Taxonomia de Malwares: Uma Avaliação dos Malwares Automaticamente Propagados na Rede

João M. Ceron[1,2], Lisandro Granville[2], Liane Tarouco[2]

[1] TRI - Time de Resposta a Incidentes de Seguranca da
Universidade Federal do Rio Grande do Sul
Centro de Processamento de Dados
Rua Ramiro Barcelos, 2574 – Porto Alegre – RS – Brasil

[2]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

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

Abstract. *Malware's automatic propagation is a serious threat on the Internet and it is responsible for a large number of compromised systems. Security tools like antivirus and firewalls have evolved considerably in recent times. However, the new generation of malware has the ability of disable antivirus software and hiding themselves into the system. This paper aims at identifying the most used features of malwares as well as evaluating the effectiveness of the available antivirus systems related to such malwares. As a result, we identify malwares behavior's trends and quantify the effectiveness of antivirus software's.*

Resumo. *A propagação atomática de malwares é uma séria ameaça na Internet e responde por boa parte dos comprometimentos de sistemas computacionais. As ferramentas de segurança como antivírus e frewalls tem evoluído consideravelmente nos últimos tempos, no entanto os malwares não ficaram para trás. A nova geração de malwares possui a capacidade de desabilitar softwares antivírus e ocultar-se no sistema. Este trabalho busca identificar as funcionalidades dos malwares mais utilizadas avaliando também a eficácia dos sistemas de antivírus com relação às mesmas. Deseja-se com isso observar tendências comportamentais dos malwares e a eficácia dos sistemas de proteção com relação a essas ameaças.*

1. Introdução

Malware é um termo genérico que abrange todos os tipos de programa especificamente desenvolvidos para executar ações maliciosas em um computador [CERT.br 2009]. A denominação *malware* comumente é empregada para se referir a vírus, *worms*, *spywares*, *trojans*, em geral, todo e qualquer software malicioso. De fato, a popularização da Internet possibilitou um maior número de usuários conectados na rede, mas por outro lado, também aumentou o número de sistemas potencialmente vulneráveis a infecção.

Os sistemas computacionais vulneráveis são facilmente explorados por atacantes que utilizam códigos maliciosos. Existem diversas técnicas para que um software malicioso seja executado nos sistemas; as mais comuns são baseadas em engenharia social, exploração remota de serviços, *cross site scripting* e injeção de código

[Bächer et al. 2007]. Uma vez que o sistema é comprometido, o mesmo fica sob o domínio do atacante que pode realizar as mais diversas ações ilícitas: roubo de dados sigilosos, envio de *spam*, até mesmo instalar programas adicionais para controlar remotamente o sistema infectado.

A propagação autonômica de *malwares* é umas maneiras mais utilizadas para o comprometimento de sistemas [Masud et al. 2008]. *Malwares* como *worms* e *bots* usualmente propagam-se de maneira muito similar: buscam por máquinas vulneráveis a fim de explorar vulnerabilidades e por fim comprometer o sistema. Os *malwares* automaticamente propagados pela rede possuem uma característica particular: são muito heterogêneos. É possível observar variantes de códigos maliciosos antigos, mas também - e mais freqüente ultimamente - *malwares* que exploram vulnerabilidades não conhecidas, também denominadas de *zero-day exploit*. A análise desse tráfego malicioso é muito importante para entender técnicas e características amplamente utilizadas pelas diferentes famílias de *malwares*.

A defesa mais importante contra códigos maliciosos são os sistemas de antivírus. Os antivírus tipicamente baseiam-se em um banco de dados de assinaturas para caracterizar um *malware* conhecido. Entretanto quando um *malware* não possui uma assinatura é necessário analisá-lo e entender como o seu comportamento afeta o sistema, para enfim desenvolver uma assinatura. Além da assinatura do *malware* é importante conhecer suas funcionalidades para que a remoção do *malware* do sistema seja efetivamente realizada.

A abordagem tradicional para analisar o comportamento de um programa é executá-lo num ambiente restrito - denominado *sandbox* - e observar suas ações. Os *sandboxes*, em particular, possuem a capacidade de executar um programa num ambiente virtualizado e fornecem relatório detalhado de suas ações.

Quando se deseja identificar funcionalidades de um *malware* como, por exemplo, a desativação do sistema de antivírus é necessária correlacionar os eventos apresentados no relatório de funcionalidades do *malware*. Logo, este trabalho busca identificar características utilizadas pelos *malwares* com base na correlação de ações desempenhadas pelo mesmo. Através disso busca-se identificar características mais utilizadas pelos *malwares*, como por exemplo, inibição do sistema de antivírus, funcionalidades de registro de teclas, capacidade de ocultação no sistema, entre outras.

Em complementação, será observada a eficácia de 39 sistemas de antivírus frente aos *malwares* que são propagados automaticamente pela rede. Muitas vezes os *malwares* que exploram certas vulnerabilidades são disseminados antes que as companhias de antivírus tenham a efetiva vacina para tal *malware*. Este trabalho mapeia o quão eficientes são os sistemas de antivírus atuais com relação aos *malwares* coletados. Este artigo apresenta resultados da coleta de *malwares* obtidos no período entre dezembro de 2008 a maio de 2009, totalizando 5 meses. Todos os *malwares* foram coletados sob o âmbito da rede da RNP (Rede Nacional de Ensino e Pesquisa) com auxílio uma *honeynet*. Os arquivos foram avaliados segundo ferramentas de análise comportamental de binários e sistemas antivírus disponível *on-line*. A compilação final dos resultados se deu com auxílio de programas desenvolvidos pelos autores, o que permitiu identificar as principais funcionalidades dos *malwares* e a eficácia dos sistemas antivírus em relação aos mesmos.

