# Uma Arquitetura Baseada em Assinaturas para Mitigação de *Botnets*

**João Marcelo Ceron**[1,2], **Lisandro Zambenedetti Granville**[2], **Liane Margarida Rockenbach Tarouco**[2]

[1] Time de Resposta a Incidentes de Segurança (TRI)
Universidade Federal do Rio Grande do Sul(UFRGS)
Rua Ramiro Barcelos, 2574 – Porto Alegre, RS

[2]Instituto de Informática
Universidade Federal do Rio Grande do Sul (UFRGS)
Av. Bento Gonçalves, 9500 – Porto Alegre, RS

{joao.ceron, granville, liane}@inf.ufrgs.br

***Abstract.*** *Botnets are one of the most serious security threats of the current Internet. Such threats are characterized by being very dynamic, frequently incorporating into their structure new features whose goal is to decrease the efficiency of systems like anti-viruses and IDSes. This paper presents an architecture for a signature-based tool for botnet detection and mitigation. Identifying botnets' signatures in an automated fashion helps to detect compromised machines and to mitigate the damages caused by botnets.*

***Resumo.*** *As* botnets *são consideradas uma das principais ameaças à segurança da Internet. Tais ameaças caracterizam-se por serem muito dinâmicas, frequentemente incorporando novas características às suas estruturas, de forma a diminuir a efetividade de sistemas, por exemplo, de antivírus e IDS's. Este artigo apresenta uma arquitetura de ferramenta para mitigação e detecção de* botnets *baseada em assinaturas de rede de máquinas comprometidas por* bots. *Identificar assinaturas de* botnets *de forma automatizada auxilia no processo de detecção de máquinas comprometidas e no processo de atenuação dos danos causadas pelas mesmas.*

## 1. Introdução

Atualmente, uma das mais sérias ameaças à segurança da Internet são as *botnets*. Uma *botnet* é composta por um conjunto de máquinas comprometidas (*bots*) que podem ser controladas remotamente por uma entidade denominada *botmaster* [Wang et al. 2010]. As ações desempenhadas pelas *botnets*, em geral, são danosas à estrutura da Internet, tais como ataque de negação de serviço distribuído (DDoS), envio de *spams*, distribuição de *malwares* e roubo de informações. Estima-se que atualmente o número de máquinas infectadas (*bots*) na Internet seja superior a dezenas de milhões [Kreibich et al. 2009].

Neste contexto, a comunidade científica de segurança da informação vem estudando meios de detecção e prevenção de *botnets* [Gu et al. 2008a] [Goebel and Holz 2007] [Gu et al. 2007]. No entanto, a maioria das metodologias e

técnicas estão relacionadas a tipos específicos de *botnets*, como por exemplo, *botnets* que utilizam estrutura centralizada e baseadas em certos protocolos de comunicação.

Considerando tal cenário, este trabalho apresenta uma arquitetura para, automaticamente, gerar assinaturas de *botnets* tendo como base a análise de arquivos maliciosos. Para isso, a arquitetura emprega algumas estratégias para gerar assinaturas por meio de análise dinâmica de arquivos maliciosos - coletados por uma *honeynet* - e analisados em ferramentas automatizadas *on-line* conhecidas por *sandboxes*. Além da especificação e implementação da arquitetura de mitigação e detecção de *botnets*, o presente trabalho discute detalhes da implementação e emprego da solução no contexto da Universidade Federal do Rio Grande do Sul, considerando um período de 8 meses.

O restante deste trabalho esta organizado da seguinte maneira. Na Seção 2 são revisados trabalhos de mitigação e detecção de *botnets*. A Seção 3 apresenta a arquitetura proposta, bem como a descrição dos seus componentes. Os detalhes de implementação são apresentados na Seção 4, enquanto que na Seção 5 são descritos resultados atual alcançados com a solução. Por fim, o artigo é finalizado com a apresentação dos resultados obtidos e as conclusões do trabalho.

## 2. Trabalhos relacionados

Os *malwares* classificados como *bots* possuem uma característica singular: os *bots*, após serem executados num sistema vulnerável, estabelecem um canal de comunicação com o atacante através do qual podem receber instruções remotamente. Dessa forma, o atacante (*botmaster*) utiliza uma estrutura de comunicação - composta por protocolos e serviços de rede - para enviar comandos e controlar as máquinas comprometidas. Devido a tais características, as *botnets* podem ser bastante dinâmicas e difíceis de serem combatidas com as ferramentas tradicionais de segurança (*e.g.*, antivírus e sistemas de detecção de intrusão).

Atualmente, existem diversas pesquisas com metodologias para combater as *bonets* e seus efeitos. As soluções mais significativas no contexto deste artigo são apresentadas a seguir. Na ferramenta `Botsniffer` [Gu et al. 2008b], os autores especificam uma metodologia para identificar controladores de *botnets* baseada na correlação de padrões de comunicação (similaridade de tráfego). Para isso, a ferramenta emprega um algoritmo de correlação que atua em *botnets* de estrutura centralizada e baseadas em protocolos específicos (IRC e HTTP). Nesse algoritmo, são observados *bots* pertencentes a mesma *botnet* que demonstram uma sincronização muito forte em relação às suas mensagens de respostas na rede. Através de diferentes análises, a ferramenta possibilita uma correlação espaço-temporal do tráfego e identifica nodos de *botnet* com baixa taxa de falsos positivos. O Botsniffer mostra-se relativamente eficaz. No entanto, é apenas aplicável a um conjunto restrito de *botnets* baseado no protocolo IRC e HTTP.

