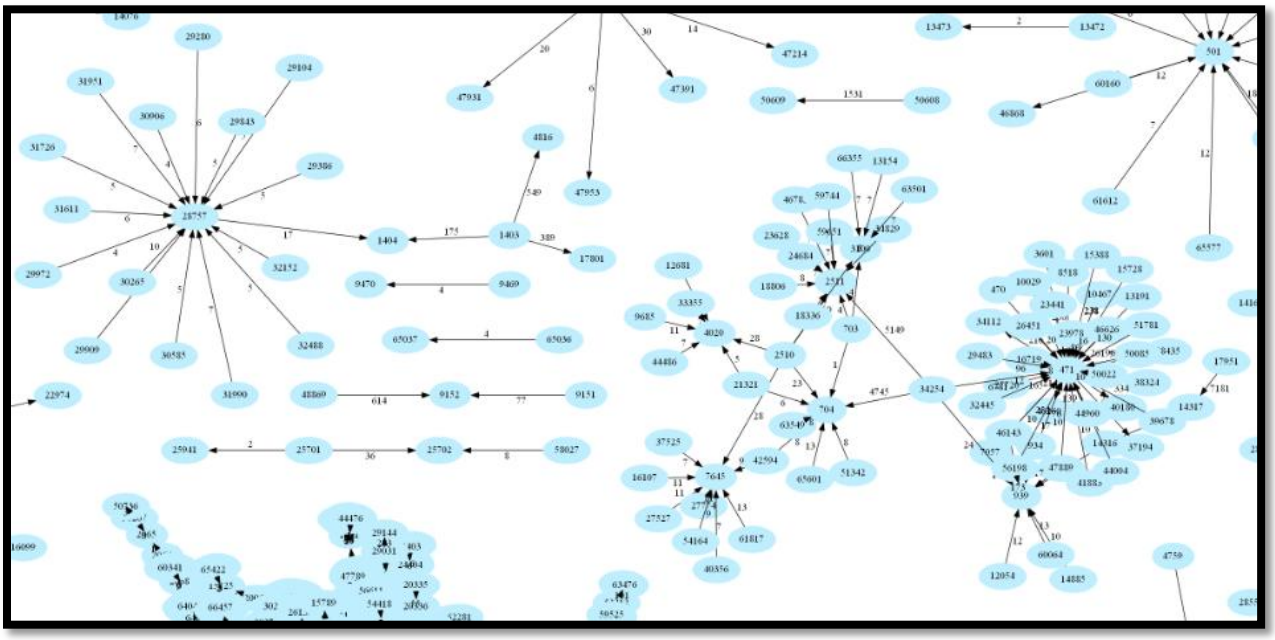

FIG.5.2. Representa parte do grafo de contato do tráfego HTTP capturado para o dia 04/04/11.

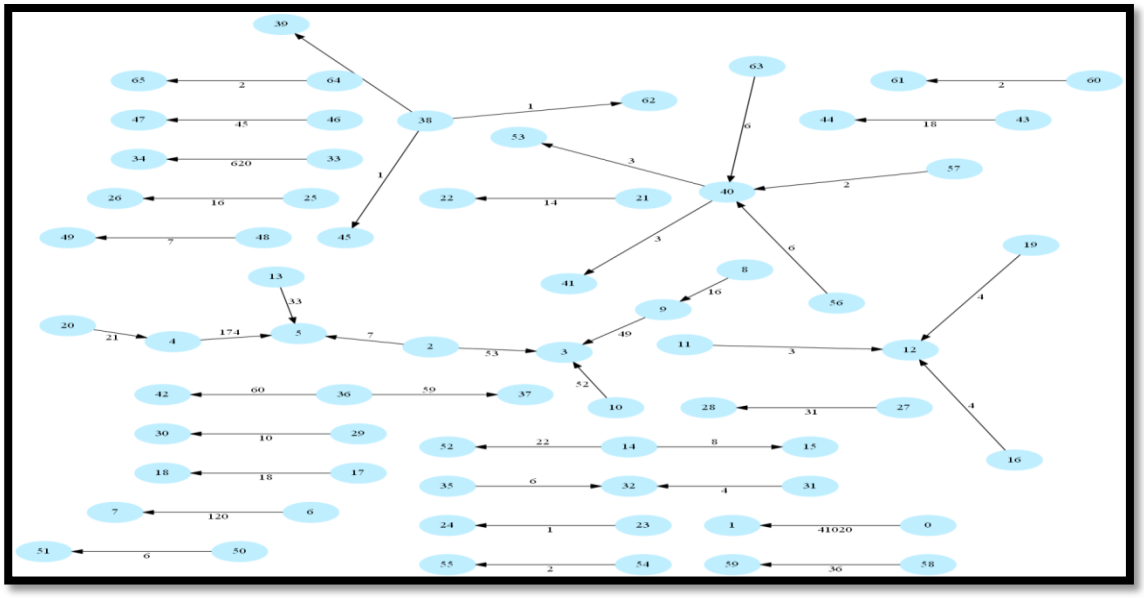

FIG.5.3. Representa todo tráfego IRC do dia 04/04/11.

