Álvaro Vilobaldo Rios da Silva

Antiforense com uso de rootkits

São Paulo

2013

Álvaro Vilobaldo Rios da Silva

Antiforense com uso de rootkits

Monografia de Conclusão de Curso apresentada à Universidade Presbiteriana Mackenzie de São Paulo como requisito parcial à obtenção do título de Especialista em Computação Forense do Curso Lato Sensu em Computação Forense

Orientador:
Ivete Irene dos Santos

UNIVERSIDADE PRESBITERIANA MACKENZIE

São Paulo

2013

Monografia sob o título Antiforense com uso de rootkits, desenvolvida por Álvaro Vilobaldo Rios da Silva, e aprovada em 30 de março de 2013, São Paulo capital, pela banca constituída por:

Ivete Irene dos Santos
Orientadora

Ana Cristina Azevedo
Orientadora

Epígrafe

"Se você conhecer o inimigo e a si mesmo, não precisa temer o resultado de uma centena batalhas. Se você se conhecer a si mesmo, mas não o inimigo, para cada vitória você também sofrerá uma derrota. Se você não conhecer nem o inimigo, nem a si mesmo, você sucumbirá em todas as batalhas." (TZU, IV a.C)

Resumo

Esta monografia é um estudo sobre antiforense com utilização de rootkits visando mostrar como funciona um rootkit e como ele pode ser usado na antiforense. Definições de antiforense e rootkits com suas práticas comuns e respectivos históricos junto com algumas técnicas de subversão demonstrarão que o uso combinado de técnicas antiforenses incorporadas a rootkits cria mecanimos de destruíção de provas de forma sistemática e eficiênte.

Abstract

Write here the English version of your ’Resumo’...

Lista de Figuras

Lista de Tabelas

Sumário

Introdução

A monografia apresenta uma incursão na antiforense focada na utilização de rootkits em ambientes Microsoft principamente Windows, visando mostrar seu funcionamento, as práticas mais comuns e como o rootkit pode ser usado na antiforense.

O rootkit é em geral utilizado por atacantes avançados com fins maliciosos e oferece grande obstáculo para ser detectado, pois seu uso força o perito a ter um conhecimento amplo além de usar técnicas sofisticadas. Entender o seu funcionamento é determinande para uma pericia bem sucedida. Para desenvolver esse tema foi utilizada vasta bibliografia, exemplos práticos e aplicações de rootkits comuns.

Ao longo dos capítulos são desenvolvidos os conhecimentos para o entendimento do tema proposto, onde cada um foi dividido em um assunto específico. No primeiro capítulo será mostrado uma visão ampla do que é a ciência forense e como ela é aplicada na computação. No segundo capítulo são apresentados conceitos de antiforense e os tipos de antiforense. Por fim no terceiro capítulo conduz a uma visão do funcionamento de um rootkit e algumas técnicas para subverter os sistema.

1 *Forense Computacional*

A humanidade sempre teve assuntos divergentes, onde em outrora esses assuntos eram resolvidos com o mais forte ou hapto sobrepujando a vontade doutro, com o advento da civilização se convencionou o uso de um mediador que em teoria deve ser alguem neutro.

Quando trata-se de assuntos de disputa de interesse, espera-se que o mediador busque e chegue o mais próximo possível da verdade antes de tomar uma decisão. Hoje é dado esse poder de mediação ao juiz, contudo em diversos conflitos se faz necessário conhecimentos técnicos especificos. Nesses casos ele é auxiliado por um especialista ou perito no assunto técnico em discussão. Para tal, esse especialista utiliza-se da ciência forense para trazer luz os fatos respaldando a decisão do juiz sobre o assunto.

Segundo Houaiss e Villar (2009) ciência é um "corpo de conhecimentos sistematizados adquiridos via observação, identificação, pesquisa e explicação de determinadas categorias de fonômenos e fatos, e formulados metódica e racionalmente." e forense é "relativo aos tribunais e à justiça". Logo as ciências forenses é a utilização da ciência "à análise de vestígios, no intuito de responder às demandas judiciais" (VELHO; GEISER; ESPINDURA, 2012, p. -3).

