Universidade de São Paulo

Instituto de Ciências Matemáticas e de Computação

Departamento de Ciências da Computação e Estatística

1º Relatório de Iniciação Científica

"Especificação e Implementação de Agentes Móveis em um Sistema de Detecção de Intrusão"

Processo 99/12226-4

Francisco Gomes Milagres
(francisco@milagres.com)

Orientador:

Prof. Dr. Edson dos Santos Moreira
(edson@icmc.sc.usp.br)

USP - São Carlos
Janeiro de 2001

Índice

1. Resumo

O projeto *"Especificação e Implementação de Agentes Móveis em um Sistema de Detecção de Intrusão*" apresenta a modelagem, implementação e validação de um dos cenários de execução necessários para a constituição de um Sistema de Detecção de Intrusões (SDI) baseado em Agentes Móveis: **a Identificação de Usuários Anômalos**, conforme especificação no plano de pesquisa do mesmo.

Entre os objetivos deste projeto, encontram-se a redução dos custos apresentados por um SDI monolítico com a implementação de um cenário de execução para agentes móveis, a contribuição para avaliação e a validação da arquitetura proposta em [BERNARDES, 1999] através de sua implementação e posterior utilização no sistema acadêmico de nosso Instituto, proporcionando a extensão dos sistemas de detecção de intrusão existentes com a inserção da tecnologia de agentes móveis e também contribuindo para o avanço das tecnologias de segurança computacional.

Este relatório descreve a execução dos 1°, 2° e início do 3° segmentos do cronograma de trabalho deste projeto de Iniciação Científica, *"Especificação e Implementação de Agentes Móveis em um Sistema de Detecção de Intrusão*", como mostra o diagrama abaixo:

	Agosto/2000	**Setembro/2000**	**Outubro/2000**	**Novembro/2000**	**Dezembro/2000**	**Janeiro/2001**	**Fevereiro/2001**	**Março/2001**	**Abril/2001**	**Maio/2001**	**Junho/2001**
1. Análise de Requisitos	■	■									
2. Modelagem do Cenário Proposto			■	■							
3. Implementação do Cenário Proposto					■	■	■				
4. Avaliação do Cenário Implementado								■			
5. Validação do Cenário Implementado									■	■	
6. Relatório Final											■

O projeto em desenvolvimento segue a seqüência de eventos do cronograma inicial e seu estado atual é o terceiro item, que trata da implementação do cenário proposto, como se segue:

1. *Análise de Requisitos*: Entendimento e especificação formal das funções que o sistema deverá desempenhar;
2. *Modelagem do Cenário Proposto*: Representação lógica das funções a serem implementadas através de uma modelagem com DFD;
3. *Implementação do Cenário Proposto*: [início] Codificação do modelo conforme a modelagem e a proposta do projeto de iniciação científica.

As bibliotecas de suporte para desenvolvimento dos agentes - Aglets -, terão também um capítulo para explanação, bem como a codificação dos agentes que fazem parte do cenário proposto no projeto, no caso, para *identificação de usuários anômalos* no sistema de detecção de intrusões.

A seguir, será feita uma introdução demonstrando os motivadores deste trabalho e projetos que seguem a mesma linha de desenvolvimento deste, sendo destaque para os desenvolvidos no grupo de pesquisa do Intermídia e também trabalhos envolvendo atividades de segurança crítica e que utilizam agentes móveis.

2. Introdução

Um número crescente de atividades essenciais é realizado através das redes (principalmente através da Internet) e seu funcionamento correto e confiável torna-se de vital importância. Por outro lado, atos de pirataria, tentativas de ataque e invasões consumadas têm se tornado freqüentes [CSI, 2000; MÓDULO, 2000; SPAFFORD et. al., 1998] e envolvem um número crescente de computadores destinando-se geralmente ao roubo, destruição ou alteração das informações. Este cenário mostra a necessidade de técnicas especiais de segurança nos sistemas de computação modernos; técnicas que vão além da tradicional prática "*locking-up-the-doors*".

O desenvolvimento de um Sistema de Detecção de Intrusão baseado em Redes Neurais foi o tema de pesquisa do Grupo de Segurança da Informação / Intermídia / USP por quatro anos, gerando teses, dissertações e artigos [CANSIAN, 1997; CANSIAN et al., 1997a; CANSIAN et al., 1997b; CANSIAN et al., 1997c; CANSIAN, 1998; BONIFÁCIO Jr, 1998; BONIFÁCIO Jr. et al., 1998a; BONIFÁCIO Jr. et al., 1998a]. O sistema baseia-se na coleta de dados que fluem na rede, sua interpretação e submissão a uma rede neural treinada com padrões de intrusão conhecidos. Este sistema provou ser bastante eficiente para intrusões oriundas de máquinas externas.

Pesquisas recentes, porém, mostram que o número de atividades intrusivas originadas dentro das próprias organizações podem chegar a 70% do total [CSI, 2000; SECURENET, 2000; MÓDULO, 2000]. Este cenário inviabiliza a estratégia usada até então de fixar um "guarda" na entrada do site (normalmente o sistema era associado a *firewalls*).

A possibilidade do desmembramento dos módulos do sistema de detecção de intrusão e sua implementação sobre agentes [NWANA, 1996; SPAFFORD et al., 1998] que podem se mover pela rede foi uma solução avaliada pelo grupo [REAMI, 1998; BERNARDES, 1999; BERNARDES, 2000; BERNARDES 2000a] e é tema de projetos desenvolvidos por grupos de pesquisas e respostas a incidentes em segurança, como o CERIAS *(Center for Education and Research in Information Assurance and Security at Purdue University*), que possui uma implementação de um ambiente que segue as idéias do sistema proposto por [CROSBIE & SPAFFORD, 1995a, 1995b].

Este ambiente, denominado *Autonomous Agents for Intrusion Detection* (AAFID) foi implementado utilizando-se a linguagem de *script*s *Perl* e diversos recursos de administração de sistemas e segurança comuns em ambiente UNIX. O ambiente AAFID - que já está em sua segunda versão, o AAFID2 - possui duas entidades distintas que suportam a execução dos agentes do sistema: *Transceivers* e *Monitors*, sendo estes últimos entidades de mais alto nível, que podem detectar possíveis eventos intrusivos não notados por entidades mais simples como *Transceivers*. As informações sobre o sistema AAFID e AAFID2 podem ser encontradas em [ZAMBONI et al, 1998].

Outro trabalho na área de aplicação de agentes autônomos em sistemas de detecção de intrusão é apresentado por Barrus e Rowe [BARRUS & ROWE, 1998]. O projeto dos autores é utilizar agentes autônomos estáticos que se comunicam através de um sistema de mensagens de alerta através de uma arquitetura distribuída. Algumas abordagens interessantes sugeridas são: a utilização de agentes especializados na detecção baseada em anomalia, detecção baseada em uso indevido e a criação de objetos específicos para tratar os diversos tipos de ataque.

Os motivadores e destaques para este projeto se resumem nas novas tendências da segurança da informação, que definem conceitos como *mobilidade* e *política de segurança* como pontos principais para um eficiente Plano de Segurança, seja para um site, seja para uma grande corporação, fazendo o paradigma da segurança móvel, aliada à definição de normas para cada aplicação crítica ser tema de trabalhos e artigos publicados [KARJOTH, 1997; MILAGRES, 2000] para análise de segurança móvel em corporações no Brasil e exterior.