O grafo que representa o relacionamento foi feito apenas para o protocolo IRC. Isso se deve ao fato da complexidade embutida para gerar um grafo desses (grafo completo) com muitos nós, pois o número de arestas geradas será da ordem de n^2 onde n é o número de nós no grafo, necessitando de muito tempo e recursos computacionais para tal. A FIG.5.4 mostra o grafo de relacionamento resultante para o tráfego IRC do dia 04/04/11. Como se pode observar na FIG.5.4 a imagem, mesmo com poucos nós, já apresenta dificuldade em identificá-los.

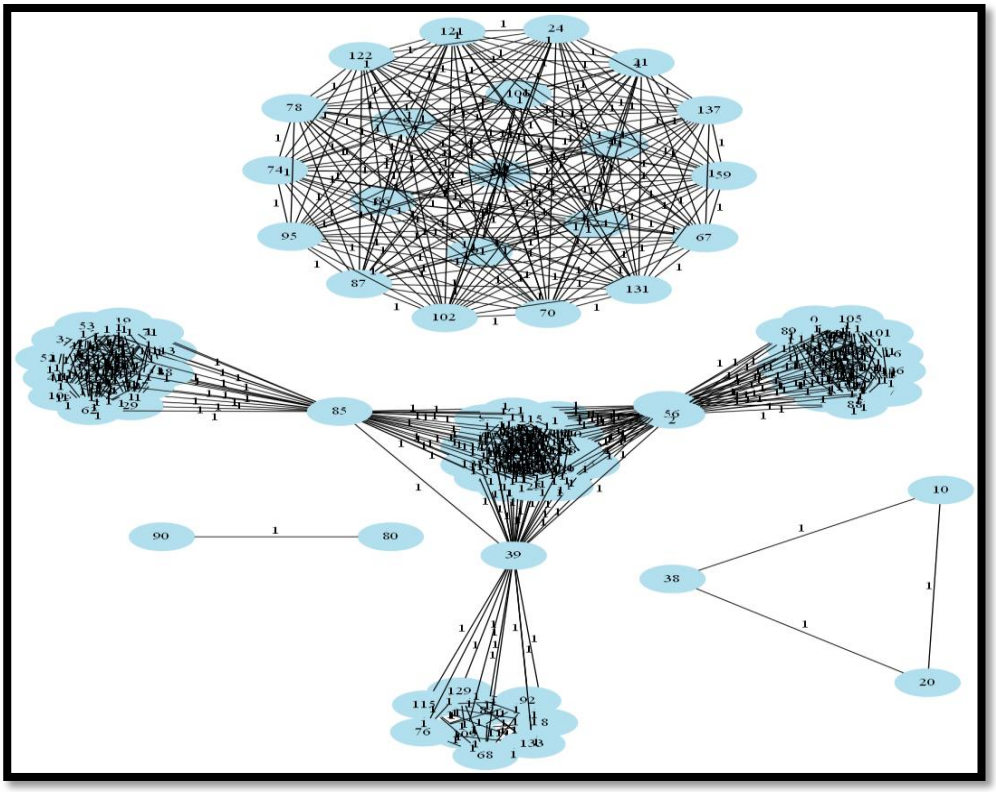

FIG.5.4. Grafo de relacionamento para o IRC do dia 04/04/11.

5.2 ESTATÍSTICAS SOBRE OS GRAFOS

Nesta seção serão apresentadas as estatísticas geradas a partir dos grafos de contato dos sete *traces* analisados para os protocolos HTTP e IRC.

O objetivo é gerar a distribuição da média de pacotes trocados e seu desvio padrão e dos graus de chegada de cada nó do grafo de contato entre nós que apresentam grau de chegada maior que dois e seus respectivos nós de origem, com o propósito de se estabelecer o que se pode entender como tráfego *stealth*, baixa média de pacotes trocados e baixo desvio padrão. Assim foram calculadas as médias dessas medidas obtidas separadamente de cada *trace* após a execução dos passos descritos na seção 5.1, de forma a se obter uma média geral delas levando em conta todos os sete *traces* analisados. A TAB.5.1 e a TAB.5.2 mostram as médias dessas

medidas para o grafo de contato dos protocolos HTTP e IRC de cada *trace* em questão que foram feitas de forma manual juntando os resultados.

TAB.5.1 Estatísticas para o protocolo HTTP.

MAWI \HTTP			
***Trace*\Média**	Média pkt	Desvio Padrão	Grau Chegada
01/04/2011	238,5912	272,665	9,5790
02/04/2011	198,2533	245,5629	9,3078
03/04/2011	128,8145	149,3078	5,9708
04/04/2011	156,1613	195,3629	6,9020
05/04/2011	152,3831	127,4232	7,9343
06/04/2011	172,6023	204,2353	6,9258
07/04/2011	163,1647	185,3823	7,1172
Média geral	172,8529	197,1342	7,6767

TAB.5.2 Estatísticas para o protocolo IRC.