O restante deste artigo está organizado da seguinte forma. Na Seção 2 é apresen-

tada uma síntese dos trabalhos relacionados. Na Seção 3 são apresentadas características e funcionalidades dos *malwares*. Na Seção 4 a estrutura de coleta e análise de binários é descrita, bem como, características de implementação da análise. Por fim, as conclusões e trabalhos futuros são discutidos na Seção 5, onde o artigo é encerrado.

2. Trabalhos relacionados

O desenvolvimento de técnicas para analisar arquivos executáveis, sobretudo, arquivos supostamente maliciosos, é um tema que tem gerado interesse na comunidade científica. No entanto a análise de arquivos maliciosos não é trivial. Os *malwares* estão constantemente em evolução e implementando novas técnicas para que seu funcionamento permaneça desconhecido. De fato, existem muitos trabalhos que endereçam o problema de analisar um arquivo malicioso para descobrir o seu funcionamento no sistema.

No trabalho descrito em [Bächer et al. 2007], os autores apresentam uma técnica de análise sem a execução do executável, apenas minerando informações do próprio arquivo suspeito. Já em [Moser et al. 2007], é ilustrado uma técnica - também sem a execução do arquivo - onde informações são colhidas baseando-se no fluxo de execução das instruções do binário em questão. Ultimamente, a técnica de analisar arquivos de forma estática, ou seja, sem a execução do mesmo, tem se mostrado limitada para os recentes *malwares* que utilizam técnicas de ofuscação de código. Diante disto, outras técnicas estão sendo empregadas, como é o caso da análise dinâmica.

A análise dinâmica busca observar as propriedades do comportamento do *malware* executando o mesmo em ambiente simulado. Recentemente, a análise dinâmica de *malwares* obteve grandes avanços graças a novas ferramentas que utilizam técnicas, como virtualização, e possibilitam avaliar executáveis de forma escalável. A análise dos resultados produzidos por estas ferramentas, aliadas ao processo de analisar uma grande quantidade de binários suscitou novos trabalhos na área. Em sua maioria, os trabalhos que avaliam o comportamento do *malwares*, visam identificar possíveis variantes de *malwares*[1].[Bayer et al. 2009] apresentam um sistema para identificar dinamicamente variantes de *malwares* agrupando-os com base ao seu comportamento. Para isso é implementado o mapeamento de chamadas de sistema de baixo nível, classificando-as segundo algoritmo desenvolvido pelos autores. Em outro trabalho, descrito em [Siddiqui et al. 2008], é apresentado um sistema similar, porém com uma metodologia um tanto quanto diferente: o sistema busca por *malwares* homólogos utilizando um antivírus para classificar o conjunto de treino.

Outra abordagem de análise dinâmica de *malwares* é apresentada em [Willems et al. 2007]. Nele, os autores buscam identificar as diferenças entre *malwares* e executáveis benignos tendo como base funcionalidades e características no processo de execução. Em contrastes às soluções descritas, o objetivo do nosso trabalho é identificar as ações desempenhadas pelos *malwares* e não discriminá-los entre famílias de *malwares* nem tampouco especificar se o mesmo é benigno ou não.

A abordagem que mais se aproxima da proposta deste trabalho é a apresentada por Rajab *et al.* [Rajab et al. 2007]. No referido trabalho no qual o autor apresenta métodos

[1]Uma variante de código malicioso corresponde a pequenas modificações - adição de novas funcionalidades - a um *malware* cujo comportamento já é caracterizado.

para monitorar e mitigar botnets, é brevemente abordada uma taxonomia dos softwares infectados por *bots*. A análise dos softwares tem como base a monitoração de chamadas de sistemas específicas aliado a modificações do registro do sistema, ambas obtidas num ambiente virtual controlado. Nosso trabalho, porém, busca fazer um estudo mais detalhado e expansivo: discriminando um maior número de funcionalidades e, com mais abrangência analisar arquivos suspeitos observados na rede.

3. Análise de código malicioso

O interesse por entender o funcionamento e características de um arquivo malicioso é uma preocupação recorrente. A maioria dos produtos de segurança como antivírus e sistemas de detecção de intrusão trabalham com assinaturas - uma seqüência característica de bytes - para identificar um código malicioso [Stinson e Mitchell 2008]. Empresas de sistemas de antivírus estão constantemente analisando novos *malwares* para a identificação de sua assinatura. Cada nova assinatura de *malware* encontrada é incorporada à base de dados de assinaturas e por fim propagadas aos usuários do sistema de antivírus. Como não existe uma base de assinaturas unificadas cada empresa desenvolve a sua, o que influencia diretamente na eficiência dos antivírus. Sabe-se da existência de algumas iniciativas para avaliar a eficiência de detecção dos diferentes antivírus existentes [Raghunathan e Partha 2009].

Uma iniciativa interessante e consolidada na comunidade de segurança da informação é o VirusTotal [VirusTotal 2009]. O VirusTotal é uma ferramenta que permite a análise um arquivo binário em 39 diferentes sistemas de antivírus. Além disso, o sistema informa a assinatura do *malware* analisado, caso o mesmo tenha sido detectado como malicioso. O resultado do processamento da ferramenta é apresentado no relatório apresentado na Figura 1.

