UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LUCAS THIAGO ASSUMPÇÃO GOUVÊA

TÉCNICAS ULTRALEVES PARA DETECÇÃO DE *MALWARE* BASEADA EM ASSINATURAS PARA REDES DE COMPUTADORES

BAURU
2016

LUCAS THIAGO ASSUMPÇÃO GOUVÊA

TÉCNICAS ULTRALEVES PARA DETECÇÃO DE *MALWARE* BASEADA EM ASSINATURAS PARA REDES DE COMPUTADORES

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

BAURU
2016

LUCAS THIAGO ASSUMPÇÃO GOUVÊA
TÉCNICAS ULTRALEVES PARA DETECÇÃO DE *MALWARE* BASEADA EM ASSINATURAS PARA REDES DE COMPUTADORES/ LUCAS THIAGO ASSUMPÇÃO GOUVÊA. – Bauru, 2016-
37 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

Trabalho de Conclusão de Curso – Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Ciência da Computação, 2016.

1. Tags 2. Para 3. A 4. Ficha 5. Catalográfica

LUCAS THIAGO ASSUMPÇÃO GOUVÊA

TÉCNICAS ULTRALEVES PARA DETECÇÃO DE *MALWARE* BASEADA EM ASSINATURAS PARA REDES DE COMPUTADORES

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. Kelton Augusto Pontara da Costa
Orientador

Simone das Graças Domingues Prado
Convidado 1

João Paulo Papa
Convidado 2

Bauru, _____ de _____ de _____.

Dedico esse texto a todos que valorizam a liberdade e o conhecimento.

Agradecimentos

Agradeço antes de tudo a meus pais, que me deram estrutura, amor, educação, força e qualidade de vida para que eu chegasse até aqui. Agradeço por, num país tão desigual como o nosso, ter a chance e as condições de se formar numa universidade pública de qualidade. Tenho total consciência do quão privilegiado eu sou por ter tudo isso e por ter tudo o que tenho na vida. Agradeço ao Kelton, meu orientador, por acreditar em mim e me auxiliar com sua paciência e o seu conhecimento no desenvolvimento deste trabalho Agradeço também, à música, minha grande paixão, refúgio e meio de expressão, por ter sido crucial na manutenção da minha sanidade e da minha alegria nos momentos mais atormentados que cruzaram o meu caminho. Sem a música, a minha alma e creio que todo este mundo, estariam completamente desolados. Agradeço aos meus alunos por construírem arte e coisas positivas com o conhecimento que compartilhei com eles. Agradeço à minha família pelo apoio e pelo carinho comigo durante toda a minha vida. Agradeço aos amigos que tenho e que me estimam tanto. Em especial a Marcelo Montanha (meu irmão), Fernando Lima Fernandes, Michel Sunsin, Gabriel Naro, Caetano Ranieri, Luís Morais, Luís Paulo Jarussi, Felipe Leme e Victor Frascarelli por compartilharem os palcos comigo durante esses anos e à República Toca, por ter sido uma segunda casa e uma grande escola. Agradeço à Barbara Toscano Barali, pelo amor e pela amizade, por todos os momentos compartilhados e pela ajuda e força inestimáveis que me deu nos meus dias mais tristes. Agradeço a todos que sorriem ao se lembrar de mim e a todos que me fazem sorrir.

Paciência, aplicação, perseverança e, acima de tudo, a vontade inabalável de atingir à meta.
(Beethoven)

Resumo

Desde as origens da Internet, existe uma grande preocupação com a privacidade das comunicações e com a prevenção de danos causados por ataques que se aproveitam de vulnerabilidades nas redes corporativas e públicas. Com o aumento constante do uso de dispositivos conectados à internet e da intensidade e variedade de dados transmitidos entre as pessoas, o número de falhas de segurança em redes ainda é preocupante e tende a continuar subindo. Contudo, os pacotes de alguns programas utilizados nesses ataques, conhecidos como *Malware*, possuem aspectos que são rastreáveis e que, quando incorporados aos métodos de busca dos sistemas de segurança, podem ajudar a detectar esses programas maliciosos antes que eles causem dano a outros pontos da rede infectados. A esses aspectos damos o nome de assinaturas de *Malware* e o objeto de estudo deste trabalho são as técnicas de captura de *Malware* que envolvem assinaturas, novas ferramentas do segmento e as tendências e que estão se apresentando no cenário tecnologico para esse tipo de abordagem. O trabalho propõe também a implementação de um ambiente de aplicação e testagem de assinaturas de *Malware* sobre um conjunto de programas contendo traços de código malicioso, com o auxílio de ferramentas modernas que também são utilizadas por grandes empresas de segurança digital e desenvolvimento, além de um protótipo de textitwebservice para testagem de assinaturas de *malware* e verificação de arquivos *online*.

Palavras-chave: Segurança de Redes, *Malware*, Vírus de Computador.

Abstract

Since the Internet's origins, there's a great preoccupation with the privacy of communications and with preventing damage caused by attacks that exploit vulnerabilities in corporative and private networks. With the increase in usage of internet connected devices and of the intensity and variety of transmitted data between people, the number of security flaws in networks is still worrisome and tends to keep growing. However, packages of some programs used in those attacks, known as malware, have traceable aspects that when incorporated to the search methods of security systems, might help detecting them before causing damage to other infected points ofthe network. To those aspects, we give the name of malware signatures, and the study object of this work are the malware capturing techniques involving signatures, new tools in the segment and the trends showing up in the technologic scenario to this kind of approach. The work also proposes the implementation of an environment to apply and test malware signatures on a set of programs containing traces of malicious code, with the help of modern tools that are also being used by big companies of digital security and development, and also a Web application prototype for online malware signature testing and file verification.

Keywords: Network security. Malware. Computer Viruses.

Lista de ilustrações

Figura 1 – Classificação das técnicas de detecção de *Malware* 15
Figura 2 – Alguns componentes da internet 19
Figura 3 – Exemplo de anomalia pontual 25
Figura 4 – Exemplo de anomalia contextual 25
Figura 5 – Estrutura do projeto 28
Figura 6 – Regra de detecção para o *malware* "Zeus". Imagem elaborada pelo autor . 30
Figura 7 – Regra de detecção do "Superfish" 31

Lista de tabelas

Lista de Quadros

4.1 Famílias de *Malware* e fatores comparativos 24
4.2 Tecnologias recentes de detecção em nível de rede 26
4.3 Tecnologias recentes de detecção em nível de aplicação 27

Sumário

Lista de Quadros **11**

1 INTRODUÇÃO **14**

2 PROBLEMA **16**

3 OBJETIVOS **17**
3.0.1 Objetivo Geral 17
3.0.2 Objetivos Específicos 17
3.0.3 Justificativa 17

4 FUNDAMENTAÇÃO TEÓRICA **18**
4.1 Redes de Computadores e Internet **18**
4.2 Segurança em redes de computadores **20**
4.2.1 Estado atual da área de segurança de redes 20
4.3 Malware **21**
4.3.1 Técnicas de criação 21
4.3.2 Malware baseado em rede 22
4.3.3 Malware Comum 23
4.3.4 Exemplos de Código de Malware 24
4.4 Detecção **24**
4.4.1 Detecção baseada em funcionalidade 25

