Honeynets: Invasores, Ferramentas, Técnicas e Táticas.

E.C.M.G. Jabour,
Universidade Federal do Rio de Janeiro (UFRJ), COPPE-Poli, GTA,
Brasil, CEP 24.210-240

Otto C. M. B. Duarte,
Universidade Federal do Rio de Janeiro (UFRJ), COPPE-Poli, GTA,
Brasil, CEP 24.210-240

{eugenia, otto}@gta.ufrj.br

Resumo. Honeynets *são ferramentas de pesquisa que consistem de uma rede projetada exclusivamente para ser comprometida. Uma vez comprometida, ela é utilizada para observar o comportamento dos invasores, suas táticas, ferramentas e motivações. Neste artigo serão introduzidos os conceitos básicos de* Honeynets*, seus tipos, suas vantagens e desvantagens e um exemplo de implementação.*

1. Introdução

Atualmente, existem, na Internet, uma grande variedade de programas que visam comprometer os computadores que estão permanentemente conectados à rede. Estes servidores podem se tornar alvos de vários tipos de ataques, o que os impede temporariamente ou permanentemente de atender às requisições normais de serviço. Isto exige, dos administradores de sistemas, monitoramento constante.

Estes administradores usam vários métodos para proteger suas redes. O uso de *firewalls* é um destes métodos. Eles são usados para controlar o tráfego entre a rede local e a Internet. Baseam-se nos serviços requisitados e nos endereços de origem e destino dos pacotes para decidir se vão ou não deixar que o tráfego passe pela rede local.

Outro método usado para detectar possíveis ataques maliciosos à rede é o chamado Sistema de Detecção de Intrusão (IDS - Intrusion Detection System). Seus sensores devem ser colocados em vários pontos da rede, principalmente nos pontos críticos que fazem a divisa da rede local com a Internet. O IDS se baseia em comportamentos pré-definidos de eventos maliciosos, chamados assinaturas.

O uso de IDS e *firewalls* garante um certo nível de segurança aos administradores de sistemas. Contudo, um *firewall* possui alguns problemas, como, por exemplo, não impedir ataques oriundos da própria rede local. Acredita-se também que,

em certos casos, o IDS falha em prover segurança e só aumenta a complexidade de gerenciamento.

Surge, então, a necessidade de uma nova camada de segurança na rede, e [5] propõe que o uso de *Honeynets* venha preencher esta lacuna.

Este trabalho está organizado da seguinte maneira: na segunda seção tem-se um pequeno histórico, na terceira seção são apresentados os conceitos de *Honeynet*, e seus principais tipos. Na quarta seção, descreveremos um exemplo de como implementar uma *Honeynet*. Na quinta seção as conclusões obtidas com este trabalho.

2. Histórico

Segundo [6] a primeira referência à implementação de mecanismos de acompanhamento das atividades de invasores data de 1988, quando Clifford Stoll tornou públicou a história da invasão ocorrida nos sistemas do Lawrence Berkeley Laboratory (LBL). Neste caso, ao invés de fechar as portas de acesso para o invasor, ele tomou a decisão de acompanhar e registrar todos os seus passos, com a intenção de conseguir rastrear a origem do ataque. Este acompanhamento levou quase um ano e revelou não só a origem do ataque, mas também os motivos do atacante e quais eram as redes em que ele estava interessado.

Em 1992, Bill Cheswick publicou um artigo descrevendo o acompanhamento de uma invasão em um dos sistemas da AT&T, que havia sido projetado especificamente para ser invadido. Steven Bellovin também participou deste projeto, desenvolvendo ferramentas que foram utilizadas tanto como armadilhas quanto para capturar as ações do invasor. Em 1998 Fred Cohen desenvolveu o *Deception Toolkit* (DTK), a primeira ferramenta de código aberto cujo objetivo é explicitamente iludir atacantes.

Esta ferramenta emula diversas vulnerabilidades e coleta as informações sobre os ataques sofridos. Nesta época surgiu o termo *honeypot* como definição para um recurso de segurança preparado especificamente para ser sondado, atacado ou comprometido e para registrar essas atividades.