MAWI\IRC			
***Trace*\Média**	Média pkt	Desvio Padrão	Grau Chegada
01/04/2011	33,1333	13,1861	4,5714
02/04/2011	209,5278	74,6873	2,5833
03/04/2011	24,3809	15,0614	2,7142
04/04/2011	27,2	15,6849	2,8
05/04/2011	24,575	20,5775	3,625
06/04/2011	32,6839	24,3102	6,8
07/04/2011	34,5866	6,8747	3
Média	55,1553	24,3403	3,7277

A partir desse ponto, pretende-se estipular com base nas médias de desvio padrão, da média de pacotes trocados e do grau de chegada, valores de corte que indicariam quem pode ser considerado suspeito. O gráfico de Desvio Padrão vs Média a seguir mostra a área em que se espera encontrar a maioria dos elementos suspeitos de uma *botnet*, pois nada mais é do que

aqueles que apresentam uma quantidade de pacotes média trocados inferior a um determinado valor com um desvio padrão associado.

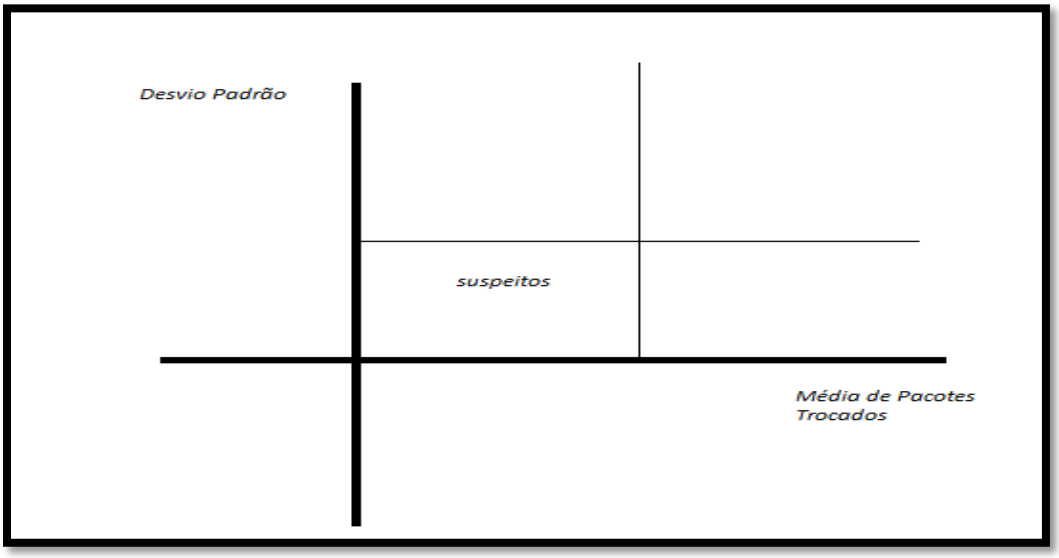

FIG.5.5. Gráfico utilizado para determinar nós suspeitos.

O quadrante em que se enconta a palavra “suspeitos” é a região em que os possíveis nós suspeitos devem ser encontrados. Entretanto, resta ainda determinar que valor é o ideal de média e desvio padrão a se adotar para fazer uma triagem dos nós de maneira que não se generalize demais gerando um grande volume de dados a ser analisado, o que não seria interessante quando a quantidade de tráfego é grande, e também de maneira que não se limite tanto ao ponto de se perder a maioria da informação contida no grafo.

5.3 RESULTADO PARA UM TRACE

Foi escolhido o *trace* do dia seis de abril de 2011 para se aplicar a metodologia completa de forma a gerar uma lista com os nós suspeitos.

Para isso foi estipulado como um valor de 5% da média e 1.5% do desvio padrão dos valores médios da TAB5.1 resultando em valores de 8.64 pacotes em média e um desvio de 2.95.

O gráfico de Desvio Padrão vs Média para esse *trace* é o dado na FIG5.6, obedecendo uma escala logarítmica.

