UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

VUNDA DA CONCEIÇÃO XAVIER

UMA ABORDAGEM FORENSE COMPUTACIONAL VOLTADA À PREVENÇÃO E DETECÇÃO DE ROOTKITS EM SISTEMAS LINUX

CRICIÚMA

2013

VUNDA DA CONCEIÇÃO XAVIER

UMA ABORDAGEM FORENSE COMPUTACIONAL VOLTADA À PREVENÇÃO E DETECÇÃO DE ROOTKITS EM SISTEMAS LINUX

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. MSc. Paulo João Martins

CRICIÚMA

2013

VUNDA DA CONCEIÇÃO XAVIER

UMA ABORDAGEM FORENSE COMPUTACIONAL VOLTADA À PREVENÇÃO E DETECÇÃO DE ROOTKITS EM SISTEMAS LINUX

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Perícia Forense Computacional.

Criciúma, 24 de Junho de 2013.

BANCA EXAMINADORA

Prof. MSc. Paulo João Martins - (Unesc) - Orientador

Prof. MSc. Rogério Antonio Casagrande - (Unesc)

Prof. Esp. Sérgio Coral - (Unesc)

Aos meus pais, irmãos e amigos.

AGRADECIMENTOS

Primeiramente agradeço a Deus por me ter iluminado a vida inteira. Aos meus pais Renato José Xavier e Margarida da Conceição Gaspar; e, irmãos, por todo amor, carinho, ensinamentos, suporte e apoio que me têm dado.

À Sociedade Nacional de Petróleos de Angola (Sonangol) por me apoiar nesta árdua e longa caminhada acadêmica e pela oportunidade de inserção em seu grande projeto de desenvolvimento de Angola.

A todo corpo docente e admimistrativo da Universidade do Extremo Sul Catarinense (Unesc) pela dedicação, atenção e apoio dado aos estudantes brasileiros e estrangeiros.

Ao professor e orientador Paulo João Martins por sua disponibilidade e dedicação.

Deixar um agradecimento especial aos colegas e amigos Aguinaldo Crispim, Anderson Silva, Eliúde Caracol e Rodolfo Mendes que muito me têm ajudado e companheiros em grandes debates filosóficos, científicos e tecnológicos.

E a todos os familiares e amigos que sempre me apaioram.

“A boa vontade desarmada da ciência e experiência não basta para fazer homens peritos.”

Desconhecido

RESUMO

Com a sociedade moderna cada vez mais dependente de meios computacionais, atualmente os crimes cibernéticos têm aumentado e prejudicado usuários e instituições. A internet é o meio mais utilizado para compartilhamento de dados e troca de informações; portanto, criminosos aproveitam-se de tal fator para concretização de ações ilícitas. Criminosos digitais usam ferramentas específicas para realização de suas atividades. Para facilitar os invasores, existem os *Rootkits* que são *softwares* maliciosos que permitem acesso irrestrito e auxiliam na ocultação e destruição de provas digitais. Para desvendar *cyber* crimes, a forense computacional auxilia administradores de sistemas e usuários em geral na adoção de técnicas e ferramentas permitindo descobrir evidências ligadas aos incidentes. No intuito de trazer ao conhecimento de profissionais voltados à segurança da informação, dentre outros, este trabalho teve como objetivo a análise forense de *Rootkits* em sistema *Linux* fornecendo auxilio para uma melhor compreensão das características intrusivas de forma a que se possa encontrar provas que confirmem a presença destes *malwares*. Utilizou-se o procedimento de pesquisa bibliográfica e elaboração de testes em ambiente *Linux* controlado onde foram instalados *Rootkits* para ataques e ferramentas para identificação e detecção. Também foram descritos conceitos sobre perícia computacional e metodologias, *Rootkits* e suas funcionalidades em sistemas operacionais *Linux* incluindo as formas de prevenção e detecção a incidentes de segurança envolvendo utilização destes programas maliciosos.

Palavras-chave: Segurança; Crimes Digitais; Forense Computacional; *Rootkits*, Sistemas *Linux*.

ABSTRACT

With the modern society more and more dependent on computational means, currently, the cyber crimes has increased and impaired users and institutions. The internet is an environment most used for data sharing and information's exchange; therefore, criminals take advantage of this factor for achieving illicit actions. Cybercriminals use specific tools to carry out their activities. To facilitate the invaders, there are Rootkits that are malicious software that allow unrestricted access and assist in the concealment and destruction of digital evidence. To unveil cyber crimes, computer forensics to assist system administrators and general users in the adoption of techniques and tools allowing uncover evidence related to the incidents. In order to bring to the knowledge of professionals focused on information security, among others, this article aimed to the forensics of Rootkits in Linux providing aid to a better understanding of the characteristics intrusive so we could find evidence of the presence of these malware. We used the procedure of bibliographical research and preparation of testing in controlled Linux environment where Rootkits were installed to attacks and tools for identification and detection. Were also described concepts of computational forensic and methodologies, Rootkits and their functionalities on Linux operating systems including ways to prevent and detect security incidents involving use of these malicious programs.

Key-words: Security, Digital Crimes, Computer Forensics, Rootkits, Linux systems.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

Art	Artigo
CAINE	Computer Aided Investigate Environment
CD	Compact Disk
CP	Código Penal
DEFT	Digital Evidence & Forensic Toolkit
DNS	Domain Name System
CAINE	Prefeitura Municipal de Criciúma
DVD	Digital Video Disk
Ext	Extended Filesystem
FAT	File Allocation Table
FCCU	Federal Computer Crime Unit
FDTK	Forense Digital Toolkit
FTK	Forensic Toolkit
FTP	File Transfer Protocol
GNOME	GNU Network Object Model Environment
GNU	Gnu's Not Unix
GPS	Global Positioning System
HD	Hard Disk
HTML	Hyper Text Markup Language
KFF	Known File Filter
LINUX	Linux Is Not Unix
LXDE	Lightweight X 11 Desktop Environment
MD5	Message-Digest algorithm 5
NTFS	New Technology File System
OOV	Order of Volatility
PL	Pleito
POPs	Procedimentos Operacionais Padrão
UFS	Unix File System
UNIX	Uniplexed Information and Computing System
VM	Virtual Machine

SUMÁRIO

1 INTRODUÇÃO

Atualmente os sistemas operacionais *Linux* são utilizados para atender diversos serviços dentre eles: hospedagem *web*, *File Transfer Protocol* (FTP), *backup*/restauração, *Domain Name System* (DNS), banco de dados, entre outros. Dada a relevância computacional dos serviços descritos acima surge a necessidade de manutenção constante desses ambientes operacionais para que funcionem adequadamente e, alguns especialistas como administradores de sistemas e peritos forenses devem possuir conhecimentos mínimos para mantê-los seguros e investigar os possíveis incidentes de segurança quando invasores acessam ou tentam acessá-los.

Segurança da Informação é todo procedimento adotado que envolve a proteção de dados e informações em sistemas por meio de negação de acessos não autorizados, bem como contra invasões e alterações de conteúdos confidenciais armazenados em dispositivos físicos ou em rede; sendo que engloba também as metodologias que visam a detecção de possíveis ameaças de segurança (ARAUJO, 2000).

Para se analisar os possíveis incidentes de segurança é pertinente conhecer-se as técnicas utilizadas pelos invasores em acessos não autorizados aos sistemas assim como os procedimentos para investigar e identificar os suspeitos e suas atividades no sistema. Um exemplo de estratégia usada por possíveis autores de crimes digitais durante invasões de sistemas é a utilização de *Rootkits*. São códigos maliciosos ou *malwares* utilizados na tentativa de invasão de sistemas computacionais, sendo que estes permanecem ocultos no sistema mesmo com a existência de antivírus, *antispyware* e outros *softwares* de segurança (FREITAS, 2006).

Cada *Rootkit* pode ser instalado em diferentes níveis do sistema. Para cada nível, se estabelece de uma forma diferente e requer, portanto, estratégias diferenciadas para ser detectado. Sendo assim, eles podem ser classificados em: binários, modo *kernel* (*LKM/drivers*) e de biblioteca (modo usuário) (ROSANES, 2011).

Para eficiente detecção dos *Rootkits* e suas atividades maliciosas no sistema, a Perícia Forense auxilia na busca de respostas por meio de procedimentos

padrões de Investigação. A Perícia destina-se a desvendar as seguintes perguntas pertinentes: O que foi feito? Quando foi feito? Como foi feito? E como se infiltrou? (FREITAS, 2006).

Sendo assim, nesta pesquisa realizaram-se testes com a simulação de ataques de *Rootkits*, foram utilizados *softwares anti-rootkits* e ferramentas de perícia para detecção dos mesmos e decreveu-se o comportamento dos *softwares*, o que possibilitou descobrir se um computador com sistema Operacional *Linux* foi comprometido e como fazer para solucionar o problema encontrado.

1.1 OBJETIVO GERAL

Apressentar técnicas de prevenção e detecção de *Rootkits* para auxiliar as equipes de resposta a incidentes de segurança em Sistemas *Linux* comprometidos com a presença destes *malwares*.

1.2 OBJETIVOS ESPECÍFICOS

Para que este trabalho se concretize, foram definidos os seguintes objetivos:

a) descrever a cerca de diferentes tipos de *Rootkits*;

b) identificar *Rootkits* binários;

c) entender e aplicar os conceitos sobre Perícia Forense Computacional;

d) aplicar os procedimentos computacionais de virtualização e usabilidade de *softwares anti-rootkits* para realizar a Forense Computacional de *Rootkits* em Sistemas *Linux*;

e) realizar testes envolvendo intrusão e detecção de *Rootkits* binários em ambiente *Linux*.

1.3 JUSTIFICATIVA

A tecnologia tem evoluído dia após dia e a Internet é parte deste avanço. Portanto, muitos criminosos usam o meio eletrônico para realização de atividades

ilícitas, o que obriga as empresas legais a se preocuparem com futuras investigações computacionais. Com a difusão da Internet nas mais variadas instituições sociais; torna-se cada vez mais difícil identificar os criminosos por estarem em várias localizações geográficas, dificultando o trabalho dos peritos em combater e/ou desvendar os crimes. Com o constante avanço tecnológico é difícil imaginar empresas, fábricas, comércios, prestadores de serviço ou até mesmo universidades sem computadores e afins.

A interligação das demais instituições sociais por meio da Internet propicia o crescimento de crimes de meios digitais, surgindo assim à necessidade de uma área de pesquisa denominada Investigação Forense ou Perícia Forense ou Computação Forense que se encarrega de desvendar os mais variados crimes digitais.

Computação Forense é a ciência voltada para o estudo e avaliação de situações que envolvam a computação como meio para cometer crimes (COSTA, 2003).

Logo define-se a Computação Forense como sendo a ciência que visa estudar, analisar, recuperar e coletar dados que posteriormente irão servir como meio de provar ou inocentar os indivíduos que fazem parte da investigação, e desta forma utiliza as provas em meios digitais, onde estas necessitam ser documentadas. A Computação Forense é uma área de pesquisa nova e que está consideravelmente em constante desenvolvimento.

Entretanto, os sistemas operacionais *Linux* têm sido alvos constantes de ataques de *Rootkits*. Estes têm como objetivo acessar contas de usuário em sistemas procurando se esconder dos s*oftware*s de segurança. Partindo desse contexto surge então a necessidade de se adotar procedimentos forenses computacionais para análise, detecção e coleta de evidências. Esses procedimentos englobam a utilização de ferramentas para detectar *Rootkits* em Sistemas *Linux*.

1.4 ESTRUTURA DA PESQUISA

Como dito anteriormente, este trabalho tem como finalidade o estudo de análise forense de *Rootkits* em *Linux*, focando principalmente nas técnicas e programas utilizados atualmente na identificação e detecção desses *malwares*.

Entretanto, o trabalho foi repartido em duas etapas e/ou partes sendo que a primeira foi composta por um levantamento bibliográfico afim de se adquirir material teórico suficiente para a pesquisa, e em sequência a parte prática onde foram definidos os testes envolvendo intrusão de *Rootkits* em ambiente *Linux* controlado e instalação de ferramentas que visam à detecção destas intrusões.

No primeiro capítulo é apresentado uma breve introdução do tema e definidos os objetivos geral e específicos assim como a justificativa para a elaboração do trabalho. Em seguida apresenta-se conceitos sobre segurança da informação e crimes digitais e algumas questões envolvendo legislação, isto é, no capítulo dois.

O terceiro capítulo aborda os conceitos sobre a forense computacional, metodologias que devem ser adotados, os tipos de perícia existentes e os diversos *softwares* aplicados durante as etapas de investigação. No quarto capítulo é apresentada uma breve descrição do sistema operacional *Linux* e alguns motivos nos quais muitos investigadores forenses utilizam este ambiente e, em sequência são descritos conceitos sobre os *Rootkits*, tipos e técnicas de prevenção e detecção.

O capítulo cinco apresenta alguns trabalhos correlatos na área de Perícia Forense Computacional. No sexto capítulo tem-se o trabalho proposto onde são descritos os programas *Rootkits* utilizados para os testes, as ferramentas de detecção e os resultados obtidos.

E finalmente está descrita a conclusão e as recomendações para trabalhos futuros envolvendo a Forense Computacional de *Rootkits*.

2 SEGURANÇA DA INFORMAÇÃO, CRIMES DIGITAIS E LEGISLAÇÃO

Ao longo dos anos, a sociedade tem sido acompanhada por inovações tecnológicas que permitem com que as pessoas usem-nas diariamente no intuito de adquirir novos conhecimentos e trocarem informações. Esta evolução é conhecida também como revolução informática, que possibilita dentre as diversas ações, a substituição das atividades humanas por máquinas; reduzindo acima de tudo o esforço das pessoas (CRESPO, 2011).

O homem sempre procurou criar máquinas e ferramentas mais sofisticadas, o que torna as atividades diárias pouco difíceis e mais agradáveis. Desta forma, os computadores também foram feitos para facilitarem as nossas tarefas; visto que guardam e transformam informações por meio de instruções pré-estabelecidas (PAIVA, 2012).

Com enfoque computacional, o termo informação está associado à forma com que dados são organizados de maneira que sejam compreendidos e com a finalidade de serem prestativos durante nossa tomada de decisão. Porém, a segurança da informação consiste em três princípios básicos: a confidencialidade, a integridade e a disponibilidade da informação (COSTA, 2011).

A confidencialidade garante que apenas usuários autorizados tenham acesso a determinada informação; o que significa proteger informações contra pessoas que não são explicitamente autorizadas. A integridade é a proteção das informações exatas e completas e das técnicas de processamento; portanto ela é garantida quando a informação acessada está completa, sem alterações e, entretanto, confiável. Já a disponibilidade permite a obtenção de conhecimentos por parte de pessoas autorizadas; em outras palavras, é quando a informação está acessível, por pessoas autorizadas, sempre que necessário (COSTA, 2011).

A segurança da informação é definida conforme Moraes (2010) como o procedimento que visa a proteção das informações de uso indevido seja de maneira proposital ou acidental por indivíduos pertencentes à organização ou fora dela.

Entretanto, a segurança da informação tem tido bastante impacto visto que as empresas têm armazenado e processado suas informações através de meios e recursos computacionais; para realização de atividades com maior eficiência, as organizações dependem do ambiente computacional e por outro lado todas as

informações estão disponíveis a todos os membros da corporação (FONTES, 2006).

As organizações atuais são submetidas a diversas ameaças de segurança no intuito de que estas possam prejudicar as atividades, causando prejuízos, interrupção de serviços, de vendas e atendimentos. Considerando que a informação e todos os sistemas da empresa são pertinentes para o sucesso organizacional, é indispensável estarem protegidos para se conservar a imagem da empresa, seu estatuto, a competitividade, o faturamento e muito mais (COSTA, 2011).

É importante salientar que nenhuma informação, rede ou sistema de segurança é completamente seguro, ou opera sem falhas. Sempre que acessamos a Internet na tentativa de procurarmos mais conhecimentos estamos, ao mesmo tempo, correndo ameaças de segurança; visto que ela tem uma abrangência mundial, vulnerável e sujeita a ataques entre usuários (MORAES, 2010).

A importância da segurança da informação dentro de uma organização tem sido bastante compreendida. A legislação, a regulamentação e a boa governança são motivadores para que as organizações considerem o papel que a segurança da informação desempenha na proteção de dados. Enquanto melhor for entendida, a adoção de boas práticas de segurança da informação, isto está longe de ser uniforme em todas as organizações, com empresas corporativas se saindo melhor do que muitas organizações menores que estão arrastando em seus conhecimentos a implantação de práticas seguras. Com o crescimento significativo de ameaças decorrentes de crimes cibernéticos e atividades relacionadas, é cada vez mais importante que todas as organizações abordem a questão da garantia da boa segurança da informação (CLARKE, 2010, tradução nossa).

De maneira geral, uma organização de sucesso deve ter as seguintes camadas múltiplas de segurança para proteger suas operações (WHITMAN; MATTORD, 2012, tradução nossa):

a) segurança física, para proteger itens físicos, objetos ou áreas de acesso não autorizado e uso indevido;

b) pessoal de segurança, para proteger o indivíduo ou grupo de indivíduos que estão autorizados para acessar a organização e suas operações;

c) operações de segurança, para proteger os detalhes de uma

determinada operação ou série de atividades;

d) comunicações de segurança, para proteger os meios de comunicação, tecnologia e conteúdo;

e) rede segura, para proteger os componentes de rede, conexões e conteúdos;

f) segurança da informação para proteger a confidencialidade, integridade e disponibilidade dos ativos de informação, seja em armazenamento, transmissão ou processamento. Ela é alcançada por meio da aplicação da política, educação, treinamento, conscientização e tecnologia.

2.1 CRIMES DIGITAIS E LEGISLAÇÃO

Os grandes gênios e inventores de máquinas são os grandes responsáveis pelo fato destas realizarem determinadas tarefas anteriormente feita em sua plenitude por homens. Estes benefícios contribuíram significativamente para o desenvolvimento industrial, porém, infelizmente por trás disto surgiram também os prejuízos. Visto que no nosso cotidiano a usabilidade de computadores, Internet e outros meios são cada vez maiores, torna-se impossível livrar-se dessas tecnologias, pelo fato de que são proveitosas e benéficas; assim sendo, os criminosos têm se aproveitado de tais situações para praticarem suas atividades conhecidas como crimes digitais (LIMA, 2011).

Os crimes digitais são as ações realizadas por criminosos na tentativa de acesso a sistemas computacionais ilegalmente; dado que essas condutas englobam interrupções dos sinais de comunicação, alterações e/ou exclusões de dados e informações, violação de direitos de autoria, estímulo ao ódio e discriminação, menosprezo de crenças religiosas, terrorismo, a disseminação de cenas de pornografia infantil e muito mais. Como o próprio nome já indica, pode-se de maneira mais simplificada conceituar este termo como: atos ilícitos realizados por meio de máquinas computadorizadas, sendo que na maioria dos casos os delitos são praticados via Internet e os meios mais utilizados são os computadores (PAIVA, 2012).

Entretanto, não existe uma denominação única para esses crimes.

Diversas denominações são atribuídas mundialmente podendo ser chamados também de: crimes de informática ou *cyber* crimes, crimes de computador, crimes eletrônicos, delitos computacionais, crimes informacionais, crimes telemáticos, crimes cibernéticos e crimes virtuais (LIMA, 2011).

Os criminosos têm adotado diversas maneiras para cometer seus delitos. Uma delas é a utilização dos famosos vírus de computadores; sendo que estes são *softwares* maliciosos e quando instalados ou com acesso a determinada máquina têm a função de destruir ou dificultar o funcionamento de outros programas. Por outro lado, estes *softwares* se aproveitam das falhas de segurança e aumentam a vulnerabilidade nos sistemas operacionais provocando danos sérios no computador (CRESPO, 2011).

Ataques ocorrem quando um projeto individual ou em grupo implanta *softwares* para invadir um sistema. Maior parte deles é conhecida por código malicioso ou *software* malicioso, ou, por vezes, *malware*. Estes componentes de *software* ou programas são projetados para danificar, destruir ou ocultar arquivos em sistemas de destino. Alguns dos exemplos mais comuns de códigos maliciosos são os vírus e *worms*, *rootkits* e cavalos de tróia (WHITMAN; MATTORD, 2012, tradução nossa).

Entretanto, atitudes que visam produzir e propagar vírus são tidas como criminosas e punidas em lei. Assim sendo, quanto à inserção e difusão de programas maliciosos, a constituição brasileira no Art. 163-A do PL n. 84/99 diz: a inserção ou difusão de código malicioso em dispositivos de comunicação, rede de computadores ou sistema informatizado tem como pena – reclusão, de um a três anos, e multa. O mesmo artigo afirma que (CRESPO, 2011):

> a) se o crime em questão resultar em danificação, deterioração, dificuldade de funcionamento ou funcionamento não autorizado pelo proprietário do sistema, a pena será de: reclusão de dois a quatro anos, e multa;
>
> b) já se o criminoso utilizar identidade falsa ou de outras pessoas para a prática do delito, a pena é aumentada de sexta parte.

Diversas caracterizações sobre crimes virtuais são encontradas no Código Penal Brasileiro. Os principais crimes digitais já se encontram inseridos na legislação penal do Brasil, dentre eles, destacam-se (LIMA, 2011):

a) art. 171, estelionato;

b) art. 172, expedição de duplicata simulada;

c) crimes contra o privilégio de invenção;

d) art. 295, falsificação de documento público e particular;

e) falsidade ideológica (art. 299);

f) delitos contra a inviolabilidade da correspondência (art. 151);

g) crimes contra a correspondência comercial (art. 152);

h) inclusão de dados falsos em sistemas de informação, art. 313-A: inserir dados falsos, modificar ou excluir injustamente dados corretos em sistemas computadorizados ou banco de dados envolvendo a administração pública no intuito de adquirir vantagens indevidas sejam individuais, coletivas ou para provocar prejuízos;

i) alteração não autorizada de sistema de informação, art. 313-B: o indivíduo que alterar o sistema ou *software* sem a prévia autorização ou requisição de sua autoridade competente.

Na Tabela 1 são listadas algumas atividades tidas como criminosas.

Tabela 1 – Ações ilícitas e tipologias

Ação	Tipo
Falar em um chat que alguém cometeu crime	Calúnia
Enviar Email para terceiros com informação considerada confidencial	Divulgação de segredo
Enviar vírus que de destroem equipamentos ou informações	Dano
Copiar um conteúdo e não mencionar a fonte	Violação ao direito autoral
Enviar email com remetente falso (exemplo: spam)	Falsa identidade
Efetuar cadastro com nome falso em uma loja virtual	Inserção de dados falsos em sistemas
Entrar na rede de uma organização e alterar informações	Adulteração de dados em sistema de informações
Falar em um chat que alguém é isso ou aquilo em função de sua cor	Preconceito ou discriminação racial

Enviar fotos de crianças nuas online	Pedofilia
Usar logomarca de uma organização em página da Internet sem a autorização do titular	Crime contra a propriedade industrial
Usar cópia de software sem ter a licença para tal	Crimes contra pirataria de software

Fonte: Ng (2007).

De uma maneira geral, os crimes informáticos são punidos em lei em muitos países. Por exemplo, em Portugal os crimes digitais são penalizados em função da lei n. 109/91. Na legislação portuguesa são punidas as seguintes atividades criminosas (CRESPO, 2011):

a) falsidade informática. Descrita no art. 4° em que a lei pune ações ilícitas que envolvem acesso, alteração, e anulação de dados ou *softwares* com a finalidade de adulterarção das informações;

b) perda de dados e programas, cuja descrição consta no art. 5° por meio do qual cita a destruição de informações eletrônicas e dos *softwares*;

c) o art. 6° trata sobre a sabotagem informática. Sendo que a lei pune as ações de exclusão, alteração e acesso não autorizado (de dados e programas) que causam transtornos no desempenho dos sistemas computacionais e na comunicação dos dados;

d) o acesso ilícito definido no art. 7° que pune todas as ações feitas pelos criminosos na tentativa de invasão de sistemas;

e) o art. 8° faz uma descrição acerca da intercepção ilegítima que penaliza todas as atividades ilegais que envolvem interceptações ilícitas que ocorrem em meios computacionais;

f) a reprodução e a utilização indevida de programas que segundo o art. 9° pune a divulgação, transferência e cópias destes sem autorização.

No capítulo a seguir apresenta os conceitos sobre a investigação forense computacional numa visão geral; destacando-se a priori sua pertinência e aplicações com enfoque computacional e judicial. Posteriormente são definidas as metodologias necessárias para o sucesso das atividades periciais; métodos estes que envolvem as fases de coleta, exame (preservação), e os resultados. Abordar-se-á também

alguns tipos de *softwares* de perícia que auxiliam bastante o perito na obtenção das evidências, os tipos de investigação e alguns exemplos relevantes.

3 PERÍCIA FORENSE COMPUTACIONAL

Vive-se num mundo cada vez mais tecnológico no qual as pessoas têm acesso às mais variadas informações. Com a proliferação de usuários com acesso à Internet e computadores, algumas pessoas têm se aproveitado desses meios para realizarem atividades ilícitas tais como: acesso não autorizado de informações secretas, divulgação de informações ofensivas, fraudes bancárias, racismo, pedofilia dentre outras.

Dada a relevância computacional das atividades descritas, surge a perícia forense como a ciência que se preocupa em investigar e desvendar os criminosos e suas ações ilícitas.

Desta forma, a perícia procura responder perguntas importantes como: O que aconteceu? Quando aconteceu? Por que aconteceu? E, principalmente quem é ou foi o responsável pelo ato? Durante todo o procedimento de investigação é necessário estabelecer uma relação entre as informações recolhidas a partir de fontes distintas, mas que obedecem aos critérios de confiabilidade com a finalidade de que: o incidente seja explorado com maior precisão, se tenha uma visão mais ampla do ocorrido, se elimine as incertezas nos dados. É de salientar que a ausência de informações acarreta consequências e não prova se determinado ato tenha acontecido ou não (MACEDO, 2010).

Ao longo dos anos, conceitos e aplicações sobre a forense computacional têm sido cada vez mais desenvolvidos por profissionais em estágios avançados de qualificação e especialização; com isso, muitos vestígios criminosos deixados em locais de crime atualmente só são comprovados após uma série de testes laboratoriais. Diariamente pessoas cometem diversos delitos e por vezes têm a sorte ou até mesmo a maestria de não deixarem vestígios e, em casos idênticos, a perícia atua em desvendar as pistas impossíveis de serem vistas a olho nu; entretanto elas são reconstituídas por meio de testes em laboratórios por meio de uma sequência de regras e padrões estipulados de maneira que as provas incriminadas a serem desvendadas sejam válidas para que possam ser apresentadas durante o julgamento (TOLENTINO; SILVA; MELLO, 2011).

“A Perícia forense é a aplicação de conhecimentos em informática e técnicas de investigação com a finalidade de obtenção de evidências” (FREITAS, 2006, p. 1).

Assim sendo, a perícia é o procedimento baseado em buscas, extração, análises de vários fatos ocorridos em diferentes maneiras, tempos e locais e que podem ser fundamentais em algum processo judicial (CAIXETA, 2011).

Com enfoque computacional está totalmente interligada com as técnicas investigativas de crimes digitais por meio de uma recolha de dados a serem utilizados nas fases de identificação, preservação, análise e documentação para que o perito possa conseguir as evidências em formato digital e apresentá-las para incriminar o suspeito em questão (TOLENTINO; SILVA; MELLO, 2011).

Existem quatro procedimentos essenciais envolvidos na realização da perícia forense computacional: o perito deve reconhecer todas as evidências, seguida da conservação das mesmas para que se faça uma posterior análise e com a finalidade de que as evidências sejam apresentadas com precisão e coerência (FREITAS, 2006).

A análise das evidências deve ser feita com rigor e com a máxima responsabilidade por parte do perito; pois evita possíveis irregularidades na investigação do caso e pode facilitar o trabalho pericial e influenciar o juiz a considerar como coerentes e justificáveis as provas apresentadas (COSTA, 2008).

Por conseguinte, pode-se afirmar que todo e qualquer ato considerado ilegal e ao mesmo tempo cometido utilizando-se recursos computacionais, é suscetível de uma perícia e, por conseguinte, deve ser feita por profissionais especialistas na área computacional ou peritos forenses por meio de uma sequência de processos bem estipulados e aceitos judicialmente. Sendo assim a forense computacional deve utilizar técnicas investigativas para obtenção de evidências criminosas aceitas em lei como é o caso de espionagens empresariais e a utilização imprópria de *softwares* de determinada empresa, dentre outras (CAIXETA, 2011).

Objetivando a investigação forense computacional, seja ela judicial ou corporativa, pode-se inferir que ela influência nos procedimentos adotados pelo perito. Num sistema comprometido, quanto mais informações retiradas, maior é a probabilidade de ocorrência de alterações indevidas. Contrariamente as outras áreas voltadas à computação, podemos perceber que a questão prioritária da perícia

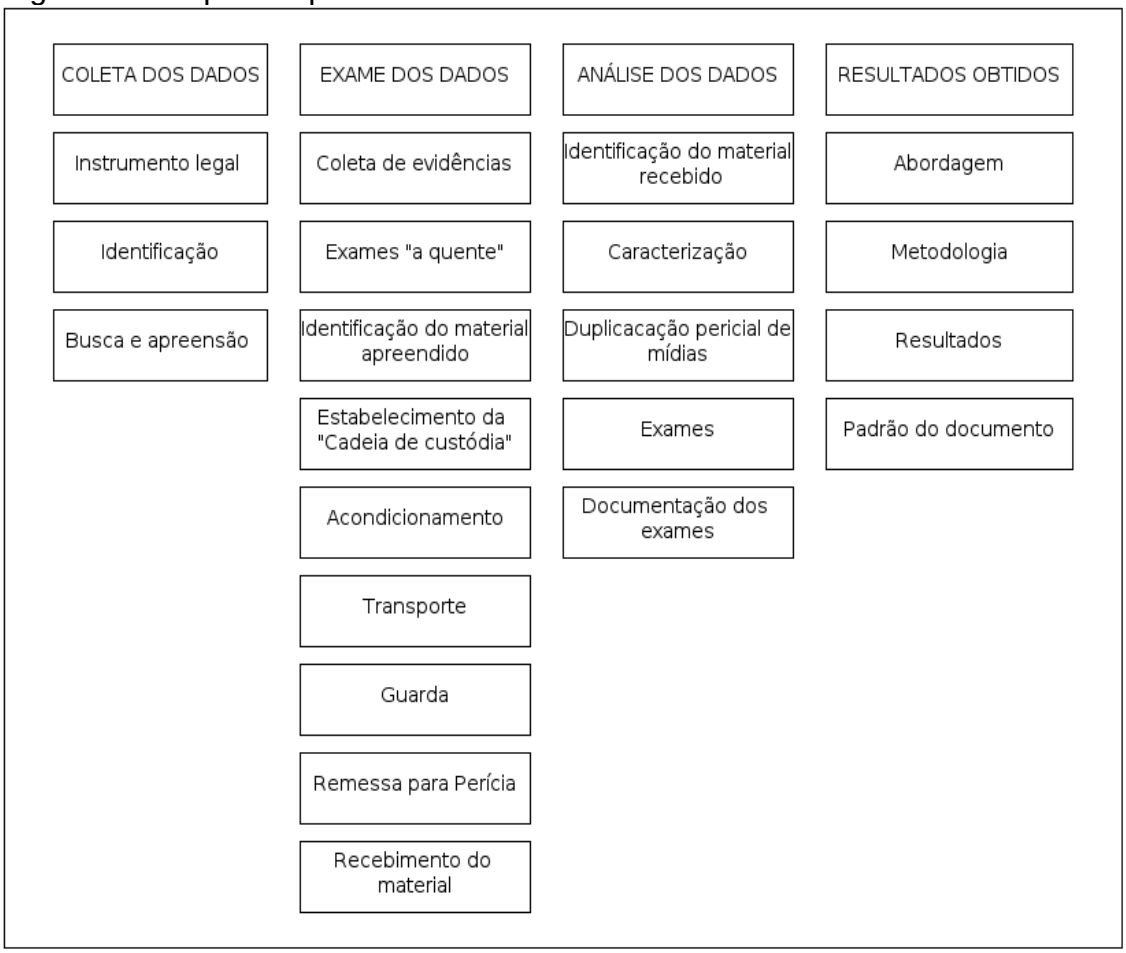
forense computacional não é a velocidade, mas sim buscar a integridade e a exatidão das evidências (LIMA, 2009).