Após o surgimento do DTK diversas outras tecnologias de *honeypots* foram desenvolvidas, incluindo diversos produtos comerciais como o *Cybercop Sting*, o *NetFacade* e o *NFR BackOfficer Friendly* .

O Projeto *Honeynet* começou em 1999 como uma lista de discussão informal composta por um pequeno grupo de indivíduos. Em pouco tempo este grupo percebeu que para analisar as informações geradas pelos ataques era preciso um número maior de pessoas. A lista de discussão cresceu e passou a contar com vários especialistas da área. Em junho de 2000 este grupo ganhou o nome de *Honeynet Project*. Anos depois, o grupo foi formalizado como uma organização não governamental. Ainda assim, possuindo cerca de 30 membros, o grupo não dispunha de recursos suficientes para pesquisar e desenvolver múltiplas *Honeynets*. Em Janeiro de 2002 foi formada a *Honeynet Research Alliance* que hoje inclui mais de dez organizações no Brasil, Grécia, Índia, México, Irlanda e Estados Unidos.

Os pesquisadores e profissionais, membros do projeto, projetaram uma rede especificamente para ser comprometida com o objetivo de revelar as ferramentas, táticas e motivações dos invasores.

3. Conceitos

As *Honeynets* são um tipo de *honeypot*. Conceitualmente, *honeypot* é um recurso de segurança, especificamente criado para ser atacado. Não possui qualquer atividade produtiva ou autorizada. Logo, qualquer pacote que entrar ou sair deste sistema deve ser considerado como um vírus, *scan* ou ataque.

A principal vantagem de um *honeypot* é que ele não depende de assinaturas, regras ou algoritmos avançados, ele simplesmente captura os ataques que lhe são direcionados, facilitando a análise e correlação da pequena, mas valiosa, coleção de dados.

Entretando, pode-se citar duas desvantagens. A primeira, diz respeito à sua capacidade limitada de visão, ou seja, ele só captura os dados direcionados a ele mesmo. Não é capaz de reconhecer, por exemplo, ataques a um servidor web, ou a um servidor de dados, se este tráfego não for direcionado a ele. A segunda refere-se ao risco. Toda vez que uma nova tecnologia é introduzida, especialmente se ela possui a pilha IP, existe o risco deste recurso ser quebrado, ou no caso específico dos *honeypots*, ser usado para atacar outros sistemas. Em função destas questões, os *honeypots* não devem ser usados para substituir as tecnologias de segurança existentes, mas sim, para complementá-las e aumentar seu valor.

Para melhor entender os *honeypots*, deve-se ressaltar suas duas categorias: produção e pesquisa. O principal objetivo de um *honeypot* de produção é proteger a organização, distraindo o invasor. Em contrapartida, o *honeypot* de pesquisa tem como principal objetivo capturar o maior número possível de informações sobre os ataques.

A *Honeynet* é essencialmente um conjunto de *honeypots* de pesquisa, ou seja, é uma ferramenta de pesquisa que consiste de uma rede projetada especificamente para ser comprometida. Uma vez comprometida, é utilizada para observar o comportamento dos invasores, possibilitando a realização de análises detalhadas das ferramentas utilizadas, de suas motivações e das vulnerabilidades exploradas [6].

Segundo [2], tradicionalmente, os métodos de segurança são puramente defensivos. *Firewalls*, IDS e criptografia são exemplos de mecanismos utilizados defensivamente para proteger recursos. A estratégia é defender a organização, o melhor possível, detectar qualquer falha que possa vir a acontecer na defesa e reagir a estas falhas. As *Honeynets* vão tentar mudar esta estratégia defensiva obtendo informações a respeito dos ataques já existentes. Novas ferramentas podem ser descobertas, padrões de ataques podem ser determinados e a movimentação dos atacantes pode ser estudada. Estas informações podem ser usadas contra futuros ataques.

Existem dois principais tipos de *Honeynets*, as clássicas e as virtuais. As clássicas são construídas usando sistemas reais (físicos), em cada um de seus *honeypots*. Este tipo não é muito usado em função do maior preço e do aumento da dificuldade de administração. As *Honeynets* Virtuais funcionam baseadas em um grupo de sistemas virtuais, ou seja, emuladores de sistemas operacionais, criando um ambiente de produção virtual que não possui as desvantagens do anterior.