Arquivo **8d7329c9580407c74200fadff34f0a63** recebido em **2009.05.09 21:54:59**
Resultado: **37/39 (94.87%)**

Antivírus	Versão	Última Atualização	Resultado
a-squared	4.0.0.101	2009.05.09	Generic.Banker.OT!IK
AhnLab-V3	5.0.0.2	2009.05.09	Win32/Virut.D
AntiVir	7.9.0.166	2009.05.08	W32/Virut.Gen
Antiy-AVL	2.0.3.1	2009.05.08	-
Authentium	5.1.2.4	2009.05.09	W32/Backdoor2.ASZB
Avast	4.8.1335.0	2009.05.09	Win32:Virut
AVG	8.5.0.327	2009.05.09	Win32/Virut
BitDefender	7.2	2009.05.09	Backdoor.SDBot.DFDQ
CAT-QuickHeal	10.00	2009.05.09	W32.Virut.D
ClamAV	0.94.1	2009.05.09	W32.Virut.ia

Figura 1. Análise de um binário em diferentes sistemas de antivírus demonstrando assinaturas detectadas.

Por questão de espaço, o relatório acima, representa apenas uma parte do fornecido pela ferramenta. No entanto, é possível observar na parte superior que o sistema analisou o binário em 39 diferentes antivírus, e 37 deles detectaram uma assinatura para o *malware*. O relatório descreve informações relativas ao sistema de antivírus utilizado, a versão do software, a última atualização da base de dados de vacinas - e o tipo de assinatura detectado. A análise nos diferentes sistemas é útil para a comparação da eficiência dos diferentes sistemas de antivírus em relação ao um arquivo malicioso em específico.

Além da busca por assinaturas de um arquivo malicioso é possível analisar um binário utilizando outras abordagens, como por exemplo, a análise comportamental. A análise comportamental consiste em traçar os eventos ou ações desempenhadas por um arquivo binário no sistema operacional no qual é executado. Essa análise do funcionamento de arquivos binários pode ser realizada de duas diferentes maneiras: de forma estática ou dinâmica.

A análise estática consiste em avaliar o funcionamento de um programa sem a execução do mesmo, baseando-se apenas no seu código. As técnicas mais tradicionais consistem em extrair instruções do código ASSEMBLY do programa e inferir conclusões com base na seqüência das instruções [Siddiqui et al. 2008]. No entanto, a principal deficiência desse método é a possibilidade do código de máquina analisado não refletir o mesmo comportamento que o *malware* apresenta quando executado. Em particular, a análise estática é pouco eficiente para programas que utilizam técnicas de ofuscação ou polimorfismo [Linn e Debray 2003] [Newsome e Song 2005], como é o caso do *malware* Conficker [Leder e Werner 2009]. Tais limitações da análise estática incitaram o surgimento de técnicas complementares, como é o caso da análise dinâmica.

A análise dinâmica, por outro lado, consiste em observar as características funcionais do *malware* através da sua execução num ambiente controlado. Segundo o trabalho [Willems et al. 2007] as principais metodologias de análise dinâmica baseiam-se em: (a) comparação do *status* do sistema operacional antes e imediatamente após a execução do arquivo; e (b) monitoramento das ações em tempo de execução. Na primeira abordagem busca-se fazer uma comparação do sistema operacional completo identificando alterações causadas pelo arquivo binário executado. Como resultado, essa técnica traça uma visão geral das funcionalidades do binário, como arquivos criados, dados removidos, entre outros. Essa solução, entretanto, não determina mudanças dinâmicas intermediárias ao estado inicial e final da comparação do sistema. Mudanças como a criação de arquivos durante a execução e a deleção de arquivos antes do final do processo são transparentes a essa análise.

Por outro lado, na segunda abordagem cuja monitoração das ações do *malware* é dada durante a execução, tais ações são traçadas. Mesmo sendo mais complexa de implementar, a análise de binários durante a execução do mesmo, vem popularizando-se devido ao bom resultado da técnica perante códigos polimórficos e ofuscados. A principal limitação da análise dinâmica de *malware* é a possibilidade de executar apenas uma amostra de binário de cada vez. Afinal, a execução de outros binários no mesmo ambiente controlado dificulta a distinção das ações de cada *malware*. Recentemente, com os melhoramentos de tecnologias de virtualização de sistemas, a análise dinâmica de *malwares* ganhou outra dimensão. De fato, a facilidade de reconstruir um ambiente virtualizado propiciou o surgimento de ferramentas mais detalhistas e escaláveis para a análise dinâmica,

como é o caso dos *sandboxes*.

Ferramentas como CWSandbox [Willems et al. 2007], Norman Sandbox [Norman 2009], Anubis [Alkassar e Siekmann 2008] são ferramentas automatizadas para análise dinâmica de arquivos binários. Essas ferramentas caracterizam-se por executar um arquivo num ambiente controlado registrando em forma de relatório das ações realizadas pelos *malwares*. Como característica, os *sandboxes* tradicionalmente simulam o sistema operacional Windows, já que a grande maioria de *malwares* existentes é escrita para o mesmo [Barford e Blodgett 2007]. Funções comuns observadas pelo *sandbox* descrevem funcionalidades como:

- Arquivos criados ou modificados
- Acessos ou modificações a chave do registro do sistema
- Bibliotecas dinâmicas carregadas
- Áreas da memória virtual utilizada
- Processos criados
- Conexões de rede instanciadas
- Dados transmitidos pela rede

Os relatórios disponibilizados pelos *sandboxes* podem variar bastante devido a particularidades de implementação dos mesmos [Willems et al. 2007]. Por exemplo, o *sandbox* Norman simula um computador conectado na rede reimplementando partes do núcleo do sistema Windows emulado [Stapp et al. 2006]. Já o *sandbox* Anubis é implementado com base sistema Qemu [Bellard 2005] emulando a arquitetura i386. Além das características inerentes às particularidades do sistema emulado, os *sandboxes* especificam um conjunto limitado de chamadas de sistemas a ser monitorada.