3. Análise de requisitos

A arquitetura para o Sistema de Detecção de Intrusões baseado em Agentes Móveis, original do trabalho de Mauro César Bernardes [BERNARDES, 1999] é o cerne do desenvolvimento deste trabalho. Partindo deste modelo principal, o desenvolvimento de um cenário para execução de agentes móveis com uma tarefa específica – Identificação de Usuários Anômalos - determina o objetivo deste trabalho.

Este capítulo demonstra a análise de requisitos proposta neste trabalho e em conjunto com o trabalho de Mauro César Bernardes [BERNARDES, 1999], que em sua dissertação de mestrado discute sobre as principais tecnologias de suporte ao ambiente proposto e faz a avaliação do uso dos agentes móveis para aplicações de segurança computacional, tendo sido este trabalho o motivador para a especificação e implementação deste projeto.

A análise de requisitos é uma tarefa da engenharia de software que efetua a ligação entre a alocação de software em nível de sistema e o projeto de software [PRESSMAN, 1995].

Uma compreensão completa dos requisitos de software é fundamental para seu desenvolvimento bem-sucedido. Não importa quão bem projetado ou codificado seja, um programa mal analisado e especificado decepcionará o usuário e trará aborrecimentos ao desenvolvedor.

A tarefa de análise de requisitos é um processo de descoberta, refinamento, modelagem e especificação. O passo inicial é identificar os casos de uso do ambiente de *identificação de usuários anômalos* do sistema de detecção de intrusões (SDI) em questão. A partir disto, é possível ter uma visão mais precisa das funcionalidades que o sistema deve realmente suportar, para que decisões mais explícitas possam ser tomadas durante as fases de projeto e implementação.

Em seu trabalho, Bernardes [BERNARDES, 1999] apresenta um estudo detalhado da análise de requisitos para os agentes que farão parte do sistema de detecção de intrusão. Nele, é possível identificar os seguintes casos de uso de alto nível do sistema: manter lista de equipamentos monitorados, monitorar equipamentos remotamente e monitorar agentes.

a. Manutenção da lista de equipamentos monitorados

- O sistema deve permitir a manutenção de uma lista de equipamentos monitorados, ou seja, deve permitir as tradicionais operações de inclusão, remoção, alteração e consulta;
- O sistema deve permitir a definição de certos atributos pertinentes aos equipamentos, mais especificamente, nome, endereço IP, arquitetura, localização física e outros atributos que o usuário possa desejar manter;
- O sistema deve permitir a criação livre de grupos de equipamentos a partir da lista de equipamentos monitorados pelo SDI;
- O sistema deve permitir que todas as operações realizadas sobre equipamentos possam também ser efetuadas sobre grupos de equipamentos;
- O sistema não deve permitir que usuários/agentes não autorizados tenham acesso à lista de equipamentos monitorados;

b. Monitoria remota dos equipamentos

- O sistema deve permitir que o usuário associe agentes a equipamentos, ou grupos de equipamentos;
- O sistema deve permitir que o usuário visualize, para cada equipamento gerenciado, uma lista dos eventos ocorridos;
- O sistema deve permitir que o usuário visualize, para cada equipamento monitorado, os resultados de execução dos agentes;

c. Monitoria de agentes

- O sistema deve permitir que o usuário faça o rastreamento dos agentes do sistema, ou seja, deve permitir que o usuário identifique quais agentes estão executando em quais equipamentos da rede;
- O sistema deve permitir que o usuário tenha informações sobre o estado atual de execução de um agente, ou seja, se ele está ativo ou não;
- O sistema deve permitir que o usuário altere o estado de execução de um agente, ou seja, ao usuário deve ser dado o direito de ativar ou interromper a execução de um agente;

- O sistema deve permitir que o usuário defina um intervalo de tempo para execução das tarefas dos agentes
- O sistema deve permitir a criação de um itinerário de execução para os agentes, ou seja, o sistema deve permitir que o usuário defina qual equipamento, ou grupo de equipamentos, será visitado por um agente;
- O sistema deve permitir que o agente informe o usuário de eventos ocorridos, tais como, suspeitas de invasão, falhas de execução, etc.
- O sistema deve permitir a criação de agentes através de uma linguagem de *scripts*;
- O sistema deve permitir que o usuário faça composição de agentes predefinidos, criando novos agentes com o comportamento resultante da execução daqueles agentes;
- O sistema deve permitir que o usuário associe recursos adicionais que possam ser necessários a execução de um agente, um exemplo de tal recurso seria o arquivo de uma biblioteca dependente de máquina que implementa captura de pacotes da rede;

d. Requisitos de qualidade

- O sistema deve ser de fácil manutenção, considerando-se que outras pessoas devem continuar o projeto;
- O sistema deve ser portável, considerando-se que as redes de computadores ainda são sistemas heterogêneos, ou seja, diversas arquiteturas e sistemas operacionais estão presentes;
- O sistema deve apresentar facilidade de uso (características de usabilidade), incluindo-se neste critério: facilidade de instalação, presença de interface gráfica, possibilidade de programação visual e uso de *scripts*, etc;
- O sistema deve possuir recursos avançados de segurança, pois se trata de uma tarefa crítica para o funcionamento da rede: gerenciar a segurança da mesma.

4. Modelagem para o Cenário de Execução dos Agentes no SDI

A seguir é apresentada uma modelagem para um sistema de detecção de intrusão baseado em agentes móveis e a definição do cenário de identificação de usuários anômalos. Este sistema tem por finalidade a redução dos custos apresentados por um SDI monolítico. Ele utilizará uma grande quantidade de pequenos agentes móveis para desempenharem todas as ações de monitoria, tomada de decisão, notificação e reação a tentativas de intrusão. Cada agente opera de forma independente um dos outros, porém, todos cooperam na monitoria do sistema formando um completo SDI. Como visto, esta abordagem apresenta vantagens significativas em termos de *overhead*, escalabilidade e flexibilidade.

Arquitetura do Sistema Proposto

A proposta do projeto é de, usando as vantagens principais de um sistema baseado em agentes móveis e autônomos - a simplicidade e a modularidade -, permitir a implementação de um cenário de execução destes agentes para fins específicos.

Com agentes trabalhando de forma cooperativa e trocando informações, uma atividade considerada suspeita por um agente é notificada a um grupo de outros agentes, sob forma de suspeita. A análise então é feita por agentes mais especializados naquela situação e, a partir daí, tomam decisões.

Isso demonstra que uma decisão deverá ser tomada em conjunto e nenhum agente possui a autonomia de identificação de uma intrusão. Essa decisão será tomada com base no consenso de vários agentes no sistema. Entretanto, se mais de um agente suspeitam de um comportamento anômalo, então há uma maior probabilidade de ser uma intrusão potencial e neste caso, poderá ser tomada a decisão de comunicar um operador humano ou acionar agentes especializados em contra ataque.

A Figura 4.1, já apresentada no projeto de pesquisa, apresenta a arquitetura (modelo em camadas) para o SDI proposto. As camadas são numeradas a partir da camada de Vigilância (camada 1), e cada uma delas representa um grupo de tarefas específicas desempenhadas por agentes especializados nas funções desta camada. Através do mecanismo de troca de mensagens, um agente em uma camada aciona um ou mais agentes em uma camada superior.