5 METODOLOGIA **28**
5.1 Métodos e Planejamento **28**
5.2 Materiais Utilizados **29**
5.2.1 Servidor Local 29
5.2.2 Yara 29
5.2.2.1 Regras de detecção do Yara 30
5.2.3 *Logs* de resultados do Yara 31
5.2.4 Python 32
5.2.5 Clam AV 32
5.2.6 Resumo da metodologia do projeto 33

6 DESENVOLVIMENTO **34**
6.1 Obtenção das regras de detecção **34**

7 CRONOGRAMA 35

8 CONCLUSÃO 36

REFERÊNCIAS 37

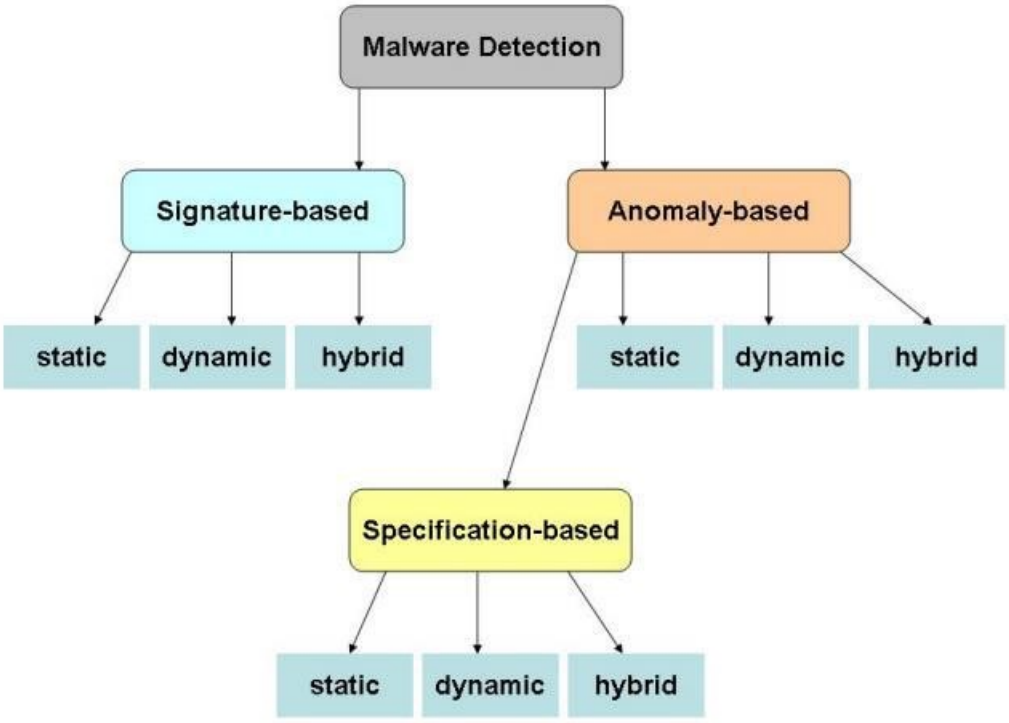
1 INTRODUÇÃO

A presença das redes de computadores no cotidiano vem aumentando de modo muito veloz há anos e apresentou um crescimento em especial com o advento das redes sem fio e da produção massiva e a queda do custo de dispositivos portáteis capazes de se conectarem à internet. De maneira proporcional, aumentaram-se também os riscos relacionados à segurança de redes e à privacidade de informações de diversas espécies que trafegam por entre elas e seus usuários. Numa proposição inicial, este trabalho define como *Malware* todo tipo de programa que seja capaz de alterar o comportamento de um software num dispositivo conectado a uma rede qualquer, feito com o desejo de vender informações de usuários, causar desordem, roubar credenciais, enviar spams, alimentar motores de otimização de busca (SEO) com informações falsas sobre visitas a páginas da Web ou ainda de chantagear usuários, como foi o caso do *malware* japonês denominado *Kenzero*, que publicava históricos de navegação juntamente com a identificação de seus usuários, a menos que os mesmos pagassem 1500 ienes para que seus dados fossem retirados do acesso ao público. SAWLE et al. (2014).

De acordo com IDIKA (2007), pode-se classificar os tipos de métodos de detecção de maneira mais rudimentar em duas categorias: detecção baseada em anomalia e detecção baseada em assinatura. A detecção por anomalia é uma técnica que consiste em treinar um algoritmo de deteção para que ele "saiba" o que constitui o comportamento normal de um programa, para que ele possa decidir sobre a periculosidade dos programas a serem inspecionados. Essa mesma abordagem também dá origem a uma técnica muito similar conhecida como detecção baseada em especificações, onde levantam-se especificações e conjuntos de regras para que a partir daí defina-se o comportamento de um programa normal. A técnica de detecção por assinaturas, por sua vez, envolve a caracterização do que já seria sabidamente malicioso para encontrar traços semelhantes e encontrar os programas anômalos. Todas as técnicas citadas acima podem ser aplicadas com um de três modos diferentes, sendo eles análise estática, análise dinâmica e análise híbrida, onde a análise estática num contexto de detecção por assinaturas, verificaria aspectos estruturais de programas inspecionados, como sequências de *bytes*, e a análise dinâmica buscaria informações de maneira concorrente ao tempo de execução do programa. Em suma, a análise estática procura investigar o programa antes que ele seja executado, e a análise dinâmica atua ou na execução do programa, ou depois que ele já foi executado. As técnicas híbridas são simplesmente combinações dessas duas anteriores, juntando informações estáticas e dinâmicas para que se conclua um parecer sobre a maliciosidade dos códigos analisados.

Dessa maneira, é possível exemplificar as técnicas de deteção com a figura:

Figura 1 – Classificação das técnicas de detecção de *Malware*



Fonte: Elaborada pelo autor.

2 Problema

Os dispositivos móveis atuais ainda não possuem capacidade computacional de sobra para que se façam varreduras constantes que utilizem as técnicas de detecção dinâmicas em seus respectivos sistemas à procura de código malicioso, e com a abrangência da internet se tornando cada vez mais expressiva, o dano em potencial que um malware pode causar a um determinado indivíduo, ou ainda em grupos massivos também torna-se mais preocupante. Sob a ótica de Szewczyk (2012), as redes de computadores, principalmente sem fio, começam a apresentar problemas de vulnerabilidade a partir do momento da compra do *hardware* necessário para implementação da rede e da configuração desse equipamento, onde muitas vezes os vendedores destes produtos tentam passar a ideia de que os equipamentos por si são capazes de blindar uma rede contra qualquer intrusão, a fim de obter sucesso comercial valendo-se da ingenuidade de usuários leigos no assunto de segurança de redes. As técnicas de detecção por assinatura, apesar de não serem as mais eficazes à disposição, são muito mais simples de se implementar e resolverem o problema da infecção por malwares conhecidos ou ainda malwares novos com características semelhantes às dos conhecidos.

3 OBJETIVOS

3.0.1 Objetivo Geral

Analisar o resultado da ação de diferentes técnicas de detecção de *malware* baseadas em assinatura num ambiente de testes para obter métricas relevantes quanto à eficiência nas varreduras realizadas, número de falsos positivos encontrados e se possível determinar qual delas é a mais apropriada para uso em ambientes mobile e de redes de computadores em geral.

3.0.2 Objetivos Específicos

- Para embasar o projeto, será levantado um histórico sobre segurança em redes de computadores e assuntos correlacionados principalmente às técnicas de detecção a serem estudadas.