A grande parte das ferramentas de *sandbox* existentes, mesmo as comerciais, disponibiliza uma interface sem restrição de acesso para a avaliação de resultados. Por meio de uma interface Web, por exemplo, os usuários podem submeter arquivos suspeitos à análise dinâmica do *sandbox*. Como resultado, a ferramenta disponibiliza um relatório com as ações desempenhadas pelo arquivo submetido. De fato, algumas ferramentas comerciais como o Norman Sandbox apresentam relatórios pouco descritivos na versão disponível na Web, em contraste à versão comercializada. Já ferramentas como o Anubis e CWSandbox disponibilizam um nível maior de detalhamento. Na seqüência são apresentados exemplos de relatórios disponibilizados pelo *sandbox* Norman Sandbox e Anubis resultante da análise do mesmo arquivo.

A Figura 2 apresenta um relatório de funcionalidades de um *malwares* analisado pela ferramenta Norman Sandbox. Como pode ser observado, o resultado da análise do *sandbox* disponível gratuitamente na Web é pouco descritivo apresentando apenas linhas gerais da análise. Assinatura do *malware*, método de compressão do binário, tamanho do arquivo, informações sobre os processos, são apenas alguns exemplos de informações descritas pelo sandbox. Por outro lado, o relatório disponibilizado pela ferramenta de *sandbox* Anubis é muito mais detalhado. A Figura 3 apresenta apenas um fragmento do relatório do Anubis, mesmo assim já é possível ter uma noção do nível de detalhamento fornecido pela ferramenta. Nesse relatório são evidenciadas duas características de execução do binário analisado: na parte superior todos os arquivos modificados no sistema; e na parte inferior os processos instanciados pelo mesmo.

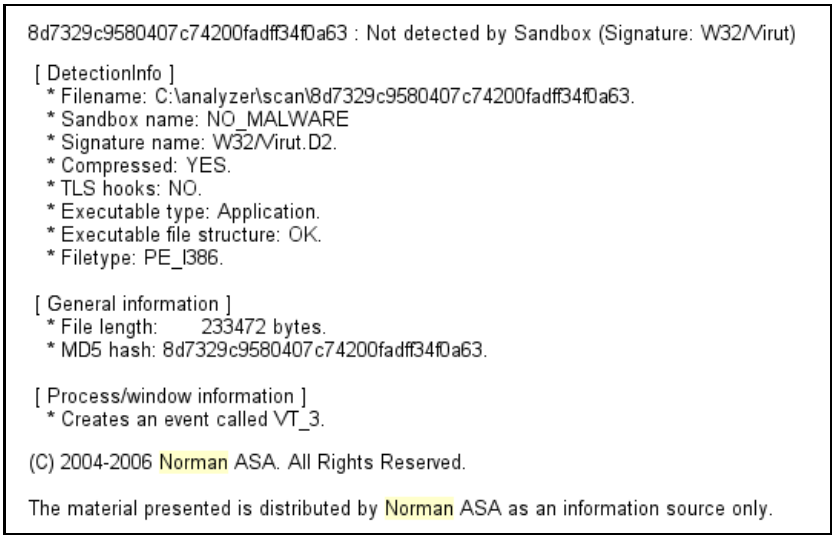

Figura 2. Relatório das ações de um *malware* disponibilizado pelo *sandbox* Norman.

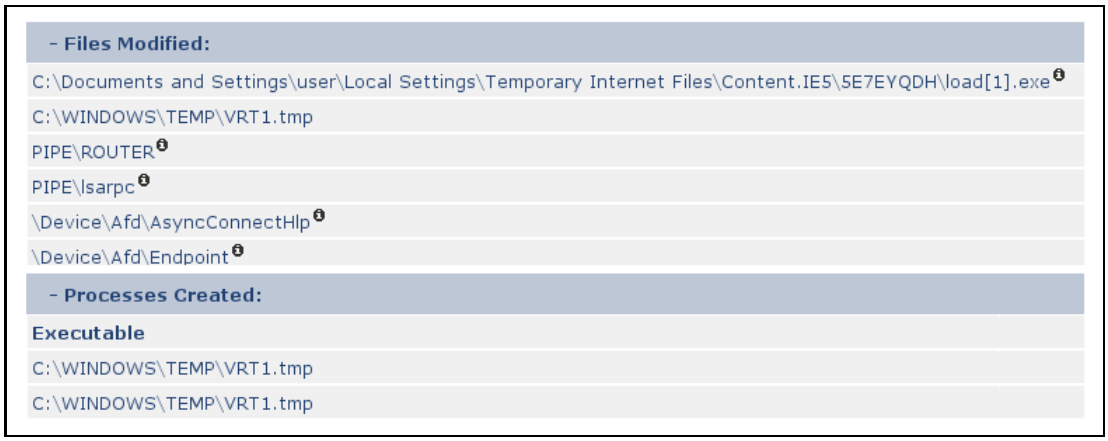

Figura 3. Fragmento de um relatório das ações de um *malwares* disponibilizado pelo *sandbox* Anubis.

