Universidade Federal de Pernambuco
Centro de Informática

PIPS: UM SISTEMA PROATIVO DE PREVENÇÃO CONTRA INTRUSÕES

Por

Cleiton Soares Martins

Dissertação de Mestrado

Recife, Agosto de 2005

Cleiton Soares Martins

PIPS – UM SISTEMA PROATIVO DE PREVENÇÃO CONTRA INTRUSÕES

Este trabalho foi submetido à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação

Orientador: Prof. Dr. André Santos

Recife, Agosto de 2005

Martins, Cleiton Soares
PIPS- um sistema proativo de prevenção contra intrusões / Cleiton Soares Martins. – Recife : O Autor, 2005.
68 folhas : tab., fig.

Dissertação (mestrado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2005.

Inclui bibliografia.

1. Informática – Sistemas distribuídos – Rede de computadores. 2. Segurança da informação – Sistemas de detecção de intrusões (IDS) – Desenvolvimento. 3. Agentes inteligentes – Utilização – Alertas de intrusão – Inventário de rede - Correlação. I. Título.

004.492.2 CDU (2.ed.) UFPE
005.8 CDD (22.ed.) BC2006 – 558

Agradecimentos

Gostaria primeiramente de agradecer a minha família e amigos pelo incentivo nos momentos mais difíceis e pela paciência e compreensão ao longo do desenvolvimento deste trabalho.

Obrigado a todos do Centro de Informática da UFPE em particular ao Prof. André Santos pela orientação direta e a todos os professores que indiretamente contribuíram, muitas vezes sem saber, com idéias e inspirações indispensáveis.

Aos colegas de trabalho que sempre atenderam com presteza aos pedidos de ajuda e que muito colaboraram para o amadurecimento das idéias e forneceram uma visão prática do problema, visão esta que sempre norteou o desenvolvimento deste trabalho e contribuiu de forma definitiva para os resultados alcançados.

Dedico este trabalho a minha mãe e meus irmãos, sem eles nada disso teria sido possível.

Resumo

O processo de proteger um sistema computacional envolve uma série de procedimentos de segurança seguidos de constante monitoração. Uma das maneiras mais comuns de monitorar redes de computadores contra ameaças externas são os Sistemas de Detecção de Intrusão (IDS). Mais recentemente os IDS evoluíram passando também a poder evitar ataques em tempo real, sendo comercialmente denominados Sistemas de Prevenção contra Intrusões (IPS). Além da atuação de ambos os sistemas ser essencialmente reativa, eles possuem capacidade de análise limitada por ter uma visão restrita da rede.

Nesse trabalho apresentamos uma nova abordagem para o problema da contínua monitoração denominada **Proactive Intrusion Prevention Systems (PIPS)**. Esse sistema atua de maneira proativa monitorando constantemente a rede através de varreduras periódicas que montam um perfil ativo da rede. Além de fornecer uma visão detalhada do estado da rede, o perfil é utilizado para correlações com eventos de IDS produzindo análises mais refinadas. A abordagem consiste na utilização de agentes, que realizam coleta dos dados de forma distribuída e os disponibiliza para serem processados por um analisador central que possui uma visão global da rede. O uso de sistemas especialistas baseados em regras de produção permite a correlação entre eventos gerados nos sensores e o estado ativo da rede. A arquitetura flexível possibilita a utilização de ferramentas já consagradas, aproveitando sua maturidade e agregando valor ao sistema.

Testado em redes de grande porte o sistema mostrou-se robusto, produzindo resultados satisfatórios tanto na redução do número de falsos positivos quanto no fornecimento de métricas relacionadas a ameaças e vulnerabilidades. Essas métricas podem, posteriormente, ser utilizadas como fontes para análise de risco.

Palavras Chave: Sistemas de Detecção de Intrusões; Sistemas de Prevenção de Intrusões; Inventário de Rede; Correlação de Dados; Sistemas de Produção; Sistemas Proativos.

Abstract

Protecting a computer system involves a set of security procedures followed by continuous monitoring. Intrusion Detection Systems (IDS) have historically been used to monitor computer networks against threats. Recently the IDS's have evolved to allow real-time protection against attacks and have been marketed as Intrusion Prevention Systems (IPS). These systems are essentially reactive and are limited in their analyses because they lack a global view of the monitored network.

In this work we present a solution to the continuous monitoring problem called **Proactive Intrusion Prevention Systems (PIPS)**. The system works in a proactive way, building an active view of the network using port scanners and vulnerability assessment tools. Besides providing a detailed vision of the network state, this profile is used for correlations with IDS events resulting in more refined analyses. PIPS is built upon distributed agents, that permanently collect data and deliver it to a central analysis system that keeps a global view of the network. The use of expert systems based on production rules allows the correlation between events gathered by the sensors and the network state. A flexible architecture, using plug-ins, allows the use of well-tested tools adding value to the system.

Thoroughly tested in production environments the system delivered satisfactory results both in reducing false positives and providing metrics for threats and vulnerabilities. These metrics can further be used as real world sources of data for risk analysis.

Key words: Intrusion Detection Systems (IDS); Intrusion Prevention Systems; Network Inventory; Data Correlation; Production Systems; Proactive Systems.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Segurança de Sistemas

A infra-estrutura sobre a qual a Internet cresceu não foi projetada com segurança como um requisito fundamental. Pelo contrário, a funcionalidade e a robustez sempre foram os objetivos principais. A preocupação com a segurança dos sistemas cresceu proporcionalmente à sensibilidade das informações trocadas através da grande rede.

Alguns eventos ao longo da curta história da Internet contribuíram para aumentar a consciência geral da necessidade do desenvolvimento de sistemas e redes mais seguros. Entre eles vale destacar:

- (1979) Primeiro Estudo de Vulnerabilidades em Senhas: nesse estudo Morris e Thompson [1] demonstraram que deduzir senhas é bem mais efetivo que decifrá-las. Foi mostrado que um percentual significativo das senhas estudas podiam ser deduzidas a partir dos nomes de usuários, seus endereços, números de identidades, telefones e outros dados pessoais publicamente acessíveis.
- (1988) "The Internet Worm": o primeiro grande incidente de segurança na Internet [2]. Robert T. Morris (filho do co-autor de "Primeiro Estudo de Vulnerabilidades em Senhas") escreve e inicia a propagação do código auto-replicável que invadiu cerca de 5% dos computadores conectados à ARPANET causando lentidão de processamento e significativo congestionando tornando a então rede acadêmica praticamente inutilizável por alguns dias. Aliado à invasão de máquinas da MILNET (rede militar norte americana) esse evento levou à criação, poucos dias depois, do CERT (*Computer Emergency Response Team*).
- (1994) Kevin Mitnick x Tsutomu Shimomura: no natal de 94 Mitnick, invadiu o computador pessoal do especialista em computação Shimomura utilizando as técnicas, até então pouco utilizadas, de *IP-Spoofing* e previsão de números de seqüência TCP. Desencadeou-se então uma caçada cinematográfica (contando com o auxílio de Shimomura) que culminou na prisão de Mitnick. A história obteve ampla divulgação na mídia resultando na publicação de três livros (*Takedown, The Fugitive Game, The Cyberthief and The Samurai)* dois filmes (*Takedown e Freedom Downtime*) além de inúmeros artigos em grandes jornais.
- (2003) *Slammer/Sapphire*: Explorando uma vulnerabilidade conhecida do servidor SQL da Microsoft, foi o *worm* que mais rápido se espalhou na história [3]. O número de máquinas infectadas dobrou a cada 8.5 segundos após o início da propagação. Mais de 90% das máquinas vulneráveis foram infectadas em pouco mais de 10 minutos. Ao todo mais de 75 mil máquinas foram contaminadas. Contendo apenas 376 bytes de código e usando UDP para transporte esse *worm* causou problemas antes inimagináveis como cancelamento de vôos, interferência em eleições e mal-funcionamento de caixas eletrônicos.

Ao longo desses anos o interesse acadêmico e comercial em métodos de tornar os sistemas computacionais mais seguros tem crescido rapidamente. Segurança de Sistemas atualmente está se consagrando como uma disciplina fundamental dentro da computação. A ISC^2 (*International Information Systems Security Certifications*

Consortium), órgão que gerencia as certificações de maior reconhecimento na área subdividiu os conhecimentos dessa disciplina em dez grandes tópicos que constituem o Corpo Comum de Conhecimentos (CBK) da área [4]:

1. Metodologias e Sistemas de Controle de Acesso
   Controle de acesso é o conjunto de mecanismos que permite aos administradores de um sistema direcionar ou restringir o comportamento, uso e conteúdo desse sistema. Ele possibilita a especificação do que os usuários podem fazer, quais recursos eles podem acessar e quais operações eles podem executar no sistema.
   Um sistema razoavelmente seguro deve garantir que os usuários e processos só têm acesso aos recursos indispensáveis para o seu bom funcionamento. Registros de comportamentos anômalos devem ser gerados fornecendo material para a contínua monitoração da boa utilização do sistema. Testes periódicos devem ser realizados para validar a política de controle de acesso.

2. Segurança em Redes e Telecomunicações.
   Preocupa-se com as estruturas, métodos de transmissão, formatos de transporte e medidas de segurança adotadas para prover integridade, disponibilidade, autenticação e confidencialidade das transmissões sobre meios de comunicação públicos ou privados. Cada uma das camadas do modelo de referência OSI possui problemas de segurança associados a ela. Garantir a segurança de uma rede envolve compreender todos os riscos envolvidos e diminuir esse risco para um nível aceitável, através da utilização de sistemas computacionais especializados como *firewalls* e *proxies*, emprego de protocolos mais seguros como SSH, TLS e S/MIME e técnicas de tunelamento, redes privadas virtuais (VPN) e tradução de endereços de rede (NAT). Monitores de rede e "*sniffers*" [5] permitem às equipes de administração de redes tomarem plena ciência do que trafega em suas redes, identificando comportamentos perigosos e/ou suspeitos.

3. Gerenciamento de Segurança
   A gerência de segurança engloba a identificação dos recursos de informação da organização e o desenvolvimento, documentação e implantação de políticas, padrões e procedimentos que garantam confidencialidade, integridade e disponibilidade. Ferramentas de gerenciamento, classificação de dados, avaliação e análise de riscos são utilizadas para identificar as ameaças, classificar os recursos e suas vulnerabilidades de forma que controles efetivos de segurança possam ser utilizados. Infelizmente a maior parte das organizações realiza a análise de riscos baseando-se em estudos teóricos e/ou estatísticas externas. Idealmente a análise deveria usar dados reais colhidos recentemente por ferramentas internas de monitoração.

4. Desenvolvimento de Sistemas e Aplicações
   As aplicações são fontes inesgotáveis de vulnerabilidades. Equipes de desenvolvimento de sistemas internos e homologação de sistemas de terceiros devem estar familiarizadas com os principais métodos de ataque a aplicações e as contra-medidas necessárias para dificultar esses ataques. Segundo pesquisa realizada entre 2000 e 2004 pela Imperva[1], empresa de destaque em soluções de segurança para aplicações, apenas 10% das aplicações *web* estão seguras contra

[1] http://www.imperva.com

técnicas comuns de invasão. Oitenta por cento das aplicações testadas estavam vulneráveis a *Cross-site Scripting [6]*, 62% delas suscetíveis a injeção de comandos SQL e 60% sujeitos à manipulação de parâmetros. Práticas de desenvolvimento seguro e metodologias de testes ajudam a tornar aplicações mais seguras. Além de serem resistentes a tentativas de invasão elas devem prover um bom nível de auditabilidade através da geração de registros consistentes e, sempre que possível, integrando-se aos sistemas de gerenciamento de segurança adotados na corporação.

5. Criptografia

   Provavelmente a disciplina mais antiga relacionada à segurança de sistemas. Povos antigos já utilizavam técnicas de substituição e transposição para tornar suas escritas menos inteligíveis por seus inimigos. Durante muito tempo viveu-se um período de grande evolução onde novas técnicas criptográficas eram inventadas e quebradas sucessivamente. Hoje em dia, a qualidade da criptografia é medida pelo tempo computacional necessário para quebrar uma determinada cifra, sendo possível prever quando um algoritmo estará obsoleto e possibilitando ajustes nos algoritmos e principalmente nos tamanhos das chaves criptográficas utilizadas.

   Criptografia é a principal ferramenta utilizada para dificultar o acesso a dados sensíveis, seja em trânsito ou armazenados em mídia. Os principais serviços alcançados com o seu uso são:

   - segredo – certeza que os dados não serão acessados sem autorização;
   - autenticidade – garantia de verificação da origem da informação;
   - integridade – segurança de que a mensagem não foi modificada no caminho;
   - não-repúdio – impede quem envia de negar a origem da mensagem.

6. Modelos e Arquitetura de Segurança

   Nem sempre é fácil classificar um sistema quanto ao seu nível de segurança. Corporações em busca de certificações internacionais normalmente têm que se adequar a modelos clássicos como os de controle de acesso definidos por Bell-LaPadula, Clark-Wilson ou Biba [7]. Buscando normalizar os critérios utilizados para classificação de sistemas operacionais e dispositivos computacionais foram criados alguns padrões, entre eles: ITSEC, TCSEC (“*Orange Book*”) *[8]* e o *Common Criteria*.

   Embora o foco principal desses modelos seja o controle de acesso e os níveis de proteção das informações, um dos critérios considerados é a auditabilidade, ou seja, a capacidade dos sistemas avaliados de gerar registros das ações realizadas por seus usuários permitindo a reconstrução posterior de todos os eventos ocorridos em um determinado intervalo de tempo.

7. Segurança de Operações

   Operações de computadores engloba tudo que é feito no dia-a-dia para manter um sistema computacional funcionando. Isso inclui tanto administração de sistemas quanto tarefas externas aos sistemas, tais como a manutenção da documentação.

   O principal item relacionado à segurança de operações é a separação de papéis na administração, operação e manutenção dos sistemas. Idealmente deve-se garantir que funcionários e usuários só possuam acesso aos recursos essencialmente necessários para a execução de suas tarefas rotineiras. Isso normalmente é

garantido através do princípio de privilégios mínimos e enfatizado através da contínua monitoração.

8. Plano de Continuidade de Negócios
   Também faz parte da segurança de uma organização prover os meios adequados para garantir que suas operações técnicas e comerciais terão continuidade no evento de um desastre ou interrupção inesperada.
   O plano de contingência deve levar em consideração, no mínimo as etapas de:
   - identificação de funções críticas para a missão do negócio;
   - identificação de recursos que suportam essas funções críticas;
   - antecipação de riscos de desastres potenciais;
   - seleção de estratégias de planejamento de contingência;
   - implementação dessas estratégias;
   - testes e revisão do plano.

9. Leis, investigações e ética
   No decorrer de suas atividades, as organizações devem sempre ter o cuidado de não infringir leis e regulamentações locais, nacionais e internacionais. Normalmente essa preocupação não faz parte da formação das equipes técnicas e administrativas, obrigando as organizações a depender de consultorias externas que nem sempre estão preparadas para interagir com as particularidades e operações técnicas da organização.
   Um fator importante do ponto de vista técnico é desenvolver procedimentos de rastreabilidade que permitam eventuais investigadores, internos ou externos, refazer todos os passos que levaram a um eventual incidente.

10. Segurança Física
    São as medidas tomadas para proteger os sistemas, prédios e infra-estrutura de suporte contra ameaças associadas com o meio-ambiente físico. Os riscos incluem interrupção de serviços, danos físicos, vazamento não autorizado de informações sensíveis, perda de integridade e roubo.
    Dependendo da natureza das atividades da organização, os sistemas de controle físicos podem ser interligados com os sistemas de controle lógicos, provendo uma interface unificada e consistente de identificação de ameaças, sejam elas físicas ou digitais.

Idealmente o processo de tornar um sistema computacional mais seguro deve levar em consideração, com diferentes graus de importância, todos esses tópicos. É interessante notar que as medidas associadas a quase todas essas sub-áreas só são completamente efetivas quando combinadas a procedimentos de monitoração contínua.

De acordo com a ICSA [9], empresa afiliada do Gartner Group o processo de gerenciamento de segurança de sistemas deve levar em consideração políticas e procedimentos da corporação, arquitetura de segurança, treinamento, tecnologias e monitoração segundo a hierarquia mostrada no diagrama da Figura 1. O uso de ferramentas e produtos (camada 4) e a efetiva monitoração, auditoria e investigação (camada 5) só faz sentido quando precedidas das três camadas inferiores que abrangem definições de critérios menos técnicos e mais administrativos e gerenciais.

Figura 1 - Hierarquia de Segurança da Informação

Muitas empresas sofrem prejuízos ao concentrar seus investimentos e esforços na implantação de ferramentas e produtos sem se preocupar com os outros aspectos definidos nas camadas inferiores.

Figura 2 - Mercado de Segurança da Informação

A Figura 2 mostra uma visão geral do mercado de segurança da informação, destacando no centro os principais tipos de produtos na área e, aos lados, os serviços de gerenciamento, administração e consultorias.

1.2 Detecção de Intrusão

Conforme visto na seção anterior a monitoração contínua é um processo essencial para a manutenção da segurança dos sistemas computacionais. Neste contexto surgiram as ferramentas de detecção de intrusão: sistemas que procuram perceber possíveis tentativas de invasão através da monitoração da rede e dos sistemas computacionais em busca de tentativas de intrusões. Intrusões, por sua vez, podem ser entendidas como tentativas de comprometer a confidencialidade, integridade e/ou disponibilidade dos sistemas ou de transpor os mecanismos de segurança implantados para proteger esses sistemas. As intrusões podem ser causadas por atacantes

acessando os sistemas através da Internet, usuários autorizados em busca de elevação de privilégios ou efetuando o mau uso dos privilégios a eles fornecidos.

O modelo CIDF ("*Common Intrusion Detection Framework*") *[10]* define de forma genérica os componentes básicos de um sistema de detecção de intrusão divididos em quatro caixas identificadas pelas letras A,C,D e E que representam respectivamente **A**nálise, **C**ontra-medidas, armazenamento de **D**ados e geração de **E**ventos. O módulo de geração de eventos ("Caixa E") é responsável pela coleta das informações da fonte de dados brutos. Nas "Caixa A" ocorrem as análises baseadas nos eventos gerados. Os eventos e os resultados das análises são guardados nos módulos de armazenamento ("Caixa D"). Ações geradas como conseqüência das análises e dos eventos são de responsabilidade dos componentes de contra-medidas ou "Caixa C". Os inter-relacionamentos entre os módulos são representados pelas setas pontilhadas e as saídas esperadas de cada fase são destacadas junto às caixas.

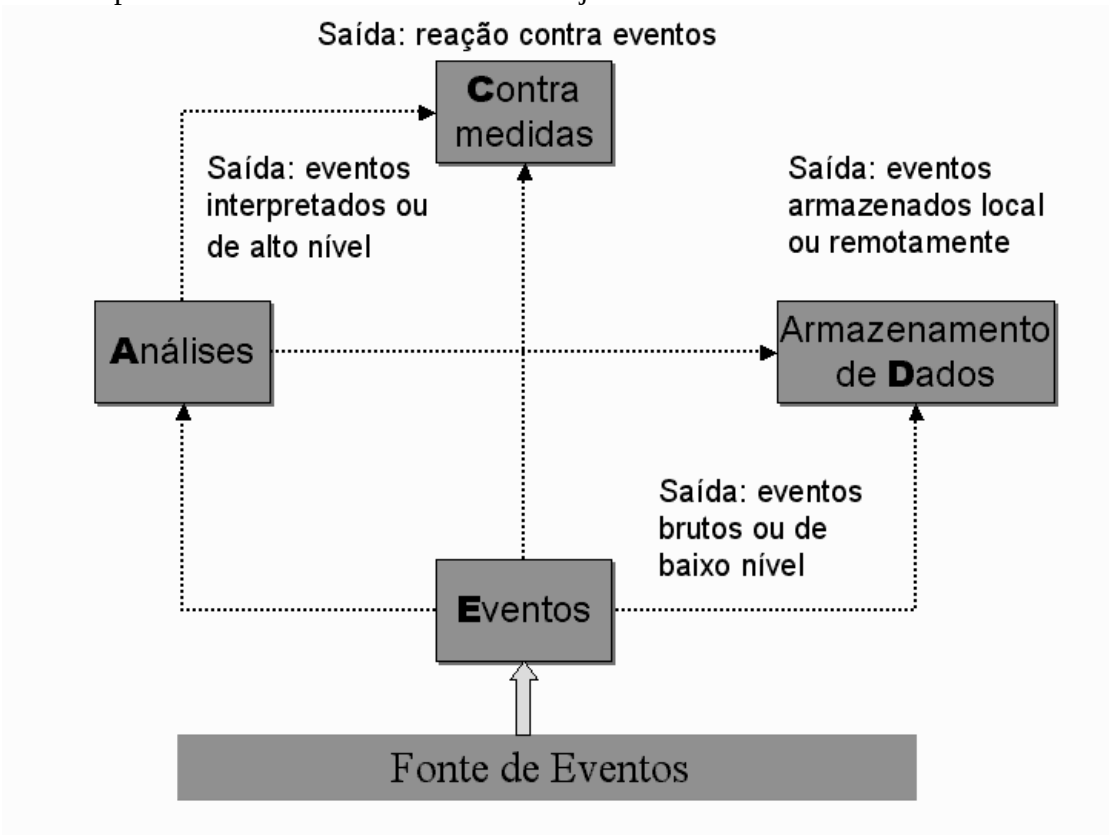


Figura 3 - Modelo CIDF: Common Intrusion Detection Framework

Por ser um modelo abstrato o CIDF pode ser utilizado para caracterizar a maioria dos sistemas de detecção de intrusão existentes. Variações nas fontes de eventos e em alguns dos componentes básicos desse modelo podem ser usadas para classificar os sistemas de detecção de intrusão. É possível, portanto, classificar os sistemas de detecção de intrusão quanto à origem dos dados analisados (fonte de eventos), quanto à periodicidade de geração de eventos, quanto ao tipo de análise e, finalmente, quanto às formas de reação ou contra-medidas [9].

A principal separação entre os tipos de sistemas de detecção de intrusão diz respeito ao tipo de dado analisado. Historicamente esses sistemas têm sido classificados quanto à **fonte de eventos** em IDS de rede ou de *host*. Os **IDS de rede** coletam os pacotes que trafegam na rede monitorada. Normalmente capturam dados

nos pontos de estrangulamento da rede (roteadores ou *firewalls*), seja executando diretamente nesses elementos ou através de cópia de dados, utilizando dispositivos de rede como *hubs* e *switches.* Por coletar tráfego de todos os *hosts* da rede, esses sistemas possuem uma visão mais ampla e podem mais facilmente detectar ataques direcionados a diversas máquinas ao mesmo tempo. Embora possam inferir baseando-se no conteúdo dos pacotes esses sistemas não têm informações sobre o que está de fato acontecendo localmente nos servidores. Já os **IDS de *host*** utilizam informações coletadas nos sistemas operacionais tais como registros de auditoria (logs), saída de programas de monitoração, informações de estado do SO, perfis de execução de programas e de atividades dos usuários. Possuem, portanto, acesso a informações mais detalhadas sobre a atividade local nos sistemas operacionais e podem analisar aplicações e serviços que utilizam criptografia de dados. Por outro lado, esses sistemas possuem uma visão restrita da rede e a disponibilidade e qualidade dos dados por ele analisados dependem diretamente das aplicações que os geram.

Outra distinção clara diz respeito ao **método de análise** utilizado. Os sistemas **baseados em assinaturas** utilizam características comuns de ataques bem-conhecidos para classificar os dados coletados em normais ou maliciosos. Aqueles **baseados em anomalias estatísticas** constroem padrões de uso normal dos sistemas e detectam desvios significativos desses padrões como indicativo de possíveis atividades maliciosas. A principal vantagem dos métodos estatísticos é a possibilidade de detectar ataques previamente desconhecidos e que não possuem assinaturas próprias. No entanto, a caracterização do estado normal dos sistemas e redes é um processo complexo e tem maior tendência à geração de falsos positivos devido a variações causadas por acessos legítimos, porém incomuns. O conhecimento prévio das assinaturas de ataque permite otimizações nas análises, o que normalmente torna os sistemas baseados em assinatura mais precisos e eficientes.

Quanto à **periodicidade** da coleta de dados e geração de eventos os sistemas podem ser divididos em sistemas de **tempo real** onde os dados são coletados e os eventos gerados em paralelo à atividade normal dos sistemas ou de **processamento em lote** onde registros de auditoria das aplicações e sistemas operacionais são periodicamente analisados à procura de atividades suspeitas. O volume de dados trafegados na maioria das redes torna inviável o processamento em lote em IDS de rede que normalmente funcionam em tempo real. Embora menos utilizados os sistemas com processamento em lote são ideais para sistemas onde o tempo adicional de processamento para detecção de intrusões não é aceitável e podem prejudicar substancialmente o funcionamento das outras aplicações. Nesses casos os sistemas só terão capacidade de detectar ataques já ocorridos impossibilitando a reação a ataques enquanto eles estão acontecendo.

Uma caracterização menos freqüente dos IDS diz respeito às **contra-medidas** tomadas quando uma intrusão é detectada. Nos sistemas **passivos** a reação ocorre na forma de geração de alertas para as equipes de administração. Os sistemas **ativos** podem tanto alterar configurações do ambiente, por exemplo, para aumentar a quantidade de informações e registros armazenados quanto atuar diretamente na origem do problema, fechando conexões ou terminando a execução de programas afetados. Em muitos casos, o próprio fato de o sistema reagir de forma ativa pode ser usado em favor do atacante para causar efeitos nocivos ao sistema, eventualmente fechando conexões e programas legítimos resultando em ataques de negação de serviço. Alguns sistemas mais modernos combinam funcionalidades de filtro de pacotes e detecção de intrusão. Maiores detalhes serão vistos na seção 2.4 que trata dos IPS (Sistemas de Prevenção de Intrusões).

A forma como os diversos componentes do sistema são posicionados e como eles se relacionam também permite classificar os sistemas de detecção de intrusão quanto à sua **arquitetura**. Os sistemas que concentram todos os módulos em um único programa são chamados **monolíticos** e os que dividem os módulos em várias máquinas ou programas diferentes são considerados **distribuídos**. Cada uma dessas abordagens possui deficiências que prejudicam a completude e a precisão das análises. Em sistemas distribuídos a presença de um analisador central cria um ponto único de falhas. Se um invasor consegue de alguma forma impedir seu funcionamento, por exemplo, causando negação de serviços na máquina que o hospeda, toda a rede pode ficar desprotegida. A escalabilidade desses sistemas costuma ser limitada. O processamento centralizado numa única máquina impõe um limite no tamanho da rede e na quantidade de dados que podem ser analisados em paralelo. Após esse limite, o analisador central torna-se incapaz de processar o fluxo de informações e tende a descartar informações.