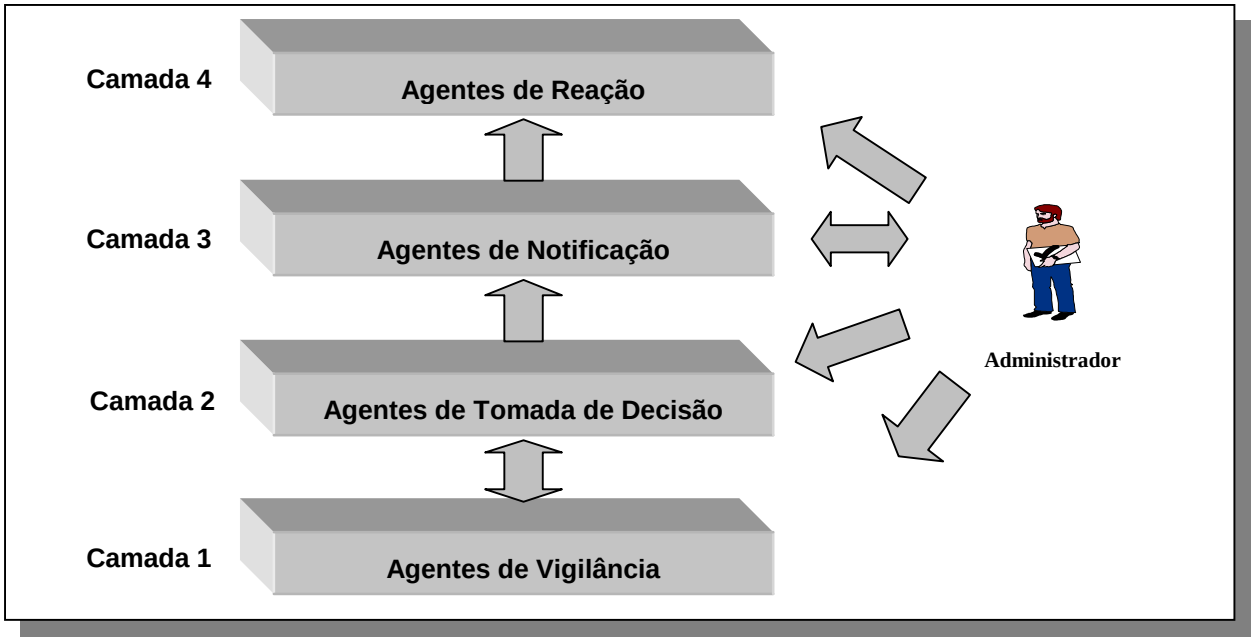

Figura 4.1 - Modelagem em Camadas para o Sistema Proposto

Com base em informações coletadas pelos *Agentes de Vigilância*, *Agentes de Tomada* de Decisão entrarão em ação, analisando e identificando possíveis intrusões. Caso uma ação seja considerada suspeita por estes agentes, *Agentes de Notificação* serão acionados e cuidarão de notificar o administrador da rede ou acionar os agentes de nível superior. Em último nível, encontram-se os *Agentes de Reação*. Estes agentes, cuidarão de contra-atacar automaticamente as possíveis intrusões, com base nas informações dos agentes de notificação ou ainda, serem acionados através de uma intervenção do administrador da rede.

Apesar do cenário acima exemplificar uma comunicação *bottom-up* através das camadas da arquitetura proposta, há a possibilidade de uma comunicação *top-down* entre a camada de tomada de decisão e a camada de vigilância.

A ampliação do SDI para atender uma nova configuração de ataque poderá envolver o desenvolvimento e conseqüente adição de novos agentes em uma única camada ou

ainda, a criação de um novo cenário que envolverá a adição de agentes em todas as camadas. A seguir, serão discutidas, com maiores detalhes, as funções de cada uma destas camadas.

A seguir, são explanadas as camadas que fazem parte da modelagem sugerida:

A Camada 1 – Agentes de Vigilância

Esta camada, representa o primeiro nível de agentes do sistema proposto. Este conjunto de agentes será responsável pela monitoria, coleta de informações, testes de ambiente e ajustes de configuração a partir de arquivos de perfil de segurança que serão alocados estrategicamente no ambiente ao qual se quer proteger.

Uma analogia para os Agentes de Vigilância seria compará-los a guardas noturnos em uma empresa. Estes, ao invés de serem alocados estaticamente em pontos estratégicos do ambiente, seriam responsáveis em fazer a "ronda" pelo sistema em busca de portas abertas ou padrões que caracterizem possíveis intrusões. A seguir, são relacionados um conjunto de agentes de vigilância e suas respectivas funções:

- **Agentes de verificação de usuários conectados ao sistema:** Agente responsável pela obtenção de uma lista contendo os usuários conectados a um servidor, a hora de início e a origem destas conexões;
- **Agente de verificação de hospedeiros:** agente responsável pela verificação dos hospedeiros em execução no ambiente.
- **Agente de captura de pacotes:** agente que trabalha em modo promíscuo na rede ativando a captura de pacotes (*sniffing*);
- **Agentes de *scanning*:** conjunto de agentes que, através de tentativas de quebra de segurança, irão validar os serviços, processos e demais atividades que vão contra a política de segurança do ambiente;
- **Agente de verificação de serviços:** agente responsável pela verificação dos serviços em execução em máquinas servidoras.
- **Agente de monitoria dos serviços:** agente responsável por monitorar e filtrar as requisições de entrada dos serviços da rede, fornecendo informações para agentes da camada superior;

- **Agente de verificação de utilização de recursos do sistema:** agente responsável pela monitoria da utilização de recursos de sistema como CPU e *Hard Disk*, em momentos considerados estratégicos e não-críticos;
- **Agente especializado em identificação de *backdoors*:** agente que, com base nos padrões conhecidos e divulgados, irá rastrear o sistema em busca de *backdoors*. Este agente deverá apresentar uma grande facilidade em sua capacidade de reconfiguração com base em novos conhecimentos adquiridos, tendo em vista que constantemente são relatadas novas formas desse tipo de ataque;
- **Agente de validação de serviços:** agente responsável pela validação dos serviços em execução. Cabe a este agente a configuração do *profile* que autoriza a execução de um ou mais serviços em uma máquina/servidor do ambiente;
- **Agentes de configuração de *Profiles*:** com base na percepção de novas formas de intrusão ou variação do perfil de utilização dos usuários, proceder com as devidas (re)configurações dos *profiles*. Esses arquivos irão conter informações de padrões de normalidade de utilização do sistema, como: padrão dos horários de conexão de usuários; estatísticas de utilização de dispositivos do sistema; etc.

A Camada 2 – Agentes de Tomada de Decisão

Nesta camada, encontram-se os agentes que exercem todas as funções de tomada de decisão no sistema, constituindo-se o "cérebro" do mesmo. Um agente desta camada irá receber uma mensagem ou um conjunto de dados dos agentes da camada inferior (a camada de Vigilância) e, com base em uma análise criteriosa destas informações, poderá identificar uma intrusão (ou tentativa), no momento de sua ocorrência, ou ainda, acionar novos Agentes de Vigilância para a coleta de informações complementares.