3.1 METODOLOGIAS APLICADAS NA FORENSE COMPUTACIONAL

Para o alcance dos objetivos propostos numa investigação forense é importante que siga uma metodologia organizada e bem elaborada para que o trabalho pericial tenha aprovação e aceitação judicial. Sendo assim, ela é focada em procedimentos forenses de investigação digital definidos com clareza, precisão e objetividade. Portanto os procedimentos são divididos em quatro etapas: coleta dos dados, o exame dos dados coletados, análise e os resultados obtidos; sendo que a coleta abrange: o isolamento do local, a coleta das evidências, a identificação, a garantia da integridade, etiquetar as evidências e a cadeia de custódia. O exame engloba a identificação, a extração, a filtragem, a preservação das evidências e a documentação. Fazem parte da análise a identificação (de pessoa, eventos e lugares), a correlação destas, a maneira como a cena é construída e a documentação. Redigir o laudo pericial, anexar as evidências e outros documentos, a comprovação, os formulários e registros são características importantes para a aquisição dos resultados (CAIXETA, 2011).

Entretanto, segundo Costa (2011), as etapas de investigação forense podem ser detalhadas por meio de um diagrama no qual cada etapa possui as ações necessárias para que o perito possa realizar suas atividades com a maior eficiência e eficácia possível. A figura a seguir apresenta o diagrama das etapas de perícia.

Figura 1 – Etapas de perícia forense.



Fonte: Adaptado de Costa (2011).

3.1.1 Coleta dos dados

Este é o procedimento inicial a ser seguido pelo perito. Se a coleta for feita incorretamente, a investigação acaba sendo comprometida e os dados poderão ser obtidos com irregularidade. Baseando-se na característica volátil dos meios eletrônicos, por exemplo, um simples desligamento do computador em poucas frações de segundo pode danificar ou alterar as evidências. Será necessário definir estratégias indispensáveis de forma a garantirem que os dados a serem coletados não sejam destruídos ou perdidos (LIMA, 2009).

É durante esta fase, que o perito tem a obrigação de se familiarizar com os fatos verificando todos os detalhes do caso. Deve focar-se principalmente nos dados armazenados nos mais variados dispositivos: discos rígidos dos computadores e outros meios computacionais essenciais achados no local do crime. Deverá também priorizar as análises *in vivo*, isto é, deve fazer uma perícia mais

precisa de todos os dispositivos que estiverem conectados recolhendo assim muitas evidências importantes (CAIXETA, 2011).

As evidências encontradas devem estar interligadas com o tipo de crime. Por exemplo, em crimes envolvendo pornografia procuram-se por imagens armazenadas e apagadas no computador, lista de sites acessados, arquivos, documentos excluídos e todos os *Emails* enviados e recebidos na máquina comprometida. Em todos os dispositivos de armazenamento de dados e/ou informações encontradas na cena do crime, o perito deverá ter a árdua missão de procurar por evidências. Anotações, nome de pessoas e empresas, datas, números de telefone e documentos são informações importantes durante a coleta dos dados (PIMENTA, 2007).

Durante uma investigação forense, os possíveis dispositivos através dos quais os dados são coletados são: dispositivos de armazenamento em rede, pendrives e cartões de memória, CDs e DVDs, máquinas fotográficas, computadores, celulares e *smartphones* e *tablets* (CAIXETA, 2011).

Nota-se também que durante a coleta, os dados podem encontrar-se armazenados em locais não físicos como provedores de Internet, FTP e servidores corporativos. Desta forma, o trabalho pericial só será realizado por meio de uma ordem judicial. É importante também que o perito faça uma cópia de todos os dados encontrados durante a investigação; pois garante a proteção integral das evidências para que sejam validadas perante a justiça. Por conseguinte deve ser feito um *Backup* de todos os materiais, pelo menos duas cópias seguras: uma para recolha dos dados importantes e a outra deve ser conservada em local seguro caso seja necessário a realização de novas análises (SILVA, 2010).

A garantia da integridade das evidências é um fator importantíssimo. Baseia-se principalmente na utilização de *softwares* específicos que aplicam um algoritmo *hash* e que geram uma sequência única para a imagem lógica garantindo que se por alguma razão os dados sofrerem alterações, o *hash* obtido não continuará o mesmo provando assim que houve mudanças após a coleta dos dados. Esta é uma prática necessária para o eficiente trabalho do perito. Outro fator de realce é a ótima conservação de todos os meios recolhidos no local do crime, sendo que os mesmos devem ser obrigatoriamente identificados, embalados e anexados em um único documento a ser inserido na cadeia de custódia (CAIXETA, 2011).

Ainda durante esta etapa, o perito deve verificar cópias de arquivos, e não só os arquivos originais. Durante a fase de coleta, o analista forense deve fazer várias cópias dos arquivos desejados ou sistemas de arquivos, geralmente cópia mestra e uma cópia de trabalho. O profissional pode então trabalhar com a cópia de trabalho dos arquivos sem afetar os arquivos originais ou a cópia mestra. (KENT et al, 2006, tradução nossa).

3.1.2 Exame dos dados coletados

As informações importantes do processo de investigação pericial são extraídas e filtradas justamente nesta etapa. Por exemplo, num arquivo de *logs* contendo cerca de mil entradas, há a probabilidade de que somente algumas delas sejam interessantes para o trabalho do perito. Ainda nesta etapa, deve-se fazer uma observação mais clara dos dados achados: arquivos excluídos e posteriormente restaurados, arquivos ocultos, fragmentos de arquivos encontrados em áreas não alocadas e em regiões alocadas, mas que não estão sendo utilizadas (SILVA, 2010).

Examinar os dados coletados é um procedimento de extrema importância para o sucesso da atividade do perito. Porém é uma etapa difícil, pois os meios computacionais atuais são gigantescos em: capacidade de armazenamento, diversidade na topologia de arquivos (imagens, áudio, arquivos criptografados e compactados); visto que vários arquivos ocultam informações dentro de outros sem necessidade de alterações nas suas propriedades básicas, o que faz com que o profissional em perícia adote uma postura mais cautelosa voltada na atenção e aptidão em identificar e restaurar os dados. Neste amplo repositório de dados restaurados podem estar contidas diversas informações não importantes, sendo que estas podem ser descartadas num dado instante, porém o perito tem a obrigação de ter cautela para não desconsiderar informações relevantes (CAIXETA, 2011).

A preservação das evidências é também um fator de realce durante o exame pericial. A princípio, em uma investigação forense, as evidências encontradas não devem ser desfeitas e nem modificadas; assim sendo, o perito deve conservá-las para que não haja incertezas na sua veracidade. Portanto, para que as provas não sejam colocadas em perigo, o perito deve adotar as seguintes providências: criar imagens do sistema periciado ou duplicação pericial; em casos de perícia *in*

vivo, guardar as evidências em um meio de armazenamento que possa ser bloqueado contra regravação; lacrar todas as evidências em sacos e etiquetas; etiquetar todos os cabos e componentes do computador (FREITAS, 2006).

3.1.3 Análise dos dados

Após o exame ser completado, o próximo passo é o de executar a análise dos dados extraídos. Existem muitas ferramentas disponíveis que podem ser úteis na análise de diferentes tipos de dados. Ao usar essas ferramentas ou realizar revisões manuais de dados, os peritos devem estar cientes do valor do uso de tempos do sistema de arquivos. Saber quando ocorreu um incidente, o instante em que um arquivo foi criado ou modificado, ou um *Email* enviado pode ser crítico para análise forense. Por exemplo, essas informações podem ser usadas para reconstruir um cronograma de atividades. Embora isso possa parecer uma tarefa simples, é muitas vezes complicada por causa das diferenças intencionais ou não em configurações de tempo entre sistemas. Saber a hora, data e fuso horário para um computador cujos dados serão analisados pode ajudar muito a perícia (KENT et al, 2006, tradução nossa).

Antes do início do procedimento de análise deve-se determinar quê informações são importantes. A seleção dos dados leva em conta o estado em que o sistema operacional se encontra. Perguntas como: quais dados têm prioridade de análise? O que deve ser analisado? É necessário analisar todos os dados? Será que algumas evidências podem ficar comprometidas pelo fato de não serem analisadas em período de execução? (SANTOS, 2008).

A análise consiste na avaliação de todas as informações adquiridas durante o exame dos dados com a finalidade de que as provas digitais encontradas em todos os equipamentos sejam reconhecidas e tenha ligação direta com o caso investigado. É responsabilidade do perito, analisar detalhadamente todos os arquivos que têm relação direta com o delito, ainda que os mesmos levam um longo período de tempo de análise (ELEUTÉRIO; MACHADO, 2010).

Após à avaliação de todos os dados e obtenção das imagens forenses destes, a análise é iniciada. Este procedimento é o cerne da investigação. A principal questão que o perito deverá ter em mente sempre que executar a análise é

a completude. Deve assegurar-se que observou em todos os cantos e que não perdeu-se nada de relevante. Advogados odeiam quando o conselho de oposição encontra novas evidências que destroem o caso. O profissional deve ser completo e criativo; pensamento não convencional ajuda muito nesta fase (PHILIPP; COWEN; DAVIS, 2010, tradução nossa).

Entretanto, para que a análise pericial seja realizada com mais eficiência, algumas técnicas são fundamentais. Dentre elas destacam-se: a utilização de *Known File Filter*, pesquisa por palavras-chave, navegação pelo sistema de pastas e arquivos, a visualização adequada de arquivos, utilização de ferramentas de apoio e a virtualização (ELEUTÉRIO; MACHADO, 2010).

A técnica de *Known File Filtrer* (KFF) consiste na elaboração de uma lista resumida de arquivos a serem usados durante a análise dos dispositivos ligados à investigação; desta forma, o perito pode reduzir significativamente o número de arquivos a ser verificado no disco rígido descartando acima de tudo aqueles sem interligação com o caso investigado (ELEUTÉRIO; MACHADO, 2010).

O KFF torna a análise pericial mais eficiente desde que o investigador forense consiga distinguir quais arquivos são pertencentes ao processo investigativo e quais são descartáveis. Depois de feita a listagem, o perito pode optar em buscas mais rápidas no sistema, sendo que a mesma é realizada por meio da técnica de pesquisa por palavras-chave; por exemplo, se o perito procurar pela palavra furto, entretanto todos os arquivos e parte destes que contém tal palavra poderão ser encontrados, até mesmo os arquivos recuperados, alterados e excluídos (ELEUTÉRIO; MACHADO, 2010).

Esta técnica é fundamental para a análise na medida em que auxilia o investigador forense em recolher a maior parte das evidências fundamentais para a aquisição dos resultados. Posteriormente, o perito pode efetuar uma busca por todas as pastas e arquivos. Como os usuários normalmente armazenam seus arquivos por meio de pastas, esta técnica é fundamental para localização de evidências que interessam a investigação. Outra técnica relevante para a perícia é a adequação com que os dados são visualizados; entretanto, esta técnica só funciona com o auxilio de softwares específicos pelo que, estes programas têm como funcionalidade fazer com que o conteúdo contido nos diferentes arquivos seja visualizado em

formato certo. Durante a análise forense, alguns *softwares* são indispensáveis (ELEUTÉRIO; MACHADO, 2010).

O perito pode também usar ferramentas especiais que podem gerar prazos forenses com base em dados de eventos. Tais ferramentas normalmente fornece aos analistas forenses uma interface gráfica para visualização e análise de sequências de eventos. Uma característica comum destas ferramentas é permitir o agrupamento de eventos relacionados. Em muitos casos, a análise forense envolve não só dados de arquivos, mas também dados de outras fontes, tais como o estado operacional, o tráfego da rede ou aplicações (KENT et al, 2006, tradução nossa).

Ferramentas como o Encase e o Forensic Toolkit auxiliam bastante o perito nas pesquisas por palavras chaves, na visualização dos variados arquivos e na busca de evidências importantes em sistemas de arquivos e em pastas, o que torna a atividade pericial menos trabalhosa. Entretanto, outra técnica importante para a análise pericial é a virtualização. Esta técnica consiste na utilização de *softwares* que trabalham como máquinas virtuais; porém, aconselha-se para que não ocorram modificações no sistema a ser periciado é importante que não se ligue uma máquina apreendida com os procedimentos normais; pelo que ligar a máquina a ser investigada por meio de programas de virtualização é o procedimento mais aconselhável durante a análise pericial. Um exemplo de *software* muito usado em operações de virtualização é o famoso *VMware* (ELEUTÉRIO; MACHADO, 2010).

3.1.4 Os resultados da perícia

A obtenção dos resultados é a última etapa em uma investigação forense. O individuo ou entidade responsável pela perícia deve elaborar um documento no qual poderá constar tudo o que foi encontrado no local de investigação. Tecnicamente este documento é conhecido como laudo pericial e para que seja aceito judicialmente deve estar devidamente comprovado e documentado. O laudo é praticamente um resumo detalhado de todas as evidências e incidentes encontrados, devendo constar também o motivo pelo qual ocorreram os incidentes, a data, os responsáveis e quais métodos e técnicas foram adotadas para se averiguar os fatos, assim como toda a avaliação conclusiva em relação ao caso investigado (CAIXETA, 2011).

O perito durante este procedimento de investigação tem por objetivo principal apresentar todos os resultados, pois, as demais ações são realizadas nas fases de coleta, exame e análise. O especialista também deve mostrar quais mecanismos usou para alcançar os resultados. Informações sobre técnicas, metodologias e *softwares* são pertinentes e devem ser anexados e apresentados num relatório final – o laudo pericial. Normalmente, em casos de julgamentos, todos os acontecimentos ou ocorrências passam sempre por período de constantes interrogações; sendo assim, é necessário que se apresente todas as provas do caso sem exceção. A apresentação completa das evidências também facilita o trabalho do juiz (NG, 2007).

> O laudo pericial tem como objetivo principal oferecer e dispor para o magistrado – encarregado de julgar o caso –, e as partes do processo, todo o procedimento realizado pelo perito em questão. É este documento que atesta de forma positiva ou negativa seus passos ao longo de uma perícia (QUEIROZ; VARGAS, 2010, p. 29).

É imprescindível que este material de caráter técnico esteja correto e acima de tudo compreensível. O laudo deve ser elaborado com palavras claras, precisas e objetivas para que não haja incertezas ou má interpretação dos juízes, advogados ou outros indivíduos envolvidos no caso; visto que, se a linguagem for de difícil compreensão pode-se exigir outro laudo mais ajustável e bem escrito e, como consequência, o perito pode ser retirado do caso e sendo substituído por um profissional mais competente. Entretanto, o documento pericial também deve seguir um conjunto de regras pré-estabelecidas. No Brasil, a elaboração deste material deve seguir as indicações descritas na norma exposta pela Associação Brasileira de Normas Técnicas, dentre elas a 6022, 6023, 6024, 10520 e 14724 com foco na criação de documentos, citações e referências. Por exemplo, devem-se seguir detalhadamente os Procedimentos Operacionais Padrão (*POPs*) das organizações peritas para que não sejam comprometidos os critérios de convergência durante a elaboração do laudo; por outro lado, existe até normas estipuladas para a elaboração de imagens forenses, coleta de emails, captura de sites na Internet e construção de novos *POPs* (COSTA, 2011).

Por conseguinte, deduz-se que as informações que compõem o relatório pericial são fundamentais para o fechamento da investigação na medida em que acarreta uma série de evidências criminais importantes. Elaborado pelo perito, o laudo destaca cuidadosamente item por item coletados, examinados ou analisados

durante toda a atividade investigativa. Ele deve ser realizado com a máxima atenção possível com a finalidade de se adquirir resultados satisfatórios sem a necessidade de se fomentar dúvidas sobre a sua produção. Todas as informações contidas têm que ser verdadeiras e completas para que estejam livres de irregularidades (QUEIROZ; VARGAS, 2010).

> É fundamental que o relatório apresente a conclusão com base nas evidências encontradas. O perito deve sempre fornecer informações que possam ser utilizadas para a conclusão do processo e jamais se envolver a ponto de ter que decidir o processo ou defender alguma parte (NG, 2007, p. 100).

Normalmente o laudo pericial possui uma organização bem detalhada; entretanto, o perito deve focar-se nos seguintes itens que o compõe (ELEUTÉRIO; MACHADO, 2010):

a) a identificação;

b) o histórico: no qual são destacados acontecimentos antigos mas que são necessários e importantes para a construção final do laudo;

c) equipamentos: descrição de todos os meios utilizados durante a investigação;

d) metas: consiste em verificar se os resultados apresentados no laudo estão de acordo com os objetivos traçados durante toda a investigação forense;

e) argumentos técnico-periciais: elaboração de conceitos e informações ligadas às análises forenses feitas e que são de caráter importante para a compreensão do laudo;

f) as análises: engloba tanto a parte relativa à descrição quanto à experimentação;

g) respostas e argumentos às questões e conclusões: após os exames e aquisição dos resultados é essencial que se faça um resumo bem detalhado.

Os relatórios/laudos podem ser de três tipos, sendo que cada um contém informações singulares (NG, 2007):

a) laudo de alto nível: é o relatório elaborado no qual consta desde informações sobre incidentes, procedimentos de investigação e análise até conclusões de alto nível;

b) relatório detalhado: laudo voltado principalmente para casos concluídos, sendo que, em cada caso deverá conter também as evidências recolhidas e utilizadas;

c) laudo de recomendações: relatório feito pelo perito em caso de sugestões propostas no intuito de melhorar ou aperfeiçoar suas técnicas em futuros casos de investigação.

3.2 SOFTWARES DE PERÍCIA

Há diversas ferramentas forenses que podem ser utilizadas em uma investigação. Entretanto, os responsáveis pela perícia devem definir de antemão quais *softwares* existem e que atendem as suas necessidades. É responsabilidade destes, escolher os programas em função do tipo de caso a ser investigado. As ferramentas de perícia são comerciais ou gratuitas. Entre os principais *softwares* destacam-se: *Helix*, *EnCase*, *Autopsy*, *FCCU*, *CAINE*, *DEFT*, *PeriBr*, *Forensic Toolkit*, *Forensic Digital Toolkit – UbuntuBr* (QUEIROZ; VARGAS, 2010).

Ainda sobre ferramentas de perícia, mencionam-se outras que também são de caráter importante para a resolução de casos de investigação forense computacional. Os *softwares* são: *DumpSec v.2.8.6 Systemtools*, *Foundstone*, *Fport*, *Attacker v3.0*, *Visual Route* (NG, 2007).

Segue-se a descrição de algumas ferramentas utilizadas atualmente em vários casos de investigação forense computacional.

3.2.1 Autopsy

É um *software* de perícia que roda em sistemas *Linux*. O conjunto de ferramentas contido no *Autopsy* auxilia o perito durante as realizações de análises forenses em sistemas de arquivos NTFS, FAT, Ext2, Ext3, UFS1 e UFS2. O kit de ferramentas *Autopsy* pode ser obtido na Internet (NG, 2007).

Desenvolvido por Brian Carrier, a ferramenta *Autopsy* agrega os *softwares* numa interface gráfica que possibilita a verificação das pastas e arquivos, diretórios, armazenamento de dados, dados alocados ou excluídos presentes no sistema. Por

meio de sua interface gráfica, as imagens do sistema de arquivos do computador comprometido podem ser analisadas em maior nível de abstração.

Dentre outras funcionalidades destacam-se: pesquisa por palavras-chave, criação de linha de tempo de *mactimes* nos arquivos e diretórios. (HOLPERIN; LEOBONS, 2012).

3.2.2 Helix

Este *software* possui um conjunto de alternativas que podem ser seguidas detalhadamente durante a elaboração de uma investigação forense. Esta ferramenta é vantajosa para seus usuários, visto que, opera normalmente em ambientes operacionais *Windows* e *Linux* sem restrição de funcionamento nas distribuições destes sistemas operativos (QUEIROZ; VARGAS, 2010).

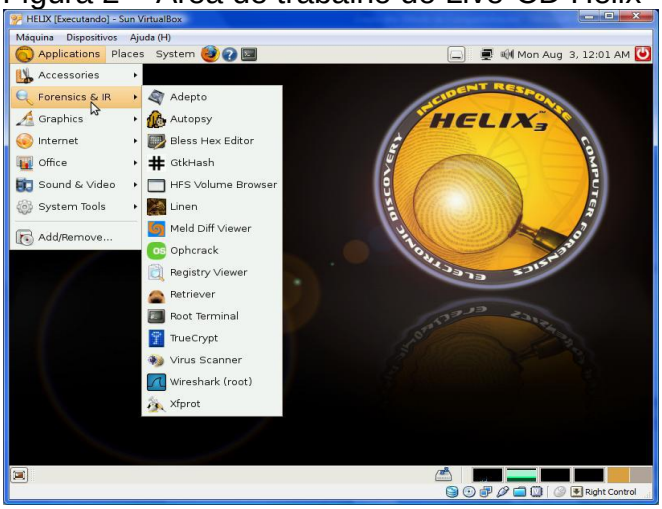
O *Helix* é uma ferramenta que reúne um conjunto de *softwares* utilizados especificamente em procedimentos de análises periciais. As ferramentas que a compõe funcionam como examinadores de arquivos, de antivírus e recuperadores de pastas em sistemas Ext2 (NG, 2007).

Até setembro de 2008, a distribuição *Helix* não visava lucro, agora existe uma nova versão desta voltada para o *Ubuntu*, sendo que obedece aos critérios de investigação forense ou perícia computacional. As distribuições atuais do *Helix* são comerciais, pelo que, é necessário a realização de um cadastro e posterior compra de um código para que se tenha acesso total ao *software* e as atualizações mais recentes (ROVER, 2012).

O *Helix* é desenvolvido e mantido pela empresa *e-fense* sendo criado baseando-se no *Ubuntu*. Tem uma função específica para *Windows* voltada a resposta de incidentes de segurança em *Windows*. O ambiente de trabalho desta ferramenta esta vinculado ao *Gnome* e possui menus personalizados e bem montados, e para a geração de relatórios usa o navegador *Firefox* como *browser* e *Open Office* (RODRIGUES, 2009).

A figura 2 apresenta uma imagem do ambiente de trabalho de um *Live-CD Helix* contendo alguns *softwares* de investigação forense.

Figura 2 – Área de trabalho do Live-CD Helix



Fonte: Rover (2012).

O *Helix* tem em torno de 150 *softwares* e suas versões atuais pagas são subdivididas em três (RAMOS; SATURNINO; FERREIRA, 2009):

a) *live response*: ferramenta voltada à obtenção de dados voláteis e histórico de páginas *web*;

b) *helix3 enterprise*: versão utilizada para proteção de redes contra códigos maliciosos e invasão de privacidade;

c) *helix3 pro*: tem uma interface intuitiva e auxilia os peritos forenses na obtenção de respostas aos incidentes de segurança, entretanto, apenas está acessível a usuários e integrantes do fórum da *e-fense*.

3.2.3 EnCase

Ferramenta tida como uma das mais usadas em processos de análises em casos de investigações forenses e desenvolvida em linguagem C++ especificamente para executar em plataforma *Windows*. Para cada sistema investigado, o *EnCase* ordena os dados em casos e registra-os em arquivos diferentes e não em forma de diretórios; sendo que os arquivos contidos nos casos são constituídos por: bloco de notas, cabeçalho e analisadores de integridade (LIMA, 2009).

A *Guidance* é uma empresa norte-americana responsável por produzir o *EnCase*. Entretanto este *software* possui vários mecanismos que ajudam o perito

durante as análises dos dados armazenados nos meios computacionais. Esta ferramenta possui as seguintes funções: a restauração de arquivos excluídos, busca por palavras-chave e a visualização dos dados em formato apropriado (ELEUTÉRIO; MACHADO, 2010).

> Entre as funcionalidades do *EnCase* pode-se citar: criação de imagens de disco bit a bit, montagem dessas imagens em modo somente leitura, verificação de integridade de dados, recuperação de arquivos removidos, visualização do arquivo de memória virtual, classificação dos arquivos de acordo com o tipo, busca de palavras-chave, pré-visualização de alguns tipos de arquivos no próprio programa, dentre outras (LIMA, 2009, p. 26).

Ela inclui funções de duplicar discos, e pode ser bastante útil para o perito durante todas as fases da análise pericial. Contudo, não possui uma interface gráfica muito intuitiva; o que obriga a seus usuários a necessidade de constantes treinamentos individuais. A ferramenta tem como ênfase a possibilidade de se projetar funções novas através de *EnScripts* (ELEUTÉRIO; MACHADO, 2010).

O *EnCase* é uma ferramenta comercial. Além de ser destinada à perícia forense de crimes cibernéticos, reúne um kit de *softwares* que possibilitam procurar as informações com maior rapidez e agilidade em uma mídia digital; sendo que, pode ser utilizado local ou remotamente no intuito de facilitar as atividades dos indivíduos responsáveis pela investigação (NG, 2007).

Este *software* é utilizado pela polícia e é considerada uma das ferramentas mais eficazes, complexa de ser utilizada e possibilita a obtenção de resultados satisfatórios e completos em uma perícia forense assim como permite também que o perito crie o laudo com sucesso depois do término da investigação (QUEIROZ; VARGAS, 2010).

3.2.4 Forensic Toolkit

É um *software* de perícia desenvolvido pela corporação *AcessData*, sendo que, pode ser obtido gratuitamente. Possui funções essenciais que auxiliam o perito nos procedimentos de análises forenses em meios de armazenamento de dados. Sua interface gráfica é clara e fácil de ser compreendida e possibilita o entendimento de maneira mais ampla de todas as informações que necessitam ser analisadas. O *Forensic Toolkit* ou simplesmente FTK é uma das ferramentas mais interessantes em casos da forense em sistemas *Linux* e possui as seguintes funcionalidades: a

indexação de dados, data carving restauração de arquivos, visualização de imagens e mensagens eletrônicas, distinção de arquivos, utilização da técnica *de Known File Filter*, de pesquisas por palavras chaves e muito mais (ELEUTÉRIO; MACHADO, 2010).

3.2.5 Forense Digital Toolkit (FDTK) – UbuntuBr

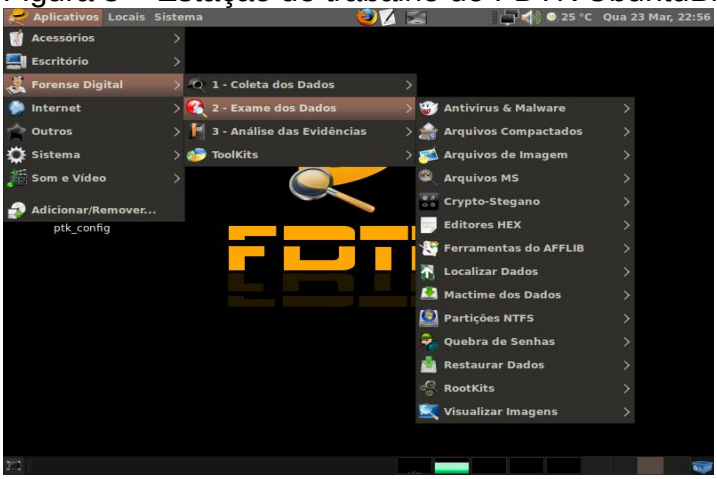
Faz parte de um conjunto de *softwares* de perícia e é uma ótima ferramenta para casos da forense em ambiente *Linux*. É um *software* livre, desenvolvido pelo professor de graduação tecnológica em segurança da informação Paulo Neukamp por meio de um trabalho de conclusão de curso, no qual definiu que a ferramenta tem como finalidades (QUEIROZ; VARGAS, 2010):

a) auxiliar os profissionais de perícia em suas atividades;

b) ser uma ferramenta voltada ao ensino no intuito de que os acadêmicos possam compreender acerca da importância da perícia forense computacional.

O *FTDK-UbuntuBr* é um projeto voltado para a perícia em sistemas *Linux* cuja meta é o desenvolvimento e manutenção de distribuições que visam analisar e recolher dados de uma investigação. Foi desenvolvido a partir do *Ubuntu* sendo que possui mais de 100 *softwares* cujas funcionalidades auxiliam os peritos durante suas atividades em todas as etapas da forense computacional. As ferramentas contidas podem ser utilizadas como um *Live-CD* instaladas em máquinas e convertendo-as em locais de perícia. O *Forense Digital Toolkit* está na sua versão 3.0 e tem sido aperfeiçoado por desenvolvedores de sistemas; porém, não se define apenas pela grande quantidade de ferramentas, mas acima de tudo por possuir uma interface intuitiva e organizada de acordo com os procedimentos de investigação forense e também por estar disponível em português (NEUKAMP, 2008).

Figura 3 – Estação de trabalho do FDTK UbuntuBr



Fonte: Neukamp (2008).

3.2.6 Federal Computer Crime Unit (FCCU)

Este *software* é uma distribuição *GNU/Linux* e possui um *kit* de ferramentas destinadas à análise forense de dados em discos rígidos, memória principal, rede, quebra de senhas, detecção de códigos maliciosos dentre outras funcionalidades. São mais de 170 ferramentas incluídas no *Live-CD* do FCCU. Fundamentado no *Debian Live Projet*. Está agora na sua versão 12.1 e possui uma interface gráfica intuitiva e clara que facilita aos usuários de forma a utilizar adequadamente a ferramenta. O *software* é livre. Tem de negativo o fato de que o *Live-CD* vem com o teclado configurado no padrão Belga e também a inexistência do *Firefox* (RAMOS; SATURNINO; FERREIRA, 2009).

Na figura 4 tem-se o ambiente de trabalho do FCCU.

Figura 4 – Ambiente de trabalho do FCCU

Fonte: Ramos, Saturnino e Ferreira (2009).

3.2.7 Computer Aided Investigate Environment (CAINE)

É um *kit* de ferramentas utilizado para a investigação forense de crimes digitais e desenvolvido com base no *Ubuntu*. Criado em Itália no ano de 2008 pelo desenvolvedor de *software* Giancarlo Giustini e destinado a atender as necessidades do Centro de Investigação sobre Segurança deste mesmo país. O programa de elaboração do *CAINE* teve ajuda também da universidade de Modela na Itália (RAMOS; SATURNINO; FERREIRA, 2009).