Para construir com sucesso uma *Honeynet*, é preciso tratar dois requisitos críticos: Controle e Captura de Dados. Qualquer falha nestes requisitos implica em uma falha geral.

O Controle de Dados [2] é o controle de toda a atividade do invasor. Quando se está trabalhando com invasores, sempre existe risco, e nós devemos minimizá-lo. Deve-se garantir, principalmente, que, uma vez comprometido, um *honeypot*, não possa ser usado para danificar qualquer outro sistema. Todavia, o desafio é controlar o fluxo de dados sem que os invasores desconfiem.

Uma vez que o sistema está comprometido, os invasores freqüentemente abrem conexões com a internet para obter ferramentas, conectar-se em canais IRC ou enviar mensagens. É preciso dar a eles flexibilidade suficiente para executar estas tarefas, mesmo porque, são estes passos que precisamos analisar e aprender. Além disso, os invasores podem ficar desconfiados se não conseguirem abrir conexões externas. Em geral, quanto mais flexibilidade eles possuem, mais consegue-se aprender com eles, contudo, maior torna-se o risco.

Segundo [7] alguns requisitos mínimos devem ser seguidos para garantir o controle de dados:

O Controle de Dados pode ser implementado através de respostas automáticas ou intervenção manual. Além disso, devem existir duas camadas de controle de dados, para proteger contra falhas. Todavia, se ainda assim acontecerem falhas o sistema deve automaticamente bloquear todos os acessos de entrada e saída para o *honeypot*.

Manter o estado de todas as conexões de saída e entrada é imprescindível. Além disso controlar as conexões sem que os invasores percebam, torna-se um desafio.

Controle de dados deve ser constantemente monitorado pelo administrador, inclusive remotamente e os alarmes automáticos devem ser enviados quando o *honeypot* estiver comprometido.

A Captura de Dados [2] é a captura de todos as informações geradas pelo invasor. Estas informações serão estudadas posteriormente, possibilitanto o aprendizado de ferramentas, táticas e motivos da comunidade invasora. O desafio é capturar o maior número possível de dados sem deixar que o invasor perceba. Isto é feito através de pequenas modificações nos *honeypots.* As informações capturadas não devem ser armazenadas localmente, pois, podem ser detectadas e destruídas pelo invasor.

Um terceiro requisito, a Coleção de Dados, deve ser usado por organizações que possuam mais de uma *Honeynet*. Os dados gerados por cada uma destas redes podem ser combinados para aumentar seu valor.

De acordo com [2] existem dois tipos de *Honeynets* que podem ser desenvolvidos em uma rede: Primeira ou Segunda Geração. O tipo a ser usado depende de vários fatores, tais como disponibilidades dos recursos, tipos de ataque e invasores que deseja-se detectar e efetivamente da experiência metodológica que se possui sobre o assunto.

3.1. Primeira Geração

A Primeira Geração implementa o Controle e Captura de Dados de maneira simples, mas efetiva. Os iniciantes nesta tecnologia devem começar sua implementação por esta geração. Ela provavelmente não será tão efetiva como a segunda, mas devido a

sua simplicidade e à quantidade de testes efetuada durante os últimos anos, é a mais recomendada para esta situação.

A figura 1 representa um mapa detalhado de uma *Honeynet* Primeira Geração.