Holz [Holz et al. 2008], por outro lado, apresenta uma técnica para mitigar a *botnet* `Storm Worm`. A `Storm Worm` é baseada em protocolos P2P e sua principal funcionalidade é o envio de *spams*. Trata-se de uma *botnet* bastante popular e, segundo estatísticas [Holz et al. 2008], já foi considerada responsável por mais de 10% de todo *spam* gerado na Internet. No trabalho, os autores analisaram uma grande quantidade de *spams* e foram aptos a capturar exemplares de arquivos malicioso. Logo após, foi realizada uma engenharia reversa dos arquivos maliciosos visando entender os vetores de

propagação e os mecanismos de acesso à rede de controle da *botnet*. Como resultado, os autores decifraram o protocolo de comunicação e infiltraram-se na rede da própria *botnet*. Um estudo mais intensivo possibilitou identificar algumas limitações do protocolo de comunicação e controle da *botnet* como, por exemplo, a falta de autenticação para a troca de mensagens. De posse dessas informações, os autores conseguiram enviar mensagens na rede - utilizando o protocolo da própria *botnet* - solicitando o descadastramento dos integrantes (máquinas infectadas).

A solução que mais se assemelha com a solução proposta por este trabalho é apresentada por Peter Wurzinger *et al.* [Wurzinger et al. 2009]. Na solução é descrita uma técnica de detecção de *botnets* baseada em assinatura (tráfego de rede) de máquinas infectadas. Para isso, um conjunto de *malwares* é analisado num ambiente controlado - num *sandbox* comercial - e suas atividades de rede monitoradas. A arquitetura proposta em nosso trabalho, entretanto, utiliza um conjunto de serviços *on-line* gratuitos - tal como analisadores de arquivos - para compor assinaturas de *botnets*. Além disso, a estrutura é concebida de forma modular possibilitando com que novos componentes sejam implementados como, por exemplo, novas metodologias para análise de *malwares*. Adicionalmente, nossa arquitetura permite localizar máquinas comprometidas num contexto temporal, localizando assinaturas no tráfego armazenado pela solução `Netflow` [Cisco 2010]. Desta forma, máquinas comprometidas mas não ativas no momento podem ser identificadas. Finalmente, a arquitetura proposta visa construir colaborativamente com novas técnicas para detectar *botnets*, com base nas análises dos *malwares* armazenados na base de dados do sistema.

## 3. Arquitetura de detecção e mitigação de *botnets*

A arquitetura da nossa solução é apresentada na figura 1. É importante notar a existência de uma base de dados (**Botnet/Malware**) que é responsável por concentrar informações de toda a arquitetura proposta. Todos os componentes atuam diretamente ou indiretamente com os dados da base de informações da arquitetura. Na sequência, os componentes da arquitetura são descritos, agrupados em conjuntos funcionais.

### 3.1. Coleta e Análise de Malwares

Os componentes que correspondem a esse conjunto são responsáveis por coletar arquivos maliciosos e avaliar o seu funcionamento através de uma análise dinâmica de *malwares*. O componente **Coleta de Malware** é responsável por coletar arquivos suspeitos, sejam eles obtidos através de uma submissão manual (formulário Web), via uma *honeynet* ou ainda analisando URLs de *spams*. Os arquivos binários são a base do sistema. Tais arquivos permitem com que o comportamento de uma *botnet* seja traçado e convertido numa assinatura de rede.

O componente **Processador de Binários** é responsável por armazenar, na base de dados, os arquivos binários coletados. Para isso, inicialmente, os arquivos são pré-avaliados quanto a sua integridade e classificados quanto ao tipo do *malware* (assinatura de antivírus). Dessa forma, evita-se que arquivos duplicados ou corrompidos sejam adicionados na base de dados. No caso de um arquivo já estar presente na base de dados, apenas informações de data e horário são armazenadas, para que assim a frequência de ocorrência dos *malwares* seja analisada.