Pensando em seus usuários, foi desenvolvido com uma interface clara e aceitável visto que fornece uma lista de *softwares* específicos que atendam as necessidades da forense computacional. É pertinente lembrar que as ferramentas *CAINE* servem tanto para *Linux* como *Windows* e são de código aberto o que possibilita que seus usuários analisem seu código fonte e tenham o poder de alterá-lo de acordo com suas necessidades. Este conjunto de *softwares* pode ser também utilizado como um *Live-CD*. As primeiras versões do *CAINE* tinham a configuração do teclado em italiano, porém as recentes versões deste já possuem o teclado ajustado ao inglês americano (RAMOS; SATURNINO; FERREIRA, 2009).

A distribuição do *CAINE* é voltada especificamente em *Linux*. Além de ser utilizada para solucionar casos da forense computacional em geral, também é focada em situações que envolvem segurança de dados e informações. Possui uma interface clara e simples de ser compreendida por seus usuários. Os *softwares* são utilizados em procedimentos simples e avançados que envolvem perícia forense (BASTO, 2012).

Com o propósito de auxiliar os peritos forenses durante a realização das suas atividades no intuito de obter os resultados da investigação com simplicidade e eficiência, o projeto de implementação e aperfeiçoamento das ferramentas que compõe o *CAINE* possui as seguintes finalidades (RAMOS; SATURNINO; FERREIRA, 2009):

a) ajudar os profissionais em perícia durante todas as etapas de investigação forense computacional;

b) fornecer uma interface gráfica simples e agradável;

c) fornecer resultados confiáveis que auxilie o perito na construção do laudo pericial.

É um dos poucos *Live-CD* que pode ser instalado e utilizado em um computador como um ambiente operativo. Tem aproximadamente 80 *softwares* de perícia forense voltada à computação com especial realce para os programas de *Office* (RAMOS; SATURNINO; FERREIRA, 2009).

3.2.8 Digital Evidence & Forensic Toolkit (DEFT)

Distribuição voltada ao *Linux*, utilizada em casos específicos de segurança computacional e perícia forense. Possui uma ótima coleção de *softwares* destinados para a elaboração de análise periciais, inteligência cibernética e execução de testes (BASTO, 2012).

Projeto desenvolvido por Stefano Fratepietro, Andrea Ghirardini e Massimiliano Dal Cero. É um *Live-CD GNU/Linux*. Está na sua versão 4.2 e possui em torno de 40 *softwares*. A distribuição *DEFT* também pode ser adquirida gratuitamente. Entretanto, possui também ferramentas reservadas para resposta a incidentes em sistemas *Windows*, sendo 32 na sua totalidade (RAMOS; SATURNINO; FERREIRA, 2009).

Inúmeras ferramentas fazem parte do *DEFT* e possuem as seguintes funcionalidades (BASTO, 2012):

a) analisar aplicações baseadas na *web*;

b) copiar discos;

c) restaurar pastas e arquivos;

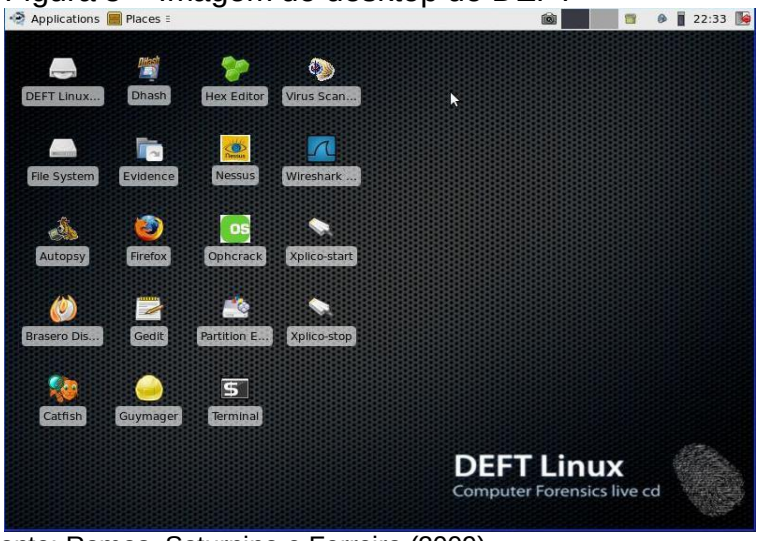
d) proteger identidade;

e) encontrar informações de rede sem exclusividade;

f) capturar informações dentro de redes sociais.

A figura 5 apresenta uma representação visual do ambiente de trabalho do *DEFT*.

Figura 5 – Imagem do desktop do DEFT



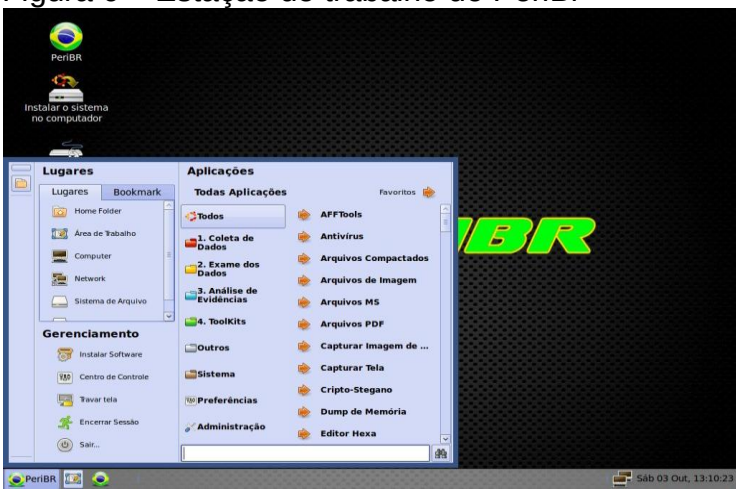
Fonte: Ramos, Saturnino e Ferreira (2009).

3.2.9 PeriBr

É um projeto novo sobre perícia computacional baseado no *Ubuntu 9.4* e possui uma interface intuitiva baseada no *Gnome*. O *PeriBr* é um *Live-CD* rico em *softwares*, possui ao todo 170 ferramentas destinadas a auxiliar o perito nas etapas de coleta, exame e análise dos dados armazenados em dispositivos físicos e em rede. Este *kit* de ferramentas voltado para *Linux* tem um menu categórico das etapas de investigação forense, o que possibilita os usuários a encontrarem quais *softwares* pretendem utilizar e que atendam as suas necessidades. O *PeriBr* foi concebido por meio da elaboração de trabalho de pós-graduação da Universidade Católica de Brasília. É também uma distribuição *open source* (RAMOS; SATURNINO; FERREIRA, 2009).

A figura 6 apresenta a área de trabalho da distribuição PeriBr.

Figura 6 – Estação de trabalho do PeriBr



Fonte: Ramos, Saturnino e Ferreira (2009).

3.3 COMPARAÇÕES ENVOLVENDO FERRAMENTAS FORENSES

A tabela 2 apresenta as diferenças e semelhanças das características funcionais de alguns *softwares* de perícia assim como as vantagens e desvantagens de cada.

Tabela 2 – Quadro comparativo entre softwares

Ferramentas	Vantagens	Desvantagens	Catacterísticas
HELIX	- Boa documentação - Interface resposta a Incidentes	- Software comercial	- Faz tudo ao seu alcance para evitar qualquer alteração no sistema quando inicializado - Nenhum arquivo ou sistema swap é automaticamente montado
FDTK	- Praticamente possui todas as ferramentas acessíveis pelo menu - Continuidade de manutenção com a versão 2.0	- Faltam softwares de forense em rede - Teclado brasileiro (ABNT2) como default	- Ótima para usuários de países de língua portuguesa
FCCU	- Diversas atualizações por ano - Acompanha constante-	- Ausência do Firefox - Teclado confi-	- Criar cópias de imagens de dispositivos

	mente a evolução da forense computacional, lançamento de novos softwares. - Diversidade de utilitários	gurado para o idioma belga - Menu meio pobre de opções	antes de analisar
CAINE	- Produto em desenvolvimento, tem recebido atualizações frequentemente - Não usa swap nem monta o drive automaticamente	- Teclado configurado para o formato italiano - Pouca documentação - Ainda não oferece - muito a ponto de ser escolhido para trabalhos mais sérios	- Gera relatórios automaticamente
DEFT	- Atualizado com frequência - Excelente documentação	- Faltam algumas ferramentas essenciais para análise (memória, browser, registry)	- Possui ferramentas exclusivas (Dhasg, Xplico e Catfish)
PeriBr	- Quantidade enorme de softwares - Menu organizado e De fácil manuseio	- Possui teclado brasileiro (ABNT2) como default - Falta de ferramenta de monitoramento de de rede	- Tem diversas ferramentas para Windows

Fonte: Adaptado de Ramos, Saturnino e Ferreira (2009).

3.4 TIPOS DE INVESTIGAÇÃO FORENSE

Em casos de perícia forense computacional, existe a alternativa de o perito iniciar a investigação dos fatos ainda com a máquina ligada. Esta forma de perícia é conhecida como investigação forense *in vivo* ou *live analysis*, e baseia-se principalmente na recolha de dados importantes tidos como voláteis que, desapareceriam se o equipamento estivesse desligado (MELO, 2009).

A forense *in vivo* resume-se num conjunto de métodos adotados durante a obtenção dos dados quando o sistema não está desligado. Neste tipo de investigação, o profissional em perícia tem como prioridade coletar informações com maior grau de volatilidade, porém, em muitos casos os dados mais voláteis perdem-

se em função do desligamento do computador ou outra máquina. Ao contrário da *live analysis*, a perícia *post mortem* obtem as informações depois da máquina ou sistema ser desligado (SANTOS, 2008).

Três formas de perícia forense podem ser utilizadas e escolhidas pelo perito no intuito de realizar uma investigação. São classificados em: forense *in vivo*, forense em rede e forense *post mortem*. Entretanto, para que as provas digitais sejam preservadas pelo perito com o propósito de serem bem documentadas, comprovadas e aceites judicialmente, é fundamental não se efetuar uma análise a partir das evidências originais, mas em cópias delas, exceto em análises *in vivo* (MACEDO, 2010).

3.4.1 Investigação forense in vivo (live analysis)

Para que a perícia *in vivo* seja realizada com eficiência e sem a destruição de evidências, o especialista em perícia pode adotar a utilização de s*oftwares* de investigação baseados num *Live-CD*. Ao realizar-se o procedimento da forense *in vivo*, é também essencial que se faça uma cópia integral do sistema a ser periciado, desde as informações e dados inerentes à volatilidade até ao disco rígido. É importante salientar que neste procedimento de investigação são obtidas evidências interessantes que contribuem para a construção do caso (TEIXEIRA, 2010).

Dentre as informações voláteis que podem ser perdidas em função do desligamento da máquina destacam-se: aquisição de dados de memória, processos ativos, documentos abertos, e conexões de rede. Por outro lado, durante uma forense *in vivo* o perito não deve aplicar *os softwares* já instalados no computador no intuito de coletar e analisar o sistema operativo, pois estes programas provavelmente foram alterados pelo invasor com a finalidade de ocultarem as ações criminais. O profissional em perícia deve estar bem equipado com suas ferramentas específicas e próprias para se efetuar uma investigação forense com sucesso (MACEDO, 2010).

Durante a *live analysis*, o perito deve capturar os dados de acordo com a ordem de volatilidade destes. A ordem de volatilidade (OOV) resume-se no tempo de vida que os dados permanecem no sistema (FARMER; VENEMA, 2007).

A tabela a seguir apresenta um modelo de representação do tempo de vida de diferentes tipos de dados armazenados de acordo com a ordem de volatilidade.

Tabela 3 – Ciclo de vida dos dados

Tipos de Dados	Tempo de vida
Registradores, memória periférica, caches	Nanossegundos
Memória principal	Dez nanossegundos
Estado da rede	Milissegundos
Processos em execução	Segundos
Disco	Minutos
Disquetes, CD-ROMs e mídia de backup	Anos

Fonte: Farmer e Venema (2007).

> O processo da perícia in vivo deve ser cuidadosamente documentado, pois a captura de informações irá interagir de alguma forma com o sistema sendo analisado, seja criando, modificando e/ou apagando dados do sistema e/ou apresentando informações precisas (MACEDO, 2010, p. 19).

Se o perito ao efetuar a investigação forense apenas achar conveniente fazê-la com base em informações e dados armazenados em disco e modificados há bastante tempo, é desnecessário que adote o procedimento de forense *in vivo*, sendo preciso realizar a perícia *post mortem* (SANTOS, 2008).

3.4.2 Investigação forense em rede (network forensic)

Neste procedimento de perícia forense, o perito deve focar-se em todos os equipamentos básicos e conexões de rede que possuem informações relevantes sobre o caso investigado. Neste processo também existe a possibilidade de se restaurar conexões de rede cortadas com o propósito de se recuperar transmissões de informações que deverão ser previamente analisadas durante a forense *post mortem*. Porém a forense em rede só é feita se as máquinas a serem investigadas estiverem num ambiente de rede seja ele corporativo ou doméstico (TEIXEIRA, 2010).

> Por meio do tráfego de rede é possível analisar toda a comunicação entre a máquina atacante e a que foi invadida, tornando possível estabelecer uma sequência de eventos que poderá ser comparada com as outras evidências encontradas ao longo da perícia (SANTOS, 2008, p. 36).

A investigação forense de rede tem como finalidade recolher dados e informações sobre o caso. Em algumas perícias é bastante difícil recolher evidências precisas quando o invasor utilizou *malwares* e artifícios baseados em criptografia para esconder suas atividades no sistema (MELO, 2009).

3.4.3 Investigação forense post mortem (post mortem forensic)

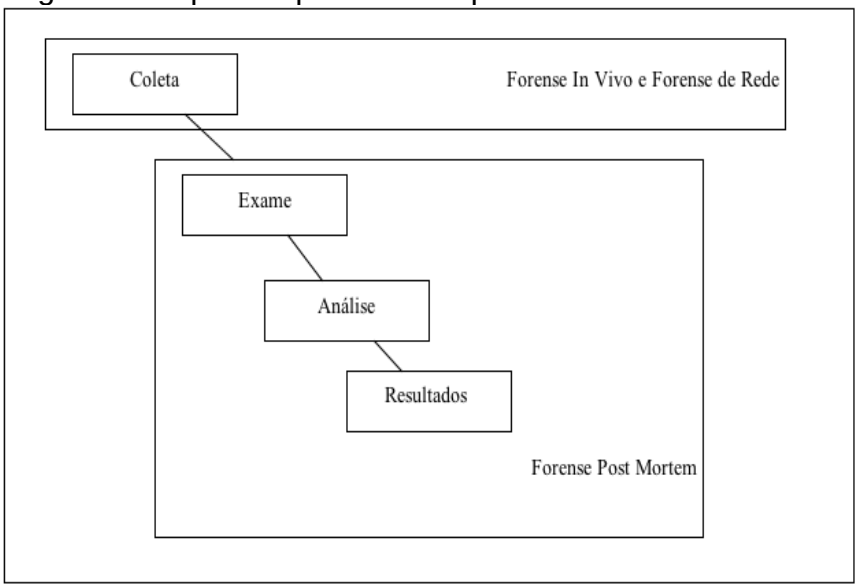
No procedimento de perícia *post mortem*, deve-se trabalhar com o computador desligado. Surge a necessidade também de se criar uma imagem da mídia periciada de maneira que a análise dos arquivos, da extensão *swap*, das informações ocultas e arquivos excluídos sejam realizada com segurança e com o propósito de não se destruir evidências importantes (MACEDO, 2010).

Porém, em função da ordem de volatilidade dos dados descrita anteriormente, a forense *post mortem* é realizada apenas com a máquina ou sistema desligado; entretanto neste instante trabalha-se única e exclusivamente com dispositivos de armazenamento não voláteis para análise dos dados e/ou informações. Estes dispositivos podem ser discos rígidos e CD-ROMs (SANTOS, 2008).

Além da coleta dos dados, a *post mortem forensic* abrange também outras etapas de investigação: o exame, a análise e a aquisição dos resultados. Este tipo de perícia fundamenta-se principalmente na realização de análises em todos os conteúdos coletados no decorrer da forense *in vivo* e em rede (TEIXEIRA, 2010).

A figura 7 ilustra como os três tipos de investigação forense se associam e se enquadram com as diferentes etapas de perícia.

Figura 7 – Tipos de perícia e etapas



Fonte: Adaptado de Teixeira (2010).

É importante lembrar-se sempre que se o perito optar em realizar a forense *post mortem* é necessário que não se ligue o computador visto que informações e dados podem ser alterados e perdidos durante a inicialização da máquina. A perda de informações deve-se pelo fato de os *softwares* de gerenciamento do *hardware* realizarem limpezas constantes em função das ações de ligar e desligar o computador e também por ações que envolvem a implantação de vírus por parte dos criminosos. É imprescindível também concebermos que os sistemas operacionais diferenciam-se nos seus procedimentos de desligamento e inicialização, pelo que, é importante que o perito conheça o funcionamento do sistema operacional a ser investigado (MACEDO, 2010).

A perícia *post mortem* é tida como a mais trabalhosa visto que leva muito tempo para ser completada. O tempo demorado para a concretização desta perícia deve-se pela grande quantidade de informações que necessitam ser analisadas. Entretanto, neste processo recomenda-se que a aquisição dos dados deve ser realizada num computador diferente, preservando neste caso o disco rígido original. O profissional deve criar uma cópia do HD do computador suspeito (SANTOS, 2008).

Para a realização de uma cópia do disco rígido para uma investigação *post mortem* é necessário seguirem-se as seguintes etapas (MACEDO, 2010):

a) retirar o disco rígido da máquina comprometida;

b) ligar o HD no computador destinado para análise mas usando *hardware* ou *softwares* que bloqueiam escrita;

c) efetuar um *hash* do HD por meio de máquinas ou programas;

d) criar uma imagem do disco rígido utilizando ferramentas específicas;

e) fazer um *hash* da imagem por meio de *softwares*;

f) fazer comparações entre o *hash* criado a partir do HD com o *hash* com o *hash* obtido da imagem;

g) elaborar um *hash* composto por todos os arquivos pertencentes ao disco rígido;

h) criar proteção no HD, em todos os *hash* obtidos e guardá-los em ambientes seguros;

i) a partir desta etapa toda a perícia *post mortem* é realizada por meio de uma cópia do disco rígido.

O capítulo quatro trata sobre os *Rootkits* e sistema operacional *Linux*. A princípio, tem uma simples descrição referente ao ambiente *Linux*. Posteriormente é descrita a parte dos *Rootkits* onde aparecem os seus conceitos, os tipos mais comuns e alguns procedimentos essenciais que visam a proteção e detecção destas ferramentas maliciosos que tornam os ambientes operacionais mais inseguros.

4 SISTEMA OPERACIONAL LINUX E ROOTKITS

Atualmente os peritos forenses têm utilizado com bastante frequência os ambientes *Linux* no intuito de realizarem suas atividades investigativas. Este sistema é eficiente e vantajoso na medida em que oferece diversas soluções para a forense computacional. A utilização do *Linux* em casos envolvendo perícia forense de crimes digitais tem os seus benefícios. Um dos pontos vantajosos deste ambiente operacional é o fato de existir a possibilidade de criarem-se unidades de dados acessados apenas em modo de leitura, sendo que não necessita de um bloqueador independente para que as informações e/ou dados armazenados em disco não sejam modificados. Por outro lado pode-se afirmar que é um sistema ajustável que possui diversos sistemas de arquivos próprios para serem executados em projetos de perícia forense. Existem diversas distribuições *Linux* e várias ferramentas de perícia para esse sistema operacional (REYES; WILES, 2007, tradução nossa).

4.1 ROOTKITS

Esta-se na era da informação por meio da qual não se deve confiar 100% em computadores. Por exemplo, como é que tem-se certeza que o nosso fabricante de *hardware* não escondeu código malicioso no *microchip* do sistema? Ou que o nosso sistema operacional recém-instalado não contém *backdoors* criados por um desenvolvedor desonesto da equipe de programação? O fato é que não se pode confiar plenamente que nossos computadores estão totalmente livres de *malwares* (METULA, 2011, tradução nossa).

Infelizmente a necessidade de usar-se um computador supera a falta de confiança a este respeito. De salientar que, *malware* é um pedaço de *software* projetado para realizar atividades maliciosas na máquina da vítima sem o consentimento desta. *Malware* é uma expressão utilizada computacionalmente para especificar *softwares* mal intencionados tais como vírus, cavalos de Tróia, *backdoors*, *worms* e *Rootkits*. Basicamente qualquer tipo de código projetado para causar dano ou espionar as atividades de uma vítima. Uma vez que o *malware* é instalado, a intenção do atacante é permanecer despercebido tanto tempo quanto possível, mantendo o controle do sistema (METULA, 2011, tradução nossa).

*Rootki*t é conceituado como sendo um *kit* de ferramentas utilizado por *hackers* e *crackers* com objetivo obter acesso a computadores de forma ilegal. Esses *softwares* são utilizados no intuito ocultar arquivos e processos, o que possibilita aos invasores à permanência de conexão ao sistema, e praticando atividades criminosas. Existem *Rootkits* para diversos ambientes operacionais com destaque para *Linux*, *Solaris* e *Microsoft Windows*. Dentre as funcionalidades dos *Rootkits* destacam-se (AVILA, 2010, tradução nossa):

a) ocultação de outros *softwares* em execução no sistema;

b) possibilita a ocultação de arquivos armazenados e processos em execução;

c) permite a ocultação de diretórios importantes do sistema;

d) ocultação de portas que possibilita o atacante acessar determinado computador por meio da Internet.

Seguindo Macedo (2010) os *Rootkits* escondem-se dos usuários e gerentes de sistemas no intuito de que não haja suspeita de invasores infiltrados que posteriormente poderão ter permissão de r*oot*. Ainda segundo o autor, os *Rootkits* ocultam informações relevantes no sistema, destacando-se:

a) arquivos de configurações do sistema;

b) *softwares*, arquivos e portas de segurança;

c) procedimentos e conexões estabelecidas pelos atacantes;

d) acesso de logs realizados durante as ações dos invasores.

Os *Rootkits* são geralmente utilizados para ocultação de diversas aplicações que estejam em funcionamento no sistema comprometido. Eles normalmente englobam *backdoors* que auxiliam o invasor para acessos fáceis no sistema, uma vez que obteve sucesso durante a primeira tentativa de acesso. Por exemplo, este tipo de *malware* pode ocultar um dado aplicativo sempre que o atacante se conectar ao sistema através de uma porta. Por outro lado, um tipo de código malicioso utilizado para obter informações de forma ilegal, pode ser escondido por *Rootkits* (AVILA, 2010, tradução nossa).

4.1.1 Tipos: nível de usuário e kernel

Existem dois tipos comuns de *Rootkits*: em nível de usuário e de *kernel* (MACEDO, 2010).

Os *Rootkits* em nível de usuário são executados em sistemas e contexto de segurança de um usuário no sistema. Por exemplo, se alguém estiver logado em sua estação de trabalho como um usuário qualquer e não tiver permissões administrativas, o *Rootkit* irá filtrar e dar acesso a determinadas portas para todas as aplicações em execução na conta. Normalmente a maioria das contas de usuário também têm privilégios administrativos, sendo assim, um *Rootkit* em modo usuário oculta as suas ações no sistema operacional e pode impedir o funcionamento de alguns processos em execução (DAVIS; BODMER; LEMASTERS, 2010, tradução nossa).

Estes *Rootkits* podem também ser chamados de *user level*, *user mode* ou *application level*. São os tipos mais simples e são classificados em *Rootkits* binários e de biblioteca. Os binários caracterizam-se por fazer parte das funcionalidades dos outros *softwares* instalados; já os *rootkits* de biblioteca desempenham o papel de substitutos das bibliotecas mais importantes do ambiente operativo (MACEDO, 2010).

Rootkits de biblioteca são os mais comuns em sistemas operacionais do tipo *Unix*. Eles filtram aplicaçõs e provavelmente são mais eficazes do que os *Rootkits* em modo de usuário, mas não tão eficientes ou de difícil remoção como *Rootkits* em modo *kernel* a ser descrito a seguir (DAVIS; BODMER; LEMASTERS, 2010, tradução nossa).

Chamados de *kernel level* ou *kernel mode*, alteraram e escondem dados e/ou informações a partir do núcleo do sistema. São códigos malignos executados automaticamente no núcleo do sistema cuja funcionalidade principal é alterar a organização de todo o conteúdo do *kernel* a partir da memória principal (MACEDO, 2010).

Rootkits em modo de *kernel* operam dentro do sistema operacional no mesmo nível como as unidades de *hardware*, como placa gráfica, ou mouse. A instalação de um *Rootkit* para uso dentro do *kernel* de um sistema operacional é muito mais difícil do que instalar um *Rootkit* modo de usuário e requer uma

habilidade muito maior definido a partir do atacante para implementar. Além disso, uma vez que muitos sistemas operacionais alteram partes do seu *kernel* com atualizações e novas versões, *Rootkits* de *kernel* nem sempre poderão funcionar para todas as versões. A maioria das vezes descobre-se a presença de *Rootkits* no seu sistema quando o desempenho do núcleo diminui e como consequência, é notável o surgimento erros de reinicialização instantânea do sistema operacional (DAVIS; BODMER; LEMASTERS, 2010, tradução nossa).

4.1.2 Prevenção

Uma das estratégias que os usuários e administradores de sistemas têm adotado como defesa contra os famosos *Rootkits* é prevenir-se destes, pois, após o acesso não autorizado por meio de *Rootkits*, o sistema operacional torna-se mais vulnerável ou inseguro (MACEDO, 2010).

Se o sistema estiver infectado, o que fazer? Porém pode-se remover *Rootkits*, apesar de ser uma tarefa difícil de ser concretizada. A remoção destes é feita com sucesso por meio da utilização de ferramentas conhecidas como anti-*Rootkits*. Estas são adquiridas gratuitamente na Internet (AVILA, 2010, tradução nossa).

É fundamental salientar-se que não há ambientes de segurança ou *softwares* de proteção de sistemas totalmente seguros. Nem sempre os resultados apresentados por determinado *software anti-rootkit* são satisfatórios, podendo estes ser modificados pelos próprios *Rootkits*. *Rootkits* podem interferir em análises realizadas por ferramentas *anti-rootkits* de maneira que os resultados alcançados durante a análise do sistema apesar de falsos, ser vistos como verídicos em função da infiltração dos *Rootkits*. Sendo assim há que se ter conhecimento necessário sobre o funcionamento dos *Rootkits* e das ferramentas *anti-rootkits* (MACEDO, 2010).

A primeira forma de prevenção contra os *Rootkits* é a não utilização destes, o que indica a possibilidade de evitarem-se acessos indevidos no sistema. A não utilização dos *Rootkits* refere-se ao fato destes *malwares* instalarem-se acidentalmente através de *websites* suspeitos e inseguros com presença desses programas maliciosos. Recomenda-se também a adoção de procedimentos que

envolve configuração segura e manutenção constante dos sistemas operacionais. Outras formas de prevenção seguidas são (MACEDO, 2010):

a) devem-se desativar os processos e *softwares* desnecessários que estão sendo executados no sistema;

b) controles especializados: consiste na utilização de *firewall*, antivírus, *anti-malwares* e *anti-rootkits* no intuito de que se alguma destas ferramentas for monitorada por *Rootkits*, outras poderão travar o ataque;

c) execução constante de testes no sistema em busca de agentes intrusos que comprometem a segurança do sistema.

4.1.3 Detecção

Quando tem-se casos de intrusão de sistemas por meio de *Rootkits*, a melhor maneira de detê-los é através da utilização de ferramentas *anti-rootkits*; com especial destaque para os *softwares chkrootkit* e o *rootkit hunter*. Entretanto esses dois programas são os mais usados em ambientes *Linux* e são gratuitos (MACEDO, 2010).

> O chkrootkit é uma ferramenta desenvolvida por Nelson Murilo e Klaus Steding Jessen capaz de identificar uma série de rootkits e sinais de invasão, como a retirada de entradas no registro wtmp do sistema. O “Rootkit Hunter”, conhecido popularmente como “rkhunter”, é uma ferramenta desenvolvida por Michael Boelen capaz de identificar rootkits, backdoors e sniffers (OLIVEIRA, 2005, p. 6).

O próximo capítulo aborda os trabalhos correlatos. São trabalhos sobre investigação forense computacional já elaborados e que têm interligação com este trabalho de pesquisa, desde os conceitos que envolve a perícia até as metodologias, técnicas e *softwares* destinados a auxiliar os pesquisadores e profissionais desta área de atuação.

5 TRABALHOS CORRELATOS

Há um crescimento acelerado dos crimes digitais que estão afetando a sociedade nos dias de hoje. Os mais variados recursos computacionais utlizados diariamente têm os seus benefícios, mas também os perigos. Apesar de a sociedade estar rodeada de dispositivos e tecnologias avançadas, destacando a Internet, os *smartphones*, *tablets*, celulares, GPS e câmeras digitais, é de realçar que ainda não se atingiu o nível adequado de segurança com base em intrusões e crimes computacionais que têm aumentado gradualmente na medida em que criminosos adotam novas técnicas de invasão; procedimentos estes que envolve utilização, criação, propagação e disseminação de *malwares*. Entretanto, a área da forense computacional tem efetuado pesquisas e desenvolvimento de *softwares* capazes de desvendar as atividades ilícitas praticadas nos diversos meios computacionais. Sendo assim, neste capítulo serão mencionados alguns trabalhos sobre investigação forense computacional.

5.1 INVESTIGAÇÃO FORENSE DIGITAL DE ROOTKITS EM SISTEMAS UNIX

Trabalho de graduação apresentado ao curso de ciências da computação da Universidade Federal do Rio Grande do Sul, e realizado em 2010. Este trabalho tem como objetivo apresentar metodologias importantes de perícia com o objetivo de ajudar as equipes de investigação forense em incidentes que envolvem a presença de *Rootkits* em sistemas *Unix*. No decorrer do trabalho, são detalhados os conceitos de perícia forense digital, as etapas de perícia, os dois tipos mais comuns de perícia (*in vivo* e *post mortem*) e posteriormente faz uma descrição detalhada sobre os *Rootkits* abordando os tipos destes, os perigos que causam nos sistemas operativos assim como os procedimentos mínimos que se deve adotar para prevenção e detecção (MACEDO, 2010).

5.2 UMA VISÃO FORENSE DOS ROOTKITS EM SISTEMAS LINUX

Monografia de pós-graduação apresentada no departamento de ciências

da computação, como requisito para a obtenção do título de Especialista em Administração em Redes *Linux*, realizado em 2005 na Universidade Federal de Lavras em Minas Gerais. Este trabalho de pesquisa tem como foco introduzir uma abordagem específica sobre a perícia forense de invasão em ambientes computacionais *Linux*. O trabalho objetiva proporcionar argumentos detalhados sobre os diferentes tipos de intrusões em sistemas envolvendo o uso de *Rootkits*. Durante o trabalho, são abordados conceitos sobre *Rootkits*, a classificação destes, é realizada uma análise de comportamento do *Rootkit Adore-ng* que é um dos mais conhecidos para ambiente Linux. São listados e descritos também um conjunto de diretórios pertences ao sistema operacional *Linu*x e apresentado um arquivo de logs das ferramentas *anti-rootkit chkrootkit* e *determine* usadas para a detecção do *rootkit Adore-ng* (TEIXEIRA, 2005).