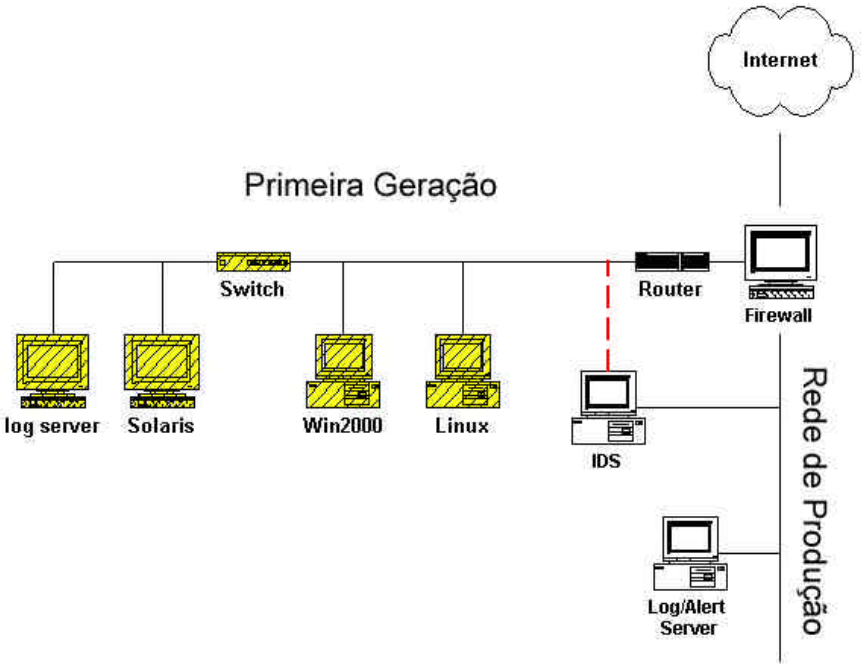

Figura 1 – Primeira Geração de *Honeynets*

Na figura 1 pode-se notar um *firewall* da camada três separando a rede em três sub-redes distintas: a Rede Administrativa, a própria *Honeynet* e a Internet. Qualquer pacote que entrar ou sair da *Honeynet* terá que passar tanto pelo *firewall* quanto pelo roteador.

O *Firewall* é a ferramenta primária de controle das conexões de entrada e saída. Ele permite todas as conexões de entrada na *Honeynet* e permite conexões de saída até um limite estipulado. Isto dá ao invasor a flexibilidade necessária para fazer o que quer que ele precisa sem permitir o abuso. Não existe um número exato de conexões externas estipuladas, tudo vai depender das funcionalidades que se deseja explorar. Experiências feitas pelo Projeto *Honeynet* demonstram que de 5 a 10 conexões por hora são suficientes para um bom resultado.

Existem várias ferramentas diferentes para controlar as conexões de saída da *Honeynet*. A primeira ferramenta testada pelo Projeto *Honeynet* foi um *firewall* chamado *CheckPoint FireWall-1* que trabalhava em conjunto com um script que contava quantas conexões de saída estavam sendo abertas. Quando o limite era alcançado, o script enviava um alerta para o administrador e configurava o *firewall* para não aceitar mais conexões originadas pelo *honeypot* comprometido.

Outra ferramenta testada foi a de nome *IPTables*. É um *firewall* de código aberto que pode ser usado também para controlar as conexões de saída da *Honeynet*, bloqueando-as ao atingir seus limites previamente configurados. A terceira opção foi o

OpenBSD´s packet filter, que funcionava da mesma maneira que as ferramentas anteriores.

O roteador é colocado entre o *firewall* e a *Honeynet* por duas razões. A primeira é que o roteador esconde o *firewall*, criando uma situação mais realista quando um *honeypot* for comprometido e seus invasores encontrarem um roteador entre eles e a rede externa. A segunda razão do posicionamento do roteador é trabalhar como um segundo ponto de controle de acesso ajudando o *firewall* a evitar ataques oriundos da *Honeynet*.

O roteador funciona como uma segunda camada de controle de conexões. Primariamente, ele evita *spoofs*, *DoS* ou ataques baseados em ICMP permitindo que somente os pacotes com IP de origem iguais ao da *Honeynet* saiam. Nenhuma *Honeynet* deve depender de uma única camada para garantir o Controle dos Dados.

Uma vez implementado o Controle dos Dados, o próximo passo é a Captura. O mesmo *firewall* usado anteriormente pode ser usado agora, para capturar dados. Ele deve manter um registro sobre todas as conexões iniciadas e terminadas na *Honeynet*, uma vez que, todas estas conexões são suspeitas. Estes registros serão usados posteriormente para descobrir novos padrões de ataques chamados assinaturas.