- Montar um ambiente para que ele seja populado com conjuntos de programas a serem investigados e executar diferentes algoritmos para que posteriormente classifiquem-se seus resultados. Por questões de relevância, o escopo do trabalho envolverá sistemas *Windows* e *Android* pois a maioria dos usuários comuns utiliza estes sistemas operacionais.

- Comparar todos os métodos previamente escolhidos e testados para apresentar suas respectivas características e desempenhos.

3.0.3 Justificativa

O desenvolvimento de novas técnicas de segurança computacional, bem como a análise e aprimoramento de técnicas já construídas anteriormente, representam sempre um avanço na área de segurança de redes, pois todo tipo de sistema operacional possui alguma deficiência no quesito de segurança, haja visto que mesmo agências de inteligência como a NSA enfrentam de tempos em tempos problemas envolvendo vazamentos de dados e intrusões. De acordo com Hassan e Muhammad (2010), os tipos de ataque mais poderosos e bem sucedidos ocorrem de dentro da rede, pois em diversas situações, o administrador de rede confia em todos os usuários internos e não suspeita de ataques vindos de seu próprio lado. Por isso, é sempre um investimento desenvolver e manter uma política de aprimorar as defesas das redes de computadores. O contexto do projeto é o da esfera do usuário comum, que utiliza dispositivos comuns mas também é alvo de ataques que às vezes são menores e até inofensivos, que contudo, ocorrem com frequência.

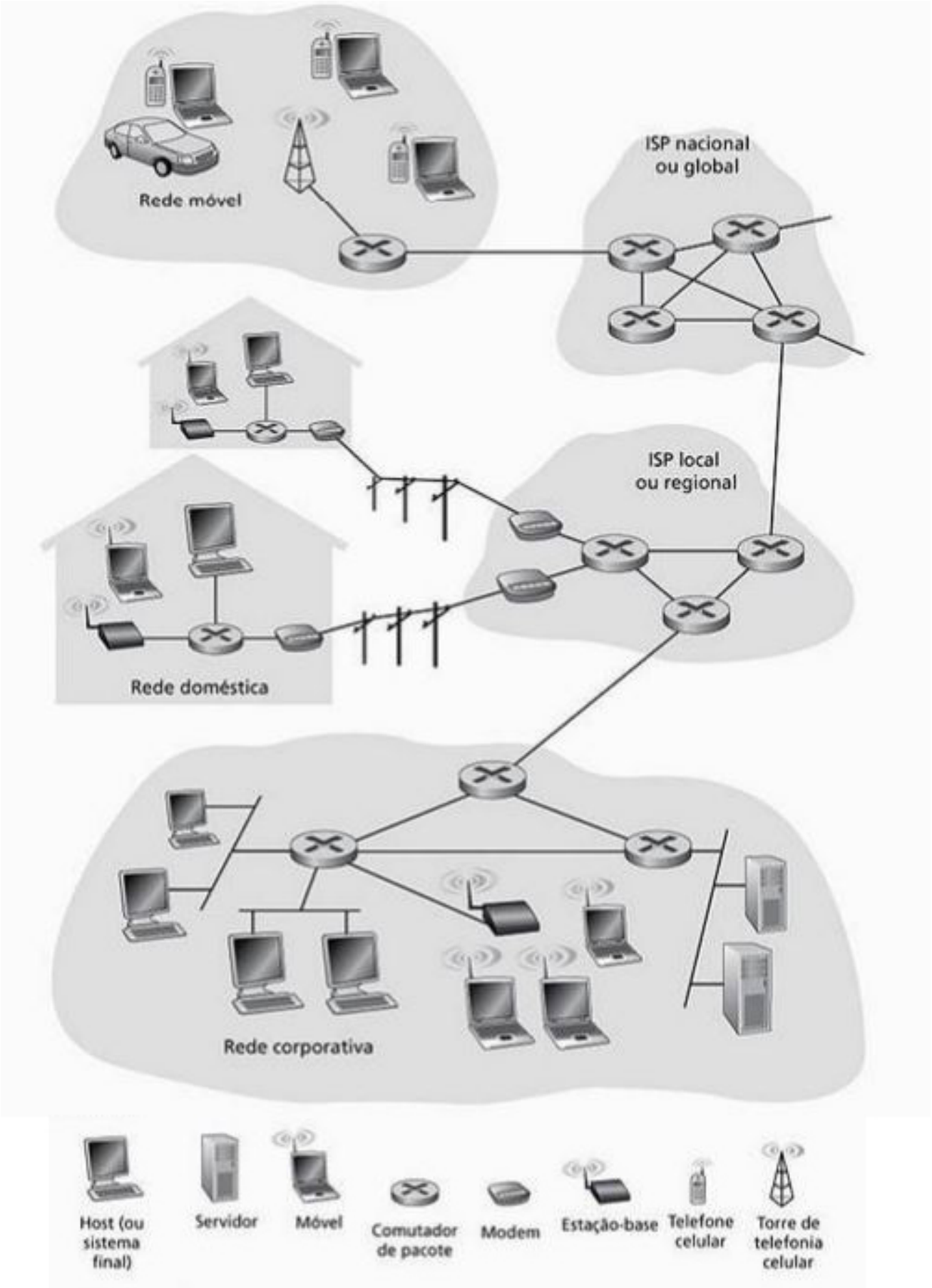
4 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo é apresentado o plano de fundo teórico que envolve os tópicos pertinentes ao trabalho, destacando as definições e conceitos essenciais dos seguintes assuntos: Redes de Computadores, Internet, Segurança em redes de Computadores (estado da arte), *Malware* e suas respectivas técnicas de detecção.

4.1 Redes de Computadores e Internet

Segundo o trabalho de KUROSE (2009), pode-se definir a Internet como a rede de computadores que interconecta milhões de dispositivos computacionais ao redor do mundo. Há pouco tempo a gama de dispositivos conectados à internet era mais baixa e se resumia a servidores e computadores de mesa. Hoje pode-se ver que cada vez mais dispositivos não tradicionais estão sendo ligados à internet. Assim, defini-se como sistemas finais, ou hospedeiros, quaisquer dispositivos que possuam tal capacidade de conexão. Em julho de 2008 estimavam-se 600 milhões de sistemas finais ligados à internet sem contar dispositivos que são conectados de maneira intermitente (como laptops e smartphones). Quando sistemas finais se comunicam, trocam arquivos segmentados, que denominam-se como pacotes, por meio de enlaces de comunicação (links), que podem ser físicos ou por ondas, no caso de redes sem fio e via rádio. Os dispositivos periféricos que são responsáveis por essa comunicação são chamados comutadores de pacotes, vistos comumente como roteadores e switchs. O que torna essa comunicação válida entre tantos dispositivos diversos são os protocolos de rede, como TCP/IP. Tais protocolos apenas definem a estrutura dos pacotes que serão trocados e a classificação dos dados dentro destas estruturas. Dessa maneira, os comutadores de pacotes conseguem interagir sem problemas e dividir as cargas da rede com eficiência. Neste trabalho não se faz necessária uma abordagem mais profunda sobre estruturas de pacotes e de componentes mais específicos da rede, haja visto que o foco do trabalho é voltado para segurança em redes e uma revisão completa sobre fundamentos básicos de redes não elucida conteúdos dessa vertente. Os pacotes ainda serão citados no decorrer do trabalho e terão seus conceitos devidamente esclarecidos na seção apropriada. A ideia geral de redes de computadores suficiente para esse trabalho pode ser ilustrada com a seguinte imagem:

Figura 2 – Alguns componentes da internet



Fonte: Kurose (2009)

4.2 Segurança em redes de computadores

Ainda no material de Kurose pode-se definir vários conceitos em segurança de redes. A ideia em si vem de imaginar que as pessoas desejam se falar através da internet, por exemplo, e que essas pessoas não queiram que ninguém mais seja capaz de interceptar esses dados mesmo que elas estejam conectadas em locais considerados inseguros, que deixam computadores serem publicamente utilizados. Além disso, imaginar que as pessoas também querem garantir que as mensagens trocadas não sejam originadas de um intruso, ou passem por um intruso na rede. Com essa ideia pode-se elencar as propriedades necessárias para uma comunicação considerada *segura*. São elas:

- Confidencialidade: A garantia de que somente respectivos remetentes e destinatários sejam capazes de ver e entender o conteúdo das mensagens transmitidas.
- Autenticação do ponto final: Remetentes e destinatários precisam ter meios de confirmar a identidade uns dos outros durante uma comunicação.
- Integridade da mensagem: Além de identificarem uns aos outros, os envolvidos na comunicação querem assegurar que o conteúdo que enviem seja exatamente o conteúdo que as outras partes recebam. As mensagens podem se corromper durante uma comunicação tanto por acidentes, como por má intenção de intrusos na rede.
- Segurança Operacional: Qualquer empresa atualmente vai ter uma rede local conectada à rede pública. Os mecanismos responsáveis por filtrar o que a rede local vai receber da rede pública são chamados mecanismos de segurança operacional, como firewalls, que são servidores que se posicionam entre as duas redes e intermedeiam a comunicação entre elas, e também como sistemas de detecção de intrusão, foco maior do trabalho, que são sistemas que analisam os pacotes de maneira mais profunda e os identificam para separar e tratar pacotes potencialmente suspeitos.

Qualquer ação de um intruso na rede configura uma falha na presença de um ou mais dos quesitos elencados acima; quando ocorre tal brecha e um usuário estranho consegue acesso não autorizado a uma rede, ele ganha muitas possibilidades de uso pra essa rede. Um invasor pode agir de modo passivo apenas monitorando as comunicações que pode ver, ou então agir com um caráter mais nocivo e alterar os conteúdos que acessa, seja corrompendo, alterando ou removendo arquivos, senhas e dados diversos. Os sistemas de detecção de intrusão buscam maneiras de identificar acessos ilegais e tráfego de dados estranhos na rede.

4.2.1 Estado atual da área de segurança de redes

De acordo com Hurd (2015) uma área bastante promissora é a de *Cyber Higiene*, e seus aspectos incluem gerenciamento e proteção de credenciais, auditoria de roubo de credenciais e

dados analíticos sobre comportamento de acesso do usuário, sendo esta última subárea a que demonstra a maior parcela de progresso recente e num futuro de médio prazo. Nesse segmento, o conceito de usuário pode ser especificado além do usuário comum, classificando novos perfis como atacantes e autores (programadores). O comportamento envolve características como duração do tempo de acesso, preferência de dispositivo utilizado, utilização de navegador e outras diversas informações sobre as atividades desenvolvidas por um usuário e quais seriam seus fins. Utilizando a internet como "observatório" , a análise do comportamento de usuários tem sofrido uma revolução em virtude do crescimento da internet das coisas, onde cada vez mais objetos estão conectados à rede, com grande parte destes objetos sempre reportando informações, como sensores e câmeras de trânsito.

4.3 Malware

O conceito de *Malware* já foi abordado de maneira mais rudimentar na introdução do trabalho. Porém, como o viés do projeto caminha muito próximo de classificações mais profundas desse tópico, faz-se necessário recorrer a uma bibliografia mais completa sobre os processos relacionados ao *Malware*. Na pesquisa desenvolvida por SAHEED(2013), pode-se aproveitar muitas informações sobre o patamar do desenvolvimento deste tipo de *software*. De acordo com pesquisas levantadas pela McAfee, esses programas continuam a se tornar mais numerosos dia-a-dia, sendo encontrados novos "indivíduos" aos milhares. Em contrapartida, pesquisadores e laboratórios especializados em segurança aprimoram novos métodos para construir *anti-malware* mais eficiente e mais autônomo. A seguir, diferenciam-se os tipos de *Malware* com base nas técnicas de criação dos mesmos, separando-os por técnicas de ofuscamento, métodos de invocação, plataforma de atividade, abrangência e técnicas de propagação. Nem todo *software* malicioso é executado de dentro de máquinas infectadas; atualmente a maioria dos ataques são originários da internet, pois as vulnerabilidades encontradas são mais proeminentes do que em ambientes de rede local, em virtude de a internet ser um espaço, de modo geral, público, beneficiando atacantes que tenham meios de acessar um sistema alvo remotamente. Um sistema de detecção de *malware* tem majoritariamente duas tarefas: busca e análise, ou seja, saber encontrar material não confiável e saber identificar com precisão que o material em questão realmente possui traços nocivos de código, ou faz com que algum outro programa apresente comportamento considerado anormal.

4.3.1 Técnicas de criação

Para criar esse tipo de código, atacantes utilizam de vários meios desde a inserção de pequenos pedaços de código em outros programas, até a algoritmos de alta complexidade, feitos com o intuito de esconder o *malware* ou torná-lo *polimórfico*. Um *malware* que se classifique como polimórfico consiste em um programa capaz de trocar pedaços do código sem mudar a semântica e o resultado da execução a cada infecção, para driblar verificações de

sintaxe, geralmente com a ajuda de técnicas avançadas de encriptação. *Malwares* polimórficos são o grande ponto fraco das técnicas de detecção por assinatura, pois eles tornam inviável a geração de assinaturas únicas para classificação do código malicioso. Pode-se dizer que um *Malware* é *ofuscado* quando ele é polimórfico ou *metamórfico*, onde o código original é transformado em algo funcionalmente equivalente, porém muito mais difícil de se compreender quando lido. As técnicas utilizadas para que isso ocorra são a utilização de código morto, que é uma grande parcela de código que não faz absolutamente nada, acoplado ao código malicioso para mascará-lo, ou também a transportação de código, que consiste em realizar um grande número de "jumps" no programa, porém sem alteração no fluxo de controle do mesmo.

4.3.2 Malware baseado em rede

Para dar início à classificação dos tipos de *Malware*, primeiro observa-se com destaque aqueles que se utilizam da rede como meio de propagação e de atuação. Nesse segmento, é possível identificar o que são os *Adwares*, *Spywares*, *Cookies*, *Backdoors*, *Trojans*, *Sniffers*, *Spam* e *Botnet*.

Spyware é um tipo de *malware* que se instala de maneira secreta num computador, com o objetivo de coletar informações sobre o uso de determinados programas ou sites visitados por um indivíduo sem o conhecimento do mesmo. Curiosamente, até mesmo grandes empresas como Google e Microsoft se utilizam desse tipo de *Malware* para coletar informações de seus clientes.

Adwares são feitos com a finalidade de servirem como atalhos publicitários para outros programas que têm seu desenvolvimento mantido com fundos oriundos de propaganda na internet. Por natureza, não são essencialmente danosos a sistemas e no máximo causam transtorno às pessoas por apresentarem anúncios indesejados, entretanto, existem algumas variações nessa família de *malware* que possuem *spyware* acoplados juntamente com as propagandas, invadindo a privacidade dos usuários.