5.3 COMPUTAÇÃO FORENSE EM SISTEMAS GNU/LINUX

Monografia de pós-graduação apresentada ao departamento de ciências da computação da Universidade federal de Lavras para a obtenção do título de Especialista em Administração em Redes *Linux*, pesquisa elaborada em 2008. O trabalho visa principalmente apresentar procedimentos de análise inerentes à forense computacional em ambientes *GNU/Linux*. O pesquisador definiu como meta a demonstração das estratégias e metodologias que visam facilitar o trabalho dos investigadores forenses durante as etapas de coleta, análise e exame realizadas em sistemas operacionais *Linux*. Conceitos e procedimentos da forense *in vivo*, *post mortem* e de rede também são descritos nesta monografia (SANTOS, 2008).

5.4 3AP.BR – UM NOVO CONCEITO DE LIVE CD PARA FORENSE COMPUTACIONAL

Este é um projeto de conclusão de curso apresentado na Faculdade de Tecnologia SNAI de Desenvolvimento Gerencial – FATESG de Goiânia, para a obtenção do título de Graduado em Tecnologia em Redes de Computadores, sendo que, o trabalho foi concluído em 2009.

Este trabalho descreve a perícia forense computacional com ênfase na

coleta das provas digitais, na aquisição dos resultados e também legislação focada aos *cyber* crimes. A partir destes conceitos relevantes, estão descritos diversos *kit* de ferramentas periciais que são usados como *Live-CD*. Diversos *Live-CDs* com instruções sobre suas funções, licença de uso e quais sistemas operacionais são executados estão descritos neste trabalho no intuito de facilitar os peritos forenses durante a busca e realização de análises forenses em sistemas, dispositivos e meios de armazenamento computacionais (RAMOS; SATURNINO; FERREIRA, 2009).

O próximo capítulo é sobre o trabalho proposto. Nele, são definidos os *Rootkits* utilizados durante os testes, os ambientes de teste, as ferramentas utilizadas para a detecção dos *malwares* e em sequência são descritos os resultados obtidos e a conclusão.

6 ANÁLISE COMPORTAMENTAL DOS ROOTKITS FUCKIT E XZIBIT EM AMBIENTE LINUX COM UTILIZAÇÃO DE SOFTWARES DE DETECÇÃO

Rootkits são ferramentas que após instaladas no sistema, têm o poder de esconderem-se em arquivos e diretórios. A funcionalidade destes depende essencialmente da maneira como foi criado para atuar em determinado ambiente operacional. Entretanto, também podem instalar *keyloggers* e *backdoors* que, facilitam acesso irrestrito de usuários. Por outro lado, *Rootkits* são normalmente instalados por usuários com privilégios administrativos do sistema ou da instalação acidental de *softwares* e/ou *drivers* de dispositivo com *trojans* compilados em suas configurações; e, essas ferramentas maliciosas tendem a manter controle do sistema (WMLUG, 2009, tradução nossa).

Segundo Teixeira (2005), *Rootkits* alteram o funcionamento de um sistema operacional de maneira que os invasores tenham maior facilidade durante a realização de suas atividades.

6.1 PROPOSTA DE UM AMBIENTE DE TESTE

Com a finalidade de obterem-se informações sobre o comportamento dos *Rootkits* em ambiente *Linux*, foram criadas máquinas virtuais com a ferramenta de virtualização *VMware*. As máquinas virtuais foram instaladas utilizando a distribuição *Ubuntu 11.10*.

De maneira mais abrangente, máquinas virtuais são ferramentas cuja finalidade é estabelecer uma interligação entre o usuário e a máquina; entretanto é possível criar-se um ou vários novos sistemas computacionais com a utilização desta tecnologia. Estes sistemas são tidos como ambientes controlados por meio dos quais os diversos elementos físicos do computador são simulados (TEIXEIRA, 2005).

Pode-se conceber ainda que as máquinas virtuais visem criar cópias do sistema a ser periciado, sendo que, estas encontram-se isoladas do computador real que executa a máquina virtual. O isolamento entre as diversas máquinas é vantajoso na medida em que o mau funcionamento e/ou falha crítica de uma não afeta o desempenho das restantes máquinas. É ainda imprescindível afirmar que nestes

ambientes todas as máquinas virtuais criadas estão interligadas a apenas uma máquina virtual (neste caso o *VMware* utilizado durante os testes).

Inseriu-se no *VMware* as ferramentas *Backtrack 5*, *PeriBr*, *FDTK-Ubuntu-Br* e o *Ubuntu 11.10*; todas elas em formato *ISO*. A partir do terminal *Linux* das máquinas virtuais criadas foi instalado os *Rootkits fuckit* e *xzibit*. Após a instalação destes *malwares*, verificaram-se os ambientes virtuais no intuito de: identificar a presença dos *Rootkits* e buscar sinais deixados por estes no sistema. A verificação e/ou checagem foi feita com os *softwares chkrootkit* e *Rootkit hunter*. As ferramentas instaladas e testadas a partir do *VMware* bem como as suas funcionalidades no sistema estão descritas com maior ênfase nas seções posteriores.

A Tabela 4 representada abaixo, lista diversos diretórios *Linux* que podem ser alterados com a presença dos *Rootkits*.

Tabela 4 – Diretórios do Linux

Diretórios do Linux e sua descrição
/ : É o diretório raiz, todos os demais diretórios estão abaixo dele.
/bin : Contém arquivos programas do sistema que são usados com freqüência pelos usuários.
/boot : Arquivos estáticos e gerenciador de inicialização.
/dev : Arquivos de dispositivos (periféricos).
/etc : Arquivos de configuração do sistema, específicos da máquin a.
/home : Contém os diretórios dos usuários.
/lib : Bibliotecas essenciais compartilhadas e módulos do kernel.
/mnt : Ponto de montagem para montar um sistema de arquivos temporariamente.
/proc : Diretório virtual de informações do sistema .
/root : Diretório home do usuário root.
/sbin : Diretório de programas usados pelo superusuário root, para administração e controle do funcionamento do sistema.
/tmp : Arquivos temporários.

/usr : Contém a maior parte de seus programas. Normalmente acessível somente como leitura

Fonte: Adaptado de Teixeira (2005).

6.2 ROOTKIT FUCKIT

Sendo necessário criar máquinas virtuais que serviram de suporte para os testes envolvendo a utilização de *Rootkits*, efetuou-se o *download* de uma versão operacional do *Rootkit fuckit* (versão 0.4). Este inicialmente foi instalado e testado na máquina virtual *Ubuntu* 11.10 e posteriormente nas restantes máquinas.

Dentre as funcionalidades mais destacáveis do *Rootkit fuckit* após ser instalado no sistema, destaca-se que o mesmo faz com que o diretório do qual os seus binários estão localizados esteja invisível e inacessível. Por meio do comando ls executado a partir do terminal, não irá mostrar mais esse diretório, sendo que, dificulta mais o trabalho pericial. Este é um *Rootkit* que atua em nível de usuário e do tipo binário. O Apêndice A mostra a sua descompactação e instalação. Para a compilação dos binários executados na máquina, utilizou-se o comando *install* que é um script que configura as opções necessárias ao funcionamento deste *Rootkit*.

6.3 ROOTKIT XZIBIT

Foi realizado um segundo estudo de caso com a utilização do *Rootkit xzibit* obtido através de *download*. Semelhante ao *Rootkit fuckit*, uma das funcionalidades destacáveis após a instalação é que o mesmo também oculta o diretório onde estão os códigos fontes utilizados para comprometer o ambiente operacional. Se executado o comando *ls* no terminal do *Ubuntu* em busca do diretório onde estão os arquivos binários desta ferramenta, simplesmente este local estará oculto e inacessível tornando a perícia mais difícil. É também um *Rootkit* binário.

No apêndice D vê-se todo o procedimento de descompactação e instalação do *Rootkit*. Para a criação dos binários deste programa foi necessário executar o comando *install* a partir do terminal *Linux*. O install é um *script* cuja

funcionalidade é configurar as todas as opções fundamentais para que o *Rootkit xzibit* funcione.

6.4 FERRAMENTAS PÓS-INTRUSÃO

Quando se está diante de um sistema comprometido com a presença de *Rootkits*, é necessário ter-se em mãos um conjunto de *softwares* capazes de auxiliar na investigação pericial. As ferramentas utilizadas no intuito de identificar-se *Rootkits* no sistema são: o *chkrootkit* e o *Rootkit hunter* que são programas que visam detectar se determinado ambiente *Linux* está vulnerável com estes *malwares*. Em sequência, instalou-se o *software fix-fu* cuja funcionalidade é remover os *Rootkits fuckit* e *xzibit* do sistema. Além dos testes realizados na máquina virtual *Ubuntu* 11.10, foram feitos também testes nas máquinas *Backtrack 5*, *PeriB*r e *FDTK-UbuntuBr*.

6.4.1 Chkrootkit

Ferramenta utilizada no intuito de identificar sinais deixados por *Rootkits*. O *chkroottkit* tem as seguintes funcionalidades: verifica os arquivos do sistema para encontrar alterações feitas por *Rootkits*, faz uma análise da interface de rede, procura por sinais de *trojans* e também verifica as possíveis alterações efetuadas por *Rootkits* em arquivos e diretórios do sistema.

O *chkrootkit* é um *software* que é valioso para a análise pericial de *Rootkits* em ambiente Linux. Esta ferramenta é um *scanne*r que identifica a presença de *Rootkits* instalados por meio da utilização da técnica de comparação de assinaturas (TEIXEIRA, 2005).

6.4.2 Rootkit hunter

Software Linux que visa em determinado sistema alterado por *Rootkits* a identificação desses *malwares*, também foi utilizado durante os testes para identificar a presença dos *Rootkits fuckit* e *xzibit*.

O *Rootkit hunter* verifica e procura pelas assinaturas de vários tipos de *Rootkits* em ambientes *Linux*; encontra-se licenciado por meio da GPL. Entretanto, o *Rootkit hunter* possui as seguintes funcionalidades: comparação de *hash MD5*, analisa e verifica arquivos constantemente utilizados por *Rootkits*, checa permissões incorretas para arquivos binários, faz um exame de *strings* suspeitas nos módulos *LKM* e *KLD*, verifica e procura por arquivos ocultos e faz uma análise completa em arquivos textos e binários (JUNIOR; MARANGON, […2005]).

6.4.3 Fix-fu

Esta é uma ferramenta desenvolvida para remover *Rootkits* em sistemas *Linux* com especial destaque para a distribuição *Ubuntu*. Foi utilizada e testada nas máquinas virtuais com o objetivo verificar se é capaz de remover os *Rootkits fuckit* e *xzibit*.

Em Apêndice G nota-se todo um processo de instalação deste *software*; sendo que, para configuração e compilação de seus binários foi necessário executar o comando *delrk* no terminal *Linux*.

6.4.4 Usabilidade das ferramentas Backtrack 5, PeriBr e FDTK-UbuntuBr

Distribuição *Linux* que contém um conjunto de ferramentas necessárias para realizar alguns testes de segurança de redes e aplicações. Entretanto esta distribuição pode ser configurada em diferentes maneiras; sendo que, pode ser criado um *Live-CD* ou drive USB de inicialização e executá-lo em um ambiente vivo, é possível instalar em uma máquina virtual (VM) ou ser instalado diretamente em um disco rígido e inicializar a ele como o sistema operacional principal. Entretanto, cada método tem as suas vantagens e desvantagens, mas, em casos de realizações constantes de avaliações e testes é recomendável que se crie um *Backtrack 5* por meio de uma máquina virtual (HACKING9-TEAM, 2012, tradução nossa).

Com a finalidade de realizarem-se testes envolvendo a instalação dos *Rootkits fuckit* e *xzibit*, o *Backtrack 5* foi testado no *VMware*. De realçar que nesta distribuição *Linux* estão presentes também os *softwares chkrootkit* e *Rootkit hunter* descritos anteriormente.

Os testes também realizaram-se nas máquinas virtuais *PeriBr* e *FDTK-UbuntuBr*. Os *softwares chkrootkit* e *Rootkit hunter* estão integrados também no conjunto de ferramentas pertencentes aos *Live-CDs* de perícia PeriBr e FDTK-UbuntuBr.

6.5 RESULTADOS OBTIDOS E TAXONOMIAS OBSERVADAS

Uma investigação forense computacional é iniciada quando determinado ambiente operacional possui comprometimento em suas caraterísticas funcionais. Após o investigador ser informado sobre os incidintes, ele deve estar bem preparado e munido de ferramentas precisas para periciar a máquina e/ou sistema a fim de solucionar as possíveis inconsistências que podem ser encontradas.

Em casos de perícia computacional de *Rootkits* em sistema *Linux*, a análise deve ser acompanhada de ferramentas específicas que auxiliem na identificação e detecção destas intrusões. As ferramentas *chkrootkit* e *Rootkit hunter* foram utilizadas durante os testes no intuito de identificar a presença dos *Rootkits* e verificar a existência de arquivos ou diretórios infectados.

De posse dos *logs* obtidos pelo *software chkrootkit*, no Apêndice C é possível observar que o diretório */bin/ps* encontra-se inacessível e imprime a mensagem "*No such file or directory*" ou "Não existe tal arquivo ou diretório". As inconsistências encontradas neste diretório foram causadas pelo *Rootkit fuckit*. Ainda no mesmo apêndice vê-se que o diretório */sbin/init* está infectado o que caracteriza a intrusão efetuada pelo *Rootkit fuckit*.

Com o auxilio do *chkrootkit* também foi possível checar o sistema em busca de sinais deixados pelo *Rootkit xzibit* e infecções causadas por este. No Apêndice F é possíver observar-se que os comandos *ifconfig*, *netstat*, *hdparm*, *top* e o prototocolo de acesso remoto de correio eletrônico (*pop3*) estão infectados. Estas inconsistências encontradas poderiam ser causadas pelo *Rootkit xzibit*.

Outra opção para detecção dos *Rootkits fuckit* e *xzibit* foi o *software Rootkit hunter*. Esta ferramenta foi capaz de identificar e mostrar com certeza e precisão a presença destes *Rootkits*; o que faz com que se torne peça fundamental em respostas a incidentes de segurança envolvendo *Rootkits*, por ser um *software* eficaz e eficiente na detecção desses *malwares*. Nos Apêndices B e E é possível

observar-se os *logs* desta ferramenta em busca dos *Rootkits fuckit* e *xzibit*. Ambos foram detetados assim como diversos diretórios infectados por estes.

É importante salientar ainda que nos binários dos *Rootkits fuckit* e *xzibit* estão compilados também outros *malwares*. No primeiro está também presente o *Rootkit tuxtendo* enquanto que nos binários do *Rootkit xzibit* estão implementados os *Rootkits devil*, *MRK*, *component* e *unknown*. Porém, a ferramenta *Rootkit hunter* identificou todos estes *Rootkits* complementares.

Porém, corre-se o risco de alterar-se as evidências digitais durante a execução dos programas de detecção de *Rootkits*. Nem sempre os resultados apresentados pelos *softwares anti-rootkits* são satisfatórios, podendo estes ser modificados pelos próprios *Rootkits*. Eles podem interferir em análises realizadas por ferramentas *anti-rootkits* para que os resultados alcançados durante a análise do sistema, apesar de falsos, ser vistos como verídicos em função da infiltração dos *Rootkits*. Sendo assim, há que ter-se conhecimento necessário sobre o funcionamento destes *malwares* e das ferramamentas de detecção (MACEDO, 2010).

Uma vez instalados os *Rootkits* e dada que a remoção é difícil, a solução a ser adotada é fazer-se o *backup* dos dados e reinstalar o sistema comprometido. A adoção deste procedimento deve-se principalmente pelo fato destes *malwares* trazerem consigo *backdoors*, *sniffers* e *keyloggers* que capturam informações do teclado. Em caso de ataques e acesso remoto, o invasor poderá capturar tudo o que a vítima digitar.

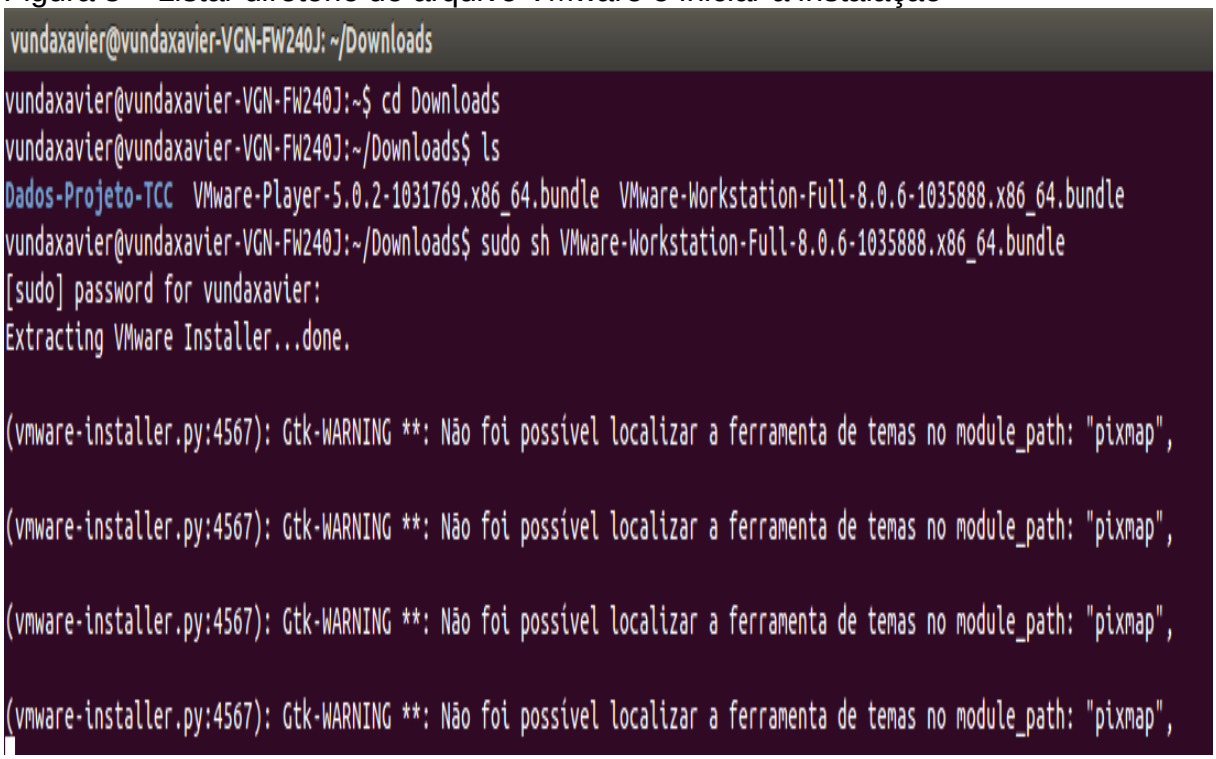
A etapa seguinte foi a utilização do *software fix-fu*, programa cuja função é remover os *Rootkits* instalados. Infelizmente, nos testes realizados, o *fix-fu* não conseguiu remover os códigos maliciosos.

Entretanto, o resultado dos testes realizados na máquina virtual *Ubuntu* 11.10 foi o mesmo obtido nas *VM PeriBr*, *FDTK-UbuntuBr* e *Backtrack 5*; pelo que, todas estas distribuições *Linux* têm em comum o simples fato de, estar voltada para o Ubuntu e ter as ferramentas *chkrootkit* e *Rootkit hunter*.

6.5.1 Instalação do VMware no Ubuntu 11.10

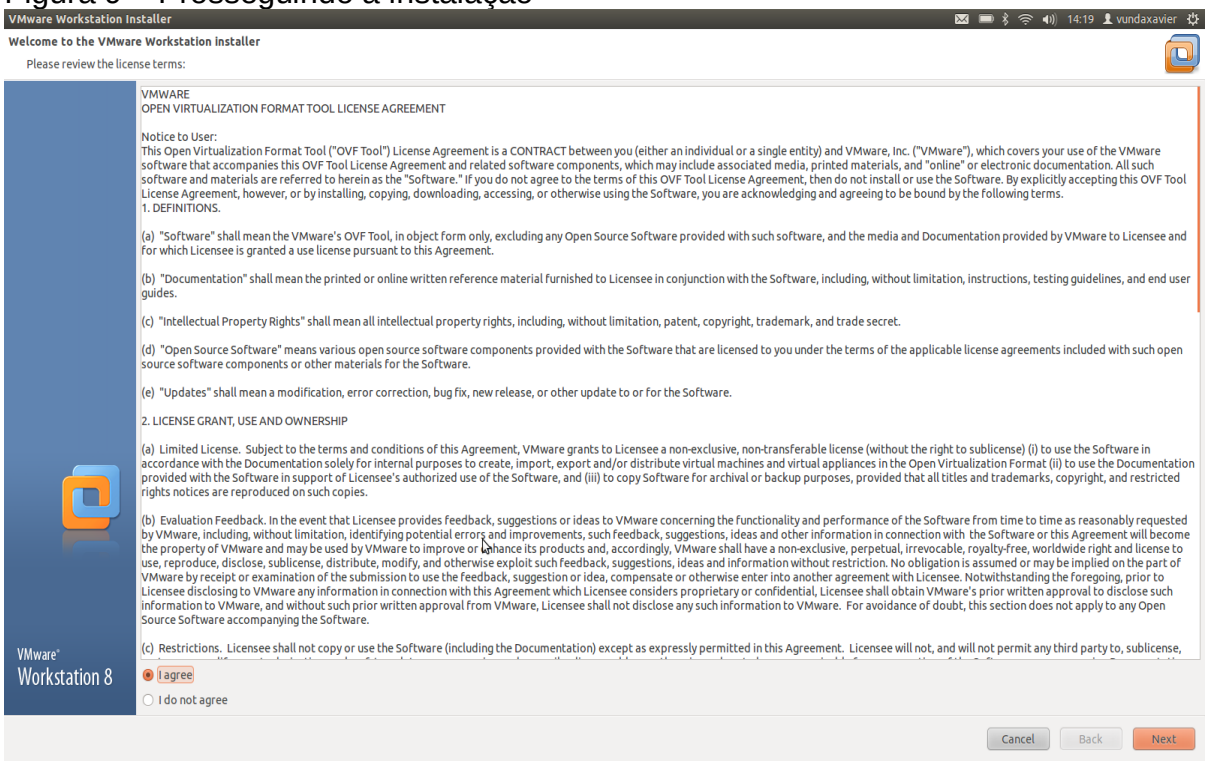
A seguir é apresentado todo o procedimento de instalação do *VMware* no *Ubuntu* 11.10, para a criação das máquinas virtuais utilizadas durante os testes de intrusão e detecção de *Rootkits*.

Figura 8 – Listar diretório do arquivo VMware e iniciar a instalação



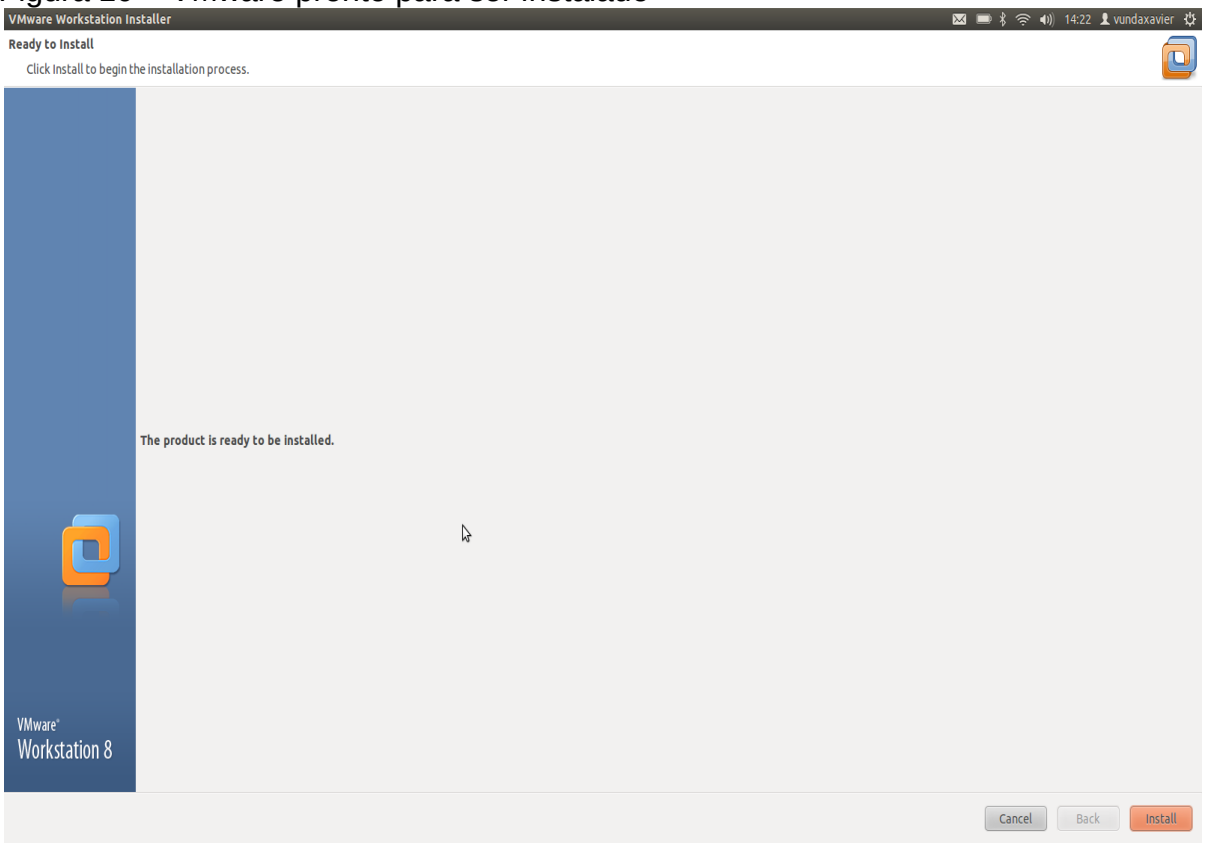
Fonte: Dados do pesquisador.

Figura 9 – Prosseguindo a instalação



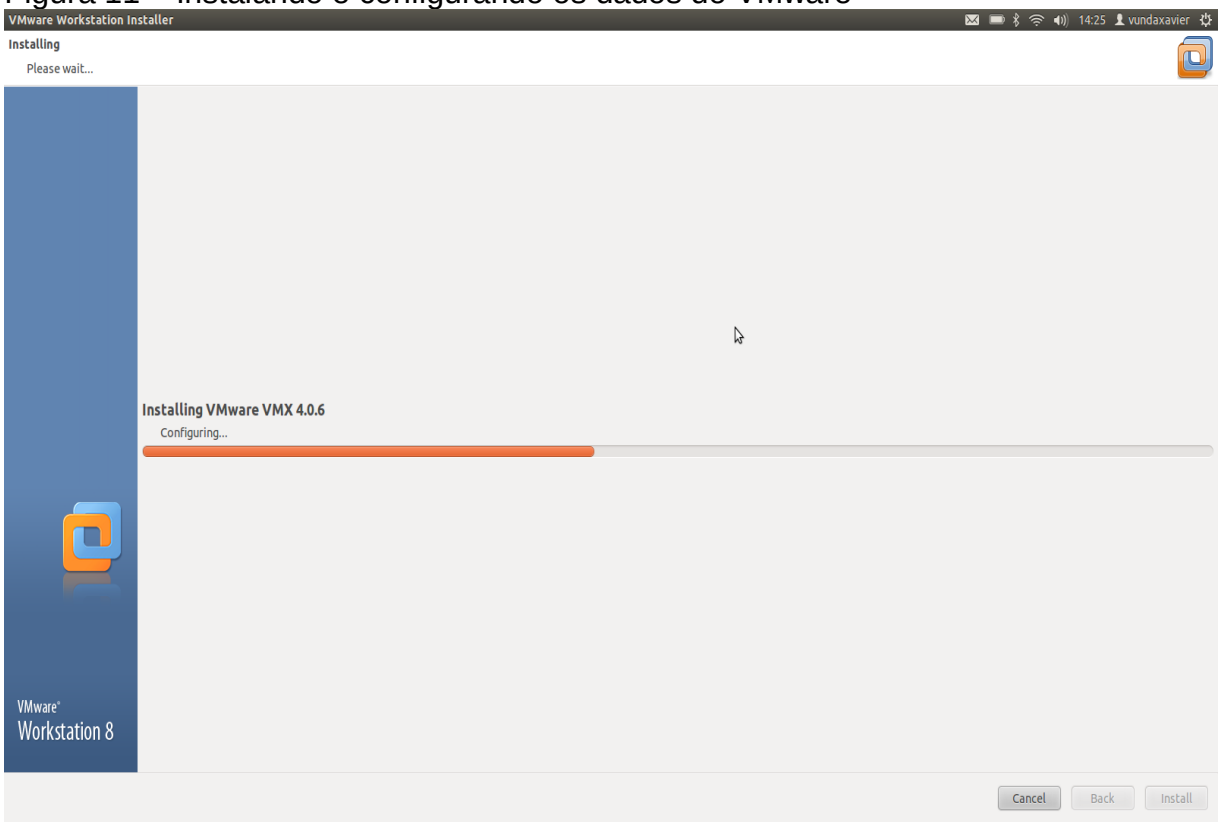
Fonte: Dados do pesquisador.