O Sistema IDS possui dois propósitos. O primeiro é capturar toda a atividade da rede e o segundo é emitir alertas quando seus sensores detectarem qualquer atividade suspeita. È importante salientar que os sistemas IDS baseiam-se em assinaturas armazenadas em bases de dados, ou seja, quando um pacote da rede coincide com uma das assinaturas ele é considerado suspeito. Os sensores IDS disponibilizam informação detalhada sobre uma conexão específica. Uma ferramenta interessante para implementar o IDS é o *Snort*, disponível em: http://www.snort.org.

A terceira camada da Captura de Dados é a dos sistemas operacionais. Deve-se capturar toda a atividade do sistema e do usuário. A primeira maneira de fazer isto é manter todos os registros (logs) gerados pelo sistema operacional em um servidor remoto.

Não existe a necessidade de esconder o servidor remoto de registros. Se o invasor detectar sua existência, o pior que ele pode fazer é desabilitá-lo (o que normalmente é feito pela maioria dos invasores). Isto significa que não será feito mais nenhum registro, contudo, a esta altura, o servidor de registros já armazenou informações de como o invasor teve acesso à *Honeynet* e qual a sua origem.

Invasores mais avançados vão tentar comprometer os registros armazenados no servidor. Esta é mais uma tarefa que se deseja registrar, pois, como normalmente o servidor de registros é a parte mais segura do sistema, o invasor terá que usar técnicas mais avançadas, que pode-se capturar e estudar. Se o servidor de registros for comprometido, não se perde nada, uma vez que o IDS também esteve capturando e armazenando todas as atividades na rede.

Uma segunda maneira de capturar dados do sistema é utilizar capturadores de teclado e tela, e enviar remotamente os dados para o servidor de registros. Apesar de efetivo, este método pode ser facilmente descoberto com um *sniffer*.

3.2. Segunda Geração

Os mecanismos descritos anteriormente, para Controle e Captura de Dados, são efetivos, mas podem ser melhorados. O objetivo principal da Segunda Geração de *Honeynets* é criar uma solução mais fácil de se implementar e mais difícil de se detectar. Isto é feito através da criação de respostas mais inteligentes e flexíveis às ações dos invasores.

Em uma *Honeynet* de segunda geração, o Controle, a Captura de Dados e a coleção destes dados vão acontecer em um único dispositivo. Isto vai tornar mais fácil a implantação e manutenção da rede. Este dispositivo é um *gateway* da camada dois, isto é, uma ponte (*bridge*), sem endereço IP. Ele não faz roteamento nem decrementa o TTL(*Time to live*) dos pacotes. Com isso, os invasores não conseguem perceber que o tráfego está sendo analisado. A segunda vantagem é que, como um *gateway*, todo o tráfego de entrada e saída passa por ele. A figura 2 representa um *Honeynet* de Segunda Geração.

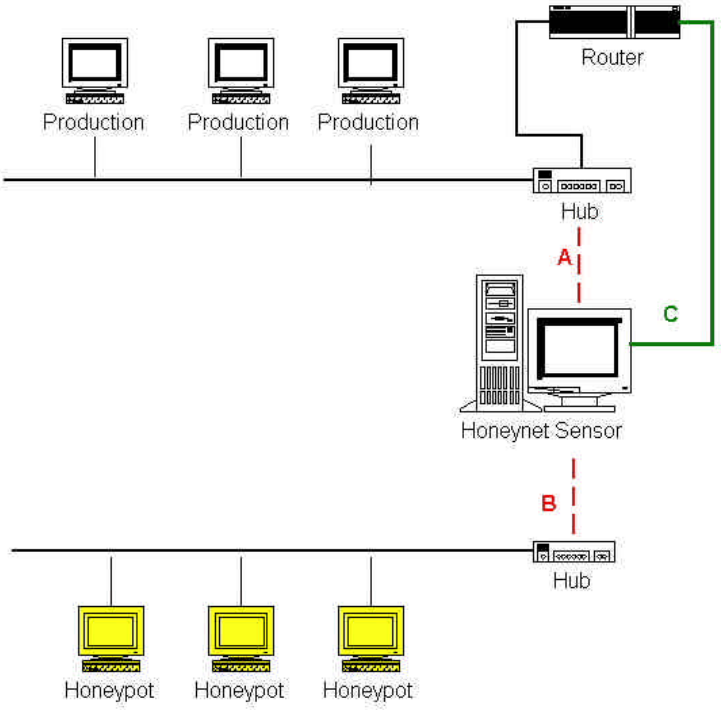

Figura 2 – Uma *Honeynet* de Segunda Geração

O Sensor *Honeynet* mencionado na figura 2 representa um único sistema responsável por Controlar e Capturar os dados (*gateway*). Este equipamento possui três interfaces, duas delas (A e B) funcionando na camada 2, ou seja, como um comutador (*switch)*. A terceira interface possui um endereço IP para possibilitar conexões remotas, atendendo não só a Coleção de Dados como também à administração.