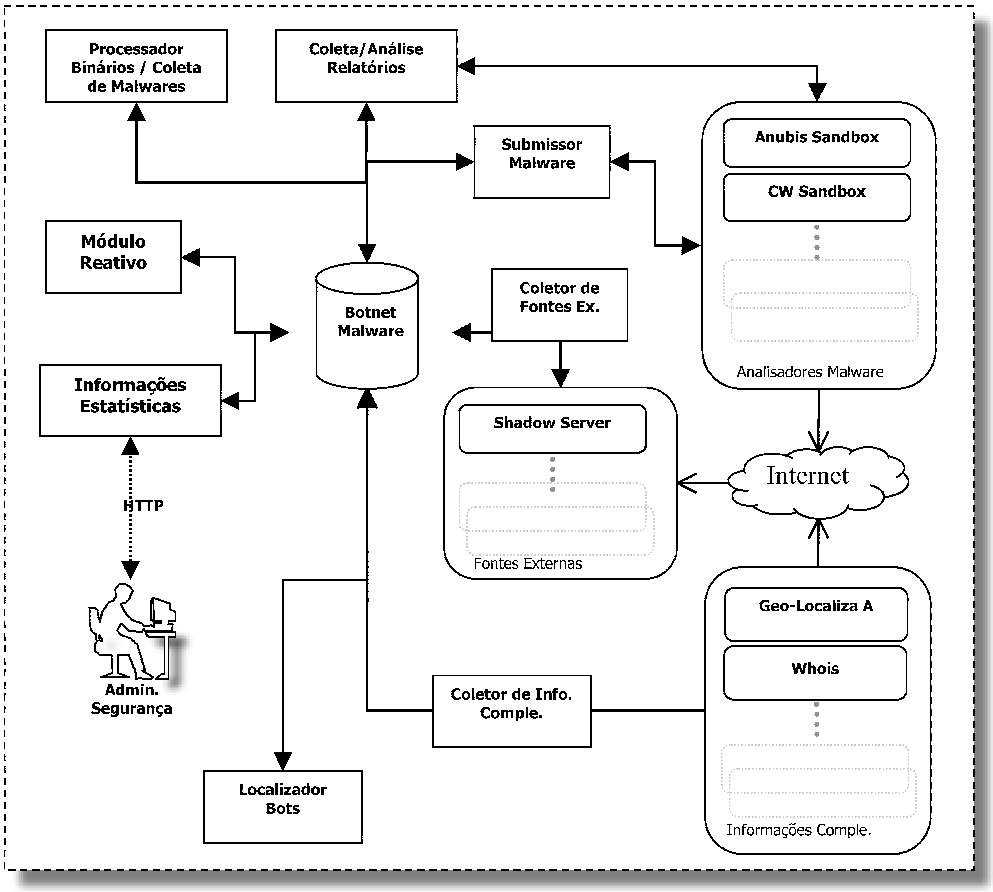


**Figura 1. Arquitetura da solução proposta.**

O **Submissor de Malware** é o componente responsável por submeter os arquivos maliciosos para os diferentes serviços de análise dinâmica de *malware*, tais como os *sandboxes* disponíveis na Internet e as ferramentas de verificação por assinaturas de vírus. Submeter os arquivos para diferentes fontes de análise permite um conjunto maior de informações que podem ser úteis no processo de geração de assinaturas de *bots*.

### 3.2. Processamento de Assinaturas

O processamento de assinaturas de *botnets* é desenvolvido por um conjunto de componentes responsáveis por obter as informações das ferramentas de análise de arquivos e desenvolver assinaturas. A obtenção das respostas das ferramentas de análise é realizada pelo componente **Coleta e Análise de Relatórios**. Além de obter os relatórios dos arquivos submetidos às fontes externas de análise, esse componente localiza possíveis padrões de comunicação de *bots*. As seguintes etapas são definidas para o efetivo processamento no componente:

1. Obtenção de relatórios - Cada um dos arquivos submetidos para análise em ferramentas *on-line* produzem como resposta um relatório de funcionalidades. Entretanto, a análise das funcionalidades do arquivo submetido não é imediata porque o processamento de cada arquivo depende das ferramentas e o tempo de resposta pode variar em muitas horas. Logo, o componente é responsável por fazer uma

checagem periódica e, assim que o relatório estiver disponível, armazená-lo localmente;

2. Geração de assinaturas - Esta etapa visa identificar padrões de comunicação do canal de controle de uma *botnet*, tendo como base os relatórios das ferramentas de análise. A primeira checagem consistem em verificar se o *malware* analisado possui um controlador, ou seja, se o *malware* em questão é um *bot*. Em caso positivo, o padrão de comunicação é mapeado e armazenado na base de dados. A comunicação com um controlador possui uma série de características. Na listagem 1, por exemplo, é ilustrada uma comunicação real observada num arquivo analisado. Nessa listagem, é possível observar características de uma conexão destinada ao endereço IP 83.133.119.206.

```
190.964397 192.168.0.2 83.133.119.206 TCP mtqp > 65520 [PSH, ACK]
Seq=91 Ack=229 Win=17292 Len=12
```

**Listagem 1. Informações sobre uma conexão realizada por um *bot*.**

Já na listagem 2 é apresentado o conteúdo de uma conversação (IRC) realizado por um *bot*, que também foi obtida na análise automatizada do *malware* anteriormente especificado.

```
NICK cucdaseb
USER b020501 . . :-Service Pack 3
JOIN &virtu
:u. PRIVMSG cucdaseb :!get http://updatemania.info/build/setup17.exe
:u. PRIVMSG cucdaseb :!get http://file0129.iwillhavesexygirls.com
   :88/erdown.txt
:u. PRIVMSG cucdaseb :!get http://pozeml.com/oc/box.txt
PING :j.
PONG :j.
JOIN &virtu
PING :j.
PONG :j.
JOIN &virtu
```

**Listagem 2. Comunicação com o canal de controle de uma *botnet*.**

O componente **Coleta de Fontes Externas** representa aqui um conjunto de recursos externos utilizados para coletar informações (assinaturas) de outras *botnets* detectadas por terceiros. Além de tornarem o sistema mais eficiente, é possível traçar paralelos entre as diferentes fontes de informações externas e até mesmo testar a sua efetividade. Já o componente **Coletor de Informações Complementares** busca analisar os dados presentes na assinatura, como endereço IP, e levantar informações relativas as mesmas. Os serviços acessados pelo **Coleta de Fontes Externas** é definido pelo componente **Informações Complementares**. As fontes de informações definidas pela arquitetura estão disponíveis na Internet e podem ser facilmente acessadas via rede, como por exemplo, fontes de geo-localização, bases de informações de roteamento e base `whois`.

### 3.3. Rastreamento e mitigação de *bots*

Os componentes desse conjunto buscam localizar máquinas infectadas na rede monitorada. O componente **Localizador de Bots** é responsável por obter os padrões de

comunicação detectados (assinaturas) e formatar um filtro para que os padrões sejam mapeados na rede local. Esse mapeamento consiste na inspeção de tráfego da rede com o objetivo de localizar máquinas comprometidas que apresentam o padrão de tráfego previamente traçado.