Em ações mais simples, estes agentes podem identificar uma anomalia, ou uso indevido, simplesmente comparando os dados obtidos com padrões de utilização do sistema (perfis de utilização). Entretanto, como esta camada representa o ponto de inteligência do sistema, deverão ser implementados agentes dotados de características de inteligência artificial para se alcançar um bom nível de reconhecimento de ações indevidas.

Entre essas características, é desejável que estes agentes sejam capazes de aprender coisas novas e se adaptarem a novas situações, em vez de simplesmente fazerem o que

lhes foi atribuído. Entre os estímulos de ativação da aprendizagem, deve-se utilizar as ações tomadas pelo administrador ao ser notificado pelos agentes da camada de nível superior (a camada de notificação). Dessa forma, fica evidente a necessidade do desenvolvimento de agentes que exerçam com funções de sistemas especialistas, o que poderá ser feito com a utilização das modernas técnicas de redes neurais e algoritmos genéticos, comumente referenciados em trabalhos relacionados à Inteligência Artificial.

A seguir são apresentados um conjunto de agentes de tomada de decisão e suas respectivas funções:

- **Agente de controle:** agente responsável pelo acionamento dos demais agentes. Constituindo-se o principal agente desta camada, este ficará em execução contínua à espera de uma mensagem de um agente da camada de vigilância. Ao receber e identificar a mensagem, este agente irá consequentemente acionar (ativar) o agente especializado em atender àquela chamada.

- **Agente de verificação de usuários anômalos:** agente que, baseado em um padrão de normalidade (*perfis* de usuário, como: horário de utilização, principais serviços utilizados, utilização de recursos do sistema, origem da conexão, etc), procura desvios do comportamento padrão, utilizando modelos estatísticos ou sistemas especialistas. Dessa forma, também será capaz de determinar possíveis personificações de usuários, baseando-se em uma análise do comportamento passado e na determinação de comportamentos que fujam ao padrão esperado de um comportamento aceitável;

- **Agente de verificação de hospedeiros não autorizados:** agente responsável pela verificação de existência de hospedeiros não autorizados na rede. Cabe a este agente a responsabilidade de validação dos ambientes servidores de agentes inseridos no ambiente;

- **Agente de Análise de pacotes:** agente que recebe os pacotes do agente de captura (Camada de Vigilância) e procede com sua análise. As atividades desempenhadas por este agente representam um nível de complexidade elevado, uma vez que será dotado de algum mecanismo de tomada de decisão;

- **Agente de identificação de pontos falhos:** agente que irá identificar os pontos falhos no ambiente com base nas informações dos agentes de *scanning.*

- **Agente de verificação de serviços:** agente responsável pela verificação da integridade e autorização dos serviços em execução em máquinas servidoras;
- **Agente de monitoria de *logs*:** agente que ativa a captura de informações de *logging* adicional quando um nível de periculosidade mais elevado é notificado;
- **Agente de monitoria de usuário indevido:** com base na detecção de uma possível personificação, proceder com o acompanhamento deste usuário em busca de assinaturas de ataque;
- **Agente de identificação de novos padrões de ataque:** agentes inteligentes responsáveis pelo aprendizado e identificação de novos padrões de ataque.

A Camada 3 – Agentes de Notificação

Esta é a camada menos populosa em quantidade de agentes. Os agentes desta camada são responsáveis por, com base nas mensagens recebidas da camada 2 (Agentes de Tomada de Decisão), notificar o administrador da rede e acionar os agentes da camada 4 (Agentes de Reação). Dessa forma, toda vez que os Agentes de Tomada de Decisão identificarem um nível de periculosidade acima do aceitável ou a necessidade de atualização de algum novo padrão identificado, Agentes de Notificação entrarão em ação.

A princípio, pode-se pensar que os agentes desta camada desempenham funções muito elementares, o que justificaria a agregação de suas funções à camada 4 (ocasionando a eliminação da camada 3). Entretanto, uma decisão tomada na camada de nível 2 poderá necessitar de diversas formas de notificação, ocasionando a construção de um agente muito complexo, quer seja agregando suas funções a esta camada ou em uma camada de nível superior do modelo. Isso iria contra a proposta de pequenos agentes desempenhando funções específicas na tentativa de minimizar a degradação do ambiente.

- **Agente de notificação de usuário indevido:** agente que irá alarmar a suspeita de uma personificação utilizando o sistema;
- **Agente de notificação de hospedeiro não autorizado:** agente que irá notificar a existência de hospedeiros não autorizados no ambiente;

- **Agente de notificação de tentativas de ataque:** agente que irá notificar as tentativas de conexão não autorizadas no ambiente;
- **Agente de notificação de serviço não autorizado:** agente que irá notificar a execução de algum serviço não-autorizado na rede;
- **Agente de notificação de furos de segurança:** Notificar furos de segurança provenientes de testes de *scanning;*
- **Agente de notificação de utilização de recursos de sistema:** agente que irá notificar a suspeita de utilização indevida de recursos do sistema;
- **Agente de notificação de tentativas de intrusão:** agente que irá notificar tentativas de intrusão por utilização de técnicas como quebra de senha por força bruta;
- **Agentes de acionamento de contra ataque:** Conjunto de agentes responsáveis por acionar cada ação de reação e contra-ataque;
- **Agente de notificação de identificação de novos padrões:** Agente que irá notificar a necessidade de reconfiguração ou desenvolvimento de novos agentes para atender um novo padrão de intrusão identificado.

A Camada 4 – Agentes de Reação

Esta camada apresenta um conjunto de agentes que serão acionados pelos agentes da camada 3 (Agentes de Notificação) ou ainda, pela ação direta do administrador humano. Responsáveis por reagir (contra atacar), recuperar e reconfigurar o sistema, estes agentes representam a última instância de recursos do sistema modelado.

- **Agentes de encerramento de conexão:** agente responsável por cancelar e bloquear uma conexão para um possível usuário anômalo;
- **Agentes de exclusão de hospedeiros:** agente responsável pela remoção de hospedeiros não autorizados no ambiente;
- **Agente de corte de conexão:** agente responsável pelo acionamento do corte e/ou bloqueio da conexão de um possível intruso;

- **Agentes de eliminação de furos de segurança:** conjunto de agentes responsáveis pela reconfiguração dos serviços na tentativa de eliminação de furos de segurança;
- **Agente de bloqueio de serviço:** agente responsável pelo bloqueio da execução de serviços não autorizados;
- **Agente de recuperação de arquivos alterados:** agente que cuida da recuperação do estado inicial, baseado em um repositório, caso seja efetuada uma alteração não validada em um arquivo,
- **Agentes de reconfiguração:** agente responsável pela recuperação do estado anterior de serviços autorizados;
- **Agente de ativação de dispositivos auxiliares:** agente responsável pela ativação de dispositivos auxiliares à captura de informações sobre o usuário interno. Um exemplo de um dispositivo auxiliar poderia ser uma câmera de vídeo.

Classificação do sistema

Conforme visto anteriormente, as principais metodologias de classificação para os Sistemas de Detecção de Intrusão são expressas em termos da forma como o sistema aborda o problema de detectar a intrusão e em termos do tratamento dos dados.