A análise de dados de rede também pode ser burlada. Quando o processamento é feito numa máquina diferente da destinatária final, pode ser dada ao atacante a possibilidade de utilizar técnicas de inserção e evasão [11]. Esses ataques utilizam falhas nas pilhas de protocolos de diferentes sistemas operacionais para esconder os ataques ou causar negação de serviços.

A ausência de correlação entre dados capturados em diversos pontos de análise pode impedir a detecção de determinados tipos de ataque. Algumas poucas tentativas de *logon* mal-sucedidas em um único serviço podem ser consideradas normais e toleradas, porém a repetição dessas falhas em vários servidores da rede pode indicar a tentativa de invasão.

1.3 Objetivos

A tarefa de tornar um sistema computacional mais seguro envolve procedimentos relacionados não só ao sistema em si mas a todo o ambiente que o suporta. Uma falha em qualquer parte da infra-estrutura tecnológica ou operacional do sistema pode levar ao seu comprometimento, possivelmente produzindo uma reação em cadeia que afete todo o ambiente. Uma vez alcançado um nível de segurança esperado, é necessário mantê-lo. Para essa finalidade é utilizada uma miríade de ferramentas de segurança, entre elas: *firewalls*, detectores de intrusão, utilitários para varredura de redes e levantamento de vulnerabilidades. Embora possuam funcionalidades complementares e várias interseções no seu escopo de atuação, os dados produzidos por essas ferramentas normalmente são semanticamente incompatíveis e raramente correlacionados.

Nesse trabalho propomos uma nova abordagem para a manutenção de segurança de redes de computadores: um sistema proativo formado por agentes distribuídos na rede agindo como coletores de informações e sensores de eventos. Os dados coletados nos agentes são enviados para um sistema central que processa esses dados, realiza uma normalização semântica das informações coletadas e as armazena de forma estruturada em bancos de dados relacionais.

Uma vez armazenados esses dados podem ser analisados propiciando a correlação entre dados colhidos em diversas fontes e por ferramentas distintas. O sistema é extensível, permitindo a contínua incorporação de novas ferramentas através de adaptadores dinamicamente acopláveis aos agentes. O processo de análise é realizado

na central de processamentos, utilizando sistemas especialistas baseados em regras de produção [35].
De maneira a permitir a contextualização do leitor, iniciamos essa dissertação com uma visão geral dos aspectos envolvidos na segurança de sistemas computacionais e, em particular na área de monitoramento e detecção de intrusões. Em seguida, apresentamos o estado da arte na área de detecção de intrusões, destacando a utilização de sistemas distribuídos e de técnicas de Inteligência Artificial. Partindo dessa base fundamental, apresentamos a proposta e implementação de um sistema proativo de prevenção contra intrusões (PIPS), enfatizando os resultados alcançados até agora com a utilização do sistema em redes reais.

1.4 Estrutura da Dissertação

No Capítulo 2 serão estudados alguns dos principais sistemas de detecção de intrusão divididos em três categorias: os sistemas de código aberto mais comumente utilizados, as principais propostas e implementações de sistemas de detecção de intrusão distribuídos e os principais trabalhos relacionados à utilização de agentes inteligentes para detecção de intrusão. As principais deficiências das ferramentas e técnicas apresentadas também são destacadas no final deste capítulo.
No Capítulo 3 apresentamos o PIPS: uma nova abordagem para prevenção de intrusões baseada em uma postura de monitoração pró-ativa aliada à correlação entre informações geradas por ferramentas consagradas e o estado ativo da rede. Nesse capítulo também são apresentados os requisitos, os componentes e a arquitetura de um sistema (também denominado PIPS) que implementa os conceitos propostos através da utilização de agentes distribuídos para coleta de dados e uma central de processamento onde os dados coletados por esses agentes são correlacionados entre si e com outras informações armazenados no inventário de rede.
A implementação do PIPS é descrita no Capítulo 4. Nele são detalhadas as implementações dos agentes e da central de processamentos, justificando as escolhas realizadas e detalhando as especificidades de cada componente. Destaque especial é dado para as análises e correlações realizadas na central de processamentos.
No Capítulo 5 são resumidos os principais resultados obtidos com o desenvolvimento do PIPS com destaque para os resultados práticos alcançados com a utilização do sistema em redes reais de médio e grande porte.
Concluindo a dissertação, o Capítulo 6 destaca as principais contribuições do trabalho e os próximos passos no desenvolvimento do PIPS. Na última seção são tecidas as considerações finais a respeito do trabalho e de sua contribuição para a área de segurança de sistemas e detecção de intrusões.

2 Sistemas Existentes

Neste capítulo serão examinados os principais sistemas de detecção e propostas de aperfeiçoamento através do uso do sistemas distribuídos e agentes inteligentes. Inicialmente serão mostrados os sistemas tradicionais mais constantemente utilizados. Em seguida serão detalhados alguns protótipos e implementações de sistemas de detecção de intrusão distribuídos e utilizando agentes inteligentes. Também será discutido o conceito de IPS, mais recente tendência na área de detecção de intrusão. Por fim serão expostas as principais limitações dos sistemas existentes e que motivam o desenvolvimento desse trabalho.

2.1 Sistemas Tradicionais

2.1.1 Snort

O *Snort* [12] é uma ferramenta de detecção de intrusão de rede baseada em assinaturas, que funciona em várias plataformas e pode ser usado para monitorar redes TCP/IP e detectar uma grande variedade de tráfego de rede suspeito e ataques explícitos. Ele pode prover aos administradores dados suficientes para tomar decisões sobre a atitude a ser tomada em face às atividades suspeitas.

A grande vantagem do *Snort* em relação a muitos produtos comerciais é a velocidade com que assinaturas para novos ataques são lançadas. Por contar com uma grande quantidade de usuários em todo o mundo, muitos dos quais capazes de desenvolver rapidamente assinaturas, o *Snort* consegue se manter atualizado em relação aos problemas de segurança, enquanto os fornecedores de produtos comerciais tendem a demorar na publicação de suas atualizações.

O seu funcionamento interno é baseado na *libpcap*[2], uma biblioteca de captura de pacotes amplamente utilizada, desenvolvida inicialmente para ser usada no programa *tcpdump*. Sua principal característica é a utilização de regras para o casamento de padrões com o conteúdo dos pacotes com o objetivo de detectar ataques direcionados as principais classes de vulnerabilidades conhecidas. O engenho de detecção é programado utilizando uma linguagem (regras) que descreve testes e ações a serem realizados para cada pacote capturado.

A principal fraqueza do *Snort* é grande quantidade de alarmes falsos tipicamente emitidos ao utilizar bases de assinaturas genéricas. Quanto mais refinada e adequada à rede monitorada for a base de regras, menor o número de falsos positivos gerados, porém mais complexa se torna sua manutenção. Novas regras são disponibilizadas com grande freqüência e sua inserção em bases de assinaturas altamente personalizadas constitui uma tarefa não trivial.

2.1.2 Tripwire/AIDE

Uma classe importante de detectores de intrusão é a dos verificadores de integridade de arquivos. Seu funcionamento baseia-se na geração de uma base de assinaturas contendo *hashes* de um subconjunto do sistema de arquivos. Em execuções subseqüentes os arquivos são verificados comparando os *hashes* atuais com os da base

[2] Libpcap/Tcpdump: http://www.tcpdump.org/

de assinaturas. Diferenças detectadas geram, entre outros, os eventos de remoção, adição e modificação de arquivos.

O primeiro verificador de integridade amplamente utilizado foi o *Tripwire*, desenvolvido por Gene Kim e Gene Spafford no CERIAS (*Center for Education and Research in Information Assurance and Security*) da Universidade Purdue. Inicialmente esse produto foi distribuído livremente até sua versão 1.2. Em 1997 a fundação de pesquisas de Purdue (que possuía direitos sobre o código) licenciou os direitos comerciais para a empresa de Gene Kim, que mais tarde veio a ser chamada de Tripwire Inc.

Descontente com a nova licença do Tripwire e notando a ausência de competidores à altura, Rami Lehti's escreveu o AIDE (*Advanced Intrusion Detection Environment*) [13] que rapidamente evoluiu como um ótimo substituto do *Tripwire* passando a ser adotado pela maioria das corporações.

2.2 IDS Distribuídos

Muitas das limitações dos sistemas tradicionais podem ser minimizadas pela utilização de características presentes em sistemas distribuídos:

- *Tolerância a Falhas*: o sistema deve ser capaz de continuar funcionando caso alguma parte dele apresente falhas, sejam elas causadas por acidente ou por eventuais ataques ao sistema.
- *Escalabilidade*: novas partes do sistema devem ser facilmente adicionadas sem haver a necessidade de se reconfigurar e reiniciar todo o sistema. Alterações em partes específicas do sistema podem ser realizadas de forma localizada sem afetar as outras partes.
- *Distribuição de carga*: como parte do processamento é realizado em cada um dos servidores da rede monitorada pode-se evitar a sobrecarga característica de detectores centralizados.
- *Correlação de dados*: a troca de mensagens entre as diversas partes do sistema possibilita a análise global dos eventos ocorridos na rede monitorada.

Nessa seção serão mostradas algumas propostas teóricas e implementações práticas de protótipos de sistemas de detecção de intrusão que aplicam técnicas de computação distribuída para a análise, classificação, caracterização, agrupamento e respostas a eventos considerados intrusivos.

2.2.1 The DIDS prototype

O *Distributed Intrusion Detection System* (DIDS) [14] monitora redes locais e os vários *hosts* conectados na rede. Seus componentes incluem o controlador DIDS um único monitor para cada máquina da rede e um monitor de rede para cada segmento de rede monitorado.

As informações coletadas por esses componentes distribuídos são transportados e analisados numa localização centralizada através de um sistema especialista que é um sub-componente do controlador. Desta forma ele provê a capacidade de agregar

informações de diferentes origens. Qualquer informação de auditoria pode ser analisada desde que se especifique os eventos de interesse.

O DIDS é projetado para operar em ambientes heterogêneos composto de computadores classificados como C2 [8] ou superiores. A exigência de sistemas classificados como C2 deve-se a necessidade de consistência do conteúdo dos registros de auditoria, permitindo a construção de representações padrão dentro das quais pode-se mapear registros de UNIX, VMS ou qualquer outro sistema com as capacidades de auditoria exigidas pelo C2. Essa exigência também garante a integridade e segurança dos registros de auditoria, como parte da TCB (Trusting Computing Base).

Sistemas que não se adequarem ao C2, apesar de não poderem ser diretamente monitorados pelos monitores de *host*, suas informações de rede ainda podem ser coletadas pelo monitor de LAN. Como todo o tráfego de rede deve passar pelo monitor de LAN quaisquer ataques direcionados a esses sistemas poderão ser detectados.

O monitor de *host* é composto de dois componentes: o Gerador de Eventos de *Host* (GEH) e o Agente de *Host*. O GEH coleta registros de auditoria gerados no sistema operacional. Esses registros são analisados a procura de "*eventos notáveis"* que são transações interessantes independente de outros eventos. Entre elas pode-se incluir falhas de autenticação, alterações no estado de segurança do sistema e acessos de rede tais como *rsh* ou *rlogin*. Esses eventos são enviados para o controlador para analises posteriores. Encontram-se em desenvolvimento melhorias no GEH para detectar alterações significativas no comportamento dos usuários através da análise de suas sessões ao longo do tempo. Toda a comunicação entre o monitor de *host* e o controlador é feita através do agente de *host*.

De forma semelhante, o monitor de rede consiste de um Gerador de Eventos de Rede (GER) e de um Agente de LAN. Sua principal responsabilidade é observar todo o tráfego no seu segmento de rede, monitorar comunicações de entre *hosts* tais como sessões de telnet ou *rlogin*, uso de serviços relacionados a segurança e mudanças nos padrões de tráfego da rede.

O controlador é formado por três processos independentes que podem estar posicionados num mesmo servidor central ou distribuídos. O Gerenciador de Comunicações é responsável pela transferência de dados entre o controlador e os monitoradores de *host* e de LAN e enviá-los para o sistema especialista, além de também enviar eventos gerados pelo sistema especialista ou da interface do usuário para os monitoradores solicitando maiores informações sobre um determinado acontecimento. O sistema especialista no DIDS é um sistema baseado em regras com pouca capacidade de aprendizagem. Ele é responsável por avaliar e gerar relatórios sobre o estado de segurança do sistema monitorado. Ele recebe registros dos vários monitoradores e faz inferências sobre a segurança de cada nó da rede, assim como sobre o estado da rede como um todo.

2.2.2 GRIDS

Protótipo desenvolvido por estudantes da Universidade da Califórnia o GRIDS (*Graph Based Intrusion Detection System*) [15] constrói e mantêm grafos baseados em tráfego coletado nas redes monitoradas tentando correlacioná-los com definições de grafos representativos de ataques bem sucedidos à essas redes.

Alguns tipos de ataque possuem padrões peculiares de tráfego, que quando analisados de forma global podem ser modelados utilizando grafos. Por exemplo, a proliferação de um *worm* normalmente segue a estrutura de árvore: uma máquina é inicialmente infectada e passa a gerar um determinado tipo de tráfego para suas adjacentes. Essas, uma vez infectadas, fazem o mesmo infectando novas máquinas.

Além dos nós (os membros da rede) e das arestas (o tráfego entre eles) os grafos do GRIDS podem conter outras informações significativas, por exemplo um determinado tipo de *worm* pode ser caracterizado não só pelo tipo de conexão que ele efetua com outras máquinas como também pelo tipo de dados transferidos nessas conexões. Essas informações são associadas aos nós ou as arestas e um grafo só é considerado suspeito quando além de sua topologia característica também apresenta comportamentos descritos nas meta-informações.

A rede monitorada é dividida e organizada segundo a metáfora de uma empresa que contém vários departamentos que podem conter sub-departamentos ou indivíduos (os *hosts* da rede). Vários grafos são construídos dentro de cada departamento e reduzidos para formar grafos maiores entre departamentos. Toda a rede pode então ser visualizada como um conjunto de grafos que são formados pela redução de outros grafos em seus nós. Para não haver perda de informações as reduções podem gerar meta-dados para os grafos maiores, tais como número e tipo de grafos naquele departamento e características das redes desses departamentos.

Nem todo par de atividades de rede é suficientemente relacionado para pertencer ao mesmo grafo. Vários grafos contendo diferentes padrões de comportamento são mais fáceis de analisar do que um grafo grande representando toda a informação. O engenho de grafos é capaz de manter múltiplos espaços de grafos de acordo com seus tipos. O tipo de um grafo é determinado pelas regras que descrevem como criá-los.

As regras de formação dos grafos são definidas numa linguagem própria. As entradas e saídas dessas regras são grafos em si e o seu processamento pode gerar a criação de novos grafos de estado que podem ou não servir de entrada pra outras regras. Cada regra possui uma ou mais pré-condições que dizem respeitos a determinadas características da entrada. Essas características podem ser relativas a propriedades dos nós da entrada, das arestas ou dos grafos de entrada como um todo.

Embora a especificação do GRIDS preveja a utilização de vários tipos de entradas de dados o protótipo implementado utiliza apenas um tipo de monitor baseado no programa de domínio público *tcpdump*. O *sniffer* examina dados brutos na rede analisada e relata o estado dos canais de comunicação entre entidades do sistema (usuários, máquinas, programas, etc) para seus agregadores. Esses analisam os relatos e detectam padrões de comunicação que possam representar mal-uso e possam ser úteis para a criação de grafos.

A arquitetura do GRIDS é constituída de conjuntos de controladores de módulos responsáveis por analisar dados de um determinado conjunto de agregadores ou de outros módulos de outros departamentos. A comunicação entre diferentes módulos pode ser realizada através de mensagens definidas numa API própria e transportadas em conexões TCP ou UDP. Cada mensagem tem formato bem definido utilizando cabeçalhos e regras de formatação de dados definidas na gramática do sistema.

O protótipo foi desenvolvido como prova de conceito e possui diversas deficiências que o impede de ser utilizado em ambientes de produção. Quesitos de design como tolerância a falha e segurança nas comunicações foram deixados de lado em prol da simplicidade.

2.2.3 EMERALD

EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances) [16] é um ambiente para detecção de anomalias de mau uso e subseqüente análise do comportamento de sistemas e redes. Seus objetivos incluem detecção em tempo real, análises e repostas a ameaças internas ou externas que tentam abusar de recursos de sistemas ou redes.

O ambiente combina componentes de análise estatísticas e baseadas em assinaturas com um processador que interpreta os resultados das análises, que podem ser usados interativamente e hierarquicamente. Seus módulos são projetados para serem independentemente úteis, dinamicamente implantáveis e amplamente interoperáveis. Seus objetivos incluem alcançar habilidades analíticas inovadoras, rápida integração com ambientes de rede existentes e grande flexibilidade de vigilância quando alterações de configuração de redes ocorrem.

A arquitetura do EMERALD é baseada na criação de blocos construtivos utilizando monitores independentemente configuráveis que podem detectar e responder a atividades maliciosas de forma local e podem interoperar para criar uma hierarquia de análise. Existem três tipos básicos de unidades de análise: analisadores de perfil, engenhos de assinaturas e resolvedores, todos focados em objetos relacionados ao recurso analisado. Existe também a possibilidade de integração com módulos de terceiros, tanto como entrada de dados, como saída de análises para outras plataformas ou para administradores.

Um monitor é dinamicamente implantado em um domínio administrativo para prover análise localizada e em tempo real da infra-estrutura (roteadores ou *gateways*) e serviços (subsistemas privilegiados com interfaces de rede). Um monitor pode interagir com seu ambiente de forma passiva (lendo registros ou pacotes da rede) ou ativa (através de coletas que completam os eventos normais). Ao produzirem resultados analíticos os monitores são capazes de disseminar esse conhecimento de forma assíncrona para outros monitores clientes. Esses últimos podem operar em determinados domínios correlacionando resultados dos monitores de serviços ou em camadas superiores analisando os resultados de vários domínios. Sob esse *framework*, uma hierarquia de análises em camada pode ser formada para suportar o reconhecimento de ameaças entre diferentes domínios incluindo tentativas coordenadas de infiltração ou destruição da conectividade numa grande rede.

No entanto o EMERALD não requer que seus monitores operem em conjunto. Cada monitor tem autonomia suficiente para realizar análises localizadas e possui capacidades de tomadas de decisão e interface com outros módulos (possivelmente de terceiros). A semântica de análise foi retirada do código base (*core*) dos monitores possibilitando a criação desde monitores simples e leves até a plataformas de análise mais completas e centralizadas.

Num determinado ambiente, os monitores de serviços podem ser independentemente distribuídos para analisar a atividade de múltiplos serviços de rede (FTP,SMTP, HTTP, etc) ou elementos de rede (roteador, *firewall*).

O núcleo do código de um monitor é independente do tipo de análise a ser realizado. Ao mudar de um ambiente para outro a única coisa que precisa ser modificada é o objeto de recurso. Um exemplo de objeto de análise é o de FTP que possui regras para detectar falhas de autenticação, estouros de pilhas e ataques do tipo *SYN flood* (ataque de negação de serviço que utiliza características do protocolo TCP para exaurir espaço de memória alocado para essa finalidade).

Os objetos de recurso atendem aos objetivos principais do projeto: reusabilidade e integração a novos ambientes. A idéia é criar uma biblioteca populada de objetos de recursos construídos para analisar vários serviços e elementos de rede. Os usuários receberão o código base dos monitores e poderão baixar os objetos apropriados para o tipo de análise que eles irão realizar.

Novas técnicas de correlação de análises e gerenciamento de serviços analíticos não chegaram a ser desenvolvidas, ficando como trabalhos futuros. O conceito de vigilância composta permitirá a agregação de análises de monitores independentes num esforço para isolar características comuns ou tendências em seqüências de alarmes que podem indicar uma ameaça global mais séria. Tais análises agregadas podem ser classificadas em quatro categorias gerais:

- Detecção de características comuns*:* envolve a busca por indicadores comuns de alarmes gerados em análises de eventos independentes. Nesses casos a análise pode ocorrer abaixo de limites que garantam a certeza de ataques, mas que em conjunto com resultados de outras análises podem representar um risco mais sério numa perspectiva global. Essa abordagem pode detectar ataques lentos e distribuídos, ataques cooperativos e efeitos de proliferação de contaminações.
- Análise de multi perspectiva: refere-se ao esforço de analisar o mesmo recurso de múltiplas perspectivas (por exemplo a análise dos registros de auditoria de um servidor *web* em paralelo a monitoração do tráfego de rede direcionado a ele).
- Correlação de alarmes: o EMERALD pode vir a ter a habilidade de detectar a correlação (causa e efeito) entre a ocorrência de alarmes em origens independentes. Por exemplo: um alarme sobre uma atividade detectada em um *host* de um determinado domínio pode disparar um indicador de uma ameaça a um segundo *host*, possivelmente de outro domínio.
- Tendências seqüenciais: busca detectar padrões em alarmes gerados dentro de ou através de diferentes domínios. Esses padrões de atividade agressiva podem gerar a necessidade de uma resposta global gerando contra-medidas impossíveis em um contexto puramente local.

O projeto EMERALD representa um esforço de combinar as pesquisas em sistemas distribuídos de correlação de eventos em grande escala com a experiência adquirida em décadas de pesquisa em detecção de intrusão. A herança de técnicas bem desenvolvidas e aceitas de detecção de intrusão sendo aplicadas em um *framework* altamente reutilizável, interoperável e escalável. Apesar das ótimas perspectivas teóricas o projeto parou de evoluir e desde 2001 não houveram publicações ou avanços em implementação. As principais evoluções relacionadas a correlações e análises ficaram como projetos futuros e não chegaram a ser implementadas.

2.2.4 AAFID

Autonomous Agents For Intrusion Detection 0 é uma arquitetura para construção de IDS que utiliza agentes como o seu elemento de mais baixo nível de coleta de dados e que aplica uma estrutura hierárquica permitindo a escalabilidade do sistema.
Os componentes principais do AAFID são:

- **Agentes -** uma entidade independente que coleta certos aspectos de um determinado *host* e reporta eventos interessantes para os transceptores. Por exemplo, um agente poderia ficar procurando por um alto número de conexões *telnet* para um determinado host e considerar o acontecimento desse evento como

suspeito. O agente geraria então um alarme para o transceptor. Um agente não tem a autonomia de gerar alarmes diretamente para o usuário. Normalmente um transceptor ou um monitor irá gerar um alarme para o usuário baseado nos alarmes gerados por um ou mais agentes. Analisando os dados relatados por vários agentes os transceptores constroem uma imagem do estado dos *hosts* e os monitores por sua vez recebem informações de diversos transceptores construindo uma imagem do estado das redes por eles monitorados.

Nessa arquitetura os agentes não se comunicam entre si, eles enviam todas as suas mensagens para os transceptores que decidem o que fazer com a informação baseado em configurações de informação dos agentes. Deve-se notar que a arquitetura não especifica nada sobre a complexidade dos agentes, eles podem ser simples contadores que analisam a quantidade de conexões telnet para determinados hosts até um controlador complicado que monitora as chamadas de sistema de um determinado programa construindo perfis de execução e detectando mudanças de padrão. A única exigência dos agentes é que eles produzam suas saídas em formato adequado e as envie para os transceptores. Internamente os agentes podem executar quaisquer funções necessárias, incluindo:

- Os agentes podem evoluir ao longo do tempo utilizando algum algoritmo genético
- Eles podem possuir mecanismos de manutenção de estado ao longo do tempo permitindo detecção de ataques lentos ou de alterações no comportamento de sistemas.
- Os agentes poderiam migrar de um *host* para outro combinando o funcionamento do AAFID com alguma técnica que programação móvel

Os agentes podem ser escritos em qualquer linguagem de programação. As funcionalidades básicas de todo agente (relatórios, sincronização e comunicação) podem ser codificadas em bibliotecas compartilhadas tornando a construção de novos agentes uma tarefa relativamente fácil.

- **Transceptores -** são as interfaces externas de comunicação de cada *host*. Possuem dois papéis: controle e processamento de dados. Todo *host* de uma arquitetura AAFID deve, obrigatoriamente, possuir um transceptor. No seu papel de controle um transceptor efetua as seguintes tarefas:

  - Iniciar, parar e reconfigurar agentes
  - Manter uma tabela com os agentes ativos
  - Responder a comandos do monitor, retornando informações ou realizando as tarefas solicitadas.

  Na parte de processamento ele tem as seguintes funções:

  - Receber relatórios gerados pelos agentes;
  - Realizar o processamento apropriado (análise ou redução) nesses dados;
  - Distribuir as informações dos agentes para outros agentes ou para o monitor apropriado.

- **Monitores -** são as entidades de mais alto nível na hierarquia do AAFID, possuindo também capacidades de controle e processamento (de forma similar aos transceptores) além de poder controlar entidades (transceptores ou outros monitores) rodando em diferentes *hosts* da rede.

  No processamento de informações os monitores possuem acesso a dados de diferentes origens e podem fazer correlações de mais alto nível que os transceptores.

Além disso os monitores são responsáveis pela comunicação com a interface dos usuários provendo para esses um ponto de acesso para toda a hierarquia controlada por eles.

- **Interfaces de Usuários -** a arquitetura claramente separa a interface de usuários dos elementos de processamento e controle. Uma interface necessariamente precisa se comunicar com um monitor para poder requisitar informações do sistema. Essa separação permite a criação de diferentes interfaces de acordo com as necessidades dos usuários. Atualmente a interface existe é bastante simples e baseada em comandos digitados pelo administrador.

O projeto foi validado através da implementação de protótipos. O primeiro protótipo foi montado utilizando diversos pequenos programas em diferentes linguagens de programação e tinha o objetivo de servir como prova dos conceitos propostos. O segundo protótipo foi totalmente reescrito utilizando a linguagem PERL com objetivo de aumentar a portabilidade, implementar a infra-estrutura e disponibilizar a API para o desenvolvimento de novos agentes e definição de mecanismos de comunicação e via troca de mensagens em formato próprio.