De forma complementar, o componente **Módulo Reativo** é responsável por impedir que as máquinas infectadas (*bots*), detectados pelo sistema, continuem ativas na rede, ou pelo menos que elas não sejam mais controladas pelos atacantes. Para isto, é implementada uma assinatura para ferramentas de segurança para cada máquina detectada. Esta assinatura de mitigação pode ser desenvolvida para diferentes ferramentas, tais como regras de filtro de pacotes e assinaturas de sistemas de detecção de intrusão. Como resultado, as assinaturas de mitigação de *botnets* para um conjunto de ferramentas utilizadas são armazenadas na base de dados do sistema, permitindo assim que o administrador de segurança seja munido de possíveis contramedidas para as *botnets* encontradas.

Adicionalmente, a arquitetura define o componente **Informações Estatísticas** que monitora o funcionamento da solução e também apresenta os dados referentes aos processamentos e ações desempenhadas pelo sistema. Esse componente consulta informações na base de dados do sistema e as apresenta para o administrador de segurança por meio de uma interface Web. Desta forma, é possível traçar estatísticas tanto dos arquivos binários coletados como dos controladores detectados e clientes infectados.

## 4. Implementação

Baseando-se na arquitetura apresentada na seção anterior, foi implementado um protótipo de sistema cuja visão geral, bem como um cenário de implantação, são apresentados na figura 2.

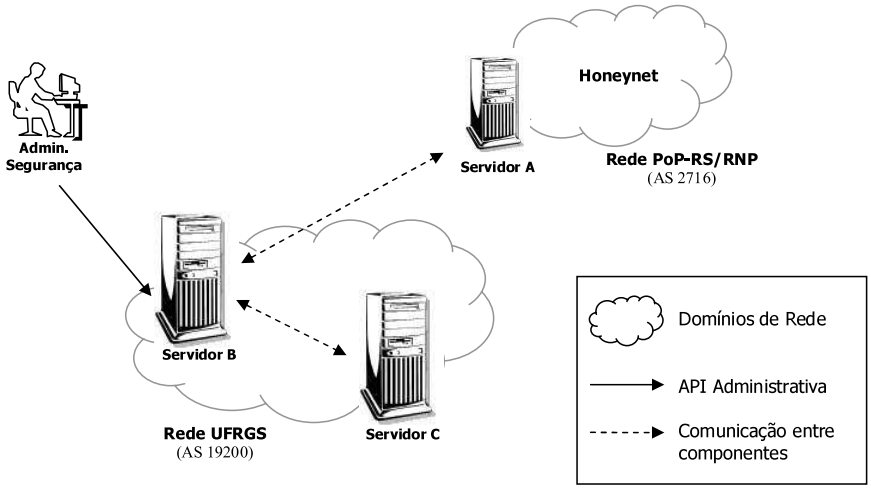


**Figura 2. Cenário de Implementação.**

O cenário é composto por três servidores localizados em domínios administrativos diferentes: o *servidor A* está sediado na Rede Nacional de Ensino e Pesquisa (Sis-

tema Autônomo 2716); já o *servidor B* e *servidor C* localizam-se na rede da Universidade Federal do Rio Grande do Sul (Sistema Autônomo 19200). Além dos servidores, o cenário apresenta um Administrador de Segurança que interage com o sistema disparando ações por meio de uma interface Web.

Todos os componentes da arquitetura foram implementados por softwares e *scripts* `shell`, `PHP` e `Perl`. Na sequência, a implementação de cada componente é apresentada. Por exemplo, o componente **Coleta de Malwares** foi implementado por uma *honeynet* - uma rede de máquinas com serviços vulneráveis aptas à coletarem informações dos mesmos. A *honeynet* foi implementada num sistema a parte, uma máquina dedicada - servidor A - exclusivamente para emular aplicações vulneráveis, utilizando o software `Nepenthes` [Nepenthes 2010]. Os arquivos coletados pelo `Nepenthes` são armazenados num diretório e constantemente processados pelo componente **Processamento de Malwares**.

O **Processamento de Malwares** insere o *malware* na base de dados da arquitetura e, também, faz uma verificação do arquivo em si. A verificação é realizada em duas etapas: a primeira tem o objetivo de descartar arquivos corrompidos ou vazios ocasionados por algum erro na coleta. Esse procedimento é instanciado pela ferramenta externa `file` disponível na maioria dos sistemas Unix. Já a segunda fase da verificação visa fazer uma pré-classificação do arquivo binário tendo como base a assinatura de um sistema de antivírus. Nessa implementação utilizou-se o antivírus `clamAV` que é disponibilizado segundo a licença GPL. A avaliação das funcionalidades dos *malwares* foi realizada utilizando as ferramentas externas e gratuitas `CWSandbox` e `Anubis`. O processo de submissão aos *sandboxes* é realizado pelo componente **Submissor de Malware**. Basicamente, esse componente foi implementado por *scripts* de submissão disponibilizados pelas próprias ferramentas originais.

O componente **Coleta e Análise de Relatórios** busca verificar a disponibilidade dos relatórios e encontrar indícios de um canal de controle de *botnet*. Tal componente foi implementado em três etapas, conforme pode ser observado na figura 3: *obtenção de relatórios* (Figura 3, item 1), *geração de assinaturas* (Figura 3, item 2) e *processamento de relatório* (Figura 3, item 4).