Uma vez que a detecção de anomalia identifica atividades intrusivas como sendo um subconjunto que foge da atividade normal, o sistema proposto possui um conjunto de agentes que procuram quantificar o comportamento usual ou aceitável, armazená-lo em *profiles* de usuário e posteriormente, identificar outros comportamentos irregulares como intrusivos. Entretanto, o sistema também possui agentes que procuram por ataques que podem ser precisamente identificados pela forma como acontecem, ou seja, intrusões que seguem um padrão bem definido de ataque (assinaturas de ataque), características estas do modelo de detecção de uso indevido (abuso).

Assim, em relação à forma com que aborda o problema de detectar a intrusão, o sistema proposto apresenta-se como um híbrido entre o modelo de detecção de anomalia e o modelo de detecção baseado em uso indevido. Isso também pode ser considerado uma vantagem significativa deste sistema, uma vez que sistemas híbridos monolíticos

apresentam-se como sistemas complexos que implicam em severas penalidades de performance no ambiente a ser monitorado, o que não ocorre com a proposta modular.

Em termos de tratamento dos dados, o sistema apresenta-se como um híbrido entre um modelo baseado em *host (host based)* e um modelo baseado em redes *(network based).* Entre as características de um sistema baseado em *host*, o sistema apresenta um conjunto de agentes que, baseados em perfis de utilização de um equipamento, procuram por desvios de comportamento padrão, utilizando-se modelos estatísticos ou sistemas especialistas. Entretanto, também possui agentes que monitoram o tráfego da rede, capturam pacotes e procuram por "impressões digitais" do ataque, acontecendo em tempo real.

O Cenário de Execução do Ambiente Modelado com DFD

Para a representação do cenário de execução do ambiente proposta para este trabalho, foi utilizada uma ferramenta lógica conhecida como Diagrama de Fluxo de Dados (*DFD)*, por ser uma ferramenta simples, de fácil entendimento e largamente utilizada na modelagem de sistemas de informação. A figura 4.2 - já apresentada no projeto de pesquisa -, destaca os símbolos utilizados e sua representação neste contexto. Maiores informações sobre DFDs podem ser observadas em [GANE, 1988] e [GANE & SARSON, 1994].

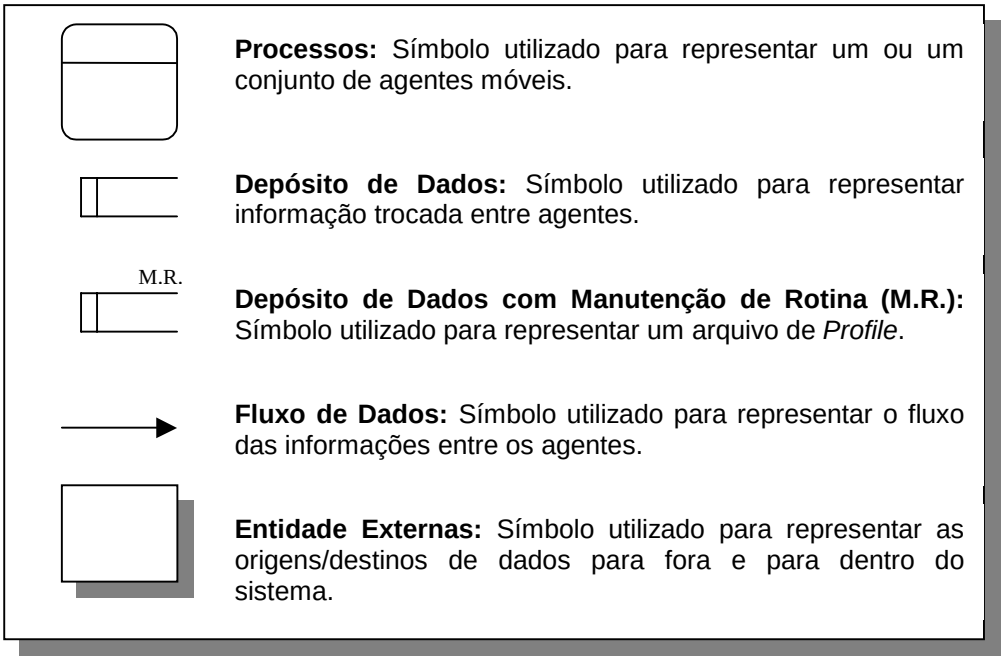

Figura 4.2- Representação dos elementos de DFD utilizados na modelagem

A seguir, será apresentado o modelo (utilizando-se DFDs), para o cenário de execução do sistema. Este cenário está representado por um modelo de alto nível composto por quatro processos que estão associados respectivamente a cada uma das camadas do modelo apresentado na figura 4.1. Assim, a numeração utilizada em cada processo, além de identificar a seqüência lógica de execução do cenário, faz a associação do processo à sua respectiva camada no modelo. Como exemplo, temos que o processo 1 contém um ou mais agentes da camada 1 (a Camada de Vigilância) e assim sucessivamente.

Dependendo da complexidade, um processo (agente) neste modelo de alto nível pode ser "expandido" para tornar-se um novo diagrama (conjunto de agentes). Cada cenário representa uma fronteira de automação para o desempenho de uma função de segurança computacional no sistema proposto. Assim, a criação de novos cenários de execução corresponderá à implementação e posterior inserção de novas funções no sistema proposto.

O cenário de Identificação de Usuários Anômalos

Um intruso (interno ou externo) que consiga um *username* e uma senha válidos, poderá abrir uma conexão e ter acesso ao sistema. Uma vez dentro do sistema (mesmo em acesso comum, ou seja, não como *root*) poderá usar diversas técnicas disponíveis para executar suas intenções, conseguir privilégios de *root* ou ainda, agir ilegalmente em nome do usuário legítimo.

Na tentativa de identificação deste tipo de ataque, este cenário apresenta as características de um Sistema de Detecção de Intrusão baseado em anomalias. O objetivo é analisar comportamentos que possam identificar conexões que fujam dos padrões previamente estabelecidos para cada usuário. Para isso, o sistema possui: agentes que coletam uma lista contendo informações dos usuários conectados ao sistema (horário de *login*, origem da conexão, etc); agentes que irão proceder com a identificação de possíveis anomalias tomando como base *profiles* previamente estabelecidos para os usuários; agentes de notificação do grau de suspeita e ainda, agentes para execução de medidas reativas, tais como bloqueio de conexão.

O fluxo lógico da ação dos agentes descritos acima é modelado e apresentado na figura 4.3. Conforme pode ser observado, apesar de todos os processos representarem

agentes autônomos que estarão em ação sem a supervisão humana, o administrador poderá interagir diretamente com cada um deles, utilizando-se das primitivas de controle de agentes. Entre as primitivas encontram-se: criação, clonagem, envio, recuperação, suspensão de execução, interrupção de processamento, etc. Isso também é válido para os demais cenários apresentados.