As diferentes técnicas de análise de arquivos binários descritas acima, em última análise, possuem o mesmo objetivo: lidar com a árdua tarefa de identificar as funcionalidades de um arquivo binário. A utilização das distintas metodologias, em particular, pode ser combinada e servir de base para identificar novas informações desse conjunto de dados como proposto por este trabalho.

4. Implementação do sistema de análise

Para que os objetivos deste trabalho fossem atingidos, foi desenvolvido um processo completamente automatizado de coleta e análise de binários. Todo o processo consiste em 7 etapas de características distintas, a saber: coleta de binários, análise dinâmica, mapeamento de funcionalidades, sumarização de resultados, análise de assinaturas, análise de resultados e base de dados estatísticos. O fluxo de informação entre as etapas do processo é representado na Figura 4 e na seqüência descrita em detalhes.

a) **Coleta de binários**: Como anteriormente comentando, um dos objetivos deste traba-

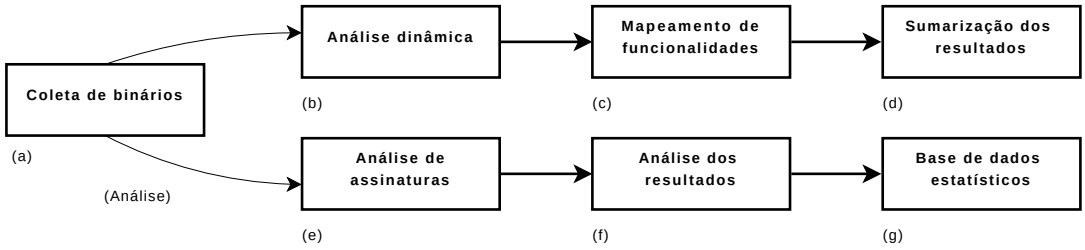

Figura 4. Visão geral do processo implementado para análise dos resultados.

lho é analisar funcionalidades de *malwares* que são propagados de forma autônoma na rede. Tradicionalmente a propagação de *malwares* é desempenhada pela exploração remota de vulnerabilidades conhecidas. Uma metodologia que tem se demonstrado bastante efetiva para a coleta deste tipo de *malwares* são as *honeynets* [Virti et al. 2006] [Ceron et al. 2008]. As *honeynets* são redes compostas por *honeypots*, sistemas vulneráveis que funcionam como uma isca para ataques. Os *honeypots* permitem capturar tentativas de exploração de vulnerabilidades conhecidas e interagir com o ataque de modo a coletar o máximo de informação a respeito da sondagem. O software Nepenthes [Abu Rajab et al. 2006] implementa essa idéia emulando vulnerabilidades conhecidas em serviços de rede. A ferramenta detecta tentativas de exploração e interage com o ataque de forma limitada. Baseado numa análise automatizada do *payload* do ataque, o software extrai informações suficientes para obter uma cópia do binário atacante. O processo de coleta e obtenção de arquivos binários implementado pelo software Nepenthes é rubusto o suficiente para coletar *malware* que se propagam de forma autônoma na Internet, explorando vulnerabilidades de serviços conhecidos.

A *honeynet* implementanda para a coleta de binários corresponde a 254 *honeypots* - máquinas emuladas com serviços vulneráveis - com serviços acessíveis na rede sem nenhuma restrição. Como resultado, cada máquina emulada coletava binários suspeitos oriundo de ataques a os serviços vulneráveis disponíveis. Por segurança, os binários coletados eram constantemente transmitidos via conexão segura para outro servidor que faz o papel de servidor central de *malwares*. Assim que um novo binário é inserido no servidor central de *malwares* dois processos são disparados concorrentemente: análise dinâmica e análise por assinaturas.

b) **Análise dinâmica**: A análise dinâmica busca identificar as ações desempenhadas por um arquivo suspeito num sistema operacional. Esta etapa foi realizada com auxílio da ferramenta de *sandbox* Anubis. Tal *sandbox* tem sido utilizado pela comunidade de segurança de informação e tem apresentado resultados consistentes, como descrito pelos autores em [Willems et al. 2007] [Bächer et al. 2007]. O Anubis possui uma interface Web bastante simplificada o que permitiu a automatização do envio de *malwares* para a análise. Como resultado, a ferramenta disponibiliza um relatório de ações realizadas para cada binário. O grande volume de relatórios fomentou a necessidade de desenvolver uma ferramenta para mineração de dados e por fim mapear as funcionalidades do *malware*, conforme descrito na próxima etapa.

c) **Mapeamento de funcionalidades**: O objetivo desta etapa é fazer um processamento nos relatórios do *sandbox* e mapear as ações no sistema com possíveis funcionalidades. Por exemplo, qualquer alteração causada pelo binário nas chaves de registro listadas

abaixo representam a intenção de iniciar o *malware* assim que o sistema for inicializado:

```
HKCU\\Software\\Microsoft\\ Windows\\CurrentVersion\\RunOnce
HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\RunServices
HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\RunServicesOnce
HKCU\\CurrentVersion\\Policies\\Explorer\\Run
```