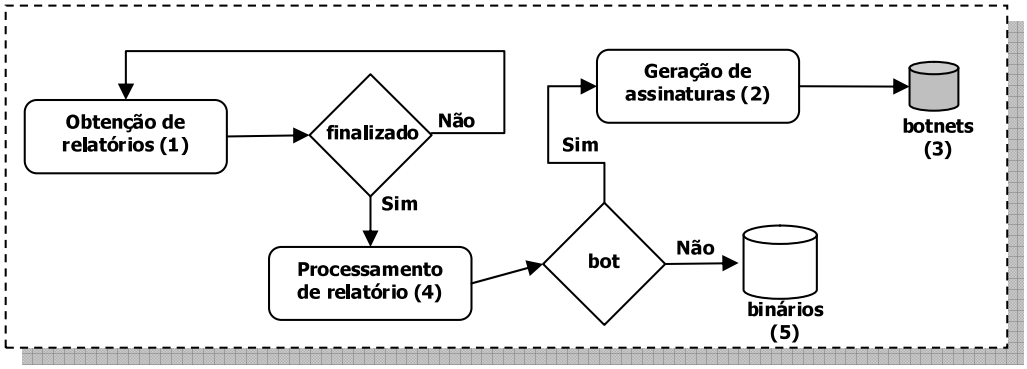


**Figura 3. Esquemático da coleta e análise de relatórios de funcionalidades de binários.**

*Obtenção de relatórios* - Essa etapa consiste em verificar se o arquivos previamente submetido para análise já foi processado pelo sistema de análise (*sandboxes*). Este

procedimento faz uma verificação periódica - a cada dez minutos - e verifica se o relatório já está disponível na URL especificada (obtida na fase de submissão). Em caso afirmativo, essas informações são salvas em memória e encaminhadas à próxima etapa (processamento de relatório), como no fluxo da figura 3. Especificamente, cada arquivo analisado produz o seguinte conjunto de dados: a) Relatório de funcionalidades do *malware* em formato XML (`CWSandbox`); b) Relatório de funcionalidades do *malware* em formato XML (`Anubis`); c) Tráfego de rede gerado pelo *bot* em formato `pcap` (`Anubis`).

De posse dessas informações, foi desenvolvida uma heurística para gerar assinaturas com base nos relatórios. Caso uma comunicação típica de *botnet* seja observada em ambos os relatórios, uma assinatura é automaticamente gerada. A listagem 3 ilustra uma assinatura automaticamente gerada pela implementação da arquitetura. Nessa listagem é possível observar um conjunto de informações das atividades do *bot*, tal como endereço de comunicação, porta e protocolo utilizados.

```
<xml>
<botnet_signature>
  <id>0234</id>
  <remoteaddr>85.11.143.208</remoteaddr>
  <remoteport>65520</remoteport>
  <protocol>TCP</protocol>
  <date>Ter Jul 27 15:57:43 UTC 2010</date>
  <comment>automatically generated</comment>
</botnet_signature>
</xml>
```

**Listagem 3. Assinatura de uma *botnet* gerada automaticamente com base à análise de relatórios de funcionalidades de *malwares*.**

As assinaturas desenvolvidas são armazenadas na base de dados para que outros componentes da arquitetura as utilizem. O componente **Coletor de Fontes Externas**, por exemplo, analisa as assinaturas e popula a base de dados com informações de geo-localização, consultando os serviços *on-line* `IPinfoDB` e `geoLocation` [GeoIP 2010]. As assinaturas também possibilitam que métodos de mitigação sejam implementados pelo componente **Módulo Reativo**. Esse último foi implementado traduzindo assinaturas das *botnets* para assinaturas do IDS `Snort`. A listagem 4 descreve uma regra gerada para o `Snort`, tendo em vista a assinatura da listagem 3.

```
alert tcp $HOME_NET any -> 85.11.143.208
any (content:'JOIN', msg:" Known Bot signature - BLOCKING";
flags:S; reference:url,www.botlog.org; threshold: type limit, track
by_src, seconds 3600, count 1; classtype:trojan-activity; sid:940006;
    rev:001;
fwsm dst, 30 days;)
```

**Listagem 4. Regra para detecção de um controlador de rede para o IDS *Snort*.**

Adicionalmente, regras para filtro de pacotes foram geradas pelo **Módulo Reativo**. As regras consistem numa série de comandos que podem ser executados no *firewall* para impedir que a máquina suspeita de estar infectada tenha acesso à rede (internamente ou externamente). Por fim, isso impede que a máquina comprometida possa ser

utilizada para propagar *malwares* na rede local ou ser utilizada como intermediária para ataques na Internet. A listagem 5 apresenta o formato de uma assinatura de bloqueio para o filtro de pacotes `Iptables`.

```
1  iptables -A security -s 143.54.224.30 -j DROP
2  iptables -A security -d 143.54.224.30 -j DROP
3  iptables -A PREROUTING -s 143.54.224.30 -p tcp -m physdev
4     --physdev-in eth0 -m        tcp --dport 80 -j DNAT
5   --to-destination 143.54.1.38:80
```

**Listagem 5. Regra para bloqueio de uma máquina comprometida no filtro de pacotes *Iptables*.**

A listagem 5 ilustra 3 regras ou instruções para o filtro de pacotes. Na primeira linha é feito o bloqueio para todas as conexões oriundas da máquina 143.54.224.30. Na segunda, o oposto, *i.e.*, são bloqueadas todas as conexões destinadas à máquina 143.54.224.30. A terceira regra faz um redirecionamento de todas as conexões TCP para uma página de aviso (Figura 4), a qual informa ao usuário o motivo do bloqueio, além de instruções e informações para contato. Todo esse procedimento tem como objetivo restringir uma máquina infectada o mais rapidamente possível, tão logo detectada pelo sistema.


**Figura 4. Informações disponibilizada para um usuário que possivelmente possua a sua máquina infectada por um *bot*.**

O componente **Estatística** é implementado por meio de uma página Web utilizando a linguagem PHP. Esta página faz a apresentação de diversos relatórios que dinamicamente são construídos com base nas informações do banco de dados. Além dos relatórios descritivos, foi possível também traçar mapas da localização geográfica

dos controladores. Os mapas são criados utilizando a ferramenta `plot-latlong` do CAIDA [CAIDA 2010]. Essa ferramenta permite plotar pontos em mapas geográficos tendo como base um par latitude-longitude (Figura 5). Dessa forma, optou-se por plotar um mapa com informações de controladores encontrados a cada dia. O conjunto destes mapas possibilitou a criação de uma animação que serve para entender a dinâmica dos controladores.