Cookies, na verdade, são informações de usuário guardadas pelos navegadores de internet; eles servem primeiramente como meio de autenticação e armazenamento de configurações entre cliente e servidor. Um cookie é um pequeno arquivo de texto com essas informações, e na maioria das vezes vai ter um prazo de validade fixado pelo servidor que o usuário acessa. Também não são danosos, mas os *cookies* são usados por *spywares* e também no roubo de credenciais de usuários.

Backdoors ou *Trapdoors* são código malicioso atrelado a aplicações ou sistemas operacionais que concedem a alguém acesso a um sistema sem que seja necessário passar por métodos comuns de autenticação. Eventualmente esses programas também podem ser usados como meio para conseguir acesso remoto a sistemas.

Um *Trojan Horse* (cavalo de tróia), ou apenas *Trojan*, descreve um programa que

aparenta ser útil ou inofensivo, mas que ao ser executado pode corromper dados ou roubar informações. Um *Trojan* é desenvolvido para que o usuário, além de não suspeitar dele, seja levado a executá-lo por curiosidade, então é muito comum ver esse tipo de programa escondido em arquivos multimídia que vão seduzir algum nicho de usuários.

Sniffers são capazes de monitorar e armezenar dados a respeito do tráfego de uma rede. Eles capturam os pacotes que estão sendo trocados e os decodificam para ganhar acesso aos dados crus dos campos dos pacotes. *Sniffers* podem ser usados como um passo inicial para uma invasão, para que posteriormente sejam obtidas senhas e outras informações confidenciais que sejam alvo de um atacante.

O *Spam* talvez seja o tipo de *malware* mais conhecido por usuários mais leigos; basicamente é um pacote de *software* com a funcionalidade de transmitir mensagens em larga escala via *e-mail*. O impacto negativo do *spam* é a possibilidade de lentidão causada pelo alto número de requisições que será realizado por ele para os servidores, atrapalhando o recebimento de mensagens mais relevantes e poluindo as caixas de *e-mail*. Embora seja uma prática muito mal vista na internet, nos Estados Unidos o *spam* é legalizado, graças ao ato CAN-SPAM, desde 2003.

Botnet não é na verdade um *malware* de código, mas uma coleção de computadores infectados e que são controlados remotamente por um atacante que pode usá-los para realizar atos maldosos sem que seja preciso se autenticar nessas máquinas. Esse é o conceito por trás dos ataques de negação de serviço (DoS), onde um grande número de máquinas controladas sobrecarrega um determinado serviço na rede com um número absurdo de requisições aos servidores.(SAHEED, 2013)

4.3.3 Malware Comum

O *Malware* comum, em contraste com o de rede, é feito para ser executado em ambiente local sem a necessidade de qualquer conexão a rede, pois também pode ser propagado via dispositivos físicos como pendrives, HDs externos e periféricos com *software* malicioso embarcado. Nessa categoria, classificam-se os vírus, *worms* e bombas lógicas.

Vírus de computador é um conceito que muitas vezes acaba generalizando o *Malware*. É muito comum um usuário que foi infectado apenas com *adware* dizer que "está com vírus no PC" e assim segue; Na verdade, um vírus de computador é um código capaz de se replicar durante a infecção em qualquer programa ou documento. Por exemplo, sistemas Windows em muitas mídias se utilizam de um arquivo "autorun.inf" para automatizar a execução de tarefas para montagem de pastas ou imagens de disco. Logo, um código maldoso que possa se acoplar silenciosamente a esse tipo de arquivo, com certeza conseguirá infectar um sistema Windows.

Os Worms são códigos capazes de se replicar em múltiplas máquinas de modo independente, isto é, sem precisar se atrelarem a um determinado programa para iniciar seu ciclo de

vida. Eles podem, entre outras coisas, consumir a banda da rede e privar o usuário infectado de utilizá-la, além de criarem várias cópias de si para aumentar a taxa de programação. Esta última característica é o que os *softwares* de anti-vírus utilizam para identificar Worms; se um arquivo está se repetindo muitas vezes com atributos iguais, ele é considerado suspeito nesses sistemas de detecção.

A bomba lógica é um *software* que permanece quieto até que se atinja uma determinada condição. Para isso, o programa fica apenas checando a data do sistema até que seja a data especificada e só então ele vai, de fato, executar seu código.(SAHEED, 2013)

Dadas essas classificações, pode-se ilustrar os tipos de *malware* e suas características com o seguinte quadro:

Quadro 4.1: Famílias de *Malware* e fatores comparativos

Fatores de comparação	Família do Malware	Spyware	Adware	Cookies	Trapdoor	Trojan	Sniffers	Spam	Botnet	Logic bomb	Worm	Virus
Técnicas de Criação	Padrão	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Ofuscado	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Polimórfico	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Kit de Ferramentas(toolkit)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Ambiente de execução	Rede	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	×
	Execução remota pela web	✓	✓	✓	✓	✓	✓	✓	✓	×	×	×
	PC	×	×	×	×	×	×	×	×	✓	✓	✓
Mídia de propagação	Rede	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Discos removíveis	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Downloads	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Impactos negativos	Brecha de confidencialidade	✓	×	✓	×	✓	✓	×	×	×	×	×
	Incômodo aos usuários	×	✓	×	×	×	×	✓	×	×	×	×
	Negação de serviços	×	×	×	✓	×	×	✓	✓	✓	✓	✓
	Corrupção de dados	×	×	×	✓	×	×	✓	✓	✓	×	✓

Fonte: Saeed (2013) (traduzida pelo autor).

4.3.4 Exemplos de Código de Malware

Alguns *malwares* acabam tendo seu código aberto e disponibilizado para estudos, ora por vontade de seus próprios autores, ou por vazamento de informações feito por outros *Hackers* interessados em tornar público esse tipo de informação. Nesta seção encontram-se algumas explicações sobre a arquitetura interna dos *malwares* e trechos de código-fonte obtidos a partir de pesquisas em repositórios públicos da *web*

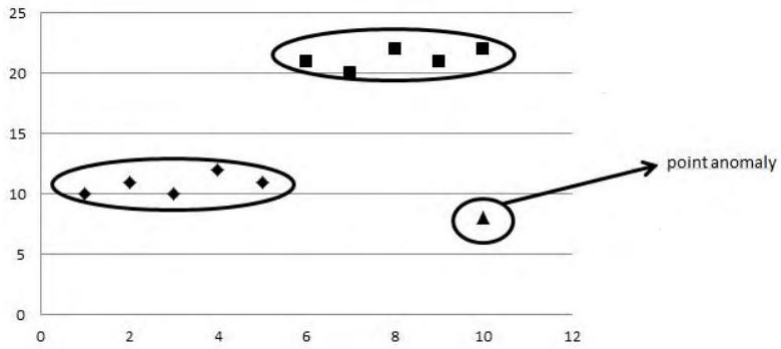
- Citar os repositórios - Imagens do código dos malwares

4.4 Detecção

Nesta seção, é exposto o estado atual das tecnologias de detecção e para onde elas estão caminhando no futuro. O trabalho de Baddar et al. (2014) nos mostra uma perspectiva bastante atual do cenário e de como funcionam, de modo geral, os processos de detecção das anomalias.