A Segunda Geração de *Honeynets* fez um grande avanço no Controle de Dados. Já não é mais preciso limitar o número de conexões originadas pela *Honeynet*, podendo-se detectar que tipo de operação está sendo feita pelo invasor e, assim, responder especificamente a estas ações.

A maneira de responder às operações não autorizadas também melhorou. Não se deve bloquear os ataques, mas sim modificá-los. Desta maneira, o ataque sai da *Honeynet*, mas não é eficaz. Esta modificação é feita quando os pacotes passam pelo *gateway* da camada dois.

Na Captura de Dados também aconteceu grande avanço em relação à geração anterior. Os dados, agora, são capturados no nível do *kernel*, através do driver dos dispositivos tty, e não na camada de rede como acontecia na Primeira Geração. Desta maneira os dados são capturados antes de ser criptografados.

3.3. *Honeynet* Virtual

As *Honeynets* consomem muitos recursos, são difíceis de se construir e complexas para se manter. Estes problemas podem ser minimizados com as *Honeynets* virtuais. Esta solução permite a execução de todas as ferramentas necessárias em um só computador. O Termo virtual é usado, porque os diferentes sistemas operacionais contidos no software de virtualização parecem estar rodando em seus próprios equipamentos. Esta tecnologia tem vantagens e desvantagens que veremos a seguir.

Custo reduzido e gerenciamento fácil são as principais vantagens da *Honeynet* virtual. Antes, era preciso adquirir, por exemplo, oito computadores diferentes para construir uma *Honeynet* completa, agora, precisa-se apenas de um equipamento. Todavia, esta simplicidade possui seu custo. Primeiro, o software de virtualização limita o hardware e o tipo de sistema operacional que você pode simular. Segundo, um invasor pode vir a comprometer o software de virtualização, e assim, comprometer a totalidade da *Honeynet*. Por fim, se os invasores quebrarem os sistemas contidos na *Honeynet* virtual, eles estarão aptos a determinar todos os sistemas que estão rodando no ambiente virtual.

As *Honeynets* de primeira e segunda gerações podem ser implementadas virtualmente. Além disso, existem dois outros tipos de *Honeynets* Virtuais, as Híbridas e as Auto-Contidas. A seguir pode-se encontrar a descrição de cada um destes tipos.

A Auto-Contida está inteiramente condensada em um só computador. Normalmente uma *Honeynet* é tipicamente composta por um *gateway* que Controla e Captura Dados além dos *honeypots.* Neste caso, todos estes componentes estão fisicamente em um único computador. A *Honeynet* Híbrida possui *firewall*, IDS e servidor de registros em máquinas isoladas. A figura 3 a seguir representa uma *Honeynet* Auto-Contida.

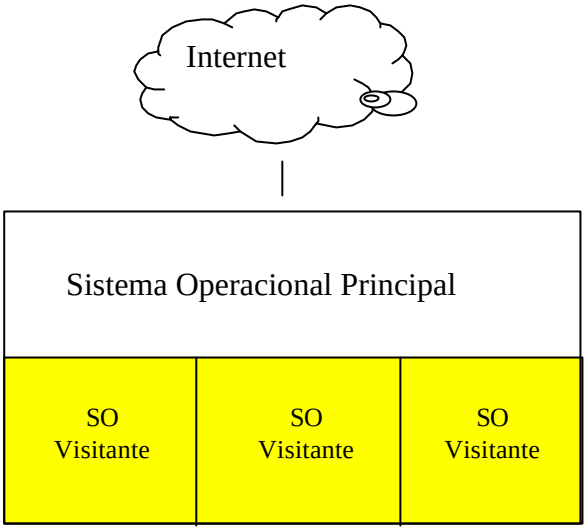

Figura 3 – *Honeynet* Auto-Contida

4. Como construir uma *Honeynet*

Depois de definir os conceitos de *Honeynets* Virtuais, podemos especificar as possíveis soluções para implementá-las. Não existe uma solução melhor do que a outra, cada uma possui suas vantagens e desvantagens.