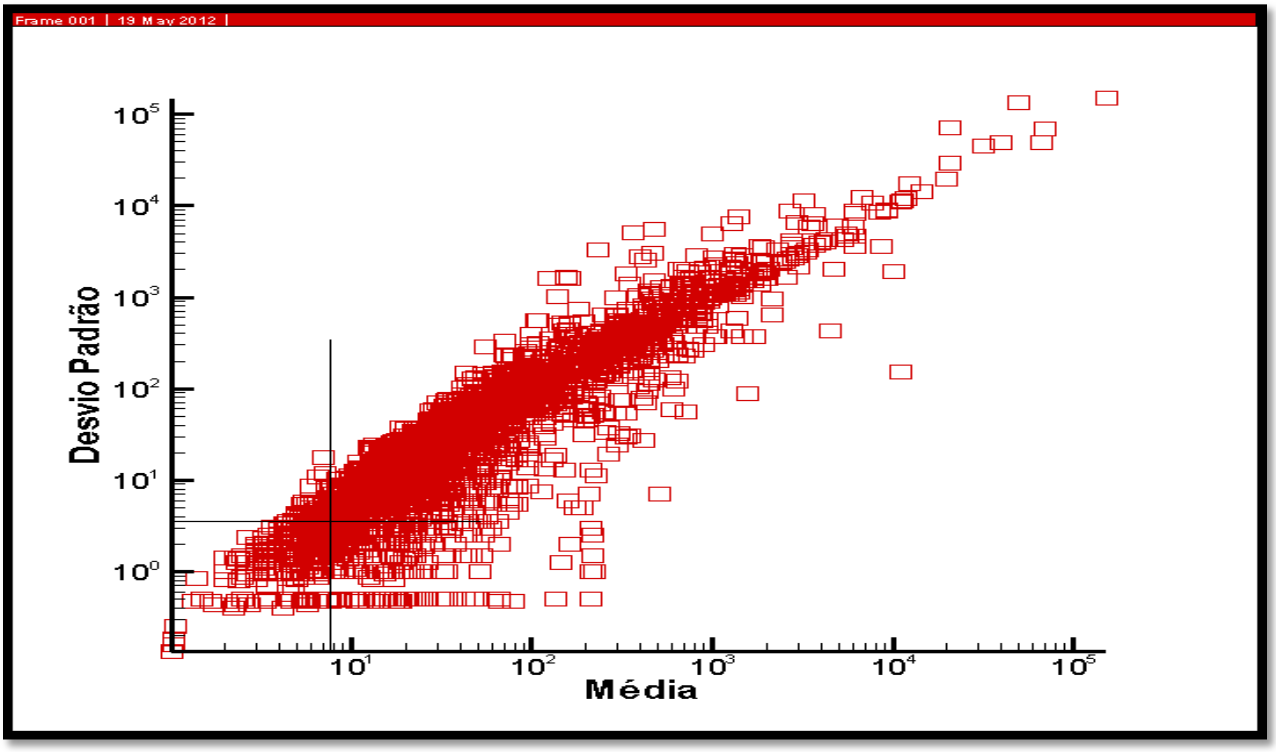

FIG.5.6. Gráfico Desvio Padrão vs Média.

O resultado da análise para o grafo de contato deste *trace* foi uma lista com 2987 nós suspeitos. Com a visualização sempre é um fator que ajuda em qualquer análise de gráficos, o grafo desses nós suspeitos com as suas respectivas origens foi gerado e pode ser visto na FIG.5.7. Um número que a princípio parece ser alto, mas que com a ajuda da figura se pode entender melhor o resultado obtido de forma diminuir a quantidade de nós a se verificar.

Além do que era previsto para um modelo centralizado de se mapear os possíveis centros de comando e controle, foi também obtido o mapeamento dos possíveis *bots*. Essa fato se deve a três motivos:

- A caracteristica temporal do *trace* analisado.
- A possibilidade de existir mais de um centro de C&C por *botnet*.
- A possibilidade de uma determinada máquina pertencer a mais de uma *botnet*.

Assim o grande volume de arestas saido de poucos nós, pode ser entendido como a comunicação de dois ou mais C&C com um mesmo *bot* em comum ou então uma mesma máquina que possa conter *malwares* de diferentes *botnets*, o que pode indicar um novo tipo de comportamento que pode ser modelado em futuras análises, uma vez que um computador que não apresente nenhuma ou insuficiente proteção acaba ficando vulnerável aumentando as chances de ser infectato por mais de um *malware*. Isso também é resultato da amostra do *trace* que tem apenas um janela de duração de quinze minutos o que dificulta a representação da comunicação como um todo.