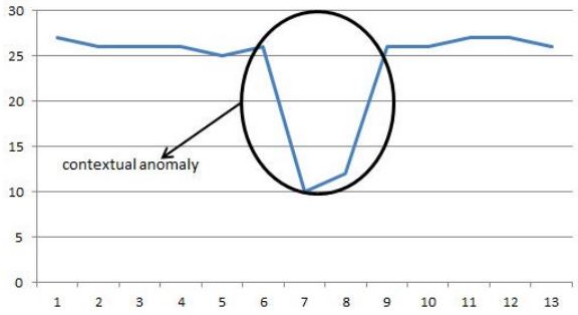
Um sistema de detecção pode agir em ambiente de rede, verificando as características do tráfego da rede, ou em nível de aplicação, onde vai varrer os dados do disco a procura de assinaturas conhecidas de anomalia ou de comportamentos fora do normal. A detecção por comportamento anômalo precisa ser capaz de filtrar casos anômalos isolados, denominados como anomalias pontuais, de casos com suspeita mais palpável de anomalia, que pode-se chamar de anomalia de contexto. Uma anomalia pontual pode ser exemplificada do seguinte modo: uma tentativa de acesso de um usuário num sistema corporativo não é, a princípio, um ato que possa ser classificado como anômalo; porém, se essa mesma tentativa se repete várias vezes em horários não condizentes com o expediente comercial, pode haver alguém suspeito tentando roubar as credenciais desse usuário. As figuras abaixo exemplificam as ideias de anomalia pontual e contextual:

Figura 3 – Exemplo de anomalia pontual



Fonte: Baddar. (2014)

Figura 4 – Exemplo de anomalia contextual



Fonte: Baddar. (2014)

4.4.1 Detecção baseada em funcionalidade

Destaca-se esse segmento em especial pois é daqui que se origina a ideia de detecção baseada em assinaturas e em comportamentos. A detecção baseada em assinaturas compara casos suspeitos com uma relação já conhecida de código comprovadamente nocivo, que fica

armazenada geralmente num banco de dados da empresa que desenvolve o sistema de detecção. Motores de busca de anti-vírus são o exemplo mais clássico de detecção baseada em assinaturas. Numa outra abordagem, a detecção por comportamento vai tentar "aprender" como vive um programa normal dentro do sistema, e como é o ciclo de um programa suspeito, geralmente recorrendo a dados de tempo de execução dos programas que estão sendo processados e classificando essas características com o auxílio de técnicas como aprendizado de máquina e redes neurais (BADDAR, 2014). A maior parte do conjunto recente das aplicações de detecção em nível de rede tem adotado a abordagem do aprendizado de máquinas para detectar *software* por comportamento, quando em nível de rede. Em nível de aplicação, a grande maioria também procura traçar perfis de comportamento utilizando técnicas de detecção em modo dinâmico como pode-se ver nos quadros abaixo:

Quadro 4.2: Tecnologias recentes de detecção em nível de rede

1st Author, Year and Ref.	Approach	NC vs. DC	Scalable?	on/off-line
AbuAitah, 2014 [71]	Machine learning	DC	No	Online
Barani, 2014 [73]	Machine learning	NC	Yes	Online
Salem, 2014 [74]	Machine learning	DC	No	Online
Fiore, 2013 [29]	Machine learning	NC	No	Offline
Guo, 2013 [79]	Machine learning	NC	No	Online
Louvieris, 2013 [81]	Machine learning	NC	No	Offline
Lin, 2012 [83]	Machine learning	NC	No	Offline
Hayes, 2014 [84]	Statistical	DC	Yes	Online
Sun, 2014 [86]	Statistical	NC	No	Offline
Xie, 2014 [90]	Statistical	DC	Yes	Online
Bayraktar, 2014 [91]	Statistical	DC	No	Online
Soltanmohammadi, 2013 [93]	Statistical	DC	No	Online
Limthong, 2013 [95]	Mixed	NC	No	Online
Wang, 2013 [97]	Mixed	NC	No	Offline
Egilmez, 2014 [99]	Spectral	DC	Yes	Online
Cohen, 2014 [101]	Inf. Theoretic	DC	No	Online
Cuadra-Sanchez, 2014 [54]	Inf. Theoretic	NC	No	Offline
Huang, 2014 [102]	Streaming	NC	Yes	Online
Liu, 2012 [104]	Streaming	NC	No	Online

Fonte: Baddar. (2014)

Quadro 4.3: Tecnologias recentes de detecção em nível de aplicação

1st Author, Year and Ref.	Stat./Dyn.	on/off-device	Appr.	Scal.?	on/off-line	Sign./Behav.
Damopoulos, 2014 [114]	Dynamic	Mixed	ML	Yes	Online	Behavioral
Shabtai, 2014 [117]	Dynamic	On-device	ML	Yes	Online	Behavioral
Amos, 2013 [120]	Dynamic	Off-device	ML	Yes	Online	Behavioral
Ham, 2013 [35]	Dynamic	Off-device	ML	No	Offline	Behavioral
Wang, 2013 [121]	Dynamic	Off-device	Stat.	Yes	Online	Behavioral
Mas'ud, 2014 [36]	Dynamic	Off-device	ML	No	Offline	Behavioral
Alam, 2013 [123]	Dynamic	Off-device	ML	No	Offline	Behavioral
Wei, 2013 [124]	Dynamic	Off-device	ML	No	Mixed	Behavioral
Shabtai, 2012 [37]	Dynamic	Off-device	ML	No	Online	Behavioral
Yan, 2012 [130]	Dynamic	Off-device	–	No	Offline	Signature
Zheng, 2014 [129]	Dynamic	On-device	–	Yes	Online	Signature
Zhou, 2012 [131]	Dynamic	Off-device	Algo.	No	Online	Mixed
Arp, 2014 [105]	Static	On-device	ML	Yes	Online	Behavioral
Liang, 2014 [109]	Static	Off-device	ML	No	Offline	Behavioral
Cen, 2014 [34]	Static	Off-device	Stat.	No	Offline	Behavioral
Glodek, 2013 [111]	Static	Off-device	ML	No	Offline	Behavioral
Yerima, 2013 [112]	Static	Off-device	ML	No	Offline	Behavioral
Sahs, 2012 [113]	Static	Off-device	ML	No	Offline	Behavioral

Fonte: Baddar. (2014)

5 METODOLOGIA

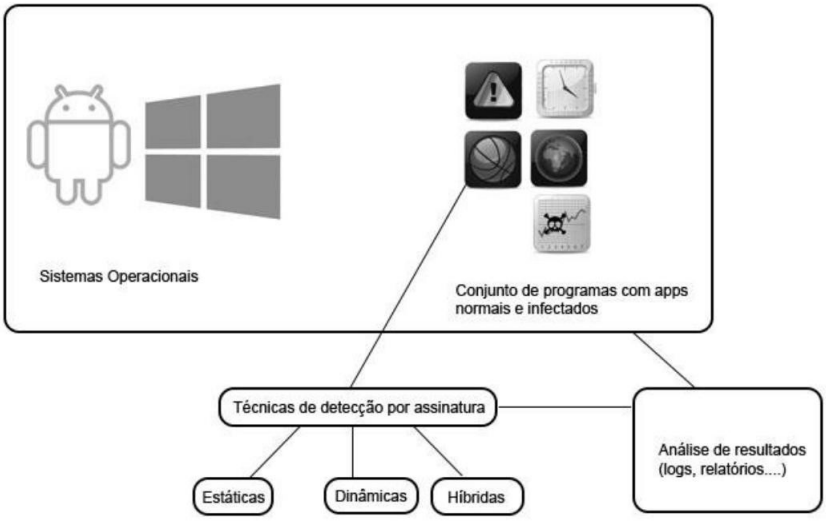
5.1 Métodos e Planejamento

Planeja-se dividir o estudo de acordo com as seguintes etapas:

- Levantamento bibliográfico
- Análise e desenvolvimento das técnicas de detecção baseadas em assinatura utilizadas atualmente
- Implementação Computacional
- Análise dos resultados e conclusão da monografia.