Figura 10 – VMware pronto para ser instalado



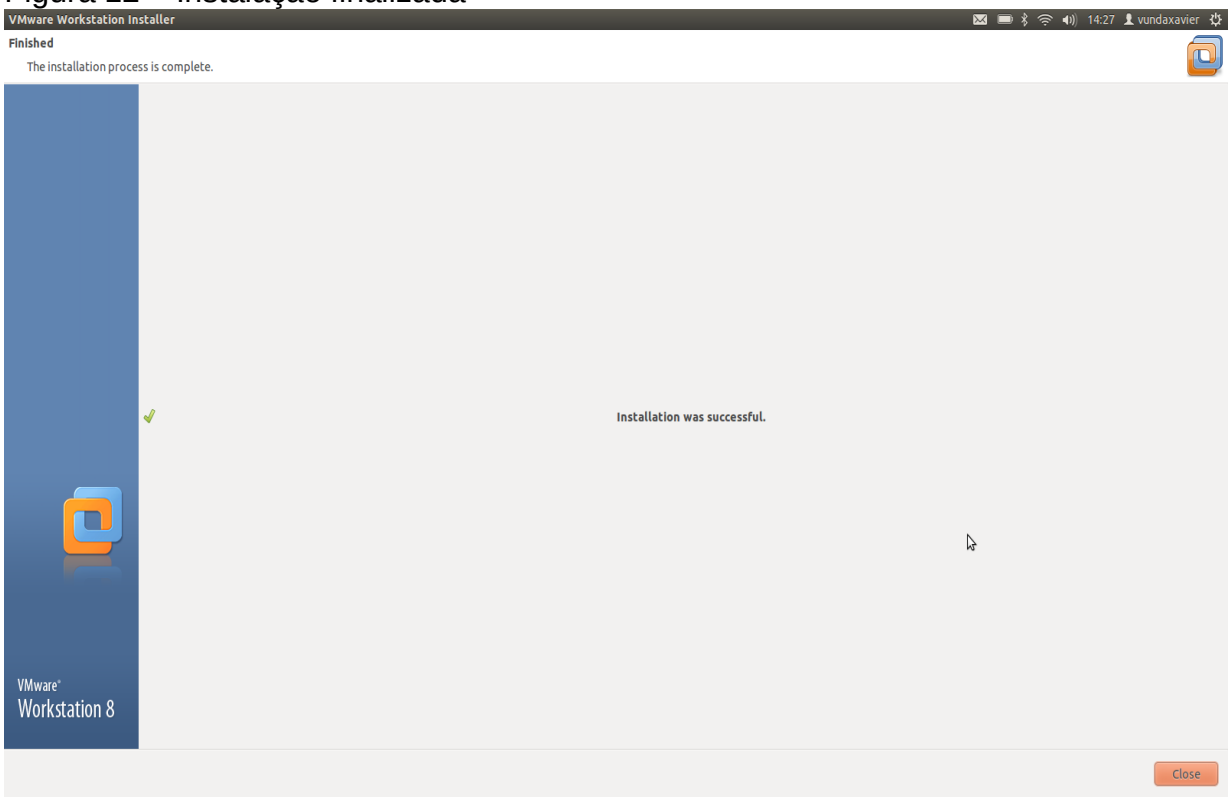
Fonte: Dados do pesquisador.

Figura 11 – Instalando e configurando os dados do VMware



Fonte: Dados do pesquisador.

Figura 12 – Instalação finalizada

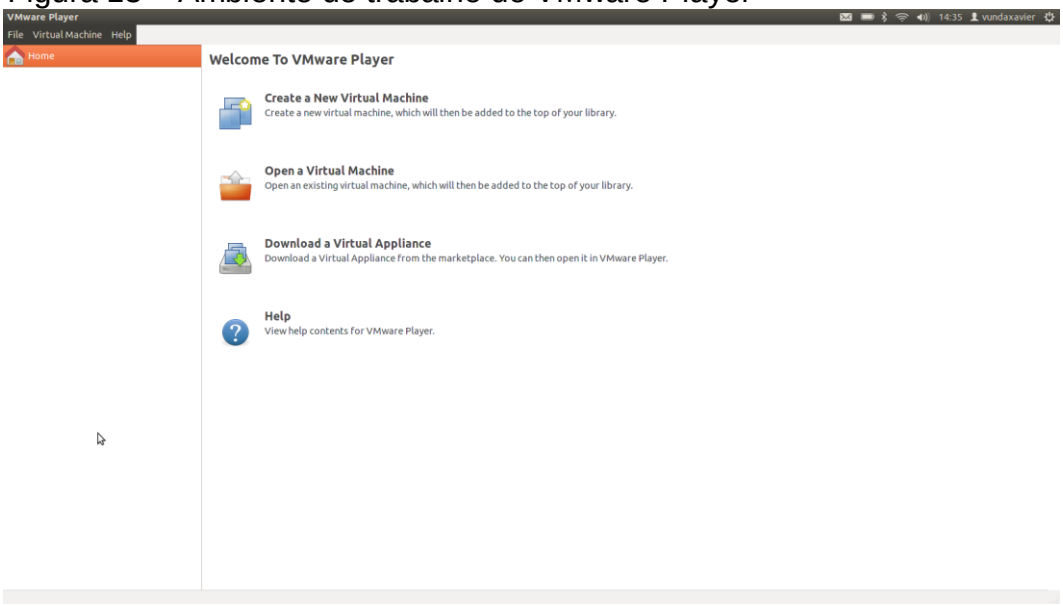


Fonte: Dados do pesquisador.

6.5.2 Criação da máquina virtual Ubuntu 11.10

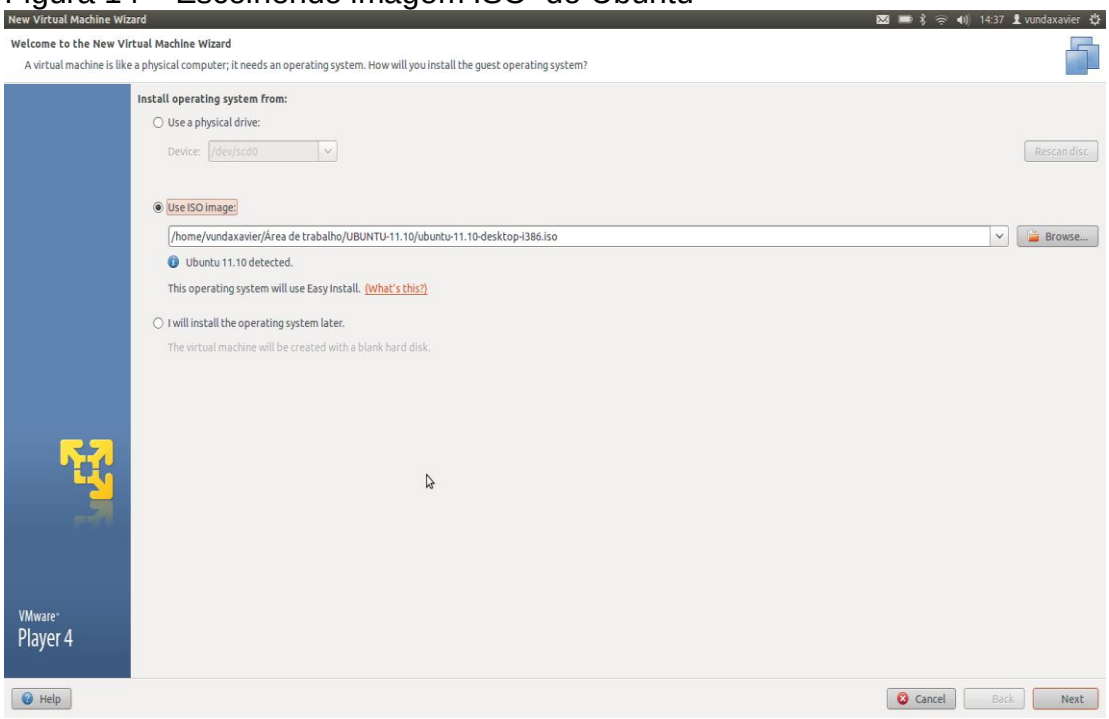
As imagens apresentadas a seguir representa todas as etapas seguidas durante a instalação da *VM Ubuntu* 11.10 a partir do *software* de virtualização Vmware player.

Figura 13 – Ambiente de trabalho do VMware Player



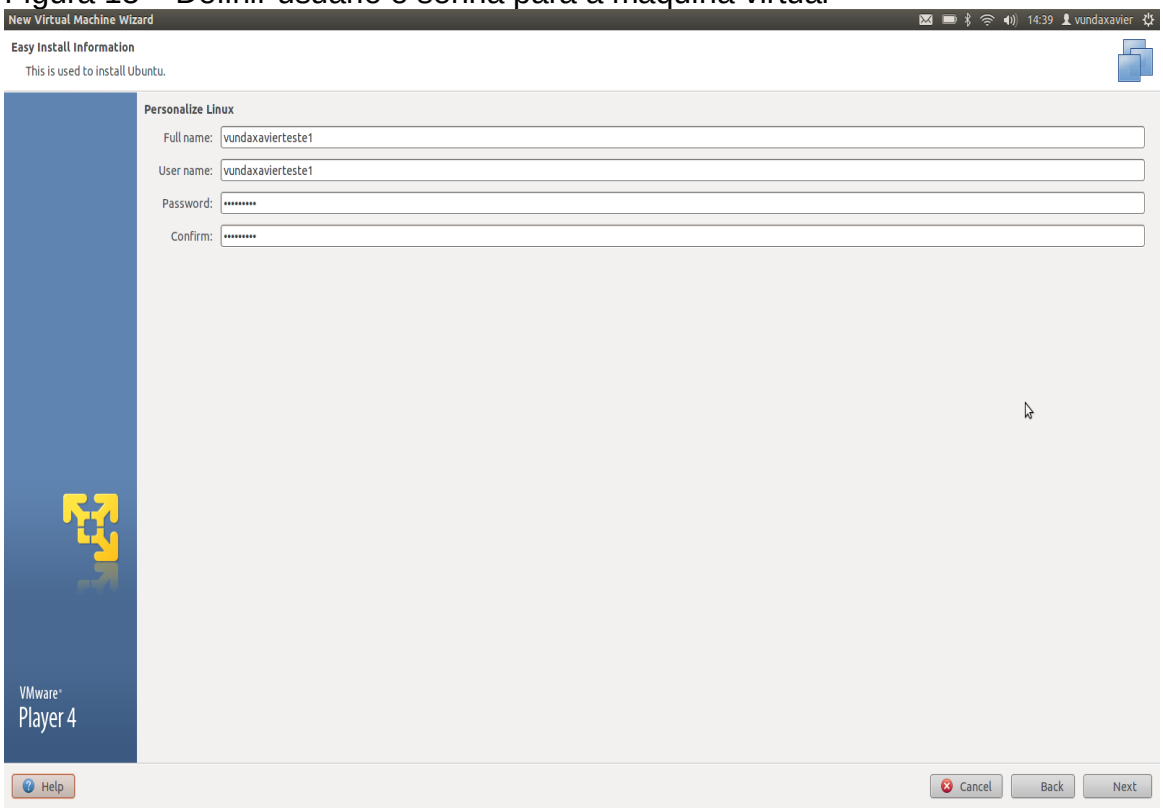
Fonte: Dados do pesquisador.

Figura 14 – Escolhendo imagem ISO do Ubuntu



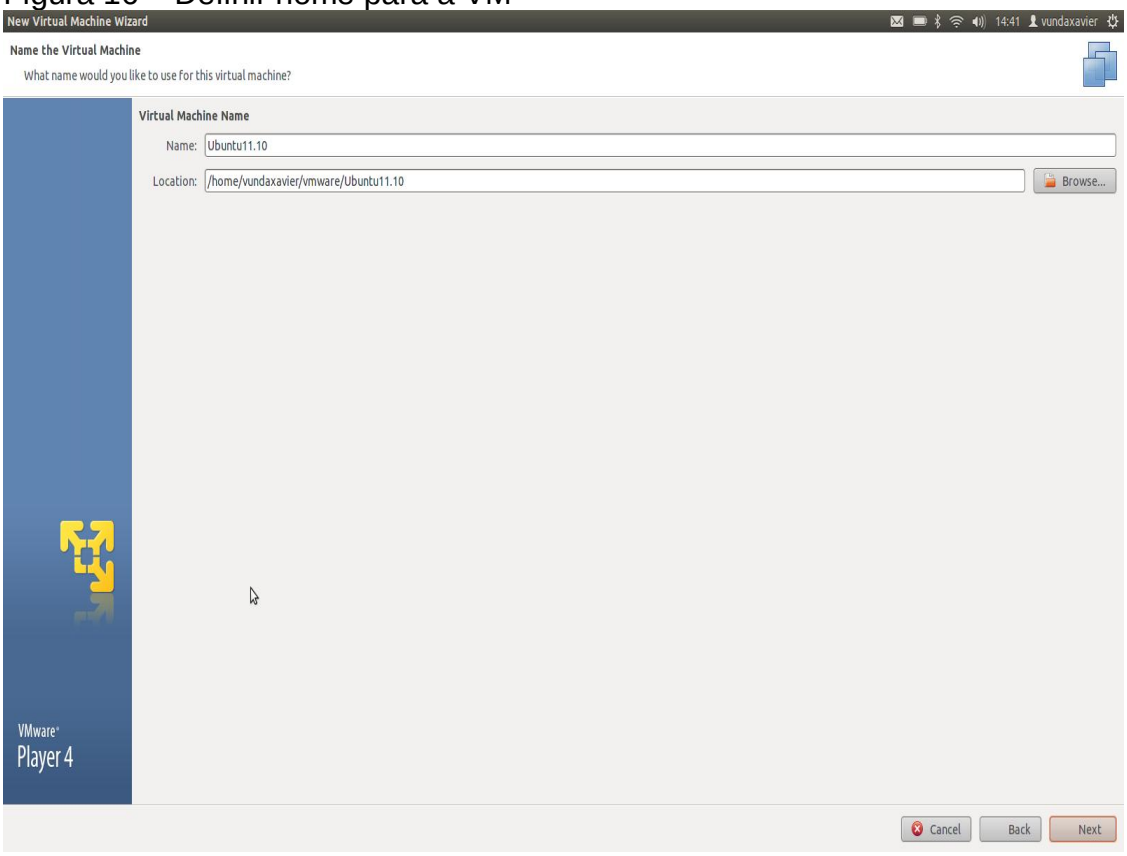
Fonte: Dados do pesquisador.

Figura 15 – Definir usuário e senha para a máquina virtual



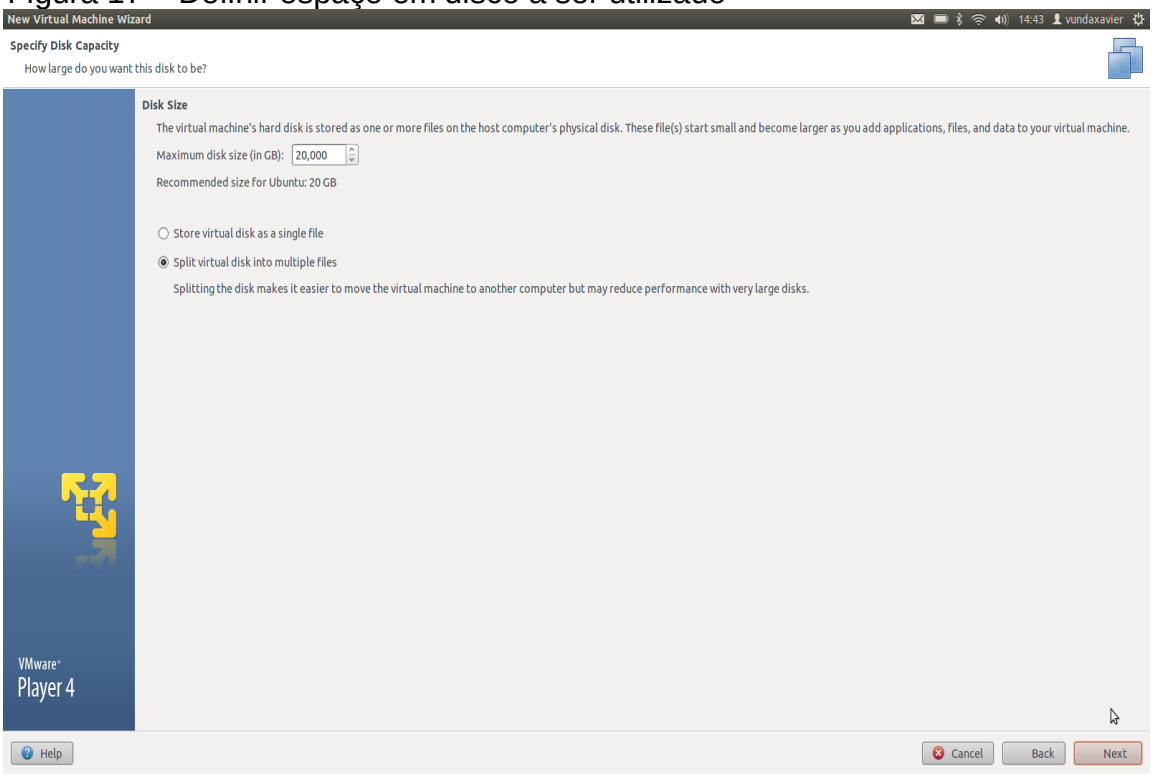
Fonte: Dados do pesquisador.

Figura 16 – Definir nome para a VM



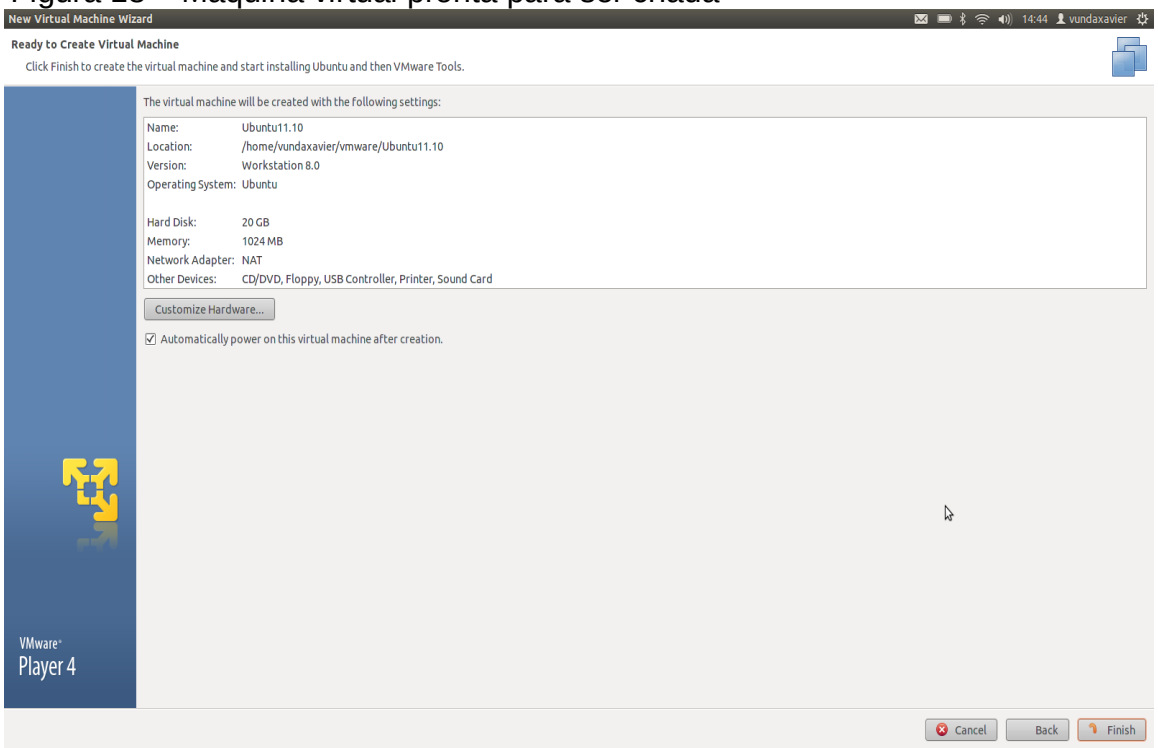
Fonte: Dados do pesquisador.

Figura 17 – Definir espaço em disco a ser utilizado



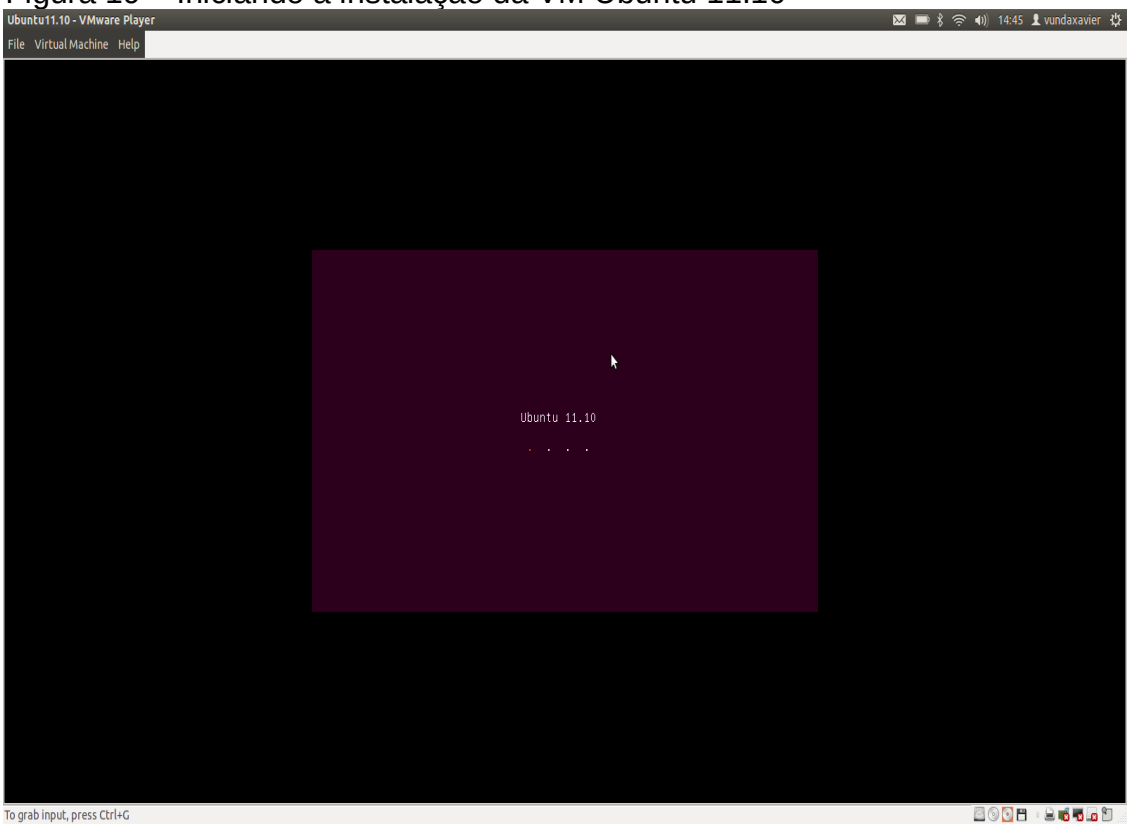
Fonte: Dados do pesquisador.

Figura 18 – Máquina virtual pronta para ser criada



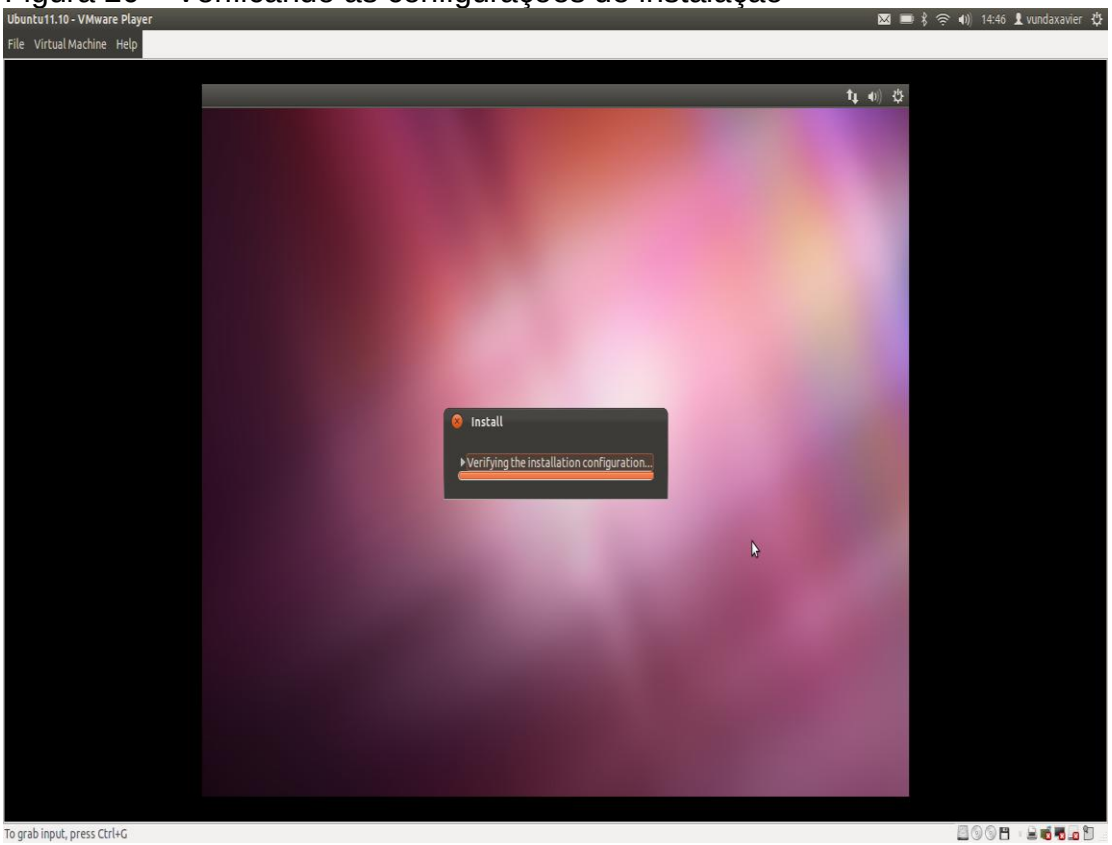
Fonte: Dados do pesquisador.

Figura 19 – Iniciando a instalação da VM Ubuntu 11.10



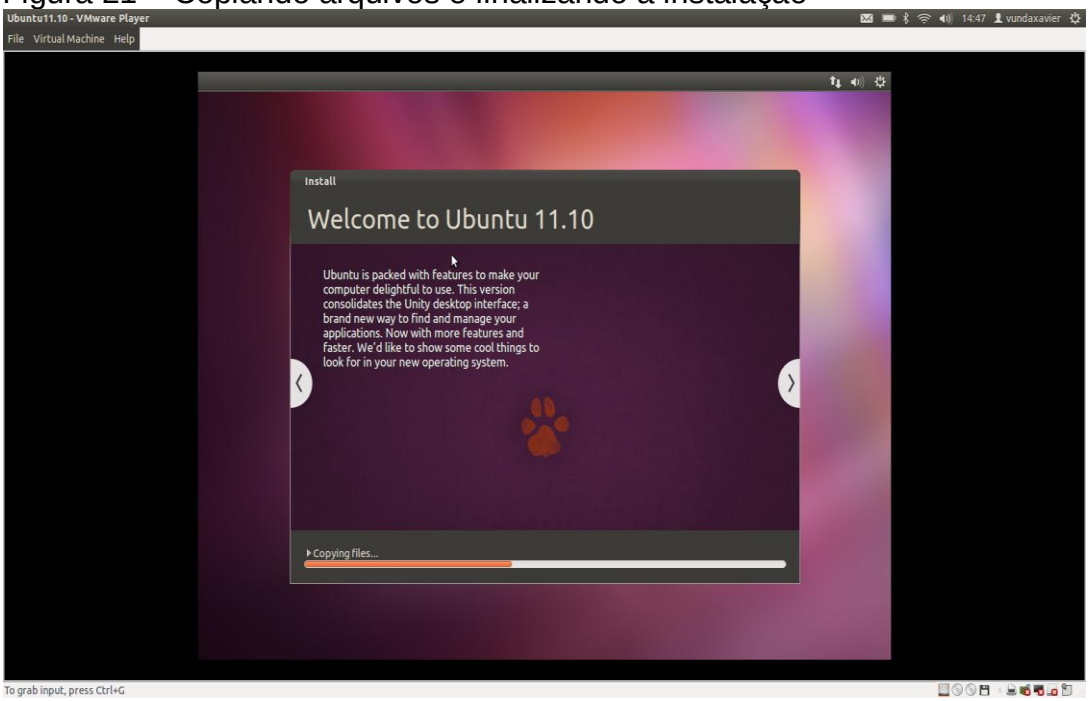
Fonte: Dados do pesquisador.

Figura 20 – Verificando as configurações de instalação



Fonte: Dados do pesquisador.

Figura 21 – Copiando arquivos e finalizando a instalação



Fonte: Dados do pesquisador.

6.5.3 Instalação e execução do chkrootkit e Rootkit hunter

As tabelas 5 e 6 a seguir, apresentam os passos mínimos para a instalação e execução dos programas *chkrootkit* e *Rootkit hunter* respectivamente.

Tabela 5 – Informações sobre o chkrootkit

Instalação	Execução
sudo apt-get install chkrootkit	sudo chkrootkit

Fonte: Adaptado de Software Livre Brasil (2009).

Tabela 6 – Informações sobre o rkhunter

Instalação	Execução
sudo apt-get install rkhunter	sudo rkhunter -c

Fonte: Adaptado de Software Livre Brasil (2009).

7 CONCLUSÃO

No decorrer dos anos, atividades e projetos que envolvem propagação e disseminação de *Rootkits* em sistema *Linux* têm aumentado em larga escala. *Rootkits* são vistos como ferramentas que normalmente ocultam arquivos e processos em execução e por outro lado escondem-se dos *softwares* de segurança. Suas funcionalidades dificultam significativamente as atividades que envolvem a perícia computacional dentre elas a identificação e remoção desses *malwares*.

O presente trabalho foi uma demonstração de como os *Rootkits* binários operam em determinado sistema e como as equipes de resposta a estes incidentes atuam no intuito de definir estratégias para detecção destas ameaças. Foi desenvolvido em ambiente *Linux* controlado onde os ataques dos *Rootkits fuckit* e *xzibit* foram pré-definidos.

Este trabalho proporcionou informações pertinentes de como usuários *Linux* poderão atuar em casos de intrusão de *Rootkits* binários, adotando a utilização de máquinas virtuais e ferramentas que auxilie na detecção destas intrusões que comprometem a segurança do sistema computacional.

Dentre as técnicas computacionais que foram adotadas para auxiliar a investigação forense computacional de *Rootkits* destacam-se a virtualização e a usabilidade das ferramentas *chkrootkit* e *Rootkit hunter* em busca de evidências digitais deixadas por *Rootkits*.

Como trabalhos futuros com relação a este tema, sugere-se a realização de testes de *Rootkits* em plataforma *Windows* com base na detecção e remoção de *Rootkits* nos níveis de usuário e *kernel* respectivamente.

REFERÊNCIAS

ARAUJO, Everson Santos. **Introdução à Segurança da Informação**. 2000. Disponível em: <http://everson.com.br/files/Introdu%C3%A7%C3%A3o%20%C3%A0%20Seguran%C3%A7a%20da%20Informa%C3%A7%C3%A3o.pdf>. Acesso em: 10 set. 2012.

AVILA, Alex. **Rootkits**. 2010. Disponível em: <http://www.slideshare.net/alex.avila1976/rootkit-3226171#btnNext>. Acesso em: 24 nov. 2012.

BASTO, Fabrício Cristian. **Computação Forense com Software Livre**. 2012. Disponível em: < http://www.slideshare.net/Analistati/sistemas-computacao-forense#btnPrevious>. Acesso em: 20 nov. 2012.

CAIXETA, Thiago Fernandes Gaspar. **Perícia Forense Digital:** A tecnologia da informação como ferramenta estratégica no combate ao crime. 2011. 63 f. Trabalho de Conclusão de Curso (Especialização) - Curso de Gestão de Segurança da Informação, Universidade Fumec, Belo Horizonte, 2011.

CLARKE, Nathan. **Computer Forensics**. A Pocket Guide. United Kingdon. IT Governance Publishing. 2010.

