ALEX JAECKEL

ESTUDO DE IMPACTO NA EFICIÊNCIA DOS ATAQUES ATRAVÉS DA UTILIZAÇÃO DE IPS TRANSPARENTE

Monografia apresentada ao Departamento de Computação da Universidade Federal de Lavras, como parte das exigências do curso de Pós-Graduação *Lato Sensu* em Administração em Redes Linux, para a obtenção do título de especialização.

Orientador:
Prof. Denilson Vedoveto Martins

LAVRAS
MINAS GERAIS – BRASIL
2008

ALEX JAECKEL

ESTUDO DE IMPACTO NA EFICIÊNCIA DOS ATAQUES ATRAVÉS DA UTILIZAÇÃO DE IPS TRANSPARENTE

Monografia apresentada ao Departamento de Computação da Universidade Federal de Lavras, como parte das exigências do curso de Pós-Graduação *Lato Sensu* em Administração em Redes Linux, para a obtenção do título de especialização.

APROVADA em ____ de _____ de 2008.

_____
Prof. Herlon Ayres Camargo

_____
Prof. Sandro Pereira Melo

_____
Prof. Denilson Vedoveto Martins
(Orientador)

LAVRAS
MINAS GERAIS – BRASIL
2008

Resumo

Apesar da Internet ter se tornado um meio de comunicação muito utilizado na atualidade, existe um grande problema: a falta de segurança. É crescente o número de ataques que ocorrem na Internet, e para combater esta situação, muitas técnicas e métodos foram criados para garantir maior segurança às redes. Com o objetivo de entender e analisar uma ferramenta capaz de oferecer maior segurança às redes, este trabalho apresenta o estudo de impacto na eficiência dos ataques através da utilização de IPS (Intrusion Prevention System) transparente, demonstrando o quanto essa solução pode interferir na eficiência de um ataque. Esta ferramenta pode ser conectada a um segmento de rede sem interferir na configuração dos computadores e dispositivos de rede.

Agradecimentos

A Deus, pela concessão da Sua Divina Luz, proteções e orientações.

Aos meus pais e familiares, por me apoiarem e me incentivaram.

Ao meu orientador, Denilson Vedoveto Martins, pelo apoio e contribuição dispensados ao longo deste trabalho.

Aos meus colegas de curso, pelo companheirismo, incentivo e apoio.

Aos demais professores do curso Pós-Graduação em Administração de Rede Linux – ARL/UFLA, que também compartilharam os seus conhecimentos e contribuíram com o meu crescimento.

E a todos aqueles que, de algum modo, me ajudaram.

Sumário

Lista de Siglas

ACL – Access Control List

BASE – Basic Analisys and Security Engine

DMZ – Desmilitarized Zone

IDS – Intrusion Detection System

IP – Internet Protocol

IPS – Intrusion Prevention System

NASL – Nessus Attack Scripting Language

NIDS – Network Intrusion Detection System

PERL – Practical Extraction and Report Language

PHP – Hypertext Preprocessor

SMB – Server Message Block

SNMP – Simple Network Management Protocol

TCP – Transmission Control Protocol

XML – Extensible Markup Language

Lista de Figuras

Lista de Tabelas

1 Introdução

Com o surgimento da Internet, a velocidade na troca de informações e a facilidade de acesso aos recursos disponibilizados aumentaram tanto, que, em pouco tempo, se tornou um meio de comunicação muito utilizado. Diante disso, houve um aumento significativo no número de pessoas e organizações que se conectaram à Internet para acessar e/ou disponibilizar recursos (Coelho, 2004 e RedHat, 2005).

Para um grande número de organizações, existe a necessidade destas manterem-se interligadas por meio de redes de computadores, que podem até mesmo abranger grandes extensões territoriais. Através destas redes, tais organizações acessam informações cruciais, parte importante dos seus recursos, para conduzirem seus negócios. A confidencialidade, a integridade e a disponibilidade destas informações poderão determinar o sucesso ou o fracasso de uma organização (Melo, 2004 e RedHat, 2005).

No entanto, no início, a Internet e seus protocolos foram desenvolvidos num ambiente de confiança, sem preocupação com a segurança. Até hoje, não existe algum padrão de segurança definitivo ou amplamente utilizado para a pilha de comunicações do TCP/IP, o que deixa portas abertas para usuários e processos maliciosos através da rede. Assim, devido a essa falta de segurança e a falta de configurações adequadas em muitos sistemas, ocorrem incidentes onde o roubo de informação ou uso indevido passaram a ser relatados.

A Figura 1.1 demonstra o total de incidentes reportados, por ano, ao Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.br, 2007).

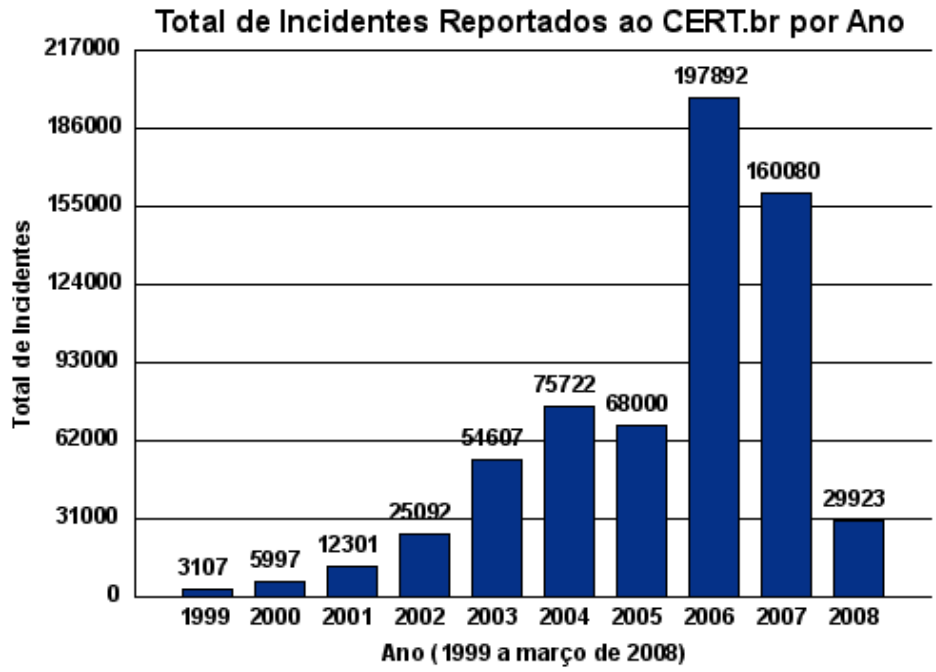

Figura 1.1: Total de Incidentes Reportados

Como contramedida para os problemas de segurança registrados, ao longo dos anos, muitos métodos foram desenvolvidos para aumentar a segurança da infra-estrutura de redes e das comunicações através da Internet, dentre os quais, podemos citar o uso de *firewalls*, detecção e prevenção de intrusos, e ferramentas de auditoria para verificar as vulnerabilidades das redes.

Além disto, o conhecimento das fases de um ataque, contribui para a correta configuração e implementação de metodologias e ferramentas que irão contribuir para a manutenção de um ambiente mais seguro. A fase inicial de um ataque é a coleta de dados sobre o *site* ou servidores do alvo. Para obter a listagem dos serviços TCP e UDP disponíveis, o sistema operacional, os aplicativos em uso e suas respectivas versões, o invasor faz uso de técnicas de *portscan* e *fingerprint*. Depois o invasor passa a testar diretamente os *hosts*

buscando suas vulnerabilidades e falhas de segurança. De posse destes dados, o invasor sabe quais os pontos fracos do sistema e pode decidir como irá comprometer ou invadir o sistema.

Outro tipo de ataque é o *botnet*. Um código malicioso, chamado de *bot*, transforma um computador infectado em um zumbi, isto é, uma máquina que pode ser controlada à distância pelo invasor, independente das ações do usuário. Uma rede de computadores infectados é chamada de *botnet*. Estas redes podem ser usadas para atacar *sites*, roubar dados, enviar *spam*, hospedar *sites* falsos e realizar ataques de negação de serviço (HSBC, 2008).

Além deste, temos os *worms*. O *worm* é um programa malicioso que se espalha automaticamente, controlando e obtendo informações de transmissão de dados da máquina infectada. Possui grande capacidade de se replicar, infectando muitas máquinas em poucos minutos (HSBC, 2008).

Atualmente, conforme testes realizados pelo NSS Labs, existem diversas aplicações comerciais de IPS certificadas: Broadweb NetKeeper NK-3256T v3.6, Cisco IPS-4240, Cisco IPS-4255, ISS Proventia NIPS GX4004, ISS Proventia NIPS GX5108, Juniper IDP 600F, McAfee IntruShield 4010 e SecureWorks iSensor 850 (NSS Labs, 2008).

Foram encontrados dois softwares, *open source*, para implementar o IPS: o *Snort* (www.snort.org) e o *Snort_inline* (snort-inline.sourceforge.net). Por apresentar uma tecnologia de detecção e prevenção de intrusão empregada largamente mundo afora e devido a maior quantidade de documentação existente, e conseqüente contribuição para a sua configuração, o IPS foi implementado com o *Snort* (Snort.org, 2008).

Com o objetivo de entender melhor os problemas de segurança e contribuir com a literatura atual que é muito escassa sobre a eficiência das

ferramentas de segurança, este trabalho apresenta um estudo de impacto na eficiência dos ataques através da utilização de um sistema de prevenção de intrusos, um IPS (*Intrusion Prevention System*) transparente, utilizando o *Snort* e o BASE sobre o sistema operacional Fedora, que são softwares gratuitos, *open source* e muito utilizados pela comunidade Linux. Este sistema pode ser conectado num segmento da rede sem interferir na configuração dos computadores e equipamentos de rede.

Visando impedir e/ou atrasar as fases anteriores à invasão, este trabalho demonstra os bloqueios realizados pelo IPS e os atrasos provocados por ele frente as tentativas de levantamento de informações realizadas pelas ferramentas de segurança: *Nmap*, *Nessus* e *Nikto*.

Para realizar o estudo, foi instalado um servidor HTTP, com uma versão antiga de sistema operacional, o RedHat 7.3, com o Apache 1.3.23, com configuração padrão, sem atualização, para garantir a existência de vulnerabilidades conhecidas. Outro computador, que estava interligado ao servidor através do IPS transparente, realizou as tentativas de levantamento de informações através do uso das ferramentas de segurança.

Isto posto, a metodologia utilizada para medir a eficiência dos ataques é quantitativa e comparativa, ou seja, medimos a quantidade de resultados obtidos pelas ferramentas de segurança com a ausência e com a presença do IPS na rede, e, depois comparamos com os resultados obtidos.

O trabalho aqui apresentado foi dividido conforme denotado a seguir. O Capítulo 2 apresenta as definições das tecnologias de segurança utilizadas pelo trabalho. O Capítulo 3 descreve a configuração do IPS e as ferramentas instaladas para possibilitar a monitoração dos ataques e a administração das regras adotadas no ambiente de testes onde a solução foi implementada. O Capítulo 4 descreve os testes realizados e resultados obtidos. Já o Capítulo 5,

relata as conclusões obtidas e as idéias para trabalhos futuros. Além dos capítulos, o Apêndice A contêm o tutorial de instalação e configuração do IPS. Já o Apêndice B contêm o resultado dos *scans* realizados pelo *Nmap e Nessus*. Por fim, o Apêndice C contêm o arquivo de configuração do *Snort*: o *snort.conf*.

2 Tecnologias de Segurança

Para oferecer maior segurança a um ambiente de rede, não basta apenas manter os softwares atualizados e manter apenas os serviços necessários ativos. Devido à grande complexidade dos softwares, muitas vulnerabilidades podem ser encontradas. Estas poderão ser exploradas por usuários maliciosos e causar danos à rede. Para evitar estas situações, muitas ferramentas de segurança foram criadas.

Neste capítulo serão apresentadas algumas ferramentas empregadas no ambiente de rede, dentre elas: *firewall*, IDS, IPS, *scanner* de rede e *scanner* de vulnerabilidades.

2.1 Firewall

Muitos consideram que a ferramenta básica para garantir a segurança de uma rede é o *firewall*. Os *firewalls* são posicionados no perímetro de uma rede para cumprir uma política de segurança que irá permitir ou negar a passagem de certos tráfegos da rede (SNAC, 2007).

Os *firewalls* podem ser classificados em três tipos:

- o Filtro de Pacotes *Stateless*;
- o Filtro de Pacotes *Stateful*;
- o *Proxy* de Aplicação.

2.1.1 Filtro de Pacotes *Stateless*

O filtro de pacotes *stateless* faz o controle do tráfego baseado apenas no cabeçalho dos pacotes. O cabeçalho possui em seus campos os endereços IP de origem e destino, as portas de origem e destino e o protocolo usado.

A filtragem de pacotes *stateless* é normalmente aplicada nos roteadores na forma de listas de controle de acesso (ACLs), sendo rápida e eficiente. Para criar as regras é preciso defini-las para cada protocolo, faixas de endereços, e direção do fluxo.

Os filtros de pacotes *stateless* são extremamente rápidos. Uma vez que examinam apenas a porta de destino e/ou o endereço IP de origem/destino, eles possuem muito pouco trabalho para fazer. Os filtros de pacotes simples são escolhas excelentes para segmentos de rede de alto tráfego.

A rápida ativação é um ponto importante do filtro de pacotes *stateless*. Basta saber as portas necessárias e/ou de onde o tráfego pode vir e para onde pode ir que pode-se configurá-lo em poucos minutos.

Porque os filtros de pacotes *stateless* são simples e realizam inspeções simples, eles são menos seguros em relação aos outros tipos de *firewall*, pois permitem a passagem de qualquer pacote chegando de uma subrede permitida para uma porta permitida, sem fazer uma inspeção rigorosa no pacote.

Os filtros de pacotes não acompanham de onde vem os pacotes de entrada, ou garantem que os pacotes de retorno sejam direcionados ao mesmo local. E também, que não se pode mover de portas estáticas baixas para dinâmicas altas. Muitas aplicações usam isto após o *handshake* inicial e as duas máquinas concordarem em como comunicar. A aplicação irá requisitar a mudança para uma porta alta para liberar as portas estáticas baixas para outros *handshakes* iniciais. Com um filtro de pacotes *stateless*, isto requer abrir a

maioria, senão todas, as portas dinâmicas, o que torna o filtro de pacotes *stateless* sem utilidade. Para contornar este problema e permitir que o filtro de pacotes stateless suporte mudanças de portas é necessária a instalação de módulos adicionais. No caso do iptables é o módulo ip_conntrack. Este módulo permite o acompanhamento do estado de cada conexão (`Andreasson, 2008`).

O *ftp*, um protocolo padrão, pode se comportar de modo estranho. Já que a comunicação ocorre na porta 21, mas a transferência de dados é modificada para a porta 20, muitos filtros de pacotes falham em passar corretamente pacotes *ftp*, portanto a transferência de arquivos é interrompida.

Arquiteturas de redes complexas podem tornar difícil a tarefa de fazer as regras de filtro de pacotes *stateless*, especialmente com IP *masquerading* ou subredes locais e redes desmilitarizadas (DMZ).

2.1.2 Filtro de Pacotes *Stateful*

Assim como o filtro de pacotes *stateless*, o filtro de pacotes *stateful* atua nas camadas de rede e transporte examinando o cabeçalho dos pacotes. Diferente do filtro de pacotes *stateless*, se o fluxo é permitido, um filtro de pacotes *stateful* realiza o acompanhamento do estado da conexão e o registra numa tabela de estados. Estes registros contêm os endereços IP de origem e destino, as interfaces de rede do *firewall*, as portas de origem e destino, o protocolo, os números de ACK e SEQ, etc.

Este conceito permite que as regras sejam mais concisas do que no filtro de pacotes *stateless* uma vez que regras referentes aos outros pacotes que fazem parte da conexão fazem parte de uma mesma regra e não de várias inseridas

manualmente.

O desempenho do sistema é maior pois os pacotes que entram no *firewall* são primeiro comparados à tabela de estados, e não a ACL. Se um pacote for parte de uma conexão estabelecida então será permitido. Se um pacote não estiver associado a uma conexão estabelecida, então será feito uma verificação completa nas regras de ACL.

Os filtros de pacotes *stateful* possuem limites no conhecimento do protocolo e falta a habilidade de filtrar tráfego criptografado. Este último problema é uma característica comum a todos os tipos de *firewall*.

2.1.3 *Proxy* de Aplicação

Além das características do filtro de pacotes *stateless* e do filtro de pacotes *stateful*, o *proxy* de aplicação contêm processos cliente e servidor para cada protocolo suportado. Eles irão estabelecer duas seções completamente separadas: uma do cliente para o *firewall*, e a outra do *firewall* para o servidor . Desta forma, eles estabelecem um *proxy* que inspeciona e reescreve todos pacotes de um dado protocolo, evitando o contato direto entre o cliente e o servidor.

Em comparação com o filtro de pacotes *stateless* e o filtro de pacotes *stateful*, o desempenho do *proxy* de aplicação é mais lento devido à complexidade de filtrar todos os dados da aplicação. Também são mais vulneráveis a ataques de *overflow* por serem mais complexos e realizarem mais verificações (SNAC, 2007).

2.2 IDS

A detecção de invasão é um conjunto de técnicas e métodos utilizados para detectar atividades suspeitas na rede e nos *hosts* (Rehman, 2003).

Os sistemas de detecção de intrusão, IDS (*Intrusion Detection System*), podem ser divididos em duas categorias: baseados em assinaturas e baseados em anomalias.

Os intrusos têm assinaturas, assim como os vírus de computadores, que podem ser detectados com o emprego de software. Pode-se encontrar pacotes de dados que contêm qualquer assinatura conhecida relacionada a invasão ou anomalias relativas ao conjunto de protocolos TCP/IP. Baseados num conjunto de assinaturas e regras, o sistema de detecção é capaz de encontrar e registrar atividades suspeitas, além de gerar alertas.

Ao achar assinaturas e as utilizar em regras, deve-se tomar cuidado especial, pois quanto mais regras forem utilizadas, mais poder de processamento será necessário para processar os dados capturados em tempo real. É importante implementar o máximo de assinaturas que puder usando o mínimo de regras possíveis.

A detecção de invasão baseada em anomalias normalmente depende de anomalias presentes nas partes do cabeçalho do protocolo, ou em um comportamento atípico da rede.

2.2.1 Assinaturas

A assinatura é um padrão que se procura dentro de um pacote de dados.

Uma assinatura é usada para detectar um ou múltiplos tipos de ataques. Por exemplo, a presença de "*scripts/iisadmin*" num pacote direcionado ao servidor *Web* pode ser indicação de atividade de um invasor.

As assinaturas podem estar presentes em diferentes partes do pacote de dados dependendo da natureza do ataque. Por exemplo, pode-se encontrar assinaturas no cabeçalho IP, no cabeçalho da camada de transporte (cabeçalho TCP ou UDP) e/ou cabeçalho da camada de aplicação ou na área de dados.

O IDS precisa ser atualizado constantemente para adicionar novas assinaturas de novos ataques que forem descobertos.

2.2.2 Alertas

Os alertas são qualquer tipo de comunicado ao usuário referente a atividade de intrusão. Quando um IDS detecta uma intrusão, deve avisar ao administrador de segurança através de um alerta. Os alertas podem ser na forma de *pop-up windows*, realizando *log* na console, enviando *email*, etc. Os alertas também são armazenados em arquivos de *log* ou banco de dados onde poderão ser visualizados mais tarde por profissionais de segurança.

2.2.3 Logs

As mensagens de *log* normalmente são armazenadas num arquivo no formato texto ou binário. Registrar *logs* no formato binário é mais rápido pois economiza tempo de formatação. Os arquivos binários podem ser visualizados

através do Snort, do programa *tcpdump*[1] ou do *Barnyard*[2]. Em implementações de alta velocidade é necessário registrar no modo binário.

2.2.4 Falso Positivo

O falso positivo é um alerta gerado devido à identificação de uma situação que não é uma atividade de intrusão.

Para evitar os falsos positivos, deve-se modificar e ajustar diferentes regras padrão. Em alguns casos, pode ser necessário desabilitar algumas das regras.

2.2.5 Falso Negativo

O falso negativo ocorre numa situação que é uma atividade de intrusão e por não haver assinatura que a identifique, nenhum alerta é gerado. Para evitar estas situações, deve-se manter as regras e assinaturas atualizadas ou gerar novas assinaturas e regras.

1 http://www.tcpdump.org
2 http://sourceforge.net/projects/barnyard

2.2.6 Snort

O *Snort* é uma ferramenta de código aberto, desenvolvido em C, bastante popular por sua flexibilidade nas configurações de regras e constante atualização frente às novas ferramentas de invasão . Possui um grande número de assinaturas, é leve, pequeno, e permite a captura, a partir de um micro, dos pacotes que trafegam na rede e a verificação de anomalias que se encontram nela.

O seu código e as regras de detecção estão em constante desenvolvimento e atualização.

Basicamente é um IDS baseado em regras, no entanto, tem plugins que detectam anomalias nos cabeçalhos dos protocolos.

O *Snort* utiliza regras que são armazenadas em formato texto que podem ser modificadas por um editor de texto. As regras são agrupadas em categorias. As regras que pertencem a cada categoria são armazenadas em arquivos separados. Estes arquivos são então incluídos no *snort.conf*, o arquivo de configuração do *Snort*. O *Snort* lê as regras na inicialização e monta uma estrutura interna de dados ou *chain* para aplicar estas regras aos dados capturados.

O *Snort* possui um farto conjunto de regras pré-definidas para detectar atividades de intrusão e oferece liberdade para a criação de novas regras. Pode-se também remover algumas das regras existentes para evitar alertas falsos (conhecidos também como Falsos Positivos).

O *Snort* é composto de múltiplos componentes. Estes componentes trabalham em conjunto para detectar ataques específicos e para gerar saída do sistema de detecção em formato definido. Um IDS baseado no *Snort* é composto dos seguintes componentes:

- o decodificador de pacotes;
- os preprocessadores;
- o mecanismo de detecção;
- o sistema de *log* e alerta;
- os módulos de saída.

A Figura 2.1 demonstra como os componentes estão dispostos. Qualquer pacote com dados provenientes da Internet entra no decodificador de pacotes. No seu caminho para os módulos de saída, ou será descartado, ou será registrado ou irá gerar um alerta (Rehman, 2003).

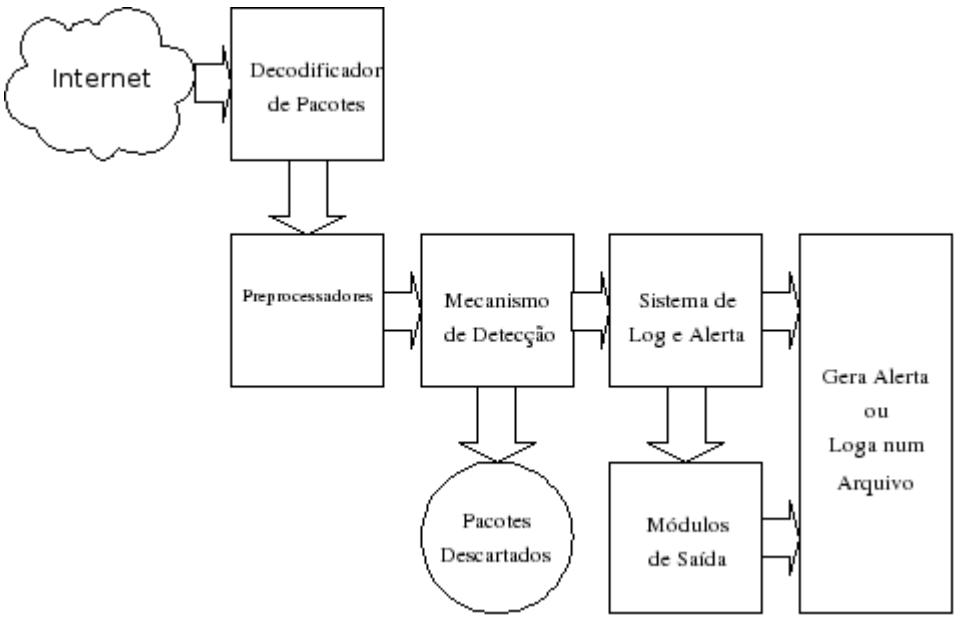

Figura 2.1: Componentes do *Snort*

O decodificador de pacotes obtém pacotes de diferentes tipos de interfaces de rede e prepara os pacotes para serem preprocessados ou para serem enviados para o dispositivo de detecção. As interfaces podem ser Ethernet, SLIP,

PPP, etc.

Os preprocessadores são componentes ou *plugins* que podem ser usados com o *Snort* para preparar ou modificar pacotes de dados antes que o dispositivo de detecção faça alguma operação para descobrir se o pacote está sendo usado por um intruso. Alguns preprocessadores também realizam detecção para identificar anomalias nos cabeçalhos dos pacotes e gerar alertas. Os preprocessadores são muito importantes para preparar os pacotes de dados para que possam ser analisados no dispositivo de detecção. Um intruso pode usar diferentes técnicas para enganar um IDS. Por exemplo, se houver uma regra que busca pela assinatura "*scripts/iisadmin*" nos pacotes HTTP. Se estiver buscando um *string* exato, poderá ser enganado por um intruso que faz pequenas mudanças no *string*. Por exemplo:

"*scripts/./iisadmin*";

"*scripts/examples/../issadmin*".

O intruso pode também inserir na *string* algum caractere hexadecimal URI (*Uniform Resource Identifier*) ou caracteres Unicode que são perfeitamente legais. Os servidores web normalmente entendem todos estes *strings* e serão capazes de preprocessá-los para o *string* pretendido "*scripts/iisadmin*". Como o IDS está procurando por um casamento exato não irá detectar o ataque. Um preprocessador pode ajeitar o *string* de modo que seja detectável pelo IDS.

Os preprocessadores também são usados para desfragmentar pacotes. Quando um grande pacote é transferido para um *host*, o pacote normalmente é fragmentado. O pacote deverá ser remontado na sua forma original para que o IDS possa aplicar as regras e tentar encontrar assinaturas. Por exemplo, metade de uma assinatura pode estar num pacote o a outra metade em outro segmento. Os intrusos utilizam a fragmentação para enganar sistemas de detecção de intrusos.

O mecanismo de detecção é a parte mais importante do *Snort*. Sua responsabilidade é detectar se existe qualquer atividade de invasão num pacote fazendo uso das regras do *Snort*. Se um pacote casar qualquer regra, uma ação apropriada, registrar o pacote ou gerar alerta, será tomada, senão o pacote será descartado.

O mecanismo de detecção é uma parte crítica do *Snort*. Dependendo do poder de processamento e de quantas regras foram definidas, poderá levar tempos diferentes para responder a diferentes pacotes. Se o tráfego na rede for muito alto quando o *Snort* estiver executando no modo NIDS, poderá descartar alguns pacotes e não poderá efetuar respostas em tempo real. A carga no dispositivo de detecção depende dos seguintes fatores:

- o número de regras;
- poder de processamento da máquina que executa o *Snort*;
- a velocidade do barramento interno usado pela máquina do *Snort*;
- a carga na rede.