De certo a área médica foi a primeira a ser requisitada em tribunais construindo técnicas que levaram ao desenvolvimento ao longo dos anos da medicina legal. No Império Romano médicos eram chamados para lucidar mortes diz França (2008), outro exemplo histórico da importância do parecer técnico pode ser visto no *Código Criminal Carolino* feito em 1532 por Carlos Magno que definia a análise médica em determinados crimes.

Com o avanço da tecnologia e o valor da pespectiva especializada, foi natural que outras áreas também fossem utilizadas no foro. A computação forense, apesar da tecnologia e inovação inatas, conceitualmente ainda se propõe a grosso modo a usar a ciência para mostrar fatos demandados judicialmente, como podemos ver em Machado e Eleutério (2011), "[...] computação forense tem como objetivo principal determinar a dinâmica, a materialidade e autoria de ilícitos ligados à área da informática, tendo como questão principal a identificação e o processamento de evidências digitais em provas materiais [...], por métodos técnicos-científicos, conferindo-

lhes validade probatória em juízo.”.

2 *Antiforense Computacional*

Harris (2006) define antiforense como método para prevenir ou agir contra a ciência usada a favor das leis civis e criminais que são aplicadas por órgãos como a polícia. Berinato (2007), amplia essa ideia mostrando que é mais que uma técnica usada, é uma abordagem crimosa. Assim todas as tentativas de afetar negativamente a existência, quantidade e / ou qualidade da evidência de uma cena de crime, ou fazer a análise e exame das provas difíceis ou impossíveis de realizar, para Rogers (2005) será considerada antiforense. Dessa forma concluímos resumidamente que a antiforense computacional pode ser definida como qualquer ação praticada para obstruir, dificultar ou destruir evidências ou provas no âmbito computacional.

Dentre as modalidades de antiforense computacional, Blunden (2009a) as categoriza em cinco grandes grupos, sendo eles: destruição de dados, ocultação de dados, corrupção de dados, fabricação de dados e eliminação da fonte de dados.

3 *Rootkits*

Malware é a junção das palavras em inglês *malicious* e *software*, termos em inglês que pode ser traduzido livremente como aplicativo malicioso. Dessa forma podemos considerar como malware os vírus, worms, trojans, botnets kit dentro outros, contudo o inverso não é valido, ou seja, todo vírus é um malware, mas nem todo malware é um vírus.

Os vírus e worms são feitos para se espalha a diferença está em como eles se espalham. O vírus precisa ser ativado ou executado pelo usuário e em geral fica atrelado a algum executavel que pode ser ou não um programa legitimo subvertido (LUDWIG, 1995), ao contrário do worm que não precisam da ação direta do usuário para se espalhar e permanece apenas na memória (MCAFEE, 2013).

Uma *botnet* é uma rede de computadores controlados por cybercriminosos(KASPERSKY, 2013). Basicamente quando um computador é infectado pelo agente da *botnet* o mesmo se torna parte da rede e é controlado pelo dono da botnet.

No universo *UNIX*[1] ou *UNIX-like*[2] a conta de usuário com menor restrição de segurança é referênciada como conta *root* sendo que em alguns sistemas o nome de usuário é literalmente root, mas isso é apenas uma convenção histórica do que uma imposição (BLUNDEN, 2009b). Enquanto *kit* significa conjunto de peças (SANTOS, 2000).

Rootkit pode ser visto como um "kit" composto de pequenos programas úteism por exemplo, binários, scripts, arquivos de configuração que permitem a um atacante manter o acesso "root". Em outras palavras, um rootkit é um conjunto de programas e de código, que permite a presença permanente ou consistente, não detectável em um computador (HOGLUND; BUTLER, 2005).