A confecção da monografia vai ocorrer juntamente com o andamento das demais etapas definidas durante toda a extensão do projeto. A ideia é utilizar máquinas virtuais e emuladores para implementar em caráter de teste os algoritmos de detecção baseados em assinatura sobre um conjunto de programas contendo alguns programas já infectados com amostras de *malware* diversas para observar, em termos gerais, a eficiência do tempo de execução dos algoritmos e suas respectivas taxas de sucesso. O ambiente pode ser montado em qualquer máquina que suporte a plataforma Windows e emulador Android, então não há necessidade de que se reserve material da UNESP para uso exclusivo do projeto. A figura a seguir ilustra o planejamento do projeto.

Figura 5 – Estrutura do projeto



Fonte: Elaborada pelo autor.(2016)

5.2 Materiais Utilizados

5.2.1 Servidor Local

Foi utilizada uma máquina com 4GB de RAM, 1TB de armazenamento e sistema operacional Windows 10 de arquitetura 64 bits.

5.2.2 Yara

Yara é uma ferramenta de código aberto recentemente desenvolvida, cujo intuito é auxiliar desenvolvedores a identificar e classificar amostras de *malware*. Ele é construído em C, porém, com o auxílio de algumas bibliotecas, há a possibilidade de incorporá-lo em scripts de outras linguagens, como Python.Segundo ALVAREZ (2014), O funcionamento do Yara consiste em comparar regras criadas pelo desenvolvedor que descrevem famílias de *malware*, com o código de executáveis quaisquer, ou também de arquivos comprimidos em alguns formatos comuns. As regras podem variar bastante em nível de complexidade e mapearem diversas características do comportamento dos executáveis, que serão detalhadas posteriormente. O projeto no momento encontra-se bastante popular e o programa está sendo utilizado como ferramenta de construção de assinatura de *malware* por desenvolvedores de diversos grupos importantes no ramo, como:

- ActiveCanopy
- Adlice
- CrowdStrike FMS
- Fox-IT
- Heroku
- Kaspersky Lab
- Picus Security
- ReversingLabs
- Symantec
- ThreatStream, Inc.
- Trend Micro
- VirusTotal Intelligence
- We Watch Your Website

O Yara será utilizado no presente trabalho de conclusão de curso como meio de aplicação dos algoritmos de assinatura nos programas que serão examinados. Com o auxílio de scripts desenvolvidos em Python, simularemos uma varredura nesse conjunto de programas aplicando regras obtidas através de sites como o Yara Rules e o Sane Security, que disponibilizam sem custo algum assinaturas de milhões de códigos maliciosos já capturados pela web. Com esse grande arsenal de assinaturas de *malware*, é possível testarmos com certa completude as técnicas e assinaturas que são desenvolvidas no mercado e utilizadas nos sistemas de detecção de intrusão.

5.2.2.1 Regras de detecção do Yara

O Yara funciona interpretando arquivos que contém características que possivelmente apontam para o comportamento dos elementos da família do *malware* que deve ser isolado. As regras são escritas com uma sintaxe simples que remete à construção de uma estrutura de dados em C, contendo strings a serem comparadas tanto em formato texto quanto hexadecimal, conjuntos de strings, endereços da memória que ficarão sob monitoramento, limitações de tamanho de arquivo, expressões regulares, pontos de entrada de execução e variáveis externas. Todos esses dados do arquivo examinado podem ativar condições, listadas na construção dessas regras, que vão determinar se o arquivo corresponde ou não à família definida nas regras de detecção. Para exemplificar mais claramente a construção e o funcionamento das regras, seguem alguns exemplos de código obtido do repositório do projeto Yara Rules no Github.

```
import "pe"

rule Windows_Malware : Zeus_1134
    {
            meta:
                    author = "Xylitol xylitol@malwareint.com"
                    date = "2014-03-03"
                    description = "Match first two bytes, protocol and string present in Zeus 1.1.3.4"
                    reference = "http://www.xylibox.com/2014/03/zeus-1134.html"

            strings:
                    $mz = {4D 5A}
                    $protocol1 = "X_ID: "
                    $protocol2 = "X_OS: "
                    $protocol3 = "X_BV: "
                    $stringR1 = "InitializeSecurityDescriptor"
                    $stringR2 = "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; SV1)"
            condition:
                    ($mz at 0 and all of ($protocol*) and ($stringR1 or $stringR2))
    }
```

Figura 6 – Regra de detecção para o *malware* "Zeus". Imagem elaborada pelo autor

Na imagem acima temos exemplo de uma descrição para o *malware* chamado "Zeus" (também conhecido como "RPG", "ZBot", "Infostealer", entre outros), que é membro da família dos *Trojans*. Segundo informações da Wiki Security2016 os danos causados por ele incluem roubo de informações pessoais como senhas de email e de bancos, para posteriormente

efetuarem-se transferências para "contas-mula" que lavariam o dinheiro das vítimas até que ele chegasse aos atacantes. O *malware* foi escrito majoritariamente em C com alguns pequenos trechos em php, e seu código fonte, vazado em 2011, pode ser encontrado neste outro repositório público do Github. Como podemos observar, a regra escrita para marcar o trojan verifica os dois primeiros *bytes*, *strings* e o protocolo utilizados por uma determinada versão do "Zeus" e fica ativa quando a primeira e uma das outras duas condições são satisfeitas.

Outro exemplo de regra de detecção envolvendo parâmetros semelhantes aos anteriores é a que descreve o *malware* "Lenovo Superfish"2016; na verdade esse é o nome de um serviço que vinha embutido em algumas séries de equipamentos lançados pela fabricante Lenovo entre 2014 e 2015, cuja meta era ajudar consumidores a encontrar produtos similares aos que eles buscavam pela web. Esse serviço possuía uma vulnerabilidade de protocolos que foi explorada na criação do *malware* homônimo. No caso, as características que estão sendo mapeadas são novamente os dois primeiros *bytes* do programa, porém com strings em formato e encodificações diferentes das encontradas na regra anterior, como segue na figura:

```
rule VisualDiscovery_Lonovo_Superfish_SSL_Hijack {
    meta:
        description = "Lenovo Superfish SSL Interceptor - file VisualDiscovery.exe"
        author = "Florian Roth / improved by kbandla"
        reference = "https://twitter.com/4nc4p/status/568325493558272000"
        date = "2015/02/19"
        hash1 = "99af9cfc7ab47f847103b5497b746407dc566963"
        hash2 = "f0b0cd0227ba302ac9ab4f30d837422c7ae66c46"
        hash3 = "f12edf2598d8f0732009c5cd1df5d2c559455a0b"
        hash4 = "343af97d47582c8150d63cbced601113b14fcca6"
    strings:
        $mz = { 4d 5a }
        //$s1 = "VisualDiscovery.exe" fullword wide
        $s2 = "Invalid key length used to initialize BlowFish." fullword ascii
        $s3 = "GetPCProxyHandler" fullword ascii
        $s4 = "StartPCProxy" fullword ascii
        $s5 = "SetPCProxyHandler" fullword ascii
    condition:
        ( $mz at 0 ) and filesize < 2MB and all of ($s*)
}
```

Figura 7 – Regra de detecção do "Superfish"

As regras que utilizaremos no decorrer do projeto serão unificadas num arquivo de índice, compiladas via um*script* em Python para a interpretação do Yara e combinadas com o conjunto completo de arquivos de uma só vez.