O mecanismo de detecção pode desmembrar o pacote e aplicar regras em diferentes partes. Estas partes poderão ser:

- o cabeçalho IP do pacote;
- o cabeçalho da camada de transporte. Incluindo ICMP, TCP, UDP e outros cabeçalhos de camadas de transporte;
- o cabeçalho da camada de aplicação. Os cabeçalhos da camada de aplicação incluem, mas não estão limitados a eles, cabeçalho DNS, cabeçalho FTP, cabeçalho SNMP, e cabeçalho SMTP;
- área de dados. Pode-se criar uma regra que será usada pelo dispositivo de detecção para encontrar um *string* dentro dos dados contidos no pacote.

No sistema de *log* e de alerta, dependendo do que o dispositivo de

detecção encontrou dentro do pacote, o pacote deverá ser usado para registrar a atividade ou para gerar um alerta.

Os módulos de saída ou plugins controlam o tipo de saída que será gerado pelo sistema de *log* e de alerta. Dependendo da configuração, poderemos ter como saída, por exemplo:

- registrar no arquivo */var/log/snort/alerts* ou em outro arquivo;
- enviar traps SNMP;
- enviar mensagens para o *syslog*;
- registrar num banco de dados, MySQL ou Oracle, por exemplo;
- gerar saída XML;
- modificar a configuração de roteadores e *firewall*;
- enviar mensagens SMB, *Server Message Block*, para máquinas com Windows.

2.3 IPS

Quando se faz referência a técnicas de segurança para redes, o termo prevenção de intrusão em redes, normalmente, é aplicado a dispositivos *inline* (*bridge* ou *firewall*) que tem a capacidade de modificar ou descartar pacotes de ataque que tentem atravessar as interfaces do dispositivo (`Beale, 2007`).

Em relação a modificação de pacotes, o objetivo é anular os ataques realizados contra os dispositivos internos conectados a um sistema de prevenção de intrusos (IPS) ou através dele.

O termo resposta ativa, se aplica a qualquer função que altera ou bloqueia tráfego de rede como resultado de eventos de detecção de intrusão. Tais

funções não necessariamente são implementados num dispositivo *inline*. No entanto, tais capacidades não são fortes o suficiente para serem classificadas como IPS porque permitem que certos ataques possam causar muitos estragos. A grande diferença é que qualquer dispositivo que não seja *inline* não é capaz de impedir tal tráfego de chegar ao destino almejado. Poderá reagir de diversas maneiras, mas qualquer dispositivo que não seja *inline* irá criar uma condição de corrida entre o tráfego malicioso e a resposta (que poderá chegar primeiro ao alvo).

Esta corrida nem sempre será vencida pelo mecanismo de resposta. Responder ativamente a um pacote depois que tenha entrado na rede não é o suficiente. Um meio de aliviar o efeito de um ataque é prevenir que pacotes maliciosos entrem na rede, e um dispositivo *inline* pode fazer isto. Pode-se considerar o IPS como o subgrupo mais poderoso das funções de resposta ativa.

2.3.1 Snort_inline

A proposta deste trabalho está baseada no *Snort* no modo *inline* que funciona como um IPS. Este por sua vez oferece flexibilidade nas configurações de regras, é leve e pequeno e está em constante atualização. Nesta configuração, o *Snort* recebe os pacotes do *iptables*, ao invés do *libpcap*.

A principal diferença que torna um sistema de resposta ativa num IPS é ter a capacidade de descartar ou modificar pacotes em tempo real assim que eles tentam entrar na rede ou sair dela. Isto é, os pacotes devem passar através do IPS, e portanto deve ser um dispositivo *inline*.

O IPS deve ser um *hop* na rota pela qual os pacotes passam na medida

em que entram na rede ou saem dela, ou agir como uma *bridge* entre dois segmentos de rede *ethernet*. Se o IPS agir como uma *bridge* não será reconhecido como um *hop* adicional porque o valor do *Time to Live* não será decrementado quando os pacotes passarem através das suas interfaces.

Um dispositivo *inline* pode tanto descartar ou rejeitar pacotes individuais baseado nos dados da camada de aplicação, quanto modificar os dados de aplicação dentro do dispositivo, antes de enviá-los adiante. Em muitos casos, isto permite ao IPS anular ataques e dificultar para o lado cliente a detecção da modificação da aplicação antes que o ataque possa causar algum dano (Beale, 2007).

2.4 Scanners

Um *scanner* de redes (por exemplo: *Nmap*) pode fornecer informação sobre quais portas estão abertas ou fechadas em um sistema. Não consegue determinar de forma confiável qual serviço está sendo executado numa determinada porta. Por exemplo, um servidor *Web* pode ser configurado para executar na porta 22, que normalmente está reservada para o *SSH*. O *scanner* de redes irá informar que a porta 22 está aberta, e que o serviço *SSH* está disponível no sistema alvo (May, 2003).

Para ajudar a identificação dos serviços, o Nmap, com a opção "-sV", informa os serviços e versão que estão executando em cada porta.

Um *scanner* de vulnerabilidades, tal como o *Nessus*, por outro lado, obtêm informação fornecida por um *scan* de porta e realiza várias verificações para determinar qual serviço está sendo executado e suas vulnerabilidades.

Considerando-se o exemplo anterior, um scanner de vulnerabilidades poderia informar que um serviço *Web* está sendo executado na porta 22, que o servidor é o *Apache 1.3.26* e que contêm quatro vulnerabilidades. Estas informações serão fundamentais para a configuração de um sistema mais seguro.

2.4.1 Nmap

O *Nmap*, *Network Mapper*, é uma ferramenta de código aberto, utilizada para executar auditorias de segurança em redes de computadores, servidores de fronteira e mesmo computadores isolados. O *Nmap* foi projetado para fazer rapidamente a varredura de diversos tipos de redes de computadores. Seus testes e varreduras são feitos utilizando-se pacotes IP, TCP, e UDP, enviados para diferentes direções. Por meio desta técnica, é possível avaliar, por exemplo, quantos computadores fazem parte de uma rede, que serviços estão sendo executados em um computador, que sistemas operacionais são utilizados em cada *host*, se algum tipo de *firewall* está sendo utilizado, dentre várias outras características (`Muresan, 2007`).

2.4.2 Nessus

O *Nessus* é um programa de auditoria de segurança, capaz de realizar seus testes sob e contra diversos sistemas operacionais. Em seu modo operacional padrão, o *Nessus* inicia sua auditoria varrendo as portas lógicas do sistema utilizando *Nmap* ou seu próprio software de varredura de portas. Esse

processo é seguido da utilização de diversos *exploits* que atacam serviços presentes no sistema. Os testes de vulnerabilidade estão disponíveis sob o formato de uma grande lista de *plugins* escritos em NASL, *Nessus Attack Scripting Language*. Opcionalmente, os resultados da auditoria podem ser exportados em relatórios de vários formatos, como texto, XML, HTML, e LaTeX (`Muresan, 2007`).

2.4.3 Nikto

O *Nikto* é um *scanner* de servidores *Web* e *CGI*, que foi desenvolvido em *PERL*. É uma ferramenta utilizada para examinar a segurança do servidor *Web* e *CGI*. Como faz muitas requisições ao servidor remoto, em alguns casos poderá causar a sua queda(*crash*) (`Cirt.net, 2007`).

Realiza um *portscan* e verifica a existência de um servidor *Web* ativo em qualquer porta aberta. Pode realizar verificações em HTTP e HTTPS. Além disto, verifica diversos itens, tais como:

- erros de configuração;
- arquivos e *scripts* padrões;
- arquivos e *scripts* inseguros;
- software desatualizados.

O *Nikto* não tem apenas a capacidade de verificar vulnerabilidades de servidor *Web* e *CGI*, mas também de fazê-lo de maneira evasiva, para enganar sistemas de detecção de intrusão (`RedHat, 2005`).

As verificações e o código do *Nikto* podem ser automaticamente atualizadas para garantir que sejam verificadas as vulnerabilidades mais recentes.

3 Configuração e Administração do IPS

Para a fase de implementação e testes deste trabalho, foi empregado o *Snort* no modo *inline*. Nesta configuração, as tentativas de invasão que forem reconhecidas são imediatamente bloqueadas.

Inicialmente, foi testado um IPS com *Snort* no modo *inline* e *FreeBSD 6.1.* Nesta configuração, com o FreeBSD no modo bridge, o Snort no modo inline não funciona com o IPFW. Para que o *Snort* funcione no modo *inline* é necessário realizar NAT (o *Snort* exige a ativação da opção de kernel *IPDIVERT*) (Rogness, 2008). Esta opção foi então descartada, pois seria necessário modificar a configuração dos *hosts,* o que descaracterizaria o objetivo inicial deste trabalho que é funcionar de forma transparente.

A instalação do *Fedora Core 5* foi descartada por apresentar problemas de compatibilidade com o hardware do computador disponível.

A tentativa seguinte foi com o *Snort 2.6* e o *Fedora Core 6* que vieram a se tornar a base para o presente trabalho. Posteriormente, o Snort foi atualizado para a versão 2.8.0.1.

3.1 Configuração

A configuração do *Snort* no modo *inline* envolve três passos principais: configurar o Linux para funcionar em modo bridge entre dois segmentos *ethernet*, definir as políticas do *iptables* para enviar os pacotes para o *QUEUE* e definir as regras de bloqueio no *Snort* (Beale et al., 2007).

Neste trabalho o sistema *Snort_inline* tem duas interfaces *ethernet*, eth0 e eth1. O *script* da Tabela 3.1, que mostra a sequência de comandos para configurar o sistema, realiza os seguintes passos: configurar a *bridge*, br0, desabilitar o *ip_forwarding*, e inicializar o *queueing* de pacotes do *iptables* na *chain FORWARD* e o *Snort* no modo *inline*.

Um item importante a ser notado é que o *ip_forwarding* deve ser desabilitado. Isto acontece pois o *Snort_inline* é responsável por construir pacotes (via *libnet*) na interface de saída ao invés da pilha IP nativa do sistema. Isto permite ao *Snort_inline* passar adiante apenas aqueles pacotes que não casam com uma regra no dispositivo de detecção do *Snort*, ou aqueles pacotes que casaram e foram alterados. Isto também significa que se o processo do *Snort_inline* for finalizado por qualquer motivo, toda conectividade da rede será penalizada para os segmentos de rede que estão conectados pela *bridge* do sistema no qual o *Snort_inline* é empregado.

As regras do *Snort* tem uma ação padrão de regra que é *alert* (alerta). Para o *Snort_inline* a ação padrão de regra foi modificada para *drop* para que todos pacotes que casarem com alguma regra sejam descartados. Para realizar esta mudança foi utilizado o *script* "*Muda-para-DROP.sh*" que se encontra no Apêndice A.

Tabela 3.1: Script de Inicialização do *Snort_inline*

```
#!/bin/sh

# comandos
BRIDGE=/usr/sbin/brctl
IFCONFIG=/sbin/ifconfig
IPTABLES=/usr/sbin/iptables
ECHO=/bin/echo
SNORT=/usr/local/bin/snort

# configura e ativa as interfaces sem IP
$IFCONFIG  eth0  0.0.0.0  up
$IFCONFIG  eth1  0.0.0.0  up

# cria bridge
$BRIDGE  addbr  br0
$BRIDGE  addif  br0  eth0
$BRIDGE  addif  br0  eth1

# ativa bridge
$IFCONFIG  br0  0.0.0.0  up

#carrega o modulo ip_queue
modprobe ip_queue

# limpa as regras do iptables e envia todos pacotes do chain FORWARD
# para o QUEUE para que o Snort_inline possa examiná-los.
$IPTABLES  -F
$IPTABLES  -A  FORWARD  -p all  -j QUEUE

# desabilita ip forwarding
$ECHO  0 >  /proc/sys/net/ipv4/ip_forward

#ativa o Snort_inline
$SNORT  -Q  -D  -c /etc/IPS/snort.conf
```

O *Snort_inline* acrescenta três ações que poderão ser especificadas nas regras do *Snort*: *drop*(descarta), *reject*(rejeita) e *sdrop*(descarta sem registrar). A ação de regra *drop* instrui o *Snort_inline* a descartar o pacote via *Netfilter* e registrar a ocorrência. A ação de regra *reject* gera um reset TCP para sessões TCP e uma mensagem ICMP *port-unreachable* será gerada para pacotes UDP. A ação de regra *sdrop* faz o mesmo que o *drop* só que neste caso não registra.

O *Snort_inline* implementa também a regra *replace* que realiza a substituição de um conteúdo que case com um conteúdo especificado pelo administrador, conforme exemplo da Figura 3.1.

```
alert tcp any any <> any 80 (msg: "tcp replace"; content: "GET"; replace: "BET";)
```

Figura 3.1: *Replace*

A opção *replace* só pode substituir o conteúdo de um pacote por novos dados que tenham o mesmo tamanho. No caso do UDP, o campo de comprimento no cabeçalho especifica o comprimento em bytes do cabeçalho e dos dados contidos nele.

3.2 Monitoração dos Ataques e Administração das Regras

Para facilitar a monitoração dos ataques e a administração das regras foram instaladas duas ferramentas: o *BASE*[3] e o *Pmgraph*[4].

A Figura 3.2 demonstra o *BASE*, "*Basic Analysis and Security Engine*", desenvolvido em PHP, que é uma ferramenta para listar e analisar os alertas gerados pelo *Snort_inline* no banco de dados MySQL. É uma ferramenta poderosa que permite muitas opções de pesquisa e a habilidade de agrupar alertas baseados nos endereços IP de origem ou destino, nos tipos de ataques, por protocolo, os ataques mais recentes e outros.

3 http://base.secureideas.net/
4 http://cerberus.sourcefire.com/~jbrvenik/pmgraph-0.2.tar.gz

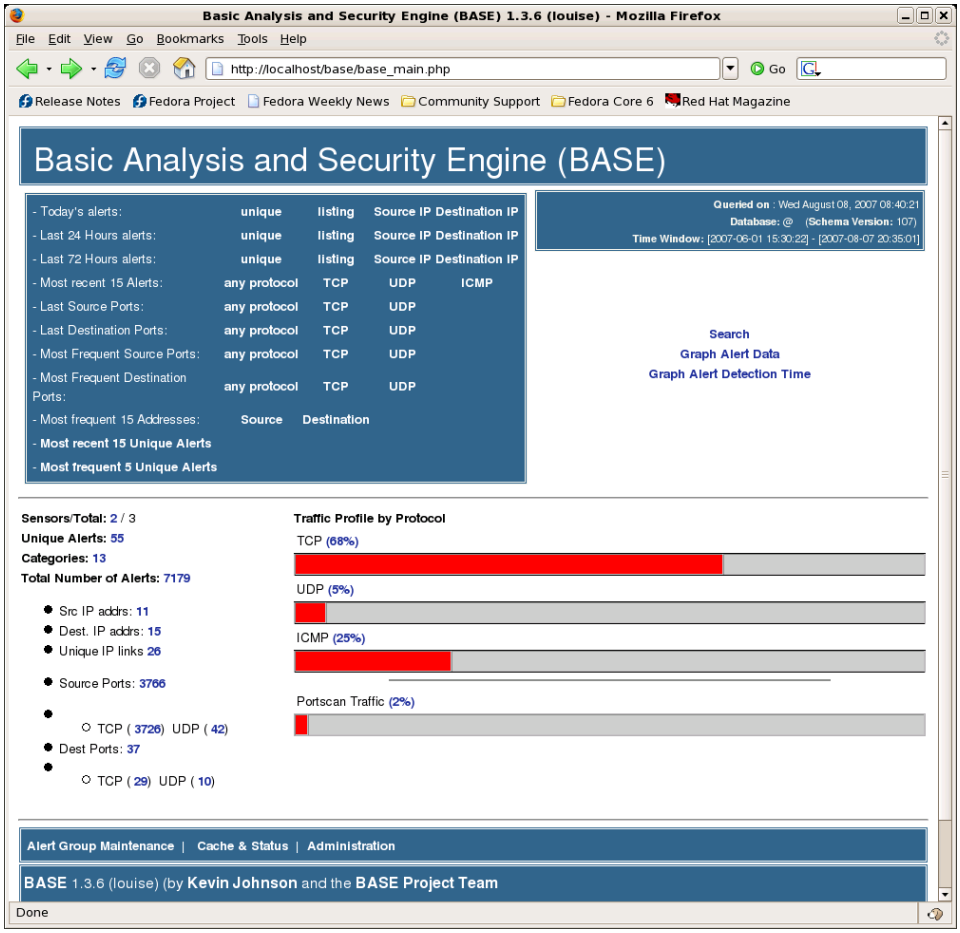

Figura 3.2: *BASE*

O *Pmgraph,* é um *script* escrito em Perl que gera páginas HTML com tabelas que contêm as estatísticas do *Snort* (Pacotes Bloqueados, Alertas por segundo, Mbit por segundo, Pacotes por segundo (milhares de pacotes por segundo), SYN + SYN/ACK pacotes por segundo, Eventos de sessão por segundo, Sessões abertas, Eventos de *Stream* por segundo, eventos de *frag* por segundo, Média de *bytes* por pacote, Estatísticas da CPU1(%)) geradas pelo preprocessador *perfmon*.

O preprocessador *perfmon* foi configurado conforme apresentado na Figura 3.3. Nesta configuração, o *time* representa o número de segundos entre intervalos, o *file* é o arquivo onde os dados são armazenados e *pktcnt* é o número máximo de registros armazenados no arquivo.

```
time 60  file /var/log/snort/snort.stats  pktcnt 500
```

Figura 3.3: Configuração do preprocessador *perfmon*

Para gerar os gráficos das estatísticas, deve-se executar como root o comando apresentado na Figura 3.4. O *pmgraph.pl* faz a leitura dos dados das estatísticas armazenadas no arquivo *snort.stats* e gera os gráficos das estatísticas em *usage*.

```
# /usr/local/src/pmgraph-0.2/pmgraph.pl  /var/www/html/usage  /var/log/snort/snort.stats
```

Figura 3.4: Gerando estatísticas com *pmgraph*

Pode-se acrescentar o comando no crontab para atualizar as estatísticas automaticamente, conforme demonstrado na Figura 3.5. As estatística serão atualizadas a cada trinta (30) minutos.

```
*/30 * * * * /usr/local/src/pmgraph-0.2/pmgraph.pl /var/www/html/usage
/var/log/snort/snort.stats
```

Figura 3.5: Gerar estatísticas

Para visualizar as tabelas das estatísticas num *browser*, deve-se digitar a *url* conforme a Figura 3.6.

http://endereçoIPservidor/usage

Figura 3.6: Visualizar no *browser*

A Figura 3.7, mostra uma parte dos gráficos gerados pelo *pmgraph*: pacotes bloqueados, o número de alertas por segundo, o número de Mbit por segundo e Kpacotes por segundo (quantos milhares de pacotes por segundo).

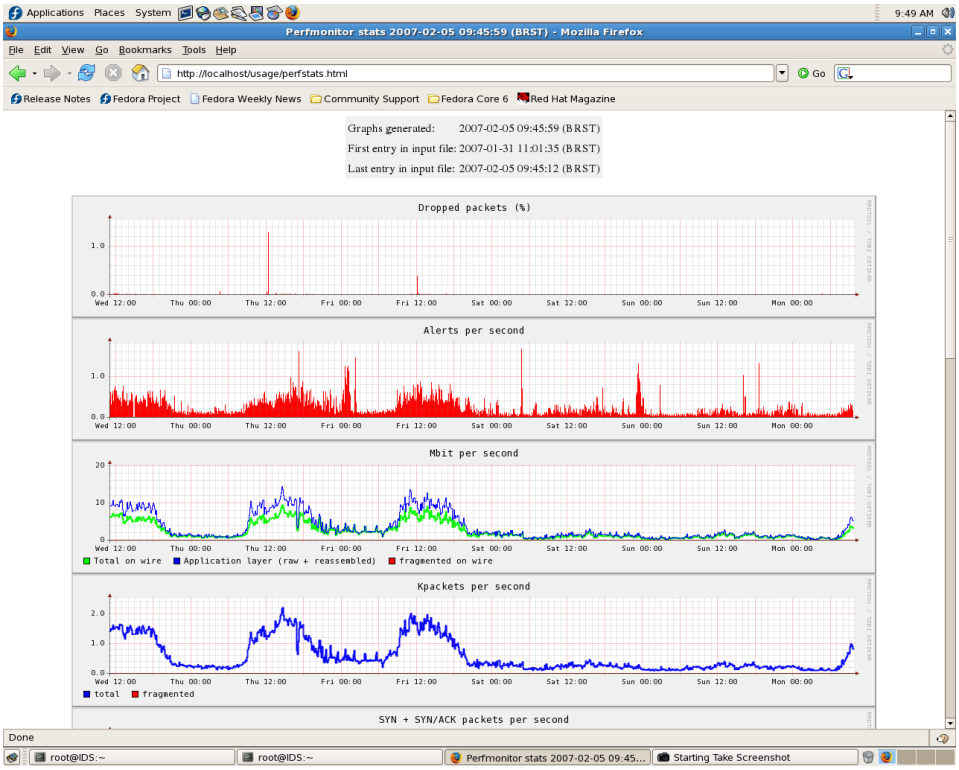

Figura 3.7: *pmgraph*

4 Validações e Resultados

A Figura 4.1 apresenta o ambiente de teste implementado para realizar o estudo de impacto na eficiência dos ataques através da utilização do IPS transparente. Um servidor Linux, com sistema operacional RedHat 7.3, configurado com os serviços padrão e o HTTP ativados, foi conectado a uma porta do *gateway*, que pode estar com ou sem o IPS ativo. A outra porta, estava conectada a máquina responsável pelos scans na máquina servidora.

O servidor Linux contém o apache/1.3.23, ssh/1.99, mod_ssl/2.8.7, openssl/0.9.6b, DAV/1.0.3, PHP/4.1.2 e mod_perl/1.26 .

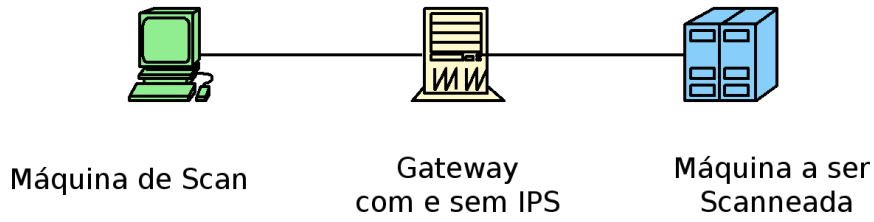

Figura 4.1: Ambiente de teste do IPS

Para realizar o estudo, foram utilizadas três ferramentas de segurança, já descritas anteriormente: o *Nmap*, o *Nessus* e o *Nikto*. As três ferramentas foram executadas duas vezes, com e sem a influência do IPS. Em todos os testes a *bridge* estava ativa.

A Figura 4.2 demonstra alguns alertas dos *scans* e das varreduras das vulnerabilidades que foram registrados e que podem ser verificados e analisados no *BASE*.

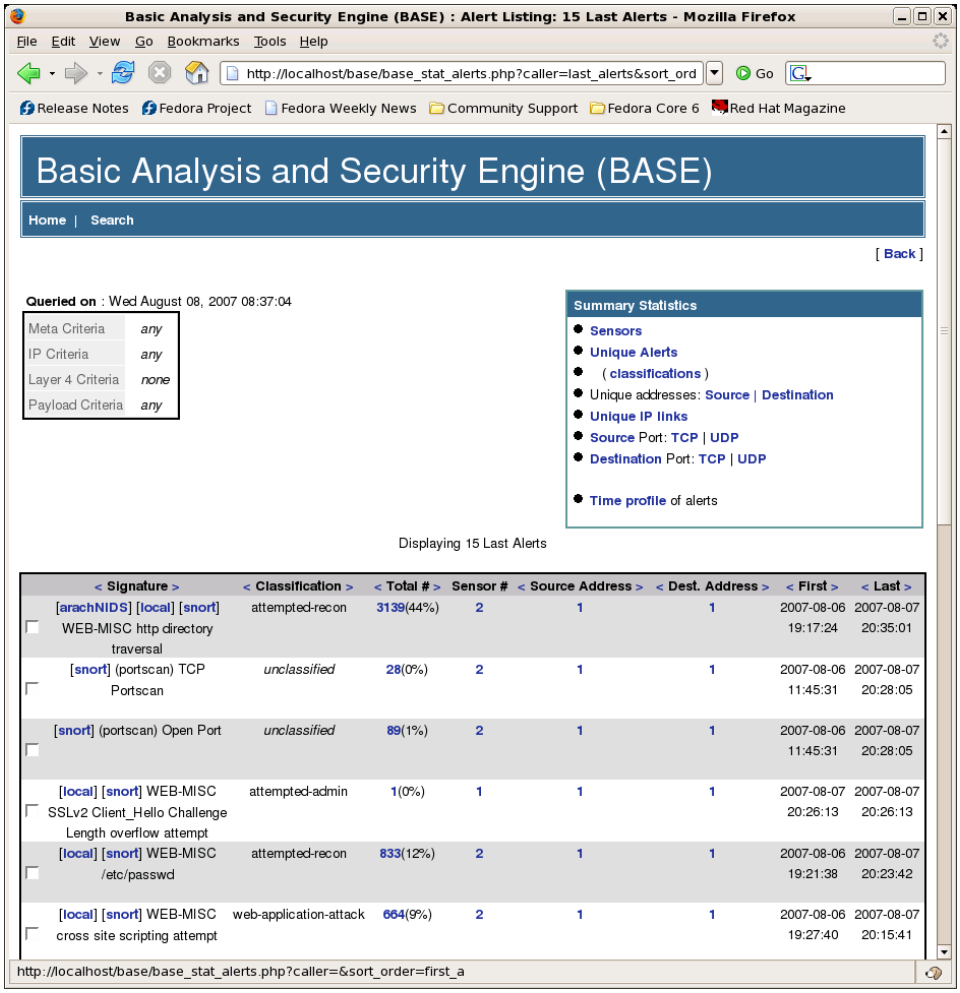

Figura 4.2: *Logs* de *portscan* e de vulnerabilidades testadas

4.1 Testes com o Nmap

O *Nmap* foi utilizado para comparar a diferença dos resultados obtidos quanto às portas ativas e a versão do sistema operacional da máquina a ser

scanneada, empregando diversos tipos de *portscan*, com e sem o IPS. Os tipos de *portscan* utilizadas foram: *Connect, TCP SYN, ACK, TCP Null, FIN e Xmas*.

Na Tabela 4.1 é apresentado um resumo de alguns testes realizados para medir a performance, e quanto a solução apresentada pode contribuir para atrasar ou impedir um *portscan*.

O portscan do tipo ACK não é para identificar portas abertas e sim para mapear a presença de firewall, se é do tipo stateful ou não e quais portas são filtradas (`Lyon, 2008`).

Os números entre parênteses são a quantidade de portas listadas. Nos *portscan* de tipo *Connect*, *TCP SYN*, *ACK* e *TCP Null*, com a presença do IPS, a quantidade de portas listadas foi maior, isto é, foram listadas portas que estavam fechadas. O IPS bloqueou estas portas de tal forma que o *Nmap* não conseguiu identificar se estavam abertas ou fechadas, por este motivo as listou e classificou como filtradas.