Essa coleção de ferramentas que permitem aos invasores ocultarem suas atividades em um computador, de modo que eles podem secretamente monitorar e controlar o sistema por um período prolongado, ou seja, existem três serviços que o são inerentes dos rootkits, ocultação,

[1] Um sistema operacional multiusuário amplamente utilizado.
[2] Sistemas Operacionais baseados no Unix, como o GNU/Linux

comando e controle (C2) e vigilância (BLUNDEN, 2009b). Ocultação, o rootkit deve passar despercebido, sem que o usuário e/ou antivírus o detecte; vigilância ou monitoramento basicamente consiste em acompanhar as ações do usuário, o rootkit tem que ser capaz de saber o que o usuário está fazendo; e comando e controle, permite ao dono do rootkit o controle remoto sobre o mesmo, definindo suas ações e direcionando. Desses três serviços o mais importante para o rootkit é a furtividade, pois um rootkit detectável vai durar muito pouco.

Como pode ser visto cada um desses agentes subversivos tem definições e características próprias diferentes e uma em comum todos eles subvertem o sistema de alguma forma, contudo é bom resaltar que nem todo rootkit é um malware, pois existem aplicações legítimas para o mesmo, como de computadores coorporativos e inclusive aplicações investigativas.

Os primeiros rootkits apareceram em de 20 anos atrás no fim dos anos 80 e início dos 90, quando alguns foram percebidos comportamentos anormais em computadores como espaço em disco utilizado sem identificação, conexões de rede não-listadas e uso anormal do CPU (ROSANES, 2011).

Basicamente o processador pode receber dois tipos de exceções, uma gerada por hardware chamada de externa e interrupções geradas por programas(INTEL, 2011, p. 6-2 Vol. 3A). A grosso modo, quando uma interrupção ou exceção ocorre o processador usa o endereço de memória armazenado no indice correspondente a interrupção lançada que por definição aponta para uma *procedure* que trata a interrupção (INTEL, 1986), ou seja, esse endereço na memória vai direcionar o processamento a *procedure* especifica para a interrupção ou exceção que ocorreu. O rootkit pode interceptar uma chamada a tabela estrutura de dados com indices dessas tratativas como *Interrupt Vector (IVT)* em modo real ou a *Interrupt Descriptor Table (IDT)* em modo protegido, ambas as estruturas possuem funções similares, apesar da forma como trabalham ser muito distinta.

tipos de rootkit http://www.terena.org/activities/tf-csirt/meeting27/oesterberg-rootkits.pdf

falando assim parece uma tarefa fácil realizar qualquer um desses metodos...

Jamie Butler, the creator of the FU rootkit

O uso de rootkits é limitado a usuários avançados sendo que seu desenvolvimento exige conhecimentos profundos tanto de arquitetura de computadores quanto do funcionamento do Sistema Operacional que o mesmo vai corromper.

Rootkits proporcionam uma grande variedades de opções e pode ser utilizado para quase tudo,

vem sendo utilizado historicamente para subverter sistemas dando permissão

O fato de alguns eventos e rootkits terem ficados famosos, como é o caso do stuxnet(), torna esse assunto muito interessante de ser abordado, porque

A grosso modo rootkits sempre tentarão esconder sua presença e seus rastros, logo a anti-forense desde o inicio já é parte ...

não são novidades e sua aplicação já não é

e durante o avanço do trabalho, poderão ser notadas diversas formas de detecção de rootkits além de indícios de seu uso em análises.

4 *Antiforense com Rootkits*

Quando o perito vai realizar a coleta do material para a análise, ele pode encontrar dois cenários o corpo de delito que na ciência forense computacional quase sempre consiste em computadores, pode estar ligado ou desligado. Quando o mesmo está ligado é possível realizar a análise viva, entretando a análise *live* de um ambiente com rootkit pode gerar dados falsos ou imprecisos.

Análise *live* ou análise viva, consiste em uma análise focada em extrair e examinar dados volatéis (MCDOUGAL, 2006), esses dados volatéis são perdidos ao desligar o computador, como por exemplo, o contéudo de registradores do processador, dados na *cache*, dados na memória etc. Essa abordagem possui diversas vantagens entre elas estão extração de dados voláteis como já citado, triagem de equipamentos, triagem de dados, preservação de dados criptografado e a possiblidade de estabelecer flagrante (MESQUITA; HOELS; RALHA, 2011).