COSTA, Daniel Morais da. **Boas Práticas Para Perícia Forense**. Monografia de Graduação. Faculdade de Jaguariúna. 2008.

COSTA, Marcelo Antonio Sampaio Lemos. **Computação Forense.** Campinas: Millennium, 2003. 150 p.

COSTA, Marcelo Antonio Sampaio Lemos. **Computação Forense.** 3. ed. Campinas: Millennium, 2011. 158 p.

CRESPO, Marcelo Xavier de Freitas. **Crimes Digitais.** São Paulo: Saraiva, 2011. 242 p.

DAVIS, Michael; BODMER, Sean; LEAMASTERS, Aaron. **Hacking Exposed Malware & Rootkits: Malware & Rootkits Security Secrets & Solutions**. New York: McGraw-Hill. 2010.

Disponível em: < http://cactus.eas.asu.edu/partha/Teaching/466.2012/Projects/Project-5-rootkit.htm >. Acesso em: 03 Mai. 2013.

Disponível em: < http://www.filewatcher.com/m/xzibit.tar.gz.308486-0.html >. Acesso em: 02 Mai. 2013.

ELEUTÉRIO, Pedro Monteiro da Silva; MACHADO, Marcio Pereira. **Desvendando a computação forense**. São Paulo: Novatec, 2010. 200 p.

FARMER, Dan; VENEMA, Wietse. **Perícia Forense Computacional.** São Paulo: Pearson Prentice Hall, 2007. 190 p.

FONTES, Edison. **Segurança de Informação: o usuário faz a diferença.** São Paulo: Saraiva, 2006. 172 p.

FREITAS, Andrey Rodrigues. **Perícia Forense Aplicada à Informática.** Rio de Janeiro: Brasport, 2006. 216 p.

HACKING9-TEAM. **Hacking On Demand**. The Guide To Backtrack. Disponível em: < http://www.backtrack-linux.org/documents/Hakin9_On_Deman_03_2012_Teasers.pdf>. Acesso em: 21 Mai. 2013.

HOLPERIN, Marco; LEOBONS, Rodrigo. **Análise Forense**. Disponível em: <http://www.gta.ufrj.br/grad/07_1/forense/afmb.html>. Acesso em: 21 Nov. 2012.

JUNIOR, Claudio Penasio; MARANGON, Sílvio Luís. **PERÍCIA FORENSE COMPUTACIONAL EM SISTEMAS ALTERADOS POR ROOTKITS**. […2005] Disponível em: < http://www.lsi.usp.br/~penasio/trabalhos/PSI5007-3009850-5223770-1-V1.10.pdf>. Acesso aos: 05. Mai. 2013.

KENT, Karen; CHEVALIER, Suzanne; GRANCE, Tim; DANG, Hung. **Guide to Integrating Forensic Techniques into Incident Response**. Gaithersburg. NIST. 2006.

LIMA, Arthur Diego de Lira. **FARO: Ferramenta para Análise Forense Computacional em Sistemas Linux Vivos.** 2009. 53 f. Trabalho de Conclusão de Curso (Engenheiro de Computação) - Curso de Engenharia de Computação, Departamento de Centro de Tecnologia, Universidade Federal do Rio Grande do Norte, Natal, 2009. Disponível em: <http://www.engcomp.ufrn.br/publicacoes/TCC-2009-2-7.pdf>. Acesso em: 24 out. 2012.

LIMA, Paulo Marco Ferreira. **Crimes de Computador e Segurança Computacional.** 2. ed. São Paulo: Atlas, 2011. 166 p.

MACEDO, Guilherme Matte. **Investigação Forense Digital de Rootkits em Sistemas Unix.**2010. 71 f. Trabalho de Conclusão de Curso (Graduação) - Curso de Ciência da Computação, Departamento de Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2010. Disponível em: <http://www.lume.ufrgs.br/handle/10183/26345>. Acesso em: 15 out. 2012.

MELO, Sandro. **Computação Forense com Software Livre.** Rio de Janeiro: Altas Books. 2009. 152 p.

METULA, Erez. **Managed Code Rootkits.** Massachusetts: Elsevier, 2011. 316 p.

MORAES, Alexandre Fernandes de. **Segurança em Redes**: fundamentos. São Paulo: Érica. 2010.

NEUKAMP, Paulo. **FDTK**. 2008. Disponível em: <http://fdtk.com.br/www/sobre/>. Acesso em: 17 nov. 2012.

NG, Reynaldo. **Forense computacional corporativa**. Rio de Janeiro: Brasport, 2007. 158 p.

OLIVEIRA, Breno Guimarães. **Procurando por Rootkits em Sistemas GNU/Linux**. Rio de Janeiro. GRIS – UFRJ. 2005.

PAIVA, Raphael Rosa Nunes Vieira. **CRIMES VIRTUAIS.** 2012. 55 f. Monografia de Conclusão de Curso (Bacharel em Direito) - Curso de Direito, Departamento de Instituto de Ciências Sociais, Centro Universitário do Distrito Federal – Udf, Brasília, 2012. Disponível em: <http://www.conteudojuridico.com.br/monografia-tcc-tese,crimes-virtuais,37145.html>. Acesso em: 10 out. 2012.

PHILIPP, Aaron; COWEN, David; DAVIS, Chris. **Hacking Exposed Computer Forensics**. Nova York. MacGraw-Hill. 2 ed. 2010.

PIMENTA, Flávio Aparecido. **Perícia forense computacional baseada em sistema operacional Windows XP Professional.** 2007. 109 f. Trabalho de Conclusão de Curso (Especialização) - Curso de Segurança de Redes e Sistemas, Centro Universitário Senac – Unidade Sorocaba, Sorocaba/sp, 2007. Disponível em: <http://www.datasecurity.com.br/index.php/biblioteca/file/12-pericia-forense-computacional-baseada-em-sistema-operacional-windows-xp-professional>. Acesso em: 15 out. 2012.

QUEIROZ, Claudemir; VARGAS, Raffael. **Investigação e perícia forense computacional.** Rio de Janeiro: Brasport, 2010. 136 p.

RAMOS, Allder Dores; SATURNINO, André Teixeira; FERREIRA, Pedro Henrique Amorim. **3AP.BR – Um novo conceito de Live CD para Forense Computacional.** 2009. 73 f. Trabalho de Conclusão de Curso (Graduado) - Curso de Tecnologia em Redes de Computadores, Faculdade de Tecnologia Senai de Desenvolvimento Gerencial – Fatesg, Goiânia, 2009. Disponível em: <http://www.forensetec.com.br/novo/monografias/monografia-allder-andre-pedro.pdf>. Acesso em: 03 out. 2012.

REYES, Anthony; WILES, Jack. **Cybercrime and Digital Forensics**. Massachusetts. Elsevier. 2007.

ROSANES, Pedro. **Rootkits**. 2011. Disponível em: <http://www.gris.dcc.ufrj.br/documentos/artigos/rootkits-survey>. Acesso em: 10 set. 2012.

RODRIGUES, Tony. **Resposta a Incidentes e Forense Computacional**. 2009. Disponível em: <http://forcomp.blogspot.com/2009/01/fccu-121.html>. Acesso em: 16 nov. 2012.

ROVER, Aires J. **Forense computacional**. Disponível em: <https://sites.google.com/a/cristiantm.com.br/forense/>. Acesso em: 10 nov. 2012.

SANTOS, Lauderino Azevedo Dos.**COMPUTAÇÃO FORENSE EM SISTEMAS GNU/LINUX.** 2008. 46 f. Monografia (Especialização) - Curso de Administração Em redes linux, Departamento de Ciência da Computação, Universidade Federal de Lavras, Goiânia, 2008. Disponível em: <http://0fx66.com/files/paper/computacao%20forense%20em%20sistemas%20gnu-linux.pdf>. Acesso em: 30 set. 2012.

SILVA, Rodrigo Segura. **Forense Digital: Produzindo Provas Legais**. 2010 Disponível em: <http://www.prevenirperdas.com.br/portal/index.php?option=com_content&view=article%20%20&id=177:forensedigital&catid=18:prevencao-a fraudes&Itemid=7%20%3E%20Acesso%20em:%20%20%2022/09/2011>. Acesso em: 28 out. 2012.

SOFTWARE LIVRE BRASIL. **USO DE CAÇADORES DE ROOTKITS NO LINUX**. 2009. Disponível em: < http://softwarelivre.org/rss/planetas/ubuntu-brasil/uso-de-cacadores-de-rootkits-no-linux>. Acesso em: 04 Jul. 2013.

TEIXEIRA, Ataliba de Oliveira. **Uma visão forense dos RootKits em Sistemas Linux.** 2005. 93 f. Monografia de Pós-graduação (Especialista em “administração em Redes Linux”) - Curso de Ciência da Computação, Departamento de Ciência da Computação, Universidade Federal de Lavras, Minas Gerais, 2005. Disponível em: <http://www.ginux.ufla.br/files/mono-AtalibaTeixeira.pdf>. Acesso em: 01 out. 2012.

TEIXEIRA, Paula Porfírio. **ANÁLISE E COMPARAÇÃO DE SOFTWARES PARA PERÍCIA FORENSE EM AMBIENTES LINUX BASEADO NA MÉTRICA FUNCTION POINT ANALYSIS.**2010. 118 f. Trabalho de Conclusão de Curso (Grau de Bacharel em Ciência da Computação) - Curso de Ciência da Computação, Departamento de Ciência da Computação, Universidade do Extremo Sul Catarinense., Criciúma, 2010. Disponível em: <http://www.kiron.unesc.net/tcc/arquivos/trabalhos/245.pdf>. Acesso em: 03 out. 2012.

TOLENTINO, Luciano Cordova; SILVA, Wanessa da; MELLO, Paulo Augusto. **Perícia Forense Computacional.** Revista Tecnologias em Projeção. 2011. Disponível em: <http://revista.faculdadeprojecao.edu.br/revista/index.php/projecao2/article/viewFile/154/136>. Acesso em: 15 out. 2012.

WHITMAN, Michael E; MATTORD, Herbert J. **Principles of Information Security**. Boston. Course Tecnology. 4 ed. 2012.

WMLUG (revista). **Rootkits**. Jul. 2009. Disponivel em: <http://www.wmlug.org/pdf/Rootkits.pdf>. Acesso aos: 02. Mai. 2013.

APÊNDICE(S)

APÊNDICE A – INSTALAÇÃO DO ROOTKIT FUCKIT

```
vundaxavierteste1@ubuntu:~$ cd Downloads
vundaxavierteste1@ubuntu:~/Downloads$ ls
delrk.tar.gz  rootkit.tgz
vundaxavierteste1@ubuntu:~/Downloads$ tar -zxvf rootkit.tgz
fk-0.4/
fk-0.4/install
fk-0.4/init
fk-0.4/filez.tgz
fk-0.4/filez/
fk-0.4/filez/bin/
fk-0.4/filez/lib/
fk-0.4/filez/dev/
fk-0.4/filez/usr/
fk-0.4/filez/bin/netstat
fk-0.4/filez/bin/ps
fk-0.4/filez/lib/libproc.so.2.0.7
fk-0.4/filez/dev/proc/
fk-0.4/filez/dev/proc/.bashrc
fk-0.4/filez/dev/proc/.bash_profile
fk-0.4/filez/dev/proc/.cshrc
fk-0.4/filez/dev/proc/fuckit/
fk-0.4/filez/dev/proc/toolz/
fk-0.4/filez/dev/proc/fuckit/hax0r
fk-0.4/filez/dev/proc/fuckit/hax0rshell
fk-0.4/filez/dev/proc/fuckit/config/
fk-0.4/filez/dev/proc/fuckit/system-bins/
fk-0.4/filez/dev/proc/fuckit/config/lports
fk-0.4/filez/dev/proc/fuckit/config/password
fk-0.4/filez/dev/proc/fuckit/config/progs
fk-0.4/filez/dev/proc/fuckit/config/rkconf
fk-0.4/filez/dev/proc/fuckit/config/rports
fk-0.4/filez/dev/proc/toolz/scan/
fk-0.4/filez/dev/proc/toolz/sploits/
fk-0.4/filez/dev/proc/toolz/scan/startwu
fk-0.4/filez/dev/proc/toolz/scan/7350wurm
fk-0.4/filez/dev/proc/toolz/sploits/rootme.c
fk-0.4/filez/usr/bin/
fk-0.4/filez/usr/lib/
fk-0.4/filez/usr/bin/skill
fk-0.4/filez/usr/bin/snice
fk-0.4/filez/usr/bin/top
fk-0.4/filez/usr/bin/watch
fk-0.4/filez/usr/bin/w
fk-0.4/filez/usr/bin/pgrep
fk-0.4/filez/usr/bin/pkill
fk-0.4/filez/usr/lib/libtty.a
fk-0.4/filez/usr/lib/libcps.a
vundaxavierteste1@ubuntu:~/Downloads$ ls
delrk.tar.gz  fk-0.4  rootkit.tgz
vundaxavierteste1@ubuntu:~/Downloads$ cd fk-0.4
vundaxavierteste1@ubuntu:~/Downloads/fk-0.4$ ls
filez  filez.tgz  init  install
vundaxavierteste1@ubuntu:~/Downloads/fk-0.4$ sudo ./install
[sudo] password for vundaxavierteste1:
 Phase #1: Unpacking & installing filez... done!
Phase #2: Killing system loggers.
./install: line 28: /sbin/service: No such file or directory
Phase #3: Starting backdoor... done!
Phase #4.1: Copying /sbin/init to /dev/proc/fuckit/system-bins/ .
Phase #4.2: Overwriting /sbin/init with custom program ... done!
Phase #5: Restarting system loggers.
./install: line 51: /sbin/service: No such file or directory
Postinstall settings.
-=- Removing /home/vundaxavierteste1/Downloads/fk-0.4 .
cp: cannot stat `README': No such file or directory
cp: cannot stat `CHANGES': No such file or directory
-=- Moving /home/vundaxavierteste1/Downloads/fk-0.4/fk.tgz to /dev/proc/fuckit ( in case u need it )
cp: cannot stat `../fk.tgz': No such file or directory
######## Installation completed! ########
```

APÊNDICE B – LOG DA FERRAMENTA ROOTKIT HUNTER REFERENTE AO ROOTKIT FUCKIT

```
[15:40:29] Running Rootkit Hunter version 1.3.8 on ubuntu
[15:40:29]
[15:40:29] Info: Start date is Tue Jun 4 15:40:29 BRT 2013
[15:40:29]
[15:40:29] Checking configuration file and command-line options...
[15:40:29] Info: Detected operating system is 'Linux'
[15:40:29] Info: Found O/S name: Ubuntu 11.10
[15:40:29] Info: Command line is /usr/bin/rkhunter -c
[15:40:29] Info: Environment shell is /bin/bash; rkhunter is using dash
[15:40:29] Info: Using configuration file '/etc/rkhunter.conf'
[15:40:29] Info: Installation directory is '/usr'
[15:40:29] Info: Using language 'en'
[15:40:29] Info: Using '/var/lib/rkhunter/db' as the database directory
[15:40:29] Info: Using '/usr/share/rkhunter/scripts' as the support script directory
[15:40:29] Info: Using '/usr/local/sbin /usr/local/bin /usr/sbin /usr/bin /sbin /bin' as the command directories
[15:40:29] Info: Using '/' as the root directory by default
[15:40:29] Info: Using '/var/lib/rkhunter/tmp' as the temporary directory
[15:40:29] Info: No mail-on-warning address configured
[15:40:29] Info: X will be automatically detected
[15:40:29] Info: Using second color set
[15:40:29] Info: Found the 'basename' command: /usr/bin/basename
[15:40:29] Info: Found the 'diff' command: /usr/bin/diff
[15:40:30] Info: Found the 'dirname' command: /usr/bin/dirname
[15:40:30] Info: Found the 'file' command: /usr/bin/file
[15:40:30] Info: Found the 'find' command: /usr/bin/find
[15:40:30] Info: Found the 'ifconfig' command: /sbin/ifconfig
[15:40:30] Info: Found the 'ip' command: /sbin/ip
[15:40:30] Info: Found the 'ldd' command: /usr/bin/ldd
[15:40:30] Info: Found the 'lsattr' command: /usr/bin/lsattr
[15:40:30] Info: Found the 'lsmod' command: /sbin/lsmod
[15:40:30] Info: Found the 'lsof' command: /usr/bin/lsof
[15:40:30] Info: Found the 'mktemp' command: /bin/mktemp
[15:40:30] Info: Found the 'netstat' command: /bin/netstat
[15:40:30] Info: Found the 'perl' command: /usr/bin/perl
[15:40:30] Info: Found the 'pgrep' command: /usr/bin/pgrep
[15:40:30] Info: Found the 'ps' command: /bin/ps
[15:40:30] Info: Found the 'pwd' command: /bin/pwd
[15:40:30] Info: Found the 'readlink' command: /bin/readlink
[15:40:30] Info: Found the 'stat' command: /usr/bin/stat
[15:40:30] Info: Found the 'strings' command: /usr/bin/strings
[15:40:30] Info: System is not using prelinking
[15:40:30] Info: Using the '/usr/bin/sha1sum' command for the file hash checks
[15:40:30] Info: Stored hash values used hash function '/usr/bin/sha1sum'
[15:40:31] Info: Stored hash values did not use a package manager
[15:40:31] Info: The hash function field index is set to 1
[15:40:31] Info: No package manager specified: using hash function '/usr/bin/sha1sum'
[15:40:31] Info: Previous file attributes were stored
[15:40:31] Info: Enabled tests are: all
[15:40:31] Info: Disabled tests are: suspscan hidden_procs deleted_files packet_cap_apps apps
[15:40:31] Info: Found ksym file '/proc/kallsyms'
[15:40:31] Info: Using 'date' to process epoch second times.
[15:40:31]
[15:40:31] Checking if the O/S has changed since last time...
[15:40:31] Info: Nothing seems to have changed.
[15:40:31] Info: Locking is not being used
[15:40:31]
[15:40:31] Starting system checks...
[15:40:31]
[15:40:31] Info: Starting test name 'system_commands'
[15:40:31] Checking system commands...
[15:40:31]
[15:40:31] Info: Starting test name 'strings'
[15:40:32] Performing 'strings' command checks
[15:40:32]   Scanning for string /usr/sbin/ntpsx            [ OK ]
[15:41:20]   /usr/bin/pgrep                                 [ Warning ]
[15:41:20] Warning: The file properties have changed:
[15:41:20]          File: /usr/bin/pgrep
[15:41:20]          Current hash: 442f1383e6c29fbc938dccf4b687fbfc640f6b5a
```

```
[15:41:20]          Stored hash : 54551e54041df65375a10802b1b3d0b8e23f131b
[15:41:20]          Current size: 12416    Stored size: 11160
[15:41:20]          Current file modification time: 1370371215 (04-Jun-2013 15:40:15)
[15:41:20]          Stored file modification time : 1026522545 (12-Jul-2002 22:09:05)
[15:41:21]   /usr/bin/pstree                          [ OK ]
[15:41:27]   /usr/bin/top                             [ Warning ]
[15:41:27] Warning: The file properties have changed:
[15:41:27]          File: /usr/bin/top
[15:41:28]          Current hash: 44a0ffadc915dbfee905ddf267eec4d4a00ddc0a
[15:41:28]          Stored hash : 920d79dd0809b0e72b34f3557fc3b348df68731d
[15:41:28]          Current size: 52336    Stored size: 38104
[15:41:28]          Current file modification time: 1370371215 (04-Jun-2013 15:40:15)
[15:41:28]          Stored file modification time : 1026522545 (12-Jul-2002 22:09:05)
[15:41:28]   /usr/bin/touch                           [ OK ]
[15:41:30]   /usr/bin/vmstat                          [ OK ]
[15:41:30]   /usr/bin/w                               [ Warning ]
[15:41:30] Warning: The file properties have changed:
[15:41:31]          File: /usr/bin/w
[15:41:31]          Current hash: fb2819df7d1c7a261311a105100fcae1333b71e7
[15:41:31]          Stored hash : c8ebb9c0987fbfba726ad42e7864d02fb547bd2f
[15:41:31]          Current size: 10020    Stored size: 8988
[15:41:31]          Current file modification time: 1370371215 (04-Jun-2013 15:40:15)
[15:41:31]          Stored file modification time : 1026522545 (12-Jul-2002 22:09:05)
[15:41:31]   /usr/bin/watch                           [ Warning ]
[15:41:31] Warning: The file properties have changed:
[15:41:31]          File: /usr/bin/watch
[15:41:32]          Current hash: 11794147771ea0c82010a929013d620ab555f067
[15:41:32]          Stored hash : 04b43c5b2bf5131aaa00634083cf3ffaf59f07ff
[15:41:32]          Current size: 9204    Stored size: 8068
[15:41:32]          Current file modification time: 1370371215 (04-Jun-2013 15:40:15)
[15:41:32]          Stored file modification time : 1026522545 (12-Jul-2002 22:09:05)
[15:41:32]   /usr/bin/wc                              [ OK ]
[15:41:35]   /usr/bin/unhide.rb                       [ Warning ]
[15:41:35] Warning: The command '/usr/bin/unhide.rb' has been replaced by a script: /usr/bin/unhide.rb: a /usr/bin/ruby -w script
text executable
[15:41:39]   /sbin/init                               [ Warning ]
[15:41:40] Warning: The file properties have changed:
[15:41:40]          File: /sbin/init
[15:41:40]          Current hash: f69357049a0bec86919683514d4795fc6a960912
[15:41:40]          Stored hash : c21a298a5eaef02475e8ea5c17d379e423a0231c
[15:41:40]          Current size: 89604    Stored size: 14109
[15:41:40]          Current file modification time: 1370371215 (04-Jun-2013 15:40:15)
[15:41:40]          Stored file modification time : 1370367833 (04-Jun-2013 14:43:53)
[15:41:40]   /sbin/insmod                             [ OK ]
[15:41:59]   /bin/mv                                  [ OK ]
[15:42:00]   /bin/netstat                             [ Warning ]
[15:42:00] Warning: The file properties have changed:
[15:42:00]          File: /bin/netstat
[15:42:00]          Current hash: b39c447172ab8825bac361036b6872b5e91284aa
[15:42:00]          Stored hash : 192704df9ff5c365660c30b3b5efd295afc4066a
[15:42:00]          Current size: 101228    Stored size: 56032
[15:42:00]          Current file modification time: 1370371215 (04-Jun-2013 15:40:15)
[15:42:00]          Stored file modification time : 1026577146 (13-Jul-2002 13:19:06)
[15:42:01]   /bin/ps                                  [ Warning ]
[15:42:01] Warning: The file properties have changed:
[15:42:01]          File: /bin/ps
[15:42:01]          Current hash: 62c7d1839f644c5dfb6179015e7e3017ac6a6afa
[15:42:01]          Stored hash : b22c5bb4990bfa024c11a13287f18f84e07e06ec
[15:42:01]          Current size: 65360    Stored size: 68072
[15:42:01]          Current file modification time: 1370371215 (04-Jun-2013 15:40:15)
[15:42:01]          Stored file modification time : 1026522473 (12-Jul-2002 22:07:53)
[15:42:10]
[15:42:10] Info: Starting test name 'rootkits'

[15:42:10] Checking for rootkits...
[15:42:10]
[15:42:10] Info: Starting test name 'known_rkts'
[15:42:10] Performing check of known rootkit files and directories
[15:42:10]
[15:42:10] Checking for 55808 Trojan - Variant A...
[15:42:10]   Checking for file '/tmp/.../r'                [ Not found ]
[15:42:39]
[15:42:39] Checking for Fuck`it Rootkit...
[15:42:39]   Checking for file '/lib/libproc.so.2.0.7'       [ Found ]
[15:42:39]   Checking for file '/dev/proc/.bash_profile'     [ Found ]
```

```
[15:42:39]   Checking for file '/dev/proc/.bashrc'           [ Found ]
[15:42:39]   Checking for file '/dev/proc/.cshrc'            [ Found ]
[15:42:39]   Checking for file '/dev/proc/fuckit/hax0r'      [ Found ]
[15:42:39]   Checking for file '/dev/proc/fuckit/hax0rshell' [ Found ]
[15:42:39]   Checking for file '/dev/proc/fuckit/config/lports' [ Found ]
[15:42:39]   Checking for file '/dev/proc/fuckit/config/rports' [ Found ]
[15:42:40]   Checking for file '/dev/proc/fuckit/config/rkconf' [ Found ]
[15:42:40]   Checking for file '/dev/proc/fuckit/config/password' [ Found ]
[15:42:40]   Checking for file '/dev/proc/fuckit/config/progs' [ Found ]
[15:42:40]   Checking for file '/dev/proc/fuckit/system-bins/init' [ Found ]
[15:42:40]   Checking for file '/usr/lib/libcps.a'           [ Found ]
[15:42:40]   Checking for file '/usr/lib/libtty.a'           [ Found ]
[15:42:40]   Checking for directory '/dev/proc'              [ Found ]
[15:42:40]   Checking for directory '/dev/proc/fuckit'       [ Found ]
[15:42:40]   Checking for directory '/dev/proc/fuckit/system-bins' [ Found ]
[15:42:41]   Checking for directory '/dev/proc/toolz'        [ Found ]
[15:42:41] Warning: Fuck`it Rootkit                          [ Warning ]
[15:42:41]          File '/lib/libproc.so.2.0.7' found
[15:42:41]          File '/dev/proc/.bash_profile' found
[15:42:41]          File '/dev/proc/.bashrc' found
[15:42:41]          File '/dev/proc/.cshrc' found
[15:42:41]          File '/dev/proc/fuckit/hax0r' found
[15:42:41]          File '/dev/proc/fuckit/hax0rshell' found
[15:42:41]          File '/dev/proc/fuckit/config/lports' found
[15:42:41]          File '/dev/proc/fuckit/config/rports' found
[15:42:41]          File '/dev/proc/fuckit/config/rkconf' found
[15:42:41]          File '/dev/proc/fuckit/config/password' found
[15:42:41]          File '/dev/proc/fuckit/config/progs' found
[15:42:42]          File '/dev/proc/fuckit/system-bins/init' found
[15:42:42]          File '/usr/lib/libcps.a' found
[15:42:42]          File '/usr/lib/libtty.a' found
[15:42:42]          Directory '/dev/proc' found
[15:42:42]          Directory '/dev/proc/fuckit' found
[15:42:42]          Directory '/dev/proc/fuckit/system-bins' found
[15:42:42]          Directory '/dev/proc/toolz' found
[15:42:42]
[15:43:34] Checking for Tuxtendo Rootkit...
[15:43:34]   Checking for file '/lib/libproc.so.2.0.7'       [ Found ]
[15:43:34]   Checking for file '/usr/bin/xsf'                [ Not found ]
[15:43:37]   Checking for directory '/dev/tux/backup'        [ Not found ]
[15:43:37] Warning: Tuxtendo Rootkit                         [ Warning ]
[15:43:37]          File '/lib/libproc.so.2.0.7' found
[15:43:37]
[15:44:31] Checking the network...
[15:44:31]
[15:44:31] Performing checks on the network ports
[15:44:31] Info: Starting test name 'ports'
[15:44:31]   Performing check for backdoor ports
[15:44:31]     Checking for TCP port 1524                   [ Not found ]
[15:44:32]     Checking for TCP port 1984                   [ Found ]
[15:44:32] Warning: Network TCP port 1984 is being used by /dev/proc/fuckit/hax0r. Possible rootkit: Fuckit Rootkit
           Use the 'lsof -i' or 'netstat -an' command to check this.
[15:44:32]     Checking for UDP port 2001                   [ Not found ]
[15:44:36]     Checking for TCP port 65535                  [ Not found ]
[15:44:36]   Checking for backdoor ports                     [ Warning ]
[15:44:37]
[15:44:37] Info: Starting test name 'hidden_ports'
[15:44:39]   Checking for hidden ports                       [ None found ]
[15:44:40]
[15:44:40] Performing checks on the network interfaces
[15:44:40] Info: Starting test name 'promisc'
[15:44:40]   Checking for promiscuous interfaces             [ None found ]
[15:44:40]
[15:44:40] Info: Test 'packet_cap_apps' disabled at users request.
[15:44:40]
[15:44:40] Info: Starting test name 'local_host'
[15:44:40] Checking the local host...
[15:44:40]
[15:44:40] Info: Starting test name 'startup_files'
[15:44:40] Performing system boot checks
[15:44:41]   Checking for local host name                    [ Found ]
[15:44:41]
[15:44:41] Info: Starting test name 'startup_malware'
[15:44:41]   Checking for system startup files               [ Found ]
[15:44:43]   Checking system startup files for malware       [ None found ]
```

```
[15:44:43]
[15:44:43] Info: Starting test name 'group_accounts'
[15:44:43] Performing group and account checks
[15:44:44]   Checking for passwd file                  [ Found ]
[15:44:44] Info: Found password file: /etc/passwd
[15:44:44]   Checking for root equivalent (UID 0) accounts   [ None found ]
[15:44:44] Info: Found shadow file: /etc/shadow
[15:44:44]   Checking for passwordless accounts          [ None found ]
[15:44:44]
[15:44:44] Info: Starting test name 'passwd_changes'
[15:44:44]   Checking for passwd file changes            [ None found ]
[15:44:44]
[15:44:44] Info: Starting test name 'group_changes'
[15:44:45]   Checking for group file changes             [ None found ]
[15:44:45]   Checking root account shell history files   [ None found ]
[15:44:45]
[15:44:45] Info: Starting test name 'system_configs'
[15:44:45] Performing system configuration file checks
[15:44:45]   Checking for SSH configuration file         [ Not found ]
[15:44:45]   Checking for running syslog daemon          [ Warning ]
[15:44:46] Warning: The syslog daemon is not running.
[15:44:46] Info: Found rsyslog configuration file: /etc/rsyslog.conf
[15:44:46]   Checking for syslog configuration file      [ Found ]
[15:44:46]   Checking if syslog remote logging is allowed    [ Not allowed ]
[15:44:46]
[15:44:46] Info: Starting test name 'filesystem'
[15:44:46] Performing filesystem checks
[15:44:47] Info: SCAN_MODE_DEV set to 'THOROUGH'
[15:44:47]   Checking /dev for suspicious file types     [ Warning ]
[15:44:48] Warning: Suspicious file types found in /dev:
[15:44:48]          /dev/proc/toolz/sploits/rootme.c: ASCII C program text, with CRLF, CR line terminators
[15:44:48]          /dev/proc/toolz/scan/startwu: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked,
corrupted section header size
[15:44:48]          /dev/proc/toolz/scan/7350wurm: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked, for
GNU/Linux 2.0.0, stripped
[15:44:48]          /dev/proc/fuckit/hax0rshell: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses
shared libs), for GNU/Linux 2.0.0, stripped
[15:44:48]          /dev/proc/fuckit/hax0r: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses
shared libs), for GNU/Linux 2.0.0, stripped
[15:44:48]          /dev/proc/fuckit/sshd.pid: ASCII text
[15:44:48]          /dev/proc/fuckit/system-bins/init: ELF 32-bit LSB shared object, Intel 80386, version 1 (SYSV), dynamically
linked (uses shared libs), for GNU/Linux 2.6.15, stripped
[15:44:48]          /dev/proc/fuckit/config/rkconf: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked
(uses shared libs), for GNU/Linux 2.0.0, stripped
[15:44:48]          /dev/proc/fuckit/config/rports: Linux Journalled Flash File system, little endian
[15:44:48]          /dev/proc/fuckit/config/progs: ASCII text
[15:44:48]          /dev/proc/fuckit/config/password: ASCII text
[15:44:48]          /dev/proc/fuckit/config/lports: Linux Journalled Flash File system, little endian
[15:44:49]   Checking for hidden files and directories   [ Warning ]
[15:44:49] Warning: Hidden directory found: /dev/.udev
[15:44:49] Warning: Hidden file found: /dev/proc/.cshrc: ASCII text
[15:44:49] Warning: Hidden file found: /dev/proc/.bashrc: ASCII text
[15:44:49] Warning: Hidden file found: /dev/proc/.bash_profile: ASCII English text
[15:44:49] Warning: Hidden file found: /dev/.initramfs: symbolic link to `/run/initramfs'
[15:44:49]
[15:44:49] Info: Test 'apps' disabled at users request.
[15:44:49]
[15:44:49] System checks summary
[15:44:49] ====================
[15:44:50]
[15:44:50] File properties checks...
[15:44:50] Files checked: 132
[15:44:50] Suspect files: 8
[15:44:50]
[15:44:50] Rootkit checks...
[15:44:50] Rootkits checked : 242
[15:44:50] Possible rootkits: 2
[15:44:50] Rootkit names    : Fuck`it Rootkit, Tuxtendo Rootkit
[15:44:51]
[15:44:51] Applications checks...
[15:44:51] All checks skipped
[15:44:51]
[15:44:51] The system checks took: 4 minutes and 18 seconds
[15:44:51]
[15:44:51] Info: End date is Tue Jun  4 15:44:51 BRT 2013
```