Atualmente descontinuados os protótipos da arquitetura deixaram algumas deficiências:

- Os monitores são pontos únicos de falha e seu mal funcionamento pode afetar todas as entidades por ele controladas.
- Não existem mecanismos de sincronização de estado entre possíveis monitores redundantes.
- Não existem mecanismos de controle de acesso ou criptografia de dados.
- A detecção de intrusão no monitor pode ser atrasada pela necessidade de recebimento de dados de diversas fontes. Mecanismos devem ser implementados para impedir que a falha de uma entidade provoque um efeito cascata reverso, ou seja, que uma falha em um simples agente possa causar problemas a todos os monitores responsáveis direta ou indiretamente por ele.

2.3 IDS com Agentes Inteligentes

A maior parte dos sistemas de detecção de intrusão tradicionais é restrita na análise dos dados coletados. A estratégia mais comum é a de coletar informações, fazer um casamento de padrões com regras pré-estabelecidas pelos administradores ou desenvolvedores do sistema e gerar alertas e relatórios quando acontecerem eventos previstos por essas regras. A quantidade de alertas falsos e o volume de informações irrelevantes gerados por esses sistemas sugerem a necessidade de um processamento mais efetivo das informações antes da geração dos alarmes e relatórios.

A utilização de agentes inteligentes tem sido considerada como estratégia para otimizar a análise realizada pelo sistema. Por um lado a utilização de técnicas de mineração de dados ajuda a identificar partes relevantes da grande massa de dados possivelmente coletadas de diversas fontes, tais como *logs*, processos de monitoramento e analisadores de tráfego.

Agentes que possuam a capacidade de aprendizagem podem ajudar na detecção de ataques realizados de formas não previstas ou que não tenham sido estudadas e documentadas previamente. A utilização de redes neurais treinadas com a utilização de ferramentas de ataque bem conhecidas é uma área incipiente de pesquisa.

Nessa seção serão mostradas algumas propostas teóricas e implementações práticas de protótipos de sistemas de detecção de intrusão que aplicam técnicas de inteligência

artificial (IA) para a análise, classificação, caracterização, agrupamento e respostas a eventos considerados intrusivos.

2.3.1 Agentes Inteligentes para Detecção de Intrusão

Proposta de Guy Helmer, Johnny Wong, Vasant Honavar e Les Miller da Universidade do Estado de Iowa para utilização de agentes móveis e mineração de dados e aprendizado de máquina para detecção de intrusão [18]. O esquema abaixo mostra a arquitetura proposta, cujos componentes são detalhados os em seguida:

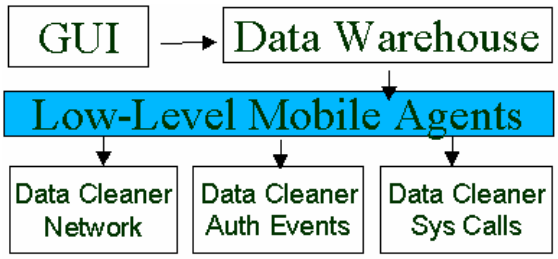


Figura 4 - Inteligent Agents For Intrusion Detection

- *Data Cleaners* – são agentes distribuídos que processam dados obtidos de arquivos de *log*, monitores de protocolos de rede e de atividades de sistemas em sistemas possivelmente heterogêneos.
- Agentes Móveis de Baixo Nível – logo acima dos *data cleaners*, formam o primeiro nível de detecção de intrusão. Utilizando tecnologia de agentes móveis eles deslocam-se até seus respectivos *cleaners* coletam informações recentes e determinam se existem atividades suspeitas ocorrendo. Colaboram entre si para tentar relacionar atividades suspeitas detectadas em fontes distintas.
- *Data Warehouse* – combinam algoritmos de aprendizagem e mineração de dados para descobrir relacionamentos entre eventos suspeitos que ocorrem em conjunto formando um determinado padrão e freqüência.
- Interface do Usuário – permite direcionar a operação dos agentes do sistema e mostram o estado reportado pelos agentes de baixo-nível.

Os únicos componentes efetivamente desenvolvidos foram os *data cleaners* e respectivos agentes móveis de baixo nível de monitoração de chamadas de sistema. Para os experimentos foram utilizadas as chamadas de sistema do servidor de correio eletrônico *sendmail*.

2.3.2 IDS usando Agentes Direcionados por Interesses

Buscando desenvolver sistemas de detecção de intrusão verdadeiramente distribuídos (que não dependam de entidades centralizadas) Rajeev Gopalakrishna e Eugene Spafford da Purdue University, propuseram um *framework* para detecção de intrusão distribuída utilizando agentes cooperativos direcionados por interesses [19].

A tradicional hierarquia de análise é evitada através da realização de toda a detecção e correlacionamento no nível dos agentes. A inteligência cooperativa é atingida através da comunicação de eventos ou alertas apenas àqueles agentes que declaram interesse nesse tipo de dado. Embora a análise não seja hierárquica, os agentes propagam seus interesses de forma hierárquica através de agentes leves. Os

agentes podem explicitar novos interesses em resposta a notificação de alguns eventos, tornando o sistema dinâmico e ativo.

A parte mais elaborada da proposta diz respeito aos tipos de interesse, que são classificados em diretos (comunicados de um agente para outro) ou propagados (através da arquitetura hierárquica), locais, de domínio ou globais, permanentes ou temporários. Também são definidos agentes específicos de registro e propagação de interesses.

A proposta, embora não tendo sido acompanhada de uma implementação de referência, desperta para a possibilidade de distribuição total, tornando o sistema muito mais robusto e extensível. O grupo de trabalho é o mesmo do AAFID e muito da terminologia e componentes daquele sistema são mantidos.

2.3.3 IDS Usando Paradigma de Comunicação de Insetos

"A distributed Intrusion Detection and Response System based on Mobile Autonomous Agents Using Social Insects Communication Paradigm" [20]. Inspirados pela tendência em Inteligência Artificial de tentativa de mímica do comportamento de insetos em sistemas computacionais e no desejo de desenvolvimento de IDS verdadeiramente distribuídos, robustos e reativos os pesquisadores Serge Fenet e Salima Hassas da Universidade Claude Bernard Lyon 1 na França propuseram a criação de um sistema de detecção de intrusão baseado em agentes móveis autônomos utilizando o paradigma de comunicação de insetos sociáveis (principalmente formigas).

O princípio de funcionamento é o seguinte: a detecção de um comportamento suspeito, ou de uma intrusão provoca a liberação através da rede de um "feromônio" de alerta num padrão gradiente. Agentes móveis de resposta – os "linfócitos"-, sensíveis a essas mensagens "químicas" convergem para sua origem e iniciam uma ação defensiva. Para isso são definidos os seguintes componentes da arquitetura:

- Servidores de Feromônio: são os agentes fixos que fornecem uma interface entre a máquina monitorada e os agentes móveis. Após se autenticarem os agentes capturam as informações desse servidor e de seus vizinhos. Por exemplo, um servidor pode codificar sua carga de CPU em formato apropriado análogo ao feromônio nos insetos.
- Observadores: coletam as informações relacionadas à detecção de intrusão e podem ser de host, rede, baseados em assinatura ou comportamento padrão. Dependendo de uma classificação interna de severidade do alerta o observador informa ao servidor de feromônio a intensidade da propagação do alerta.
- Agentes Anti-corpos: a função desses agentes de vida curta é interagir com os servidores de feromônio para espalhar os alertas (em forma de feromônio digital) de forma gradiente, diminuindo de intensidade a medida que se afasta do servidor de origem e perdendo sua força ao longo do tempo.
- Agentes Linfócitos: possuem dois modos de operação. Em espera realizam tarefas rotineiras de administração de segurança tais como rodar um verificador de integridade de arquivos ou fazer uma varredura de portas na rede. Quando "banhados em feromônio" mudam seu estado para ativo e migram para a origem do alerta procurando tomar atitudes defensivas para mitigar o efeito do ataque.

Na época da escrita desse documento os autores estavam terminando o desenvolvimento de um protótipo, mas o código do mesmo não foi disponibilizado para testes adicionais. A idéia geral é bastante criativa, porém a ausência de um protótipo deixa dúvidas sobre a viabilidade de implementação prática. Além disso, a

utilização de agentes móveis que “migram” para a área do ataque pode adicionar uma fragilidade ao sistema que pode ser induzido a analisar falsos ataques enquanto ameaças reais são efetuadas em outras partes da rede.

2.3.4 Resposta Automática a Incidentes Utilizando Agentes Inteligentes

Essa metodologia propõe uma arquitetura para resposta automatizada a incidentes de segurança através de agentes inteligentes que aprendem através de reforços fornecidos pelo administrador do sistema [21]. A arquitetura proposta é a seguinte:

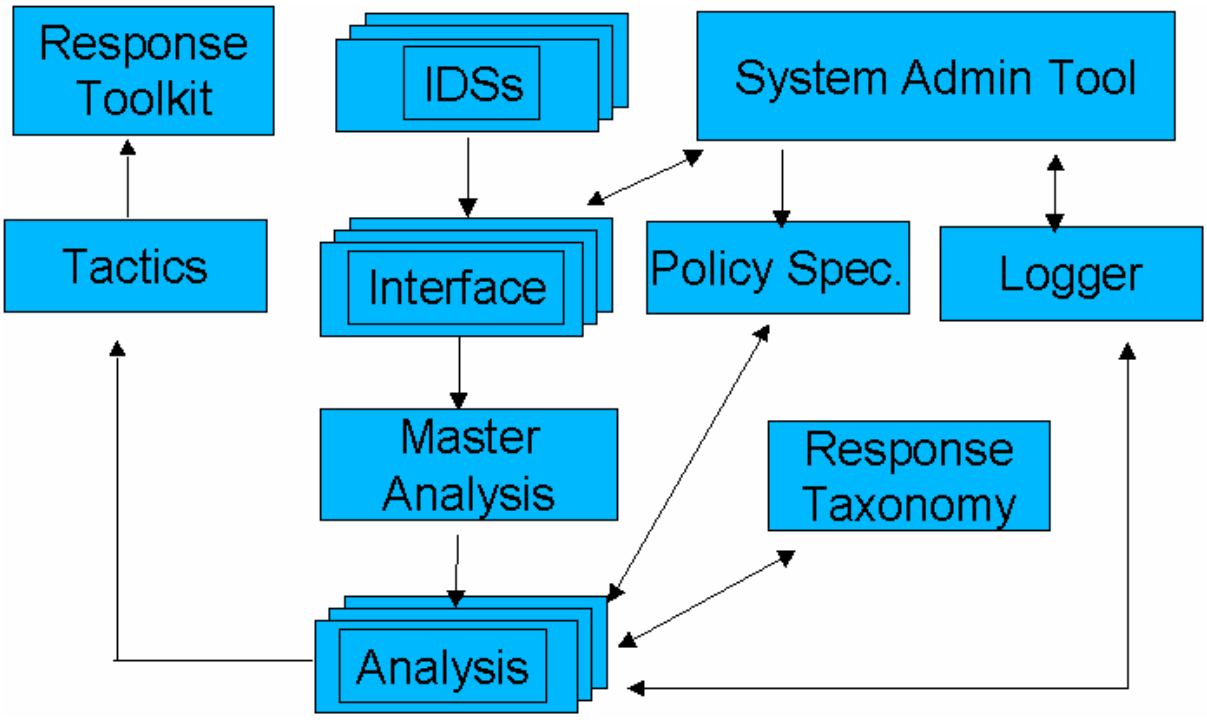


Figura 5 - Resposta Automática a Incidentes Utilizando Agentes Inteligentes

O funcionamento do sistema pode ser resumido através dos seguintes passos no processo:

1. Vários IDS monitoram o sistema e geram alarmes;
2. Agentes de Interface mantêm métricas de confiança dos IDS;
3. O módulo *Master* classifica o ataque como novo ou continuação e encaminha para o agente de análise;
4. O Agente de Análise trata o incidente até o fim e gera um plano de ação abstrato consultando a taxonomia e a política;
5. Agentes de Tática decompõe o plano abstrato em ações específicas e chama os componentes apropriados do *Toolkit* de Respostas.
6. Análise e Táticas são adaptativas dependendo nos resultados de ações anteriores
7. Ações devem ser classificadas pelo administrador, que lê os *logs* e ajusta os parâmetros do sistema.

A metodologia é bastante completa e oferece vários pontos que podem ser aproveitados no desenvolvimento do PIPS. De fato o fluxo de dados desde os IDS até o módulo de análise é bastante semelhante ao descrito no capítulo 4.

2.4 Sistemas de Prevenção de Intrusão (IPS)

Como evolução natural dos Sistemas de Detecção de Intrusão (IDS), surgiram os Sistemas de Prevenção de Intrusão (IPS) que combinam as capacidades de detecção dos IDS e dos anti-vírus com elementos ativos de rede como *firewalls* e *proxies* possibilitando a automatização de contra-medidas objetivadas a minimizar os efeitos de ataques em andamento. Para tanto faz-se necessário o posicionamento desses sistemas em pontos críticos da rede onde é concentrado a maior parte do tráfego útil.

Apesar de todo o alarde mercadológico que caracterizou essa tecnologia como revolucionária, os críticos desses sistemas argumentam que ainda é muito cedo para tal entusiasmo [22]. Muitos dos sistemas ora denominados de IPS não passam de adaptações dos antigos IDS para tomarem ações em casos de ataques. Essa nova característica pode em muitos casos ser perigosa causando efeitos indesejados e só deveriam ser utilizadas quando houvesse um alto grau de certeza a respeito da veracidade do ataque detectado.

Outra questão importante a respeito dos IPS é que por atuarem como elementos ativos da rede sua capacidade de processamento e resposta tem que ser alta, evitando que gerem latência e perda de pacotes válidos. Por serem funcionalmente muito semelhante aos IDS existe o receio de que esses sistemas possam sofrer dos mesmos problemas de eficiência e incerteza que levaram ao desuso dos seus antecessores.

2.5 Limitações das Soluções Anteriores

Além dos problemas citados na seção 1.2 a maioria dos sistemas de detecção de intrusões tradicionais possui algumas limitações e características de funcionamento que tem levado a uma queda considerável no seu uso na maioria das redes.

Por não terem uma visão global do estado e configuração do ambiente monitorado os sensores não possuem uma maneira de classificar os eventos quanto ao grau de risco oferecido. Além disso, não é mantido estado entre alertas gerados fazendo com que um mesmo ataque gere dezenas de alertas independentes. **O alto volume de alertas gerados** em decorrência desses fatores tende a insensibilizar as equipes responsáveis pela sua análise escondendo eventos realmente relevantes entre tantos outros de menor importância.

Uma quantidade considerável desses alertas é gerada a partir de situações normais de operação das redes e sistemas e não representam nenhuma ameaça aos sistemas monitorados. A esses falsos alertas costuma-se dar o nome de **Falsos Positivos** e eles representam o principal alvo de críticas aos sistemas de detecção de intrusão. Na tentativa de minimizar o volume de alertas inválidos gerados é comum a remoção de boa parte das assinaturas e/ou sub-sistemas que geram muitos alarmes falsos. Esse processo pode ser arriscado já que eventos relevantes podem ser inadvertidamente suprimidos fazendo com que tentativas reais de intrusão passem desapercebidas. Esses eventos não notificados são chamados de **Falsos Negativos** e podem também acontecer quando o sistema monitorado é alvo de ataques desconhecidos pelo sistema de detecção ou quando o atacante se vale de técnicas evasivas.

A **ausência de correlações** entre os eventos ocorridos em diversas partes da rede torna a coleta de evidências de tentativas de invasões um processo complicado e demorado resultando muito comumente em uma análise incompleta, prejudicando a eficiência do procedimento de resposta a incidentes.

A **falta de automatização** e a **gerência descentralizada** são problemas enfrentados em redes de médio e grande porte que possuem vários sistemas de monitoração implantados. Mudanças na rede como adição de novas máquinas e serviços e alterações de topologia demoram a ser refletidas nas configurações dos sistemas de gerência introduzindo um espaço temporal onde essas novas entidades não são monitoradas fazendo com que ataques a elas possam passar incólumes.

Uma característica fundamental desses sistemas é que eles são intrinsecamente **reativos**, não possibilitando a detecção de situações de risco antes que um agente externo já tenha tentado uma intrusão. Muitas vezes as tentativas de intrusão já obtiveram êxito e causaram danos aos dados e sistemas protegidos.

2.6 Síntese

Sistemas de detecção de intrusão possuem um importante papel na manutenção da segurança de sistemas e redes. Após uma fase de grande popularidade no final dos anos 90 os IDS sofreram um período de rejeição, por parte dos administradores de sistemas e segurança, principalmente devido ao alto volume de falsos positivos por eles gerados e ao significativo impacto em performance nos sistemas monitorados. Como solução para esses problemas surgiram várias propostas de detecção de intrusão distribuída e uso de técnicas de inteligência artificial para melhorar a qualidade das análises geradas. Embora bem fundamentados do ponto de vista acadêmico, esses sistemas não obtiveram sucesso na prática. Esse fato se deu provavelmente pela ausência de compatibilidade com sistemas consagrados ou pelo desinteresse dos autores em dar continuidade aos projetos. Mais recentemente surgiram os sistemas de prevenção de intrusão (IPS) que adicionam a capacidade de bloqueio dos ataques antes que eles causem dados aos sistemas protegidos. Apesar do sucesso mercadológico esses sistemas ainda não se provaram como soluções definitivas e são acusados de sofrerem das mesmas limitações dos IDS, principalmente no que diz respeito aos falsos positivos. No próximo capítulo será proposto um sistema que pretende aproveitar as melhores características dos sistemas estudados e permitir correlações de dados coletados de maneira distribuída através do uso de sistemas especialistas baseados em regras de produção.

3 Solução Proposta: PIPS

O uso de sistemas de detecção de forma isolada tem se mostrado ineficiente e caído em contínuo desuso. Apesar disso o investimento prévio em produtos e treinamento e os altos níveis de maturidade alcançados por muitos desses sistemas sugerem a necessidade de novas formas de utilização que possam aproveitar ao máximo os benefícios oferecidos por eles.

Com os objetivos de suprir as deficiências dos sistemas de detecção de intrusão tradicionais, integrar dados colhidos utilizando diversas fontes, aumentar a qualidade das análises sobre tentativas de intrusão e estudar as possibilidades de utilização de sistemas distribuídos e agentes inteligentes na detecção de intrusão propomos uma nova abordagem para o problema, doravante denominada PIPS – *Proactive Intrusion Prevention Systems* (ou Sistemas Proativos de Prevenção de Intrusão). Nesse capítulo será mostrada uma visão geral do funcionamento desses sistemas, detalhando seus requisitos, forma de atuação e resultados esperados.

3.1 Abordagem adotada

Ao contrário da maioria dos sistemas tradicionais a ação dos PIPS não se restringe à reação a eventos detectados pelos sensores. O enfoque principal é na monitoração constante da rede de forma pró-ativa tentando identificar ameaças e vulnerabilidades o mais cedo possível, evitando a exposição desses sistemas a ataques externos.

Para isso são utilizadas ferramentas de varredura, levantamento de vulnerabilidades, enumeração e de verificação de atualizações que ajudam a construir uma representação em bancos de dados do estado ativo da rede. Esse estado é permanentemente validado, servindo de base para correlações com eventos de segurança levantados pelos sistemas de detecção de intrusão. Pontos fracos identificados durante a construção e manutenção do estado ativo da rede já disparam alertas independentemente de terem sido alvos de ataques externos.

A estratégia utilizada é de aproveitar ao máximo as funcionalidades de sistemas de detecção tradicionais, como S*nort[12]*, AIDE[13], *arpwatch*[23] e *logwatch*[24] e adicionar valor à análise através do co-relacionamento dos eventos detectados nas diversas fontes procurando identificar padrões de ataques conhecidos e situações que possam representar ameaça à rede. Para montagem do estado ativo da rede são usadas ferramentas como o *Nmap [25], Nessus[26], HFNet* [27]e enumeradores NetBIOS.

O estado da rede também pode ser atualizado através de ações decorridas a partir de um evento detectado pelos sensores. Por exemplo: uma nova máquina adicionada numa rede *ethernet* é detectada pelo *arpwatch* que dispara um evento. O módulo de análises detecta que não existem informações já colhidas a respeito dessa máquina, e dispara varreduras usando o *Nmap* e *Nessus* para essa nova máquina.

3.2 Requisitos do Sistema

Nesta seção são listados de maneira informal os principais requisitos do sistema. O desenvolvimento do sistema foi norteado por esses princípios básicos e a incorporação de novas funcionalidades levou em consideração todos esses fatores.

3.2.1 Arquitetura Distribuída e Flexível

A forma como as diversas partes do sistema estão organizadas e se inter-relacionam é de vital importância para o seu funcionamento. Os diferentes componentes do sistema devem poder ser instalados em pontos distintos da rede, possibilitando não só a distribuição do processamento como também o acesso a maior volume de informações.

A flexibilidade é um requisito fundamental da arquitetura, devendo ser possível adicionar novas funcionalidades, alterar configurações e remover coletores e sensores sem prejudicar significativamente o funcionamento global do sistema. A adição de um novo tipo de coletor ou sensor deve ter o menor impacto possível. Seu desenvolvimento deve poder abstrair detalhes de comunicação, manutenção de estado, formatação e enfileiramento de mensagens, concentrando-se no tratamento de dados e, quando necessário, na interface com ferramentas externas.

3.2.2 Comunicação Segura

Os mecanismos de comunicação utilizados para troca de mensagens e dados entre as entidades do sistema devem ser projetados da forma mais eficiente possível para diminuir o impacto de sua implantação. Porém devem também observar requisitos de segurança tais como privacidade e autenticação.

Para comunicação através da rede em IDS é comum a adoção de protocolos existentes e bem conhecidos como TCP e UDP, sendo que o último apresenta melhor performance, porém não oferece garantia de entrega. No entanto esses protocolos não oferecem garantias de privacidade e autenticação, gerando a necessidade de uma camada adicional com protocolos de criptografia. O formato das mensagens deve ser padronizado e flexível, criando a necessidade da utilização de algum padrão de formatação como o XML.

3.2.3 Integração a sistemas de Monitoração

As arquiteturas distribuídas, além de resolverem problemas clássicos de detecção de intrusão, podem proporcionar o desenvolvimento e integração de sistemas complexos e inteligentes de monitoração de serviços e recursos. A análise e a correlação de dados coletados em fontes diferentes da rede podem agilizar a detecção de pontos de falha e ajudar as equipes de administração de sistemas na resolução de problemas.

Por exemplo, um determinado servidor - que oferece os serviços de HTTP, POP3 e SMTP - pode ser monitorado por agentes locais através de verificação da presença de *daemons* escutando nas respectivas portas. No entanto, isso não garante que o serviço esteja realmente funcionando perfeitamente. Outro agente poderia verificar se conexões de teste a esses serviços são bem sucedidas. Ainda assim, a utilização de apenas um ponto de coleta remoto pode implicar em falsos positivos e falsos negativos. Diversos pontos de coleta estrategicamente posicionados ajudam na identificação dos possíveis pontos de falha e podem gerar relatórios mais precisos.

3.2.4 Relatórios e Estatísticas

Um ponto importante na implementação de bons monitores e detectores de intrusão é a capacidade de geração de relatórios e estatísticas periódicas, simples e representativas. Enquanto as equipes técnicas podem ter maior interesse em analisar os detalhes do funcionamento desses sistemas, as equipes de gerência tipicamente estão interessadas na apresentação de resultados qualitativos e quantitativos que justifique seus investimentos em pessoal, software e hardware e que enfatizem a evolução da segurança dos sistemas ao longo do tempo.

3.2.5 Interface Funcional

À medida que eventos são coletados, classificados, reduzidos e correlacionados, informações devem ser repassadas para os usuários do sistema de forma clara e objetiva. Decisões sobre as configurações e o estado do sistema devem poder ser alteradas pelos usuários que possuem conhecimento de fatores externos ao sistema, tais como política de segurança e necessidades das equipes de administração.

3.3 Componentes do Sistema

Nesta seção serão vistas descrições gerais dos tipos de componentes que formam o sistema. Detalhes específicos de quais componentes já foram desenvolvidos e encontram-se em produção serão vistos no próximo capítulo. A caracterização é realizada de acordo com as funcionalidades dos componentes, sendo possível que um sub-sistema exerça as funções de dois ou mais componentes, assim como é possível que uma mesma função seja exercida de forma colaborativa por várias partes do sistema distribuídas em diversas partes da rede.

3.3.1 Agentes

São responsáveis pela formatação, enfileiramento e troca de mensagens entre componentes localizados em nós distintos do sistema. Diversas situações geram necessidade de comunicação entre as partes do sistema: envio de alertas para a central de processamentos, solicitação de informações entre os nós e mensagens de re-configuração. Um formato unificado e estruturado de mensagens XML foi definido para essa comunicação. Autenticação e Criptografia são adicionadas ao sistema através de SSL/TLS com o uso de certificados digitais de cliente e de servidor fornecidos por uma Autoridade Certificadora membro de uma infra-estrutura de chaves públicas gratuitas [28].

Os agentes oferecem uma interface de programação unificada que permite o fácil desenvolvimento de sensores e coletores, permitindo que esses sejam adicionados ao sistema na forma de *plugins*. Cada sensor ou coletor fica responsável apenas pelo tratamento dos dados especificamente associados à sua função, deixando a cargo dos agentes a transformação desses dados para XML e o envio para a central de processamento.

3.3.1.1 Coletores

Responsáveis pelo levantamento de informações sobre os sistemas monitorados, serviços por eles fornecidos, versões dos servidores, vulnerabilidades conhecidas e informações diversas como: usuários, programas instalados e em execução e utilização de recursos (memória, CPU, rede, etc). As informações levantadas por esses coletores são utilizadas para construção e manutenção do estado ativo da rede, além de serem armazenados ao longo do tempo para identificação de tendências históricas e geração de análises sobre a evolução do estado de segurança dos sistemas.

O disparo de um coletor pode ser efetuado manualmente pelos operadores do sistema, automaticamente durante uma análise ou ser agendado para ser executado periodicamente. No caso de coletores formados por ferramentas de terceiros (como por exemplo o N*essus* e o N*map)* deve ser possível a execução desses utilitários de forma externa ao sistema e os resultados poderem ser importados através de utilitários desenvolvidos para essa finalidade ou através da própria interface do sistema.