É importante ressaltar que toda ferramenta ou *hardware* utilizado para a coleta de dados em equipamentos que ainda estão em execução, vai depender de dados fornecidos pelo equipamento periciado, ou seja, em algum momento essas ferramentas forenses vão requisitar dados de um sistema comprometido e passível de alguma interceptação mesmo quando a solução utilizada para coleta é um hardware (RUTKOWSKA, 2007). Logo em um ambiente dominado por um rootkit não é um ambiente confiável, pois ele pode interceptar e alterar chamadas das ferramentas de análise, impossibilitando uma coleta real e fiel.

A metodologia empregada na análise de um ambiente supostamente com um rootkit varia muito, mas em termos gerais a análise *Post mortem* é menos arriscada e indicada para casos de rootkits. As medidas antiforense que podem ser tomadas pelo rootkit dependem do ambiente de execução e engloba também a equipe que o administra. No melhor dos cenários para o atacante é quando o sistema é administrado por leigos, não capacitados, sobrecarregados e/ou os sistemas não são atualizados com frequencia tornando mais fácil para o rootkit esconder-se exigindo menos do atacante. No pior dos cenários o administradores são altamente qualificados, realizam rotinas frequentes, esta sempre atualizado e é auditorado (BLUNDEN, 2009a). Entender isso

pode indicar ao perito o quão complexo o rootkit é, pois quanto mais próximo do pior cenário mais avançado tem que ser o rootkit o que pode caracterizar um *Advanced persistent threat (APT)* ou ameaça avançada e persistente, ou seja, o ataque utilizando o rootkit foi orquestrado e planejado apenas para esse alvo específico.

Não existe técnica 100% antiforense, então o objetivo do atacante é tornar a análise cansativa para induzir ao erro podendo adicionar inclusive falsos rootkits ou malwares para despistar, iludir ou levar a conclusões precipitadas. Em (BLUNDEN, 2009a) são mostrados passos para a análise *Post mortem* onde se suspeita de rootkit:

- Clone do *hard disk drive (HDD)*;
- Recuperar arquivos;
- Coletar metadados dos arquivos;
- Retirar arquivos conhecidos;
- Análise estática dos executáveis suspeitos desconhecidos; e
- Análise dinâmica dos executáveis suspeitos.

Esses passos pode ser vistos como filtros, onde cada passo retira cada vez mais ruído, eliminando em cada etapa arquivos superfulos. A seguir serão descritos cada um dos passos e alguns dos problemas que o mesmo pode gerar.

4.1 Cópia forense do HDD

A primeira fase e mais importante é a coleta, que nesse caso é clone do HDD. O clone do HDD, pode ser feito com software, como o dd[1] ou por hardware específico para clonagem de HDDs. Por via de regra essas ferramentas são homologados por órgãos como *National Institute of Standards and Technology (NIST)* que atestam a qualidade e eficácia dessas ferramentas em produzir cópias fidedignas que podem ser usadas em disputas legais como evidências. Basicamente todos essas ferramentas fazem cópias de setor por setor de todo o disco, para que o perito possa trabalhar nessa cópia preservando o HDD original. Essas cópias já foram exaustivamente usadas em tribunais e são muito úteis e indispensáveis para a análise do perito. Hoje sua eficácia não é questionada, pois essa cópia ou clone tem os dados idênticos ao do HDD objeto de análise

[1] http://pubs.opengroup.org/onlinepubs/9699919799/utilities/dd.html

e essa igualdade pode ser atestada com *checksums* como os algoritmo de *hash MD5* ou *SHA1* (MULVEY, 2007). Essa abordagem pode ter alguns empecilhos e o rootkit pode não vir a ser clonado junto com os dados do HDD.

Primeiro e mais óbvio é que o rootkit pode nem estar no disco clonado, como estar somente na memória ou ser alocado em um outro dispositivo. O rootikit poder ser projetado para ficar somente na memória, entretando isso pode impedir que o mesmo persista já que é apagado sempre que o computador desliga (MEDINA, 2009). Contudo o mesmo esteja num parque de máquinas que permita infectar o computador toda vez que este for iniciada, como em grandes *data centers* em geral nunca tem todas as máquinas desligadas o rootkit pode ficar vivo na rede e persistir infectando novamente as máquinas assim que elas iniciarem.