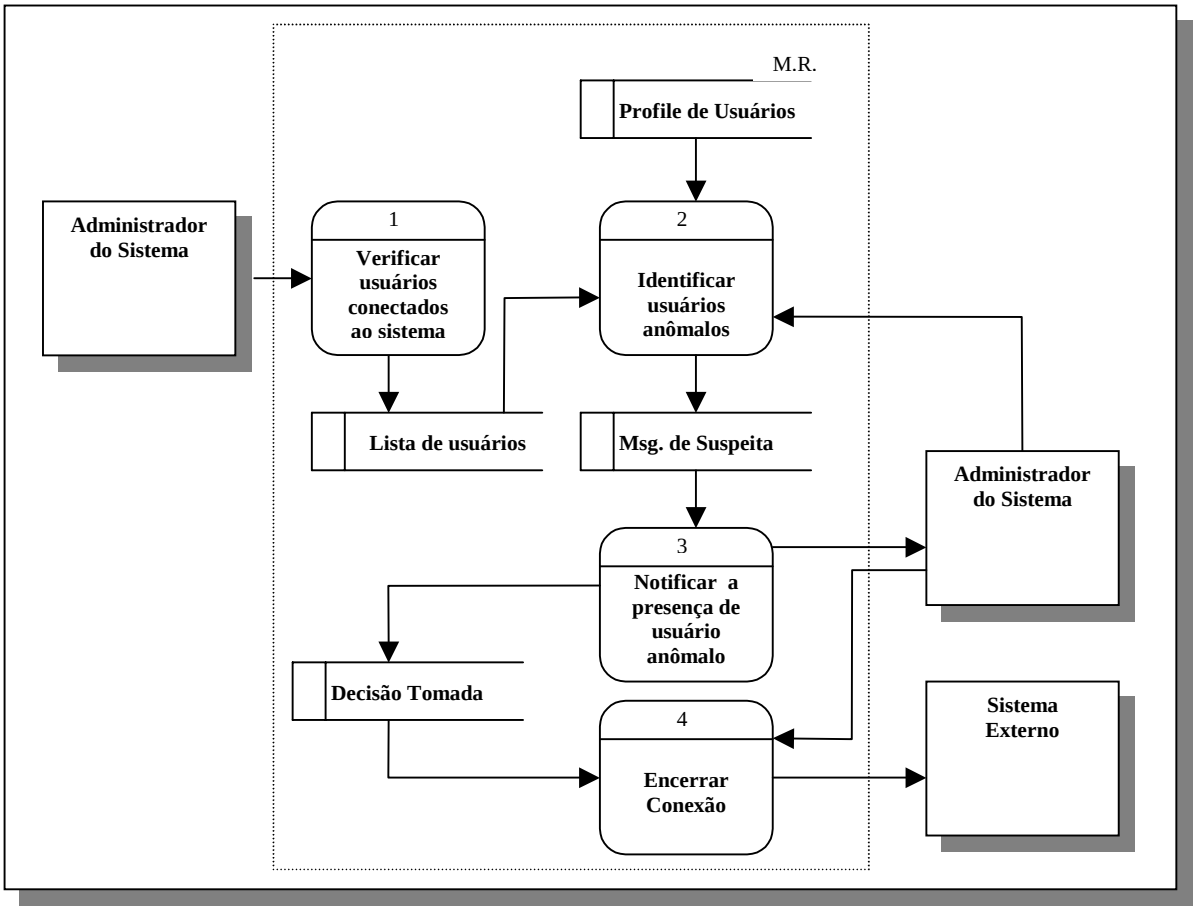

Figura 4.3 - Modelagem do Cenário de Identificação de Usuários Anômalos

Características desejáveis do ambiente

No desenvolvimento de um ambiente de detecção de intrusão conforme o que discutimos até o presente momento, algumas características tornam-se desejáveis e desta forma, o sistema deve:

- estar em execução contínua com o mínimo possível de supervisão humana;
- ser capaz de executar em *background* no ambiente que está sendo observado e relatar suas conclusões;

- impor o mínimo de *overhead* possível no ambiente onde está sendo executado, de forma a não interferir nas operações normais;
- apresentar facilidades de configuração para atender a política de segurança do ambiente que está monitorando;
- adaptar-se às mudanças do ambiente e dos comportamentos dos usuários através do tempo (permissão para execução de novas aplicações, usuários trocando de atividade ou novos recursos sendo utilizados);
- monitorar um grande número de *hosts* enquanto providencia resultados em tempo hábil e de maneira exata;
- apresentar uma moderada degradação de serviço, uma vez que algum componente do sistema pode ter seus serviços interrompidos por qualquer razão, o resto deverá ser afetado da menor forma possível;
- Permitir reconfiguração dinâmica, uma vez que um grande número de *hosts* está sendo monitorado, torna-se impraticável reinicializar o SDI toda vez que um novo cenário for implementado.

5. Avaliação da Tecnologia de Implementação: Java Aglets

O *Aglet* é um agente móvel Java que suporta os conceitos de execução autônoma e roteamento dinâmico em seu itinerário. Conforme pode ser observado na figura 5.1, *Aglets* são hospedados por um servidor *Aglet* de forma similar à forma como os *applets* são hospedados por um *Web browser*. O servidor *Aglet* provê um ambiente para os *Aglets* executarem e uma *Java Virtual Machine (JVM)* que, juntamente com o gerenciador de segurança do *Aglet,* tornam este servidor seguro para receber e hospedar os *Aglets.*

Desta forma, a origem da palavra *Aglet* é simples: ela deriva do termo "*lightweight agent"* da mesma forma que um *Applet* deriva de "*lightweight application*". O termo *Aglet* é uma combinação das palavras *Agent* e *Applet.*

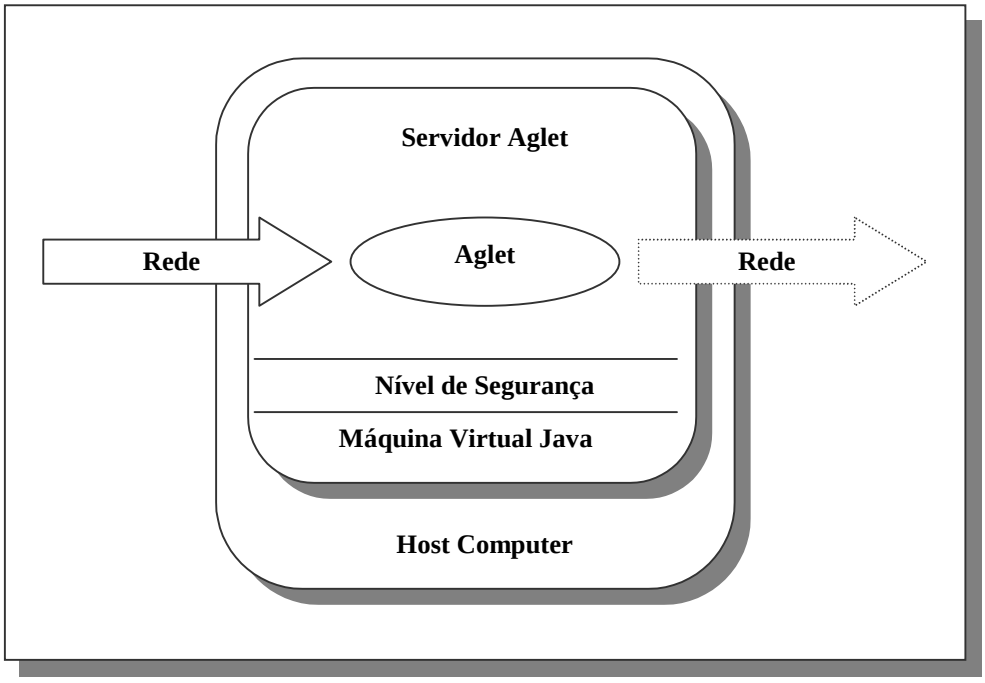

Figura 5.1 - Ambiente Seguro para *Aglets* visitantes [LANGE & OSHIMA, 1998]

A API *Aglet* é um kit de desenvolvimento de agentes ou , em outras palavras, um conjunto de classes Java e interfaces que permitem a criação de agentes móveis Java. O grupo *IBM Tokyo Research Laboratory* desenvolveu a API *Aglet* no Japão, tendo sua

primeira versão liberada em 1995 em resposta à necessidade de uma plataforma uniforme para agentes móveis em ambientes heterogêneos como a Internet

Sendo a API *Aglet* desenvolvida em Java, um agente móvel desenvolvido com sua utilização será capaz de executar em qualquer máquina que suporta Java. Não é preciso informações sobre o hardware ou sobre o sistema operacional e dessa forma, a API *Aglet* espelha o modelo de *applet* em Java.