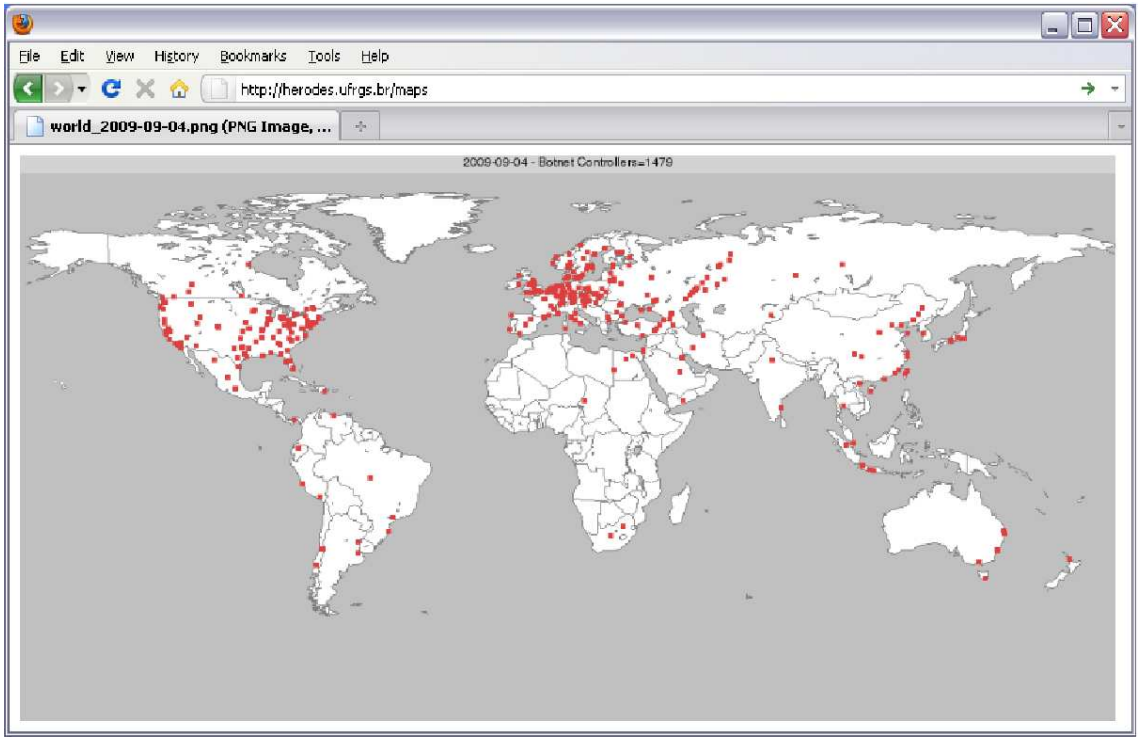


**Figura 5. Dispersão geográfica dos controladores de *botnets*.**

Em relação ao componente **Coletor de Fontes Externas**, não se tem conhecimento de uma base de informações (assinaturas) de *botnets* pública. Sendo assim, o mesmo foi implementado utilizando *blacklists* do grupo de pesquisa Shadowserver. Este grupo disponibiliza diariamente uma lista de controladores de rede. Logo, esse componente concentra-se em obter essa lista diariamente e a converte para o formato da arquitetura (listagem 3) para que posteriormente as mesmas sejam identificadas na rede.

Por fim, o componente **Localizador de Bots** foi implementado através da análise de fluxos de rede (tecnologia `Netflow` [Cisco 2010]). Cada assinatura de *botnet* foi mapeada, segundo a sintaxe dos fluxos, e periodicamente avaliada na rede. Logo, todas as máquinas que utilizam a assinatura de alguma *botnet* são identificadas. Dessa forma, o administrador da rede pode aplicar regras mitigatórias (**Módulo Reativo**) para evitar a propagação de ações maliciosas na rede.

## 5. Resultados

A definição da arquitetura, sobretudo, possibilitou que uma ferramenta fosse implementada e validada no contexto de uma rede acadêmica por um período de 8 meses, de Agosto de 2009 a Março de 2010. Nesse período foi observado um total de 4.149 máquinas com-

prometidas detectadas pela arquitetura. A figura 6 ilustra o número de *bots* identificados segundo a distribuição mensal de detecção.

É importante notar no gráfico que existem duas categorias ilustradas em cada mês representado. A primeira (*Total*) representa o número total de clientes identificados; já a categoria (*Únicos*) descreve a quantidade de clientes (endereçamento IP) distintos. Analisando o gráfico e considerando as características da rede monitorada - a universidade utiliza endereçamento fixo e globalmente roteável - foi possível concluir que existe uma grande variabilidade de *bots*. Ou seja, existem poucas máquinas reincidentes mensalmente.