É possível notar que o tempo para varredura de portas aumentou em todos os tipos de *scan* realizados pelo *Nmap*. No pior caso, o aumento para a execução do *portscan* foi na ordem de dez (10) vezes, como pode-se verificar no tipo "*Connect*". Já no melhor caso, o aumento no tempo para executar o *portscan* foi na ordem de onze mil e quatrocentos e noventa e sete (11497) vezes, que é o caso do tipo "*ACK Scan*".

Nota-se também pelos resultados dos *portscans* que o IPS conseguiu impedir totalmente a identificação das portas abertas nos tipos "*FIN scan*" e "*Xmas scan*", pois das cinco (5) portas abertas (22/tcp ssh, 80/tcp http, 111/tcp rpcbind, 443/tcp https e 6000/tcp X11), nenhuma foi identificada. Entretanto, nos demais tipos de *scan* (exceto o "*ACK scan*"), todas as cinco (5) portas abertas foram descobertas, o único ganho na utilização do IPS nesses outros tipos é que o tempo para execução do *portscan* aumentou. Talvez não seja tão

perceptível o ganho desse atraso neste ambiente de teste, mas em uma rede real, com centenas de máquinas, com certeza a ação desta técnica de ataque seria afetada com a presença do IPS.

Tabela 4.1: Identificação de portas Sem IPS versus Com IPS

TIPO DE *SCAN*	SEM IPS	IDENTIFICAÇÃO SEM IPS	COM IPS	IDENTIFICAÇÃO COM IPS
Connect (-sT)	1,307s	Identificou (5)	12,845s	Identificou (28)
TCP SYN (-sS)	0,275s	Identificou (5)	349,359s	Identificou (28)
ACK (-sA)	0,276s	Não Identificou	3173,187s	Não Identificou (5)
TCP Null (-sN)	1,550s	Identificou (5)	75,319s	Identificou (8)
FIN (-sF)	1,429s	Identificou (5)	36,544s	Não Identificou
Xmas (-sX)	1,429s	Identificou (5)	36,532s	Não Identificou

A Tabela 4.2, sumariza os testes realizados para medir a performance, e quanto a solução apresentada pode contribuir para atrasar ou impedir um *portscan* que utiliza a opção para detectar o Sistema Operacional da máquina testada (opção "-O" do *Nmap*).

Em relação ao tempo necessário para executar o *portscan*, os resultados encontrados seguem o padrão dos testes apresentados na Tabela 4.1.

Ainda na Tabela 4.2, nas colunas "Identificação Sem IPS" e "Identificação Com IPS" mostram os resultados do mecanismo de identificação do Sistema Operacional pelo *Nmap*. Nela, pode-se verificar que o IPS dificultou todos os tipos de *scan*, principalmente o tipo "*TCP Null*" que sem IPS identificou com sucesso a versão do Sistema Operacional mas com IPS não o identificou.

Tabela 4.2: *Nmap* com opção de identificação do sistema operacional

TIPO DE *SCAN* C/ OPÇÃO "-O"	SEM IPS	IDENTIFICAÇÃO SEM IPS	COM IPS	IDENTIFICAÇÃO COM IPS
Connect (-sT)	1,611s	Não Identificou	28,936s	Não Identificou (Guess)
TCP Syn (-sS)	1,648s	Não Identificou	105,501s	Não Identificou
ACK (-sA)	0,572s	Identificou	3148,779s	Identificou
TCP Null (-sN)	1,847s	Identificou	80,650s	Não Identificou
FIN (-sF)	1,843s	Identificou	37,250s	Identificou
XMAS (-sX)	1,747s	Identificou	37,271s	Identificou

4.2 *Testes com o Nessus*

Comparando-se as Figuras 4.3 e 4.4, pode-se notar que o IPS protegeu o serviço HTTP durante a varredura de vulnerabilidades efetuada pelo *Nessus*. O IPS impediu o *Nessus* de obter o nome e a versão do servidor HTTP (*Apache*).

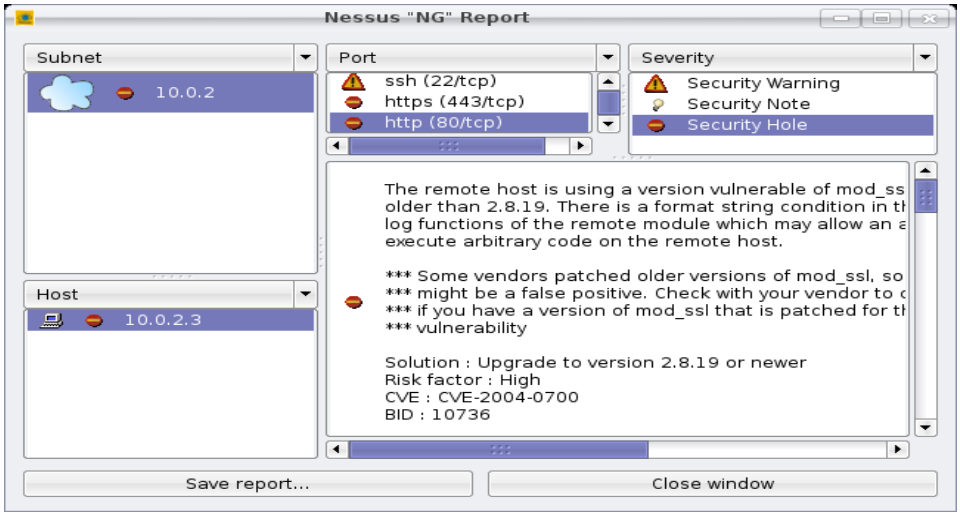

Figura 4.3: *Nessus* sem IPS

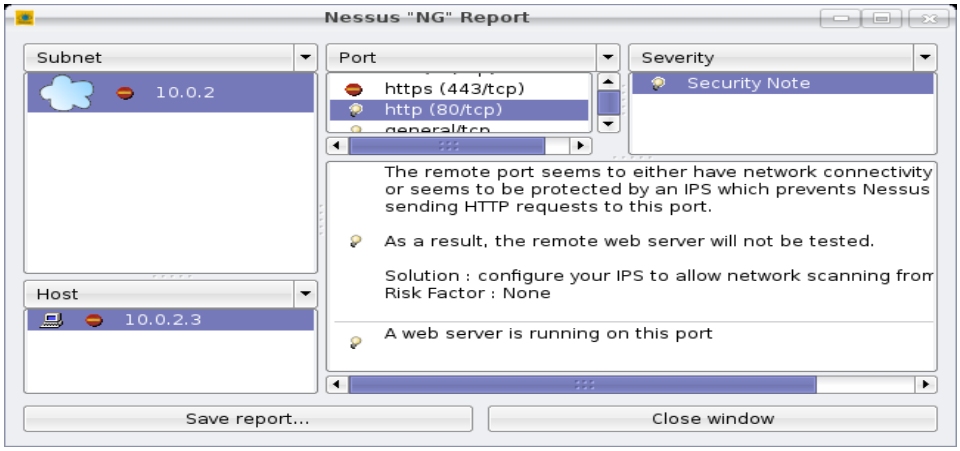

Figura 4.4: *Nessus* com IPS

Foram encontradas as seguintes vulnerabilidades, sem a presença do IPS, referentes ao HTTP e HTTPS:

- versão antiga do apache/1.3.23;
- versão antiga do mod_ssl/2.8.7;
- versão antiga do PHP/4.1.2;
- versão antiga do OpenSSL/0.9.6b.

Estas versões antigas possuem vulnerabilidades conhecidas, mas segundo o *Nessus*, podem ser solucionadas com a atualização para uma versão atual.

A Tabela 4.3, demonstra a diminuição de vulnerabilidades, alertas e notas de segurança proporcionada pela presença do IPS. As vulnerabilidades encontradas se devem ao serviço HTTPS, pois devido a ser criptografado, não pôde ter o tráfego inspecionado e não foi protegido pelo IPS. O serviço SSH foi identificado pelo *Nessus* mas foi impedido pelo IPS de realizar qualquer análise.

Tabela 4.3: Vulnerabilidades, Alertas e Notas obtidos pelo *Nessus*

	SEM IPS	COM IPS
Vulnerabilidades	13	7
Alertas	15	9
Notas de Segurança	32	21

4.3 Testes com o Nikto

A Figura 4.5 mostra que o *Nikto 2.01*, sem o IPS, demorou oito (8) segundos para ser executado e encontrou vinte e quatro (24) vulnerabilidades no servidor *Web* com *Apache 1.3.23*. As vulnerabilidades encontradas foram as seguintes:

- doze (12), protocolo HTTP;
- manual do servidor *web* encontrado;
- indexação de diretório habilitada: /icons
- versão antiga do apache/1.3.23;
- versão antiga do mod_ssl/2.8.7;
- versão antiga do OpenSSL/0.9.6b;
- versão antiga do DAV/1.0.3;
- versão antiga do PHP/4.1.2;
- versão antiga do mod_perl/1.26;
- versão do apache é vulnerável a um DoS (Denial of Service) remoto e execução de código;
- versão do PHP pode permitir o acesso a arquivos desautorizados a invasores;
- versão do apache é vulnerável a um *buffer overflow,* que permite aos

invasores encerrar qualquer processo do sistema;

– versão do apache é vulnerável a *overflows* no *mod_rewrite* e *mod_cgi*.

```
nikto-2.01$ ./nikto.pl -h 10.0.2.3
---------------------------------------------------------------------------
- Nikto 2.01/2.01     -     cirt.net
+ Target IP:       10.0.2.3
+ Target Hostname: 10.0.2.3
+ Target Port:     80
+ Start Time:      2007-12-29 10:23:05
---------------------------------------------------------------------------
+ Server: Apache/1.3.23 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b DAV/1.0.3 PHP/4.1.2 mod_perl/1.26
+ Allowed HTTP Methods: GET, HEAD, OPTIONS, TRACE, POST, PUT, DELETE, CONNECT, PATCH, PROPFIND,
PROPPATCH, MKCOL, COPY, MOVE, LOCK, UNLOCK
+ OSVDB-877: HTTP method ('Allow' Header): 'TRACE' is typically only used for debugging and should be disabled. This
message does not mean it is vulnerable to XST.
+ OSVDB-397: HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server.
+ OSVDB-5646: HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.
+ OSVDB-0: HTTP method ('Allow' Header): 'CONNECT' may allow server to proxy client requests.
+ OSVDB-13431: HTTP method ('Allow' Header): 'PROPFIND' may indicate DAV/WebDAV is installed. This may be used
to get directory listings if indexing is allowed but a default page exists.
+ OSVDB-425: HTTP method ('Allow' Header): 'PROPPATCH' indicates DAV/WebDAV is installed.
+ OSVDB-5647: HTTP method ('Allow' Header): 'MOVE' may allow clients to change file locations on the web server.
+ Apache/1.3.23 appears to be outdated (current is at least Apache/2.2.6). Apache 1.3.39 and 2.0.61 are also current.
+ mod_ssl/2.8.7 appears to be outdated (current is at least 2.8.30) (may depend on server version)
+ OpenSSL/0.9.6b appears to be outdated (current is at least 0.9.8f) (may depend on server version)
+ DAV/1.0.3 appears to be outdated (current is at least 2)
+ PHP/4.1.2 appears to be outdated (current is at least 5.2.4)
+ mod_perl/1.26 appears to be outdated (current is at least 5.8.0)
+ Apache/1.3.23 - Apache 1.x up 1.2.34 are vulnerable to a remote DoS and possible code execution. CAN-2002-0392.
+ PHP/4.1.2 mod_perl/1.26 - PHP below 4.3.3 may allow local attackers to safe mode and gain access to unauthorized files.
BID-8201.
+ Apache/1.3.23 - Apache 1.3 below 1.3.27 are vulnerable to a local buffer overflow which allows attackers to kill any
process on the system. CAN-2002-0839.
+ Apache/1.3.23 - Apache 1.3 below 1.3.29 are vulnerable to overflows in mod_rewrite and mod_cgi. CAN-2003-0542.
+ OSVDB-637: GET /~root - Enumeration of users is possible by requesting ~username (responds with Forbidden for real
users, not found for non-existent users).
+ OSVDB-682: GET /usage/ : Webalizer may be installed. Versions lower than 2.01-09 vulnerable to Cross Site Scripting
(XSS). CA-2000-02.
+ OSVDB-877: TRACK / : TRACK option ('TRACE' alias) appears to allow XSS or credential theft. See
http://www.cgisecurity.com/whitehat-mirror/WhitePaper_screen.pdf for details
+ OSVDB-877: TRACE / : TRACE option appears to allow XSS or credential theft. See
http://www.cgisecurity.com/whitehat-mirror/WhitePaper_screen.pdf for details
+ OSVDB-3092: GET /manual/ : Web server manual found.
+ OSVDB-3268: GET /icons/ : Directory indexing is enabled: /icons
+ 4343 items checked: 24 item(s) reported on remote host
+ End Time:       2007-12-29 10:23:13 (8 seconds)
---------------------------------------------------------------------------
•      1 host(s) tested
```

Figura 4.5: *Nikto* sem IPS

Na Figura 4.6, com o IPS, demorou aproximadamente dois (2) segundos,

mas não conseguiu identificar a porta HTTP aberta e suas respectivas vulnerabilidades. O IPS impediu a varredura.

Uma dúvida foi gerada, quanto ao resultado obtido pelo Nikto, devido ao fato dele não ter identificado a porta HTTP aberta. Por isto, para confirmar o funcionamento do IPS, foi ativado o *tcpdump* no servidor *Web* para verificar a chegada de pacotes com e sem IPS ativado.

No *log* do *tcpdump* com IPS (que está presente no Apêndice B) o *Nikto* demorou aproximadamente dois (2) segundos fazendo as tentativas de varredura na porta HTTP. Em alguns momentos, alguns pacotes passaram e pelo menos duas (2) conexões foram estabelecidas, mas mesmo assim ele não reconheceu a porta HTTP aberta. Isto pode ser erro de programação na ferramenta.

```
nikto-2.01$ ./nikto.pl -h 10.0.2.3
---------------------------------------------------------------------------
- Nikto 2.01/2.01     -     cirt.net
+ No HTTP(s) ports found on 10.0.2.3 / 80
+ 1 host(s) tested
```

Figura 4.6: *Nikto* com IPS

5 Conclusões

Este trabalho apresentou um estudo de impacto na eficiência dos ataques através da utilização de IPS transparente. A implementação do ambiente de testes foi realizado com o *Snort*, o *BASE* e o *pmgraph* que garantiram a detecção e o bloqueio das tentativas de intrusão, facilidade de leitura dos *logs* e o acompanhamento das estatísticas do *Snort*. A configuração do IPS em modo *bridge*, transparente, permite a sua instalação em um ambiente de rede sem a necessidade de reconfigurar os *hosts* e dispositivos de rede.

Os testes realizados com o *Nmap* demonstraram que, em todos casos, o IPS gerou grande atraso nos *scans* e, em certos casos, impediu a identificação das portas ativas e a identificação do sistema operacional do servidor.

Os testes com o *Nessus* demonstraram que o IPS protegeu as portas HTTP e SSH impedindo a obtenção das suas vulnerabilidades e falhas de segurança. O IPS bloqueou a varredura realizada pelo *Nikto* e, por isto, nenhuma vulnerabilidade ou falha de segurança foi encontrada.

Portanto, os testes realizados com o *Nmap*, *Nessus* e *Nikto* confirmaram que o IPS pode trazer ganhos substanciais de segurança em um ambiente de rede.

Cumpre ressaltar que, em um ambiente de rede, a utilização do IPS não exclui o emprego do IDS. O IPS deve ser usado para proteger o sistema em conjunto com o IDS que analisa todo tráfego para verificar se existem mais pacotes maliciosos trafegando pela rede. O IDS deve ser usado para apurar a eficiência do IPS e também para testar as regras do *Snort* antes de serem ativadas no IPS, evitando assim os falsos positivos.

Insta notar que, nenhum dispositivo de rede irá garantir 100% de

segurança. Por este motivo, deve-se contar com mais de um dispositivo. Deve-se adotar uma estratégia de segurança em profundidade (*security in depth*), empregando múltiplos mecanismos entre os adversários e seus alvos, incluindo métodos de proteção e detecção.

Ao longo do desenvolvimento do trabalho, idéias complementares e novas idéias surgiram para trabalhos futuros. A seguir, são listadas algumas dessas idéias:

- Implementação de um IPS com *honeypot*[5];
- Implementação de uma solução integrada de IPS com IDS;
- Implementação de um IPS com múltiplos processos do *Snort* no modo *inline*;
- Integrar o IPS com o anti-virus *clamav*;
- IPS em *cluster*[6], pois se houver apenas um IPS e este parar de repassar os pacotes válidos, o segmento a ele ligado perderá toda a conectividade à rede.

5 O *honeypot* é uma ferramenta que tem como principal função colher informações para estudo de ataques. http://www.honeynet.org

6 Um *cluster*, ou aglomerado de computadores, é formado por um conjunto de computadores, que utiliza-se de um tipo especial de sistema operacional classificado como sistema distribuído. É construído muitas vezes a partir de computadores convencionais (*desktops*), sendo que estes vários computadores são ligados em rede e comunicam-se através do sistema de forma que trabalham como se fosse uma única máquina de grande porte (`Botelho, 2006`).

6 Referências Bibliográficas

ANDREASSON, Oskar, Iptables Tutorial 1.1.19. The conntrack entries. Disponível na Internet via www. url: http://www.faqs.org/docs/iptables/theconntrackentries.html. Arquivo capturado em 25 de abril de 2008.

BEALE, Jay, BAKER, Andrew R., ESLER, Joel. Snort – IDS and IPS Toolkit. Burlington:Syngress, 2007. 734p.

BOTELHO, Marco Antônio Faria. Alta Disponibilidade em Firewall utilizando PFSYNC e CARP sobre FreeBSD. Lavras, UFLA, 2006. (Monografia apresentada ao Departamento da Ciência da Computação da UFLA para obtenção de Título de Especialista em Administração de Redes Linux).

CERT.br. Gráficos de Incidentes Reportados. Disponível via url http://www.cert.br/stats/incidentes/ . Arquivos capturados em 10 de agosto de 2007.

CIRT.net. Nikto 1.36. Disponível na Internet via www. url:http://www.cirt.net/code/nikto.shtml. Arquivo capturado em 14 de agosto de 2007.

COELHO, Laura C. M., BENTO, Ricardo J. Ferramentas de Esteganografia e seu uso na Infowar. In: 1º Conferência Internacional de Perícias em Crimes Cibernéticos, 2004, Brasília. Anais... Brasília: Departamento de Polícia Federal. 2004, p. 14.

EDGE, Intrusion Detection with BASE and Snort. Disponível na Internet via url http://www.howtoforge.com/intrusion_detection_base_snort. Arquivo capturado em 20 de novembro de 2006.

HENMI, Anne, LUCAS, Mark, SINGH, Abhishek, CANTRELL, Chris. Firewall Policies and VPN Configurations. Rockland: Syngress, 2006. 482p.

HSBC. O que é uma Botnet?. Disponível na Internet via www. url: http://www.hsbc.com.br/common/seguranca/artigo-seguranca-o-que-e-botnet.shtml. Arquivo capturado em 25 de abril de 2008.

HSBC. Qual a diferença entre vírus, worm e cavalo de tróia?. Disponível na Internet via www. url: http://www.hsbc.com.br/common/seguranca/artigo-seguranca-diferenca-virus-worm.shtml. Arquivo capturado em 25 de abril de 2008.

LYON, Gordon "Fyodor", Nmap Network Scanning. Port Scanning Techniques. Disponível na Internet via www. url: http://nmap.org/book/man-port-scanning-techniques.html. Arquivo capturado em 25 de abril de 2008.

MAY, Chris, BAKER, Marie, GABBARD, Derek, GOOD,Travis, GRIMES, Galen, HOLMGREN, Mark, NOLAN, Richard, NOWAK, Robert, PENNLINE, Sean. Advanced Information Assurance Handbook. CERT/CC Training and Education Center, 2004.

MELO, Leonardo Garcia. BFW e MAILRELAY – Uma Abordagem para Firewalls com Baixa Intrusão. In: 1º Conferência Internacional de Perícias em Crimes Cibernéticos, 2004, Brasília. Anais... Brasília: Departamento de Polícia Federal. 2004, p. 143.

MetU, Pillo. IPS – Intrusion Detection & Prevention System.
Disponível na Internet via url http://snortattack.org . Arquivo capturado em 15 de novembro de 2006.

MURESAN, Marius. Nessus/Nmap. Linux Magazine – Anuário de Tecnologias Abertas 2007, São Paulo, v. 1, p.99, 2007.

NSS Labs. Intrusion Prevention (IPS). Disponível na Internet via www. url: http://nsslabs.com/ips. Arquivo capturado em 25 de abril de 2008.

NSS Labs. Intrusion Prevention Systems (IPS). Disponível na Internet via url http://nsslabs.com/white-papers/intrusion-prevention-systems-ips.html . Arquivo capturado em 20 de março de 2008.

PALAZZOLI, P., Valenza, M., Snort_inline as a solution. 2006.
Disponível na Internet via url http://www.hakin9.org
http://www.snortattack.org

RED HAT. Red Hat Enterprise Linux 4: Security Guide. Red Hat. 2005.

REHMAN, Rafeeq. Intrusion Detection Systems with Snort. New Jersey:

Prentice Hall, 2003. 263p.

ROGNESS, Nick, A How-To Guide for running snort_inline on FreeBSD. Disponível na Internet via www. url: http://freebsd.rogness.net/snort_inline Arquivo capturado em 25 de abril de 2008.

SANTOS, B. R. dos, Detecção de Intrusos Utilizando o Snort. Lavras, UFLA, 2005. (Monografia apresentada ao Departamento de Computação da UFLA para obtenção de Título de Especialista em Administração de Redes Linux).

SAVAGE, P., Snort Inline. Linux Gazette, Disponível na Internet via url http://linuxgazette.net/117/savage.html, Arquivo capturado em 20 de novembro de 2006.

SNAC. Systems and Network Analisys Center. National Security Agency. Enterprise Firewall Types, Disponível na Internet via www. url: http://www.nsa.gov/ . Arquivo capturado em 21 de junho de 2007.

Snort Brasil. Snort - The Open Source Network Intrusion Detection System. Disponível na Internet via www. url: http://www.clm.com.br/snort/necessidade.asp . Arquivo capturado em 25 de julho de 2007.

SNORT INLINE. What is snort_inline?. Disponível na Internet via www. url: http://snort-inline.sourceforge.net. Arquivo capturado em 25 de abril de 2008.

SNORT.ORG. What is snort?. Disponível na Internet via www. url: http://www.snort.org. Arquivo capturado em 25 de abril de 2008.

Apêndice A

Tutorial de Instalação e Configuração do IPS

Este tutorial foi baseado em MetU (2006), Palazzoli et al. (2006), Savage (2006), Edge (2006) e Beale et al.(2007).

Para implementar o IPS, foi instalado um computador com *Fedora Core 6*, *iptables*, *Snort* no modo *inline*, *bridge*, BASE e *Pmgraph* (*perfmonitor*).

Inicialmente foi instalado o *Fedora Core 6* no computador e realizada a atualização conforme Figura A.1.

```
# yum -y update
```

Figura A.1: Atualização da máquina

Para que o IPS possa funcionar corretamente, deve-se desativar o SELinux.

Instalação dos Pacotes

Para a instalação de um IPS no *Fedora Core 6* é necessário a instalação de alguns pacotes (pode ser pelo gerenciador de pacotes), conforme Figura A.2.

```
# yum -y install libnet10 rrdtool rrdtool-devel rrdtool-perl gcc httpd ntp php-mysql mysql-server mysql-devel iptables-devel libstdc++-devel gcc-c++ gcc-objc glib2-devel bridge-utils pcre pcre-devel libpcap-devel php php-devel
```

Figura A.2: Pacotes necessários para o IPS

Instalação do Código Fonte do Kernel

No caso de um sistema *RedHat/Fedora Core* é necessário corrigir os cabeçalhos do "*glibc*" (pacote *glibc-kernheaders*) para que se possa instalar o *Snort*. Para corrigir este problema deve-se seguir os passos seguintes.

Primeiro, é necessário verificar a versão do kernel, executando os comandos da Figura A.3.

```
# uname -a
# Linux 2.6.20.1
```

Figura A.3: Verificação da versão do kernel

Fazer o download e instalação do código fonte da versão do kernel necessário do site *http://www.kernel.org* , conforme Figura A.4.

```
# cd /usr/src
#wget  http://ftp.kernel.org/pub/linux/kernel/v2.6/linux-2.6.20.1.tar.bz2
# tar jxvf linux-2.6.20.1.tar.bz2
# cd /usr/include
# mv linux linux.ORIGINAL
# ln -s /usr/src/linux-2.6.20.1/include/linux   linux
```

Figura A.4: Download e instalação do código fonte do kernel

Instalação do Snort

Deve-se obter de *www.snort.org* a versão mais atual do *Snort* e realizar a instalação conforme Figura A.5.

```
# cd /usr/local/src
# wget http://www.snort.org/dl/current/snort-2.6.1.5.tar.gz
# tar zxvf snort-2.6.1.5.tar.gz
# cd snort-2.6.1.5
```

Figura A.5: Instalação do *Snort*

O *Snort* deve ser compilado para ter *flex-response* habilitado para resetar o tráfego que deve ser bloqueado. O *Snort* no modo *inline* derruba tráfego baseado em regras previamente carregadas. Para configurar, compilar e linkar o *Snort*, deve-se executar os comandos da Figura A.6.

```
# ./configure --enable-inline --with-mysql --enable-flexresp –enable-timestats --enable-
perfmonitor
# make
# make install
```

Figura A.6: Configurar, compilar e linkar o *Snort*

Configuração do MySQL

Para inicializar o *mysql*, deve-se executar o comando da Figura A.7.

```
# /etc/init.d/mysqld start
```

Figura A.7: Inicializar o *MySQL*

Para criar a senha do usuário root, deve-se executar os comando da Figura A.8.

```
# mysqladmin -u root password <senha>
```

Figura A.8: Criar o usuário *root*

Criar o banco de dados, conforme mostrado na Figura A.9.

```
# mysqladmin -u root -p create snort
```

Figura A.9: Criar o Banco de Dados

Criar os privilégios de acesso ao banco, conforme Figura A.10.

```
# mysql -u root -p
Enter password:
mysql> grant INSERT,SELECT on root.* to snort@localhost;
mysql> SET PASSWORD FOR snort@localhost=PASSWORD('<senha>');
mysql> grant CREATE,INSERT,SELECT,DELETE,UPDATE on snort.* to snort@localhost;
mysql> grant CREATE,INSERT,SELECT,DELETE,UPDATE on snort.* to snort;
mysql> flush privileges;
mysql> exit;
```

Figura A.10: Criar privilégios para o banco

Criar as tabelas conforme mostrado na Figura A.11.

```
# mysql -u root -p < /usr/local/src/snort-2.6.1.5/schemas/create_mysql snort
```

Figura A.11: Criar tabela

Verificar as tabelas criadas conforme apresentado na Figura A.12.

```
# mysqlshow -u snort -p snort
Enter password:
Database: snort
+---------------------------+
|      Tables               |
+---------------------------+
| data                      |
| detail                    |
| encoding                  |
| event                     |
| icmphdr                   |
| iphdr                     |
| opt                       |
| reference                 |
| reference_system          |
| schema                    |
| sensor                    |
| sig_class                 |
| sig_reference             |
| signature                 |
| tcphdr                    |
| udphdr                    |
+---------------------------+
```

Figura A.12: Verificar as tabelas criadas

Configurar o arquivo *snort.conf* para que faça uso do MySQL. Para isto, deve-se retirar o comentário da linha apresentada na Figura A.13.

```
# output database: log, mysql, user=snort password=<senha> dbname=snort host=localhost
```

Figura A.13: Ativar o *MySQL* no *Snort*

Configuração do PHP 5

Editar o arquivo */etc/php.ini* e modificar a linha apresentada na Figura A.14.

```
#cd /etc
# vi php.ini
        Mudar a linha “error_reporting=E_ALL” para:
                error_reporting=E_ALL & ~E_NOTICE
```

Figura A.14: Configurar o PHP

Configuração do Apache

Configurar o Apache conforme Figura A.15.

```
# cd /etc/httpd/conf
# vi httpd.conf
                DocumentRoot “/var/www/html/”
                ## Acrescentar no fim da seção LoadModule
                LoadModule php5_module modules/libphp5.so

                AddType application/x-httpd-php .php
                AddType application/x-httpd-php-source .phps
```

Figura A.15: Configurar o Apache

Ativação do Apache

Ativar o Apache executando o comando da Figura A.16.

```
# /etc/init.d/httpd start
```

Figura A.16: Ativar o Apache

Instalação do BASE e ADODB

Instalar o BASE e o ADODB seguindo os comandos da Figura A.17.

```
# cd /var/www/html

Baixar o BASE de base.secureideas.net .
Baixar o ADODB de adodb.sourceforge.net .

# tar zxvf base-1.3.6.tar.gz
# mv base-1.3.6   base
# chown -R root.root base

# tar zxvf adodb495a.tgz
```

Figura A.17: Instalar o BASE e o ADODB

Configuração do BASE

Para realizar a configuração é necessário modificar a permissão do diretório do BASE, conforme mostrado na Figura A.18.

```
# cd /var/www/html
# chmod 757 base
```

Figura A.18: Modificar a permissão de diretório *base*

Abrir o *browser* e digitar a linha da Figura A.19, para abrir a janela de configuração do BASE.

```
# http://localhost/base/setup
```

Figura A.19: Abrir janela de configuração do BASE

Na janela de configuração, conforme Figura A.20, deve-se clicar em "*Continue*" para passar para a próxima etapa.