3.3.1.2 Sensores

Atuam de forma passiva capturando informações relevantes em diversos pontos da rede. Tipicamente constituídos por IDS existentes com adaptadores desenvolvidos para transformação dos dados coletados para formatos intermediários próprios, cujo processamento será realizado por outros componentes do sistema. Por exemplo, um sensor formado pelo *Snort* e pelo seu adaptador (chamado de S*nortAdmin*) pode gerar alertas baseados no mecanismo de reconhecimento de assinaturas de ataques descrito na seção 2.1.1. É importante que os sensores permaneçam ativos pelo máximo tempo possível, para isso devem ser disponibilizados mecanismos de inicialização automática e enfileiramento de mensagens, permitindo que esses elementos atuem de forma autônoma em relação à central de processamentos.

3.3.2 Central de Processamento

Constitui a parte mais importante do sistema. Recebe os dados gerados nos diversos sensores e coletores, extraem dados relevantes e os armazenam em servidores de bancos de dados. Para cada sensor ou coletor do lado dos agentes existe um processador na central que também é desenvolvido de forma a abstrair detalhes de comunicação e formatação de dados. Após o armazenamento os dados são disponibilizados para o módulo de análises que os processam e efetuam associações de eventos procurando relacionar dados coletados nas diversas fontes.

A **matriz de correlacionamentos** é o módulo de análises formado por um conjunto de sistemas especialistas que produzem classificações mais refinadas através de sistemas de pontuação para alertas. Durante o processo de análise novas informações podem ser solicitadas aos agentes posicionados em pontos da rede afetados pelo possível ataque. Um dos sistemas especialistas, desenvolvidos durante o trabalho, melhora o esquema de classificação do S*nort* através do cruzamento dos eventos detectados com informações presentes no estado ativo da rede sobre os nós envolvidos. Várias assinaturas de ataques só fazem sentido se o servidor alvo for de um determinado modelo e versão. Ao receber notificação de um alerta, o sistema verifica se o servidor alvo pode ser afetado pelo ataque e classifica o alerta

adequadamente. Esse sistema especialista ajuda a diminuir o volume de falsos positivos normalmente gerados pelos detectores de intrusão de rede tradicionais.

Uma tarefa importante do sistema é agrupar as informações processadas para a geração de relatórios periódicos, que servirão tanto para manter o histórico dos eventos ocorridos ao longo do tempo como para justificar os investimentos realizados. No caso de uma detecção comprovada os relatórios devem prover informações necessárias e suficientes para a realização de contra-medidas técnicas ou administrativas.

3.3.3 Interfaces Gráficas

Existem dois tipos de interface: uma destinada aos operadores e administradores do sistema e outra para visualização de resultados. A interface administrativa permite iniciar e parar a execução de coletores e sensores, verificar estado dos agentes e de análises ativas, efetuar modificações nas configurações do sistema e ajustes nos dados e parâmetros de execução, além do agendamento de tarefas. A interface de visualização de resultados é destinada a apresentação de relatórios e exibição de dados armazenados pela central de processamentos. Essa interface é exclusivamente de consulta e pode ser acessada remotamente através de navegadores *web* tradicionais.

3.4 Arquitetura

O relacionamento entre os componentes do sistema obedece a seguinte hierarquia: vários sensores e coletores podem estar associados a um mesmo agente funcionando como *plugins* desse agente. A comunicação entre cada sensor ou coletor e o seu agente é realizado através de um mecanismo simples de comunicação inter-processos. Os agentes são distribuídos de forma estratégica na rede de forma a otimizar a coleta de dados. Vários agentes podem ser necessários para monitorar uma mesma rede, assim como um mesmo agente pode monitorar várias sub-redes. Todos os agentes enviam os dados gerados pelos coletores e processadores para a central de processamento através de um canal de comunicação cifrado estabelecido entre o agente e a central. Os dados são então processados e armazenados em bancos de dados, além de serem disponibilizados para análises ativas e para serem exibidos na interface administrativa para os operadores. Os dados gerados durante o processamento e após as análises ficam armazenados no banco de dados e podem ser visualizados através da interface *web* pelas equipes de gerência. A Figura 6 mostra os principais componentes do sistema e como eles se relacionam.

A forma como os sensores e coletores são posicionados na rede monitorada deve ser avaliada de acordo com a topologia da rede, escolhendo-se os pontos que ofereçam melhor acesso para a coleta dos dados. Cada nó da rede pode existir independentemente dos outros, coletando os dados e enfileirando-os para posterior análise. Falhas temporárias em partes do sistema não atrapalham as análises de longo prazo embora possam impossibilitar análises instantâneas. Em geral é recomendado que exista pelo menos um agente por segmento de rede, possibilitando a atuação dos coletores sem que sejam afetados por filtros de pacotes ou outros elementos ativos da rede, como roteadores, proxies ou IPSs. Os sensores também devem ser posicionados de forma estratégica de acordo com os tipos de tráfego a ser monitorado. Detectores

de intrusão de rede devem obedecer às boas práticas de posicionamento, em geral em pontos de estrangulamento da rede.

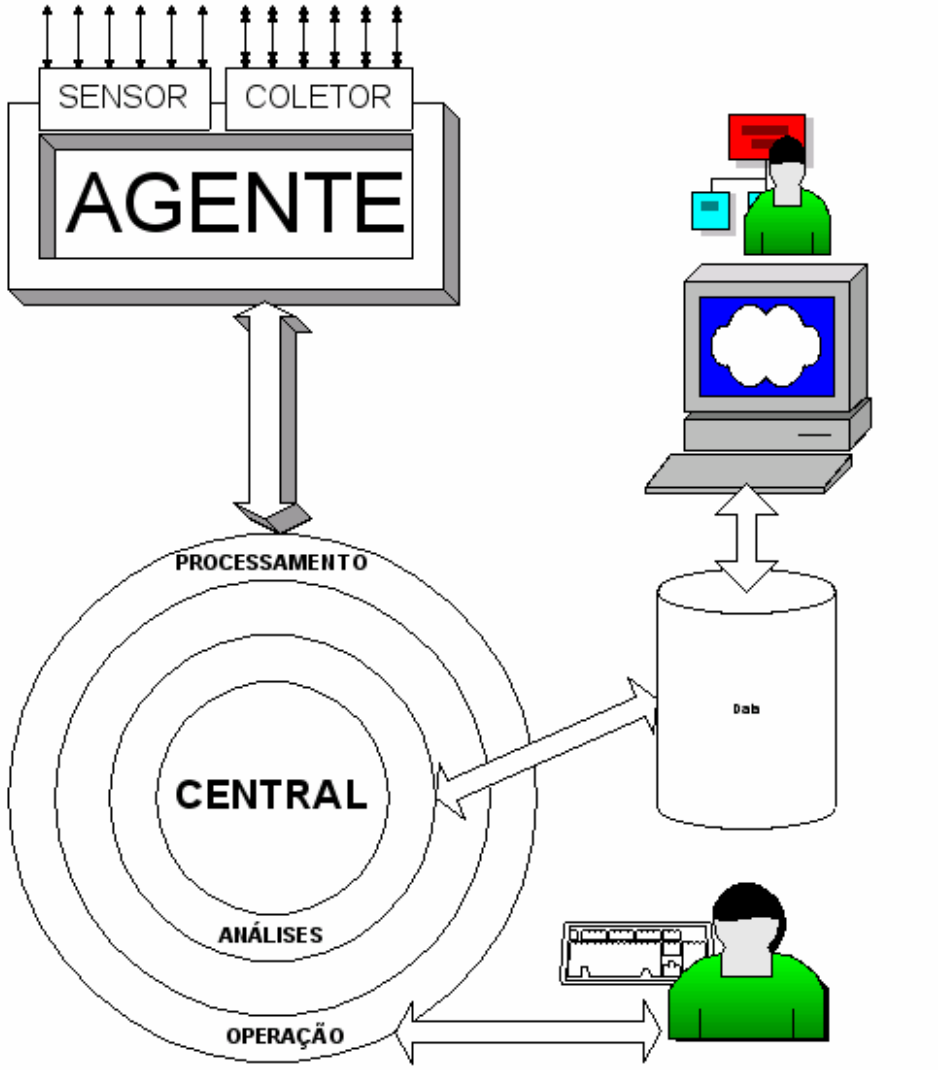


Figura 6 - Componentes da solução proposta

É importante ressaltar que não existe a necessidade de instalar um agente em cada nó da rede, já que grande parte dos sensores e coletores pode agir de forma remota. Servidores mais sensíveis requerem monitoramento local através de detectores de intrusão de *host*, analisadores de *log*, etc. Nesses servidores faz-se necessária a instalação de sensores e coletores em agentes locais. A Figura 7 mostra algumas possíveis localizações dos componentes do sistema em uma rede típica.

Nesse exemplo, uma máquina (1) foi dedicada para monitoração da rede de servidores, tendo acesso a todo o tráfego dessa rede. Nela podem ser instalados sensores para monitoração de tráfego ARP, detecção de intrusão de redes e coletores para levantamento remoto de informações para montagem do estado ativo da rede. No servidor de páginas *web* (2) foi instalado um agente local para detecção de intrusão de *host* com um verificador de integridade como o AIDE. O serviço de diretórios nessa rede é oferecido num servidor Windows (3). Nesse servidor foi instalado um agente

contendo coletores para verificação de versões de software e monitoração de mudanças em arquivos e no registro.

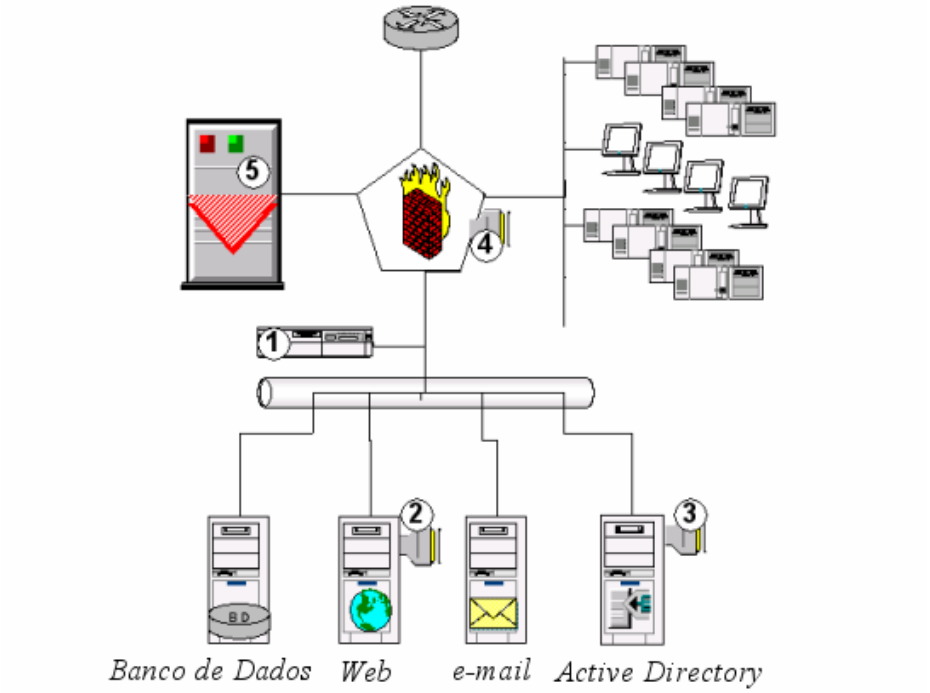


Figura 7 - Exemplo de posicionamento de componentes em rede típica

No *firewall* (4) foi instalado um agente com um detector de intrusão de *host*, um analisador de *logs* e um coletor para monitoração de mudanças na base de regras. Esse agente hospedado no *firewall* também pode ser usado para realizar varreduras na rede interna, desde que a base de regras não restrinja as conexões dessa máquina para as máquinas da rede interna. Um servidor foi dedicado para hospedar a central de processamento (5), na figura esse servidor está isolado em uma sub-rede, mas poderia também estar posicionado na DMZ ou ligado através de uma VPN com a rede de um prestador de serviços de monitoração continuada de segurança. Os servidores de e-mail, banco de dados e as máquinas da rede interna não possuem agentes instalados. Toda sua monitoração é realizada remotamente através do agente instalado na máquina dedicada (1).

3.5 Síntese

O Sistema Proativo de Prevenção contra Intrusões (PIPS) procura solucionar os problemas dos sistemas de detecção de intrusões tradicionais através da análise inteligente de dados coletados de maneira distribuída e da correlação dos eventos de IDS com o estado ativo da rede.
Além de possuir uma arquitetura distribuída, o sistema permite a reutilização de ferramentas consagradas na área de segurança através do uso de *plugins*. Os principais componentes do sistema incluem os agentes, que possuem coletores e sensores e a

central de processamento onde são realizadas as análises e que disponibiliza os resultados para serem consultados pelas interfaces gráficas.

4 Implementação

Neste capítulo serão apresentados os detalhes de implementação de um sistema proativo de prevenção de intrusão (PIPS) baseado na arquitetura e atendendo aos requisitos e funcionalidades apresentados no capítulo anterior. O sistema aqui apresentando foi desenvolvido como parte de uma plataforma para o gerenciamento de segurança de ambientes computacionais de grande escala, complexo e heterogêneo. A plataforma, denominada *Matrix [29]*, integra dados de segurança a partir de uma série de diferentes ferramentas, fornecendo a possibilidade de uma interface única de gerência, consolidando dados de diferentes origens e proporcionando uma visão integrada da segurança do ambiente, sem perder a flexibilidade necessária para análises de segurança.

4.1 Visão Geral do Sistema

O PIPS é uma parte do Matrix, uma plataforma para gerenciamento escalável de segurança de ambientes, com foco na minimização da intervenção manual para uma análise de segurança, mantendo a ótica da continuidade do processo. Esta plataforma provêm uma estrutura que permite a realização de levantamentos de segurança automatizados, contando também com recursos para armazenamento, extração e análise cumulativa dos dados de segurança recolhidos.

Existe no mercado uma vasta gama de ferramentas para análise de segurança, mas a maioria é bem específica, fortemente orientada a uma determinada plataforma ou para determinados serviços. Como não seria factível reimplementar todas estas ferramentas de forma integrada, a plataforma é flexível e permite a integração das várias ferramentas existentes, normalizando os diferentes formatos de saída, de forma a facilitar o posterior processamento e análise. Neste sentido, a plataforma funciona como um agregador das diferentes ferramentas existentes no mercado, que se integram a ela. Essa integração também provê outros benefícios: a plataforma esforça-se para prover uma console única de gerência das ferramentas, tanto para execução e fornecimento de opções quanto para o armazenamento e recuperação de informação.

Depois que as informações recolhidas pelas ferramentas são normalizadas e integradas na plataforma, existe espaço para realização de análises mais detalhadas e que agregam uma visão abrangente do ambiente, ao invés da visão fragmentada provida por cada ferramenta individualmente. Por exemplo: análises históricas de tendências, ataques e falhas de configuração; geração de relatórios consolidados com a visão geral da segurança do ambiente em termos de vários critérios (vulnerabilidades, novos pontos ativos, etc.); disparo de alarmes programados em caso de eventos de segurança pré-definidos, como a descoberta de um sistema vulnerável, ou um registro de ataque oriundo de um IDS; correlação de eventos de intrusão que individualmente não se destacam, mas que analisados em conjunto podem revelar ataques.

4.2 O PIPS como parte do Matrix

O Matrix é responsável pela manutenção e monitoração contínua do ambiente. Para executar essas tarefas ele está dividido em seis domínios de segurança:

Controle de Acesso: gerência de controle de acesso em *firewalls*, VPN e roteadores e auditoria de segurança em base de usuários e senhas.

Validação de Políticas: garantia da aplicação de políticas de configuração de segurança em serviços e sistemas operacionais. Checagem de integridade e detecção de desvios da política estabelecida.

Detecção de Intrusão: gerenciamento de sistemas de detecção de intrusão, identificação de eventos, centralização e análise de logs.

Inventário de Rede: mapeamento de pontos ativos e suas características: serviços, sistema operacionais, etc. Estado do ambiente de rede como um todo.

Levantamento de Vulnerabilidades: identificação de vulnerabilidades em servidores (serviços e sistemas operacionais). Gerência da base de conhecimento e correção associada às vulnerabilidades.

Correlações de Segurança: análises que correlacionam informações de segurança vindas dos diferentes domínios, identificando pontos críticos, extraindo métricas e provendo uma visão consistente do ambiente.

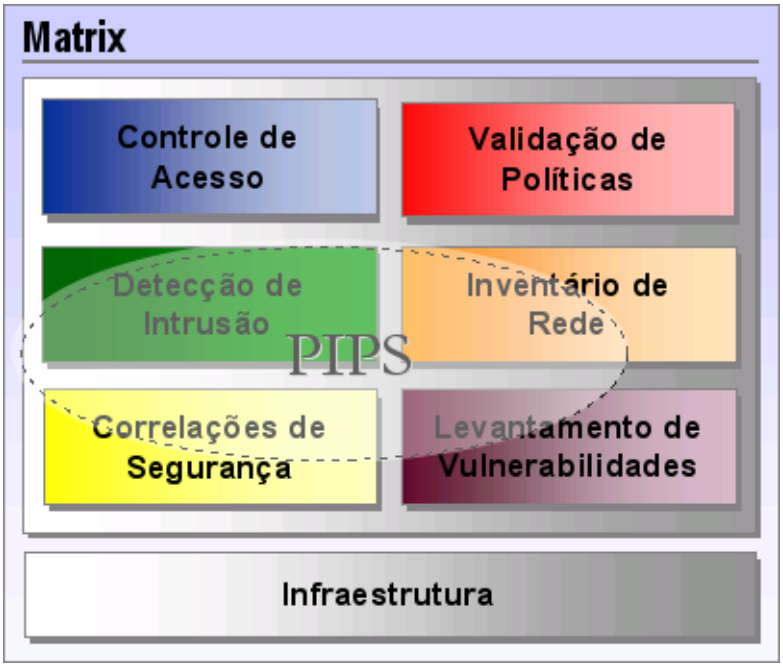


Figura 8 – O PIPS como parte do Matrix

Os principais domínios associados ao PIPS são os de Detecção de Intrusão e de Correlações de Segurança. O inventário de rede é diretamente utilizado e alimentado pelo PIPS para a construção do estado ativo da rede, que é utilizado durante as análises para caracterizar eventos de segurança. O levantamento de vulnerabilidades também é essencial para a caracterização do risco de ataques detectados. Os domínios mostrados são aspectos funcionais do sistema, representando operações de segurança visando a proteção do cliente.Todavia, é necessário o Matrix possuir uma série de recursos de infra-estrutura, sobre o qual podem ser construídas funcionalidades tão distintas como as exigidas pelos domínios. Fazem parte da infraestrutura:

armazenamento, segurança, comunicações, troca de mensagens, gerência de processos, interface análises e relatórios. Todos esses recursos da infraestrutura são utilizados pelo PIPS para realizar as tarefas sob sua responsabilidade.

4.3 Agentes

Os agentes são os componentes fundamentais da distribuição do sistema. Eles são instalados em pontos distintos da rede monitorada, coletando dados relevantes à segurança do ambiente e os enviando para a central de processamentos. Por necessitar ser instalado em sistemas operacionais diversos e não dedicados, eles precisam ser portáveis e leves e sua instalação deve causar o menor impacto possível no servidor que o hospeda. Além disso, eles devem oferecer uma interface simples para o desenvolvimento de sensores e coletores. Esses requisitos nortearam a escolha da linguagem *Perl* [30] para sua implementação, devido à conhecida flexibilidade, portabilidade, popularidade e larga biblioteca de componentes publicamente disponíveis para essa linguagem. Atualmente os agentes já foram testados com sucesso em vários sistemas operacionais, entre eles: Linux, FreeBSD, Solaris, AIX, HP-UX, Windows NT,2000,XP e 2003.

Com o objetivo de tornar o desenvolvimento dos sensores e coletores o mais simples possível, os agentes concentram as funcionalidades comuns de comunicação, armazenamento, enfileiramento e troca de mensagens, segurança e controle de processos. Dessa forma os agentes oferecem um conjunto de serviços, entre eles:

Comunicação: agindo tanto como cliente quanto como servidor, o agente utiliza conexões TCP para realizar a troca de mensagens e arquivos com a central de processamentos. Ao receber uma conexão, o agente lê os dados e realiza verificações básicas no formato da mensagem removendo os marcadores de cabeçalho e rodapé e encaminha o conteúdo para o módulo de processamento de mensagens. Ao receber resposta do processamento, a mesma conexão é utilizada para enviar a mensagem de resposta acrescida de cabeçalho e rodapé. Ao processar mensagens contendo eventos ou notificações geradas pelos sensores e coletores, o agente inicia uma nova conexão para a central de processamentos e envia os dados. Eventuais erros de comunicação são tratados e, após um número definido de tentativas, as mensagens voltam para a fila, e após um intervalo de tempo ocorrem novas tentativas de conexão. Por simplicidade, atualmente uma mesma conexão não é utilizada para troca de múltiplas mensagens entre os agentes e a central evitando assim a complexidade inerente à multiplexação de mensagens.

Criptografia e Autenticação: todas as conexões entre o agente e a central de processamentos podem ser cifradas utilizando SSL. A escolha deve ser regida pela necessidade de sigilo e nível de confiança na rede. Quando habilitado, o módulo de SSL pode ser configurado para exigir a presença de certificados emitidos por autoridade certificadora confiada, permitindo assim a autenticação das conexões. Não foi identificada até o momento a necessidade de autorização, ou seja, qualquer cliente autenticado pode executar todas as funções disponibilizadas pelo agente.

Mensagens: através desses canais de comunicação são trocadas mensagens de diferentes tipos, incluindo requisições de execução, respostas a pedidos de execução, notificações originadas pelos sensores, entre outras. As mensagens são formatadas de acordo com o padrão XML, o que traz uma série de benefícios: estrutura hierárquica de mensagens, independência dos dados, diferentes tipos de codificação e facilidades para extensão e mudanças no formato da mensagem. A Figura 9 mostra alguns dos

elementos mais importantes das mensagens trocadas entre o agente e a central. Cada mensagem tem um identificador único que é o atributo ID do elemento *Msg*. O elemento *Address* contém os endereços de origem e destino das mensagens e são utilizados pelos agentes e pela central para identificar quem enviou a mensagem e quem é o destinatário.

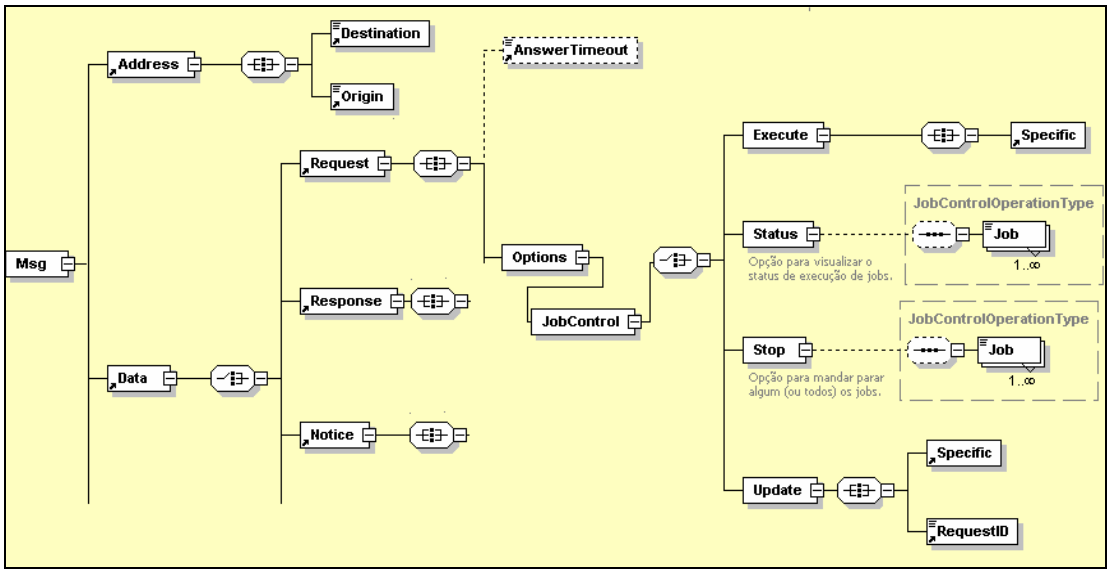


Figura 9 – Alguns elementos das mensagens XML

Existem três tipos de mensagens: requisições, respostas e notificações. As requisições são criadas pela central de processamentos para iniciar e parar a execução de sensores ou coletores, solicitar informações de estado e enviar atualizações para sensores ou coletores em execução. Os dados contidos no elemento *Specific* não são interpretados pelo agente. Eles são convertidos para uma estrutura de dados formada por *hashes*, listas e *strings* e são repassados para o sensor ou coletor, que utiliza então esses dados convertidos para realizar suas funções. Os dados produzidos pelas tarefas também não precisam ser gerados em formato XML, o agente recebe esses resultados numa estrutura de dados com os mesmos tipos citados anteriormente, converte essas informações para XML e as insere no elemento *Specific* da mensagem de resposta ou de notificação. As mensagens de resposta são geradas no mesmo fluxo de execução e são enviadas na mesma conexão TCP das respectivas requisições. As notificações são enviadas em duas situações: quando geradas por sensor que está em contínua execução ou quando um coletor demora muito para retornar os resultados (tempo superior ao especificado no elemento *AnswerTimeout*). Nesse caso o agente envia uma mensagem de resposta sem conteúdo específico informando que a resposta será atrasada. Quando o coletor termina a execução, os dados gerados são então enviados em uma mensagem de notificação.

Roteamento de Mensagens: um agente pode estar posicionado entre a central de processamento e outros agentes da rede. Nesses casos, as mensagens são enviadas para esse agente roteador que as encaminha para seus vizinhos ou para a central.

Enfileiramento: antes de serem enviadas para seus destinos, as mensagens são armazenadas no sistema de arquivos e são processadas em forma de fila. Caso existam problemas transientes de comunicação entre os agentes e a central, essas mensagens permanecem armazenadas no sistema e são enviadas na mesma ordem que foram geradas quando a comunicação é re-estabelecida.