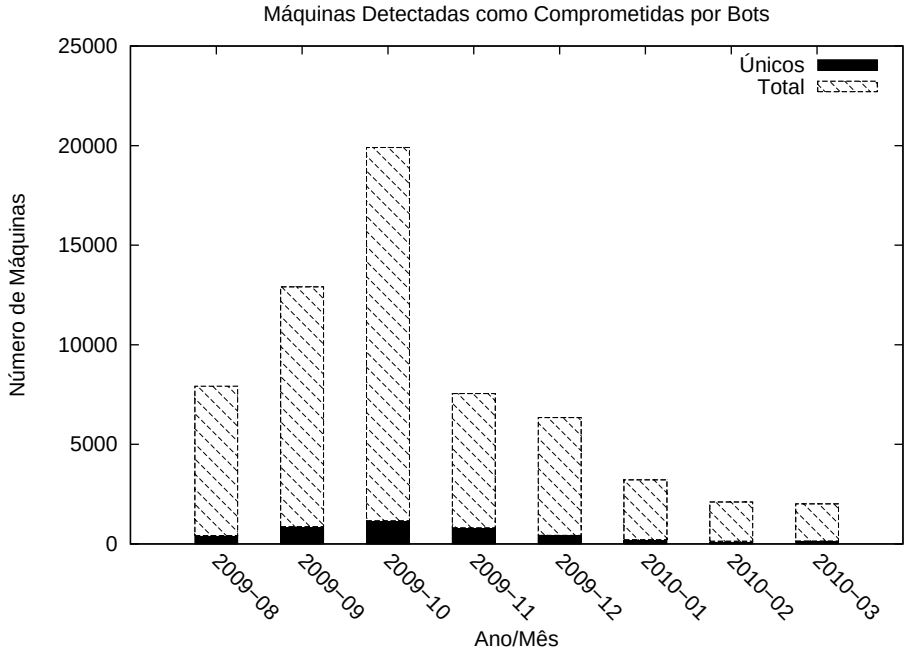


**Figura 6. Máquinas comprometidas detectadas.**

Ainda a respeito das máquinas comprometidas, observou-se que a maioria das mesmas acessam diferentes controladores de rede (IP). Um total de 42,5% dos clientes infectados acessaram outros controladores. E nesse montante, analisando apenas as máquinas que acessaram mais de um controlador, constatou-se que em média foram realizados 10 acessos a diferentes controladores. Esse comportamento é típico de técnicas que buscam incrementar a disponibilidade da *botnet* ou ainda, técnicas mais sofisticadas como `fast-flux` [Nazario and Holz 2008].

A figura 7 apresenta o número de assinaturas de *botnets* importadas no sistema mensalmente. Essas assinaturas importadas em última análise são endereços de controladores de *botnets* visam aumentar a eficácia da arquitetura frente às redes maliciosas. É interessante notar na figura 7 que existe uma grande diversidade de assinaturas distintas, ou seja, que não foram classificadas na categoria *Únicos*. Esse fato reflete do dinamismos das redes maliciosas que constantemente estão recrutando novos controladores para a administração das redes maliciosas.

Na sequência é feita uma avaliação desses controladores - obtidos através da análise de *malwares* - tendo em vista características de conexões. Na tabela 1 são

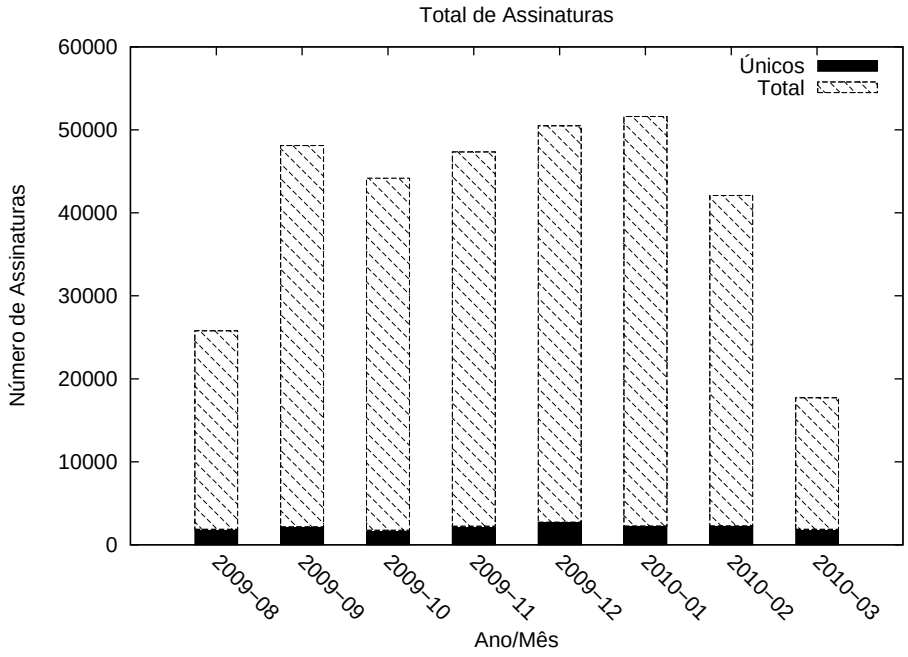


**Figura 7. Assinaturas importadas para a arquitetura.**

sumarizados os controladores mais frequentes identificados nos *malwares* analisados. É importante notar que o último octeto do IP foi anonimizado para disponibilização de informações neste artigo.

**Tabela 1. Controladores mais frequentes dos *malwares* coletados.**

| Acessos | Possível controlador de *botnet* |
|---|---|
| 392 | 83.68.16.X |
| 7 | 83.68.16.X |
| 6 | 125.214.65.X |
| 5 | 74.63.78.X |
| 4 | 69.65.19.X |
| 4 | 69.64.147.X |
| 4 | 141.152.124.X |
| 2 | 128.130.56.X |
| 2 | 103.72.47.X |
| 2 | 103.72.47.X |

Na tabela 1 é possível identificar que poucos controladores de *botnet* são responsáveis por um grande número de *bots*, ou seja, poucos controladores - super controladores - são responsáveis pelo controle de um grande número de máquinas comprometidas. O endereço que se destaca - 83.68.16.X - corresponde a um servidor alocado numa empresa de hospedagem na América do Norte. Suspeita-se que o referido IP seja utilizado como um balanceador de carga ou até mesmo atuando como um elemento centralizador utilizado por um grupo criminoso.