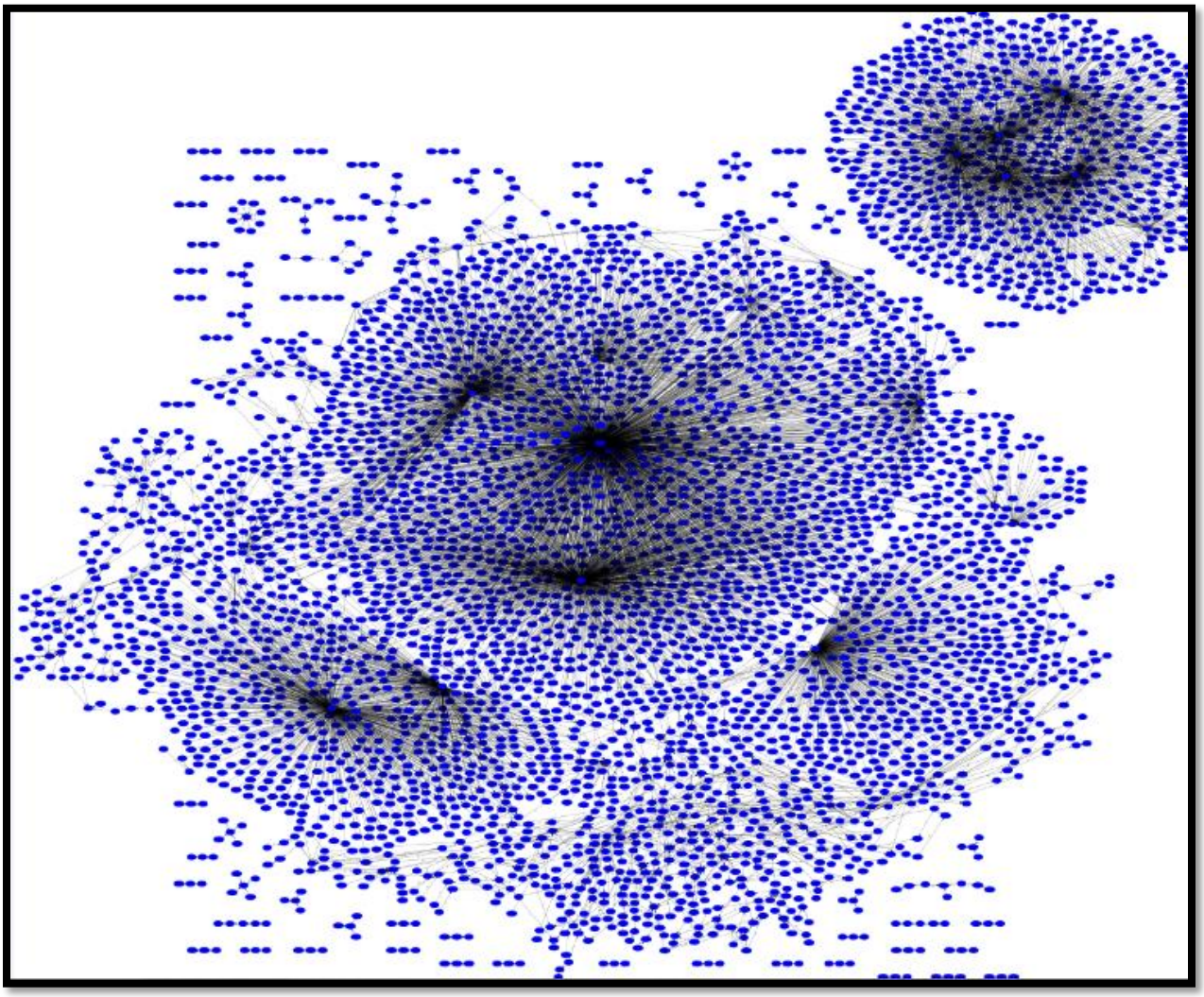

FIG.5.7. Grafo de nós suspeitos com suas origens de contato.

Alguns IPs resultantes da lista de suspeitos, com base na sua característica no grafo, como grandes centros de onde partem ou chegam arestas, foram analisado por *blacklists* da MULTIRBL (2012) e ROBTEX (2012), onde foi possível verificar que boa parte desses nós e também de alguns de seus membros, os quais também foram verificados, apresentam algum tipo de comportamento suspeito já denunciado como, por exemplo, spam e DNS dinâmico. Os mais curiosos foram os nós pertencentes a grande aglomeração superior direita da FIG.5.7, onde de fato são encontrados domínios registrados para esses IPs e que também estão disponíveis para acesso, mas que o tráfego oriundo deles e os nós em comum para os mesmos sites pode ser indício de centros de C&C sendo somente um disfarce para encobrir atividades maliciosas, podendo ser alvo para investigações mais apuradas voltadas para esses domínios.

5.4 APLICATIVO PARA GERAR A LISTA DE SUSPEITOS

Foi desenvolvido um aplicativo em Java capaz de abrir arquivos que contenham pacotes da internet nos formatos pcap, dump, tcpdum e cap e utilizando o programa descrito nos passos da FIG.5.1 para fazer a análise do arquivo. Esse programa roda apenas em Windows 32 bits e necessita da biblioteca WINPCAP (2012) instalada no sistema operacional. Entretanto, mesmo sendo o objetivo gerar uma lista de suspeitos, o ele gera todas as etapas intermediarias automaticamente descritas na FIG5.1 para que se possa repetir o processo para futuras análises. Dessa forma ao se escolher um arquivo num dos formatos acima, as saídas do programa são, além da lista de suspeitos, os seguintes arquivos texto:

- Multigrafo não ponderado resultante do filtro, onde cada linha representa um contato entre dois IPs ou nomes de domínio, em que o primeiro é a origem e o segundo o destino. Este arquivo é nomeado com o nome original do arquivo seguido de “_<protocolo_utilizado>”.

- Um grafo de contato onde cada linha representa uma interação entre dois IPs seguidos do número de pacatos trocados entre eles. Este arquivo é nomeado a partir do arquivo anterior (nome do multigrafo) seguido de "_Grafo_Contato".
- Três arquivos diferentes uma para cada grau de chegada, saída e total, onde as linhas representam o grau e a quantidade de vezes que este aparece. Este arquivo é nomeado a partir do nome do multigrafo seguido de "_Grau_<tipo>".