Transferência de arquivos: no caso de arquivos muito grandes ou que contenham dados binários ou codificados de maneira não usual, a transferência desses dados em mensagens XML pode ser ineficiente. Para possibilitar o envio desse tipo de dados o agente implementa um protocolo simples de transferência de arquivos. Ao enviar uma resposta ou notificação, o sensor ou coletor que deseja transferir um arquivo informa o caminho na estrutura de dados enviada ao agente. Ao processar essa estrutura de dados o agente cria uma entrada na fila de transferência de arquivos contendo informações sobre o arquivo solicitado, tais como caminho, tamanho e resumo MD5. A fila de arquivos é processada de forma análoga porém independente da fila de mensagens. Um coletor que utiliza esse mecanismo é o de rotacionamento de *logs* do *Firewall-1,* que é agendado para periodicamente exportar os *logs* que são guardados em formato binário, compactá-los e enviar os arquivos para a central de processamentos. Esses arquivos são então analisados em busca de indícios de varreduras.

Execução de processos: a função primordial dos agentes é efetuar o controle da execução dos coletores e sensores. Ao ser iniciado, o agente lê do seu arquivo de configuração a lista de coletores e sensores habilitados contendo o nome do coletor ou sensor, o caminho onde ele se encontra no sistema de arquivos e o tipo. Existem duas categorias de coletores e sensores: os simples, que retornam apenas uma resposta por execução; e os que permanecem ativos produzindo notificações periodicamente ou quando ocorrem eventos notáveis. Esses últimos recebem no *Matrix* o nome de *tasks*. Ao receber uma requisição de execução da central, o agente verifica na lista de coletores e sensores se o requisitado existe e está disponível. Em caso positivo o script é executado, recebendo na entrada padrão a estrutura de dados representando a parte específica da mensagem de requisição. A saída padrão do sensor ou coletor é então monitorada em busca de resultados da execução, que podem ser únicos no caso de sensores ou coletores simples ou múltiplos no caso de *tasks.* A inicialização dos sensores e coletores não precisa necessariamente ser requisitada pela central de processamentos. Eles também podem ser executados automaticamente durante a inicialização do agente ou serem agendados para execução periódica através de um mecanismo de agendamento semelhante ao *cron* dos sistemas UNIX.

4.4 Sensores e Coletores

Utilizando-se a infra-estrutura dos agentes, tornou-se relativamente simples o desenvolvimento de *plugins* que efetuassem as funções de sensores e coletores. A Tabela 1 serve para ilustrar a facilidade de desenvolver um coletor que informa a quanto tempo o sistema está ativo, quanto desse tempo o sistema passou em ocioso e o horário atual. O trecho em destaque é relativo ao processamento, enquanto as outras linhas são gerais e podem ser repetidas em outros coletores. A única obrigação é que a rotina *sendData* receba um *hash* contendo o código de resposta e os dados a serem inseridos na parte específica da mensagem de resposta. O XML contendo o resultado da execução desse coletor é mostrado logo em seguida, destacando a parte específica da mensagem.

Código de um coletor simples

```
#!/usr/local/bin/perl
#----- Utilizar a interface genérica --------#
use Jobs::Interface;
my $interface = Interface->new();
#----- Ler dados da requisição --------------#
my $data      = $interface->receiveData();
#------------ Processamento ----------------#
open(t,"/proc/uptime");
my ($up,$idle) = split(/\s|\n/,<t>);
close(t);
my $info = {'Up' => $up,'Idle' => $idle,'Time' => time};
#----------- Produzir Resposta -------------#
my %res;
$res{Results}{Code} = 'ok';
$res{Results}{Data} = $info;
#-------------- Enviar Resposta ------------#
$interface->sendData(\%res);
```

Mensagem XML de resposta produzida pela execução do coletor acima

```
<Msg ID="msgID-489800168" xmlns="TempestMatrix">
 <Address>
  <Destination>Matrix.C01.Central.Oper.Plugin.UpHandler</Destination>
  <Origin>Agent.C01.Ag01.JobControl.Plugin.Up</Origin>
 </Address>
 <Data Type="response" Delayed="no">
    <Response ID="resID-2973678784">
      <Code>ok</Code>
      <OriginalRequest>
        <MsgID>msgID-1311026023</MsgID>
        <RequestID>reqID-250460194</RequestID>
      </OriginalRequest>
      <Specific xsi:type="SimpleReplySpecificType">
        <Idle>609626.55</Idle>
        <Time>1099685852</Time>
        <Up>638528.33</Up>
      </Specific>
    </Response>
  </Data>
</Msg>
```

Tabela 1 - Código Perl e XML de um coletor simples

Seguindo a estratégia de aproveitar ao máximo as ferramentas existentes de boa qualidade e adaptá-las ao *framework*, foram desenvolvidos diversos sensores e coletores. Algumas ferramentas são utilizadas diretamente através de adaptadores que as executam através de chamadas de sistema e capturam seus resultados. Os enumeradores NBT e SNMP tiveram suas funcionalidades estendidas para melhor se enquadrar ao framework. Apenas uma ferramenta, o *Arpwatch* foi completamente reimplementada, pois o esforço para adaptá-la ao sistema seria maior do que o da nova implementação. A Tabela 2 contém uma resumo dos sensores e coletores já desenvolvidos destacando sua forma de execução (pontual ou contínua), sua função (sensor ou coletor), o grau de relevância para prevenção pró-ativa de intrusões e os sistemas operacionais que os suportam. Informações mais detalhadas são fornecidas a seguir:

Plugin	Ferramenta	Execução	Função	Relevância (PIPS)	Plataforma
SnortAdmin	Snort	Contínua	Sensor	5	Unix
Wigo	Arpwatch	Contínua	Sensor	4	Unix
NmapInventory	Nmap	Contínua	Coletor	4	Unix
NessusInventory	Nessus	Contínua	Coletor	4	Unix
HFNet	HFNetChk	Pontual	Coletor	4	Windows
EnumNBT	Enum	Pontual	Coletor	4	Windows
EnumSNMP	Athena2k	Pontual	Coletor	4	Independente
RegWatcher	--	Pontual	Coletor	3	Windows
FileWatcher	--	Pontual	Coletor	3	Windows
FwWatcher	--	Pontual	Coletor	2	Independente

Tabela 2 - Sensores e Coletores já desenvolvidos

- *SnortAdmin* – responsável pela interface do sistema com o *Snort*, IDS de rede baseado em assinaturas descrito na seção 2.1.1. Pode ser usado para iniciar e parar a execução do *Snort*, alterar configurações, adicionar e remover assinaturas e, principalmente, para enviar os eventos gerados em formato XML pelo *Snort* para a central de processamentos. Ao ser iniciado o SnortAdmin recebe os parâmetros *EventsPerMessage* e *EventsInterval*, que definem, em conjunto, como os eventos serão enviados para a central. O sensor mantém uma lista de eventos a serem enviados. Se o tamanho dessa lista superar o parâmetro *EventsPerMessage,* os eventos pendentes são imediatamente enviados para a central. Se decorrer um intervalo de tempo superior a *EventsInterval* sem envio, todos os eventos pendentes são despachados. O parâmetro *SendOldEvents* habilita o envio de eventos gerados antes da execução do sensor, possibilitando a execução do Snort independentemente do *SnortAdmin.* Ao ser executado com essa opção habilitada o SnortAdmin envia para a central todos os eventos gerados anteriormente que ainda não foram enviados. Para isso é mantida uma base local contendo uma lista de todos os arquivos de eventos monitorados e os horários (*timestamps*) dos últimos eventos enviados em cada arquivo. O *SnortAdmin* é executado de forma contínua e idealmente deve ser iniciado automaticamente pelo agente logo que esse for iniciado maximizando seu tempo de execução.
- *WIGO* – re-implementação do utilitário *Arpwatch[23]*, monitora requisições e respostas do protocolo ARP e mantém uma base de dados contendo um mapeamento entre endereços IP e endereços MAC na rede monitorada. Permite identificar situações importantes como mudanças de endereço IP, mudança do endereço MAC associado a um IP, conflitos de IP, aparecimento de novas máquinas na rede e volta a atividade de máquinas inativas por muito tempo. Além de contribuir para a manutenção na central do estado ativo da rede pode identificar indícios de ataques de falsificação ARP (ARP *Spoofing*) [31]. Ao identificar novas máquinas na rede o WIGO pode ser configurado para tentar identificar através do protocolo NBT informações como nome da máquina no domínio e usuário ativo. O nome do sensor é um acrônimo para What Is Going On (o que está havendo).
- *NmapInventory* – controla a execução de varreduras de rede utilizando o Nmap[25]. Essas varreduras são solicitadas a partir da central de processamentos e são utilizadas para criação e manutenção do inventário de rede. Embora as requisições sejam realizadas de maneira pontual uma única instância do coletor é mantida em execução contínua permitindo o controle do paralelismo das varreduras e evitando

que uma mesma máquina seja varrida mais de uma vez ao mesmo tempo. Todos os parâmetros de varredura são recebidos na mensagem de requisição e repassados para o utilitário NMAP que é instruído para gerar seus resultados em formato XML que é incorporado à mensagem de notificação de fim de varredura. Além das requisições de varredura o coletor oferece também a possibilidade de requisição de estado das varreduras ativas ou pendentes.

- *NessusInventory* – atua como um cliente *Nessus* [26]solicitando a um servidor previamente instalado a execução de varreduras de vulnerabilidades. Mantém um arquivo de configuração no formato do *Nessus* que pode ser atualizado remotamente pela central. De forma semelhante ao *NmapInventory* controla o paralelismo das varreduras e permite a consulta do estado das varreduras ativas e pendentes. O resultado das varreduras é convertido do formato nativo (NBE) para XML antes de ser enviado para a central onde é utilizado para atualização da base de vulnerabilidades no inventário de rede.
- *HFNet* - adaptador para fazer a interface com o Programa HFNetChk[27], utilitário de linha de comando que permite ao administrador de segurança testar o estado de atualização (*patches* instalados) em sistemas Windows NT 4.0, 2000, XP e 2003. Além de procurar pela existência de *patches* instalados para o próprio sistema operacional o programa HFNetChk também verifica a instalação de correções para o IIS 4.0, IIS 5.0, SQL Server 7.0/2000 (incluindo o MSDE) e Internet Explorer 5 e 6. O coletor recebe a lista de alvos para serem varridos, executa o HFNetChk e transforma os resultados para XML. O alvo pode ser especificado como endereços IP, nomes de máquinas e nomes de domínios NBT. Em alguns casos pode ser necessário o fornecimento de credenciais para acesso remoto.
- *EnumNBT* – através de sessões não autenticadas permite enumerar dados de servidores Windows como: usuários, grupos, compartilhamentos, domínios de confiança e políticas de troca de senhas. A implementação é baseada na ferramenta *Enum* [36] e o seu funcionamento é bastante simples, recebe como entrada o endereço IP da máquina a ser enumerada e retorna todas as informações enumeradas. Quando não é possível enumerar uma informação um erro informativo é retornado.
- *EnumSNMP* – outro enumerador de informações de sistemas Windows. Funciona consultando o servidor SNMP que vem habilitado por padrão em algumas versões dos sistemas NT e 2000. Seu funcionamento é idêntico ao *EnumNBT* mas algumas informações adicionais são retornadas, principalmente relacionadas às interfaces de rede e serviços ativos. Quando necessário pode receber como entrada a comunidade de leitura a ser fornecida nas consultas SNMP.
- *RegWatcher* – monitora sub-árvores do registro de sistemas Windows. Na primeira execução é montada uma representação em XML da árvore monitorada e em execuções subseqüentes o estado é comparado e as diferenças são relatadas para a central. Um dos modos de operação permite que o estado da sub-árvore do registro seja restaurado permitindo desfazer eventuais modificações realizadas como resultado de ataques bem-sucedidos.
- *FileWatcher* – funciona de forma análoga ao *RegWatcher*, monitorando sub-árvores do sistema de arquivos em máquinas Windows. Atua como verificador de integridade detectando mudanças nos conteúdos e permissões de arquivos e diretórios. Também permite restaurar o estado anterior, porém para perfeita restauração é necessário que o coletor crie uma cópia de todos os arquivos monitorados. Mudanças em arquivos e no registro de sistemas Windows podem indicar ataques bem-sucedidos e adicionam bastante valor à análise de incidentes detectados por IDS na rede.

- *FwWatcher* – monitora mudanças em regras e objetos do Checkpoint Firewall-1. Na primeira execução a base de regras atual e os objetos são enviados para a central, em execuções subseqüentes a base anterior é comparada com a atual e as diferenças são relatadas. Para comparação entre bases de regras é utilizado o módulo Algorithm::Diff baseado no algoritmo LCS (*longest common subsequence*). Dadas duas listas o algoritmo descobre a maior seqüência comum entre as listas e computa as diferenças entre os elementos restantes como adições, remoções ou alterações nos itens da lista. Para efeitos de comparação cada regra é transformada em uma string contendo os objetos de cada campo (origem, destino, serviços, ação, etc) em ordem alfabética. Além de monitorar a evolução da base de regras do *firewall* esse coletor permite manter atualizada na central uma representação fiel dessa base de regras possibilitando a melhor identificação de impacto de um determinado ataque quanto às restrições oferecidas pelo filtro de pacotes.

4.5 A Central de Processamentos

A Central de Processamentos foi implementada utilizando a linguagem Java. Essa decisão foi tomada porque esse é um sistema bastante exigido em termos de robustez, escalabilidade, extensibilidade e estabilidade. A linguagem Java rende-se melhor a este tipo de problema, por suas características de orientação a objeto e robustez. Além disso, a decisão pela utilização de um sistema de produção embarcado em uma linguagem de programação orientada a objetos norteou a escolha do JEOPS (descrito na seção 4.6) e colaborou para a adoção de Java para o desenvolvimento módulo de análise.

A central concentra todas as atividades de gerência dos agentes, agregando e integrando as informações colhidas para um repositório central, a partir do qual podem ser realizadas as análises, extraídas as métricas, gerados os relatórios e disparados alarmes. Ela pode ser dividida em quatro grandes subsistemas: transações, operação, informações e interação com usuários. Cada subsistema engloba diferentes tipos de funcionalidades e opera de maneira independente dos outros. A comunicação entre os diferentes subsistemas e entre módulos dentro dos subsistemas é realizada através do uso de Fachadas [32]. Distribuídos por todo o sistema, e sendo utilizados pelos quatro subsistemas, estão presentes os serviços de leitura e geração de mensagens XML, segurança, contendo uma estrutura de controle de acesso baseado em RBAC (*Role Based Access Control 0*), a camada de acesso a dados e o módulo de tratamento de exceções. A Figura 10 mostra as diversas partes do sistema e seus relacionamentos. A fachada central concentra a troca de informações entre os subsistemas facilitando o desacoplamento das partes em caso de necessidade de distribuição do sistema ou em situações que só exijam a implantação de partes do sistema.

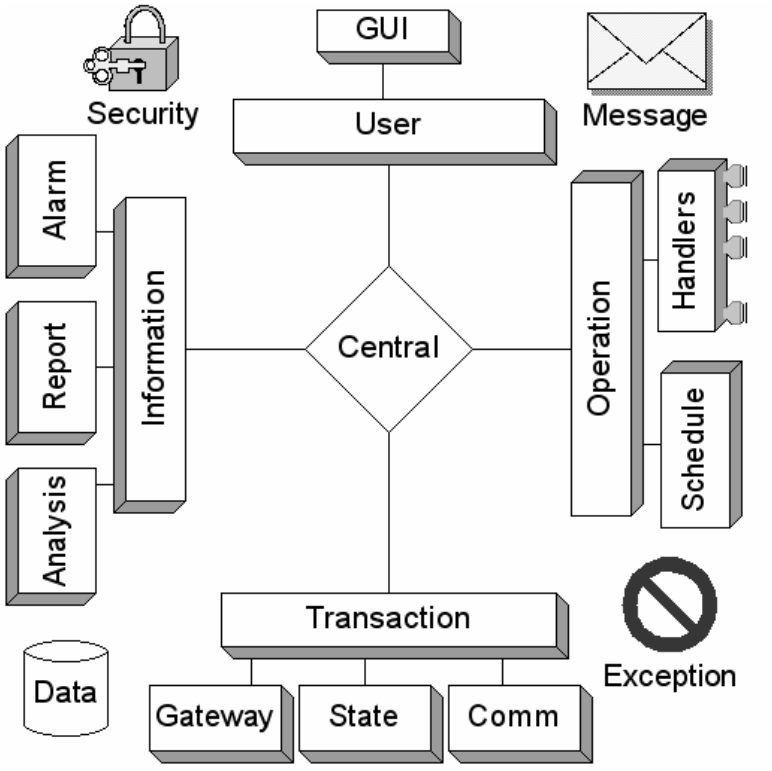


Figura 10 – Estrutura Interna da Central de Processamentos

Os dados que representam o estado da rede assim como os resultados das análises e os eventos devem ser salvos em banco de dados. Só assim esses dados podem ser mantidos após re-inícios do sistema e podem ser usados para geração posterior de relatórios, assim como para a avaliação das análises. Ao mesmo tempo esses dados devem ser acessados pelo módulo de análise da forma mais simples e direta possível. Por esses motivos optou-se pela utilização do Torque [34], uma camada de persistência que oferece os recursos necessários para a aplicação acessar o banco de dados, incluindo classes representando as tabelas e os dados, além de um ambiente de tempo de execução com características como controle de transações e pool de conexões. Algumas vantagens de usar o Torque:

- o processo de criação das tabelas no banco e das classes de acesso é semi-automático, bastando especificar o modelo de dados em XML;
- o próprio processo de criação de esquemas XML representando o modelo de objetos ajuda a criar a documentação do sistema.

O acesso aos dados é bastante facilitado, abstraindo (pelo menos para as operações básicas) a necessidade do conhecimento de SQL, JDBC, etc.

O subsistema de transações é o responsável pela gerência de transações do Matrix. Isso vai desde o gerenciamento de conexões TCP até o processamento de parte do "esqueleto" das mensagens XML que chegam e saem, com o intuito de despachar as mensagens para os módulos corretos, e de guardar informações de estado sobre as mensagens e tarefas em andamento. O subsistema foi dividido em três módulos:

- Módulo de comunicação: responsável pela gerência das conexões através das quais passarão as mensagens entre a central e os vários agentes. A central não somente "escuta" em determinadas portas para receber conexões, mas também as origina para os vários agentes que pode controlar. Este módulo gerencia isso de

forma escalável. Testes em laboratório e implantações iniciais indicaram boa *performance*, mesmo com número de conexões simultâneas (geradas e recebidas) na casa das dezenas de milhares. Este módulo não processa o conteúdo das mensagens XML trafegadas, fazendo apenas uma verificação sintática das mensagens: se uma delas estiver com XML inválido, será descartada;

- Módulo de *gateway*: este módulo é responsável por processar alguns dos elementos básicos do "esqueleto" da mensagem XML, atuando um passo antes do módulo de comunicação no envio das mensagens e um passo depois no recebimento das mensagens. Entre as tarefas realizadas, está ler o elemento dos identificadores únicos das mensagens, e incluí-los no estado através do módulo de manutenção de estado. Quando chegam mensagens dos agentes para a central, este módulo as encapsula em objetos que escondem a sua estrutura, de modo a facilitar o seu processamento no resto do sistema. Na ocasião de mensagens saírem da central para os agentes, o texto XML é extraído do objeto que encapsula a mensagem, para ser enviado através do módulo de comunicação. Além disso, examina o conteúdo dos elementos *Origin* e *Destination*, para verificar que a mensagem está endereçada corretamente e depois fazer o despacho dela, seja para dentro da central, seja para um dos agentes através do módulo de comunicação;
- Módulo de manutenção de estado: o propósito principal deste módulo é de guardar o estado das transações, para posterior consulta. É guardado o estado de cada transação ou pedido de execução, associando a ela os identificadores únicos das mensagens que a compõe (por exemplo, o ID da mensagem requisição e o ID da mensagem resposta). Alimentado pelo módulo de *gateway*, este módulo pode ser consultado para obter uma visão do estado do sistema, vendo que transações estão ativas para que agentes, suas mensagens associadas, e assim por diante.

O subsistema de operações é o responsável pelo controle direto das tarefas, abrigando funcionalidades de *handlers* e de agendamento de tarefas. O subsistema é dividido nos seguintes módulos:

- Módulo de *handlers*: este módulo agrega todas as classes responsáveis pelas regras de negócio dos sensores e coletores nos agentes. Os conceitos aplicados neste módulo são bastante parecidos com os aplicados no agente, em relação ao tratamento de sensores e coletores. Da mesma forma, existe a intenção de facilitar ao máximo o programador através do fornecimento de um *framework* de funcionalidades abrangendo um amplo espectro. Assim, o *handler* também não precisa lidar com mensagens XML diretamente. Na prática, o conteúdo do elemento *Specific* chega a ele através de objetos que encapsulam a estrutura da mensagem, de modo que a mensagem pode ser lida (ou novas mensagens montadas) através de chamadas a métodos neste objeto. Outros serviços são oferecidos pelo *framework* ao *handler*, que envolvem interação com outros módulos/subsistemas. Por exemplo, uma funcionalidade para armazenamento eficiente e simples está disponível, bem como possibilidade de disparo de alarmes, geração de relatórios ou até mesmo a execução de novas tarefas.
- Módulo de agendamento de tarefas: este módulo é o responsável pelo agendamento para execução de tarefas em determinados dias e horários, regulares ou não, provendo uma funcionalidade parecida com o *cron* do Unix. De fato, permite toda a flexibilidade de definição de dias, horas e padrões que o *cron*, só que na verdade dispara pedidos de execução ao módulo de *handler* e a outros subsistemas, visando executar análises e consolidações programadas, por exemplo. O usuário pode

utilizar as funcionalidades providas por esse módulo para cadastrar entradas agendadas, informando os detalhes do que ele quer que seja executado.

O **subsistema de usuários** agrega as funções de interface com os usuários, e através dele, partem boa parte das chamadas para os outros subsistemas. Ele gerencia sessões de diferentes usuários, processos de autenticação e as chamadas para as funcionalidades dos outros subsistemas. As funcionalidades disponibilizadas para os usuários incluem: execução de processos nos agentes, visualização de dados de sensores e coletores ativos, execução de análises e exibição de relatórios, cadastro de agentes, controle de processos ativos na central e interface para agendamento de execuções e análises. As interações dos usuários com essas funcionalidades são realizadas através das interfaces gráficas (GUI). A interface mais madura foi desenvolvida em Java e funciona como console de operação para a central de processamentos permitindo aos operadores a gerência do sistema. A Figura 11 mostra a tela de execução do sensor *SnortAdmin* em um agente na rede de testes.

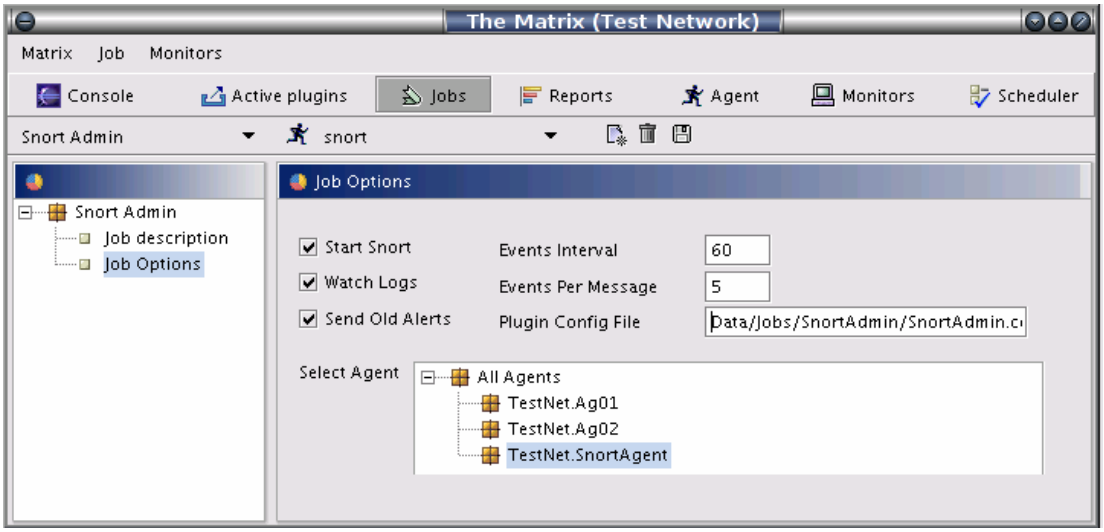


Figura 11 – GUI: Console de Operação do Matrix

O subsistema de processamento de informações agrega o processamento, análise e geração de relatórios a partir dos dados colhidos pelos agentes. Aqui está concentrada boa parte da inteligência do sistema, já que contém as classes para realizar análises históricas e de tendências em cima da massa de dados, geração de relatórios, disparo de alarmes, e correlação de eventos. Este subsistema está dividido nos seguintes módulos:

- Módulo de relatórios: neste existe funcionalidade para geração de relatórios com base nas informações armazenadas pelo módulo de acesso a dados. Existe um conjunto grande de diferentes tipos de relatório, com diferentes graus de detalhe técnico, escopo, perfil, etc. Como os dados colhidos pelos agentes já estão armazenados, o módulo tem à sua disposição uma larga base de informações para compor, permitindo visões integradas da segurança do ambiente. Existir relatórios periódicos, gerados automaticamente e relatórios gerados sob demanda pelos operadores do sistema. Os relatórios disponibilizados por esse módulo são indexados e armazenados provendo também funcionalidade para consultas e buscas.
- Módulo de alarmes: este módulo gerencia toda a parte de disparo de alarmes da central. Podem existir várias causas para disparar um alarme, como, por exemplo, um

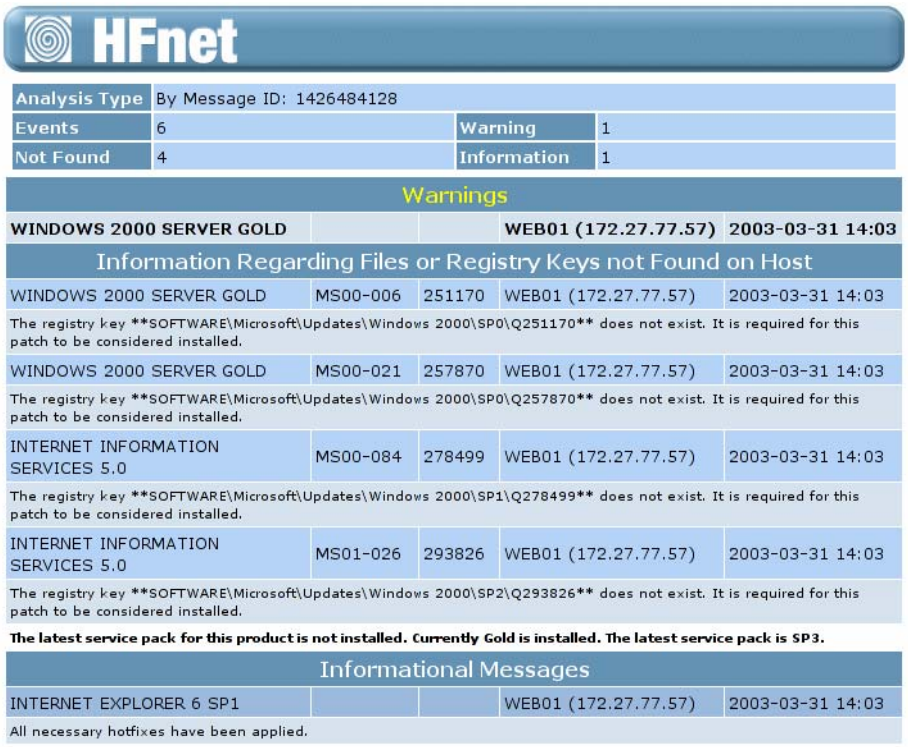
evento detectado por um sensor que é processado pelo correspondente *handler* na central. Neste módulo estão configurados os vários tipos de alarmes possíveis (email, pager, celular, visual/sonoro, etc.), os seus parâmetros particulares (endereço de e-mail, número de celular, etc.), e os eventos que fazem determinado grupo de alarmes disparar (por ex., evento de IDS na rede *X* disparará alarmes de e-mail e celular para o grupo e pessoas *Y*). Além de prover esta interface de configuração, o módulo recebe os alarmes e age de acordo com as regras configuradas, registrando de forma persistente os acontecimentos.