A análise temporal da dispersão dos controladores - segundo a plotagem diária da localização dos controladores de *botnets* - apontou fatos interessantes: a) boa parte

dos controladores ativos estão localizados em países desenvolvidos, possivelmente em empresas de hospedagens (*co-location*); b) não foi observado grandes variações quando a localização dos controladores, os mesmos migram de endereçamento mas permanecem numa mesma região geográfica.

Adicionalmente, a comunicação do controlador de *botnet* com os seus clientes se dá por meio de portas de difícil controle, tal como a porta 80/TCP (HTTP). Essas portas dificilmente são bloqueadas no filtro de pacotes das instituições e permitem o livre fluxo de informações. Isso demonstra uma preocupação dos atacantes em tornar o tráfego de controle de *botnet* mais difícil de ser detectado por sistemas tradicionais de mitigação de atividades maliciosas.

## 6. Considerações Finais e Trabalhos Futuros

A avaliação preliminar da ferramenta desenvolvida em termos de identificação de assinaturas e mapeamento de máquinas comprometidas apresentou resultados satisfatórios. As máquinas comprometidas foram rapidamente identificadas (com tempo médio inferior a 2 dias) e restringidas de acessar a rede por meio de filtros de acesso. Os responsáveis pela máquina comprometida eram notificados por e-mail e por meio de uma página Web informativa. Adicionalmente, a base de dados definida pela arquitetura apresentou informações que possibilitam entender melhor características e similaridades das diferentes *botnets* detectadas [Ceron et al. 2009].

Apesar do observado acima, ainda existem melhorias que devem ser realizadas na arquitetura. Como trabalho futuro, destaca-se o desenvolvimento e avaliação de novas metodologias de geração de assinatura de *malwares*. Tais metodologias podem descrever novos algoritmos ou heurísticas para identificar assinaturas de *bots*, tendo como base a análise de arquivos binários já realizada e presente na base de dados. Dessa forma, *malwares* não identificados como *bots* podem futuramente ser reclassificados com novos algoritmos a serem desenvolvidos. Adicionalmente, novas extensões podem ser incorporadas a arquitetura, tais como módulos que contemplem novos serviços ou metodologias de análise de arquivos maliciosos. Seria igualmente interessante desenvolver uma linguagem unificada para descrever a avaliação de arquivos maliciosos de diferentes fontes de análise. Com isso, as funcionalidades mapeadas por meio de diversos serviços *on-line*, por exemplo, poderiam ser unificadas num único arquivo com formatação padronizada. Dessa forma, o processamento de informações oriundas de análise de *malwares* poderia ser avaliadas de forma mais otimizada por mecanismos e algoritmos.

Por fim, a arquitetura pode ser aprimorada com a incorporação de novos módulos funcionais ao sistema. Módulos específicos podem ser aprimorados sem afetar o funcionamento da arquitetura auxiliando no processo extremamente dinâmico de detecção e mitigação de botnets.

## Referências

CAIDA (2010). The Cooperative Association for Internet Data Analysis. Disponível em: http://www.caida.org. Acesso em: Julho de 2010.

Ceron, J., Granville, L. Z., and Tarouco, L. (2009). Taxonomia de malwares: Uma avaliação dos malwares automaticamente propagados na rede. In *SBSeg 2009 - Artigos Completos/Full Papers*.

Cisco (2010). Cisco Netflow - Cisco Systems.

GeoIP (2010). GeoIP API - Location from IP. Disponível em: http://www.geoipapi.com/. Acesso em: Junho de 2010.

Goebel, J. and Holz, T. (2007). Rishi: identify bot contaminated hosts by irc nickname evaluation. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, Berkeley, CA, USA. USENIX Association.

Gu, G., Perdisci, R., Zhang, J., and Lee, W. (2008a). Botminer: clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *SS'08: Proceedings of the 17th conference on Security symposium*, pages 139–154, Berkeley, CA, USA. USENIX Association.

Gu, G., Porras, P., Yegneswaran, V., Fong, M., and Lee, W. (2007). BotHunter: Detecting malware infection through ids-driven dialog correlation. In *Proceedings of the 16th USENIX Security Symposium (Security'07)*.

Gu, G., Zhang, J., and Lee, W. (2008b). BotSniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*.

Holz, T., Steiner, M., Dahl, F., Biersack, E., and Freiling, F. (2008). Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm. In *LEET'08: Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, pages 1–9, Berkeley, CA, USA. USENIX Association.

Kreibich, C., Kanich, C., Levchenko, K., Enright, B., Voelker, G. M., Paxson, V., and Savage, S. (2009). Spamcraft: An inside look at spam campaign orchestration. In *Proceedings of the Second USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET)*, Boston, USA.

Nazario, J. and Holz, T. (2008). As the net churns: Fast-flux botnet observations. In *3rd International Conference on Malicious and Unwanted Software Malware'08*.

Nepenthes (2010). Nepenthes - finest collection . Disponível em: http://nepenthes.carnivore.it. Acesso em: Junho 2010.

Wang, P., Wu, L., Cunningham, R., and Zou, C. C. (2010). Honeypot detection in advanced botnet attacks. *Int. J. Inf. Comput. Secur.*, 4(1):30–51.

Wurzinger, P., Bilge, L., Holz, T., Goebel, J., Kruegel, C., and Kirda, E. (2009). Automatically generating models for botnet detection tr-iseclab-0609-001. In *Lecture Notes in Computer Science*, pages 108–125.