APÊNDICE C – LOG DO CHKROOTKIT REFERENTE AO ROOTKIT FUCKIT

```
vundaxavierteste1@ubuntu:~$ sudo chkrootkit
ROOTDIR is `/'
Checking `amd'...                                   not found
Checking `basename'...                              not infected
Checking `biff'...                                  not found
Checking `chfn'...                                  not infected
Checking `chsh'...                                  not infected
Checking `cron'...                                  not infected
Checking `crontab'...                               not infected
Checking `date'...                                  not infected
Checking `du'...                                    not infected
Checking `dirname'...                               not infected
Checking `echo'...                                  not infected
Checking `egrep'...                                 not infected
Checking `env'...                                   not infected
Checking `find'...                                  not infected
Checking `fingerd'...                               not found
Checking `gpm'...                                   not found
Checking `grep'...                                  not infected
Checking `hdparm'...                                not infected
Checking `su'...                                    not infected
Checking `ifconfig'...                              not infected
Checking `inetd'...                                 /bin/ps: error while loading shared libraries: libproc-3.2.7.so: cannot open shared
object file: No such file or directory
not infected
Checking `inetdconf'...                             not found
Checking `identd'...                                not found
Checking `init'...                                  not infected
Checking `killall'...                               not infected
Checking `ldsopreload'...                           not infected
Checking `login'...                                 not infected
Checking `ls'...                                    not infected
Checking `lsof'...                                  not infected
Checking `mail'...                                  not found
Checking `mingetty'...                              not found
Checking `netstat'...                               not infected
Checking `named'...                                 not found
Checking `passwd'...                                not infected
Checking `pidof'...                                 not infected
Checking `pop2'...                                  not found
Checking `pop3'...                                  not found
Checking `ps'...                                    not infected
Checking `pstree'...                                not infected
Checking `rpcinfo'...                               not infected
Checking `rlogind'...                               not found
Checking `rshd'...                                  not found
Checking `slogin'...                                not infected
Checking `sendmail'...                              not infected
Checking `sshd'...                                  /bin/ps: error while loading shared libraries: libproc-3.2.7.so: cannot open shared
object file: No such file or directory
not found
Checking `syslogd'...                               not tested
Checking `tar'...                                   not infected
Checking `tcpd'...                                  /bin/ps: error while loading shared libraries: libproc-3.2.7.so: cannot open shared
object file: No such file or directory
not infected
Checking `tcpdump'...                               not infected
Checking `top'...                                   not infected
Checking `telnetd'...                               not found
Checking `timed'...                                 not found
Checking `traceroute'...                            not found
Checking `vdir'...                                  not infected
Checking `w'...                                     not infected
Checking `write'...                                 not infected
Checking `aliens'...                                no suspect files
Searching for sniffer's logs, it may take a while...    nothing found
Searching for rootkit HiDrootkit's default files...     nothing found
Searching for rootkit t0rn's default files...           nothing found
Searching for t0rn's v8 defaults...                     nothing found
```

```
Searching for rootkit Lion's default files...              nothing found
Searching for rootkit RSHA's default files...              nothing found
Searching for rootkit RH-Sharpe's default files...         nothing found
Searching for Ambient's rootkit (ark) default files and dirs... nothing found
Searching for suspicious files and dirs, it may take a while... The following suspicious files and directories were found:
/usr/lib/pymodules/python2.7/.path /usr/lib/libreoffice/basis3.4/program/.services.rdb

Searching for LPD Worm files and dirs...                   nothing found
Searching for Ramen Worm files and dirs...                 nothing found
Searching for Maniac files and dirs...                     nothing found
Searching for RK17 files and dirs...                       nothing found
Searching for Ducoci rootkit...                            nothing found
Searching for Adore Worm...                                nothing found
Searching for ShitC Worm...                                nothing found
Searching for Omega Worm...                                nothing found
Searching for Sadmind/IIS Worm...                          nothing found
Searching for MonKit...                                    nothing found
Searching for Showtee...                                   nothing found
Searching for OpticKit...                                  nothing found
Searching for T.R.K...                                     nothing found
Searching for Mithra...                                    nothing found
Searching for LOC rootkit...                               nothing found
Searching for Romanian rootkit...                          nothing found
Searching for Suckit rootkit...                            Warning: /sbin/init INFECTED
Searching for Volc rootkit...                              nothing found
Searching for Gold2 rootkit...                             nothing found
Searching for TC2 Worm default files and dirs...           nothing found
Searching for Anonoying rootkit default files and dirs...  nothing found
Searching for ZK rootkit default files and dirs...         nothing found
Searching for ShKit rootkit default files and dirs...      nothing found
Searching for AjaKit rootkit default files and dirs...     nothing found
Searching for zaRwT rootkit default files and dirs...      nothing found
Searching for Madalin rootkit default files...             nothing found
Searching for Fu rootkit default files...                  nothing found
Searching for ESRK rootkit default files...                nothing found
Searching for rootedoor...                                 nothing found
Searching for ENYELKM rootkit default files...             nothing found
Searching for common ssh-scanners default files...         nothing found
Searching for suspect PHP files...                         nothing found
Searching for anomalies in shell history files...          nothing found
Checking `asp'...                                          not infected
Checking `bindshell'...                                    not infected
Checking `lkm'...                                          Usage: ./chkproc [-v] [-v] [-p procps version]
chkproc: nothing detected
chkdirs: nothing detected
Checking `rexedcs'...                                      not found
Checking `sniffer'...                                      lo: not promisc and no packet sniffer sockets
eth0: PACKET SNIFFER(/sbin/dhclient[828])
Checking `w55808'...                                       not infected
Checking `wted'...                                         chkwtmp: nothing deleted
Checking `scalper'...                                      not infected
Checking `slapper'...                                      not infected
Checking `z2'...                                           user vundaxavierteste1 deleted or never logged from lastlog!
Checking `chkutmp'...                                      ps: error while loading shared libraries: libproc-3.2.7.so: cannot open shared
object file: No such file or directory
chkutmp: nothing deleted
Checking `OSX_RSPLUG'...                                   not infected
```

APÊNDICE D – INSTALAÇÃO DO ROOTKIT XZIBIT

```
vundaxavierteste2@ubuntu:~$ cd Downloads
vundaxavierteste2@ubuntu:~/Downloads$ ls
xzibit.tar.gz
vundaxavierteste2@ubuntu:~/Downloads$ tar -zxvf xzibit.tar.gz
lamerk/
lamerk/ifconfig
lamerk/linsniffer
lamerk/logclear
lamerk/netstat
lamerk/ps
lamerk/sense
lamerk/sl2
lamerk/top
lamerk/s
lamerk/install
lamerk/sshdu
lamerk/ssh_host_key
lamerk/ssh_random_seed
lamerk/hdparm
lamerk/becys.cgi
vundaxavierteste2@ubuntu:~/Downloads$ ls
lamerk  xzibit.tar.gz
vundaxavierteste2@ubuntu:~/Downloads$ cd lamerk
vundaxavierteste2@ubuntu:~/Downloads/lamerk$ ls
becys.cgi  ifconfig  linsniffer  netstat  s     sl2  ssh_host_key  top
hdparm     install   logclear    ps       sense sshdu ssh_random_seed
vundaxavierteste2@ubuntu:~/Downloads/lamerk$ sudo ./install
[sudo] password for vundaxavierteste2:
Replacing netstat, ps, ifconfig, top... Done
Setting up the /dev filez... Done
Creating home...
Copying SSHD and shit...
mv: cannot stat `sl2new.c': No such file or directory
./install: 45: cannot create /etc/rc.d/rc.sysinit: Directory nonexistent
./install: 46: cannot create /etc/rc.d/rc.sysinit: Directory nonexistent
/usr/bin/hdparm: 4: ./linsniffer: not found
./install: 80: /sbin/ifconfig: not found
send-mail: fatal: open /etc/postfix/main.cf: No such file or directory
Can't send mail: sendmail process failed with error code 75

All done!

Way to go , Brukner!....
```

APÊNDICE E – LOG DO ROOTKIT HUNTER REFERENTE AO ROOTKIT XZIBIT

```
[09:30:02] Running Rootkit Hunter version 1.3.8 on ubuntu
[09:30:02]
[09:30:02] Info: Start date is Tue Jun  4 09:30:02 BRT 2013
[09:30:02]
[09:30:02] Checking configuration file and command-line options...
[09:30:02] Info: Detected operating system is 'Linux'
[09:30:02] Info: Found O/S name: Ubuntu 11.10
[09:30:02] Info: Command line is /usr/bin/rkhunter -c
[09:30:02] Info: Environment shell is /bin/bash; rkhunter is using dash
[09:30:03] Info: Using configuration file '/etc/rkhunter.conf'
[09:30:03] Info: Installation directory is '/usr'
[09:30:03] Info: Using language 'en'
[09:30:03] Info: Using '/var/lib/rkhunter/db' as the database directory
[09:30:03] Info: Using '/usr/share/rkhunter/scripts' as the support script directory
[09:30:03] Info: Using '/usr/local/sbin /usr/local/bin /usr/sbin /usr/bin /sbin /bin' as the command directories
[09:30:03] Info: Using '/' as the root directory by default
[09:30:03] Info: Using '/var/lib/rkhunter/tmp' as the temporary directory
[09:30:03] Info: No mail-on-warning address configured
[09:30:03] Info: X will be automatically detected
[09:30:03] Info: Using second color set
[09:30:03] Info: Found the 'basename' command: /usr/bin/basename
[09:30:03] Info: Found the 'diff' command: /usr/bin/diff
[09:30:03] Info: Found the 'dirname' command: /usr/bin/dirname
[09:30:03] Info: Found the 'file' command: /usr/bin/file
[09:30:03] Info: Found the 'find' command: /usr/bin/find
[09:30:03] Info: Found the 'ifconfig' command: /sbin/ifconfig
[09:30:03] Info: Found the 'ip' command: /sbin/ip
[09:30:03] Info: Found the 'ldd' command: /usr/bin/ldd
[09:30:03] Info: Found the 'lsattr' command: /usr/bin/lsattr
[09:30:03] Info: Found the 'lsmod' command: /sbin/lsmod
[09:30:04] Info: Found the 'lsof' command: /usr/bin/lsof
[09:30:04] Info: Found the 'mktemp' command: /bin/mktemp
[09:30:04] Info: Found the 'netstat' command: /bin/netstat
[09:30:04] Info: Found the 'perl' command: /usr/bin/perl
[09:30:04] Info: Found the 'pgrep' command: /usr/bin/pgrep
[09:30:04] Info: Found the 'ps' command: /bin/ps
[09:30:04] Info: Found the 'pwd' command: /bin/pwd
[09:30:04] Info: Found the 'readlink' command: /bin/readlink
[09:30:04] Info: Found the 'stat' command: /usr/bin/stat
[09:30:04] Info: Found the 'strings' command: /usr/bin/strings
[09:30:04] Info: System is not using prelinking
[09:30:04] Info: Using the '/usr/bin/sha1sum' command for the file hash checks
[09:30:04] Info: Stored hash values used hash function '/usr/bin/sha1sum'
[09:30:04] Info: Stored hash values did not use a package manager
[09:30:04] Info: The hash function field index is set to 1
[09:30:04] Info: No package manager specified: using hash function '/usr/bin/sha1sum'
[09:30:04] Info: Previous file attributes were stored
[09:30:04] Info: Enabled tests are: all
[09:30:04] Info: Disabled tests are: suspscan hidden_procs deleted_files packet_cap_apps apps
[09:30:04] Info: Found ksym file '/proc/kallsyms'
[09:30:04] Info: Using 'date' to process epoch second times.
[09:30:05]
[09:30:05] Checking if the O/S has changed since last time...
[09:30:05] Info: Nothing seems to have changed.
[09:30:05] Info: Locking is not being used
[09:30:05]
[09:30:05] Starting system checks...
[09:30:05]
[09:30:05] Info: Starting test name 'system_commands'
[09:30:05] Checking system commands...
[09:30:05]
[09:30:05] Info: Starting test name 'strings'
[09:30:05] Performing 'strings' command checks
[09:30:05]   Scanning for string /usr/sbin/ntpsx            [ OK ]
[09:30:05]   Scanning for string /usr/sbin/.../bkit-ava      [ OK ]
[09:31:00]   /usr/bin/top                                   [ Warning ]
[09:31:00] Warning: The file properties have changed:
[09:31:00]          File: /usr/bin/top
[09:31:00]          Current hash: 3aeea55f30ed6a53586ba1415a4dea6b92c783f8
[09:31:00]          Stored hash : 7a4abe4a19ee7b739165f7b6cf1872a82eb23175
```

```
[09:31:00]          Current inode: 1058543    Stored inode: 266969
[09:31:00]          Current size: 53588    Stored size: 72632
[09:31:00]          Current file modification time: 433385100 (25-Sep-1983 21:45:00)
[09:31:01]          Stored file modification time : 1308866480 (23-Jun-2011 19:01:20)
[09:31:01]   /usr/bin/touch                          [ OK ]
[09:31:01]   /usr/bin/tr                             [ OK ]
[09:31:07]   /usr/bin/unhide.rb                      [ Warning ]
[09:31:07] Warning: The command '/usr/bin/unhide.rb' has been replaced by a script: /usr/bin/unhide.rb: a /usr/bin/ruby -w script
text executable
[09:31:08]   /usr/bin/mawk                           [ OK ]
[09:31:11]   /sbin/ifconfig                          [ Warning ]
[09:31:11] Warning: The file properties have changed:
[09:31:11]          File: /sbin/ifconfig
[09:31:11]          Current hash: 7e8141ae50d462dbaf0f2084bf8f3577f8b72056
[09:31:11]          Stored hash : 079675775324db11974ab97956e2314658a87fe5
[09:31:11]          Current inode: 1058527    Stored inode: 917656
[09:31:11]          Current size: 19840    Stored size: 65708
[09:31:11]          Current file modification time: 433385100 (25-Sep-1983 21:45:00)
[09:31:12]          Stored file modification time : 1278055671 (02-Jul-2010 04:27:51)
[09:31:12]   /sbin/ifdown                            [ OK ]
[09:31:33]   /bin/mv                                 [ OK ]
[09:31:34]   /bin/netstat                            [ Warning ]
[09:31:34] Warning: The file properties have changed:
[09:31:34]          File: /bin/netstat
[09:31:34]          Current hash: 7c50a2f2a5857b2ecc456ff32bbd8f3a31520e5e
[09:31:34]          Stored hash : 77fb05b6622cdc4e087a6a5b26b45223c4d950a1
[09:31:34]          Current inode: 1058539    Stored inode: 786527
[09:31:34]          Current size: 35300    Stored size: 110088
[09:31:34]          Current file modification time: 433385100 (25-Sep-1983 21:45:00)
[09:31:34]          Stored file modification time : 1278055671 (02-Jul-2010 04:27:51)
[09:31:35]   /bin/ps                                 [ Warning ]
[09:31:35] Warning: The file properties have changed:
[09:31:35]          File: /bin/ps
[09:31:35]          Current hash: d5cd2670ab8c4106185d9bda20ed384eab5a2677
[09:31:36]          Stored hash : faceae36495a5eea3d7748cf3419f63137bd027e
[09:31:36]          Current inode: 1058540    Stored inode: 786561
[09:31:36]          Current size: 33280    Stored size: 87900
[09:31:36]          Current file modification time: 433385100 (25-Sep-1983 21:45:00)
[09:31:36]          Stored file modification time : 1308866480 (23-Jun-2011 19:01:20)
[09:31:36]   /bin/pwd                                [ OK ]
[09:31:41]   /bin/dash                               [ OK ]
[09:31:45]
[09:31:45] Info: Starting test name 'rootkits'
[09:31:45] Checking for rootkits...
[09:31:45]
[09:31:45] Info: Starting test name 'known_rkts'
[09:31:45] Performing check of known rootkit files and directories
[09:31:45]
[09:31:45] Checking for 55808 Trojan - Variant A...
[09:31:45]   Checking for file '/tmp/.../r'          [ Not found ]
[09:31:45]   Checking for file '/tmp/.../a'          [ Not found ]
[09:32:02] Warning: Devil RootKit                    [ Warning ]
[09:32:03]          File '/dev/dsx' found
[09:32:03]          File '/dev/caca' found
[09:32:03]
[09:32:03] Checking for Dica-Kit Rootkit...
[09:32:03]   Checking for file '/lib/.sso'           [ Not found ]
[09:32:03]   Checking for file '/lib/.so'            [ Not found ]
[09:32:31]
[09:32:31] Checking for MRK Rootkit...
[09:32:31]   Checking for file '/dev/ida/.inet/pid'          [ Found ]
[09:32:31]   Checking for file '/dev/ida/.inet/ssh_host_key' [ Found ]
[09:32:31]   Checking for file '/dev/ida/.inet/ssh_random_seed' [ Found ]
[09:32:31]   Checking for file '/dev/ida/.inet/tcp.log'      [ Found ]
[09:32:31]   Checking for directory '/dev/ida/.inet'         [ Found ]
[09:32:31]   Checking for directory '/var/spool/cron/.sh'    [ Not found ]
[09:32:31] Warning: MRK Rootkit                      [ Warning ]
[09:32:32]          File '/dev/ida/.inet/pid' found
[09:32:32]          File '/dev/ida/.inet/ssh_host_key' found
[09:32:32]          File '/dev/ida/.inet/ssh_random_seed' found
[09:32:32]          File '/dev/ida/.inet/tcp.log' found
[09:32:32]          Directory '/dev/ida/.inet' found
[09:32:32]
[09:32:32] Checking for Ni0 Rootkit...
[09:32:32]   Checking for file '/var/lock/subsys/...datafile.../...net...' [ Not found ]
```

```
[09:32:32]   Checking for file '/var/lock/subsys/...datafile.../...port...' [ Not found ]
[09:33:17]
[09:33:17] Checking for Xzibit Rootkit...
[09:33:17]   Checking for file '/dev/dsx'                  [ Found ]
[09:33:17]   Checking for file '/dev/caca'                 [ Found ]
[09:33:17]   Checking for file '/dev/ida/.inet/linsniffer'   [ Found ]
[09:33:17]   Checking for file '/dev/ida/.inet/logclear'     [ Found ]
[09:33:17]   Checking for file '/dev/ida/.inet/sense'        [ Found ]
[09:33:17]   Checking for file '/dev/ida/.inet/sl2'          [ Found ]
[09:33:18]   Checking for file '/dev/ida/.inet/sshdu'        [ Found ]
[09:33:18]   Checking for file '/dev/ida/.inet/s'            [ Found ]
[09:33:18]   Checking for file '/dev/ida/.inet/ssh_host_key' [ Found ]
[09:33:18]   Checking for file '/dev/ida/.inet/ssh_random_seed' [ Found ]
[09:33:18]   Checking for file '/dev/ida/.inet/sl2new.c'     [ Not found ]
[09:33:18]   Checking for file '/dev/ida/.inet/tcp.log'      [ Found ]
[09:33:18]   Checking for file '/home/httpd/cgi-bin/becys.cgi' [ Not found ]
[09:33:18]   Checking for file '/usr/local/httpd/cgi-bin/becys.cgi' [ Not found ]
[09:33:18]   Checking for file '/usr/local/apache/cgi-bin/becys.cgi' [ Not found ]
[09:33:18]   Checking for file '/www/httpd/cgi-bin/becys.cgi' [ Not found ]
[09:33:18]   Checking for file '/www/cgi-bin/becys.cgi'      [ Not found ]
[09:33:19]   Checking for directory '/dev/ida/.inet'         [ Found ]
[09:33:19] Warning: Xzibit Rootkit                           [ Warning ]
[09:33:19]          File '/dev/dsx' found
[09:33:19]          File '/dev/caca' found
[09:33:19]          File '/dev/ida/.inet/linsniffer' found
[09:33:19]          File '/dev/ida/.inet/logclear' found
[09:33:19]          File '/dev/ida/.inet/sense' found
[09:33:19]          File '/dev/ida/.inet/sl2' found
[09:33:19]          File '/dev/ida/.inet/sshdu' found
[09:33:19]          File '/dev/ida/.inet/s' found
[09:33:19]          File '/dev/ida/.inet/ssh_host_key' found
[09:33:19]          File '/dev/ida/.inet/ssh_random_seed' found
[09:33:19]          File '/dev/ida/.inet/tcp.log' found
[09:33:19]          Directory '/dev/ida/.inet' found
[09:33:20]
[09:33:23] Info: Starting test name 'possible_rkt_files'
[09:33:23]   Performing check of possible rootkit files and directories
[09:33:23]     Checking for file '/dev/sdr0'               [ Not found ]
[09:33:23]     Checking for file '/dev/pisu'               [ Not found ]
[09:33:31]     Checking for directory '/dev/ida'             [ Warning ]
[09:33:31]     Checking for directory '/lib/java'          [ Not found ]
[09:33:33] Warning: Checking for possible rootkit files and directories [ Warning ]
[09:33:33]          Found directory '/dev/ida'. Possible rootkit: Rootkit component
[09:33:33]
[09:33:33] Info: Starting test name 'possible_rkt_strings'
[09:33:33]   Performing check for possible rootkit strings
[09:33:33] Info: Using system startup paths: /etc/rc.local /etc/init.d
[09:33:33]     Checking for string 'phalanx'               [ Not found ]
[09:33:37]     Checking for string '/dev/caca'              [ Warning ]
[09:34:01]     Checking for string '/dev/ptyxx/.file'      [ Not found ]
[09:34:01]     Checking for string 'libproc.so.2.0.7'      [ Not found ]
[09:34:01]     Checking for string '/dev/ida/.inet'         [ Warning ]
[09:34:01] Warning: Checking for possible rootkit strings    [ Warning ]
[09:34:01]          Found string '/dev/caca' in file '/bin/netstat'. Possible rootkit: MRK Rootkit
[09:34:01]          Found string '/dev/ida/.inet' in file '/usr/bin/hdparm'. Possible rootkit: Xzibit Rootkit
[09:34:01]
[09:34:01] Info: Starting test name 'malware'
[09:34:02] Performing malware checks
[09:34:02]
[09:34:02] Info: Test 'deleted_files' disabled at users request.
[09:34:02]
[09:34:02] Info: Starting test name 'running_procs'
[09:34:04]   Checking running processes for suspicious files [ Warning ]
[09:34:04] Warning: The following processes are using suspicious files:
[09:34:04]          Command: sshdu
[09:34:04]            UID: 0    PID: 26194
[09:34:04]            Pathname: /dev/ida/.inet/sshdu
[09:34:04]            Possible Rootkit: Unknown rootkit
[09:34:04]
[09:34:04] Info: Test 'hidden_procs' disabled at users request.
[09:34:04]
[09:34:04] Info: Test 'suspscan' disabled at users request.
[09:34:04]
[09:34:04] Info: Starting test name 'other_malware'
[09:34:05]   Performing check for login backdoors
```

```
[09:34:05]    Checking for '/bin/.login'                  [ Not found ]
[09:34:10]
[09:34:10] Performing checks on the network ports
[09:34:11] Info: Starting test name 'ports'
[09:34:11]   Performing check for backdoor ports
[09:34:11]    Checking for TCP port 1524                  [ Not found ]
[09:34:11]    Checking for TCP port 1984                  [ Not found ]
[09:34:16]   Checking for backdoor ports                  [ None found ]
[09:34:16]
[09:34:16] Info: Starting test name 'hidden_ports'
[09:34:19]   Checking for hidden ports                    [ Warning ]
[09:34:19] Warning: Hidden ports found:
[09:34:19]          Port number: 46704
[09:34:20]          Port number: 5353
[09:34:20]          Port number: 631
[09:34:20]          Port number: 68
[09:34:20]          Port number: 6969
[09:34:20]
[09:34:20] Performing checks on the network interfaces
[09:34:20] Info: Starting test name 'promisc'
[09:34:20]   Checking for promiscuous interfaces          [ None found ]
[09:34:20]
[09:34:20] Info: Test 'packet_cap_apps' disabled at users request.
[09:34:20]
[09:34:20] Info: Starting test name 'local_host'
[09:34:20] Checking the local host...
[09:34:20]
[09:34:20] Info: Starting test name 'startup_files'
[09:34:21] Performing system boot checks
[09:34:21]   Checking for local host name                 [ Found ]
[09:34:21]
[09:34:21] Info: Starting test name 'startup_malware'
[09:34:21]   Checking for system startup files            [ Found ]
[09:34:23]   Checking system startup files for malware    [ None found ]
[09:34:24]
[09:34:24] Info: Starting test name 'group_accounts'
[09:34:24] Performing group and account checks
[09:34:24]   Checking for passwd file                     [ Found ]
[09:34:24] Info: Found password file: /etc/passwd
[09:34:24]   Checking for root equivalent (UID 0) accounts   [ None found ]
[09:34:24] Info: Found shadow file: /etc/shadow
[09:34:24]   Checking for passwordless accounts           [ None found ]
[09:34:25]
[09:34:25] Info: Starting test name 'passwd_changes'
[09:34:25]   Checking for passwd file changes             [ None found ]
[09:34:25]
[09:34:25] Info: Starting test name 'group_changes'
[09:34:25]   Checking for group file changes              [ None found ]
[09:34:25]   Checking root account shell history files    [ OK ]
[09:34:25]
[09:34:25] Info: Starting test name 'system_configs'
[09:34:26] Performing system configuration file checks
[09:34:26]   Checking for SSH configuration file          [ Not found ]
[09:34:26]   Checking for running syslog daemon           [ Warning ]
[09:34:26] Warning: The syslog daemon is not running.
[09:34:26] Info: Found rsyslog configuration file: /etc/rsyslog.conf
[09:34:26]   Checking for syslog configuration file       [ Found ]
[09:34:27]   Checking if syslog remote logging is allowed    [ Not allowed ]
[09:34:27]
[09:34:27] Info: Starting test name 'filesystem'
[09:34:27] Performing filesystem checks
[09:34:27] Info: SCAN_MODE_DEV set to 'THOROUGH'
[09:34:27]   Checking /dev for suspicious file types      [ Warning ]
[09:34:28] Warning: Suspicious file types found in /dev:
[09:34:28]          /dev/caca: ASCII text
[09:34:28]          /dev/dsx: ASCII text
[09:34:28]   Checking for hidden files and directories    [ Warning ]
[09:34:28] Warning: Hidden directory found: /dev/ida/.inet
[09:34:28] Warning: Hidden directory found: /dev/.udev
[09:34:28] Warning: Hidden file found: /dev/.initramfs: symbolic link to `/run/initramfs'
[09:34:59]
[09:34:59] Info: Test 'apps' disabled at users request.
[09:34:59]
[09:34:59] System checks summary
[09:34:59] ====================
```

[09:34:59]
[09:34:59] File properties checks...
[09:35:00] Files checked: 135
[09:35:00] Suspect files: 5
[09:35:00]
[09:35:00] Rootkit checks...
[09:35:00] Rootkits checked : 242
[09:35:00] Possible rootkits: 5
[09:35:00] Rootkit names : Devil RootKit, MRK Rootkit, Xzibit Rootkit, Rootkit component, Unknown rootkit
[09:35:00]
[09:35:00] Applications checks...
[09:35:01] All checks skipped
[09:35:01]
[09:35:01] The system checks took: 4 minutes and 54 seconds
[09:35:01]
[09:35:01] Info: End date is Tue Jun 4 09:35:01 BRT 2013