- Módulo de análise: elemento vital da arquitetura, este módulo agrega diferentes tipos de análises em cima da massa de dados colhida pelos agentes e armazenada pela central. Nele são processados os eventos detectados pelos sensores, que agrupa-os e classifica-os através de sistemas de produção. Durante a análise pode ser necessário acionar novos coletores em pontos distintos da rede para adicionar à base de conhecimento informações vitais para a realização da análise. Dada sua complexidade e importância o módulo de análise será mais bem detalhado na próxima seção.

4.6 Análises e Correlações

Uma vez processados e armazenados pelos *handlers*, os dados podem ser analisados de forma a melhor caracterizar a severidade dos eventos detectados e a identificar mudanças significativas no estado da rede que justifiquem atualizações no inventário de rede.

De maneira análoga ao funcionamento dos sensores e coletores e dos seus respectivos *handlers*, as análises podem ser pontuais ou contínuas. **As análises pontuais** normalmente são utilizadas para a geração de relatórios com os resultados da execução de uma determinada tarefa ou para consolidar e sumarizar resultados correlatos obtidos em um determinado intervalo de tempo. Exemplo desse tipo de análise são as realizadas sobre dados coletados pelo HFNet. Uma das análises mostra os resultados coletados pela ferramenta em uma determinada execução mostrando, para aquele instante, a necessidade de atualizações em um servidor Windows (Figura 12). A outra análise faz uma comparação entre duas execuções arbitrárias da ferramenta, enfatizando as mudanças ocorridas no intervalo. As mudanças indicam correções que foram realizadas no período, novas atualizações que precisam ser instaladas e atualizações antigas que ainda precisam ser instaladas (Figura 13). Outros tipos de análises pontuais incluem: mudanças em bases de regras de *firewalls*, novas máquinas adicionadas ao inventário em um intervalo de tempo, gráfico de uso de CPU de um determinado servidor, entre outras. Os dados utilizados em análises pontuais são obtidos do banco de dados utilizando como chaves de pesquisas os identificadores das mensagens que os geraram ou as datas em que foram executadas. Ao fim da execução de um coletor, um sumário dos resultados é apresentado ao usuário que a requisitou, oferecendo a oportunidade de executar uma análise pontual para visualizar os resultados completos.



HFnet

Analysis Type	By Message ID: 1426484128		
Events	6	Warning	1
Not Found	4	Information	1

Warnings

WINDOWS 2000 SERVER GOLD			**WEB01 (172.27.77.57)**	**2003-03-31 14:03**

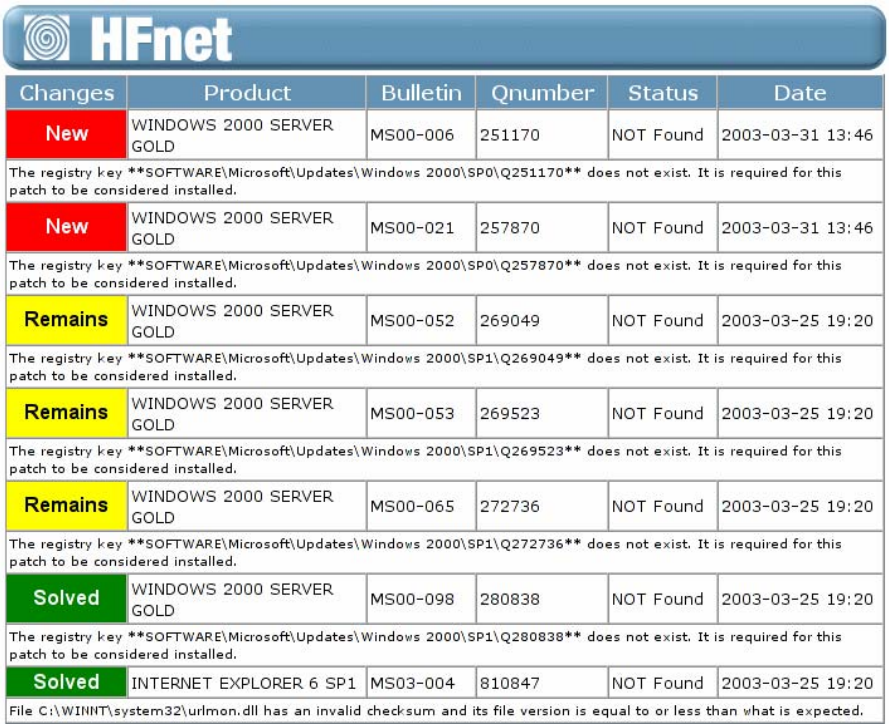
Information Regarding Files or Registry Keys not Found on Host

WINDOWS 2000 SERVER GOLD	MS00-006	251170	WEB01 (172.27.77.57)	2003-03-31 14:03
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP0\Q251170** does not exist. It is required for this patch to be considered installed.				
WINDOWS 2000 SERVER GOLD	MS00-021	257870	WEB01 (172.27.77.57)	2003-03-31 14:03
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP0\Q257870** does not exist. It is required for this patch to be considered installed.				
INTERNET INFORMATION SERVICES 5.0	MS00-084	278499	WEB01 (172.27.77.57)	2003-03-31 14:03
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP1\Q278499** does not exist. It is required for this patch to be considered installed.				
INTERNET INFORMATION SERVICES 5.0	MS01-026	293826	WEB01 (172.27.77.57)	2003-03-31 14:03
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP2\Q293826** does not exist. It is required for this patch to be considered installed.				
The latest service pack for this product is not installed. Currently Gold is installed. The latest service pack is SP3.				

Informational Messages

INTERNET EXPLORER 6 SP1			WEB01 (172.27.77.57)	2003-03-31 14:03
All necessary hotfixes have been applied.				

Figura 12 - Análises de Uma Execução do HFNet



HFnet

Changes	Product	Bulletin	Qnumber	Status	Date
New	WINDOWS 2000 SERVER GOLD	MS00-006	251170	NOT Found	2003-03-31 13:46
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP0\Q251170** does not exist. It is required for this patch to be considered installed.					
New	WINDOWS 2000 SERVER GOLD	MS00-021	257870	NOT Found	2003-03-31 13:46
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP0\Q257870** does not exist. It is required for this patch to be considered installed.					
Remains	WINDOWS 2000 SERVER GOLD	MS00-052	269049	NOT Found	2003-03-25 19:20
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP1\Q269049** does not exist. It is required for this patch to be considered installed.					
Remains	WINDOWS 2000 SERVER GOLD	MS00-053	269523	NOT Found	2003-03-25 19:20
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP1\Q269523** does not exist. It is required for this patch to be considered installed.					
Remains	WINDOWS 2000 SERVER GOLD	MS00-065	272736	NOT Found	2003-03-25 19:20
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP1\Q272736** does not exist. It is required for this patch to be considered installed.					
Solved	WINDOWS 2000 SERVER GOLD	MS00-098	280838	NOT Found	2003-03-25 19:20
The registry key **SOFTWARE\Microsoft\Updates\Windows 2000\SP1\Q280838** does not exist. It is required for this patch to be considered installed.					
Solved	INTERNET EXPLORER 6 SP1	MS03-004	810847	NOT Found	2003-03-25 19:20
File C:\WINNT\system32\urlmon.dll has an invalid checksum and its file version is equal to or less than what is expected.					

Figura 13 Análises de diferenças entre duas execuções do HFNet

As análises contínuas estão permanentemente em execução e são constantemente atualizadas com informações coletadas e com eventos detectados nos sensores. Cada um desses conjuntos de dados é classificado quanto à sua origem e tipo e encaminhado para as análises pertinentes. Tanto o processo de classificação quanto as análises em si são baseados no uso de sistemas especialistas que utilizam regras de produção que disparam ações quando o seu conjunto de premissas é satisfeito. As análises contínuas podem ser realizadas sobre dados armazenados no banco ou sobre

aqueles recém processados pelos *handlers* e tipicamente disponibilizados em estruturas de dados chamados de *buffers* circulares. Essas estruturas mantêm os últimos *N* elementos processados pelos *handlers*, uma ou mais análises podem então ler os elementos à medida que eles são armazenados, produzindo resultados à medida que os dados são coletados e processados. Quando utilizando dados armazenados no banco, as análises periodicamente realizam consultas que retornam todos os dados ainda não analisados. Essa abordagem pode produzir resultados mais lentos já que o tempo entre consultas ao banco tende a ser maior que o de leitura dos *buffers*. Por outro lado, os dados armazenados no banco estão sempre acessíveis e não estão limitados ao tamanho do buffer circular. Para maximizar a performance sem perda de informações as análises normalmente são configuradas para, inicialmente, ler todos os dados ainda não analisados presentes no banco e, em seguida, ler os dados dos *buffers.* A Figura 14 ilustra o fluxo de dados desde a coleta nos agentes até a análise:

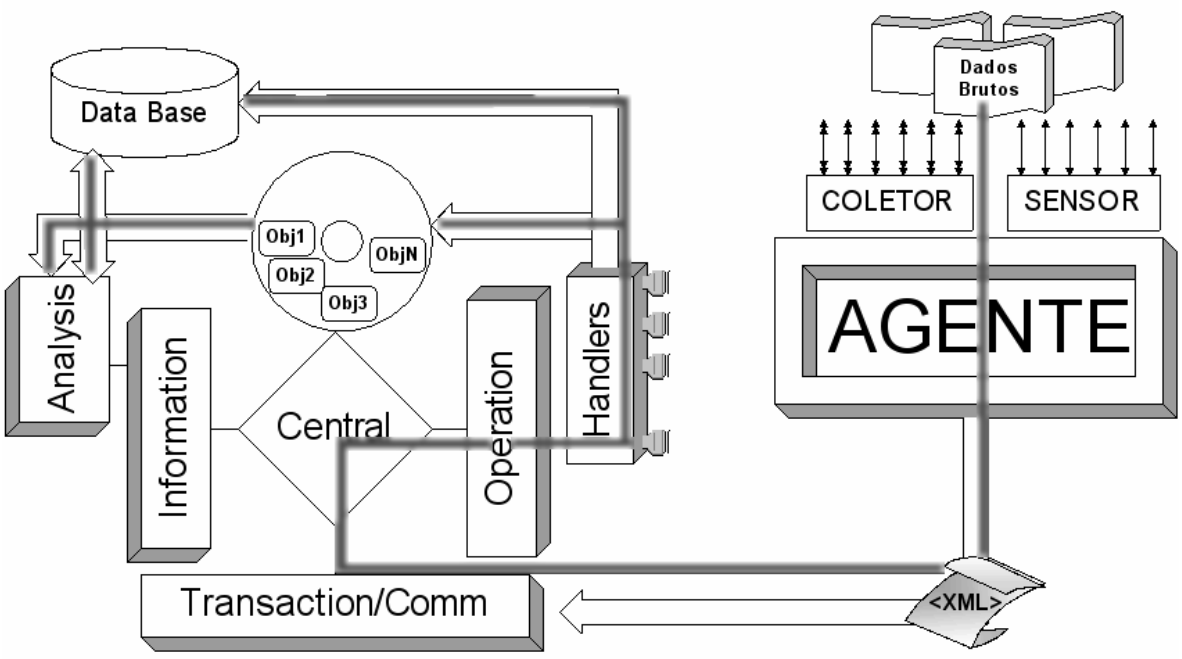


Figura 14 - Fluxo de dados desde a coleta até a análise

4.6.1 O Motor de Inferência (JEOPS)

O módulo de análise foi implementado utilizando regras de produção para agrupar e classificar os eventos. Para isso foi utilizado o JEOPS (Java Embedded Object Production System) [35], um motor de inferência de primeira ordem, com encadeamento progressivo, que visa prover capacidade de raciocínio à linguagem Java. Inicialmente foi utilizada a versão 1.0 do JEOPS devido à possibilidade de utilização de regras em arquivos texto, possibilitando protótipos e testes rápidos. À medida que as regras forem se consolidando haverá uma migração natural para as versões mais recentes do JEOPS, que permitem a compilação das regras de produção, além de permitir uma maior integração com os tipos de dados de Java.

Para auxiliar na tomada de decisão o JEOPS utiliza uma base de conhecimentos que contém um conjunto de regras definidas pelo usuário, um conjunto de objetos

referenciados pelas regras, uma estratégia de resolução de conflitos e o algoritmo de casamento de padrões Rete [43]. As regras do JEOPS são formadas por três blocos: declarações, premissas e ações. As declarações indicam quais objetos da base de conhecimento serão utilizadas naquela regra. As premissas são expressões em Java, normalmente envolvendo os elementos da base de conhecimento e que retornam valores booleanos. A forma geral de uma regra JEOPS é a seguinte:

```
rule nomeDaRegra {
declarations
 caminho.da.Classe1 objeto1;
 caminho.da.Classe2 objeto2;
preconditions
 objeto1.metodoBooleano1();
 objeto1.getValor() > 3;
actions
 netState.newIPAddress(ev.mac(),ev.ip());
 objeto1.acao1();
 objeto2.outraAcao("parametro",123);
 acao_do_jeops(objeto1);
}
```

Se todas as premissas forem satisfeitas as ações das regras são executadas. Pode acontecer de mais de uma regra estar apta a ser disparada ao mesmo tempo. Nesses casos, a estratégia de resolução de conflitos é utilizada para desempate.

A característica mais interessante, no entanto, é o encadeamento progressivo: a ação de uma regra pode alterar o estado de objetos na base de conhecimentos fazendo com que pré-condições de outras regras sejam satisfeitas. Eventos e dados coletados em fontes distintas podem desencadear o disparo de regras possibilitando a correlação de informações fornecidas por ferramentas diversas.

4.6.2 Montando o Inventário de Rede

O inventário contém informações sobre os elementos da rede, mapeando suas características, serviços oferecidos e identificações. A idéia é agregar o máximo possível de informações sobre os elementos da rede, vindo de diferentes fontes, no inventário. Neste sentido, o inventário é alimentado por vários coletores, provendo uma visão detalhada e completa dos elementos da rede. Essa visão é mais abrangente do que aquelas fornecidas por uma única ferramenta.

As principais fontes de informações para alimentação do inventário de rede são os coletores *NmapInventory* (que utiliza o utilitário NMAP para realizar varreduras de portas) e *NessusInventory* (que utiliza o *Nessus* para realizar varreduras de vulnerabilidades). A agregação dos dados fornecidos por esses coletores é efetuada na central de processamentos através de uma análise contínua que periodicamente consulta o banco de dados em busca de máquinas que estão com suas informações de inventário desatualizadas e dispara execuções dos coletores para validar esses dados. Ao receber o resultado da execução, o sistema verifica se houve mudanças entre a varredura armazenada e a recém realizada. Se houve mudanças, o inventário é atualizado com as novas informações e as antigas são armazenadas para manutenção de histórico. No caso de não haver mudanças a data de última atualização é modificada para o horário da varredura, evitando o disparo desnecessário de varreduras no próximo intervalo de atualização.

Além das varreduras periódicas disparadas pelo processo descrito acima, existem as varreduras de descoberta que são realizadas com menor freqüência com o objetivo de encontrar máquinas que ainda não foram adicionadas ao inventário. Essas varreduras de descoberta normalmente abrangem todo o espaço de endereços IP da rede monitorada e tendem a ser mais demoradas. Para evitar demoras excessivas os parâmetros dessas varreduras são ajustados reduzindo o número de portas varridas e utilizando as diversas técnicas de teste de conectividade fornecidas pelo utilitário NMAP. Por esse mesmo motivo as varreduras *Nessus* não são realizadas para descoberta de novas máquinas já que tendem a ser muito demoradas. Outra forma de descoberta de novos ativos é a monitoração de tráfego ARP conforme veremos na próxima seção.

Além de, por si só, constituir um produto interessante para o cliente, o inventário de rede é utilizado durante as análises para se ter o estado ativo do ambiente, permitindo correlações entre eventos detectados nos sensores e as informações armazenadas no banco. Por exemplo: ao perceber um padrão de ataque conhecido, o detector de intrusão gera um evento para a central contendo endereços e portas de origem e destino e outras informações pertinentes. A análise desse evento consulta o inventário de rede para verificar se o alvo do ataque existe, se o serviço atacado é oferecido e se foram detectadas vulnerabilidades associadas a esse serviço, classificando o evento conforme o impacto. O processo de análise de eventos do IDS *Snort* é melhor detalhado na seção 4.6.4.

4.6.3 Análise de Tráfego ARP

O protocolo ARP pode ser visto como uma camada de adaptação entre as camadas física e de rede realizando um mapeamento dinâmico entre endereços IP de 32 bits e endereços *Ethernet* de 48 bits. Para transmitir um pacote IP encapsulado em um quadro *Ethernet* o sistema operacional consulta sua tabela ARP e verifica se já existe um mapeamento entre o IP de destino e um endereço MAC. Se não existir o mapeamento, uma mensagem de requisição ("*who-has*") é enviada em difusão (para todas as máquinas da rede). Se alguma máquina for responsável por aquele endereço uma resposta ("*is-at*") é enviada para o solicitante informando o endereço MAC do destino. Essa informação é então utilizada para compor o quadro *Ethernet* e é armazenada na tabela ARP do sistema por algum tempo para evitar consultas futuras.

Conforme descrito na Seção 4.4 o sensor WIGO monitora tráfego ARP na rede e mantém uma tabela própria com histórico de requisições passadas. Cruzando essas informações ele consegue identificar eventos de interesse, como: mudanças de endereço MAC associado a um IP, novas máquinas na rede, conflitos de IP e volta a atividade de máquinas silenciadas há muito tempo. Ao receber esses eventos a análise do WIGO pode tomar algumas ações como:

- Atualizar o Inventário de Rede – quando ocorrem mudanças de endereço IP em máquinas que já estavam presentes no inventário, o WIGO normalmente identifica o novo endereço IP como uma nova máquina e envia o evento "New Station". Ao receber esse evento o sistema de produção avalia se o endereço MAC já estava no inventário, se o endereço IP ainda não existia e se não houve eventos recentes associados a esses endereços (IP e MAC). Satisfeitas todas essas pré-condições o novo endereço IP é adicionado à lista de IPs da máquina. Na próxima varredura dessa máquina esse endereço também será usado como alvo. Em máquinas com múltiplos

endereços, se algum deles não responder numa mesma varredura, ele é removido. A regra de produção abaixo é utilizada para identificar mudanças de endereço. Uma fila de eventos é mantida pelo sistema com objetivo de identificar situações de conflito de endereços, que normalmente disparam vários eventos.

```
rule ipAddressChange {
declarations
 db.inventory.NetInfo netState;
 db.wigo.Event ev;
 db.wigo.EventQueue queue;
preconditions
 ev != null;
 ev.isNewStation();
 ev.age() > 300;
 netState.macExists(ev.mac());
 !netState.ipExists(ev.ip());
 netState.getIP(ev.mac) != ev.ip();
 queue.countRelatedEvents(ev) == 0;
actions
 netState.newIPAddress(ev.mac(),ev.ip());
 queue.remove(ev);
 retract(ev);
}
```

- Detectar indícios de *ARP Spoofing* - O ataque de *ARP Spoofing* consiste na manipulação do mapeamento IP/MAC das vítimas forçando o envio do tráfego para o atacante possibilitando a interceptação e manipulação do conteúdo dos pacotes. Para isso o atacante envia para as vítimas respostas de requisições não realizadas, associando o seu endereço com o IP de outras máquinas da rede, tipicamente os dos roteadores [31]. Essas respostas ao passarem pelo sensor do WIGO disparam o evento de mudança de endereço MAC ou o evento de conflito de IP (conhecido como FLIP-FLOP). Ao receber esses eventos associados a endereços IP de roteadores, a análise do WIGO dispara alertas de *ARP Spoofing*. A regra a seguir é utilizada com essa finalidade.

```
rule possibleArpSpoofing {
declarations
 db.inventory.NetInfo netState;
 db.wigo.Event ev;
 db.wigo.EventQueue queue;
preconditions
 ev != null;
 ev.isFlipFlop() || ev.isChangedMac();
 netState.isRouter(ev.ip());
actions
 netState.arpSpoofingDetected(ev);
 retract(ev);
}
```

4.6.4 Análise de Eventos de IDS

À medida que os eventos são gerados pelos sensores e recebidos pela central de processamentos, as análises são realizadas para priorizar os eventos quanto ao impacto no ambiente. Essa priorização é realizada em quatro etapas: Descarte de

Falsos Positivos; Priorização quanto a Severidade (impacto), Agrupamento Lógico de Eventos e Validação Operacional.

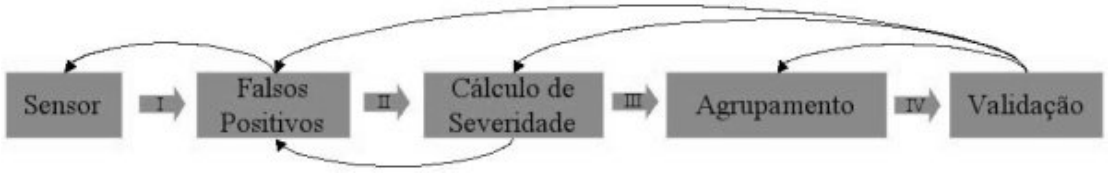


Figura 15 - Etapas na priorização de eventos do Snort

Fase I - Descarte de Falsos Positivos

A etapa de descarte de falsos positivos consiste em identificar eventos que não representam ameaça à rede monitorada. Esses eventos são disparados por tráfego legítimo que possui características semelhantes a ataques conhecidos e que casam com o padrão de uma ou mais assinaturas. Quando esse tráfego é caracterizado é possível criar regras para descartar os eventos gerados por ele, reduzindo assim o número de alertas a serem analisados nas etapas posteriores e validados pelos operadores do sistema. Os eventos descartados são guardados em uma estrutura de tabelas no banco independente das tabelas de eventos válidos e não são exibidos na interface de operação. Alguns exemplos de tráfego legítimo que tipicamente disparam alertas no IDS são mostrados a seguir:

- Varreduras originadas em máquinas confiadas – o contínuo processo de monitoramento que alimenta o inventário de rede inclui varreduras de portas e de vulnerabilidades. Dependendo do posicionamento do sensor e de sua configuração essas varreduras podem disparar eventos de intrusão. Na central esses eventos são descartados através de uma regra que consulta a lista de máquinas autorizadas a realizar varreduras. A regra para esse descarte é mostrada a seguir:

```
rule isSourceAllowed {
declarations
 db.netinfo.NetInfo netState;
 db.snort.Event ev;
preconditions
 netState.allowedSource(ev.getSource());
actions
 event.setDiscardReason("ALLOWED_SOURCE");
 event.setFiltered();
 retract(ev);
}
```

- Resposta de pacotes web requisitados pelo proxy – muitos falsos positivos são gerados por tráfego gerado como resposta a requisições realizadas pelos servidores proxy. Esses pacotes podem ser identificados por três características peculiares: o endereço de destino é o do proxy, a porta de origem é um serviço web (80 ou 443) e a porta de destino é uma porta alta, que não está presente no inventário de rede do servidor proxy. A regra abaixo identifica esse tipo de evento e os descarta:

```
rule proxyTraffic {
declarations
 db.netinfo.NetInfo netState;
```

```
 db.snort.Event ev;
preconditions
 netState.isProxy(ev.getTarget());
 netState.isWebPort(ev.getSourcePort());
 netState.serviceNotOffered(ev.getTarget(),ev.getServic
e());
actions
 event.setDiscardReason("PROXY_TRAFFIC");
 event.setFiltered();
 retract(ev);
}
```

Uma outra classe de falsos positivos diz respeito a assinaturas muito genéricas, que são constantemente ativadas pelo tráfego normal da rede. Os eventos gerados por essas assinaturas também podem ser descartados antes das outras etapas da análise. Alguns exemplos de assinaturas excessivamente genéricas:

- Trafego HTTP contendo ../
- Trafego HTTP contendo ///
- Pipes no conteúdo de arquivos binários

A regra abaixo descarta eventos gerados como resposta a assinaturas excessivamente genéricas. Essas assinaturas também poderiam ser desabilitadas diretamente no sensor, conforme será visto mais adiante na seção de realimentação.

```
rule noisySignature {
declarations
 db.netinfo.NetInfo netState;
 db.snort.Event ev;
preconditions
 netState.isNoisySignature(ev.getSigId());
actions
 event.setDiscardReason("NOISY_SIGNATURE");
 event.setFiltered();
 retract(ev);
}
```

Em alguns casos as assinaturas genéricas só são ativadas como resposta a um tipo de fluxo de dados muito freqüente na rede monitorada. Nesses casos, pode-se criar regras para descartar apenas os eventos disparados em resposta a esse tipo de fluxo sem descartar o mesmo tipo de evento para outros fluxos.