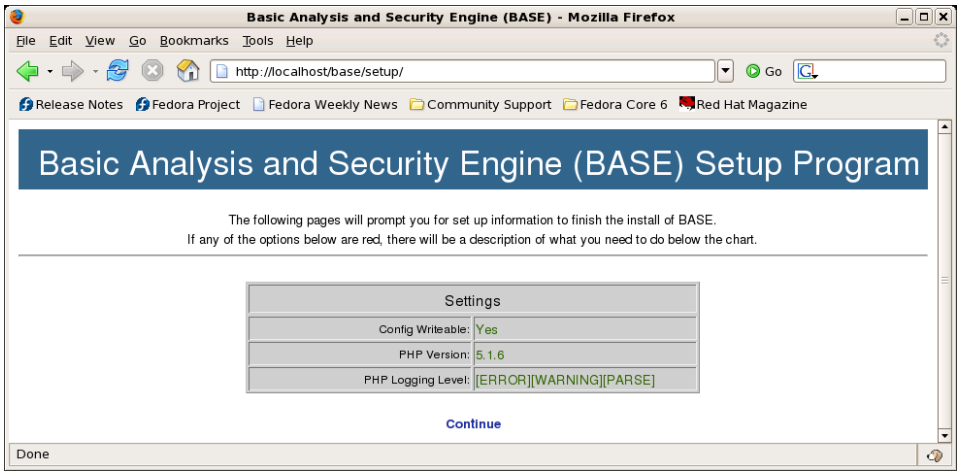

Figura A.20: Configuração do BASE

Na janela da Figura A.21, deve-se inserir o caminho para ADODB: */var/www/html/adodb*

Logo após, deve-se clicar em "*Submit Query*".

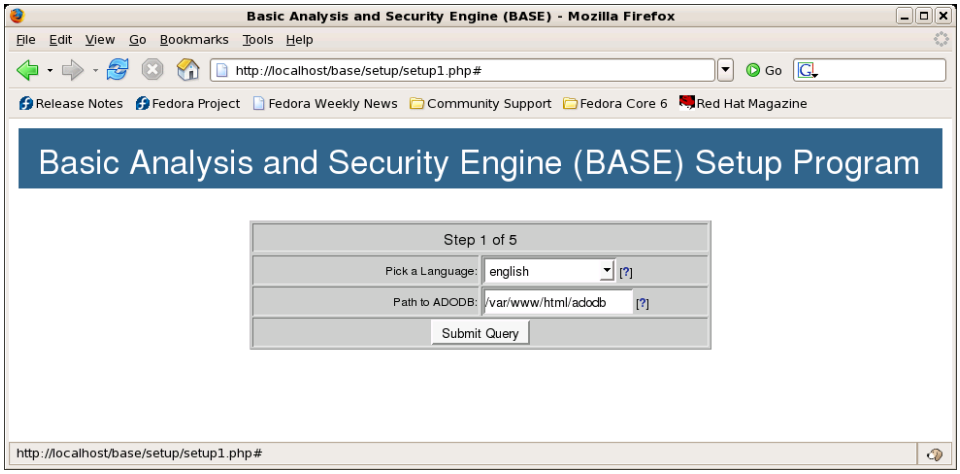

Figura A.21: Configuração do BASE - 1

Na janela da Figura A.23, deve-se definir os parâmetros listados na Figura A.22.

Definir o tipo de banco de dados: MySQL Nome do Banco de Dados: snortdb Host do Banco de Dados: localhost Port do Banco de Dados: Usuário do Banco de Dados: root Senha do Banco de Dados: <senha>

Figura A.22: Definir parâmetros

Os outros campos devem ficar em branco.

Ao termino, deve-se clicar em "*Submit Query*".

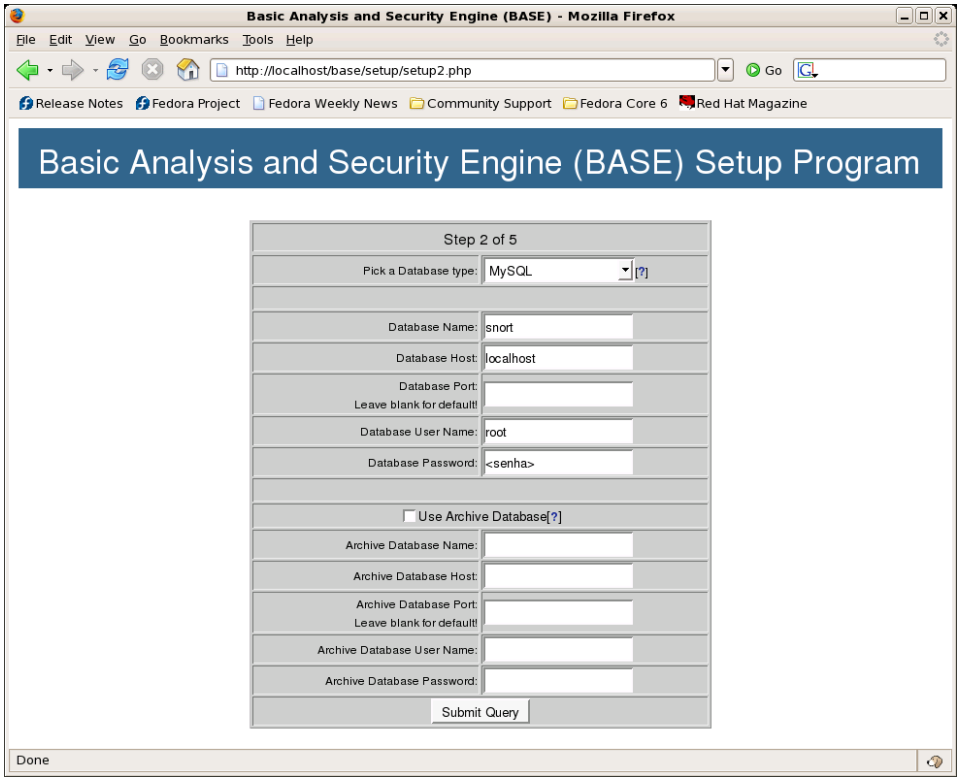

Figura A.23: Configuração do BASE - 2

Na janela da Figura A.24, deve-se deixar os espaços em branco e clicar em "*Submit Query*".

Obs.: Se desejar configurar autenticação para BASE pode-se fazer aqui.

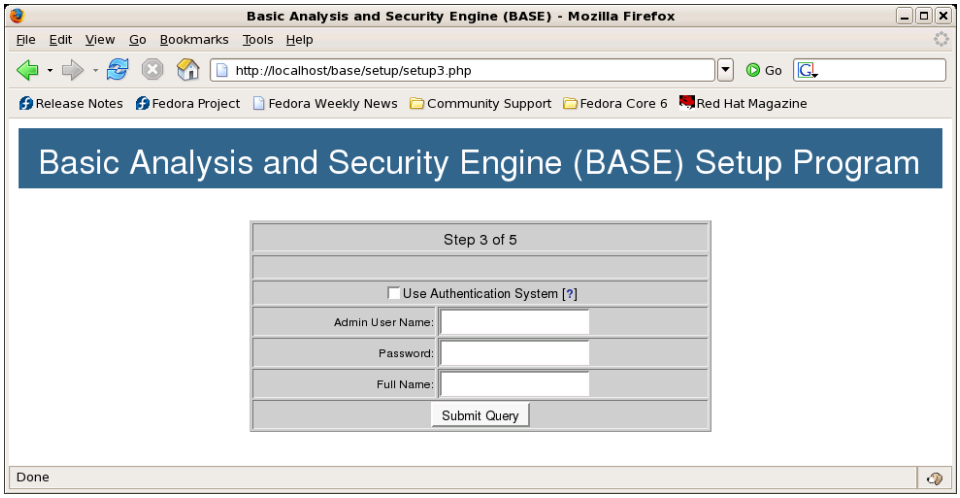

Figura A.24: Configuração do BASE - 3

Na janela da Figura A.25, deve-se clicar em "*Create BASE AG*" para criar o banco de dados.

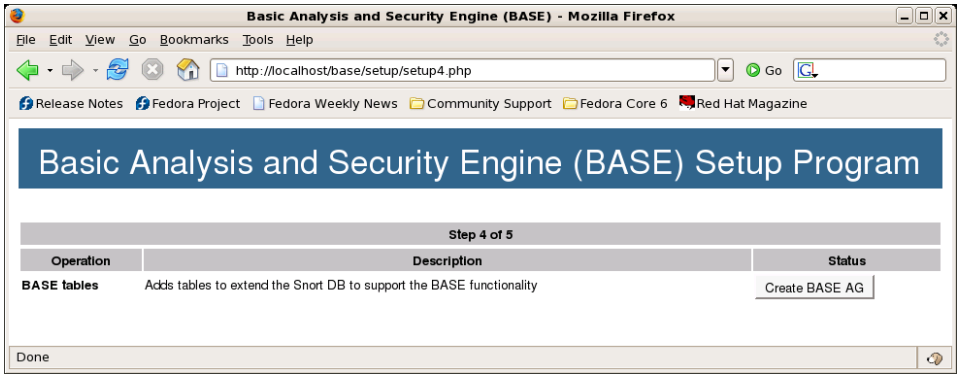

Figura A.25: Configuração do BASE - 4

Na janela da Figura A.26, quando terminar a criação do banco, deve-se clicar em "**step 5**" que se encontra na mensagem "*Now continue to **step 5**...*".

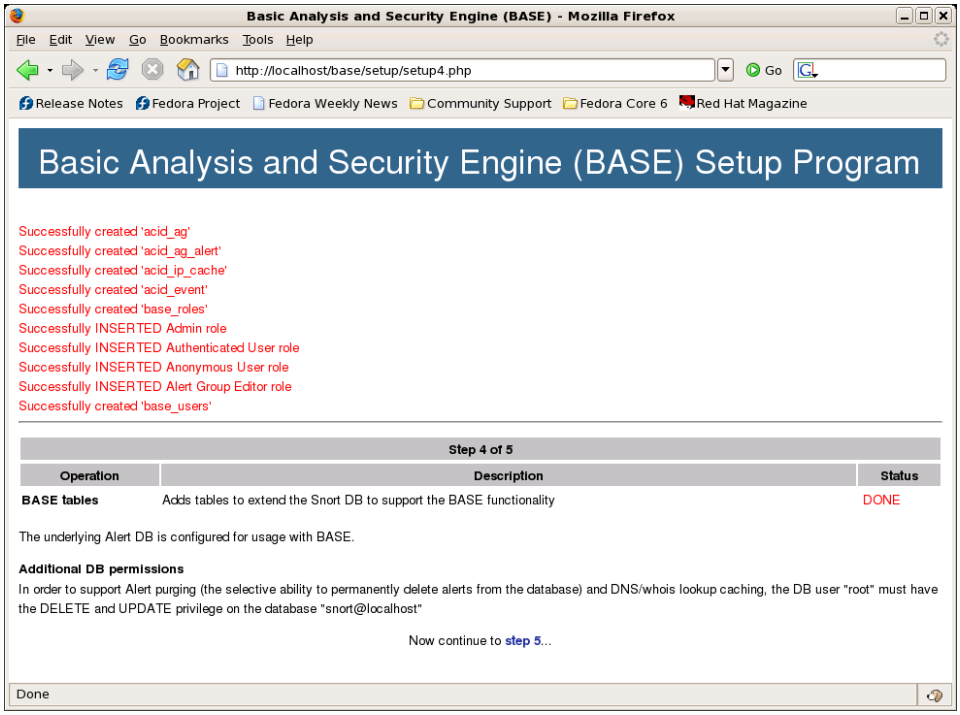

Figura A.26: Configuração do BASE - 5

Concluída a configuração, a janela do BASE será apresentada conforme Figura A.27.

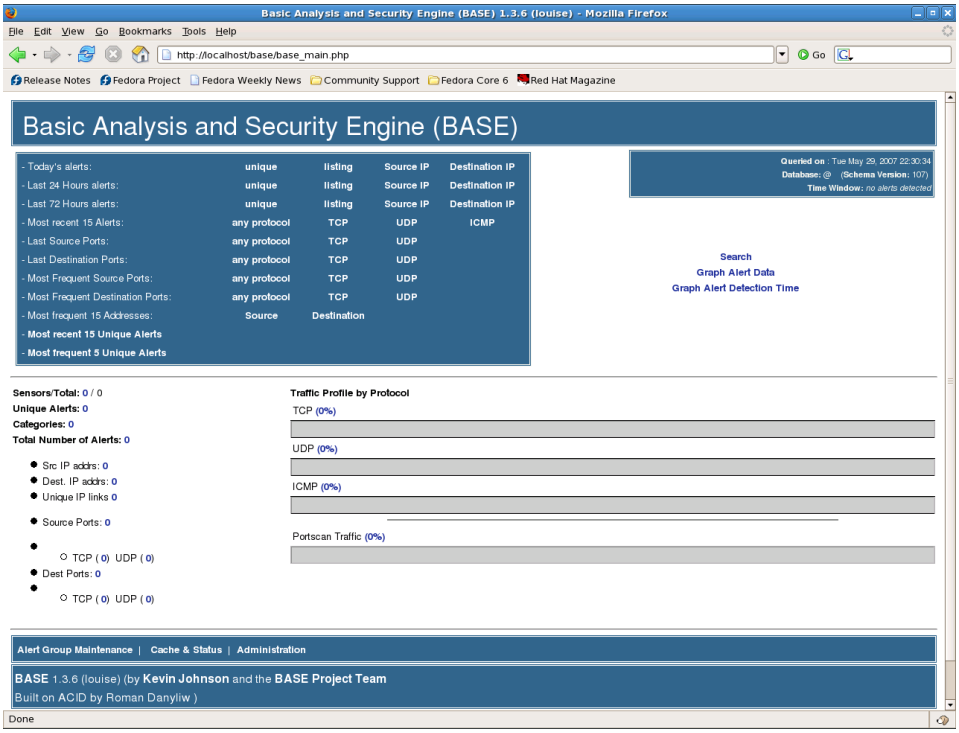

Figura A.27: BASE

Com a conclusão da configuração do BASE deve-se retornar as permissões do diretório *base* para 755, conforme mostrado na Figura A.28.

```
# cd /usr/local/www
# chmod 755 base
```

Figura A.28: Retornar as permissões padrão do diretório *base*

Para que o "*Graph Alert Data*" no BASE funcione deve-se instalar os pacotes listados na Figura A.29.

```
yum -y install gd gd-devel php-gd  php-pear php-pear-DB php-pear-Image-Canvas php-pear-Image-Color php-pear-Image-Graph php-pear-Log php-pear-MDB2
```

Figura A.29: Pacotes para *Graph Alert Data* do BASE

Configuração do Snort

Agora que o *Snort* está instalado deve-se baixar as assinaturas (regras). O site *snort.org* oferece três formas de *download*: assinatura paga, registrada e sem registro. Neste trabalho, foi utilizado o *download* registrado por oferecer regras mais atuais do que o não registrado. Por não ser paga, a atualização não é tão freqüente quanto a assinatura paga.

As regras foram baixadas no diretório "*/usr/local/src/IPS-rules*". O arquivo *snort.conf* foi copiado para o diretório "*/etc/IPS*".

Terminada a instalação das regras, deve-se configurar o *snort.conf* de acordo com o que se pretende proteger.

Para que o IPS possa bloquear os pacotes maliciosos, deve-se mudar as regras do *Snort* de *alert* para *drop*. Para facilitar este procedimento, pode-se utilizar o *script Muda-para-DROP.sh*, que se encontra neste Apêndice.

Para Ativar o IPS deve-se executar o *script IPS-ativa.sh*, conforme Figura A.30, listado neste Apêndice .

```
# /root/IPS-ativa.sh
```

Figura A.30: Ativar o IPS

Instalação do Pmgraph

Instalar o *Pmgraph* digitando os comandos conforme apresentado na Figura A.31.

```
# cd /usr/local/src
# wget http://people.su.se/~andreaso/perfmon-graph/pmgraph-0.2.tar.gz
# tar zxvf pmgraph-0.2.tar.gz
# mkdir /var/www/html/usage
```

Figura A.31: Instalar o *Pmgraph*

Executar o *Pmgraph* e gerar estatísticas conforme comando da Figura A.32.

```
# /usr/local/src/pmgraph-0.2/pmgraph.pl  /var/www/html/usage  /var/log/snort/snort.stats
```

Figura A.32: Executar o *Pmgraph* e gerar estatísticas

Pode-se acrescentar o comando no *crontab* para atualizar as estatísticas automaticamente.

Acessando os Dados do BASE e Pmgraph

Para acessar os dados de BASE e Pmgraph, deve-se digitar as url no browser, conforme apresentadas na Figura A.33.

```
http://endereçoIPservidor/base
http://endereçoIPservidor/usage
```

Figura A.33: Acessar o *BASE e Pmgraph*

Arquivos de configuração no IPS

O script da Tabela A.1, pode ser usado para mudar as regras do *Snort* de *alert* para *drop*.

Tabela A.1: *Muda-para-DROP.sh*

```
#!/bin/bash
cd /usr/local/src/IPS-snort-rules/rules
for file in $(ls -1 *.rules)
do
        sed -e 's:^alert:drop:g' ${file} > ${file}.new
        mv ${file}.new ${file} -f
done
```

O script da Tabela A.2 é utilizado para configurar e ativar o IPS.

Tabela A.2: *IPS-ativa.sh*

```
#!/bin/bash

# Ativa servicos MySQL e HTTP
/etc/init.d/mysqld start
/etc/init.d/httpd  start

# Define interfaces: interna, externa e de controle
ETHinterna="eth0"
ETHexterna="eth1"

# Configura Bridge
/sbin/ifconfig $ETHinterna 0.0.0.0 up
/sbin/ifconfig $ETHexterna 0.0.0.0 up
/usr/sbin/brctl addbr br0
/usr/sbin/brctl addif br0 $ETHinterna
/usr/sbin/brctl addif br0 $ETHexterna

# Bridge sem IP
/sbin/ifconfig br0 up

#desativa ip_forward e carrega módulo ip_queue
echo 0 > /proc/sys/net/ipv4/ip_forward
```

```
modprobe ip_queue

# Configura iptables
/sbin/iptables -F
/sbin/iptables -X RH-Firewall-1-INPUT

/sbin/iptables -i INPUT -i lo -j ACCEPT
/sbin/iptables -A INPUT -p all -j DROP
/sbin/iptables -I FORWARD -p all -j QUEUE
/sbin/iptables -A FORWARD -p all -j ACCEPT
/sbin/iptables -I OUTPUT -o lo -j ACCEPT
/sbin/iptables -A OUTPUT -p all -j DROP

/sbin/iptables -L

# Ativa IPS
/usr/local/bin/snort -Q -D -c /etc/IPS/snort.conf  --alert-before-pass

# FIM.
```

O script da Tabela A.3 serve para gerar os gráficos das estatísticas do *Snort*.

Tabela A.3: *exec-pmgraph.sh*

```
#!/bin/bash
#/usr/local/src/pmgraph-0.2/pmgraph.pl /var/www/html/usage  /var/log/snort/snort.stats
```

Apêndice B

A Tabela B.1 lista os resultados obtidos pelos testes de *scan* realizados com o *Nmap*.

Tabela B.1: Resultados obtidos com o *Nmap* com e sem IPS

Relatorio - nmap Sem IPS

nmap -sT 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 14:13 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT STATE SERVICE
22/tcp open ssh
80/tcp open http
111/tcp open rpcbind
443/tcp open https
6000/tcp open X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 1.307 seconds

nmap -sT -O 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 14:13 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT STATE SERVICE
22/tcp open ssh
80/tcp open http
111/tcp open rpcbind
443/tcp open https
6000/tcp open X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: broadband router|WAP
Running: Netgear embedded
OS details: Netgear DG834 or DG834G (wireless) DSL Router
Uptime: 0.164 days (since Fri Dec 28 10:17:12 2007)
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 1.611 seconds
nmap -sS 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 14:13 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports

```
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
111/tcp  open  rpcbind
443/tcp  open  https
6000/tcp open  X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 0.275 seconds

nmap -sS -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:13 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
111/tcp  open  rpcbind
443/tcp  open  https
6000/tcp open  X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: broadband router|WAP
Running: Netgear embedded
OS details: Netgear DG834 or DG834G (wireless) DSL Router
Uptime: 0.164 days (since Fri Dec 28 10:17:12 2007)
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 1.648 seconds

nmap -sA 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:13 BRST
All 1697 scanned ports on 10.0.2.3 are UNfiltered
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 0.276 seconds

nmap -sA -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:13 BRST
Warning:  OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1
open and 1 closed TCP port
All 1697 scanned ports on 10.0.2.3 are UNfiltered
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: WAP|broadband router
Running: D-Link Linux 2.4.X, Inventel embedded, Netgear embedded, Siemens linux
OS details: D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17, Inventel Livebox wireless
broadband router, Netgear DG834 or DG834G (wireless) DSL Router, Siemens Gigaset SE515dsl
wireless broadband router
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 0.572 seconds
```

```
nmap -sN 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:13 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
443/tcp  open|filtered https
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 1.550 seconds

nmap -sN -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:13 BRST
Warning:  OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1
open and 1 closed TCP port
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
443/tcp  open|filtered https
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: WAP|broadband router
Running: D-Link Linux 2.4.X, Inventel embedded, Netgear embedded, Siemens linux
OS details: D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17, Inventel Livebox wireless
broadband router, Netgear DG834 or DG834G (wireless) DSL Router, Siemens Gigaset SE515dsl
wireless broadband router
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 1.847 seconds

nmap -sF 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:14 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
443/tcp  open|filtered https
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 1.429 seconds

nmap -sF -O 10.0.2.3
```

```
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:14 BRST
Warning:  OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1
open and 1 closed TCP port
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
443/tcp  open|filtered https
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: WAP|broadband router
Running: D-Link Linux 2.4.X, Inventel embedded, Netgear embedded, Siemens linux
OS details: D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17, Inventel Livebox wireless
broadband router, Netgear DG834 or DG834G (wireless) DSL Router, Siemens Gigaset SE515dsl
wireless broadband router
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 1.843 seconds

nmap -sX 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:14 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
443/tcp  open|filtered https
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 1.429 seconds


nmap -sX -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:14 BRST
Warning:  OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1
open and 1 closed TCP port
Interesting ports on 10.0.2.3:
Not shown: 1692 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
443/tcp  open|filtered https
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: WAP|broadband router
```

Running: D-Link Linux 2.4.X, Inventel embedded, Netgear embedded, Siemens linux
OS details: D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17, Inventel Livebox wireless broadband router, Netgear DG834 or DG834G (wireless) DSL Router, Siemens Gigaset SE515dsl wireless broadband router
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 1.747 seconds