Num *Personal Computer (PC)* durante o *boot* ele inicia o *Basic Input/Output System (BIOS)* fornece suporte básico para os principais periféricos como teclado e vídeo em modo texto e inicia o *Power-On Self-Test (POST)* que além de realizar alguns testes inicia, com base das configurações do SETUP salvas na memória CMOS, todos os circuitos perífericos, vídeo, Sistema Operacional e passa o controle para o sistema operacional (TORRES, 2001). Durante todo esse processo o computador é executado em *Real Mode*, e como já foi descrito anteriormente, em modo real os programas não são isolados uns dos outros como no *Protected Mode*, logo qualquer programa em execução pode acessar todo conteúdo na memória. O rootkit pode persistir no *firmware*[2] de algum periférico, como a placa de vídeo (ECONOMOU; JUAREZ, 2012) ou placa de rede (BLANCO; EISSLER, 2012), dessa forma que tornariam o clone do HDD inútil além de ter acesso em *Real Mode* na inicialização do computador.

O perito deve ter em mente ainda que o rootkit pode estar em áreas reservadas do disco como *host protected area (HPA)* ou *Device configuration overlay (DCO)*, essas areas são desenvolvidas para não serem modificadas ou mesmo acessadas por usuários, BIOS ou Sistema Operacional (GUPTA et al., 2006). Dessa forma dependendo da ferramenta utilizada para clone do HDD pode ser que essas áreas não sejam clonadas ((NIST, 2008a) ou (NIST, 2008b)), inutilizando o clone do HDD.

Com o disco clonado o perito deve se ater a outros fatores, como criptografia, arquivos salvos em estruturas não convencionais etc. O disco clonado pode ter *full disk encryption* o que significa que todos os dados no disco estão criptografados (RUBENS, 2012) e que ao clonar o clone, como é uma cópia fidedigna do original, também tem todos os dados criptografados. Logo o disco deverá ser decriptado antes da análise. É improvável que um rootkit criptografe todo o disco da vítima, pois chamaria muita atenção e iria de encontro a uma de suas premissas

[2]Todo programa salvo numa *Read-only memory (ROM)* é chamado de firmware

que é se ocultar no sistema. Então quando disco está encriptado, ele provavelmente foi feito pela vítima e consequentimente a maior interessada no trabalho do perito facilitando a decriptação do disco. O mais comum nesses casos quando envolve criptografia é do rootkit encriptar somente o seu binário, mas isso vai ser discutido adiante na análise estática.

4.2 Recuperar arquivos

deletados, busca por assinaturas de binarios (pode mudar os headers, como por exemplo segundo o Windows PE Specification os primeiros 16-bits tem o magic number 0x4D54 (ASCII MZ de MArk zibibowski http://msdn.microsoft.com/en-us/magazine/cc301805.aspx) http://msdn.microsoft.com/en-us/windows/hardware/gg463119.aspx transmorgrify by mafia), data carve, fragmentos de arquivos, data streams alternativos (feature stream do NTFS), slack space e arquivos ocultos;

http://www.blackhat.com/presentations/bh-federal-06/BH-Fed-06-Thompson/BH-Fed-06-Thompson-up.pdf

4.3 Coletar metadados dos arquivos

hash, MAC, path, name, size; criando uma timeline Criar uma timeline de eventos com base nos MAC dos arquivos ajuda a entender o ciclo de contaminação, além de levantar suspeitas sobre varios dados inconsistentes, como por exemplo, porque o papel de parede do Windows tem um MAC tem a data de modificação ou criação mais recente? buscar como o rootkit pode ter se mantido no ambiente analisado,

4.4 Retirar arquivos conhecidos

base de hash de arquivos conhecidos bons e ruins (assinatura de malwares conhecidos);