- Dois arquivos que fazem o mapeamento entre nó de origem e seus respectivos nós de destino e vice-versa. O formato é dado pelo IP de origem ou destino, dependendo do arquivo, separado por tabulação e "::" seguidos do IP-peso. Este arquivo é nomeado a partir do nome do multigrafo seguido de "_No_<origem/destino>_Peso".

- Um arquivo que contem as informações estatísticas de um IP de destino sobre a média, desvio padrão e grau de chegada dos IPs de origem. O formato segue essa ordem respectivamente para cada valor. Este arquivo é nomeado a partir do nome do multigrafo seguido de "_Estatisticas_Origem".

- Por último é gerada uma lista de IPs suspeitos, contendo apenas os IPs. Este arquivo é nomeado a partir do nome do multigrafo seguido de "_NosSuspeitos".

Com uma interface simples, ele permite selecionar um ou mais arquivos para serem analisados e também removê-los caso seja necessário, através dos botões de adicionar e remover, respectivamente. Também é possível selecionar para que protocolo se deseje fazer a análise de forma que só foram implementados apenas para HTTP, IRC e DNS e portando apenas esses estão disponíveis.

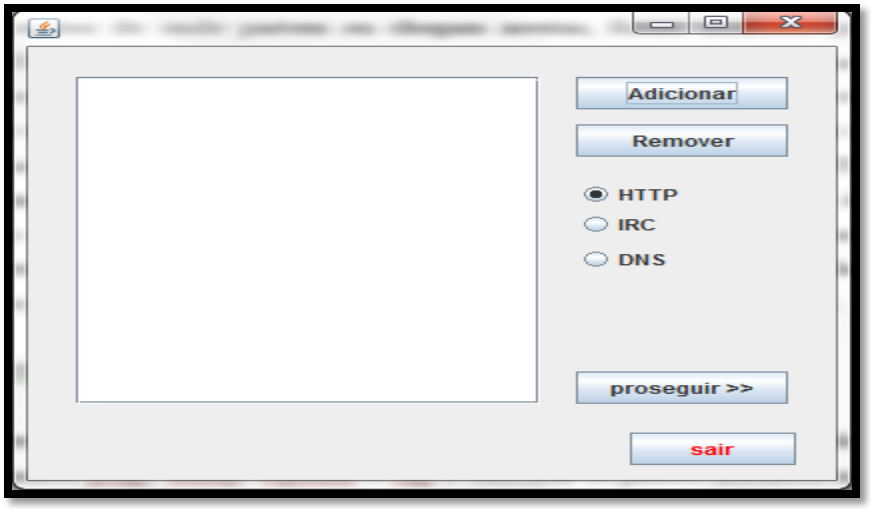

FIG.5.8. Interface inicial do aplicativo.

Os demais botões prosseguir e sair permitem que se avance para a próxima etapa e que se feche o aplicativo, respectivamente. Caso o usuário queira prosseguir, será então apresentada uma nova interface conforme descrito na FIG.5.9.

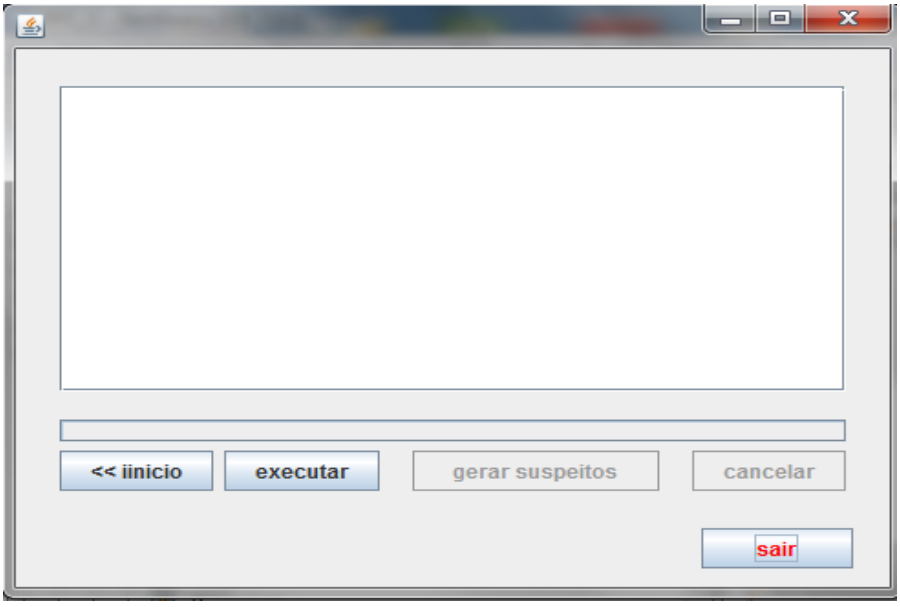

FIG.5.9. Interface de execução do aplicativo.