Fase II - Cálculo de Severidade

Os eventos não descartados podem então ser classificados quanto ao impacto que podem causar no sistema. Através da correlação com o estado da rede, cada evento recebe um escore de severidade que é montado através da seguinte fórmula:

$$\textbf{Severidade} = 10^4 \textbf{ . Vuln + Pri } (10^2 \textbf{ . Serv + Ativo)} \text{ , onde:}$$

Pri: Prioridade definida para esse evento na base de assinaturas do *snort*
Vuln: Define o grau de vulnerabilidade do serviço ao ataque (valores 0 ou 9)
Serv: Define se o serviço é oferecido na máquina alvo (valores 0 ou 1)
Ativo: Define se a máquina alvo está ativa na rede (valores 0 ou 1)

O uso das potências de 10 na equação acima gera escores que permitem, sem necessidade de cálculos adicionais, identificar se os critérios de vulnerabilidade, serviço oferecido e servidor ativo foram satisfeitos e qual a prioridade do evento original do *Snort*. Por exemplo, uma severidade de 51818 diz que a máquina alvo está ativa e o serviço é oferecido com grau de vulnerabilidade 5. Além disso sabe-se que a prioridade da assinatura original era 18. Os escores de todos os eventos de uma análise são somados criando um escore geral da análise. As análises são mostradas para o usuário pela ordem decrescente dos escores totais. As análises e os eventos são também identificados em uma escala de cores variando do verde ao vermelho, identificando visualmente a gravidade do ataque.

As regras de análise dos eventos que ajudam a definir as severidades são simples e procuram codificar o conhecimento de um especialista e reproduzir suas ações para priorizar os eventos mais graves. Basicamente elas verificam se as condições para definir se uma máquina está ativa, se um serviço é oferecido e seu grau de vulnerabilidade. Uma particularidade dessas regras é que existe um encadeamento, ou seja, as conclusões de uma regra são utilizadas nas premissas de outras, tornando a análise mais complexa do que o agrupamento. A estratégia de resolução de conflitos adotada foi a “OneShot” que diz que uma regra só pode ser disparada uma única vez.

A regra abaixo, por exemplo, faz uma consulta ao estado da rede para identificar se o alvo faz parte da rede protegida. Quando acionada ela atualiza o impacto do evento:

```
// Alvo faz parte da rede protegida
rule isValidTarget {
declarations
 db.netinfo.NetInfo netState;
 db.snort.Event ev;
preconditions
 ev != null;
 netState != null;
 netState.servExists(ev.getTarget());
actions
 event.setValidTarget(true);
 event.updateSeverity();
 modified(ev);
}
```

Já a regra abaixo verifica se o serviço do evento é oferecido pelo servidor alvo do suposto ataque:

```
// Verifica se o servico é oferecido
rule isServiceOffered {
declarations
 db.netinfo.NetInfo ni;
 db.snort.Event e;
preconditions
 e != null;
 ni != null;
 e.isTargetValid();
 e.hasPortInfo();
 ni.offered(e.target(),e.dport());
actions
 event.setServiceOffered(true);
 event.updateSeverity();
 modified(event);
}
```

O próprio processo de verificação de precondições em regras pode automaticamente ativar coletores para complementar o estado ativo da rede. Isso acontece nos casos em que não existem dados suficientes no estado para testar as precondições ou quando as informações estão desatualizadas.

Outras regras são utilizadas para classificar o grau de vulnerabilidade dos serviços alvos do ataque e são codificadas de forma a reproduzir o comportamento do especialista no processo de priorização. Por exemplo, a regra abaixo verifica se o serviço está na sua versão mais atual, aumentando o grau de vulnerabilidade em casos negativos:

```
rule isServiceOutdated {
declarations
 db.netinfo.NetInfo ni;
 db.snort.Event e;
preconditions
 e.isServiceOffered();
 ni.notLatest(e.target(),e.dport());
actions
 event.updateVulnerability(2);
 modified(event);
}
```

Conforme mencionado anteriormente, o grau de vulnerabilidade varia entre 0 e 9 e é modificado à medida que regras relacionadas a ele são ativadas no sistema de produção. A regra "***serviceHasKnownVulnerability"*,** abaixo, aumenta o nível de vulnerabilidade para serviços que possuem alguma vulnerabilidade conhecida e que foi anteriormente detectada nas varreduras de alimentação do inventário de rede. Já a regra "***serviceVulnerableToAttack"*** é disparada quando o serviço alvo é vulnerável especificamente ao tipo de ataque detectado no evento. Essa correlação é feita através das referências externas (CVE, Bugtraq, SANS, etc) presentes tanto no evento quanto no inventário de rede.

```
rule serviceHasKnownVulnerability {
declarations
 db.netinfo.NetInfo ni;
 db.snort.Event e;
preconditions
 e.isServiceOffered();
 ni.serviceHasVulnerability(e.target(),e.dport());
actions
 event.updateVulnerability(5);
 modified(event);
}
rule serviceVulnerableToAttack {
declarations
 db.netinfo.NetInfo ni;
 db.snort.Event e;
preconditions
 e.isServiceOffered();
ni.serviceHasReference(e.target(),e.dport(),e.getReference());
actions
 event.updateVulnerability(9);
 modified(event);
}
```

Fase III - Agrupamento Lógico de Eventos

Depois de classificados quanto à sua severidade, os eventos são agrupados de acordo com critérios como endereços de origem e destino, tipos de ataques, etc. Esse agrupamento é feito através de regras simples de produção. Eventualmente, um mesmo evento pode participar de mais de um grupo. O agrupamento dos eventos facilita o trabalho dos operadores que podem mais facilmente identificar conjuntos de eventos a serem descartados ou destacados para análise mais detalhada.

No módulo de análise, a base de regras de agrupamento dos eventos é simples, sendo formadas por poucas regras de produção, algumas delas listadas abaixo. A primeira regra agrupa os eventos que contém a mesma origem e mesmo destino. Ao processar um evento, o sistema verifica se existem análises ativas com a mesma origem e destino desse evento. Se existirem o evento é adicionado à essa análise:

```
rule sameSourceAndTarget {
declarations
 analysis.snort.EventClassifier ec;
 db.snort.Event ev;
preconditions
 ev != null;
 ec.byPeers(ev.source(),ev.target());
actions
 ec.addEventToGroup(ev);
 modified(ec);
}
```

Uma segunda regra trata do caso específico das varreduras (port scans), que nunca possuem o destino definido. Nesses casos o evento é adicionado a todas às análises que possuem eventos com a mesma origem da varredura:

```
rule eventIsPortScan {
declarations
 analysis.snort.EventClassifier ec;
 db.snort.Event ev;
preconditions
 ev != null;
 ev.target().equals("Port Scan");
actions
 ec.addPortScanToGroups(ev);
 modified(ec);
}
```

A última regra é disparada quando o evento não foi agrupado em nenhum dos outros critérios. Nesse caso é criado um novo grupo e o evento é adicionado a ele:

```
rule default {
declarations
 analysis.snort.EventClassifier ec;
 db.snort.Event ev;
preconditions
 ec != null; event != null;
actions
 ec.createGroup(event);
 retract(event);
 modified(ec);
}
```

A estratégia de resolução de conflitos para essa base de regras foi a de prioridade, ou seja, ordem de definição da regra. Assim, a ordem em que as regras aparecem no arquivo define qual vai ser disparada em casos de conflitos.

Fase IV - Validação Operacional

Uma vez pré-classificados e agrupados os eventos não descartados são disponibilizados para a equipe de operação que fica responsável por validar a classificação realizada pelo sistema.
Os eventos aparecem na interface de operação em uma tela de pendências. Todo evento pré-classificado pela central de processamentos precisa ser validado pelos operadores, garantindo assim que eventuais falhas na classificação possam ser corrigidas manualmente. Para facilitar a visualização dos eventos o operador pode utilizar filtros e agrupamentos por vários critérios como endereços e portas de origem e destino, classes de assinaturas, etc. Eventos considerados falsos positivos podem ser ignorados nessa etapa e os pesos calculados pelas regras de produção podem ser ajustados.
O passo de validação é essencial para o bom funcionamento do sistema, já que ajuda a ajustar o processo de classificação automática e a descobrir novas regras para descarte trivial de falsos positivos e para agrupamento de eventos.

Realimentação
Uma característica fundamental para o bom funcionamento do sistema é a possibilidade de realimentação entre as diversas fases de classificação.

Eventos descartados na Fase I podem ser filtrados diretamente no sensor, evitando o tráfego adicional na rede e o consumo de recursos na central. para processamento e armazenamento.
Falsos positivos repetidamente identificados durante a validação manual podem servir de base para a criação de novas regras de descarte automático, evitando o trabalho repetitivo de descarte manual.
As regras para cálculo de impacto dos eventos devem sempre refletir o comportamento do especialista em detecção de intrusos, tornando o trabalho manual da Fase IV cada vez menor.

4.6.5 Outras Análises

Cada um dos coletores e sensores listados na Tabela 2 possui análises associadas. Algumas dessas análises são realizadas de forma pontual, normalmente sendo disparadas pelos operadores do sistema para obter mais informações sobre o estado de um servidor monitorado. Nesta seção são listadas as análises com essas características que não foram detalhadas nas seções anteriores.

O coletor ***FwWatcher*** é executado periodicamente e monitora mudanças em regras e objetos do Checkpoint Firewall-1. Além de ajudar as equipes de operação no acompanhamento de mudanças nas bases de regras e identificação de situações perigosas, as análises sobre os dados gerados por esse coletor permitem ao operador visualizar o estado da base de regras em qualquer instante anterior à última execução, assim como listar, para esse instante, as regras que envolvem um determinado endereço ou serviço. Essa informação pode ser vital para analisar o impacto de um evento detectado, por exemplo, no IDS de rede.

O ***HFNet*** permite testar o estado de atualização (*patches* instalados) em sistemas Windows. Servidores críticos podem ser monitorados com esse coletor para fornecer relatórios periódicos de necessidade de atualizações. As análises sobre os dados armazenados permitem saber quais atualizações estavam pendentes no momento em que ocorreu um incidente ou uma suspeita de intrusão. Outra análise importante é a comparação entre execuções arbitrárias do coletor identificando quais problemas foram sanados naquele intervalo de tempo e quais atualizações ainda precisam ser instaladas. O resultado dessa comparação reflete a atenção que está sendo prestada ao servidor e pode gerar sugestões de ações corretivas.

Os **enumeradores NBT e SNMP** são utilizados para adicionar informações pertinentes ao inventário de rede quando se detecta a possibilidade de extração remota dessas informações. Quando a análise de uma varredura NMAP detecta que uma máquina permite acesso as portas utilizadas por esses coletores, eles podem (através da interface do sistema) serem agendados para periodicamente serem disparados. Quando o resultado de uma execução é recebido, uma análise é imediatamente executada para processar os resultados e atualizar o inventário de rede.

Já os monitoradores de registro (***RegWatcher***) e sistemas de arquivos no Windows (***FileWatcher***) podem, sob requisição do operador, verificar se houve mudanças em partes importantes do registro ou do sistema de arquivos, possibilitando avaliar o impacto de um ataque detectado. Por exemplo: existem alertas de IDS que informam sobre a suspeita de infecção de máquinas na rede por vírus. Ao ver esse alerta o operador pode disparar esses coletores, configurando-os para verificar partes específicas afetadas pelo vírus (ou *worm*) em questão.

4.7 Síntese

O PIPS foi implementado como parte de um sistema de co-administração de segurança conhecido como *Matrix*. Os agentes foram implementados utilizando a linguagem *Perl* e através de canais de comunicação seguros trocam mensagens XML com a central de processamentos. A central gerencia as informações coletadas nos diversos agentes e realiza análises correlacionando essas informações. As análises contínuas são viabilizadas através do uso do sistema JEOPS, que adiciona à linguagem *Java* a capacidade de raciocínio através de um motor de inferência de primeira ordem com encadeamento progressivo.
Para coleta dos dados foram desenvolvidos dez *plugins*, dois dos quais, atuam como sensores e oito como coletores. Note-se que apesar de utilizar ferramentas com licenças GPL, não existe compartilhamento do código-fonte, de modo que não se incute em obrigatoriedade da distribuição do sistema utilizando a GPL.

5 Resultados Obtidos

Desde a concepção até a implantação nas primeiras redes reais o desenvolvimento do sistema sempre foi voltado para resultados práticos, buscando automatizar ao máximo as tarefas repetitivas de co-administração de segurança de sistemas. A principal característica do sistema é o foco na proatividade em detrimento da reatividade: através da identificação de pontos vulneráveis na rede é possível antecipar ataques, tomando atitudes corretivas antes que o problema maior aconteça. Nesse contexto, os eventos de IDS são úteis principalmente para justificar esforço gasto na correção de vulnerabilidades e para ajudar no processo de levantamento de risco, através da identificação dos principais alvos de tentativas de ataques.

5.1 Implementação de Framework Distribuído

Para melhor funcionamento do sistema vários agentes devem ser posicionados em pontos estratégicos da rede monitorada de forma a obter resultados mais precisos. O framework desenvolvido possibilita a distribuição dos agentes, permitindo que cada um dos sensores e coletores exerça com maior perfeição suas tarefas. A utilização de canais cifrados e tunelados em conexões TCP facilita sua instalação em redes protegidas por *firewalls*, não havendo a necessidade de uso de portas dinâmicas tipicamente encontradas em aplicações distribuídas.

Além da natural distribuição dos agentes em relação à central de processamento, foram desenvolvidas duas interfaces de operação do sistema. Por utilizarem o banco de dados como meio de acesso, essas duas interfaces podem ser instaladas em servidores diferentes. A interface via *Web* permite a visualização dos elementos do inventário de rede e da manutenção das vulnerabilidades associadas a elementos ativos da rede. O console de operação desenvolvido em Java oferece acesso completo ao sistema, permitindo, entre outras operações, o disparo e agendamento de coletores,sensores e análises além de configurações do sistema como cadastro de agentes e visualização de tarefas ativas.

5.2 Agregação de Diferentes Ferramentas

A filosofia adotada na concepção e desenvolvimento foi a de aproveitar ao máximo as funcionalidades oferecidas por ferramentas consagradas na área de segurança e administração de sistemas. A utilização do conceito de *plugins* tanto do lado dos agentes quanto do lado da central de processamentos, facilitou bastante a integração de ferramentas tão distintas entre si como o NMAP, o Nessus, o Snort e o HFNet, entre outras (vide Tabela 2 - Sensores e Coletores já desenvolvidos). Em alguns casos, as ferramentas foram estendidas adicionando à elas uma pequena interface de adaptação para comunicação com o Agente. Na maioria das situações as ferramentas são executadas através de chamadas de sistema e seus dados de saída adaptados e, quando necessários, convertidos para formato adequado ao envio para a central.

A abstração dos mecanismos de comunicação e a conversão automática bidirecional entre mensagens XML e objetos *Java* ou estruturas de dados *Perl* representam facilidades adicionais no desenvolvimento de novos coletores e sensores. Em alguns casos a incorporação de novas ferramentas ao sistema não envolveu mais que uma semana de programação de um desenvolvedor experiente.

5.3 Criação do Inventário de Rede

Além de ter um papel vital na correlação e classificação de eventos gerados pelos sensores, o inventário de rede representa, por si só, um sub-produto do sistema. Através do inventário é possível se ter uma visão geral do nível de segurança da rede monitorada, além de possibilitar a geração de análises históricas e detectar tendências na evolução da segurança no ambiente. A partir dessas informações, é possível direcionar atitudes corretivas como atualização de serviços e investimento em treinamento do pessoal técnico. A Figura 16 mostra duas agregações realizadas sobre dados colhidos pelo *NessusInventory* e consolidados no inventário de rede. O gráfico mais a esquerda mostra as máquinas mais vulneráveis categorizando as vulnerabilidades de cada uma por perigo associado. O gráfico mais a direita sumariza, para toda a rede monitorada, os percentuais dos tipos de vulnerabilidade.

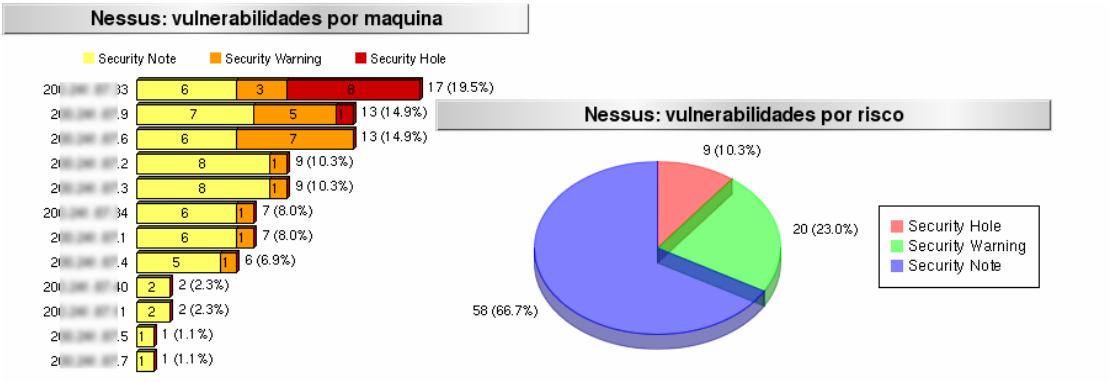


Figura 16 Gráficos gerados sobre informações presentes no inventário de rede

5.4 Análises Utilizando Sistemas de Produção

Dentre os paradigmas de inteligência artificial estudados durante a fase de levantamento do estado da arte (vide seção 2.3) o que melhor satisfez as necessidades do sistema e que apresentou maior viabilidade de implementação foi o de sistemas especialistas formados por sistemas de produção com suporte a encadeamento progressivo. A razão mais forte para escolha dessa abordagem é a possibilidade de expressar conhecimento humano em diversas regras simples de produção, que, quando utilizadas em conjunto, conseguem representar tomadas de decisão relativamente complexas. O JEOPS [35] se encaixou bem nesse contexto, possibilitando a utilização da linguagem Java, resultando em um sistema com a qualidade final de apresentação, robustez e flexibilidade desejados pelos autores, ao mesmo tempo em que permite a utilização de conceitos pouco implementados na área detecção de intrusão, como sistemas de produção e correlações com inventário de rede.

5.5 Correlação de Eventos de IDS

A utilização de regras de produção para análise e classificação de eventos do snort, permitiu um melhoramento da apresentação desses eventos para o administrador,

reduzindo significativamente o esforço necessário para implantar e utilizar no dia-dia um sistema de detecção de intrusão de redes.
A classificação, agrupamento e ordenação dos eventos permitem ao gerente ter uma visão em tempo real dos ataques a sua rede, enfatizando eventos considerados mais perigosos e minimizando eventos pouco perigosos, que, caso contrário, apenas serviriam para dificultar a identificação de potenciais problemas.
A Figura 17 mostra a interface de visualização das análises e eventos. As colunas em destaque indicam a severidade (impacto) dos eventos e das análises (grupos de eventos). A cor e o impacto da primeira análise indicam que um evento potencialmente perigoso foi detectado.

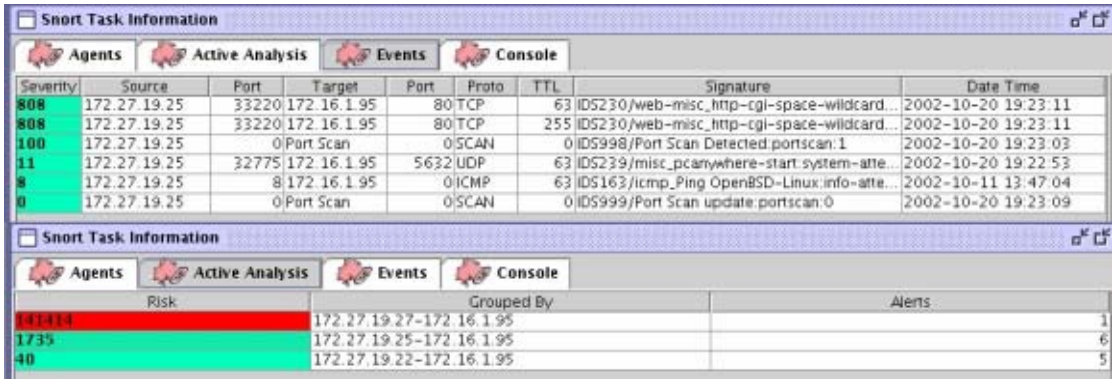
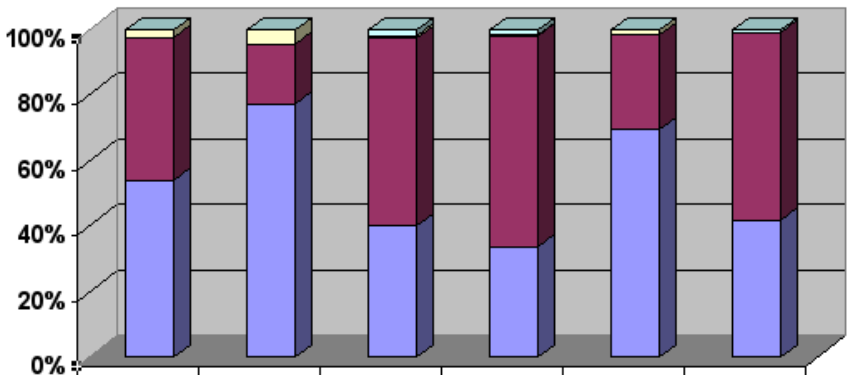


Figura 17 Tela de acompanhamento de análises ativas de eventos do Snort

5.6 Diminuição de Falsos Positivos

Quando comparado com o uso tradicional de sistemas de visualização de eventos de IDS o PIPS apresenta duas grandes vantagens: a diminuição do número de falsos positivos e o destaque para eventos mais relevantes. A Figura 18 mostra o acompanhamento do impacto dos eventos de IDS ao longo de 6 meses em uma rede real. Observando a figura é possível notar que a maioria absoluta dos eventos observados nessa rede, ou são imediatamente descartados como falsos positivos e filtrados, ou são classificados como de baixo impacto.
Entre 2% e 4% dos eventos são classificados como de médio ou alto impacto. A concentração dos esforços de análise manual nesse número reduzido de eventos gera uma otimização do tempo dos administradores dos sistemas, permitindo que o foco da administração seja voltado para a erradicação das vulnerabilidades, que, a médio prazo tende a diminuir ainda mais o impacto dos eventos detectados.



	fevereiro	março	abril	maio	junho	julho
alto impacto	0.00%	0.00%	1.91%	1.63%	0.00%	0.75%
médio impacto	2.17%	4.25%	0.68%	0.13%	1.60%	0.11%
baixo impacto	43.65%	18.36%	57.36%	64.50%	28.94%	57.13%
filtrados	54.17%	77.39%	40.06%	33.75%	69.46%	42.01%

Figura 18 - Impacto de Eventos de IDS ao longo de 6 meses

5.7 Interfaces Amigáveis

Por ter sido concebido para ser usado em ambientes de produção com objetivo de maximizar a produtividade de equipes de co-administração de segurança, as interfaces gráficas do sistema foram projetadas para serem ao mesmo tempo simples de operar e suficientemente ricas para satisfazer necessidades mais comuns das tarefas dos operadores. Com esse objetivo foram implementadas duas interfaces com visões diferentes do sistema:

A interface *Web* permite a navegação pelo inventário de rede, permitindo às equipes de operação a rápida visualização dos dados coletados pelos agentes e normalizados no banco de dados. Essa interface também permite o enriquecimento desses dados através da possibilidade de adição de meta-informações (como funções de servidores e responsáveis por sistemas) que não podem ser coletadas pelas ferramentas. Também é possível nessa interface gerar gráficos contendo agregações sobre as informações do inventário como os mostrados na Figura 16.

O console da central de processamentos foi desenvolvido utilizando a tecnologia *Java Swing*, que além de apresentar uma aparência final de qualidade ao produto, oferece a flexibilidade necessária para os operadores do sistema. Essa interface oferece um controle maior sobre o sistema, permitindo: o disparo de coletores e sensores nos diversos agentes da rede, configuração de parâmetros de execução, cadastro de agentes, visualização de tarefas em andamento (análises e *handlers* ativos) e agendamento de tarefas. A Figura 11 e a Figura 17 são algumas das visões disponíveis nessa interface.

5.8 Validação em Redes Reais

Sem sombra de dúvidas o melhor resultado obtido no desenvolvimento do PIPS foi a sua utilização em ambiente de produção monitorando redes reais. Várias funcionalidades adicionadas ao sistema vieram de necessidades levantadas pelas equipes de operação. As regras de produção, por exemplo, estão em constante

evolução. Cada rede apresenta uma particularidade que faz com que novas regras sejam necessárias, e que regras aplicadas em outras redes tornem-se inadequadas.
Na data da conclusão da escrita deste documento o sistema estava sendo utilizado em cinco redes diferentes. Uma dessas redes possui mais de 100 servidores e duas delas mais de 500 estações. Os coletores mais extensivamente testados são os que podem ser executados remotamente, não necessitando instalação de agentes diretamente nos servidores monitorados. Os menos utilizados são os que fazem monitoração de sistemas Windows (FileWatcher,RegWatcher, HFNet e EnumNBT).
Ao todo, estão instalados e funcionando 14 agentes e 5 instâncias da central de processamento (uma para cada rede monitorada). Os dados são armazenados em 5 bases de dados no mesmo servidor MySQL. A infra-estrutura de rede envolve 3 servidores de produção (central de operação, banco de dados e Servidor *Web*) e 7 postos avançados (máquinas instaladas na rede do cliente para monitoração remota da rede). Cada cliente possui uma VPN por onde trafegam de forma segura os dados trocados com a central.

5.9 Síntese

Utilizado em redes reais de médio e grande porte, o sistema atingiu os resultados esperados com a utilização de um *framework* distribuído com arquitetura flexível, possibilitando a utilização, agregando e correlacionando dados de diferentes ferramentas.
As análises e correlações de eventos de IDS têm permitido uma redução significativa no volume de falsos positivos, produzindo resultados satisfatórios no dia-a-dia da operação das ferramentas. Os dados relacionados a vulnerabilidades no ambiente monitorado e a ameaças a esse ambiente vêm possibilitando a realização de análises de risco fundamentadas em métricas reais.

6 Considerações Finais

Nos últimos dois capítulos foram mostrados os detalhes de implementação e os resultados obtidos na utilização do sistema em redes reais. Alguns desses resultados constituem contribuições relevantes para a área de segurança de sistemas e em particular para a detecção e prevenção de intrusões. Apesar de já ter rendido bons frutos no uso rotineiro o sistema está em constante evolução e no momento da escrita deste documento já existem novas ferramentas sendo incorporadas e novas análises em desenvolvimento. Nesse capítulo serão destacadas as principais contribuições do trabalho, os próximos passos de implementação e tecidas as conclusões finais.