Uma implementação da API *Aglet*, denominada ASDK, pode ser encontrada para *download* no Web site da IBM Tokyo Research Laboratory (http://www.trl.ibm.co.jp/aglets). O ASDK inclui o pacote API *Aglet*, documentação, exemplos de *Aglets*, e o servidor de *Aglets* denominado Tahiti, apresentado na figura 5.2. O Tahiti é uma aplicação Java que permite ao usuário receber, gerenciar, e enviar *Aglets* para outros computadores que estão executando o Tahiti.

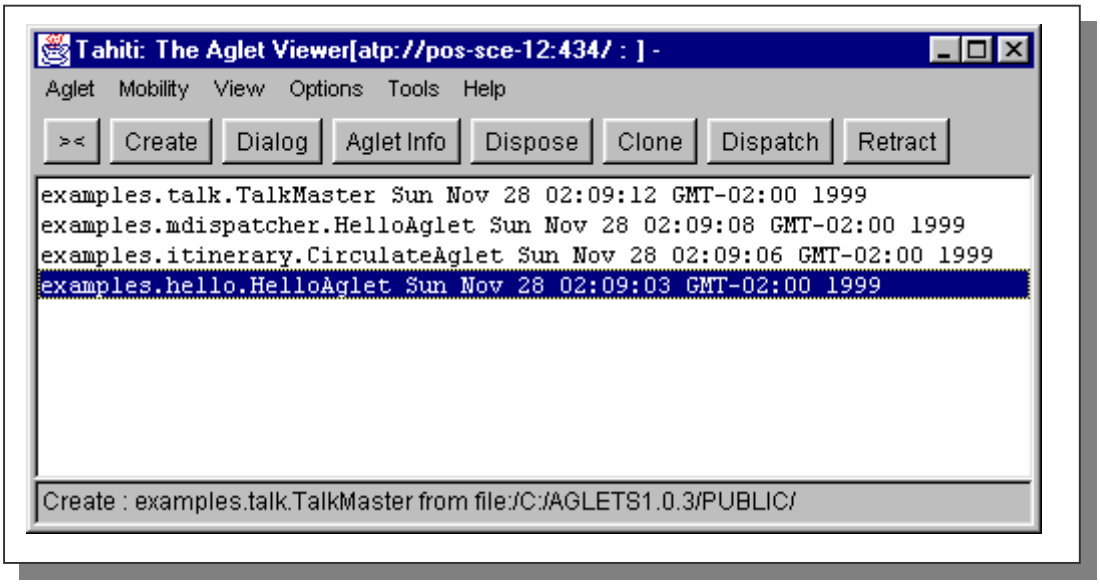

Figura 5.2 - Tahiti: Ambiente de Hospedagem de *Aglets* com 4 agentes em execução

A migração de um *Aglet* inicia-se com a interrupção da execução do *Aglet* na máquina origem, seu envio para uma máquina remota e o reinicio da execução após sua chegada ao destino. Mecanismos de segurança permitem a configuração dos direitos para que *Aglets* não autorizados (*untrusted*) não tenham acesso a determinados recursos do sistema, tornando o sistema implementado com *Aglets* seguro.

A biblioteca de classes de *Aglets* foi concebida pelos pesquisadores da IBM tendo os seguintes objetivos:

- Fornecer um modelo completo e simples de programação para a utilização de agentes móveis, sem no entanto implicar em modificações na máquina virtual Java ou em código nativo;
- Disponibilizar mecanismos de comunicações poderosos e dinâmicos que permitissem agentes comunicarem-se com outros agentes, fossem eles conhecidos ou não;
- Projetar uma arquitetura de agentes móveis que permitisse extensibilidade e reusabilidade;
- Obter uma arquitetura altamente coerente com o modelo tecnológico Web/Java.

Baseado ainda nesta biblioteca, foi desenvolvido o *Fiji kit* que permite criar *applets* na qual executam contextos *Aglets* e podem criar, despachar e receber agentes de páginas Web.

Modelo Aglet

Esse modelo foi desenvolvido para beneficiar as características dos agentes de Java enquanto supera algumas das deficiências da linguagem. No modelo de objeto *Aglet*, um agente móvel é um objeto móvel que tem sua própria *thread* de controle, é orientado a evento e comunica por passagem de mensagem.

Esse modelo define um conjunto de abstrações e o comportamento necessário para alavancar a tecnologia de agentes móveis redes de longa distância abertas, como a Internet. A abstração chave é composta por: *Aglet*, *proxy*, contexto e identificador.

- ***Aglet*** - Um *Aglet* é um objeto Java móvel que visita *hosts* habilitados em uma rede de computadores.
- ***Proxy*** - Um *proxy* é uma representação de um *Aglet*. Ele serve como uma proteção para o Aglet contra acessos diretos a seus métodos públicos (figura 5.3). O *proxy* também fornece transparência de localização para o *Aglet*; isto é, ele pode ocultar a localização real dos *Aglets*. Isso significa que um *Aglet* e seus *proxies* podem estar separados, de forma que um *proxy* local oculta à distância o *Aglet*.

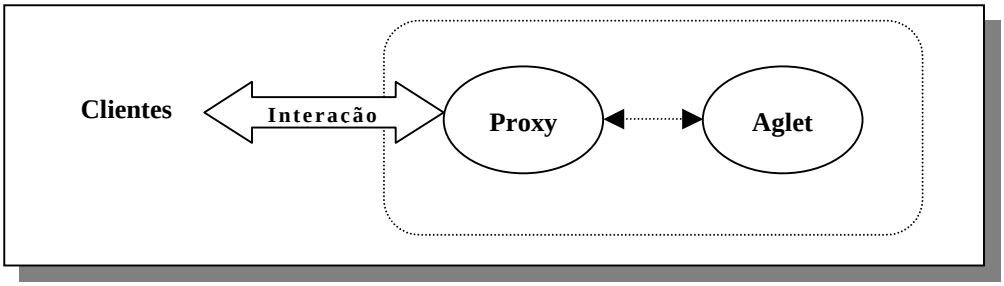

Figura 5.3 - Relacionamento entre *Aglet* e *Proxy* [LANGE & OSHIMA, 1998]

- ***Contexto*** - O contexto é o lugar de trabalho do *Aglet*. Ele é um objeto estático que fornece um meio de manter e gerenciar *Aglets* rodando em um ambiente de execução uniforme onde o sistema *host* é protegido contra *Aglets* maliciosos (figura 5.4). Um nó em uma rede de computadores pode rodar múltiplos processos servidores e cada servidor pode hospedar múltiplos contextos. Contextos são nomeados e assim podem ser localizados pela combinação do endereço do servidor e seu nome.