4.5 Análise estática dos executáveis suspeitos desconhecidos

cryptors e packers. colocar o binário numa forma encriptado e comprimido que as tools nao interpretam, o stub (esbolço) decodifica o payload e munda o controle do executável para o entry point do binario original. usado pelo skype (http://www.blackhat.com/presentations/bh-europe-06/bh-eu-06-biondi/bh-eu-06-biondi-up.pdf)

UPX _ merge secxtions (.text, .data, .idata, etc) em um unico sectior upx1, onde tem uma stub que unpack o original.

funcionamento do upx

com a toool dumpbin da para ver as sections do upx

upx0 é espaço vazio upx1 onde esta o stub e o executavel compactado, ao descompactar o executavel ele move para os endereços do upox (na low memory), assim o mesmo é decompoilado e executado e com o tempo ele sobreescreve tbm os endereços onde atnes estavam o upx1 (http://upx.sourceforge.net/)

embebed virtual machine, o rootkit esta em bytecode que sao interpretados pela vm em runtime, a tool nem mesmo o processador tem como saber como interpretar o bytecode sem a vm

compinação de ambos o stub decodifica a vm que interpreta o bytecode

basicamente o stub tem a chave para decodificar o binario, mas isso pode ser mais dificil quando crypter e packer privado, pode dividir o codigo em diversos setores e cada setor ter uma chave criptogradada diferente SALT + HWID + TIME + PCI_ROM para criar a chave

4.6 Análise dinâmica dos executáveis suspeitos

antidebugger FS flag determina o debug na arquitetura IA32, explicar que assim e processador faz uma instrução por vez

Windows SDK duas rotinas, WINAPI isDEBUggerPresent(void) user-mode ou KdRefreshDebuggerNotPresent() Kernel-mode (being debugged PEB http://www.symantec.com/connect/articles/wind
anti-debug-reference

antivirtualização, para testar qualquer código malicioso deve-se

Referências Bibliográficas

BERINATO, S. *The Rise of Anti-Forensics*. 2007. Disponível em: <http://www.csoonline.com/article/221208/the-rise-of-anti-forensics>.

BLANCO, A.; EISSLER, M. One firmware to monitor ém all. Eko Party 2012, 2012. Disponível em: <http://www.ekoparty.org//archive/2012/BlancoEissler_2012-paper.pdf>.

BLUNDEN, B. Anti-forensics: The rootkit connection. Black Hat USA 2009, 2009. Disponível em: <http://www.blackhat.com/presentations/bh-usa-09/BLUNDEN/BHUSA09-Blunden-AntiForensics-PAPER.pdf>.

BLUNDEN, B. *The rootkit arsenal*. 1ª. ed. Plano, Texas: Wordware Publishing, 2009. ISBN 978-1-59822-061-2.

ECONOMOU, N. A.; JUAREZ, D. Vga persistent rootkit. Eko Party 2012, 2012. Disponível em: <http://www.ekoparty.org/archive/2012/VGA_Persistent_Rootkit.pdf>.

FRANçA, G. V. *Medicina Legal*. 8ª. ed. Rio de Janeiro - RJ: Guanabara-kooga, 2008. ISBN 978-85-7302-9635.

GUPTA, M. R. et al. Hidden disk areas: Hpa and dco. *International Journal of Digital Evidence*, 2006. Disponível em: <http://www.utica.edu/academic/institutes/ecii/publications/articles/EFE36584-D13F-2962-67BEB146864A2671.pdf>.

HARRIS, R. Arriving at an anti-forensics consensus: Examining how to define and control the anti-forensics problem. *Digital investigation 3S (2006) s4-s49*, 2006. Disponível em: <http://www.dfrws.org/2006/proceedings/6-Harris.pdf>.

HOGLUND, G.; BUTLER, J. *Rootkits: Subverting the Windows Kernel*. 1ª. ed. [S.l.]: Addison Wesley Professional, 2005. ISBN 0-321-29431-9.

HOUAISS, A.; VILLAR, M. de S. *Dicionário Houaiss da língua portuguesa*. 1ª. ed. Rio de Janeiro - RJ: Objetiva, 2009. ISBN 978-85-7302-9635.