Da mesma forma é possível identificar outras funcionalidades interessantes do código malicioso como, por exemplo, a desativação do sistema de antivírus da máquina. Observando a lista de serviços e processos alterados - obtidos no relatório do *sandbox* é possível determinar se o processo correspondente ao antivírus foi finalizado. Com uma lista dos principais sistemas de antivírus é possível fazer essa relação, como por exemplo, a finalização dos processos “avconsol”, “ave32”, “avgcc32”, “avgctrl”, “avgnt”, “avg-serv”, “avguard”, representa a desativação do antivírus AVG [Preda et al. 2008] . Da mesma forma isto é realizado para outros de antivírus. O software desenvolvido pelos autores para realizar o mapeamento de ações versus funcionalidades foi realizado através da extensão do código implementado pelo laboratório *International Secure Systems*. Com isso, o nosso software de mapeamento conseguiu identificar as seguintes características de um *malware*:

- Auto inicialização do *malware*
- Alteração da configuração do navegador de Internet
- Inserção de complementos maliciosos no Internet Explorer (*Browser helper object*)
- Replicação e cópia do próprio binário no sistema
- Desativação do sistema de antivírus
- Download de arquivos
- Varreduras na rede
- Criação e deleção de arquivos na rede
- Alteração na resolução de nomes
- Desativação da atualização automática do Windows

A automatização do processo de mapeamento, implementada pelo nosso software, trouxe escalabilidade ao sistema. Desta forma cada relatório avaliado pelo *sandbox* foi automaticamente submetido ao nosso sistema de mapeamento de funcionalidades.

d) **Sumarização dos resultados**: Nesta etapa, buscou-se compilar todas as informações obtidas na etapa anterior e agrupá-las segundo sua freqüência de aparição. Desta maneira, foi possível catalogar as principais funcionalidades encontradas no universo de todos os binários coletados pelo nosso sistema de *honeynet*. A apresentação e discussão desses resultados são descritos na Seção 5.

e) **Análise de assinaturas**: A análise de assinaturas é um processo que ocorre de forma paralela com a “análise de binários”, pois utiliza um sistema diferente, ou seja, outra ferramenta. Nesta etapa os binários são enviados para um sistema que busca assinaturas de vírus em 39 soluções de antivírus comerciais ou não. A ferramenta utilizada para isso é o VirusTotal, já descrito na Seção 2. Da mesma forma, a submissão dos binários para avaliação é realizada de forma automatizada através de *scripts* desenvolvidos pelos autores.

f) **Análise dos resultados**: Esta etapa é responsável por processar os resultados da análise realizada na etapa anterior. O processamento dos resultados consiste em identificar se

existe uma assinatura para o *malware*; o tipo de assinatura; e quais antivírus foram eficientes na detecção. Tudo isso realizado de forma automática por *scripts* na linguagem Perl e diretamente submetidos para a próxima etapa.

g) **Base de dados estatísticos**: Esta etapa realiza uma síntese dos resultados obtidos na etapa anterior: ferramentas mais eficientes, assinaturas de vírus mais encontradas, discrepância de respostas, entre outras. Mais informações a respeito da base de dados são descrita na Seção 5 resultados.

Todo o processo descrito acima é realizado para cada amostra de *malware* obtido no sistema de *honeypot*. A automatização de todo o processo por meio de *scripts* e programas possibilitou analisar a grande quantidade de *malwares* que foi coletado durante todo tempo de observação. A próxima seção apresenta os dados resultantes desse processo.

5. Avaliação de resultados

Essa seção apresenta os resultados obtidos pela implementação do processo de análise de binários descritos na Seção 4. Inicialmente serão descritos os resultados relativos à análise de funcionalidades dos binários e posteriormente a dados relacionados às assinaturas em sistemas de antivírus.

Os resultados são expressos com base em arquivos obtidos durante o período de 5 dias em nossa estrutura de coleta. No total foram obtidos 118.678 arquivos binários únicos representando mais de 30 famílias de *malwares* diferentes. As famílias de *malwares* mais observadas pela análise são respectivamente: Allaple,W32.Virut, Trojan.Agent,Trojan Sd-Bot, Trojan.Vanbot, Trojan.Mybot,Worm.Kolab. Nesta grande quantidade de dados foram encontradas muitas variantes de *worms*, em especial do worm Allaple.

O worm Allaple caracteriza-se por implementar técnicas de polimorfismo, o que constantemente altera o conteúdo do seu arquivo para evitar detecções. Devido à grande quantidade desses *worms*, correspondente a 90% dos arquivos obtidos, optou-se por analisar apenas as diferentes variantes de *malwares*. Desta forma, evita-se analisar diferentes arquivos que correspondem a mesma assinatura de *malware*. A triagem, que identificou um arquivo único por assinatura de *malware* reduziu consideravelmente o número de binários a ser analisados. Como resultado do processo de triagem classificou-se 283 arquivos únicos e com assinaturas de *malwares* distintas.

As funcionalidades implementadas pelos *malwares* foram traçadas segundo uma correlação de ações descritas no relatório da fornecido pela ferramenta de análise de binários. Com base no programa implementado pelos autores (vide Seção 4) os seguintes resultados foram extraídos.

A Tabela 1 apresenta a freqüência em que as respectivas funcionalidades foram observadas durante o mapeamento de ações no universo de todos os *malwares* analisados. Como pode ser visto a funcionalidade “atividade no registro do sistema” é a mais implementada. Essa característica corresponde a leitura ou modificação no sistema de registro do Windows. Esse recurso é muito utilizado, pois a partir do registro podem ser configurados vários atributos do sistema como, por exemplo, ocultar a execução do *malware*.