5.2.3 *Logs* de resultados do Yara

As varreduras efetuadas pelo Yara podem ter seus resultados gravados em arquivos de texto para análise posterior, que podem também definirem-se como *logs*, que por sua vez retornam um dicionário de strings com o seguinte formato:

Algoritmo 5.1 – Conteúdo dos arquivos de resultado de varredura

```
{
'tags': ['mal', 'ware'],
'matches': True,
'namespace': 'default',
'rule': 'regras_compiladas',
'meta': {},
'strings': [(81L, '$a', 'abc'), (141L, '$b', 'def')]
}
```

O que cada parâmetro significa:

- '*tags*': conjunto dos nomes dos programas maliciosos que estão sendo buscados
- '*matches*': True significa positivo para infecção
- '*rule*': Arquivo com as regras compiladas para aplicação
- '*meta*': metadados do arquivo
- '*strings*': conjunto de strings para comparação dentro do objeto de varredura

5.2.4 Python

É uma linguagem de programação cujo desenvolvimento ocorre desde 1989. No ano de 1994, foi lançada a primeira distribuição da versão 1.0 oficial da linguagem e sua versão mais recente foi lançada em setembro de 2015. Para as necessidades do presente projeto, utilizaremos a versão 2.7 da linguagem, por suportar a implementação dos *scripts* do Yara e também ser a distribuição mais estável da linguagem, segundo o *site* oficial. O Yara suporta integração com scripts feitos tanto em código C puro como em Python, porém a escolha da linguagem se deu principalmente pela frequência, onde a grande maioria de scripts já prontos e documentados foi feita em Python. As implementações em C são mais comuns em âmbito corporativo, onde se constroem módulos mais otimizados para uso em varreduras de *malware* numa escala mais elevada e o desempenho torna-se um fator mais significativo na aplicação das regras.

5.2.5 Clam AV

O Clam AV é um antivírus de código aberto que possui um sistema muito interessante para a montagem de sua base de dados de assinaturas. O *Community Threat Tracking System* funciona coletando os dados de infecções ocorridas nos sistemas dos usuários e aprimorando suas assinaturas automaticamente com base nos dados estatísticos que a comunidade fornece. As bases buscam evidenciar quais *malwares* estão mais ativos no momento, e tornam todas as suas

descobertas públicas. Os usuários têm a opção de enviarem dados tanto anonimamente, quanto com um identificador único para cada instalação, e tais dados não são comercializados com terceiros, ou monetizados de alguma outra forma, todas as análises feitas por eles são apenas um serviço gratuito prestado à comunidade *open source*. No projeto que foi desenvolvido, utilizaram-se bases de dados com assinaturas utilizadas pelo Clam AV em suas próprias varreduras em *scripts* que as descomprimem e convertem para regras do Yara, que por sua vez podem ser editadas ou testadas isoladamente a critério do desenvolvedor.

5.2.6 Resumo da metodologia do projeto

O fluxo de desenvolvimento do projeto consiste na aplicação de um índice de regras de detecção do Yara, construído a partir da conversão de arquivos contendo bases de assinaturas do Clam AV para arquivos contendo regras de detecção do Yara, sobre o conjunto de amostras vivas de *malware* obtidas nos repositórios citados na seção do desenvolvimento do projeto. As varreduras são feitas via linha de comando e para agilizá-las, além do uso de índices para agrupar regras, também pode-se automatizar as varreduras em grupos maiores de arquivos aplicando um pouco de *shell script*. Os resultados das varreduras serão armazenados em arquivos de texto que irão conter todos os atributos analisados a cada arquivo a respeito de uma determinada regra. Vamos explicar com um pouco mais de detalhe como ficam caracterizadas as infecções nos *logs* de resultados, com casos de teste envolvendo expectativas de resultados positivos, falso-positivos e negativos. Ao final da seção de desenvolvimento, também será explicada a ideia da aplicação *web* para varreduras remotas e manipulação de regras do Yara.

6 DESENVOLVIMENTO

O desenvolvimento do projeto dividiu-se em quatro fases: a montagem do ambiente de testes de assinaturas construídas a partir de regras do Yara; a obtenção das regras e construção de um índice de agrupamento das mesmas para agilizar a execução das varreduras; obtenção e escolha de amostras de *malware* para testagem; bateria de testes final e análise de resultados. Complementarmente, foi desenvolvida a ideia de uma aplicação *web* para análise de *malware* com base em regras de detecção do Yara.

6.1 Obtenção das regras de detecção

7 CRONOGRAMA

O cronograma proposto para o caminhar do trabalho fica definido dentro das seguintes etapas:

- Etapa 1: Revisão Bibliográfica
- Etapa 2: Análise e desenvolvimento das técnicas que serão estudadas
- Etapa 3: Implementação computacional
- Etapa 4: Análise de resultados e desenvolvimento da monografia

Etapas	Meses											
	1	2	3	4	5	6	7	8	9	10	11	12
1	*	*	*	*	*	*						
2				*	*	*	*	*				
3						*	*	*	*	*		
4									*	*	*	*

8 CONCLUSÃO

Conclusão do projeto. Última coisa a ser escrita

Referências

BADDAR, S. A.-H.; MERLO, A.; MIGLIARDI, M. *Anomaly Detection in Computer Networks: A State-of-the-Art Review.* 2014. Acesso em: 15/04/2016. Disponível em: <http://isyou.info/jowua/papers/jowua-v5n4-2.pdf>. Citado 4 vezes nas páginas 24, 25, 26 e 27.

HASSAN, A. M. F. Master's thesis in network engineering. 2010. Acesso em: 15/04/2016. Disponível em: <http://www.diva-portal.org/smash/get/diva2:300733/FULLTEXT01.pdf>. Citado na página 17.

HURD, A. J. What is the current state of the science of cyber defense? 2015. Acesso em: 15/04/2016. Disponível em: <http://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-15-27889>. Citado na página 20.

IDIKA, N. M. A. A survey of malware detection techniques. 2007. Citado na página 14.

KUROSE, J. F. *Redes de Computadores e a Internet*. [S.l.: s.n.], 2009. Citado na página 18.

LENOVO. *Superfish Vulnerability.* 2016. Acesso em 30/05/2016. Disponível em: <https://support.lenovo.com/br/pt/product_security/superfish>. Citado na página 31.

MARKETING, L. W. S. B. P. *Zeus Trojan.* 2016. Acesso em 18/05/2016. Disponível em: <http://www.wiki-security.com/wiki/Parasite/ZeusTrojan/>. Citado na página 30.

SAEED ALI SELAMAT, A. M. A. A. I. A. A survey on malware and malware detection systems. *International Journal of Computer Applications (0975 – 8887)*, 2013. Citado na página 21.

SAWLE PRAJAKTA D.; GADICHA, A. Analysis of malware detection techniques in android. *International Journal of Computer Science and Mobile Computing*, 2014. Citado na página 14.

SZEWCZYK, P. A survey of computer and network security support. *Australian Information Security Management Conference*, 2012. Citado na página 16.