Relatorio - nmap Com IPS

nmap -sT 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 14:37 BRST
Interesting ports on 10.0.2.3:
Not shown: 1669 closed ports
PORT STATE SERVICE
22/tcp filtered ssh
23/tcp filtered telnet
80/tcp open http
110/tcp filtered pop3
111/tcp open rpcbind
135/tcp filtered msrpc
137/tcp filtered netbios-ns
139/tcp filtered netbios-ssn
143/tcp filtered imap
161/tcp filtered snmp
162/tcp filtered snmptrap
443/tcp open https
445/tcp filtered microsoft-ds
705/tcp filtered unknown
993/tcp filtered imaps
995/tcp filtered pop3s
1433/tcp filtered ms-sql-s
1434/tcp filtered ms-sql-m
3389/tcp filtered ms-term-serv
5800/tcp filtered vnc-http
5801/tcp filtered vnc-http-1
5802/tcp filtered vnc-http-2
5803/tcp filtered vnc-http-3
5900/tcp filtered vnc
5901/tcp filtered vnc-1
5902/tcp filtered vnc-2
5903/tcp filtered vnc-3
6000/tcp open X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 12.845 seconds

```
nmap -sT -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:38 BRST
Insufficient responses for TCP sequencing (2), OS detection may be less accurate
Insufficient responses for TCP sequencing (1), OS detection may be less accurate
Insufficient responses for TCP sequencing (2), OS detection may be less accurate
Interesting ports on 10.0.2.3:
Not shown: 1669 closed ports
PORT     STATE    SERVICE
22/tcp   filtered ssh
23/tcp   filtered telnet
80/tcp   open     http
110/tcp  filtered pop3
111/tcp  open     rpcbind
135/tcp  filtered msrpc
137/tcp  filtered netbios-ns
139/tcp  filtered netbios-ssn
143/tcp  filtered imap
161/tcp  filtered snmp
162/tcp  filtered snmptrap
443/tcp  open     https
445/tcp  filtered microsoft-ds
705/tcp  filtered unknown
993/tcp  filtered imaps
995/tcp  filtered pop3s
1433/tcp filtered ms-sql-s
1434/tcp filtered ms-sql-m
3389/tcp filtered ms-term-serv
5800/tcp filtered vnc-http
5801/tcp filtered vnc-http-1
5802/tcp filtered vnc-http-2
5803/tcp filtered vnc-http-3
5900/tcp filtered vnc
5901/tcp filtered vnc-1
5902/tcp filtered vnc-2
5903/tcp filtered vnc-3
6000/tcp open     X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Aggressive OS guesses: Netgear DG834 or DG834G (wireless) DSL Router (96%), Linux 2.4.20
- 2.4.32, Linux-based embedded device (Linksys WRT54GL WAP, Buffalo AirStation WLA-G54
WAP, Maxtor Shared Storage Drive, or Asus Wireless Storage Router) (94%), Linux 2.4.33
(94%), D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17 (94%), Siemens Gigaset
SE515dsl wireless broadband router (93%), Linux 2.4.22 (Fedora Core 1, x86) (92%), Linksys
WRT54GS WAP (Linux kernel) (92%), Linux 2.4.18-10 (Red Hat 7.3) (92%), Aladdin eSafe
security gateway (runs Linux 2.4.21) (91%), Occam ONT ON2342 Voice/Video over IP box
(91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 28.936 seconds
```

```
nmap -sS 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:38 BRST
Interesting ports on 10.0.2.3:
Not shown: 1669 closed ports
PORT     STATE    SERVICE
22/tcp   open     ssh
23/tcp   filtered telnet
80/tcp   open     http
110/tcp  filtered pop3
111/tcp  open     rpcbind
135/tcp  filtered msrpc
137/tcp  filtered netbios-ns
139/tcp  filtered netbios-ssn
143/tcp  filtered imap
161/tcp  filtered snmp
162/tcp  filtered snmptrap
443/tcp  open     https
445/tcp  filtered microsoft-ds
705/tcp  filtered unknown
993/tcp  filtered imaps
995/tcp  filtered pop3s
1433/tcp filtered ms-sql-s
1434/tcp filtered ms-sql-m
3389/tcp filtered ms-term-serv
5800/tcp filtered vnc-http
5801/tcp filtered vnc-http-1
5802/tcp filtered vnc-http-2
5803/tcp filtered vnc-http-3
5900/tcp filtered vnc
5901/tcp filtered vnc-1
5902/tcp filtered vnc-2
5903/tcp filtered vnc-3
6000/tcp open     X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 349.359 seconds

nmap -sS -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:44 BRST
Insufficient responses for TCP sequencing (1), OS detection may be less accurate
Insufficient responses for TCP sequencing (1), OS detection may be less accurate
Insufficient responses for TCP sequencing (0), OS detection may be less accurate
Interesting ports on 10.0.2.3:
Not shown: 1669 closed ports
PORT     STATE    SERVICE
22/tcp   open     ssh
23/tcp   filtered telnet
80/tcp   open     http
110/tcp  filtered pop3
```

```
111/tcp  open     rpcbind
135/tcp  filtered msrpc
137/tcp  filtered netbios-ns
139/tcp  filtered netbios-ssn
143/tcp  filtered imap
161/tcp  filtered snmp
162/tcp  filtered snmptrap
443/tcp  open     https
445/tcp  filtered microsoft-ds
705/tcp  filtered unknown
993/tcp  filtered imaps
995/tcp  filtered pop3s
1433/tcp filtered ms-sql-s
1434/tcp filtered ms-sql-m
3389/tcp filtered ms-term-serv
5800/tcp filtered vnc-http
5801/tcp filtered vnc-http-1
5802/tcp filtered vnc-http-2
5803/tcp filtered vnc-http-3
5900/tcp filtered vnc
5901/tcp filtered vnc-1
5902/tcp filtered vnc-2
5903/tcp filtered vnc-3
6000/tcp open     X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: general purposelbroadband router
Running: Linux 2.4.Xl2.5.Xl2.6.X, D-Link embedded
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 105.501 seconds

nmap -sA 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2008-01-02 12:33 BRST
Interesting ports on 10.0.2.3:
Not shown: 1692 UNfiltered ports
PORT     STATE    SERVICE
161/tcp  filtered snmp
162/tcp  filtered snmptrap
705/tcp  filtered unknown
895/tcp  filtered unknown
6112/tcp filtered dtspc
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 3173.187 seconds

nmap -sA -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2008-01-02 13:34 BRST
Warning:  OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1
open and 1 closed TCP port
```

```
Interesting ports on 10.0.2.3:
Not shown: 1693 UNfiltered ports
PORT     STATE    SERVICE
161/tcp  filtered snmp
162/tcp  filtered snmptrap
705/tcp  filtered unknown
6112/tcp filtered dtspc
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: WAP|printer|broadband router
Running: D-Link Linux 2.4.X, Inventel embedded, Lexmark embedded, Netgear embedded, Siemens linux
OS details: D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17, Inventel Livebox wireless broadband router, Lexmark T632 Network Laser Printer, Netgear DG834 or DG834G (wireless) DSL Router, Siemens Gigaset SE515dsl wireless broadband router
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 3148.779 seconds

nmap -sN 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:58 BRST
Interesting ports on 10.0.2.3:
Not shown: 1689 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
161/tcp  open|filtered snmp
162/tcp  open|filtered snmptrap
443/tcp  open|filtered https
705/tcp  open|filtered unknown
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 75.319 seconds

nmap -sN -O 10.0.2.3
Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-28 14:59 BRST
Warning:  OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
Warning:  OS detection will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
Interesting ports on 10.0.2.3:
Not shown: 1689 closed ports
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
161/tcp  open|filtered snmp
162/tcp  open|filtered snmptrap
443/tcp  open|filtered https
```

705/tcp open|filtered unknown
6000/tcp open|filtered X11
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 80.650 seconds

nmap -sF 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 15:00 BRST
All 1697 scanned ports on 10.0.2.3 are open|filtered
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 36.544 seconds

nmap -sF -O 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 15:01 BRST
Warning: OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
All 1697 scanned ports on 10.0.2.3 are open|filtered
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: WAP|printer|broadband router
Running: D-Link Linux 2.4.X, Inventel embedded, Lexmark embedded, Netgear embedded, Siemens linux
OS details: D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17, Inventel Livebox wireless broadband router, Lexmark T632 Network Laser Printer, Netgear DG834 or DG834G (wireless) DSL Router, Siemens Gigaset SE515dsl wireless broadband router
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 37.250 seconds

nmap -sX 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 15:02 BRST
All 1697 scanned ports on 10.0.2.3 are open|filtered
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Nmap finished: 1 IP address (1 host up) scanned in 36.532 seconds

nmap -sX -O 10.0.2.3
Starting Nmap 4.20 (http://insecure.org) at 2007-12-28 15:02 BRST
Warning: OS detection for 10.0.2.3 will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
All 1697 scanned ports on 10.0.2.3 are open|filtered
MAC Address: 00:A0:C9:E7:9E:C6 (Intel - Hf1-06)
Device type: WAP|printer|broadband router
Running: D-Link Linux 2.4.X, Inventel embedded, Lexmark embedded, Netgear embedded, Siemens linux
OS details: D-Link DSL-G604T ADSL router WAP, runs Linux 2.4.17, Inventel Livebox wireless broadband router, Lexmark T632 Network Laser Printer, Netgear DG834 or DG834G (wireless) DSL Router, Siemens Gigaset SE515dsl wireless broadband router
Network Distance: 1 hop

```
OS detection performed. Please report any incorrect results at http://insecure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 37.271 seconds
```

A Tabela B.2 lista o log gerado pelo tcpdump no servidor Web durante uma varedura do Nikto com IPS ativo.

Tabela B.2: *Tcpdump* no Servidor *Web* sob varredura do *Nikto* com IPS

```
11:43:54.668978 10.0.0.132.45191 > 10.0.2.3.http: S 463087389:463087389(0) win 5840 <mss
1460,sackOK,timestamp 292622 0,nop,wscale 5> (DF)
11:43:54.669054 10.0.2.3.http > 10.0.0.132.45191: S 1075354793:1075354793(0) ack 463087390
win 5792 <mss 1460,sackOK,timestamp 314567 292622,nop,wscale 0> (DF)
11:43:54.669378 10.0.0.132.45191 > 10.0.2.3.http: . ack 1 win 183 <nop,nop,timestamp 292625
314567> (DF)
11:44:04.670761 10.0.0.132.45191 > 10.0.2.3.http: F 100:100(0) ack 1 win 183
<nop,nop,timestamp 295125 314567> (DF)
11:44:04.670956 10.0.2.3.http > 10.0.0.132.45191: . ack 1 win 5792 <nop,nop,timestamp 315568
292625,nop,nop,sack sack 1 {100:101} > (DF)
11:44:04.671993 10.0.0.132.45192 > 10.0.2.3.http: S 470244694:470244694(0) win 5840 <mss
1460,sackOK,timestamp 295125 0,nop,wscale 5> (DF)
11:44:04.672026 10.0.2.3.http > 10.0.0.132.45192: S 1090045068:1090045068(0) ack 470244695
win 5792 <mss 1460,sackOK,timestamp 315568 295125,nop,wscale 0> (DF)
11:44:04.672265 10.0.0.132.45192 > 10.0.2.3.http: . ack 1 win 183 <nop,nop,timestamp 295125
315568> (DF)
11:44:53.746292 arp who-has 10.0.2.3 tell 10.0.0.132
11:44:53.746326 arp reply 10.0.2.3 is-at 0:a0:c9:e7:9e:c6
11:45:48.937165 10.0.0.132.45192 > 10.0.2.3.http: P 1:103(102) ack 1 win 183
<nop,nop,timestamp 321187 315568> (DF)
11:45:48.937329 10.0.2.3.http > 10.0.0.132.45192: . ack 103 win 5792 <nop,nop,timestamp
325994 321187> (DF)
11:45:48.938068 10.0.2.3.http > 10.0.0.132.45192: P 1:320(319) ack 103 win 5792
<nop,nop,timestamp 325994 321187> (DF)
11:45:48.938193 10.0.2.3.http > 10.0.0.132.45192: F 320:320(0) ack 103 win 5792
<nop,nop,timestamp 325994 321187> (DF)
11:45:48.938419 10.0.0.132.45192 > 10.0.2.3.http: . ack 320 win 216 <nop,nop,timestamp 321187
325994> (DF)
11:45:48.939736 10.0.0.132.45192 > 10.0.2.3.http: R 103:103(0) ack 321 win 216
<nop,nop,timestamp 321188 325994> (DF)
11:45:53.935931 arp who-has 10.0.2.3 tell 10.0.0.132
11:45:53.935962 arp reply 10.0.2.3 is-at 0:a0:c9:e7:9e:c6
```

A Tabela B.3 lista os resultados obtidos pelo Nessus, com e sem o IPS,

num servidor com RedHat 7.3 e Apache 1.3.23.

Tabela B.3: Relatório de *scan* do *Nessus* com e sem IPS

Nessus sem IPS

Nessus Scan Report

SUMMARY

- Number of hosts which were alive during the test : 1
- Number of security holes found : 13
- Number of security warnings found : 15
- Number of security notes found : 32

TESTED HOSTS

10.0.2.3 (Security holes found)

DETAILS

+ 10.0.2.3 :
. List of open ports :
  o ssh (22/tcp) (Security warnings found)
  o http (80/tcp) (Security hole found)
  o sunrpc (111/tcp) (Security notes found)
  o https (443/tcp) (Security hole found)
  o filenet-tms (32768/tcp) (Security notes found)
  o x11 (6000/tcp) (Security notes found)
  o filenet-tms (32768/udp) (Security notes found)
  o sunrpc (111/udp) (Security notes found)
  o general/tcp (Security notes found)
  o filenet-rpc (32769/tcp) (Security notes found)
  o general/icmp (Security notes found)
  o general/udp (Security notes found)

. Warning found on port ssh (22/tcp)

Synopsis :

It is possible to enumerate valid users on the remote host.

Description :

The remote host seem to be running an SSH server which can allow an attacker to determine the existence of a given login by comparing the time the remote sshd daemon takes to refuse a bad password for a non-existent login compared to the time it takes to refuse a bad password for a valid login.

An attacker may use this flaw to set up a brute force attack against the remote host.

Solution :

Disable PAM support if you do not use it, upgrade to the newest version of OpenSSH

Risk factor :

Medium / CVSS Base Score : 5.0
(CVSS2#AV:N/AC:L/Au:N/C:P/I:N/A:N)
CVE : CVE-2003-0190
BID : 7342, 7467, 7482, 11781
Other references : OSVDB:2109, OSVDB:2140

. Information found on port ssh (22/tcp)

An ssh server is running on this port

. Information found on port ssh (22/tcp)

Synopsis :

An SSH server is listening on this port.

Description :

It is possible to obtain information about the remote SSH server by sending an empty authentication request.

Risk factor :

None

Plugin output :

SSH version : SSH-1.99-OpenSSH_3.1p1
SSH supported authentication : publickey,password,keyboard-interactive

. Information found on port ssh (22/tcp)

Synopsis :

An SSH server is running on the remote host.

Description :

This plugin determines the versions of the SSH protocol supported by
the remote SSH daemon.

Risk factor :

None

Plugin output :

The remote SSH daemon supports the following versions of the
SSH protocol :

. 1.33
. 1.5
. 1.99
. 2.0

SSHv1 host key fingerprint : e3:b8:1e:1e:08:b9:e7:04:02:62:2e:dc:16:d8:8f:7f
SSHv2 host key fingerprint : e3:e7:4c:c3:17:c2:0a:27:4a:74:86:f5:c0:1c:a1:72

. Information found on port ssh (22/tcp)

Synopsis :

The remote service offers an insecure cryptographic protocol

Description :

The remote SSH daemon supports connections made
using the version 1.33 and/or 1.5 of the SSH protocol.

These protocols are not completely cryptographically
safe so they should not be used.

Solution :

Disable compatibility with version 1 of the protocol.

Risk factor :

Low / CVSS Base Score : 3
(AV:R/AC:H/Au:NR/C:P/A:N/I:N/B:C)
CVE : CVE-2001-0361
BID : 2344
Other references : OSVDB:2116

. Vulnerability found on port http (80/tcp) :

Synopsis :

Arbitrary code can be executed on the remote host

Description :

The remote host is using a version of mod_ssl which is older than
2.8.18.

This version is vulnerable to a flaw which may allow an attacker to
disable the remote web site remotely, or to execute arbitrary code on
the remote host.

Note that several Linux distributions patched the old version of this
module. Therefore, this alert might be a false-positive. Please
check with your vendor to determine if you really are vulnerable to
this flaw.

Solution :

Upgrade to version 2.8.18 (Apache 1.3) or to Apache 2.0.50.

Risk factor :

High / CVSS Base Score : 7.5
(CVSS2#AV:N/AC:L/Au:N/C:P/I:P/A:P)
CVE : CVE-2004-0488

BID : 10355
Other references : OSVDB:6472

. Vulnerability found on port http (80/tcp) :

The remote host is using a version vulnerable of mod_ssl which is
older than 2.8.19. There is a format string condition in the
log functions of the remote module which may allow an attacker to
execute arbitrary code on the remote host.

*** Some vendors patched older versions of mod_ssl, so this
*** might be a false positive. Check with your vendor to determine
*** if you have a version of mod_ssl that is patched for this
*** vulnerability

Solution : Upgrade to version 2.8.19 or newer
Risk factor : High
CVE : CVE-2004-0700
BID : 10736

. Vulnerability found on port http (80/tcp) :

The remote host appears to be running a version of Apache which is older
than 1.3.29

There are several flaws in this version, which may allow an attacker to
possibly execute arbitrary code through mod_alias and mod_rewrite.

You should upgrade to 1.3.29 or newer.

*** Note that Nessus solely relied on the version number
*** of the remote server to issue this warning. This might
*** be a false positive

Solution : Upgrade to version 1.3.29
See also : http://www.apache.org/dist/httpd/Announcement.html
Risk factor : High
CVE : CVE-2003-0542
BID : 8911
Other references : OSVDB:2733, OSVDB:7611

. Vulnerability found on port http (80/tcp) :

The remote host is running a version of PHP which is older than 5.0.3 or
4.3.10.

The remote version of this software is vulnerable to various security
issues which may, under certain circumstances, to execute arbitrary code
on the remote host, provided that we can pass arbitrary data to some
functions, or to bypass safe_mode.

See also : http://www.php.net/ChangeLog-5.php#5.0.3
Solution : Upgrade to PHP 5.0.3 or 4.3.10
Risk factor : High
CVE : CVE-2004-1018, CVE-2004-1019, CVE-2004-1020, CVE-2004-1063,
CVE-2004-1064, CVE-2004-1065
BID : 11964, 11981, 11992, 12045
Other references : OSVDB:12410

. Vulnerability found on port http (80/tcp) :

The remote host is running a version of PHP which is older than 5.0.2 or
4.39.

The remote version of this software is vulnerable to a memory disclosure
vulnerability in PHP_Variables. An attacker may exploit this flaw to
remotely read portions of the memory of the httpd process on the remote
host.

See also : http://www.php.net/ChangeLog-5.php#5.0.2
Solution : Upgrade to PHP 5.0.2 or 4.3.9
Risk factor : High
BID : 11334

. Vulnerability found on port http (80/tcp) :

The remote host appears to be vulnerable to the Apache Web Server Chunk
Handling Vulnerability.

An attacker may exploit this flaw to execute arbitrary code on the remote
host
with the privileges of the httpd process.
Solution : Upgrade to version 1.3.26 or 2.0.39 or newer
See also : http://httpd.apache.org/info/security_bulletin_20020617.txt
http://httpd.apache.org/info/security_bulletin_20020620.txt
Risk factor : High
CVE : CVE-2002-0392
BID : 5033

Other references : IAVA:2002-a-0003, OSVDB:838

. Warning found on port http (80/tcp)

The remote host appears to be running a version of
Apache which is older than 1.3.27

There are several flaws in this version, you should
upgrade to 1.3.27 or newer.

*** Note that Nessus solely relied on the version number
*** of the remote server to issue this warning. This might
*** be a false positive

Solution : Upgrade to version 1.3.27
See also : http://www.apache.org/dist/httpd/Announcement.html
Risk factor : Medium
CVE : CVE-2002-0839, CVE-2002-0840, CVE-2002-0843
BID : 5847, 5884, 5887, 5995, 5996
Other references : OSVDB:862, OSVDB:4552

. Warning found on port http (80/tcp)

It seems that your web server tries to hide its version
or name, which is a good thing.
However, using a special crafted request, Nessus was able
to determine that is is running :
Apache/1.3.23 (Unix) (Red-Hat/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b
DAV/1.0.3 PHP/4.1.2 mod_perl/1.26

Risk factor : None
Solution : Fix your configuration.

. Warning found on port http (80/tcp)

The remote host is using a version of OpenSSL which is
older than 0.9.6j or 0.9.7b

This version is vulnerable to a timing based attack which may
allow an attacker to guess the content of fixed data blocks and
may eventually be able to guess the value of the private RSA key
of the server.

An attacker may use this implementation flaw to sniff the
data going to this host and decrypt some parts of it, as well
as impersonate your server and perform man in the middle attacks.

*** Nessus solely relied on the banner of the remote host
*** to issue this warning

See also : http://www.openssl.org/news/secadv_20030219.txt
http://lasecwww.epfl.ch/memo_ssl.shtml
http://eprint.iacr.org/2003/052/

Solution : Upgrade to version 0.9.6j (0.9.7b) or newer
Risk factor : Medium
CVE : CVE-2003-0078, CVE-2003-0131, CVE-2003-0147
BID : 6884, 7148
Other references : OSVDB:3945, OSVDB:3946, RHSA:RHSA-2003:101-01,
SuSE:SUSE-SA:2003:024

. Warning found on port http (80/tcp)

The remote host is running a version of PHP which is older than 5.0.3 or
4.3.11

The remote version of this software is vulnerable to a set of
vulnerabilities in the EXIF module which have been fixed by the PHP Group.

See also : http://www.php.net/ChangeLog-5.php#5.0.4
http://www.php.net/ChangeLog-4.php#4.3.11

Solution : Upgrade to PHP 5.0.3 or 4.3.11
Risk factor : Medium
BID : 13143, 13163, 13164

. Warning found on port http (80/tcp)

The remote host is running a version of PHP <= 4.2.2.

The mail() function does not properly sanitize user input.
This allows users to forge email to make it look like it is
coming from a different source other than the server.

Users can exploit this even if SAFE_MODE is enabled.

Solution : Contact your vendor for the latest PHP release.
Risk factor : Medium

CVE : CVE-2002-0985, CVE-2002-0986
BID : 5562

. Warning found on port http (80/tcp)

Synopsis :

The remote service is prone to a denial of service attack.

Description :

According to its banner, the remote host is using a version of OpenSSL
which is older than 0.9.6m / 0.9.7d. There are several bugs in such
versions that may allow an attacker to cause a denial of service
against the remote host.

See also :

http://www.openssl.org/news/secadv_20040317.txt
http://archives.neohapsis.com/archives/bugtraq/2004-03/0165.html

Solution :

Upgrade to version 0.9.6m / 0.9.7d or newer.

Risk factor :

Medium / CVSS Base Score : 5.0
(CVSS2#AV:N/AC:L/Au:N/C:N/I:N/A:P)
CVE : CVE-2004-0079, CVE-2004-0081, CVE-2004-0112
BID : 9899
Other references : IAVA:2004-B-0006, OSVDB:4316, OSVDB:4317, OSVDB:4318

. Information found on port http (80/tcp)

A web server is running on this port

. Information found on port http (80/tcp)

Synopsis :

The remote server is running with WebDAV enabled.

Description :

WebDAV is an industry standard extension to the HTTP specification.
It adds a capability for authorized users to remotely add and manage
the content of a web server.

If you do not use this extension, you should disable it.

Solution :

http://support.microsoft.com/default.aspx?kbid=241520

Risk factor :

None

. Information found on port http (80/tcp)

Synopsis :

The remote web server is prone to denial of service attacks.

Description :

According to its banner, the version of PHP installed on the remote
host is vulnerable to a denial of service attack due to its failure to
properly validate file data in the routines 'php_handle_iff' and
'php_handle_jpeg', which are called by the PHP function
'getimagesize'. Using a specially crafted image file, an attacker can
trigger an infinite loop when 'getimagesize' is called, perhaps even
remotely in the case image uploads are allowed.

See also :

http://www.idefense.com/intelligence/vulnerabilities/display.php?id=222
http://www.securityfocus.com/archive/1/394797
http://www.php.net/release_4_3_11.php

Solution :

Upgrade to PHP 4.3.11 / 5.0.4 or later.

Risk factor :

Low / CVSS Base Score : 3
(AV:R/AC:H/Au:NR/C:N/A:C/I:N/B:N)
CVE : CVE-2005-0524, CVE-2005-0525
BID : 12962, 12963

. Information found on port http (80/tcp)

Synopsis :

Some information about the remote HTTP configuration can be extracted.

Description :

This test gives some information about the remote HTTP protocol - the version used, whether HTTP Keep-Alive and HTTP pipelining are enabled, etc...

This test is informational only and does not denote any security problem

Solution :

None.

Risk factor :

None

Plugin output :

Protocol version : HTTP/1.1
SSL : no
Pipelining : no
Keep-Alive : no
Options allowed : GET, HEAD, OPTIONS, TRACE
Headers :

Date: Fri, 28 Dec 2007 16:53:42 GMT
Server: Apache/1.3.23 (Unix) (Red-Hat/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b DAV/1.0.3 PHP/4.1.2 mod_perl/1.26
Last-Modified: Fri, 21 Dec 2007 15:53:24 GMT
ETag: "b3205-214-476be174"
Accept-Ranges: bytes
Content-Length: 532
Connection: close
Content-Type: text/html

. Information found on port sunrpc (111/tcp)

Synopsis :

An ONC RPC portmapper is running on the remote host.

Description :

The RPC portmapper is running on this port.

The portmapper allows to get the port number of each RPC service
running on the remote host either by sending multiple lookup
requests or by sending a DUMP request.

Risk factor :

None

. Information found on port sunrpc (111/tcp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on TCP port 111 :

- program: 100000 (portmapper), version: 2

. Vulnerability found on port https (443/tcp) :

Synopsis :

Arbitrary code can be executed on the remote host

Description :

The remote host is using a version of mod_ssl which is older than
2.8.18.

This version is vulnerable to a flaw which may allow an attacker to
disable the remote web site remotely, or to execute arbitrary code on
the remote host.

Note that several Linux distributions patched the old version of this
module. Therefore, this alert might be a false-positive. Please
check with your vendor to determine if you really are vulnerable to
this flaw.

Solution :

Upgrade to version 2.8.18 (Apache 1.3) or to Apache 2.0.50.

Risk factor :

High / CVSS Base Score : 7.5
(CVSS2#AV:N/AC:L/Au:N/C:P/I:P/A:P)
CVE : CVE-2004-0488
BID : 10355
Other references : OSVDB:6472

. Vulnerability found on port https (443/tcp) :

The remote host is using a version vulnerable of mod_ssl which is
older than 2.8.19. There is a format string condition in the
log functions of the remote module which may allow an attacker to
execute arbitrary code on the remote host.

*** Some vendors patched older versions of mod_ssl, so this
*** might be a false positive. Check with your vendor to determine
*** if you have a version of mod_ssl that is patched for this
*** vulnerability

Solution : Upgrade to version 2.8.19 or newer
Risk factor : High
CVE : CVE-2004-0700
BID : 10736

. Vulnerability found on port https (443/tcp) :

The remote host appears to be running a version of Apache which is older than 1.3.29

There are several flaws in this version, which may allow an attacker to possibly execute arbitrary code through mod_alias and mod_rewrite.

You should upgrade to 1.3.29 or newer.

*** Note that Nessus solely relied on the version number
*** of the remote server to issue this warning. This might
*** be a false positive

Solution : Upgrade to version 1.3.29
See also : http://www.apache.org/dist/httpd/Announcement.html
Risk factor : High
CVE : CVE-2003-0542
BID : 8911
Other references : OSVDB:2733, OSVDB:7611

. Vulnerability found on port https (443/tcp) :

The remote host is running a version of PHP which is older than 5.0.3 or 4.3.10.

The remote version of this software is vulnerable to various security issues which may, under certain circumstances, to execute arbitrary code on the remote host, provided that we can pass arbitrary data to some functions, or to bypass safe_mode.

See also : http://www.php.net/ChangeLog-5.php#5.0.3
Solution : Upgrade to PHP 5.0.3 or 4.3.10
Risk factor : High
CVE : CVE-2004-1018, CVE-2004-1019, CVE-2004-1020, CVE-2004-1063, CVE-2004-1064, CVE-2004-1065
BID : 11964, 11981, 11992, 12045
Other references : OSVDB:12410

. Vulnerability found on port https (443/tcp) :

The remote host is running a version of PHP which is older than 5.0.2 or 4.39.

The remote version of this software is vulnerable to a memory disclosure
vulnerability in PHP_Variables. An attacker may exploit this flaw to
remotely read portions of the memory of the httpd process on the remote
host.