Nessa nova etapa, pode-se voltar para a fase anterior pelo botão início, sair, que fecha o aplicativo, ou executar a análise pelo botão executar.

Caso seja selecionado executar, a execução propriamente dita da análise começa. A barra horizontal vai sendo preenchida indicando o progresso da tarefa e no painel branco vai sendo mostrado a subtarefa que esta sendo executada e apenas os botões de sair e cancelar, que cancela a tarefa atual, ficam disponíveis. Ao final da execução, pode-se voltar ao início, sair do aplicativo ou avançar selecionando "gerar suspeitos" onde se pode fornecer os valores de média, desvio padrão e grau de chegada para se gerar uma lista de suspeitos conforme a FIG.5.10.

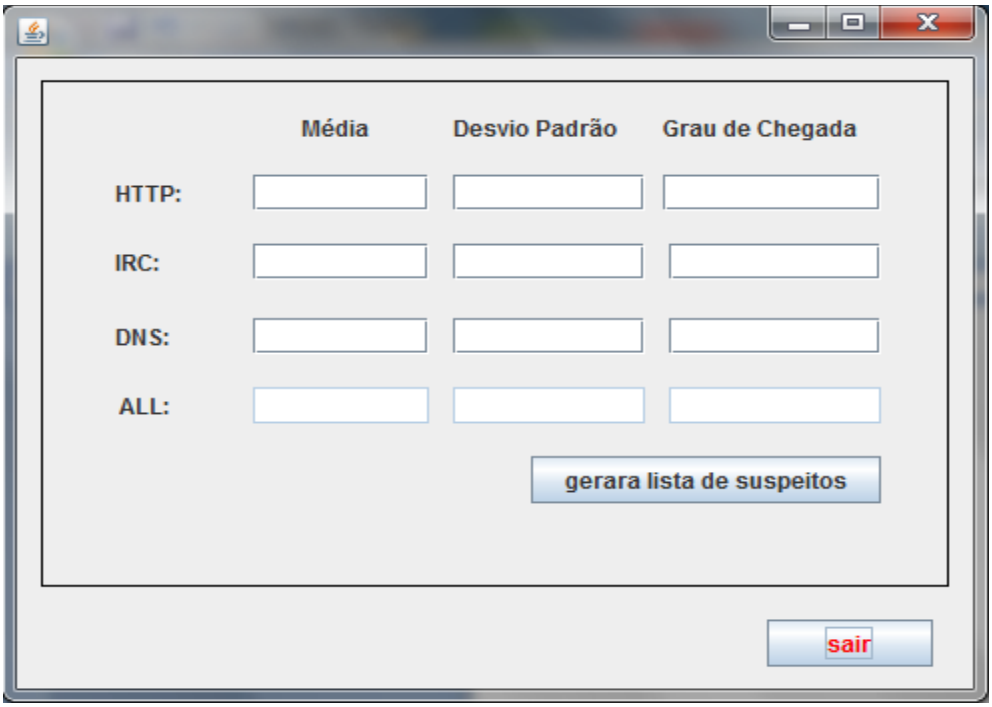

FIG.5.10. Interface para gerar suspeitos

Assim, pode-se ao selecionar "gerar lista de suspeitos" listar um conjunto de IPs de acordo com os valores estipulados. Para o mesmo *trace* analisado na seção 5.3 com valores de 10% da média e do desvio padrão da TAB5.1 e grau de chegada 9 foram obtidos 4014 IPs suspeitos. Nesse ponto, foram analisados estatisticamente de maneira a representar o

comportamento do todo em que foram obtidos 65% de nós suspeitos por se apresentarem *blacklisted* em MULTIRBL (2012) e ROBTEX (2012), com uma margem de erro de 10%.

6 CONCLUSÃO E TRABALHOS FUTUROS

A partir do estudo sobre *botnet* foi possível conhecer a importância do assunto junto ao contexto de defesa nacional, bem como a importância de se combater esse tipo de *malware* para o bom funcionamento de sistemas, sites e também da economia.

Com análise de cada tipo de arquitetura de *botnest*, mesmo as que apresentavam um modelo teórico, foi de suma importância para o desenvolvimento desse projeto, pois foi a partir delas que se levantou as primeiras e principais características para se identificar o comportamento desse tipo de *malware* em uma rede.

Com o estudo de grafos para representar tráfego de rede, os TDGs, foi possível modelar um tipo de grafo específico que não só facilitou a representação do tráfego do *trace*, como também viabilizou sua visualização, possibilitando num primeiro momento até mesmo com uma inspeção visual identificar possíveis suspeitos.

Entretanto, a grande quantidade de informações oriundas de *traces* com muitos pacotes exigiu a necessidade de se abandonar a análise pontual e trabalhar com dados estatísticos como um todo. Nesse ponto, foi necessário definir medidas de média de pacotes trocados e seu desvio padrão, bem como o grau de chegada dos nós, de forma a se levantar dados sobre o comportamento dos elementos da rede.