Para exemplificar este trabalho, descreveremos como foi construída uma *Honeynet* Virtual de Primeira Geração, utilizando softwares com código aberto. Esta implementação está detalhadamente descrita em [10].

Este exemplo será dividido em cinco partes: a primeira explica o que é *User-Mode Linux* (UML) e como ele trabalha. A segunda descreve como criar uma rede com vários sistemas em um único computador. Na terceira parte descreveremos como Controlar Dados em uma *Honeynet* UML usando *IPTables*. E finalmente como Capturar os dados.

4.1. User Mode Linux

User-Mode Linux (UML) é uma solução de código aberto que possibilita a criação de uma máquina virtual. UML nos permite executar várias instâncias do Linux no mesmo sistema e ao mesmo tempo. Deve-se usar esta funcionalidade para criar uma *Honeynet* em um único computador. Esta solução é similar à ferramenta comercial *Vmware*. Todavia, UML está atualmente limitada ao Linux.

O sistema operacional usado será *Red Hat* versão 7.3. Qualquer outra instância deste sistema operacional será chamada de visitante. Nenhum *software* de virtualização será necessário, contudo, uma correção (chamada UML *patch*) deverá ser aplicada ao *Kernel* de cada um dos visitantes.

Deve-se ressaltar que o UML tem a capacidade de detectar todos capturadores de teclado introduzidos pelos invasores, mesmo quando eles usam criptografia. Ele faz isto através do driver dos dispositivos tty.

Quando um invasor ganha acesso a um sistema operacional virtual, ele está apto a determinar se este ambiente é ou não um *honeypot*. O UML minimiza este risco com a habilidade de fazer o sistema operacional virtual parecer-se com um real. A terceira característica interessante do UML é que ele pode rodar o *Kernel* UML em uma máquina totalmente separada de seus processos, aumentando sua segurança.

Para instalar o UML bastar usar um pacote pré-compilado com terminação RPM. RPM é um tipo especial de arquivo, executável, capaz de instalar e desinstalar pacotes de software no sistema operacional Linux. Ao executarmos este pacote pré-compilado estaremos instalando o sistema operacional visitante, além de um conjunto de ferramentas utilitárias.

Uma vez instalado o sistema operacional visitante, ainda temos que instalar um sistema de arquivos para ele. Existem várias imagens de sistemas de arquivos que podem ser baixadas do *UML Download Site.* Neste exemplo será usado o *Red-Hat* 7.2 *server file system image.*

Finalmente, ao executar o comando "*linux*" pode-se verificar que um novo sistema operacional será iniciado. Uma interface lógica *(tap0)* será criada, no sistema operacional principal e receberá um endereço IP. Este processo se encerra na tela de conexão (*login)*. Basta logar com o usuário "root" e senha "root" para ter acesso ao sistema operacional.

4.2. Configurando a rede

O próximo passo é fazer com que o sistema operacional visitante (*honeypot*) roteie todos os pacotes através do *gateway*. Isto significa que qualquer pessoa que precisar falar com nosso visitante deverá passar primeiro pelo *gateway*. Logo, a interface *tap0*, do sistema operacional principal, funciona como *gateway* do sistema operacional visitante.

A seguir, deve-se confirmar se o visitante efetivamente consegue rotear seus pacotes através da interface lógica do sistema operacional principal. Este teste pode ser feito utilizando-se o comando *ping*.