See also : http://www.php.net/ChangeLog-5.php#5.0.2
Solution : Upgrade to PHP 5.0.2 or 4.3.9
Risk factor : High
BID : 11334

. Vulnerability found on port https (443/tcp) :

The remote host appears to be vulnerable to the Apache Web Server Chunk
Handling Vulnerability.

An attacker may exploit this flaw to execute arbitrary code on the remote
host
with the privileges of the httpd process.
Solution : Upgrade to version 1.3.26 or 2.0.39 or newer
See also : http://httpd.apache.org/info/security_bulletin_20020617.txt
http://httpd.apache.org/info/security_bulletin_20020620.txt
Risk factor : High
CVE : CVE-2002-0392
BID : 5033
Other references : IAVA:2002-a-0003, OSVDB:838

. Vulnerability found on port https (443/tcp) :

Synopsis :

The remote service uses a library which is affected by a buffer
overflow vulnerability.

Description :

The remote service seems to be using a version of OpenSSL which is
older than 0.9.6e or 0.9.7-beta3.

This version is vulnerable to a buffer overflow that may allow an
attacker to execute arbitrary commands on the remote host with the
privileges of the application itself.

Solution :

Upgrade to OpenSSL version 0.9.6e (0.9.7beta3) or newer.

If the remote service is Compaq Insight Manager, please visit
http://h18023.www1.hp.com/support/files/server/us/download/15803.html

Risk factor :

Critical / CVSS Base Score : 10.0
(CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C)
CVE : CVE-2002-0656, CVE-2002-0655, CVE-2002-0657, CVE-2002-0659,
CVE-2001-1141
BID : 3004, 5361, 5362, 5363, 5364, 5366
Other references : IAVA:2002-a-0004, OSVDB:853, OSVDB:857, OSVDB:3940,
OSVDB:3941, OSVDB:3942, OSVDB:3943, SuSE:SUSE-SA:2002:033

. Warning found on port https (443/tcp)

The remote host appears to be running a version of
Apache which is older than 1.3.27

There are several flaws in this version, you should
upgrade to 1.3.27 or newer.

*** Note that Nessus solely relied on the version number
*** of the remote server to issue this warning. This might
*** be a false positive

Solution : Upgrade to version 1.3.27
See also : http://www.apache.org/dist/httpd/Announcement.html
Risk factor : Medium
CVE : CVE-2002-0839, CVE-2002-0840, CVE-2002-0843
BID : 5847, 5884, 5887, 5995, 5996
Other references : OSVDB:862, OSVDB:4552

. Warning found on port https (443/tcp)

It seems that your web server tries to hide its version
or name, which is a good thing.
However, using a special crafted request, Nessus was able
to determine that is is running :
Apache/1.3.23 (Unix) (Red-Hat/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b
DAV/1.0.3 PHP/4.1.2 mod_perl/1.26

Risk factor : None
Solution : Fix your configuration.

. Warning found on port https (443/tcp)

The remote host is using a version of OpenSSL which is
older than 0.9.6j or 0.9.7b

This version is vulnerable to a timing based attack which may
allow an attacker to guess the content of fixed data blocks and
may eventually be able to guess the value of the private RSA key
of the server.

An attacker may use this implementation flaw to sniff the
data going to this host and decrypt some parts of it, as well
as impersonate your server and perform man in the middle attacks.

*** Nessus solely relied on the banner of the remote host
*** to issue this warning

See also : http://www.openssl.org/news/secadv_20030219.txt
http://lasecwww.epfl.ch/memo_ssl.shtml
http://eprint.iacr.org/2003/052/

Solution : Upgrade to version 0.9.6j (0.9.7b) or newer
Risk factor : Medium
CVE : CVE-2003-0078, CVE-2003-0131, CVE-2003-0147
BID : 6884, 7148
Other references : OSVDB:3945, OSVDB:3946, RHSA:RHSA-2003:101-01,
SuSE:SUSE-SA:2003:024

. Warning found on port https (443/tcp)

The remote host is running a version of PHP which is older than 5.0.3 or
4.3.11

The remote version of this software is vulnerable to a set of
vulnerabilities in the EXIF module which have been fixed by the PHP Group.

See also : http://www.php.net/ChangeLog-5.php#5.0.4
http://www.php.net/ChangeLog-4.php#4.3.11

Solution : Upgrade to PHP 5.0.3 or 4.3.11
Risk factor : Medium
BID : 13143, 13163, 13164

. Warning found on port https (443/tcp)

The remote host is running a version of PHP <= 4.2.2.

The mail() function does not properly sanitize user input.
This allows users to forge email to make it look like it is
coming from a different source other than the server.

Users can exploit this even if SAFE_MODE is enabled.

Solution : Contact your vendor for the latest PHP release.
Risk factor : Medium
CVE : CVE-2002-0985, CVE-2002-0986
BID : 5562

. Warning found on port https (443/tcp)

Synopsis :

The remote service is prone to a denial of service attack.

Description :

According to its banner, the remote host is using a version of OpenSSL
which is older than 0.9.6m / 0.9.7d. There are several bugs in such
versions that may allow an attacker to cause a denial of service
against the remote host.

See also :

http://www.openssl.org/news/secadv_20040317.txt
http://archives.neohapsis.com/archives/bugtraq/2004-03/0165.html

Solution :

Upgrade to version 0.9.6m / 0.9.7d or newer.

Risk factor :

Medium / CVSS Base Score : 5.0
(CVSS2#AV:N/AC:L/Au:N/C:N/I:N/A:P)
CVE : CVE-2004-0079, CVE-2004-0081, CVE-2004-0112
BID : 9899
Other references : IAVA:2004-B-0006, OSVDB:4316, OSVDB:4317, OSVDB:4318

. Warning found on port https (443/tcp)

Synopsis :

The remote service encrypts traffic using a protocol with known weaknesses.

Description :

The remote service accepts connections encrypted using SSL 2.0, which reportedly suffers from several cryptographic flaws and has been deprecated for several years. An attacker may be able to exploit these issues to conduct man-in-the-middle attacks or decrypt communications between the affected service and clients.

See also :

http://www.schneier.com/paper-ssl.pdf

Solution :

Consult the application's documentation to disable SSL 2.0 and use SSL 3.0 or TLS 1.0 instead.

Risk factor :

Medium / CVSS Base Score : 5.0
(CVSS2#AV:N/AC:L/Au:N/C:P/I:N/A:N)

. Warning found on port https (443/tcp)

Synopsis :

The remote web server is affected by a denial of service flaw.

Description :

The installed version of Apache with mod_ssl on the remote host appears susceptible to a remote denial of service flaw under certain atypical configurations. A remote attacker may be able to exploit this issue to crash individual child processes or even the entire server, thereby denying service to legitimate users.

See also :

http://issues.apache.org/bugzilla/show_bug.cgi?id=37791

Solution :

Update the Apache configuration to use "SSLRequire' whenever
'SSLCipherSuite' is used.

Risk factor :

Medium / CVSS Base Score : 5.4
(CVSS2#AV:N/AC:H/Au:N/C:N/I:N/A:C)
CVE : CVE-2005-3357
BID : 16152

. Information found on port https (443/tcp)

A SSLv2 server answered on this port

. Information found on port https (443/tcp)

A web server is running on this port through SSL

. Information found on port https (443/tcp)

Here is the SSLv2 server certificate:
Certificate:
Data:
Version: 3 (0x2)
Serial Number: 0 (0x0)
Signature Algorithm: md5WithRSAEncryption
Issuer: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
OU=SomeOrganizationalUnit,
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
Validity
Not Before: Dec 21 13:33:09 2007 GMT
Not After : Dec 20 13:33:09 2008 GMT
Subject: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
OU=SomeOrganizationalUnit,
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
00:a0:5e:67:7b:f4:df:50:c7:2f:ca:2a:16:2f:6d:
82:1f:94:e7:0d:b7:05:b9:bb:5a:aa:19:a4:1e:e1:
d9:15:36:4c:44:9a:a8:28:71:5b:35:17:a8:af:4e:
1f:ed:01:87:92:d5:88:e3:66:02:9f:64:e6:70:50:

```
                93:fc:65:c7:f0:12:e7:1c:d6:cb:55:b9:bc:c8:82:
                e1:8a:11:9d:e6:62:c6:88:f9:80:60:30:f9:1a:c8:
                86:d5:87:b9:0f:95:b3:5d:7b:a1:65:6c:b9:ea:42:
                42:7a:ec:41:d0:0f:4d:81:0d:1c:9e:34:09:f3:17:
                11:a7:27:55:35:2a:b5:c5:1b
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                F5:B9:F5:CB:CA:FA:A8:85:BA:B4:ED:26:86:CC:02:22:19:0E:B2:13
            X509v3 Authority Key Identifier:

    keyid:F5:B9:F5:CB:CA:FA:A8:85:BA:B4:ED:26:86:CC:02:22:19:0E:B2:13

    DirName:/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizational
Unit/CN=localhost.localdomain/emailAddress=root@localhost.localdomain
            serial:00

        X509v3 Basic Constraints:
            CA:TRUE
    Signature Algorithm: md5WithRSAEncryption
        74:e0:28:71:d2:fc:b9:7c:c1:e8:71:80:14:ec:60:88:37:cc:
        44:32:2b:30:1b:02:f2:06:cb:b6:68:ae:6e:3d:75:27:0a:7c:
        33:c7:bf:0d:fb:b5:85:5d:79:d9:bc:3f:4d:2c:e9:c5:99:03:
        cd:70:10:b5:c9:b9:a8:47:43:03:8a:4b:7c:64:e8:4c:45:8b:
        b4:43:ba:61:cf:9e:dc:22:da:2d:cd:2b:b3:ae:9f:af:30:f5:
        18:c4:29:64:f0:83:e5:8d:9a:fc:43:dd:e3:f8:24:5b:49:04:
        ed:85:d9:28:46:a0:5e:16:43:9a:cb:4c:92:5b:6a:61:ab:16:
        3b:28
Here is the list of available SSLv2 ciphers:
RC4-MD5
EXP-RC4-MD5
RC2-CBC-MD5
EXP-RC2-CBC-MD5
DES-CBC-MD5
DES-CBC3-MD5
The SSLv2 server offers 4 strong ciphers, but also
0 medium strength and 2 weak "export class" ciphers.
The weak/medium ciphers may be chosen by an export-grade
or badly configured client software. They only offer a
limited protection against a brute force attack

Solution: disable those ciphers and upgrade your client
software if necessary.
See http://support.microsoft.com/default.aspx?scid=kb;en-us;216482
or http://httpd.apache.org/docs-2.0/mod/mod_ssl.html#sslciphersuite
This SSLv2 server also accepts SSLv3 connections.
This SSLv2 server also accepts TLSv1 connections.
```

. Information found on port https (443/tcp)

Synopsis :

The remote server is running with WebDAV enabled.

Description :

WebDAV is an industry standard extension to the HTTP specification.
It adds a capability for authorized users to remotely add and manage
the content of a web server.

If you do not use this extension, you should disable it.

Solution :

http://support.microsoft.com/default.aspx?kbid=241520

Risk factor :

None

. Information found on port https (443/tcp)

Synopsis :

The remote web server is prone to denial of service attacks.

Description :

According to its banner, the version of PHP installed on the remote
host is vulnerable to a denial of service attack due to its failure to
properly validate file data in the routines 'php_handle_iff' and
'php_handle_jpeg', which are called by the PHP function
'getimagesize'. Using a specially crafted image file, an attacker can
trigger an infinite loop when 'getimagesize' is called, perhaps even
remotely in the case image uploads are allowed.

See also :

http://www.idefense.com/intelligence/vulnerabilities/display.php?id=222
http://www.securityfocus.com/archive/1/394797
http://www.php.net/release_4_3_11.php

Solution :

Upgrade to PHP 4.3.11 / 5.0.4 or later.

Risk factor :

Low / CVSS Base Score : 3
(AV:R/AC:H/Au:NR/C:N/A:C/I:N/B:N)
CVE : CVE-2005-0524, CVE-2005-0525
BID : 12962, 12963

. Information found on port https (443/tcp)

Synopsis :

The remote service encrypts communications using SSL.

Description :

This script detects which SSL ciphers are supported by the remote
service for encrypting communications.

See also :

http://www.openssl.org/docs/apps/ciphers.html

Risk factor :

None

Plugin output :

Here is the list of SSL ciphers supported by the remote server :

```
 Low Strength Ciphers (< 56-bit key)
   SSLv2
     EXP-RC2-CBC-MD5          Kx=RSA(512)   Au=RSA     Enc=RC2(40)
 Mac=MD5    export
     EXP-RC4-MD5              Kx=RSA(512)   Au=RSA     Enc=RC4(40)
 Mac=MD5    export
   SSLv3
     EXP-EDH-RSA-DES-CBC-SHA    Kx=DH(512)    Au=RSA     Enc=DES(40)
 Mac=SHA1   export
     EXP-DES-CBC-SHA          Kx=RSA(512)   Au=RSA     Enc=DES(40)
 Mac=SHA1   export
     EXP-RC2-CBC-MD5          Kx=RSA(512)   Au=RSA     Enc=RC2(40)
```

```
Mac=MD5    export
    EXP-RC4-MD5              Kx=RSA(512)   Au=RSA     Enc=RC4(40)
Mac=MD5    export
  TLSv1
    EXP-EDH-RSA-DES-CBC-SHA    Kx=DH(512)    Au=RSA     Enc=DES(40)
Mac=SHA1   export
    EXP-DES-CBC-SHA          Kx=RSA(512)   Au=RSA     Enc=DES(40)
Mac=SHA1   export
    EXP-RC2-CBC-MD5          Kx=RSA(512)   Au=RSA     Enc=RC2(40)
Mac=MD5    export
    EXP-RC4-MD5              Kx=RSA(512)   Au=RSA     Enc=RC4(40)
Mac=MD5    export

 Medium Strength Ciphers (>= 56-bit and < 112-bit key)
  SSLv2
    DES-CBC-MD5              Kx=RSA        Au=RSA     Enc=DES(56)
Mac=MD5
    RC4-64-MD5               Kx=RSA        Au=RSA     Enc=RC4(64)
Mac=MD5
  SSLv3
    EDH-RSA-DES-CBC-SHA        Kx=DH         Au=RSA     Enc=DES(56)
Mac=SHA1
    DES-CBC-SHA              Kx=RSA        Au=RSA     Enc=DES(56)
Mac=SHA1
  TLSv1
    EDH-RSA-DES-CBC-SHA        Kx=DH         Au=RSA     Enc=DES(56)
Mac=SHA1
    EXP1024-DES-CBC-SHA        Kx=RSA(1024)  Au=RSA     Enc=DES(56)
Mac=SHA1   export
    EXP1024-RC2-CBC-MD5        Kx=RSA(1024)  Au=RSA     Enc=RC2(56)
Mac=MD5    export
    EXP1024-RC4-MD5          Kx=RSA(1024)  Au=RSA     Enc=RC4(56)
Mac=MD5    export
    EXP1024-RC4-SHA          Kx=RSA(1024)  Au=RSA     Enc=RC4(56)
Mac=SHA1   export
    DES-CBC-SHA              Kx=RSA        Au=RSA     Enc=DES(56)
Mac=SHA1

 High Strength Ciphers (>= 112-bit key)
  SSLv2
    DES-CBC3-MD5             Kx=RSA        Au=RSA     Enc=3DES(168)
Mac=MD5
    RC2-CBC-MD5              Kx=RSA        Au=RSA     Enc=RC2(128)
Mac=MD5
    RC4-MD5                  Kx=RSA        Au=RSA     Enc=RC4(128)
Mac=MD5
  SSLv3
    EDH-RSA-DES-CBC3-SHA       Kx=DH         Au=RSA     Enc=3DES(168)
Mac=SHA1
```

DES-CBC3-SHA Kx=RSA Au=RSA Enc=3DES(168)
Mac=SHA1
RC4-MD5 Kx=RSA Au=RSA Enc=RC4(128)
Mac=MD5
RC4-SHA Kx=RSA Au=RSA Enc=RC4(128)
Mac=SHA1
TLSv1
EDH-RSA-DES-CBC3-SHA Kx=DH Au=RSA Enc=3DES(168)
Mac=SHA1
DES-CBC3-SHA Kx=RSA Au=RSA Enc=3DES(168)
Mac=SHA1
RC4-MD5 Kx=RSA Au=RSA Enc=RC4(128)
Mac=MD5
RC4-SHA Kx=RSA Au=RSA Enc=RC4(128)
Mac=SHA1

The fields above are :

{OpenSSL ciphername}
Kx={key exchange}
Au={authentication}
Enc={symmetric encryption method}
Mac={message authentication code}
{export flag}

. Information found on port https (443/tcp)

Synopsis :

Some information about the remote HTTP configuration can be extracted.

Description :

This test gives some information about the remote HTTP protocol - the version used, whether HTTP Keep-Alive and HTTP pipelining are enabled, etc...

This test is informational only and does not denote any security problem

Solution :

None.

Risk factor :

None

Plugin output :

Protocol version : HTTP/1.1
SSL : yes
Pipelining : no
Keep-Alive : no
Options allowed : GET, HEAD, OPTIONS, TRACE
Headers :

Date: Fri, 28 Dec 2007 16:53:42 GMT
Server: Apache/1.3.23 (Unix) (Red-Hat/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b
DAV/1.0.3 PHP/4.1.2 mod_perl/1.26
Last-Modified: Fri, 21 Dec 2007 15:53:24 GMT
ETag: "b3205-214-476be174"
Accept-Ranges: bytes
Content-Length: 532
Connection: close
Content-Type: text/html

. Information found on port filenet-tms (32768/tcp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on TCP port 32768 :

- program: 100024 (status), version: 1

. Information found on port x11 (6000/tcp)

Synopsis :

A X11 server is listening on the remote host

Description :

The remote host is running a X11 server. X11 is a client-server protocol
which can be used to display graphical applications running on a given
host on a remote client.

Since the X11 traffic is not ciphered, it is possible for an attacker
to eavesdrop on the connection.

Solution :

Restrict access to this port. If the X11 client/server facility is not
used, disable TCP entirely.

Risk factor :

Low / CVSS Base Score : 2
(AV:R/AC:H/Au:R/C:P/A:N/I:N/B:C)

Plugin output :

X11 Version : 11.0

. Information found on port filenet-tms (32768/udp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on UDP port 32768 :

- program: 100024 (status), version: 1

. Information found on port sunrpc (111/udp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on UDP port 111 :

- program: 100000 (portmapper), version: 2

. Information found on port general/tcp

TCP split NIDS evasion function is enabled. Some tests might
run slowly and you may get some false negative results

. Information found on port general/tcp

HTTP NIDS evasion functions are enabled.
You may get some false negative results

. Information found on port general/tcp

It was possible to log into the remote host using the supplied password
The output of "uname -a" is :
Linux localhost.localdomain 2.4.18-3 #1 Thu Apr 18 07:32:41 EDT 2002 i686
unknown

The remote Linux distribution is not supported, therefore local security
checks have not been enabled

. Information found on port general/tcp

Synopsis :

This plugin enumerates IPv4 interfaces on a remote host.

Description :

By connecting to the remote host with the supplied credentials, this
plugin enumerates network interfaces configured with IPv4 addresses.

Solution :

Disable any unused IPv4 interfaces.

Risk factor :

None

Plugin output:

The following IPv4 addresses are set on the remote host :
- 10.0.2.3 (on interface eth0)
- 127.0.0.1 (on interface lo)

. Information found on port general/tcp

Remote operating system : Linux Kernel 2.4.18-3
Confidence Level : 99
Method : uname

The remote host is running Linux Kernel 2.4.18-3

. Information found on port general/tcp

Synopsis :

The remote service implements TCP timestamps.

Description :

The remote host implements TCP timestamps, as defined by RFC1323.
A side effect of this feature is that the uptime of the remote
host can be sometimes be computed.

See also :

http://www.ietf.org/rfc/rfc1323.txt

Risk factor :

None

. Information found on port general/tcp

Information about this scan :

Nessus version : 2.2.8 (Nessus 2.2.10 is available - consider upgrading)

Plugin feed version : 200711270935
Type of plugin feed : Registered (7 days delay)
Scanner IP : 10.0.0.132
Port scanner(s) : synscan netstat nessus_tcp_scanner
Port range : 1-500
Thorough tests : no
Experimental tests : no
Paranoia level : 1
Report Verbosity : 1
Safe checks : no
Optimize the test : no
Max hosts : 1
Max checks : 8
Scan Start Date : 2007/12/28 13:48
Scan duration : 577 sec

. Information found on port general/tcp

Synopsis :

It was not possible to log into the remote host

Description :

The credentials provided for the scan did not allow us to log into the
remote host.

Risk factor :

None

Plugin output :

The local checks failed because Unsupported Linux distribution

. Information found on port filenet-rpc (32769/tcp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on TCP port 32769 :

- program: 391002 (sgi_fam), version: 2

. Information found on port general/icmp

Synopsis :

It is possible to determine the exact time set on the remote host.

Description :

The remote host answers to an ICMP timestamp request. This allows an attacker to know the date which is set on your machine.

This may help him to defeat all your time based authentication protocols.

Solution :

Filter out the ICMP timestamp requests (13), and the outgoing ICMP timestamp replies (14).

Risk factor :

None

Plugin output :

The difference between the local and remote clocks is -3799 seconds

CVE : CVE-1999-0524

. Information found on port general/udp

For your information, here is the traceroute from 10.0.0.132 to 10.0.2.3 :
10.0.0.132
10.0.2.3

This file was generated by the Nessus Security Scanner

Nessus com IPS

Nessus Scan Report

SUMMARY

- Number of hosts which were alive during the test : 1
- Number of security holes found : 7
- Number of security warnings found : 9
- Number of security notes found : 21

TESTED HOSTS

10.0.2.3 (Security holes found)

DETAILS

+ 10.0.2.3 :
. List of open ports :
  o http (80/tcp) (Security notes found)
  o sunrpc (111/tcp) (Security notes found)
  o https (443/tcp) (Security hole found)
  o ssh (22/tcp)
  o general/tcp (Security notes found)
  o filenet-tms (32768/tcp) (Security notes found)
  o filenet-rpc (32769/tcp) (Security notes found)
  o sunrpc (111/udp) (Security notes found)
  o filenet-tms (32768/udp) (Security notes found)
  o x11 (6000/tcp) (Security notes found)

. Information found on port http (80/tcp)

A web server is running on this port

. Information found on port http (80/tcp)

The remote port seems to either have network connectivity issues
or seems to be protected by an IPS which prevents Nessus from

sending HTTP requests to this port.

As a result, the remote web server will not be tested.

Solution : configure your IPS to allow network scanning from 10.0.0.132
Risk Factor : None

. Information found on port sunrpc (111/tcp)

Synopsis :

An ONC RPC portmapper is running on the remote host.

Description :

The RPC portmapper is running on this port.

The portmapper allows to get the port number of each RPC service
running on the remote host either by sending multiple lookup
requests or by sending a DUMP request.

Risk factor :

None

. Information found on port sunrpc (111/tcp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on TCP port 111 :

- program: 100000 (portmapper), version: 2

. Vulnerability found on port https (443/tcp) :

Synopsis :

Arbitrary code can be executed on the remote host

Description :

The remote host is using a version of mod_ssl which is older than
2.8.18.

This version is vulnerable to a flaw which may allow an attacker to
disable the remote web site remotely, or to execute arbitrary code on
the remote host.

Note that several Linux distributions patched the old version of this
module. Therefore, this alert might be a false-positive. Please
check with your vendor to determine if you really are vulnerable to
this flaw.

Solution :

Upgrade to version 2.8.18 (Apache 1.3) or to Apache 2.0.50.

Risk factor :

High / CVSS Base Score : 7.5
(CVSS2#AV:N/AC:L/Au:N/C:P/I:P/A:P)
CVE : CVE-2004-0488
BID : 10355
Other references : OSVDB:6472

. Vulnerability found on port https (443/tcp) :

The remote host is using a version vulnerable of mod_ssl which is
older than 2.8.19. There is a format string condition in the
log functions of the remote module which may allow an attacker to
execute arbitrary code on the remote host.

*** Some vendors patched older versions of mod_ssl, so this
*** might be a false positive. Check with your vendor to determine

*** if you have a version of mod_ssl that is patched for this
*** vulnerability

Solution : Upgrade to version 2.8.19 or newer
Risk factor : High
CVE : CVE-2004-0700
BID : 10736

. Vulnerability found on port https (443/tcp) :

The remote host appears to be running a version of Apache which is older
than 1.3.29

There are several flaws in this version, which may allow an attacker to
possibly execute arbitrary code through mod_alias and mod_rewrite.

You should upgrade to 1.3.29 or newer.

*** Note that Nessus solely relied on the version number
*** of the remote server to issue this warning. This might
*** be a false positive

Solution : Upgrade to version 1.3.29
See also : http://www.apache.org/dist/httpd/Announcement.html
Risk factor : High
CVE : CVE-2003-0542
BID : 8911
Other references : OSVDB:2733, OSVDB:7611

. Vulnerability found on port https (443/tcp) :

The remote host is running a version of PHP which is older than 5.0.3 or
4.3.10.

The remote version of this software is vulnerable to various security
issues which may, under certain circumstances, to execute arbitrary code
on the remote host, provided that we can pass arbitrary data to some
functions, or to bypass safe_mode.

See also : http://www.php.net/ChangeLog-5.php#5.0.3
Solution : Upgrade to PHP 5.0.3 or 4.3.10
Risk factor : High
CVE : CVE-2004-1018, CVE-2004-1019, CVE-2004-1020, CVE-2004-1063,
CVE-2004-1064, CVE-2004-1065
BID : 11964, 11981, 11992, 12045

Other references : OSVDB:12410

. Vulnerability found on port https (443/tcp) :

The remote host is running a version of PHP which is older than 5.0.2 or 4.39.

The remote version of this software is vulnerable to a memory disclosure vulnerability in PHP_Variables. An attacker may exploit this flaw to remotely read portions of the memory of the httpd process on the remote host.

See also : http://www.php.net/ChangeLog-5.php#5.0.2
Solution : Upgrade to PHP 5.0.2 or 4.3.9
Risk factor : High
BID : 11334

. Vulnerability found on port https (443/tcp) :

The remote host appears to be vulnerable to the Apache Web Server Chunk Handling Vulnerability.

An attacker may exploit this flaw to execute arbitrary code on the remote host
with the privileges of the httpd process.
Solution : Upgrade to version 1.3.26 or 2.0.39 or newer
See also : http://httpd.apache.org/info/security_bulletin_20020617.txt
http://httpd.apache.org/info/security_bulletin_20020620.txt
Risk factor : High
CVE : CVE-2002-0392
BID : 5033
Other references : IAVA:2002-a-0003, OSVDB:838

. Vulnerability found on port https (443/tcp) :

Synopsis :

The remote service uses a library which is affected by a buffer overflow vulnerability.

Description :

The remote service seems to be using a version of OpenSSL which is

older than 0.9.6e or 0.9.7-beta3.

This version is vulnerable to a buffer overflow that may allow an
attacker to execute arbitrary commands on the remote host with the
privileges of the application itself.