INTEL. Intel 80386 programmer's reference manual 1986. Intel, 1986. Disponível em: <http://css.csail.mit.edu/6.858/2011/readings/i386.pdf>.

INTEL. Intel 64 and ia-32 architectures software developerś manual, volume 3a: System programming guide, part 1. Intel, 2011. Disponível em: <http://download.intel.com/design/processor/manuals/253668.pdf>.

KASPERSKY. *Computer Security FAQ*. 2013. Disponível em: <http://www.kaspersky.com/threats_faq>.

LUDWIG, M. A. *The rootkit arsenal*. 1ª. ed. Show Low, Arizona: American Eagle Publications, Inc., 1995.

MACHADO, M. P.; ELEUTéRIO, P. M. da S. *Desvendando a Computação Forense*. 1ª. ed. [S.l.]: Novatec, 2011. ISBN 978-85-7522-260-7.

MCAFEE. *Glossary*. 2013. Disponível em: <http://home.mcafee.com/virusinfo/glossary>.

MCDOUGAL, M. Live forensics on a windows system: Using windows forensic toolchest(wft). Fool Moon - Sofware Security, 2006. Disponível em: <http://www.foolmoon.net/downloads/Live_Forensics_Using_WFT.pdf>.

MEDINA, P. Österberg. Detecting rootkits in memory dumps. *PTS*, Associação Brasileira De Especialistas Em Alta Tecnologia (ABEAT), 2009. Disponível em: <http://www.terena.org/activities/tf-csirt/meeting27/oesterberg-rootkits.pdf>.

MESQUITA, F. I.; HOELS, B. W. P.; RALHA, C. G. Raciocínio baseado em casos aplicado em análise live. Associação Brasileira De Especialistas Em Alta Tecnologia (ABEAT), 2011. Disponível em: <http://www.icofcs.org/2011/papers-published-007.html>.

MULVEY, B. *Hash Functions*. 2007. Disponível em: <http://home.comcast.net/ bretm/hash/>.

NIST. Test results for digital data acquisition tool: Encase linen 6.01. National Institute of Justice, 2008. Disponível em: <https://www.ncjrs.gov/pdffiles1/nij/224147.pdf>.

NIST. Test results for digital data acquisition tool: Ftk imager 2.5.3.14. National Institute of Justice, 2008. Disponível em: <https://www.ncjrs.gov/pdffiles1/nij/222982.pdf>.

ROGERS, D. M. Anti-forensic presentation given to lockheed martin. San Diego, 2005.

ROSANES, P. Rootkits. *GRIS - Grupo de Resposta a Incidentes de Segurança. Universidade Federal do Rio de Janeiro*, 2011. Disponível em: <http://www.gris.dcc.ufrj.br/documentos/artigos/rootkits-survey>.

RUBENS, P. *Buyer's Guide to Full Disk Encryption*. 2012. Disponível em: <http://www.esecurityplanet.com/mobile-security/buyers-guide-to-full-disk-encryption.html>.

RUTKOWSKA, J. Beyond the cpu: Defeating hardware based ram acquisition (part i: Amd case). Black Hat DC 2007, 2007. Disponível em: <http://www.blackhat.com/presentations/bh-dc-07/Rutkowska/Presentation/bh-dc-07-Rutkowska-up.pdf>.

SANTOS, C. M. *Dicionário Inglês-Português*. 2ª. ed. Santa Terezinha, São Paulo: Rideel, 2000.

TORRES, G. *Hardware Curso Completo*. 4ª. ed. Rio de Janeiro - RJ: Axcel Books do Brasil Editora, 2001. ISBN 85-7323-165-3.

TZU, S. *A Arte da Guerra*. Tradução de Candida de Sampaio Bastos. 1ª. ed. [S.l.]: Golden Books, IV a.C. 47 p. ISBN 978-85-7501-272-7.

VELHO, J. A.; GEISER, G. C.; ESPINDURA, A. *Ciências Forenses*. 1ª. ed. Campinas - SP: Millenniumn, 2012. ISBN 978-85-7625-249-8.