4.3. Controle de Dados

Próximo passo é controlar os dados. Neste exemplo o *IPTables* será usado para este propósito. Ele é um *firewall* de código aberto que vem junto com o Linux, capaz de limitar o número de conexões, registrar atividade e mapear endereços (NAT). O *IPTables* será configurado para trabalhar como um filtro, contando os pacotes que saem. Assim que alcançar o limite de conexões de saída, todas as outras tentativas serão bloqueadas, impedindo que a *Honeynet* comprometida ataque outros sistemas. Configurar e implementar estas regras pode ser bastante complexo. Contudo, o Projeto *Honeynet* desenvolveu um *script* chamado *rc.firewall* que faz este serviço.

O *IPTables* pode ser instalado para trabalhar na camada três ou na camada dois. A segunda opção é a mais interessante uma vez que, como foi explicado anteriormente, fica mais difícil que os invasores descubram a existência do *firewall.* Entretanto, para funcionar como uma ponte o *IPTables* necessita que o *kernel* utilizado possua suporte. Existe uma correção, que aplicada ao *kernel* faz as modificações necessárias para que o *IPTables* possa trabalhar como uma ponte.

4.4 Captura de Dados

A Captura de Dados tem o propósito de registrar toda a atividade do invasor sem que ele perceba. Para implementar esta fase, várias ferramentas estão disponíveis. Uma delas é o próprio *IPTables*, que já foi configurado com o *script rc.firewall* e por este motivo estará registrando todas as conexões de entrada e saída.

Uma outra alternativa é usar um sistema IDS, o *Snort*, para capturar e gravar toda atividade do invasor.

5. Conclusão

Honeynets são um tipo de *honeypot* projetados para capturar informação, especificamente, ferramentas, táticas e motivos da comunidade invasora. Esta informação é usada para proteger as organizações de várias ameaças.

O administrador da *Honeynet* tem a responsabilidade de garantir que nenhum outro sistema será atacado pela *Honeynet* comprometida. Sem uma administração apropriada estes riscos aumentam muito.

Existem duas gerações de *Honeynets* que se diferenciam principalmente pelo aprimoramento das tecnologias utilizadas na segunda geração. Além disso, a *Honeynet* virtual pode vir a se combinar com as anteriores para propiciar uma economia de recursos e aumentar a facilidade de manutenção.

As pesquisas demonstram que as organizações devem aplicar correções (*patch*) e desabilitar serviços que não estão sendo usados. Uma vez seguras, estas organizações estarão aptas a usar *Honeynets* para tirar vantagens e aprender mais sobre o inimigo. Esta tecnologia não deve substituir outros mecanismos de segurança, mas sim complementá-los.

Referências Bibliográficas

[1] Bill McCarty, **Botnets: Big and Bigger**, Azusa Pacific University, Computer Society, Security and Privacy, pp. 87-90.

[2] Honeynet Project, **Know Your Enemy: Honeynets**, Janeiro 2003, disponível em : http://www.honeynet.org/papers/virtual/

[3] Honeynet Project, **Know Your Enemy: Defining Virtual Honeynets**, Janeiro 2003, disponível em: http://www.honeynet.org/papers/virtual/

[4] L. Spitzner, **Honeypots – Tracking Hackres**, Indianápolis, IN: Addison-Wesley , 2002, pp. 49 – 72.

[5] John Levine, Richard LaBella, Henry Owen, Didier Contis, Brian Culver**, The Use of Honeynets to Detect Exploited Systems Across Large Enterprise Networks**, Proceedings of the 2003 IEEE, June 2003.

[6] Honeynet.BR Team, **Honeynet.BR: Desenvolvimento e Implantação de um Sistema para Avaliação de Atividades Hostis na Internet Brasileira**, INPE.

[7] Honeynet Project, **Honeynets Definitions, Requirements, and Standards**, disponível em: http://project.honeynet.org/alliance/requirements.html

[8] L. Spitzner, **The Honeynet Project: Trapping the Hackers**, Computer Society, Security and Privacy, pp..

[9] Honeynet Project, **Know Your Enemy: Gen II Honeynets**, Junho 2003, disponível em: http://www.honeynet.org/papers/virtual/

[10] Honeynet Project, **Know Your Enemy: Learning with User-Mode Linux**, Dezembro 2002, disponível em: http://www.honeynet.org/papers/virtual

[11] Honeynet Project, **Know Your Enemy: Learning with VMware**, Janeiro 2003, disponível em: http://www.honeynet.org/papers/virtual