Solution :

Upgrade to OpenSSL version 0.9.6e (0.9.7beta3) or newer.

If the remote service is Compaq Insight Manager, please visit
http://h18023.www1.hp.com/support/files/server/us/download/15803.html

Risk factor :

Critical / CVSS Base Score : 10.0
(CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C)
CVE : CVE-2002-0656, CVE-2002-0655, CVE-2002-0657, CVE-2002-0659,
CVE-2001-1141
BID : 3004, 5361, 5362, 5363, 5364, 5366
Other references : IAVA:2002-a-0004, OSVDB:853, OSVDB:857, OSVDB:3940,
OSVDB:3941, OSVDB:3942, OSVDB:3943, SuSE:SUSE-SA:2002:033

. Warning found on port https (443/tcp)

It seems that your web server tries to hide its version
or name, which is a good thing.
However, using a special crafted request, Nessus was able
to determine that is is running :
Apache/1.3.23 (Unix) (Red-Hat/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b
DAV/1.0.3 PHP/4.1.2 mod_perl/1.26

Risk factor : None
Solution : Fix your configuration.

. Warning found on port https (443/tcp)

The remote host appears to be running a version of
Apache which is older than 1.3.27

There are several flaws in this version, you should
upgrade to 1.3.27 or newer.

*** Note that Nessus solely relied on the version number
*** of the remote server to issue this warning. This might

*** be a false positive

Solution : Upgrade to version 1.3.27
See also : http://www.apache.org/dist/httpd/Announcement.html
Risk factor : Medium
CVE : CVE-2002-0839, CVE-2002-0840, CVE-2002-0843
BID : 5847, 5884, 5887, 5995, 5996
Other references : OSVDB:862, OSVDB:4552

. Warning found on port https (443/tcp)

The remote host is using a version of OpenSSL which is
older than 0.9.6j or 0.9.7b

This version is vulnerable to a timing based attack which may
allow an attacker to guess the content of fixed data blocks and
may eventually be able to guess the value of the private RSA key
of the server.

An attacker may use this implementation flaw to sniff the
data going to this host and decrypt some parts of it, as well
as impersonate your server and perform man in the middle attacks.

*** Nessus solely relied on the banner of the remote host
*** to issue this warning

See also : http://www.openssl.org/news/secadv_20030219.txt
http://lasecwww.epfl.ch/memo_ssl.shtml
http://eprint.iacr.org/2003/052/

Solution : Upgrade to version 0.9.6j (0.9.7b) or newer
Risk factor : Medium
CVE : CVE-2003-0078, CVE-2003-0131, CVE-2003-0147
BID : 6884, 7148
Other references : OSVDB:3945, OSVDB:3946, RHSA:RHSA-2003:101-01,
SuSE:SUSE-SA:2003:024

. Warning found on port https (443/tcp)

The remote web server appears to be running a version of
Apache that is less that 2.0.49 or 1.3.31.

These versions are vulnerable to a denial of service attack where a remote
attacker can block new connections to the server by connecting to a
listening
socket on a rarely accessed port.

Solution: Upgrade to Apache 2.0.49 or 1.3.31.
CVE : CVE-2004-0174
BID : 9921

. Warning found on port https (443/tcp)

The remote host is running a version of PHP which is older than 5.0.3 or
4.3.11

The remote version of this software is vulnerable to a set of
vulnerabilities in the EXIF module which have been fixed by the PHP Group.

See also : http://www.php.net/ChangeLog-5.php#5.0.4
http://www.php.net/ChangeLog-4.php#4.3.11

Solution : Upgrade to PHP 5.0.3 or 4.3.11
Risk factor : Medium
BID : 13143, 13163, 13164

. Warning found on port https (443/tcp)

The remote host is running a version of PHP <= 4.2.2.

The mail() function does not properly sanitize user input.
This allows users to forge email to make it look like it is
coming from a different source other than the server.

Users can exploit this even if SAFE_MODE is enabled.

Solution : Contact your vendor for the latest PHP release.
Risk factor : Medium
CVE : CVE-2002-0985, CVE-2002-0986
BID : 5562

. Warning found on port https (443/tcp)

Synopsis :

The remote service is prone to a denial of service attack.

Description :

According to its banner, the remote host is using a version of OpenSSL
which is older than 0.9.6m / 0.9.7d. There are several bugs in such
versions that may allow an attacker to cause a denial of service
against the remote host.

See also :

http://www.openssl.org/news/secadv_20040317.txt
http://archives.neohapsis.com/archives/bugtraq/2004-03/0165.html

Solution :

Upgrade to version 0.9.6m / 0.9.7d or newer.

Risk factor :

Medium / CVSS Base Score : 5.0
(CVSS2#AV:N/AC:L/Au:N/C:N/I:N/A:P)
CVE : CVE-2004-0079, CVE-2004-0081, CVE-2004-0112
BID : 9899
Other references : IAVA:2004-B-0006, OSVDB:4316, OSVDB:4317, OSVDB:4318

. Warning found on port https (443/tcp)

Synopsis :

The remote service encrypts traffic using a protocol with known
weaknesses.

Description :

The remote service accepts connections encrypted using SSL 2.0, which
reportedly suffers from several cryptographic flaws and has been
deprecated for several years. An attacker may be able to exploit
these issues to conduct man-in-the-middle attacks or decrypt
communications between the affected service and clients.

See also :

http://www.schneier.com/paper-ssl.pdf

Solution :

Consult the application's documentation to disable SSL 2.0 and use SSL
3.0 or TLS 1.0 instead.

Risk factor :

Medium / CVSS Base Score : 5.0
(CVSS2#AV:N/AC:L/Au:N/C:P/I:N/A:N)

. Warning found on port https (443/tcp)

Synopsis :

The remote web server is affected by a denial of service flaw.

Description :

The installed version of Apache with mod_ssl on the remote host
appears susceptible to a remote denial of service flaw under certain
atypical configurations. A remote attacker may be able to exploit
this issue to crash individual child processes or even the entire
server, thereby denying service to legitimate users.

See also :

http://issues.apache.org/bugzilla/show_bug.cgi?id=37791

Solution :

Update the Apache configuration to use "SSLRequire' whenever
'SSLCipherSuite' is used.

Risk factor :

Medium / CVSS Base Score : 5.4
(CVSS2#AV:N/AC:H/Au:N/C:N/I:N/A:C)
CVE : CVE-2005-3357
BID : 16152

. Information found on port https (443/tcp)

A SSLv2 server answered on this port

. Information found on port https (443/tcp)

A web server is running on this port through SSL

. Information found on port https (443/tcp)

```
Here is the SSLv2 server certificate:
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 0 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
 OU=SomeOrganizationalUnit,
 CN=localhost.localdomain/emailAddress=root@localhost.localdomain
        Validity
            Not Before: Dec 21 13:33:09 2007 GMT
            Not After : Dec 20 13:33:09 2008 GMT
        Subject: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
 OU=SomeOrganizationalUnit,
 CN=localhost.localdomain/emailAddress=root@localhost.localdomain
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:a0:5e:67:7b:f4:df:50:c7:2f:ca:2a:16:2f:6d:
                    82:1f:94:e7:0d:b7:05:b9:bb:5a:aa:19:a4:1e:e1:
                    d9:15:36:4c:44:9a:a8:28:71:5b:35:17:a8:af:4e:
                    1f:ed:01:87:92:d5:88:e3:66:02:9f:64:e6:70:50:
                    93:fc:65:c7:f0:12:e7:1c:d6:cb:55:b9:bc:c8:82:
                    e1:8a:11:9d:e6:62:c6:88:f9:80:60:30:f9:1a:c8:
                    86:d5:87:b9:0f:95:b3:5d:7b:a1:65:6c:b9:ea:42:
                    42:7a:ec:41:d0:0f:4d:81:0d:1c:9e:34:09:f3:17:
                    11:a7:27:55:35:2a:b5:c5:1b
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                F5:B9:F5:CB:CA:FA:A8:85:BA:B4:ED:26:86:CC:02:22:19:0E:B2:13
            X509v3 Authority Key Identifier:

    keyid:F5:B9:F5:CB:CA:FA:A8:85:BA:B4:ED:26:86:CC:02:22:19:0E:B2:13

    DirName:/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizational
Unit/CN=localhost.localdomain/emailAddress=root@localhost.localdomain
            serial:00

        X509v3 Basic Constraints:
            CA:TRUE
    Signature Algorithm: md5WithRSAEncryption
        74:e0:28:71:d2:fc:b9:7c:c1:e8:71:80:14:ec:60:88:37:cc:
        44:32:2b:30:1b:02:f2:06:cb:b6:68:ae:6e:3d:75:27:0a:7c:
        33:c7:bf:0d:fb:b5:85:5d:79:d9:bc:3f:4d:2c:e9:c5:99:03:
        cd:70:10:b5:c9:b9:a8:47:43:03:8a:4b:7c:64:e8:4c:45:8b:
        b4:43:ba:61:cf:9e:dc:22:da:2d:cd:2b:b3:ae:9f:af:30:f5:
```

18:c4:29:64:f0:83:e5:8d:9a:fc:43:dd:e3:f8:24:5b:49:04:
ed:85:d9:28:46:a0:5e:16:43:9a:cb:4c:92:5b:6a:61:ab:16:
3b:28
Here is the list of available SSLv2 ciphers:
RC4-MD5
EXP-RC4-MD5
RC2-CBC-MD5
EXP-RC2-CBC-MD5
DES-CBC-MD5
DES-CBC3-MD5
The SSLv2 server offers 4 strong ciphers, but also
0 medium strength and 2 weak "export class" ciphers.
The weak/medium ciphers may be chosen by an export-grade
or badly configured client software. They only offer a
limited protection against a brute force attack

Solution: disable those ciphers and upgrade your client
software if necessary.
See http://support.microsoft.com/default.aspx?scid=kb;en-us;216482
or http://httpd.apache.org/docs-2.0/mod/mod_ssl.html#sslciphersuite
This SSLv2 server also accepts SSLv3 connections.
This SSLv2 server also accepts TLSv1 connections.

. Information found on port https (443/tcp)

Synopsis :

The remote server is running with WebDAV enabled.

Description :

WebDAV is an industry standard extension to the HTTP specification.
It adds a capability for authorized users to remotely add and manage
the content of a web server.

If you do not use this extension, you should disable it.

Solution :

http://support.microsoft.com/default.aspx?kbid=241520

Risk factor :

None

. Information found on port https (443/tcp)

Synopsis :

The remote web server is prone to denial of service attacks.

Description :

According to its banner, the version of PHP installed on the remote host is vulnerable to a denial of service attack due to its failure to properly validate file data in the routines 'php_handle_iff' and 'php_handle_jpeg', which are called by the PHP function 'getimagesize'. Using a specially crafted image file, an attacker can trigger an infinite loop when 'getimagesize' is called, perhaps even remotely in the case image uploads are allowed.

See also :

http://www.idefense.com/intelligence/vulnerabilities/display.php?id=222
http://www.securityfocus.com/archive/1/394797
http://www.php.net/release_4_3_11.php

Solution :

Upgrade to PHP 4.3.11 / 5.0.4 or later.

Risk factor :

Low / CVSS Base Score : 3
(AV:R/AC:H/Au:NR/C:N/A:C/I:N/B:N)
CVE : CVE-2005-0524, CVE-2005-0525
BID : 12962, 12963

. Information found on port https (443/tcp)

Synopsis :

The remote service encrypts communications using SSL.

Description :

This script detects which SSL ciphers are supported by the remote service for encrypting communications.

See also :

http://www.openssl.org/docs/apps/ciphers.html

Risk factor :

None

Plugin output :

Here is the list of SSL ciphers supported by the remote server :

```
 Low Strength Ciphers (< 56-bit key)
   SSLv2
     EXP-RC2-CBC-MD5           Kx=RSA(512)   Au=RSA     Enc=RC2(40)
Mac=MD5    export
     EXP-RC4-MD5               Kx=RSA(512)   Au=RSA     Enc=RC4(40)
Mac=MD5    export
   SSLv3
     EXP-EDH-RSA-DES-CBC-SHA    Kx=DH(512)    Au=RSA     Enc=DES(40)
Mac=SHA1   export
     EXP-DES-CBC-SHA           Kx=RSA(512)   Au=RSA     Enc=DES(40)
Mac=SHA1   export
     EXP-RC2-CBC-MD5           Kx=RSA(512)   Au=RSA     Enc=RC2(40)
Mac=MD5    export
     EXP-RC4-MD5               Kx=RSA(512)   Au=RSA     Enc=RC4(40)
Mac=MD5    export
   TLSv1
     EXP-EDH-RSA-DES-CBC-SHA    Kx=DH(512)    Au=RSA     Enc=DES(40)
Mac=SHA1   export
     EXP-DES-CBC-SHA           Kx=RSA(512)   Au=RSA     Enc=DES(40)
Mac=SHA1   export
     EXP-RC2-CBC-MD5           Kx=RSA(512)   Au=RSA     Enc=RC2(40)
Mac=MD5    export
     EXP-RC4-MD5               Kx=RSA(512)   Au=RSA     Enc=RC4(40)
Mac=MD5    export

 Medium Strength Ciphers (>= 56-bit and < 112-bit key)
   SSLv2
     DES-CBC-MD5               Kx=RSA        Au=RSA     Enc=DES(56)
Mac=MD5
     RC4-64-MD5                Kx=RSA        Au=RSA     Enc=RC4(64)
Mac=MD5
   SSLv3
     EDH-RSA-DES-CBC-SHA       Kx=DH         Au=RSA     Enc=DES(56)
Mac=SHA1
     DES-CBC-SHA               Kx=RSA        Au=RSA     Enc=DES(56)
Mac=SHA1
   TLSv1
```

```
    EDH-RSA-DES-CBC-SHA       Kx=DH        Au=RSA     Enc=DES(56)
Mac=SHA1
    EXP1024-DES-CBC-SHA       Kx=RSA(1024)  Au=RSA     Enc=DES(56)
Mac=SHA1   export
    EXP1024-RC2-CBC-MD5       Kx=RSA(1024)  Au=RSA     Enc=RC2(56)
Mac=MD5    export
    EXP1024-RC4-MD5           Kx=RSA(1024)  Au=RSA     Enc=RC4(56)
Mac=MD5    export
    EXP1024-RC4-SHA           Kx=RSA(1024)  Au=RSA     Enc=RC4(56)
Mac=SHA1   export
    DES-CBC-SHA               Kx=RSA       Au=RSA     Enc=DES(56)
Mac=SHA1

 High Strength Ciphers (>= 112-bit key)
   SSLv2
    DES-CBC3-MD5              Kx=RSA       Au=RSA     Enc=3DES(168)
Mac=MD5
    RC2-CBC-MD5               Kx=RSA       Au=RSA     Enc=RC2(128)
Mac=MD5
    RC4-MD5                   Kx=RSA       Au=RSA     Enc=RC4(128)
Mac=MD5
   SSLv3
    EDH-RSA-DES-CBC3-SHA      Kx=DH        Au=RSA     Enc=3DES(168)
Mac=SHA1
    DES-CBC3-SHA              Kx=RSA       Au=RSA     Enc=3DES(168)
Mac=SHA1
    RC4-MD5                   Kx=RSA       Au=RSA     Enc=RC4(128)
Mac=MD5
    RC4-SHA                   Kx=RSA       Au=RSA     Enc=RC4(128)
Mac=SHA1
   TLSv1
    EDH-RSA-DES-CBC3-SHA      Kx=DH        Au=RSA     Enc=3DES(168)
Mac=SHA1
    DES-CBC3-SHA              Kx=RSA       Au=RSA     Enc=3DES(168)
Mac=SHA1
    RC4-MD5                   Kx=RSA       Au=RSA     Enc=RC4(128)
Mac=MD5
    RC4-SHA                   Kx=RSA       Au=RSA     Enc=RC4(128)
Mac=SHA1

The fields above are :

 {OpenSSL ciphername}
 Kx={key exchange}
 Au={authentication}
 Enc={symmetric encryption method}
 Mac={message authentication code}
 {export flag}
```

. Information found on port https (443/tcp)

Synopsis :

Some information about the remote HTTP configuration can be extracted.

Description :

This test gives some information about the remote HTTP protocol - the version used, whether HTTP Keep-Alive and HTTP pipelining are enabled, etc...

This test is informational only and does not denote any security problem

Solution :

None.

Risk factor :

None

Plugin output :

Protocol version : HTTP/1.1
SSL : yes
Pipelining : no
Keep-Alive : no
Options allowed : GET, HEAD, OPTIONS, TRACE
Headers :

Date: Fri, 28 Dec 2007 14:54:28 GMT
Server: Apache/1.3.23 (Unix) (Red-Hat/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b DAV/1.0.3 PHP/4.1.2 mod_perl/1.26
Last-Modified: Fri, 21 Dec 2007 15:53:24 GMT
ETag: "b3205-214-476be174"
Accept-Ranges: bytes
Content-Length: 532
Connection: close
Content-Type: text/html

. Information found on port general/tcp

TCP split NIDS evasion function is enabled. Some tests might
run slowly and you may get some false negative results

. Information found on port general/tcp

HTTP NIDS evasion functions are enabled.
You may get some false negative results

. Information found on port general/tcp

Synopsis :

The remote service implements TCP timestamps.

Description :

The remote host implements TCP timestamps, as defined by RFC1323.
A side effect of this feature is that the uptime of the remote
host can be sometimes be computed.

See also :

http://www.ietf.org/rfc/rfc1323.txt

Risk factor :

None

. Information found on port general/tcp

The following ports were open at the beginning of the scan but are now
closed:

Port 22 was detected as being open but is now closed

This might be an availability problem related which might be due to the
following reasons :

- The remote host is now down, either because a user turned it off during
the scan
- A selected denial of service was effective against this host
- A network outage has been experienced during the scan, and the remote
network cannot be reached from the Vulnerability Scanner any more
- This Vulnerability Scanner has been blacklisted by the system

administrator
or by automatic intrusion detection/prevention systems which have detected
the
vulnerability assessment.

In any case, the audit of the remote host might be incomplete and may need
to
be done again

. Information found on port general/tcp

Information about this scan :

Nessus version : 2.2.8 (Nessus 2.2.10 is available - consider upgrading)

Plugin feed version : 200711270935
Type of plugin feed : Registered (7 days delay)
Scanner IP : 10.0.0.132
Port scanner(s) : synscan nessus_tcp_scanner
Port range : 1-500
Thorough tests : no
Experimental tests : no
Paranoia level : 1
Report Verbosity : 1
Safe checks : no
Optimize the test : no
Max hosts : 1
Max checks : 8
Scan Start Date : 2007/12/28 11:45
Scan duration : 1085 sec

. Information found on port filenet-tms (32768/tcp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on TCP port 32768 :

- program: 100024 (status), version: 1

. Information found on port filenet-rpc (32769/tcp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on TCP port 32769 :

- program: 391002 (sgi_fam), version: 2

. Information found on port sunrpc (111/udp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.

Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on UDP port 111 :

- program: 100000 (portmapper), version: 2

. Information found on port filenet-tms (32768/udp)

Synopsis :

An ONC RPC service is running on the remote host.

Description :

By sending a DUMP request to the portmapper it was possible to
enumerate the ONC RPC services running on the remote port.
Using this information it is possible to connect and bind to
each service by sending an RPC request to the remote port.

Risk factor :

None

Plugin output :

The following RPC services are available on UDP port 32768 :

- program: 100024 (status), version: 1

. Information found on port x11 (6000/tcp)

Synopsis :

A X11 server is listening on the remote host

Description :

The remote host is running a X11 server. X11 is a client-server protocol which can be used to display graphical applications running on a given host on a remote client.

Since the X11 traffic is not ciphered, it is possible for an attacker to eavesdrop on the connection.

Solution :

Restrict access to this port. If the X11 client/server facility is not used, disable TCP entirely.

Risk factor :

Low / CVSS Base Score : 2
(AV:R/AC:H/Au:R/C:P/A:N/I:N/B:C)

Plugin output :

X11 Version : 11.0

This file was generated by the Nessus Security Scanner

Apêndice C

A Tabela C.1 contêm o arquivo *snort.conf*.