6.1 Resumo das Contribuições

6.1.1 Foco em Proatividade e Proteção

A área de detecção de intrusão obteve grandes avanços desde sua concepção na década de 70 até sua popularização nos anos 90. O aumento da integração entre as redes coorporativas e a Internet e da importância das redes para o bom funcionamento dos negócios fez com que a característica passiva dos sistemas tradicionais fosse questionada. Mais recentemente, com o surgimento dos sistemas de proteção contra intrusões (IPS), a postura deixou de ser passiva e passou a ser reativa e, em alguns casos, preventiva já que, quando bem utilizados, esses novos sistemas podem evitar ataques, reagindo em tempo real e fechando conexões consideradas perigosas. O sistema ora desenvolvido representa um passo adiante nessa evolução, permitindo a integração de várias ferramentas consagradas e focando na prevenção de incidentes através da constante monitoração das redes e dos servidores, antecipando possíveis ataques e diminuindo a janela de vulnerabilidade dos sistemas protegidos. Essa postura proativa tem se mostrado eficaz principalmente quando associada a procedimentos corretivos e a treinamento e capacitação das equipes de administração de sistemas. Um exemplo concreto dos resultados dessa nova postura foi a criação, em um dos clientes, de uma rede de homologação. Nessa rede os novos servidores e serviços são inicialmente submetidos às varreduras e testes do sistema. Eventuais problemas detectados são sanados e só então os sistemas são expostos para a Internet.

6.1.2 Métricas Reais para Análise de Risco

As técnicas de análise de risco mais utilizadas são baseadas em dados abstratos, tais como: respostas a questionários e estatísticas sobre intrusões em redes semelhantes. Tipicamente risco de um incidente é calculado em função da probabilidade de ocorrência e do efeito desse (ou perda acarretada pelo) evento [47]:

$$\textbf{\textit{Risco = f(probabilidade,efeito)}}$$

Em Segurança da Informação, a probabilidade pode ser estimada em função da vulnerabilidade e das ameaças, enquanto o efeito está diretamente ligado ao valor do recurso para a organização. Esse valor muitas vezes é associado ao prejuízo potencial

do comprometimento daquele ativo. O risco, então, pode ser calculado em função desses três fatores [49]:

Risco = f(Vulnerabilidade, Ameaça,Valor)

O maior problema com a abordagem tradicional é que todos os valores para as variáveis da equação acima não passam de estimativas obtidas por processos pouco científicos. A integração de dados colhidos por ferramentas de varreduras e sistemas de detecção de intrusão constitui uma fonte muito mais confiável para o levantamento do grau de vulnerabilidade e de ameaça aos sistemas monitorados. Como resultado das varreduras por falhas conhecidas o fator vulnerabilidade pode ser calculado em termos numéricos para cada um dos servidores envolvidos. Os eventos detectados nos IDS indicam a intensidade e freqüência com que os sistemas são alvos de ataques externos, permitindo uma estimativa mais científica para o fator Ameaça. O valor dos ativos ainda precisa ser levantado através de questionários e entrevistas, já que não existem formas automatizáveis de estimá-los. Porém, das três variáveis da equação de risco, o valor dos ativos é exatamente a que melhor pode ser estimada através de entrevistas pois esse valor depende da importância daquele ativo para o negócio da empresa.

6.1.3 Agentes Distribuídos e Análises Inteligentes

Durante a etapa de levantamento sobre o estado da arte em detecção de intrusão, descrita no Capítulo 2, foram estudados vários sistemas e propostas de sistemas que empregavam os conceitos de agentes distribuídos e inteligentes. De todos os sistemas estudados o AAFID e o EMERALD foram os que chegaram a ter implementações de fato. A principal contribuição desses sistemas foi a utilização de agentes distribuídos para detecção de intrusão, no entanto, nenhum dos dois obteve avanços significativos na inteligência das análises efetuadas. No PIPS, os conceitos de agentes distribuídos e análises inteligentes através de sistemas especialistas baseados em regras de produção foram utilizados em conjunto e bons resultados foram obtidos na prática.

O sistema desenvolvido permite a incorporação de forma extensível de novos agentes e novas análises, possibilitando correlação entre dados colhidos por diversas ferramentas. A utilização de regras de produção com encadeamento progressivo nas análises permite concatenar as ações de regras pequenas, produzindo um efeito global bastante expressivo.

6.2 Trabalhos Futuros

A característica extensível do sistema permite que ele esteja num estado de constante evolução. Os próximos passos no futuro próximo do desenvolvimento visam a incorporação de novas ferramentas e o desenvolvimento de novas análises e correlações associadas a essas ferramentas. Para médio prazo estão previstas adequações a padrões internacionais de notificação de incidentes (IDMEF) e incorporação de bases de vulnerabilidades gratuitas (OSVDB) para melhorar a classificação de vulnerabilidades e ameaças descobertas. Ainda para médio prazo estão previstos ajustes nas interfaces gráficas, visando o aumento da usabilidade e a diminuição do tempo de investigação de incidentes. Em longo prazo a meta é focar na

qualidade das análises automatizadas e no tempo de resposta do sistema para permitir a tomada de decisões relacionas a contra medidas em tempo real.

6.2.1 Incorporação de Novas Ferramentas

Várias ferramentas estão sendo estudadas para serem incorporadas ao sistema. Em um primeiro momento serão priorizadas as ferramentas que adicionem funcionalidades ainda não presentes no sistema. Em seguida serão consideradas ferramentas que melhorem a qualidade dos dados coletados e conseqüentemente das análises realizadas. Por ordem de prioridade pode-se destacar as seguintes ferramentas:

- **AIDE** – descrito na seção 2.1.2 o AIDE (*Advanced Intrusion Detection Environment*) é um verificador de integridade de arquivos que permite detectar indícios de intrusões através da percepção de alterações em arquivos e diretórios do sistema. A estratégia para incorporação do AIDE é desenvolver um coletor que, executado periodicamente no sistema de agendamento de tarefas do agente execute o AIDE e mande os resultados para a central. Na central após o processamento desses resultados uma análise deverá ser executada para priorizar os eventos gerados e correlacioná-los com outros eventos detectados em outras partes da rede.
- **Frameworks para Testes de Penetração –** um dos principais problemas com os atuais sistemas para varreduras de vulnerabilidades é a ausência de certeza quanto a real vulnerabilidade de um sistema a um ataque conhecido. Uma possibilidade em estudo é a da utilização de um *Framework* para Testes de Penetração, com o objetivo de melhorar o nível de certeza da exposição de um sistema ou serviço à uma vulnerabilidade. Dois sistemas estão sendo considerados para essa finalidade: o ***Canvas*** desenvolvido pela Immunity Inc. (www.immunitysec.com) é um produto comercial com boa taxa de atualização e completamente desenvolvido na linguagem Python. A principal vantagem para sua adoção é a maturidade do código e o bom número de classes de testes de penetração disponibilizados pelo sistema. O ***Metasploit*** é um produto de código aberto desenvolvido na linguagem *Perl*. Embora bem mais recente que o *Canvas* não existem restrições quanto ao uso do código fonte e o fato de ser desenvolvido em *Perl* pode facilitar sua incorporação nos agentes. Em ambos os casos o principal cuidado a ser tomado é em relação aos possíveis efeitos colaterais resultantes da tentativa de exploração de uma vulnerabilidade. Alguns dos testes oferecidos pelos *frameworks* podem causar indisponibilidade dos serviços o que em algumas redes e sistemas pode ser inaceitável, mesmo que indiquem uma falha grave. Em um primeiro momento os sistemas serão exaustivamente testados manualmente e só serão incorporados ao PIPS quando se tiver boa certeza da possibilidade de realização dos testes sem efeitos maléficos.
- **ISS Proventia** – uma área ainda pouco explorada em relação a extensibilidade do sistema é a possibilidade de integração com ferramentas comerciais e de código fechado. Embora não exista atualmente nenhuma necessidade de incorporação de ferramentas desse tipo é interessante testar a possibilidade já que em algum momento essa necessidade possa vir a existir. Um bom caso de testes seriam os sistemas de detecção de intrusão da ISS (www.iss.net), anteriormente conhecidos como *Real Secure* e recentemente renomeados para *Proventia Intrusion Detection* e *Proventia Intrusion Protection*. As razões para sua escolha incluem a grande base de usuários e o reconhecimento na comunidade de segurança da qualidade desses sistemas. O principal desafio na agregação desses produtos ao PIPS é a ausência de documentação de seu funcionamento interno e a falta de acesso ao código fonte.

6.2.2 Análises de Registros de Auditoria

Uma fonte importante de indícios de tentativas de invasão a sistemas computacionais é a análise de registros de auditoria (*logs*) gerados pelas aplicações. A tendência cada vez mais crescente de concentração de ataques nas camadas mais altas da pilha TCP/IP aumenta a necessidade de agregação de informações colhidas desses registros. Um problema ainda não resolvido em relação a essas análises é sua localização. Embora a tendência no PIPS seja a concentração das análises do lado da central de processamentos o volume de dados contidos nos registros de auditoria pode representar um empecilho à essa arquitetura. Uma alternativa possivelmente viável é realizar análises distribuídas: os coletores ficam responsáveis por identificar nos arquivos de *log* padrões de ataques conhecidos. Os padrões detectados viram então eventos que são enviados para a central de processamentos, onde ocorre um refinamento dos dados através de correlações com o inventário de rede e outros eventos detectados por outros sensores e coletores.

6.2.3 Notificação Padronizada de Incidentes

Atualmente os resultados das análises dos eventos de segurança detectados pelo sistema têm dois objetivos principais: realimentar o sistema com informações de ameaças, ajudando a categorizar o risco com dados reais; e prover às equipes de administração de sistemas informações necessárias para identificar incidentes de segurança e tomar as medidas cabíveis para minimizar o efeito dos ataques e notificar administradores de outras redes utilizadas como ponto de origem das tentativas de invasão. Embora o conteúdo já seja gerado automaticamente pelo sistema, a forma da notificação dos incidentes ainda está a cargo dos operadores do sistema. À medida que o processo de notificação for sendo mais automatizado, é desejável que a forma da notificação seja estruturada segundo padrões internacionais. Estão em estudo os padrões IDMEF [44] e IODEF [45]. O IDMEF (*Intrusion Detection Message Exchange Format*) é um padrão para formatação de dados produzidos por detectores de intrusão que tem como principal objetivo a normalização dos dados produzidos por diferentes ferramentas, facilitando a comunicação entre sensores e consoles de fabricantes distintos. Atualmente o PIPS só tem suporte a eventos gerados pelo IDS Snort. Tão logo novos IDS sejam suportados pelo sistema é desejável que os eventos sejam gerados em um formato padrão facilitando o seu processamento na central. O IODEF (*Incident Object Description and Exchange Format*) é uma proposta de padrão para notificação de incidentes de segurança. Seu principal objetivo é permitir a troca de informações operacionais e estatísticas entre equipes de resposta a incidentes. Ambos os padrões são aplicações da linguagem XML e possuem suas especificações e esquemas publicados e atualizados periodicamente pela IETF.

6.2.4 Correlações com a OSVDB

Ativo desde o final de 2003 a OSVDB (Open Source Vulnerability Database) [46] é atualmente a mais completa base de dados de vulnerabilidades disponível abertamente para uso em sistemas comerciais e não comerciais. Disponibilizada em formato XML e contendo amplo suporte da comunidade de segurança a base de dados pode rapidamente ser integrada ao PIPS. O principal benefício da sua adoção é a

possibilidade de novas correlações entre eventos detectados nos sensores, versões de software identificadas pelos coletores e dados disponíveis na base de conhecimentos. O método de integração pode seguir duas abordagens: *off-line*, através da importação e constante atualização da base disponibilizada em XML para o banco de dados do PIPS; ou *on-line*, através da comunicação direta com os servidores da OSVDB utilizando o padrão XML-RPC.

6.3 Conclusões

O aumento do valor associado aos dados armazenados e servidos pelos sistemas computacionais sugere a necessidade de proteção dessas informações contra tentativas de ataques e invasões. O processo de tornar um sistema em rede mais seguro envolve um bom planejamento, utilização de técnicas e ferramentas de proteção, configuração segura dos sistemas operacionais e aplicações, elaboração e implantação de procedimentos de boas práticas de segurança. Uma vez alcançado um nível aceitável de proteção é necessário mantê-lo. Para isso são utilizadas ferramentas e técnicas de monitoração contínua que permitem acompanhar o dia-a-dia da segurança dos sistemas.

Os sistemas de detecção de intrusão foram tradicionalmente utilizados como ferramentas de monitoração contra ameaças à segurança dos ambientes. O grande volume de falsos positivos e a comum indisponibilidade de tempo das equipes de administração de tratar todos esses eventos tornaram o uso dos IDS inviável para a maioria das corporações. Outra abordagem para monitoração é a de realizar varreduras periódicas de portas e levantamento de vulnerabilidades procurando sanar de maneira pontual os problemas encontrados.

Nesse trabalho apresentamos uma nova abordagem para gerenciamento de segurança e prevenção de intrusões, denominado PIPS (Proactive Intrusion Prevention Systems). Esse sistema atua de maneira proativa monitorando constantemente a rede através de varreduras periódicas que montam um perfil ativo da rede. Além de fornecer uma visão detalhada do estado da rede, o perfil é utilizado para correlações com eventos de IDS produzindo análises mais refinadas. A abordagem consiste na utilização de agentes, que realizam coleta dos dados de forma distribuída e os disponibiliza para serem processados por um analisador central que possui uma visão global da rede. O uso de sistemas especialistas baseados em regras de produção permite a correlação entre eventos gerados nos sensores e o estado ativo da rede. A arquitetura flexível possibilita a utilização de ferramentas já consagradas, aproveitando sua maturidade e agregando valor ao sistema.

Os objetivos originais do trabalho foram alcançados com sucesso. O desenvolvimento do sistema tornou possível:

- estudar as possibilidades de utilização de sistemas distribuídos e análises inteligentes na detecção de intrusão;
- suprir as deficiências dos sistemas de detecção de intrusão tradicionais, integrando dados colhidos utilizando diversas fontes e aumentando a qualidade das análises sobre tentativas de intrusão.

Testado em redes de grande porte o sistema mostrou-se robusto, produzindo resultados satisfatórios tanto na redução do número de falsos positivos quanto no fornecimento de métricas relacionadas a ameaças e vulnerabilidades. Essas métricas podem, posteriormente, ser utilizadas como fontes para análise de risco.

Bibliografia

[1] MORRIS, R.; THOMPSON, K. Password Security: A Case History. *Communications of the ACM*, v. 22, n. 11, p. 594-597, 1979

[2] SPAFFORD, E. H. An analysis of the internet worm. In: *Proceedings of the 2nd European Software Engineering Conference*, 1989, Warwick, Coventry, UK, Lecture Notes in Computer Science, Ed. Springer, Vol. 387, 1989, p. 446-468.

[3] MOORE, D. et al. The Spread of the Sapphire/Slammer Worm. Disponível em: http://www.cs.berkeley.edu/~nweaver/sapphire/ Acesso em: 24 de jul. 2005.

[4] TITTEL, E.; CHAPPLE, M.; STEWART, J.M. Certified Information Systems Security Professional Study Guide. Ed. Sybex, 2003. 944p.

[5] HORA, H.C. Sobre a percepção remota de s*niffers* para detectores de intrusão em redes TCP/IP. Mar 2001. 101 f. Dissertação (Mestrado) – Centro de Informática, Universidade Federal de Pernambuco. Recife. 2001.

[6] OPEN WEB APPLICATION SECUIRITY PROJECT. Guide To Building Secure Web Applications.
Disponível em http://prdownloads.sourceforge.net/owasp/OWASPGuideV1.1.1.pdf Acesso em 24. de jul. 2005.

[7] KURZBAN, S. Implementation Access Controls. In: TIPTON, H.F.; KRAUSE, M. *Information Security Management Handbook*. Boca Raton: CRC Press LLC, 2002. Paginação Irregular

[8] DEPARTMENT OF DEFENSE. Trusted Computer System Evaluation Criteria. Disponível em: http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.pdf Acesso em 24. de jul. 2005

[9] ICSA INTRUSION DETECTION SYSTEMS CONSORTIUM. An Introduction to Intrusion Detection & Assessment.
Disponível em http://www.icsa.net/html/communities/ids/whitepaper/Intrusion1.pdf Acesso em 24. de jul. 2005

[10] PORRAS P. et al. The Common Intrusion Detection Framework Architecture Disponível em http://www.isi.edu/gost/cidf/ Acesso em 24. de jul. 2005

[11] PTACEK, T.H.; NEWSHAM, T.N. Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection. Disponível em http://downloads.securityfocus.com/library/ids.ps Acesso em 24. de jul. 2005

[12] ROESCH, M. Snort: Lightweight Intrusion Detection for Networks. In: *Proceedings of the 13th USENIX Conference on System Administration*, 1999, Seattle, Washington, USENIX Association, Berkeley CA, 1999, p. 229-238

[13] LEHTI, R. The AIDE Manual. Disponível em http://www.cs.tut.fi/~rammer/aide/manual.html Acesso em 24. de jul. 2005

[14] SNAPP, S.R. et al. DIDS (distributed intrusion detection system): motivation, architecture, and an early prototype. In: *Internet Besieged: Countering Cyberspace Scofflaws*. New York: ACM Press, 1998. p. 211-227

[15] STANIFORD-CHEN, S et al. GrIDS: A Graph-Based Intrusion Detection System for Large Networks. In: *Proceedings of the 19th National Information Systems Security Conference*, Baltimore, 1996.

[16] NEUMANN, P.G.; PORRAS, P.A. Experience with EMERALD to Date. In: *Proceedings of the 1st Workshop on Intrusion Detection and Network Monitoring*. Berkeley, CA: USENIX Association, 1999. p. 73-80

[17] ZAMBONI, D. et al. An Architecture for Intrusion Detection using Autonomous Agents. In: *Proceedings of the 14th Annual Computer Security Applications Conference*. Washington, DC: IEEE Computer Society,1998 p.13

[18] HELMER, G. et al. Inteligent Agents for Intrusion Detection. In: *Proceedings of the IEEE Information Technology Conference,* 1998, Syracuse, NY. IEEE Computer Society, 1998. p121-124

[19] GOPALAKRISHNA, R.; SPAFFORD, E. H. A framework for distributed intrusion detection using interest driven cooperating agents *Paper for Qualifier II examination*, Department of Computer Sciences, Purdue University, May 2001.

[20] HASSAS S.; FENET S. A distributed Intrusion Detection and Response System based on mobile autonomous agents using social insects communication paradigm In: *Electronic Notes in Theoretical Computer Science, Vol. 63*. Elsevier Science 2002. p1-18

[21] CARVER, C. et al. A Methodology for Using Intelligent Agents to provide Automated Intrusion Response. In: *Proceedings of the 2000 IEEE Workshop on Information Assurance and Security*. West Point NY Jun 2000

[22] SECURE COMPUTING. IPS: Deciphering the inline Intrusion Prevention hype and working toward a real world, proactive security solution. Disponível em: http://www.securecomputing.com/pdf/Intru-Preven-WP1-Aug03-vF.pdf Acesso em 24. de jul. 2005

[23] LAWRENCE BERKELEY NATIONAL LABORATORY. Arpwatch Disponível em http://www-nrg.ee.lbl.gov/ Acesso em 24. de jul. 2005

[24] BAUER K. System Monitoring In: *Automating UNIX and Linux Administration* Apress, Set 2003. p. 337-344

[25] FYODOR The Art of Port Scanning In: *Phrack Magazine Issue 51*. Disponível em http://www.phrack.org/show.php?p=51&a=11 Acesso em 24. de jul. 2005

[26] TEENABLE NETWORK SECURITY. Nessus Open Source Vulnerability Scanner Disponível em http://www.nessus.org Acesso em 24. de jul. 2005

[27] SHAVLIK & MICROSOFT. HFNetChk Disponível em http://hfnetchk.shavlik.com/ Acesso em 24. de jul. 2005

[28] CARNUT, M.A. et al. FreeICP.ORG: Free Trusted Certificates by Combining the X.509 and PGP Hierarchy Through a Collaborative Trust Scoring System In: *2nd Anual PKI Research Workshop Proceedings*, NIST Gaithersburg MD, Set 2003 Disponível em http://middleware.internet2.edu/pki03/PKI03-proceedings.html Acesso em 24. de jul. 2005

[29] MATTOS, C.L. et al. Matrix: Gerenciamento Integrado da Segurança de Ambientes Heterogêneos. In: *6º Simpósio em Segurança da Informação*, 2004, São José dos Campos, SP. Anais. Disponível em http://www.ssi.org.br/anais/2004/at_2004.zip Acesso em 24. de jul. 2005

[30] WALL, Larry Perl, the first postmodern computer language Disponível em http://www.perl.com/pub/a/1999/03/pm.html Acesso em 24. de jul. 2005

[31] CARNUT, M.A.; GONDIM, J. ARP Spoofing Detection on Switched Networks: A Viability Study. In: 5º *Simpósio em Segurança da Informação*, 2003, São José dos Campos, SP. Anais. Disponível em http://www.ssi.org.br/anais/2003/at_2003.zip Acesso em 24. de jul. 2005

[32] GAMMA E. et al. Design Patterns: Elements of Reusable Object-Oriented Software. 1st ed. Massachusetts: Addison-Wesley, 1994. 416p.

[33] Ferraiolo, Cugini e Kuhn
Role-Based Access Control (RBAC): Features and Motivations.
In Proceedings of the Annual Computer Security Applications Conference, 1995, New Orleans, Louisianna. IEEE Computer Society Press.1995.

[34] THE APACHE SOFTWARE FOUNDATION. Apache Torque. Disponível em http://db.apache.org/torque Acesso em 24. de jul. 2005

[35] FIGUEIRA C.; RAMALHO, G. JEOPS - The Java Embedded Object Production System. In: Advances in Artificial Intelligence. Lecture Notes on Artificial Intelligence Series, vol. 1952 p. 52-61, 2000

[36] RITTER, J. Enum, A tool to enumerate, using null and user sessions, Win32 (NT) info. Disponível em http://www.bindview.com/Services/RAZOR/Utilities/Windows/enum_readme.cfm Acesso em 24. de jul. 2005

[37] CROSBIE, M. SPAFFORD, E. Defending a Computer System Using Autonomous Agents. In: *8th National Information Systems Security Conference*, Baltimore 1996

[38] HEBERLEIN, L. et al. A Network Security Monitor. In *Proceedings of the IEEE Computer Society Symposium, Research in Security and Privacy*, May 1990, p. 296-303

[39] KUMAR, S. Classification and Detection of Computer Intrusions. Aug 2001. Tese de Doutorado – Purdue University. West Lafayette, Indiana. 2001

[40] SEBRING, M.M. et al. Expert Systems in Intrusion Detection: A Case Study In: *Proceedings of the 11th National Computer Security Conference*, Baltimore Oct. 1988

[41] HOCHBERG, J. NADIR: An automated system for detecting network intrusion and misuse. In: *Computers and Security*, v. 12, n. 31. Elsevier Publications 1993 p.235-248

[42] CARNUT, M.A. et al Improving Stateful Inspection Log Analysis In: 3º *Simpósio em Segurança da Informação*, 2001, São José dos Campos, SP. Anais. Disponível em http://www.ssi.org.br/anais/2001/anais_2001.zip Acesso em 24. de jul. 2005

[43] FORGY C. L. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. In: *Artificial Intelligence*, v. 19, n. 1 1982 p. 17-37.

[44] DEBAR, H.; CURRY, D.; FEINSTEIN, B. IDMEF – The Intrusion Detection Message Exchange Format. Internet Draft. Disponível em http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-14.txt Acesso em 24. de jul. 200

[45] DANYLIW, R. The Incident Object Description Exchange Format (IODEF) Implementation Guide. Internet Draft. Disponível em http://tools.ietf.org/wg/inch/draft-ietf-inch-implement/draft-ietf-inch-implement-01.txtAcesso em 24. de jul. 2005

[46] OSVDB – The Open Source Vulnerability Database. Disponível em http://www.osvdb.org Acesso em 24. de jul. 2005

[47] KERZNER, Harold. Project Management: A Systems Approach to Planning, Scheduling, and Controlling. Eight Edition. New Jersey: John Wiley and Sons, January 31, 2003. p 653

[48] TUDOR, J. K. Information Security Architecture: An Integrated Approach to Security in the Organization. Washington: Auerbach. 2001. p 47

[49] MCBRIDE, P. et al. Secure Internet Practices: Best Practices for Securing Systems in the Internet and E-Business Age. Boca Raton: Auerbach. 2002. p 46

SERVIÇO PÚBLICO FEDERAL
UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Ata de Defesa de Dissertação de Mestrado d
Centro de Informática da Universidade Fed
ral de Pernambuco, 19 de agosto de 2005.

Ao décimo nono dia do mês de agosto d
ano dois mil e cinco, às dez horas, no Centro de Informática da Universidade Feder
de Pernambuco, teve início a dissertação de Mestrado em Ciência da Computaçã
intitulada **"PIPS – Sistema de Prevenção de Intrusões Pro-Ativo"**, do candida
Cleiton Soares Martins, o qual já havia preenchido anteriormente as dema
condições exigidas para a obtenção do grau de mestre. A Banca Examinador
composta pelos professores Fabio Queda Bueno da Silva, pertencente ao Centro d
Informática desta Universidade, Ricardo Massa Ferreira Lima, pertencente à Esco
Politécnica da Universidade de Pernambuco e André Luis de Medeiros Santo
pertencente ao Centro de Informática desta Universidade, sendo o primei
presidente da Banca Examinadora e o último orientador do trabalho de dissertaçã
resolveu: **Aprovar por unanimidade e dar o prazo de trinta dias para a entreg
da versão final do trabalho**. E para constar lavrei a presente ata que vai por mi
assinada e pela Banca Examinadora. Recife, 19 de agosto de 2005.

Maria Lília Pinheiro de Freitas
(secretária)

Prof. Fabio Queda Bueno da Silva
(primeiro examinador)

Prof. Ricardo Massa Ferreira Lima
(segundo examinador)

Prof. André Luis de Medeiros Santos
(terceiro examinador)