A modificação ou destruição de arquivos, também muito popular, é uma funcionalidade tradicionalmente utilizada pelo *malware* para instalar-se no sistema, onde o mesmo

Tabela 1. Funcionalidades mais utilizadas pelos *malwares*

Funcionalidade	**Quantidade**
Atividade no registro do sistema	242
Modificação ou destruição de arquivos	232
Varredura por endereços	178
Criação de processos	175
Criação de arquivos no diretório system do Windows	167
Capacidade de autostart	164
Alteração de características do navegador Web	43
Conecta a um servidor IRC	23
Alteração no arquivo de resolução de nomes	6
Desativação de antivírus/atualização do Windows	4

modifica arquivos para não ser detectado ou até mesmo para implementar um *backdoor*. A varredura por endereços, por sua vez, representa a busca por outros sistemas vulneráveis e passiveis de serem comprometidos. Essa técnica é muito utilizada para a propagação de *malwares* na rede interna, pois geralmente a comunicação entre máquinas da mesma rede é menos restritiva.

A criação de processos significa que o executável produziu processos durante sua execução como, por exemplo, realizou um *download* de arquivo ou instanciou um programa espião. Já a criação de processos no diretório *system* do Windows é uma artimanha bastante conhecida para armazenar arquivos no alvo, pois tradicionalmente os arquivos localizados nesta pasta não são deletados pelos usuários.

A capacidade de *autostart* retrata a característica que o *malware* possui de auto inicializar assim que o sistema é inicializado. Muito útil para *malwares* como *bots*, no qual a máquina infectada necessita conectar num controlador de botnets para receber instruções do atacante. Outra funcionalidade observada é a alteração de características do navegador Internet Explorer, o que pode afetar seriamente a segurança da navegação na Internet, por exemplo, a inserção de complementos (BHO) para roubar dados sigilosos [Daswani e Stoppelman 2007].

Já a funcionalidade de conectar a um servidor IRC basicamente refere-se a *bots* nos quais utilizam o protocolo IRC (*Internet Relay Chat*) para controlar remotamente a estação comprometida. A característica de “alteração no arquivo de resolução de nomes” é utilizada para ludibriar o acesso a determinados sites, por exemplo, direcionar para um site de bancos falsos (*phishing*). Por fim, a desativação do sistema de antivírus ou o sistema de atualização do Windows Update ainda é pouco observado nos *malwares* coletados, no entanto é uma técnica bastante eficiente e danosa para a segurança do sistema operacional. As diferentes funcionalidades descritas acima, em sua maioria, não são inovadoras, no entanto essa análise demonstrou a preocupação dos autores de *malwares* em ocultar-se no sistema. Como de conhecimento, a ocultação do comprometimento permite que o atacante utilize a estação alvo como uma plataforma para realizar ações ilícitas na rede.

5.1. Antivírus

Da mesma forma que os binários foram submetidos para análise de funcionalidades, também foram avaliados por um sistema de busca por assinaturas, assim como descrito na Seção 4. Durante a avaliação dos resultados observou-se dois pontos em especial: avaliar o arquivo binário logo após a sua coleta e utilizar a base de assinaturas mais recente disponibilizada por cada sistema de antivírus. Tais cuidados permitem verificar a existência da respectiva assinatura do *malware* analisado na última base de assinaturas disponíveis pelo sistema.

Nesse contexto foram avaliados os antivírus mais efetivos e os menos efetivos. Do montante de dados analisados, os sistemas que mais detectaram assinaturas respectivamente foram: Ikarus, Prevx1, AntiVir, BitDefender, CAT-QuickHeal. Em oposição, os sistemas que menos identificaram assinaturas para os arquivos avaliados foram: Sunbelt, Authentium, TheHacker, F-Prot, Fortinet. Essa observação também permite identificar a rapidez dos mantenedores para incorporação de novas assinaturas a base de dados; o que é de fundamental importância para a eficiência dos antivírus. É importante ressaltar que o teste de avaliação realizado acima não representa a efetividade de cada sistema de antivírus como um todo, mas sim perante a categoria particular dos *malwares* coletados por este trabalho.

6. Conclusão e trabalhos futuros

Este trabalho realizou um estudo nos arquivos maliciosos propagados dinamicamente pela rede. Esses arquivos usualmente correspondem à *worms* ou *bots* que estão constantemente varrendo sistemas vulneráveis e fazendo novos comprometimentos. Por serem ataques bastante dinâmicos, tais códigos maliciosos podem apontar novas tendências ou características funcionais. Neste contexto, desenvolveu-se um processo para identificar as funcionalidades mais populares entre esse tipo de *malwares*. Com auxílio de ferramentas como *sandboxes*, sistemas de antivírus aliados, e softwares correlacionadores de ações de *malwares* foi possível realizar um estudo detalhado. Como resultado, este trabalho apresentou as principais ações desempenhadas pelos *malwares* e também quais sistemas de antivírus são mais eficientes para a identificação dessa categoria de *malware* em particular.

Como trabalho futuro, os autores desejam aperfeiçoar o software de mapeamento de funcionalidades buscando aumentar o número de características observadas. Além disso, deseja-se avaliar novas categorias de *malwares* ampliando os sistemas de antivírus utilizados para análise.