APÊNDICE F – LOG DO CHKROOTKIT REFERENTE AO ROOTKIT XZIBIT

```
vundaxavierteste2@ubuntu:~$ sudo chkrootkit
[sudo] password for vundaxavierteste2:
ROOTDIR is `/'
Checking `amd'...                                   not found
Checking `basename'...                              not infected
Checking `biff'...                                  not found
Checking `chfn'...                                  not infected
Checking `chsh'...                                  not infected
Checking `cron'...                                  not infected
Checking `crontab'...                               not infected
Checking `date'...                                  not infected
Checking `du'...                                    not infected
Checking `dirname'...                               not infected
Checking `echo'...                                  not infected
Checking `egrep'...                                 not infected
Checking `env'...                                   not infected
Checking `find'...                                  not infected
Checking `fingerd'...                               not found
Checking `gpm'...                                   not found
Checking `grep'...                                  not infected
Checking `hdparm'...                                INFECTED
Checking `su'...                                    not infected
Checking `ifconfig'...                              INFECTED
Checking `inetd'...                                 /usr/sbin/chkrootkit: 2837: /bin/ps: not found
not infected
Checking `inetdconf'...                             not found
Checking `identd'...                                not found
Checking `init'...                                  not infected
Checking `killall'...                               not infected
Checking `ldsopreload'...                           not infected
Checking `login'...                                 not infected
Checking `ls'...                                    not infected
Checking `lsof'...                                  not infected
Checking `mail'...                                  not infected
Checking `mingetty'...                              not found
Checking `netstat'...                               INFECTED
Checking `named'...                                 not found
Checking `passwd'...                                not infected
Checking `pidof'...                                 not infected
Checking `pop2'...                                  not found
Checking `pop3'...                                  not found
Checking `ps'...                                    INFECTED
Checking `pstree'...                                not infected
Checking `rpcinfo'...                               not infected
Checking `rlogind'...                               not found
Checking `rshd'...                                  not found
Checking `slogin'...                                not infected
Checking `sendmail'...                              not infected
Checking `sshd'...                                  /usr/sbin/chkrootkit: 2837: /bin/ps: not found
not found
Checking `syslogd'...                               not tested
Checking `tar'...                                   not infected
Checking `tcpd'...                                  /usr/sbin/chkrootkit: 2837: /bin/ps: not found
not infected
Checking `tcpdump'...                               /usr/sbin/chkrootkit: 2837: /bin/netstat: not found
not infected
Checking `top'...                                   INFECTED
Checking `telnetd'...                               not found
Checking `timed'...                                 not found
Checking `traceroute'...                            not found
Checking `vdir'...                                  not infected
Checking `w'...                                     not infected
Checking `write'...                                 not infected
Checking `aliens'...
/dev/caca /dev/dsx
Searching for sniffer's logs, it may take a while...
/dev/ida/.inet/tcp.log
Searching for rootkit HiDrootkit's default files...         nothing found
Searching for rootkit t0rn's default files...               nothing found
Searching for t0rn's v8 defaults...                         nothing found
```

```
Searching for rootkit Lion's default files...          nothing found
Searching for rootkit RSHA's default files...          nothing found
Searching for rootkit RH-Sharpe's default files...          nothing found
Searching for Ambient's rootkit (ark) default files and dirs... nothing found
Searching for suspicious files and dirs, it may take a while... The following suspicious files and directories were found:
/usr/lib/scribes/GenericPlugins/PreferencesGUI/GUI/Window/.goutputstream-SAXB2U
/usr/lib/scribes/GenericPlugins/QuickOpen/.goutputstream-C18XZU   /usr/lib/scribes/GenericPlugins/ThemeSelector/.ropeproject
/usr/lib/scribes/GenericPlugins/SaveDialog/GUI/Window/.goutputstream-SAXB2U          /usr/lib/pymodules/python2.7/.path
/usr/lib/libreoffice/basis3.4/program/.services.rdb
/usr/lib/scribes/GenericPlugins/ThemeSelector/.ropeproject
Searching for LPD Worm files and dirs...          nothing found
Searching for Ramen Worm files and dirs...          nothing found
Searching for Maniac files and dirs...          nothing found
Searching for RK17 files and dirs...          nothing found
Searching for Ducoci rootkit...          nothing found
Searching for Adore Worm...          nothing found
Searching for ShitC Worm...          nothing found
Searching for Omega Worm...          nothing found
Searching for Sadmind/IIS Worm...          nothing found
Searching for MonKit...          nothing found
Searching for Showtee...          nothing found
Searching for OpticKit...          nothing found
Searching for T.R.K...          nothing found
Searching for Mithra...          nothing found
Searching for LOC rootkit...          nothing found
Searching for Romanian rootkit...          nothing found
Searching for Suckit rootkit...          nothing found
Searching for Volc rootkit...          nothing found
Searching for Gold2 rootkit...          nothing found
Searching for TC2 Worm default files and dirs...          nothing found
Searching for Anonoying rootkit default files and dirs...   nothing found
Searching for ZK rootkit default files and dirs...          nothing found
Searching for ShKit rootkit default files and dirs...       nothing found
Searching for AjaKit rootkit default files and dirs...      nothing found
Searching for zaRwT rootkit default files and dirs...       nothing found
Searching for Madalin rootkit default files...          nothing found
Searching for Fu rootkit default files...          nothing found
Searching for ESRK rootkit default files...          nothing found
Searching for rootedoor...          nothing found
Searching for ENYELKM rootkit default files...          nothing found
Searching for common ssh-scanners default files...          nothing found
Searching for suspect PHP files...          nothing found
Searching for anomalies in shell history files...          nothing found
Checking `asp'...          not infected
Checking `bindshell'...          /usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
```

```
/usr/sbin/chkrootkit: 2837: /bin/netstat: not found
not infected
Checking `lkm'...                                   You have   182 process hidden for readdir command
You have   329 process hidden for ps command
chkproc: Warning: Possible LKM Trojan installed
chkdirs: nothing detected
Checking `rexedcs'...         not found
Checking `sniffer'...lo: not promisc and no packet sniffer sockets
eth0: PACKET SNIFFER(/sbin/dhclient[714])
Checking `w55808'...                                  not infected
Checking `wted'...                                  chkwtmp: nothing deleted
Checking `scalper'...                                 /usr/sbin/chkrootkit: 2837: /bin/netstat: not found
not infected
Checking `slapper'...                                 /usr/sbin/chkrootkit: 2837: /bin/netstat: not found
not infected
Checking `z2'...                                    user vundaxavierteste2 deleted or never logged from lastlog!
Checking `chkutmp'...                                 sh: ps: not found
chkutmp: nothing deleted
Checking `OSX_RSPLUG'...                                    not infected
```

APÊNDICE G – INSTALAÇÃO DO SOFTWARE FIX-FU

```
vundaxavierteste1@ubuntu:~$ cd Downloads
vundaxavierteste1@ubuntu:~/Downloads$ ls
delrk.tar.gz  rootkit.tgz
vundaxavierteste1@ubuntu:~/Downloads$ tar -zxvf delrk.tar.gz
fix-fu/
fix-fu/skill
fix-fu/init
fix-fu/snice
fix-fu/top
fix-fu/netstat
fix-fu/ps
fix-fu/watch
fix-fu/backup-fu
fix-fu/w
fix-fu/pgrep
fix-fu/delrk
fix-fu/pkill
vundaxavierteste1@ubuntu:~/Downloads$ ls
delrk.tar.gz  init    pgrep  ps        skill  top  watch
fix-fu        netstat  pkill  rootkit.tgz  snice  w
vundaxavierteste1@ubuntu:~/Downloads$ cd fix-fu
vundaxavierteste1@ubuntu:~/Downloads/fix-fu$ ls
backup-fu  delrk  init  netstat  pgrep  pkill  ps  skill  snice  top  w  watch
vundaxavierteste1@ubuntu:~/Downloads/fix-fu$ sudo ./delrk
¨rootkit fixer to the rescue¨
```

UMA ABORDAGEM FORENSE COMPUTACIONAL VOLTADA À PREVENÇÃO E DETECÇÃO DE ROOTKITS EM SISTEMAS LINUX

Vunda da Conceição Xavier[1], Paulo João Martins[2]

[1]Curso de Ciências da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brasil

[2]Curso de Ciências da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brasil

vunda_xavier@hotmail.com, pjm@unesc.net

Abstract. *This paper describes the conclusion work submitted for obtaining the Degree of Bachelor of Computer Science at the UNESC University, whose goal was to analyze and apply the procedures of forensic computing in Linux operating environment committed to the presence of Rootkits, focusing on analysis of intrusion tests and detection of these malware. To achieve it we performed a literature search as well as tests on Linux environment controlled simulating attacks two binary Rootkits and operating at the user level. It was also described different methods for preventing Rootkits which causes the Linux operating systems will become more insurances and less vulnerable.*

Key-words: *Security, Malwares, Computer Forensics, Rootkits, Linux systems.*

Resumo. *O presente artigo descreve o trabalho de conclusão de curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do extremo Sul Catarinense, cujo objetivo foi analisar e aplicar procedimentos de perícia forense computacional em ambiente operacional Linux comprometido com a presença de Rootkits, com foco na análise de testes de intrusão e detecção destes malwares. Para a realização do mesmo efetuou-se uma pesquisa bibliográfica, bem como testes feitos em ambiente Linux controlado simulando ataques de dois Rootkits binários e que atuam em nível de usuário. Também foi descrito diferentes métodos para a prevenção de rootkits que faz com que os sistemas operacionais Linux se tornem mais seguros e menos encontram vulneráveis.*

Palavras-chave: *Segurança, Malwares, Forense Computacional; Rootkits, Sistemas Linux.*

1. Introdução

Atualmente os sistemas operacionais *Linux* são utilizados para atender diversos serviços dentre eles: hospedagem *web*, *File Transfer Protocol* (FTP), *backup*/restauração, *Domain Name System* (DNS), banco de dados, entre outros. Dada a relevância computacional dos serviços descritos acima surge a necessidade de manutenção constante desses ambientes operacionais para que funcionem adequadamente e, alguns especialistas como administradores de sistemas e peritos

forenses devem possuir conhecimentos mínimos para mantê-los seguros e investigar os possíveis incidentes de segurança quando invasores acessam ou tentam acessá-los.

Para se analisar os possíveis incidentes de segurança é pertinente conhecer-se as técnicas utilizadas pelos invasores em acessos não autorizados aos sistemas assim como os procedimentos para investigar e identificar os suspeitos e suas atividades no sistema. Um exemplo de estratégia usada por possíveis autores de crimes digitais durante invasões de sistemas é a utilização de *Rootkits*. São códigos maliciosos ou *malwares* utilizados na tentativa de invasão de sistemas computacionais, sendo que estes permanecem ocultos no sistema mesmo com a existência de antivírus, *antispyware* e outros *softwares* de segurança (FREITAS, 2006).

Para eficiente detecção dos *Rootkits* e suas atividades maliciosas no sistema, a Perícia Forense auxilia na busca de respostas por meio de procedimentos padrões de investigação. A Forense Computacional destina-se a desvendar as seguintes perguntas pertinentes: O que foi feito? Quando foi feito? Como foi feito? E como se infiltrou? (FREITAS, 2006).

2. Segurança da Informação e Crimes Digitais

Ao longo dos anos, a sociedade tem sido acompanhada por inovações tecnológicas que permitem com que as pessoas usem-nas diariamente no intuito de adquirir novos conhecimentos e trocarem informações. Esta evolução é conhecida também como revolução informática, que possibilita dentre as diversas ações, a substituição das atividades humanas por máquinas; reduzindo acima de tudo o esforço das pessoas (CRESPO, 2011).

Com enfoque computacional, o termo informação está associado à forma com que dados são organizados de maneira que sejam compreendidos e com a finalidade de serem prestativos durante nossa tomada de decisão. Porém, a segurança da informação consiste em três princípios básicos: a confidencialidade, a integridade e a disponibilidade da informação (COSTA, 2011).

A segurança da informação é definida conforme Moraes (2010) como o procedimento que visa a proteção das informações de uso indevido seja de maneira proposital ou acidental por indivíduos pertencentes à organização ou fora dela. As organizações atuais são submetidas a diversas ameaças de segurança no intuito de que estas possam prejudicar as atividades, causando prejuízos, interrupção de serviços, de vendas e atendimentos. Considerando que a informação e todos os sistemas da empresa são pertinentes para o sucesso organizacional, é indispensável estarem protegidos para se conservar a imagem da empresa, seu estatuto, a competitividade, o faturamento e muito mais (COSTA, 2011).

É importante salientar que nenhuma informação, rede ou sistema de segurança é completamente seguro, ou opera sem falhas. Sempre que acessamos a Internet na tentativa de procurarmos mais conhecimentos estamos, ao mesmo tempo, correndo ameaças de segurança; visto que ela tem uma abrangência mundial, vulnerável e sujeita a ataques entre usuários; ataques estes conhecidos como crimes digitais (MORAES, 2010).

Os crimes digitais são as ações realizadas por criminosos na tentativa de acesso a sistemas computacionais ilegalmente; dado que essas condutas englobam

interrupções dos sinais de comunicação, alterações e/ou exclusões de dados e informações, violação de direitos de autoria, estímulo ao ódio e discriminação, menosprezo de crenças religiosas, a disseminação de cenas de pornografia infantil e muito mais. Como o próprio nome já indica, pode-se de maneira mais simplificada conceituar este termo como: atos ilícitos realizados por meio de máquinas computadorizadas, sendo que na maioria dos casos os delitos são praticados via Internet e os meios mais utilizados são os computadores (PAIVA, 2012).

Os criminosos têm adotado diversas maneiras para cometer seus delitos. Uma delas é a utilização dos famosos vírus de computadores; sendo que estes são *softwares* maliciosos e quando instalados ou com acesso a determinada máquina têm a função de destruir ou dificultar o funcionamento de outros programas. Por outro lado, estes *softwares* se aproveitam das falhas de segurança e aumentam a vulnerabilidade nos sistemas operacionais provocando danos sérios no computador. Alguns dos exemplos mais comuns de ferramentas maliciosas são os vírus e *worms*, *Rootkits* e cavalos de tróia (CRESPO, 2011).

3. Perícia Forense Computacional

Vive-se num mundo cada vez mais tecnológico no qual as pessoas têm acesso às mais variadas informações. Com a proliferação de usuários com acesso à Internet e computadores, algumas pessoas têm se aproveitado desses meios para realizarem atividades ilícitas tais como: acesso não autorizado de informações secretas, divulgação de informações ofensivas, fraudes bancárias, racismo, pedofilia dentre outras.

Dada a relevância computacional das atividades descritas, surge a perícia forense como a ciência que se preocupa em investigar e desvendar os criminosos e suas ações ilícitas.

Desta forma, a perícia procura responder perguntas importantes como: O que aconteceu? Quando aconteceu? Por que aconteceu? E, principalmente quem é ou foi o responsável pelo ato? Durante todo o procedimento de investigação é necessário estabelecer uma relação entre as informações recolhidas a partir de fontes distintas, mas que obedecem aos critérios de confiabilidade com a finalidade de que: o incidente seja explorado com maior precisão, se tenha uma visão mais ampla do ocorrido, se elimine as incertezas nos dados. É de salientar que a ausência de informações acarreta consequências e não prova se determinado ato tenha acontecido ou não (MACEDO, 2010).

“A Perícia forense é a aplicação de conhecimentos em informática e técnicas de investigação com a finalidade de obtenção de evidências” (FREITAS, 2006, p. 1).

Há diversas ferramentas forenses que podem ser utilizadas em uma análise forense computacional. Entretanto, os responsáveis pela perícia devem definir de antemão quais *softwares* existem e que atendem as suas necessidades. É responsabilidade destes, escolher os programas em função do tipo de caso a ser investigado. As ferramentas de perícia são comerciais ou gratuitas. Entre os principais softwares destacam-se: *Helix*, *EnCase*, *Autopsy*, *FCCU*, *CAINE*, *DEFT*, *PeriBr*, *Forensic Toolkit*, *Forensic Digital Toolkit – UbuntuBr* (QUEIROZ; VARGAS, 2010).

Por conseguinte, pode-se afirmar que todo e qualquer ato considerado ilegal e ao mesmo tempo cometido utilizando-se recursos computacionais, é suscetível de uma perícia e, por conseguinte, deve ser feita por profissionais especialistas na área

computacional ou peritos forenses por meio de uma sequência de processos bem estipulados e aceitos judicialmente. Sendo assim a forense computacional deve utilizar técnicas investigativas para obtenção de evidências criminosas aceitas em lei como é o caso de espionagens empresariais e a utilização imprópria de *softwares* de determinada empresa, dentre outras (CAIXETA, 2011).

4. Sistema Operacional Linux e Rootkits

Os peritos forenses têm utilizado com bastante frequência os ambientes *Linux* no intuito de realizarem suas atividades investigativas. Este sistema é eficiente e vantajoso na medida em que oferece diversas soluções para a forense computacional. A utilização do *Linux* em casos envolvendo perícia forense de crimes digitais tem os seus benefícios. Um dos pontos vantajosos deste ambiente operacional é o fato de existir a possibilidade de criarem-se unidades de dados acessados apenas em modo de leitura, sendo que não necessita de um bloqueador independente para que as informações e/ou dados armazenados em disco não sejam modificados. Por outro lado pode-se afirmar que é um sistema ajustável que possui diversos sistemas de arquivos próprios para serem executados em projetos de perícia forense. Existem diversas distribuições *Linux* e várias ferramentas de perícia para esse sistema operacional (REYES; WILES, 2007, tradução nossa).

4.1. Rootkits

Esta-se na era da informação por meio da qual não se deve confiar 100% em computadores. Por exemplo, como é que tem-se certeza que o nosso fabricante de *hardware* não escondeu código malicioso no *microchip* do sistema? Ou que o nosso sistema operacional recém-instalado não contém *backdoors* criados por um desenvolvedor desonesto da equipe de programação? O fato é que não se pode confiar plenamente que nossos computadores estão totalmente livres de *malwares* (METULA, 2011, tradução nossa).

Infelizmente a necessidade de usar-se um computador supera a falta de confiança a este respeito. De salientar que, *malware* é um pedaço de *software* projetado para realizar atividades maliciosas na máquina da vítima sem o consentimento desta. *Malware* é uma expressão utilizada computacionalmente para especificar *softwares* mal intencionados tais como vírus, cavalos de Tróia, *backdoors*, *worms* e *Rootkits*. Basicamente qualquer tipo de código projetado para causar dano ou espionar as atividades de uma vítima. Uma vez que o *malware* é instalado, a intenção do atacante é permanecer despercebido tanto tempo quanto possível, mantendo o controle do sistema (METULA, 2011, tradução nossa).

*Rootki*t é conceituado como sendo um *kit* de ferramentas utilizado por *hackers* e *crackers* com objetivo obter acesso a computadores de forma ilegal. Esses *softwares* são utilizados no intuito ocultar arquivos e processos, o que possibilita aos invasores à permanência de conexão ao sistema, e praticando atividades criminosas. Existem *Rootkits* para diversos ambientes operacionais com destaque para *Linux*, *Solaris* e *Microsoft Windows*. Dentre as funcionalidades dos *Rootkits* destacam-se (AVILA, 2010, tradução nossa):

a) ocultação de outros *softwares* em execução no sistema;

b) possibilita a ocultação de arquivos armazenados e processos em execução;

c) permite a ocultação de diretórios importantes do sistema;

d) ocultação de portas que possibilita o atacante acessar determinado computador por meio da Internet.

Seguindo Macedo (2010) os *Rootkits* escondem-se dos usuários e gerentes de sistemas no intuito de que não haja suspeita de invasores infiltrados que posteriormente poderão ter permissão de r*oot*. Ainda segundo o autor, os *Rootkits* ocultam informações relevantes no sistema, destacando-se:

a) arquivos de configurações do sistema;

b) *softwares*, arquivos e portas de segurança;

c) procedimentos e conexões estabelecidas pelos atacantes;

d) acesso de logs realizados durante as ações dos invasores.

4.1.1 Tipos: nível de usuário e kernel

Existem dois tipos comuns de *Rootkits*: em nível de usuário e de *kernel* (MACEDO, 2010).

Os *Rootkits* em nível de usuário são executados em sistemas e contexto de segurança de um usuário no sistema. Por exemplo, se alguém estiver logado em sua estação de trabalho como um usuário qualquer e não tiver permissões administrativas, o *Rootkit* irá filtrar e dar acesso a determinadas portas para todas as aplicações em execução na conta. Normalmente a maioria das contas de usuário também têm privilégios administrativos, sendo assim, um *Rootkit* em modo usuário oculta as suas ações no sistema operacional e pode impedir o funcionamento de alguns processos em execução (DAVIS; BODMER; LEMASTERS, 2010, tradução nossa).

Estes *Rootkits* podem também ser chamados de *user level*, *user mode* ou *application level*. São os tipos mais simples e são classificados em *Rootkits* binários e de biblioteca. Os binários caracterizam-se por fazer parte das funcionalidades dos outros *softwares* instalados; já os rootkits de biblioteca desempenham o papel de substitutos das bibliotecas mais importantes do ambiente operativo (MACEDO, 2010).

4.1.2 Prevenção

Uma das estratégias que os usuários e administradores de sistemas têm adotado como defesa contra os famosos *Rootkits* é prevenir-se destes, pois, após o acesso não autorizado por meio de *Rootkits*, o sistema operacional torna-se mais vulnerável ou inseguro (MACEDO, 2010).

A primeira forma de prevenção contra os *Rootkits* é a não utilização destes, o que indica a possibilidade de evitarem-se acessos indevidos no sistema. A não utilização dos *Rootkits* refere-se ao fato destes *malwares* instalarem-se acidentalmente através de *websites* suspeitos e inseguros com presença desses programas maliciosos. Recomenda-se também a adoção de procedimentos que envolve configuração segura e manutenção constante dos sistemas operacionais. Outras formas de prevenção seguidas são (MACEDO, 2010):

a) devem-se desativar os processos e *softwares* desnecessários que estão sendo executados no sistema;

b) controles especializados: consiste na utilização de *firewall*, antivírus, *anti-malwares* e *anti-rootkits* no intuito de que se alguma destas ferramentas for monitorada por *Rootkits*, outras poderão travar o ataque;

c) execução constante de testes no sistema em busca de agentes intrusos que comprometem a segurança do sistema.

4.1.3 Detecção

Quando tem-se casos de intrusão de sistemas por meio de *Rootkits*, a melhor maneira de detê-los é através da utilização de ferramentas *anti-rootkits*; com especial destaque para os *softwares chkrootkit* e o *rootkit hunter*. Entretanto esses dois programas são os mais usados em ambientes *Linux* e são *open source* (MACEDO, 2010).

5. Análise Comportamental dos Rootkits Fuckit e Xzibit em Ambiente Linux

Rootkits são ferramentas que após instaladas no sistema, têm o poder de esconderem-se em arquivos e diretórios. A funcionalidade destes depende essencialmente da maneira como foi criado para atuar em determinado ambiente operacional. Entretanto, também podem instalar *keyloggers* e *backdoors* que, facilitam acesso irrestrito de usuários. Por outro lado, *Rootkits* são normalmente instalados por usuários com privilégios administrativos do sistema ou da instalação acidental de *softwares* e/ou *drivers* de dispositivo com *trojans* compilados em suas configurações; e, essas ferramentas maliciosas tendem a manter controle do sistema (WMLUG, 2009, tradução nossa).

5.1 Proposta de um ambiente de teste

Com a finalidade de obterem-se informações sobre o comportamento dos *Rootkits* em ambiente *Linux*, foram criadas máquinas virtuais com a ferramenta de virtualização *VMware*. As máquinas virtuais foram instaladas utilizando a distribuição *Ubuntu* 11.10.

Pode-se conceber que as máquinas virtuais visem criar cópias do sistema a ser periciado, sendo que, estas encontram-se isoladas do computador real que executa a máquina virtual. O isolamento entre as diversas máquinas é vantajoso na medida em que o mau funcionamento e/ou falha crítica de uma não afeta o desempenho das restantes máquinas. É ainda imprescindível afirmar que nestes ambientes todas as máquinas virtuais criadas estão interligadas a apenas uma máquina virtual (neste caso o *VMware* utilizado durante os testes).

Inseriu-se no *VMware* as ferramentas *Backtrack 5*, *PeriBr*, *FDTK-Ubuntu-Br* e o *Ubuntu 11.10*; todas elas em formato *ISO*. A partir do terminal *Linux* das máquinas virtuais criadas foi instalado os *Rootkits fuckit* e *xzibit,* ambos operam em nível de usuário e são binários. Após a instalação destes *malwares*, verificaram-se os ambientes virtuais no intuito de: identificar a presença dos *Rootkits* e buscar sinais deixados por estes no sistema. A verificação e/ou checagem foi feita com os *softwares chkrootkit* e *Rootkit hunter*.

Chkrootkit é uma ferramenta utilizada no intuito de identificar sinais deixados por *Rootkits*. O *chkroottkit* tem as seguintes funcionalidades: verifica os arquivos do sistema para encontrar alterações feitas por *Rootkits*, faz uma análise da interface

de rede, procura por sinais de *trojans* e também verifica as possíveis alterações efetuadas por *Rootkits* em arquivos e diretórios do sistema.

O *Rootkit hunter* verifica e procura pelas assinaturas de vários tipos de *Rootkits* em ambientes *Linux*; encontra-se licenciado por meio da GPL. Entretanto, o *Rootkit hunter* possui as seguintes funcionalidades: comparação de *hash MD5*, analisa e verifica arquivos constantemente utilizados por *Rootkits*, checa permissões incorretas para arquivos binários, faz um exame de *strings* suspeitas nos módulos *LKM* e *KLD*, verifica e procura por arquivos ocultos e faz uma análise completa em arquivos textos e binários (JUNIOR; MARANGON, […2005]).

5.2 Resultados obtidos e taxonomias observadas

Com o auxilio do *chkrootkit* foi possível checar o sistema em busca de sinais deixados pelos *Rootkits fuckit e xzibit* e infecções causadas por estes. Nos logs do *chkrootkit* abaixo é possíver observar-se que os comandos *ifconfig*, *netstat*, *hdparm*, *top* e o prototocolo de acesso remoto de correio eletrônico (*pop3*) estão infectados. Estas inconsistências encontradas foram causadas pelo *Rootkit xzibit*.

Outra opção para detecção dos *Rootkits fuckit* e *xzibit* foi o *software Rootkit hunter*. Esta ferramenta foi capaz de identificar e mostrar com certeza e precisão a presença destes *Rootkits*; o que faz com que se torne peça fundamental em respostas a incidentes de segurança envolvendo *Rootkits*, por ser um *software* eficaz e eficiente na detecção desses *malwares*. A seguir, é possível observar-se os *logs* desta ferramenta em busca dos *Rootkits fuckit* e *xzibit*. Ambos foram detetados assim como diversos diretórios infectados por estes.

5.2.1 Log do chkrootkit

vundaxavierteste1@ubuntu:~$ sudo chkrootkit

Searching for Suckit rootkit... Warning: /sbin/init INFECTED

Checking `hdparm'... INFECTED

Checking `ifconfig'... INFECTED

Checking `netstat'... INFECTED

Checking `ps'... INFECTED

Checking `top'... INFECTED

5.2.2 Log do Rootkit hunter referente ao Rootkit fuckit

[15:40:29] Running Rootkit Hunter version 1.3.8 on ubuntu

[15:40:29]

[15:40:29] Info: Start date is Tue Jun 4 15:40:29 BRT 2013

[15:40:29]

[15:42:41] Warning: Fuck`it Rootkit [Warning]

[15:44:49] System checks summary

[15:44:49] ==========

[15:44:50]

[15:44:50] File properties checks...

[15:44:50] Files checked: 132

[15:44:50] Suspect files: 8

[15:44:50]

[15:44:50] Rootkit checks...

[15:44:50] Rootkits checked : 242

[15:44:50] Possible rootkits: 2

[15:44:50] Rootkit names : Fuck`it Rootkit, Tuxtendo Rootkit

[15:44:51]

[15:44:51] Applications checks...

[15:44:51] All checks skipped

[15:44:51]

[15:44:51] The system checks took: 4 minutes and 18 seconds

[15:44:51]

[15:44:51] Info: End date is Tue Jun 4 15:44:51 BRT 2013

5.2.3 Log do Rootkit hunter referente ao Rootkit xzibit

[09:30:02] Running Rootkit Hunter version 1.3.8 on ubuntu

[09:30:02]

[09:30:02] Info: Start date is Tue Jun 4 09:30:02 BRT 2013

[09:30:02]

[09:33:19] Warning: Xzibit Rootkit [Warning]

[09:34:59]

[09:34:59] System checks summary

[09:34:59] ==========

[09:34:59]

[09:34:59] File properties checks...

[09:35:00] Files checked: 135

[09:35:00] Suspect files: 5

[09:35:00]

[09:35:00] Rootkit checks...

[09:35:00] Rootkits checked : 242

[09:35:00] Possible rootkits: 5

[09:35:00] Rootkit names : Devil RootKit, MRK Rootkit, Xzibit Rootkit, Rootkit component, Unknown rootkit

[09:35:00]

[09:35:00] Applications checks...

[09:35:01] All checks skipped

[09:35:01]

[09:35:01] The system checks took: 4 minutes and 54 seconds

[09:35:01]

[09:35:01] Info: End date is Tue Jun 4 09:35:01 BRT 2013

6. Considerações Finais

O presente artigo foi uma demonstração de como os *Rootkits* binários operam em determinado sistema e como as equipes de resposta a estes incidentes atuam no intuito de definir estratégias para detecção destas ameaças. Foi desenvolvido em ambiente *Linux* controlado onde os ataques dos *Rootkits fuckit* e *xzibit* foram pré-definidos.

Dentre as técnicas computacionais que foram adotadas para auxiliar a investigação forense computacional de *Rootkits* destacam-se a virtualização e a usabilidade das ferramentas *chkrootkit* e *Rootkit hunter* em busca de evidências digitais deixadas por *Rootkits*.

Portanto, este artigo científico trouxe informações pertinentes de como usuários *Linux* poderão atuar em casos de intrusão de *Rootkits* binários, adotando a utilização de máquinas virtuais e ferramentas que auxilie na detecção destas intrusões que comprometem a segurança dos sistemas operacionais *Linux*.

7. Referências

AVILA, Alex. **Rootkits**. 2010. Disponível em: <http://www.slideshare.net/alex.avila1976/rootkit-3226171#btnNext>. Acesso em: 24 nov. 2012.

CAIXETA, Thiago Fernandes Gaspar. **Perícia Forense Digital:** A tecnologia da informação como ferramenta estratégica no combate ao crime. 2011. 63 f. Trabalho de Conclusão de Curso (Especialização) - Curso de Gestão de Segurança da Informação, Universidade Fumec, Belo Horizonte, 2011.

COSTA, Marcelo Antonio Sampaio Lemos. **Computação Forense.** 3. ed. Campinas: Millennium, 2011. 158 p.

CRESPO, Marcelo Xavier de Freitas. **Crimes Digitais.** São Paulo: Saraiva, 2011. 242 p.

DAVIS, Michael; BODMER, Sean; LEAMASTERS, Aaron. **Hacking Exposed Malware & Rootkits: Malware & Rootkits Security Secrets & Solutions**. New York: McGraw-Hill. 2010.

JUNIOR, Claudio Penasio; MARANGON, Sílvio Luís. **PERÍCIA FORENSE COMPUTACIONAL EM SISTEMAS ALTERADOS POR ROOTKITS**. […2005] Disponível em: < http://www.lsi.usp.br/~penasio/trabalhos/PSI5007-3009850-5223770-1-V1.10.pdf>. Acesso aos: 05. Mai. 2013.MACEDO, Guilherme

Matte. **Investigação Forense Digital de Rootkits em Sistemas Unix.**2010. 71 f. Trabalho de Conclusão de Curso (Graduação) - Curso de Ciência da Computação, Departamento de Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2010. Disponível em: <http://www.lume.ufrgs.br/handle/10183/26345>. Acesso em: 15 out. 2012.

METULA, Erez. **Managed Code Rootkits.** Massachusetts: Elsevier, 2011. 316 p.

MORAES, Alexandre Fernandes de. **Segurança em Redes**: fundamentos. São Paulo: Érica. 2010.

PAIVA, Raphael Rosa Nunes Vieira. **CRIMES VIRTUAIS.** 2012. 55 f. Monografia de Conclusão de Curso (Bacharel em Direito) - Curso de Direito, Departamento de Instituto de Ciências Sociais, Centro Universitário do Distrito Federal – Udf, Brasília, 2012. Disponível em: <http://www.conteudojuridico.com.br/monografia-tcc-tese,crimes-virtuais,37145.html>. Acesso em: 10 out. 2012.

QUEIROZ, Claudemir; VARGAS, Raffael. **Investigação e perícia forense computacional.** Rio de Janeiro: Brasport, 2010. 136 p.

REYES, Anthony; WILES, Jack. **Cybercrime and Digital Forensics**. Massachusetts. Elsevier. 2007.

WMLUG (revista). **Rootkits**. Jul. 2009.

Disponivel em: <http://www.wmlug.org/pdf/Rootkits.pdf>. Acesso aos: 02. Mai. 2013.