Tabela C.1: Arquivo *snort.conf*

```
#############################################
# IPS 01/12/2008
#--------------------------------------------------
#   http://www.snort.org     Snort 2.8.0.1 Ruleset
#     Contact: snort-sigs@lists.sourceforge.net
#--------------------------------------------------
# $Id$
#
##################################################
# This file contains a sample snort configuration.
# You can take the following steps to create your own custom configuration:
#
#  1) Set the variables for your network
#  2) Configure dynamic loaded libraries
#  3) Configure preprocessors
#  4) Configure output plugins
#  5) Add any runtime config directives
#  6) Customize your rule set
#
##################################################
# Step #1: Set the network variables:
#
# You must change the following variables to reflect your local network. The
# variable is currently setup for an RFC 1918 address space.
#
# You can specify it explicitly as:
#
# var HOME_NET 10.1.1.0/24
#
# or use global variable $<interfacename>_ADDRESS which will be always
# initialized to IP address and netmask of the network interface which you run
# snort at.  Under Windows, this must be specified as
# $(<interfacename>_ADDRESS), such as:
# $(\Device\Packet_{12345678-90AB-CDEF-1234567890AB}_ADDRESS)
#
# var HOME_NET $eth0_ADDRESS
#
# You can specify lists of IP addresses for HOME_NET
```

```
# by separating the IPs with commas like this:
#
# var HOME_NET [10.1.1.0/24,192.168.1.0/24]
#
# MAKE SURE YOU DON'T PLACE ANY SPACES IN YOUR LIST!
#
# or you can specify the variable to be any IP address
# like this:

var HOME_NET 10.0.2.0/21

# Set up the external network addresses as well.  A good start may be "any"
#var EXTERNAL_NET !$HOME_NET
var EXTERNAL_NET any

var MYSQLPASSWD senha


# Configure your server lists.  This allows snort to only look for attacks to
# systems that have a service up.  Why look for HTTP attacks if you are not
# running a web server?  This allows quick filtering based on IP addresses
# These configurations MUST follow the same configuration scheme as defined
# above for $HOME_NET.

# List of DNS servers on your network
var DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
var SMTP_SERVERS $HOME_NET

# List of web servers on your network
var HTTP_SERVERS $HOME_NET

# List of sql servers on your network
var SQL_SERVERS $HOME_NET

# List of telnet servers on your network
var TELNET_SERVERS $HOME_NET

# List of snmp servers on your network
var SNMP_SERVERS $HOME_NET

# Configure your service ports.  This allows snort to look for attacks destined
# to a specific application only on the ports that application runs on.  For
# example, if you run a web server on port 8081, set your HTTP_PORTS variable
# like this:
#
# portvar HTTP_PORTS 8081
#
```

```
# Ports you run web servers on
portvar HTTP_PORTS 80

# NOTE:  If you wish to define multiple HTTP ports, use the portvar
# syntax to represent lists of ports and port ranges.  Examples:
## portvar HTTP_PORTS [80,8080]
## portvar HTTP_PORTS [80,8000:8080]
# And only include the rule that uses $HTTP_PORTS once.
#
# The pre-2.8.0 approach of redefining the variable to a different port and
# including the rules file twice is obsolete.  See README.variables for more
# details.

# Port https
portvar SSH_PORTS 22

# Ports you want to look for SHELLCODE on.
portvar SHELLCODE_PORTS !80

# Ports you might see oracle attacks on
portvar ORACLE_PORTS 1521

# other variables
#
# AIM servers.  AOL has a habit of adding new AIM servers, so instead of
# modifying the signatures when they do, we add them to this list of servers.
var                                                                    AIM_SERVERS
[64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/24,205.188.3.0/24,205.18
8.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.153.0/24,205.188.179.0/24,205.188.248.0/24]

# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
var RULE_PATH ../Rules-snort/rules
var PREPROC_RULE_PATH ../preproc_rules

# Configure the snort decoder
# ============================
#
# Snort's decoder will alert on lots of things such as header
# truncation or options of unusual length or infrequently used tcp options
#
#
# Stop generic decode events:
#
# config disable_decode_alerts
#
# Stop Alerts on experimental TCP options
#
```

```
# config disable_tcpopt_experimental_alerts
#
# Stop Alerts on obsolete TCP options
#
# config disable_tcpopt_obsolete_alerts
#
# Stop Alerts on T/TCP alerts
#
# In snort 2.0.1 and above, this only alerts when a TCP option is detected
# that shows T/TCP being actively used on the network.  If this is normal
# behavior for your network, disable the next option.
#
# config disable_tcpopt_ttcp_alerts
#
# Stop Alerts on all other TCPOption type events:
#
# config disable_tcpopt_alerts
#
# Stop Alerts on invalid ip options
#
# config disable_ipopt_alerts
#
# Alert if value in length field (IP, TCP, UDP) is greater than the
# actual length of the captured portion of the packet that the length
# is supposed to represent:
#
# config enable_decode_oversized_alerts
#
# Same as above, but drop packet if in Inline mode -
# enable_decode_oversized_alerts must be enabled for this to work:
#
# config enable_decode_oversized_drops
#

# Configure the detection engine
# ==============================
#
# Use a different pattern matcher in case you have a machine with very limited
# resources:
#
# config detection: search-method lowmem

# Configure Inline Resets
# =======================
#
# If running an iptables firewall with snort in InlineMode() we can now
# perform resets via a physical device. We grab the indev from iptables
# and use this for the interface on which to send resets. This config
# option takes an argument for the src mac address you want to use in the
```

```
# reset packet.  This way the bridge can remain stealthy. If the src mac
# option is not set we use the mac address of the indev device. If we
# don't set this option we will default to sending resets via raw socket,
# which needs an ipaddress to be assigned to the int.
#
# config layer2resets: 00:06:76:DD:5F:E3

###################################################
# Step #2: Configure dynamic loaded libraries
#
# If snort was configured to use dynamically loaded libraries,
# those libraries can be loaded here.
#
# Each of the following configuration options can be done via
# the command line as well.
#
# Load all dynamic preprocessors from the install path
# (same as command line option --dynamic-preprocessor-lib-dir)
#
dynamicpreprocessor directory /usr/local/lib/snort_dynamicpreprocessor/
#
# Load a specific dynamic preprocessor library from the install path
# (same as command line option --dynamic-preprocessor-lib)
#
# dynamicpreprocessor file /usr/local/lib/snort_dynamicpreprocessor/libdynamicexample.so
#
# Load a dynamic engine from the install path
# (same as command line option --dynamic-engine-lib)
#
dynamicengine /usr/local/lib/snort_dynamicengine/libsf_engine.so
#
# Load all dynamic rules libraries from the install path
# (same as command line option --dynamic-detection-lib-dir)
#
# dynamicdetection directory /usr/local/lib/snort_dynamicrule/
#
# Load a specific dynamic rule library from the install path
# (same as command line option --dynamic-detection-lib)
#
# dynamicdetection file /usr/local/lib/snort_dynamicrule/libdynamicexamplerule.so
#

###################################################
# Step #3: Configure preprocessors
#
# General configuration for preprocessors is of
# the form
# preprocessor <name_of_processor>: <configuration_options>
```

```
# Configure Flow tracking module
# ------------------------------
#
# The Flow tracking module is meant to start unifying the state keeping
# mechanisms of snort into a single place. Right now, only a portscan detector
# is implemented but in the long term,  many of the stateful subsystems of
# snort will be migrated over to becoming flow plugins. This must be enabled
# for flow-portscan to work correctly.
#
# See README.flow for additional information
#
#preprocessor flow: stats_interval 0 hash 2

# frag3: Target-based IP defragmentation
# --------------------------------------
#
# Frag3 is a brand new IP defragmentation preprocessor that is capable of
# performing "target-based" processing of IP fragments.  Check out the
# README.frag3 file in the doc directory for more background and configuration
# information.
#
# Frag3 configuration is a two step process, a global initialization phase
# followed by the definition of a set of defragmentation engines.
#
# Global configuration defines the number of fragmented packets that Snort can
# track at the same time and gives you options regarding the memory cap for the
# subsystem or, optionally, allows you to preallocate all the memory for the
# entire frag3 system.
#
# frag3_global options:
#   max_frags: Maximum number of frag trackers that may be active at once.
#              Default value is 8192.
#   memcap: Maximum amount of memory that frag3 may access at any given time.
#           Default value is 4MB.
#   prealloc_frags: Maximum number of individual fragments that may be processed
#                   at once.  This is instead of the memcap system, uses static
#                   allocation to increase performance.  No default value.  Each
#                   preallocated fragment eats ~1550 bytes.
#
# Target-based behavior is attached to an engine as a "policy" for handling
# overlaps and retransmissions as enumerated in the Paxson paper.  There are
# currently five policy types available: "BSD", "BSD-right", "First", "Linux"
# and "Last".  Engines can be bound to standard Snort CIDR blocks or
# IP lists.
#
# frag3_engine options:
#   timeout: Amount of time a fragmented packet may be active before expiring.
#            Default value is 60 seconds.
#   ttl_limit: Limit of delta allowable for TTLs of packets in the fragments.
```

```
#             Based on the initial received fragment TTL.
#   min_ttl: Minimum acceptable TTL for a fragment, frags with TTLs below this
#           value will be discarded.  Default value is 0.
#   detect_anomalies: Activates frag3's anomaly detection mechanisms.
#   policy: Target-based policy to assign to this engine.  Default is BSD.
#   bind_to: IP address set to bind this engine to.  Default is all hosts.
#
# Frag3 configuration example:
#preprocessor frag3_global: max_frags 65536, prealloc_frags 262144
#preprocessor frag3_engine: policy linux \
#                       bind_to [10.1.1.12/32,10.1.1.13/32] \
#                       detect_anomalies
#preprocessor frag3_engine: policy first \
#                       bind_to 10.2.1.0/24 \
#                       detect_anomalies
#preprocessor frag3_engine: policy last \
#                       bind_to 10.3.1.0/24
#preprocessor frag3_engine: policy bsd

preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first detect_anomalies


# stream4: stateful inspection/stream reassembly for Snort
#---------------------------------------------------------------------
# Use in concert with the -z [all|est] command line switch to defeat stick/snot
# against TCP rules.  Also performs full TCP stream reassembly, stateful
# inspection of TCP streams, etc.  Can statefully detect various portscan
# types, fingerprinting, ECN, etc.

# stateful inspection directive
# no arguments loads the defaults (timeout 30, memcap 8388608)
# options (options are comma delimited):
#   detect_scans - stream4 will detect stealth portscans and generate alerts
#                  when it sees them when this option is set
#   detect_state_problems - detect TCP state problems, this tends to be very
#                       noisy because there are a lot of crappy ip stack
#                       implementations out there
#
#   disable_evasion_alerts - turn off the possibly noisy mitigation of
#                        overlapping sequences.
#
#   ttl_limit [number]     - differential of the initial ttl on a session versus
#                         the normal that someone may be playing games.
#                         Routing flap may cause lots of false positives.
#
#   keepstats [machine|binary] - keep session statistics, add "machine" to
#                        get them in a flat format for machine reading, add
#                        "binary" to get them in a unified binary output
```

```
#                     format
#   noinspect - turn off stateful inspection only
#   timeout [number] - set the session timeout counter to [number] seconds,
#                   default is 30 seconds
#   max_sessions [number] - limit the number of sessions stream4 keeps
#                     track of
#   memcap [number] - limit stream4 memory usage to [number] bytes (does
#                   not include session tracking, which is set by the
#                   max_sessions option)
#   log_flushed_streams - if an event is detected on a stream this option will
#                     cause all packets that are stored in the stream4
#                     packet buffers to be flushed to disk.  This only
#                     works when logging in pcap mode!
#   server_inspect_limit [bytes] - Byte limit on server side inspection.
#   enable_udp_sessions - turn on tracking of "sessions" over UDP.  Requires
#                     configure --enable-stream4udp.  UDP sessions are
#                     only created when there is a rule for the sender or
#                     responder that has a flow or flowbits keyword.
#   max_udp_sessions [number] - limit the number of simultaneous UDP sessions
#                         to track
#   udp_ignore_any - Do not inspect UDP packets unless there is a port specific
#                 rule for a given port.  This is a performance improvement
#                 and turns off inspection for udp xxx any -> xxx any rules
#   cache_clean_sessions [number] - Cleanup the session cache by number sessions
#                             at a time.  The larger the value, the
#                             more sessions are purged from the cache when
#                             the session limit or memcap is reached.
#                             Defaults to 5.
#
#
#
# Stream4 uses Generator ID 111 and uses the following SIDS
# for that GID:
#  SID    Event description
# -----   -------------------
#   1      Stealth activity
#   2      Evasive RST packet
#   3      Evasive TCP packet retransmission
#   4      TCP Window violation
#   5      Data on SYN packet
#   6      Stealth scan: full XMAS
#   7      Stealth scan: SYN-ACK-PSH-URG
#   8      Stealth scan: FIN scan
#   9      Stealth scan: NULL scan
#   10     Stealth scan: NMAP XMAS scan
#   11     Stealth scan: Vecna scan
#   12     Stealth scan: NMAP fingerprint scan stateful detect
#   13     Stealth scan: SYN-FIN scan
#   14     TCP forward overlap
```

```
#preprocessor stream4: disable_evasion_alerts

# tcp stream reassembly directive
# no arguments loads the default configuration
#   Only reassemble the client,
#   Only reassemble the default list of ports (See below),
#   Give alerts for "bad" streams
#
# Available options (comma delimited):
#   clientonly - reassemble traffic for the client side of a connection only
#   serveronly - reassemble traffic for the server side of a connection only
#   both - reassemble both sides of a session
#   noalerts - turn off alerts from the stream reassembly stage of stream4
#   ports [list] - use the space separated list of ports in [list], "all"
#                  will turn on reassembly for all ports, "default" will turn
#                  on reassembly for ports 21, 23, 25, 42, 53, 80, 110,
#                  111, 135, 136, 137, 139, 143, 445, 513, 1433, 1521,
#                  and 3306
#   favor_old - favor an old segment (based on sequence number) over a new one.
#               This is the default.
#   favor_new - favor an new segment (based on sequence number) over an old one.
#   overlap_limit [number] - limit on overlaping segments for a session.
#   flush_on_alert - flushes stream when an alert is generated for a session.
#   flush_behavior [mode] -
#          default      - use old static flushpoints (default)
#          large_window - use new larger static flushpoints
#          random       - use random flushpoints defined by flush_base,
#                         flush_seed and flush_range
#   flush_base [number] - lowest allowed random flushpoint (512 by default)
#   flush_range [number] - number is the space within which random flushpoints
#                          are generated (default 1213)
#   flush_seed [number] - seed for the random number generator, defaults to
#                         Snort PID + time
#
# Using the default random flushpoints, the smallest flushpoint is 512,
# and the largest is 1725 bytes.
#preprocessor stream4_reassemble

# stream5: Target Based stateful inspection/stream reassembly for Snort
# ---------------------------------------------------------------------
# Stream5 is a target-based stream engine for Snort.  Its functionality
# replaces that of Stream4.  Consequently, BOTH Stream4 and Stream5
# cannot be used simultaneously.  Comment out the stream4 configurations
# above to use Stream5.
#
# See README.stream5 for details on the configuration options.
#
# Example config (that emulates Stream4 with UDP support compiled in)
```

```
preprocessor stream5_global: max_tcp 8192, track_tcp yes, \
                        track_udp no
preprocessor stream5_tcp: policy first, use_static_footprint_sizes
# preprocessor stream5_udp: ignore_any_rules


# Performance Statistics
# ----------------------
# Documentation for this is provided in the Snort Manual.  You should read it.
# It is included in the release distribution as doc/snort_manual.pdf
#
# preprocessor perfmonitor: time 300 file /var/snort/snort.stats pktcnt 10000

# http_inspect: normalize and detect HTTP traffic and protocol anomalies
#
# lots of options available here. See doc/README.http_inspect.
# unicode.map should be wherever your snort.conf lives, or given
# a full path to where snort can find it.
preprocessor http_inspect: global \
   iis_unicode_map unicode.map 1252

preprocessor http_inspect_server: server default \
   profile all ports { 80 8080 8180 } oversize_dir_length 500

#
#  Example unique server configuration
#
#preprocessor http_inspect_server: server 1.1.1.1 \
#    ports { 80 3128 8080 } \
#    flow_depth 0 \
#    ascii no \
#    double_decode yes \
#    non_rfc_char { 0x00 } \
#    chunk_length 500000 \
#    non_strict \
#    oversize_dir_length 300 \
#    no_alerts


# rpc_decode: normalize RPC traffic
# ---------------------------------
# RPC may be sent in alternate encodings besides the usual 4-byte encoding
# that is used by default. This plugin takes the port numbers that RPC
# services are running on as arguments - it is assumed that the given ports
# are actually running this type of service. If not, change the ports or turn
# it off.
# The RPC decode preprocessor uses generator ID 106
#
# arguments: space separated list
```

```
# alert_fragments - alert on any rpc fragmented TCP data
# no_alert_multiple_requests - don't alert when >1 rpc query is in a packet
# no_alert_large_fragments - don't alert when the fragmented
#                            sizes exceed the current packet size
# no_alert_incomplete - don't alert when a single segment
#                       exceeds the current packet size

preprocessor rpc_decode: 111 32771

# bo: Back Orifice detector
# -------------------------
# Detects Back Orifice traffic on the network.
#
# arguments:
#   syntax:
#     preprocessor bo: noalert { client | server | general | snort_attack } \
#                      drop    { client | server | general | snort_attack }
#   example:
#     preprocessor bo: noalert { general server } drop { snort_attack }

#
# The Back Orifice detector uses Generator ID 105 and uses the
# following SIDS for that GID:
#  SID     Event description
# -----   -------------------
#   1       Back Orifice traffic detected
#   2       Back Orifice Client Traffic Detected
#   3       Back Orifice Server Traffic Detected
#   4       Back Orifice Snort Buffer Attack

preprocessor bo

# telnet_decode: Telnet negotiation string normalizer
# ---------------------------------------------------
# This preprocessor "normalizes" telnet negotiation strings from telnet and ftp
# traffic.  It works in much the same way as the http_decode preprocessor,
# searching for traffic that breaks up the normal data stream of a protocol and
# replacing it with a normalized representation of that traffic so that the
# "content" pattern matching keyword can work without requiring modifications.
# This preprocessor requires no arguments.
#
# DEPRECATED in favor of ftp_telnet dynamic preprocessor
#preprocessor telnet_decode
#
# ftp_telnet: FTP & Telnet normalizer, protocol enforcement and buff overflow
# ---------------------------------------------------------------------------
# This preprocessor normalizes telnet negotiation strings from telnet and
# ftp traffic.  It looks for traffic that breaks the normal data stream
# of the protocol, replacing it with a normalized representation of that
```

```
# traffic so that the "content" pattern matching keyword can work without
# requiring modifications.
#
# It also performs protocol correctness checks for the FTP command channel,
# and identifies open FTP data transfers.
#
# FTPTelnet has numerous options available, please read
# README.ftptelnet for help configuring the options for the global
# telnet, ftp server, and ftp client sections for the protocol.

#####
# Per Step #2, set the following to load the ftptelnet preprocessor
# dynamicpreprocessor file <full path to libsf_ftptelnet_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_ftptelnet_preproc.so>

preprocessor ftp_telnet: global \
  encrypted_traffic yes \
  inspection_type stateful

preprocessor ftp_telnet_protocol: telnet \
  normalize \
  ayt_attack_thresh 200

# This is consistent with the FTP rules as of 18 Sept 2004.
# CWD can have param length of 200
# MODE has an additional mode of Z (compressed)
# Check for string formats in USER & PASS commands
# Check nDTM commands that set modification time on the file.
preprocessor ftp_telnet_protocol: ftp server default \
  def_max_param_len 100 \
  alt_max_param_len 200 { CWD } \
  cmd_validity MODE < char ASBCZ > \
  cmd_validity MDTM < [ date nnnnnnnnnnnnnn[.n[n[n]]] ] string > \
  chk_str_fmt { USER PASS RNFR RNTO SITE MKD } \
  telnet_cmds yes \
  data_chan

preprocessor ftp_telnet_protocol: ftp client default \
  max_resp_len 256 \
  bounce yes \
  telnet_cmds yes

# smtp: SMTP normalizer, protocol enforcement and buffer overflow
# --------------------------------------------------------------------------
# This preprocessor normalizes SMTP commands by removing extraneous spaces.
# It looks for overly long command lines, response lines, and data header lines.
# It can alert on invalid commands, or specific valid commands.  It can optionally
# ignore mail data, and can ignore TLS encrypted data.
```

```
#
# SMTP has numerous options available, please read README.SMTP for help
# configuring options.

#####
# Per Step #2, set the following to load the smtp preprocessor
# dynamicpreprocessor file <full path to libsf_smtp_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_smtp_preproc.so>

preprocessor smtp: \
  ports { 25 587 691 } \
  inspection_type stateful \
  normalize cmds \
  normalize_cmds { EXPN VRFY RCPT } \
  alt_max_command_line_len 260 { MAIL } \
  alt_max_command_line_len 300 { RCPT } \
  alt_max_command_line_len 500 { HELP HELO ETRN } \
  alt_max_command_line_len 255 { EXPN VRFY }

# sfPortscan
# ----------
# Portscan detection module.  Detects various types of portscans and
# portsweeps.  For more information on detection philosophy, alert types,
# and detailed portscan information, please refer to the README.sfportscan.
#
# -configuration options-
#     proto { tcp udp icmp ip all }
#       The arguments to the proto option are the types of protocol scans that
#       the user wants to detect.  Arguments should be separated by spaces and
#       not commas.
#     scan_type { portscan portsweep decoy_portscan distributed_portscan all }
#       The arguments to the scan_type option are the scan types that the
#       user wants to detect.  Arguments should be separated by spaces and not
#       commas.
#     sense_level { low|medium|high }
#       There is only one argument to this option and it is the level of
#       sensitivity in which to detect portscans.  The 'low' sensitivity
#       detects scans by the common method of looking for response errors, such
#       as TCP RSTs or ICMP unreachables.  This level requires the least
#       tuning.  The 'medium' sensitivity level detects portscans and
#       filtered portscans (portscans that receive no response).  This
#       sensitivity level usually requires tuning out scan events from NATed
#       IPs, DNS cache servers, etc.  The 'high' sensitivity level has
#       lower thresholds for portscan detection and a longer time window than
#       the 'medium' sensitivity level.  Requires more tuning and may be noisy
#       on very active networks.  However, this sensitivity levels catches the
#       most scans.
#     memcap { positive integer }
```

```
#       The maximum number of bytes to allocate for portscan detection.  The
#       higher this number the more nodes that can be tracked.
#     logfile { filename }
#       This option specifies the file to log portscan and detailed portscan
#       values to.  If there is not a leading /, then snort logs to the
#       configured log directory.  Refer to README.sfportscan for details on
#       the logged values in the logfile.
#     watch_ip { Snort IP List }
#     ignore_scanners { Snort IP List }
#     ignore_scanned { Snort IP List }
#       These options take a snort IP list as the argument.  The 'watch_ip'
#       option specifies the IP(s) to watch for portscan.  The
#       'ignore_scanners' option specifies the IP(s) to ignore as scanners.
#       Note that these hosts are still watched as scanned hosts.  The
#       'ignore_scanners' option is used to tune alerts from very active
#       hosts such as NAT, nessus hosts, etc.  The 'ignore_scanned' option
#       specifies the IP(s) to ignore as scanned hosts.  Note that these hosts
#       are still watched as scanner hosts.  The 'ignore_scanned' option is
#       used to tune alerts from very active hosts such as syslog servers, etc.
#     detect_ack_scans
#       This option will include sessions picked up in midstream by the stream
#       module, which is necessary to detect ACK scans.  However, this can lead to
#       false alerts, especially under heavy load with dropped packets; which is why
#       the option is off by default.
#
preprocessor sfportscan: proto  { all } \
                     memcap { 10000000 } \
                     sense_level { low }

# arpspoof
#----------------------------------------
# Experimental ARP detection code from Jeff Nathan, detects ARP attacks,
# unicast ARP requests, and specific ARP mapping monitoring.  To make use of
# this preprocessor you must specify the IP and hardware address of hosts on
# the same layer 2 segment as you.  Specify one host IP MAC combo per line.
# Also takes a "-unicast" option to turn on unicast ARP request detection.
# Arpspoof uses Generator ID 112 and uses the following SIDS for that GID:

#  SID    Event description
# -----   -------------------
#   1      Unicast ARP request
#   2      Etherframe ARP mismatch (src)
#   3      Etherframe ARP mismatch (dst)
#   4      ARP cache overwrite attack

#preprocessor arpspoof
#preprocessor arpspoof_detect_host: 192.168.40.1 f0:0f:00:f0:0f:00

# ssh
```

```
#--------------------------------------
# EXPERIMENTAL CODE!!!
#
# THIS CODE IS STILL EXPERIMENTAL AND MAY OR MAY NOT BE STABLE!
# USE AT YOUR OWN RISK!  DO NOT USE IN PRODUCTION ENVIRONMENTS.
# YOU HAVE BEEN WARNED.
#
# The SSH preprocessor detects the following exploits: Gobbles, CRC 32,
# Secure CRT, and the Protocol Mismatch exploit.
#
# Both Gobbles and CRC 32 attacks occur after the key exchange, and are
# therefore encrypted.  Both attacks involve sending a large payload
# (20kb+) to the server immediately after the authentication challenge.
# To detect the attacks, the SSH preprocessor counts the number of bytes
# transmitted to the server.  If those bytes exceed a pre-defined limit
# within a pre-define number of packets, an alert is generated.  Since
# Gobbles only effects SSHv2 and CRC 32 only effects SSHv1, the SSH
# version string exchange is used to distinguish the attacks.
#
# The Secure CRT and protocol mismatch exploits are observable before
# the key exchange.
#
# SSH has numerous options available, please read README.ssh for help
# configuring options.

#####
# Per Step #2, set the following to load the ssh preprocessor
# dynamicpreprocessor file <full path to libsf_ssh_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_ssh_preproc.so>
#
#preprocessor ssh: server_ports { 22 } \
#                max_client_bytes 19600 \
#                max_encrypted_packets 20

# DCE/RPC
#--------------------------------------
#
# The dcerpc preprocessor detects and decodes SMB and DCE/RPC traffic.
# It is primarily interested in DCE/RPC data, and only decodes SMB
# to get at the DCE/RPC data carried by the SMB layer.
#
# Currently, the preprocessor only handles reassembly of fragmentation
# at both the SMB and DCE/RPC layer.  Snort rules can be evaded by
# using both types of fragmentation; with the preprocessor enabled
# the rules are given a buffer with a reassembled SMB or DCE/RPC
# packet to examine.
#
# At the SMB layer, only fragmentation using WriteAndX is currently
```

```
# reassembled.  Other methods will be handled in future versions of
# the preprocessor.
#
# Autodetection of SMB is done by looking for "\xFFSMB" at the start of
# the SMB data, as well as checking the NetBIOS header (which is always
# present for SMB) for the type "SMB Session".
#
# Autodetection of DCE/RPC is not as reliable.  Currently, two bytes are
# checked in the packet.  Assuming that the data is a DCE/RPC header,
# one byte is checked for DCE/RPC version (5) and another for the type
# "DCE/RPC Request".  If both match, the preprocessor proceeds with that
# assumption that it is looking at DCE/RPC data.  If subsequent checks
# are nonsensical, it ends processing.
#
# DCERPC has numerous options available, please read README.dcerpc for help
# configuring options.

#####
# Per Step #2, set the following to load the dcerpc preprocessor
# dynamicpreprocessor file <full path to libsf_dcerpc_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_dcerpc_preproc.so>

preprocessor dcerpc: \
   autodetect \
   max_frag_size 3000 \
   memcap 100000

# DNS
#--------------------------------------
# The dns preprocessor (currently) decodes DNS Response traffic
# and detects a few vulnerabilities.
#
# DNS has a few options available, please read README.dns for
# help configuring options.

#####
# Per Step #2, set the following to load the dns preprocessor
# dynamicpreprocessor file <full path to libsf_dns_preproc.so>
# or use commandline option
# --dynamic-preprocessor-lib <full path to libsf_dns_preproc.so>

preprocessor dns: \
   ports { 53 } \
   enable_rdata_overflow

##################################################################
# Step #4: Configure output plugins
#
```

```
# Uncomment and configure the output plugins you decide to use.  General
# configuration for output plugins is of the form:
#
# output <name_of_plugin>: <configuration_options>
#
# alert_syslog: log alerts to syslog
# ----------------------------------
# Use one or more syslog facilities as arguments.  Win32 can also optionally
# specify a particular hostname/port.  Under Win32, the default hostname is
# '127.0.0.1', and the default port is 514.
#
# [Unix flavours should use this format...]
# output alert_syslog: LOG_AUTH LOG_ALERT
#
# [Win32 can use any of these formats...]
# output alert_syslog: LOG_AUTH LOG_ALERT
# output alert_syslog: host=hostname, LOG_AUTH LOG_ALERT
# output alert_syslog: host=hostname:port, LOG_AUTH LOG_ALERT

# log_tcpdump: log packets in binary tcpdump format
# -------------------------------------------------
# The only argument is the output file name.
#
# output log_tcpdump: tcpdump.log

# database: log to a variety of databases
# ---------------------------------------
# See the README.database file for more information about configuring
# and using this plugin.
#
output database: log, mysql, user=snort password=$MYSQLPASSWD dbname=snort
host=localhost
# output database: log, mysql, user=root password=test dbname=db host=localhost
# output database: alert, postgresql, user=snort dbname=snort
# output database: log, odbc, user=snort dbname=snort
# output database: log, mssql, dbname=snort user=snort password=test
# output database: log, oracle, dbname=snort user=snort password=test

# unified: Snort unified binary format alerting and logging
# -------------------------------------------------------------
# The unified output plugin provides two new formats for logging and generating
# alerts from Snort, the "unified" format.  The unified format is a straight
# binary format for logging data out of Snort that is designed to be fast and
# efficient.  Used with barnyard (the new alert/log processor), most of the
# overhead for logging and alerting to various slow storage mechanisms such as
# databases or the network can now be avoided.
#
# Check out the spo_unified.h file for the data formats.
#
```

```
# Two arguments are supported.
#    filename - base filename to write to (current time_t is appended)
#    limit    - maximum size of spool file in MB (default: 128)
#
# output alert_unified: filename snort.alert, limit 128
# output log_unified: filename snort.log, limit 128


# prelude: log to the Prelude Hybrid IDS system
# ---------------------------------------------
#
# profile = Name of the Prelude profile to use (default is snort).
#
# Snort priority to IDMEF severity mappings:
# high < medium < low < info
#
# These are the default mapped from classification.config:
# info   = 4
# low    = 3
# medium = 2
# high   = anything below medium
#
# output alert_prelude
# output alert_prelude: profile=snort-profile-name


# You can optionally define new rule types and associate one or more output
# plugins specifically to that type.
#
# This example will create a type that will log to just tcpdump.
# ruletype suspicious
# {
#   type log
#   output log_tcpdump: suspicious.log
# }
#
# EXAMPLE RULE FOR SUSPICIOUS RULETYPE:
# suspicious tcp $HOME_NET any -> $HOME_NET 6667 (msg:"Internal IRC Server";)
#
# This example will create a rule type that will log to syslog and a mysql
# database:
# ruletype redalert
# {
#   type alert
#   output alert_syslog: LOG_AUTH LOG_ALERT
#   output database: log, mysql, user=snort dbname=snort host=localhost
# }
#
# EXAMPLE RULE FOR REDALERT RULETYPE:
```

```
# redalert tcp $HOME_NET any -> $EXTERNAL_NET 31337 \
#   (msg:"Someone is being LEET"; flags:A+;)

#
# Include classification & priority settings
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\etc\classification.config
#

include classification.config

#
# Include reference systems
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\etc\reference.config
#

include reference.config

##################################################################
# Step #5: Configure snort with config statements
#
# See the snort manual for a full set of configuration references
#
# config flowbits_size: 64
#
# New global ignore_ports config option from Andy Mullican
#
# config ignore_ports: <tcp|udp> <list of ports separated by whitespace>
# config ignore_ports: tcp 21 6667:6671 1356
# config ignore_ports: udp 1:17 53


##################################################################
# Step #6: Customize your rule set
#
# Up to date snort rules are available at http://www.snort.org
#
# The snort web site has documentation about how to write your own custom snort
# rules.

#========================================
# Include all relevant rulesets here
#
# The following rulesets are disabled by default:
#
#   web-attacks, backdoor, shellcode, policy, porn, info, icmp-info, virus,
#   chat, multimedia, and p2p
#
```

```
# These rules are either site policy specific or require tuning in order to not
# generate false positive alerts in most enviornments.
#
# Please read the specific include file for more information and
# README.alert_order for how rule ordering affects how alerts are triggered.
#========================================

include $RULE_PATH/local.rules
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/exploit.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/tftp.rules

include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-client.rules
include $RULE_PATH/web-php.rules

include $RULE_PATH/sql.rules
include $RULE_PATH/x11.rules
include $RULE_PATH/icmp.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/oracle.rules
include $RULE_PATH/mysql.rules
include $RULE_PATH/snmp.rules

include $RULE_PATH/smtp.rules
include $RULE_PATH/imap.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules

include $RULE_PATH/nntp.rules
include $RULE_PATH/other-ids.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/shellcode.rules
```

```
include $RULE_PATH/policy.rules
# include $RULE_PATH/porn.rules
include $RULE_PATH/info.rules
include $RULE_PATH/icmp-info.rules
# include $RULE_PATH/virus.rules
# include $RULE_PATH/chat.rules
# include $RULE_PATH/multimedia.rules
# include $RULE_PATH/p2p.rules
include $RULE_PATH/spyware-put.rules
include $RULE_PATH/specific-threats.rules
include $RULE_PATH/experimental.rules

# include $PREPROC_RULE_PATH/preprocessor.rules
# include $PREPROC_RULE_PATH/decoder.rules

# Include any thresholding or suppression commands. See threshold.conf in the
# <snort src>/etc directory for details. Commands don't necessarily need to be
# contained in this conf, but a separate conf makes it easier to maintain them.
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\etc\threshold.conf
# Uncomment if needed.
# include threshold.conf
```