7. Agradecimentos

Os autores deste trabalho agradecem especialmente o Ponto de Presença da Rede Nacional de Pesquisa do Rio Grande do Sul (PoP/Rs) [PoP-RS 2009] pelo espaço de endereçamento disponibilizado para a emulação dos serviços vulneráveis. Da mesma forma, os autores agradecem ao laboratório de pesquisa *International Secure Systems Lab* sediado na universidade tecnológica de Viena [IsecLab 2009] pela disponibilização do software base utilizado para correlacionar os eventos dos *malwares*.

Referências

Abu Rajab, M., Zarfoss, J., Monrose, F., e Terzis, A. (2006). A multifaceted approach to understanding the botnet phenomenon. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, páginas 41–52, New York, NY, USA. ACM.

Alkassar, A. and Siekmann, J. H., editors (2008). *Sicherheit 2008: Sicherheit, Schutz und Zuverlässigkeit. Konferenzband der 4. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI), 2.-4. April 2008 im Saarbrücker Schloss*, volume 128 of *LNI*. GI.

Bächer, P., Holz, T., Kötter, M., and Wicherski, G. (2007). Know your enemy: Tracking botnets. http://www.honeynet.org/papers/bots/. The Honeynet Project & Research Alliance.

Barford, P. and Blodgett, M. (2007). Toward botnet mesocosms. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, páginas 6–6, Berkeley, CA, USA. USENIX Association.

Bayer, U., Comparetti, P. M., Hlauschek, C., Krügel, C., and Kirda, E. (2009). Scalable, behavior-based malware clustering. In *NDSS*. The Internet Society.

Bellard, F. (2005). Qemu, a fast and portable dynamic translator. In *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, páginas 41–41, Berkeley, CA, USA. USENIX Association.

Ceron, J. M., Lemes, L. L., Tarouco, L., e Bertholdo, L. M. (2008). Vulnerabilidades em aplicações web: Uma análise baseada nos dados coletados em honeypot. In *VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg 2008)*.

CERT.br (2009). Cartilha de Segurança para Internet. Disponível em: http://www.cert.br/cartilha. Acesso em: Março de 2009.

Daswani, N. e Stoppelman, M. (2007). The anatomy of clickbot.a. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, páginas 11–11, Berkeley, CA, USA. USENIX Association.

IsecLab (2009). Vienna University of Technology - International Secure Systems Lab. Disponível em: http://www.iseclab.org/. Acesso em: agosto de 2009.

Leder, F. e Werner, T. (2009). Know your enemy: Containing conficker. http://www.honeynet.org/papers/. The Honeynet Project & Research Alliance.

Linn, C. e Debray, S. (2003). Obfuscation of executable code to improve resistance to static disassembly. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, páginas 290–299, New York, NY, USA. ACM.

Masud, M. M., Gao, J., Khan, L., Han, J., e Thuraisingham, B. (2008). Peer to peer botnet detection for cyber-security: a data mining approach. In *CSIIRW '08: Proceedings of the 4th annual workshop on Cyber security and information intelligence research*, páginas 1–2, New York, NY, USA. ACM.

Moser, A., Kruegel, C., and Kirda, E. (2007). Exploring multiple execution paths for malware analysis. In *SP '07: Proceedings of the 2007 IEEE Symposium on Security and Privacy*, páginas 231–245, Washington, DC, USA. IEEE Computer Society.

Newsome, J. e Song, D. (2005). Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software. In *Proceedings of the Network and Distributed System Security Symposium (NDSS 2005)*.

Norman (2009). Norman Sandbox Information Center. Disponível em: http://http://www.norman.com. Acesso em: Agosto de 2009.

PoP-RS (2009). Ponto de Presença da Rede Nacional de Pesquisa no estado do Rio Grande do Sul - POP-RS/RNP. Disponível em: http://www.pop-rs.rnp.br. Acesso em: Agosto de 2009.

Preda, M. D., Christodorescu, M., Jha, S., and Debray, S. (2008). A semantics-based approach to malware detection. volume 30, páginas 1–54, New York, NY, USA. ACM.

Raghunathan e Partha (2009). Detecting internet worms using data mining techniques. Disponível em: http://nepenthes.carnivore.it. Acesso em: Agosto de 2009.

Rajab, M., Zarfoss, J., Monrose, F., e Terzis, A. (2007). My botnet is bigger than yours (maybe, better than yours): why size estimates remain challenging. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, páginas 5–5, Berkeley, CA, USA. USENIX Association.

Siddiqui, M., Wang, M. C., e Lee, J. (2008). Detecting trojans using data mining techniques. In *IMTIC*, páginas 400–411.

Stapp, M., Volz, B., e Rekhter, Y. (2006). The Dynamic Host Configuration Protocol (DHCP) Client Fully Qualified Domain Name (FQDN) Option. RFC 4702 (Proposed Standard).

Stinson, E. e Mitchell, J. C. (2008). Towards systematic evaluation of the evadability of bot/botnet detection methods. In *WOOT'08: Proceedings of the 2nd Usenix Workshop on Offensive Technologies*, San Jose, CA, USA. USENIX Association.

Virti, É. S., Ceron, J. M., Tarouco, L. M. R., Granville, L. Z., e Bertholdo, L. M. (2006). Honeypots as a security mechanism . In *IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation (MonAM 2006)*, Tübingen, Germany. IEEE.

VirusTotal (2009). VirusTotal - Free Online Virus and Malware Scan. Disponível em: http://www.virustotal.com. Acesso em: agosto de 2009.

Willems, C., Holz, T., e Freiling, F. (2007). Toward automated dynamic malware analysis using cwsandbox. *IEEE Security and Privacy*, 5(2):32–39.