Com base no gráfico Desvio Padrão vs Média, foi evidenciado o que o seria interessante considerar respeitando as premissas de comportamento robótico, tráfego *stealth* e interesse em comum, como suspeito ao se realizar a análise.

Os resultados obtidos foram satisfatórios demonstrando que a técnica empregada é valida e atente ao objetivo de detectar *botnets*. Entretanto, como trabalhos futuros, ainda se pode aprimorá-la de forma estender a modelagem do relacionamento entre membros de uma rede de *bots* observando também a comunicação entre C&C e seus *bots* explorando a característica temporal dos *traces* analisados.

Assim, esse projeto pode contribuir para uma nova maneira de detectar *botnets* levando em consideração suas características comportamentais junto com uma análise do registro de um tráfego de rede.

7. BIBLIOGRAFIA

ABREU, N., DEL-VECCHIO, R., VINAGRE, C., STEVANOVIC, D. Introdução à teoria espectral de grafos com aplicações. SBMAC. 2007.

COOKE, Evan, JAHANIAN, Farnam, MCPHERSON Danny. The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets. University of Michigan, Electrical Engineering and Computer Science Department Arbor Networks. Novembro de 2006. 6 p.

FORTUNATO, S. Community detection in graphs, arXiv:0906.0612v2 [physics.soc-ph] 25 Jan 2010.

GRAPHVIZ, Graph Visualization Software. 2012. Disponível em: http://graphviz.org/ [Acessado em: 4 Mar 2012].

IANA, Service Name And Transport Protocol Port Number Registry. 2012. Disponível em:http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml. [Acessado em: 4 Mar 2012].

IANELLI, Nicholas, HACKWORTH, Aaron. Botnets as a Vehicle for Online Crime. CERT Coordination Center. Dezembro de 2005. 30 p.

ILIOFOTOU, Marios. PAPPU, Prashanth. FALOUTSOS, Michalis. MITZENMACHER, Michael. Singh, Sumeet. Varghese, George. **Network Monitoring using Traffic Dispersion Graphs (TDGs)**, Internet Measurement Conference 2007, San Diego, California, USA. Outubro 2007.

JPCAP. A Java Library for Capturing and Sending Network Packets. 2007. Disponível em: http://netresearch.ics.uci.edu/kfujii/Jpcap/doc/ [Acessado em: 4 Mar 2012].

MAWI Working Group Traffic Archive. 20120. Disponível em: http://mawi.wide.ad.jp/mawi/ [Acessado em: 4 Mar 2012].

MOCKAPETRIS, P. Request for Comments: 1035: Domain Names - Implementation and Specification. Nov. 1987. Disponível em: http://tools.ietf.org/html/rfc1035[Acessado em: 4 Mar 2012].

MULTIRBL. Blacklist, Whitelist and FCrDNS check tool. 2012. Disponível em: http://multirbl.valli.org/lookup/ [Acessado em: 27 Fev 2012].

OIKARINEN, J., REED, D. Internet Relay Chat Protocol. 1993. Disponível: http://www.irchelp.org/irchelp/rfc/rfc.html [capturado em 15 ago. 2011].

PCAP, Packet Capture library. Novembro 2003. Disponível em: http://www.tcpdump.org/ pcap3_man.html. [Acessado em: 4 Mar 2012].

PLOHMANN, Daniel, GERHARDS-PADILLA, Elmar, LEDER, Felix. Botnets: Measurement, Detection, Disinfection and Defence. ENISA. Mar, 2011. 153p.

PRASOLOV, V. V. The Problems and Theorems in Linear Algebra, American Mathematical Society, 1994.

ROBTEX. Swiss Army Knife Internet Tool. 2012. Disponível em: http://www.robtex.com/ [Acessado em: 27 Fev 2012].

ROMERA, Ranieri. Discerning Relationships: The Mexican Botnet Connection. A Trend Micro Research Paper. Setembro 2010. 34p.

SAHA, Basudev, GAIROLA, Ashish. Botnet: An Overview. Indian Computer Emergency Response Team. India, junho de 2005. 12 p.

SNAP, Stanford Network Analysis Platform. 2010. Disponível em: http://snap.stanford.edu/ [Acessado em: 4 Mar 2012].

SYMANTEC, Fraud Activity Trends. 2010. Disponível em: http://www.symantec.com/threatreport/topic.jsp?id=fraud_activity_trends&aid=originating_sources_of_spam. [Acessado em: 13 Maio 2012].

TREND MICRO. Taxonomy of Botnet Threats. A Trend Micro White Paper. Novembro de 2006. 15 p.

WHATISMYIPADDRESS. Blacklist check. Disponível em : http://whatismyipaddress.com/blacklist-check [Acessado em: 27 Fev 2012].

WINPCAP, The Industry-Standard Windows Packet Capture Library. 2012. Disponível em: http://www.winpcap.org/default.htm[Acessado em: 4 Mar 2012].