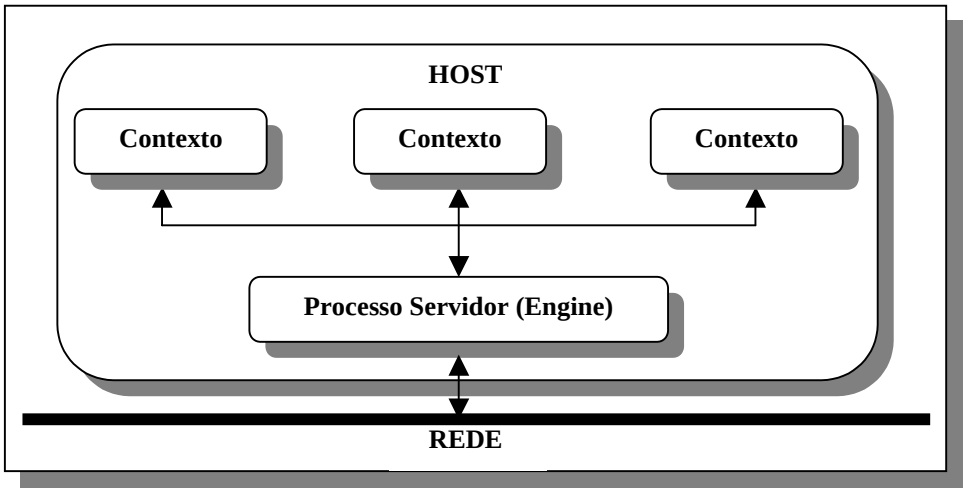

Figura 5.4 - Relacionamento entre Host, Processo Servidor (Engine) e Contextos

- ***Identificador*** - Um identificador é associado para cada *Aglet*. Esse identificador é único e imutável ao longo do ciclo de vida do *Aglet*.

Os comportamentos de um modelo de objeto *Aglet* são baseados em uma cuidadosa análise da vida e morte de agentes móveis. Há somente duas formas de dar vida a um

Aglet: uma é instanciá-lo no momento de sua criação e outra é copiá-lo de um *Aglet* existente (*clone*). Para controlar a população de *Aglets* pode-se, é claro, destruí-los (*disposal*). *Aglets* são móveis de duas formas: ativa e passiva. Em uma abordagem ativa, eles movem-se por conta própria do *host* corrente para o *host* remoto (*dispatching*). O *host* remoto invocando o *Aglet* do *host* corrente (*retracting*) caracteriza o tipo passivo de mobilidade do *Aglet*. Quando *Aglets* estão em execução, eles consomem recursos. Para reduzir esse consumo, *Aglets* podem ser colocados em modo dormente temporariamente, liberando seus recursos (*deactivation*) e, posteriormente, podem ser colocados novamente no modo de execução (*activation*). Finalmente, múltiplos *Aglets* podem trocar informações para completar uma determinada tarefa (*messaging*).

Este é o conjunto mínimo de operações necessárias para criar e gerenciar um ambiente de agentes móveis distribuído.

A seguir são apresentadas as operações fundamentais de um *Aglet*. Estas, são representadas graficamente na figura 5.5:

- ***Creation*** - um novo *Aglet* é associado a um identificador, inserido dentro do contexto e inicializado.
- ***Cloning*** - o *cloning* de um Aglet produz uma cópia quase idêntica do *Aglet* original no mesmo contexto. Somente as diferenças de identificador e o fato que a execução inicializa um novo *Aglet*. *Threads* de execução não são clonadas.
- ***Dispatching*** - despachar um *Aglet* de um contexto para outro irá removê-lo do contexto corrente e inseri-lo no contexto destino, onde ele reiniciará a sua execução (*threads* de execução não migram). Diz-se que o *Aglet* foi enviado para um novo contexto.
- ***Retraction*** - a recuperação de um *Aglet* irá chamá-lo (removê-lo) do contexto corrente e inseri-lo no contexto do qual a chamada foi realizada.
- ***Activation*** e ***deactivation*** - a desativação de um *Aglet* é a habilidade de, temporariamente, parar sua execução e armazenar o seu estado em um dispositivo de armazenamento secundário. A ativação de um *Aglet* irá restaurá-lo no mesmo contexto.

- ***Disposal*** - a liberação de um *Aglet* irá terminar sua execução e removê-lo do contexto corrente.

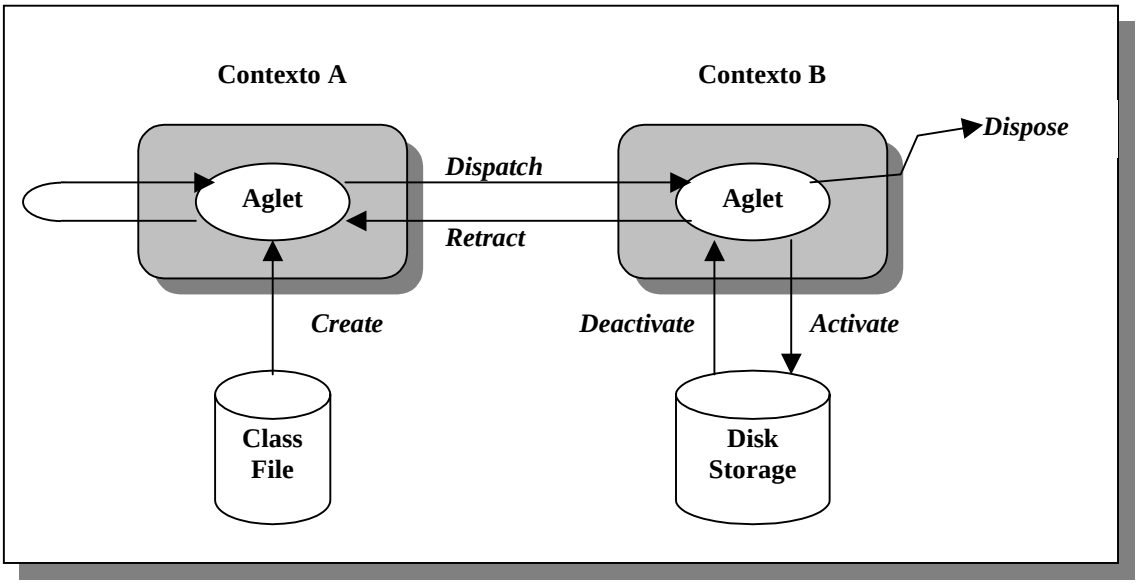

Figura 5.5 - Modelo do Ciclo de vida de um *Aglet*

6. Conclusões e Atividades Previstas para o Próximo Período

Como conclusões de destaque até este estágio do trabalho, pode-se listar a aprovação prática da plataforma de desenvolvimento para o projeto, a linguagem Java em conjunto com o conjunto de bibliotecas Aglets.

Foi verificada, através de trabalhos de pesquisa já existentes [BERNARDES, 1999; PEREIRA FILHO, 2000] a viabilidade do uso desta plataforma específica para agentes móveis em ambientes críticos de segurança e a sua aplicação em outros trabalhos, sejam corporativos ou acadêmicos [KARJOTH, 1997; MILAGRES, 2000].

Dentre as atividades previstas para o próximo período deste trabalho, destaca-se a codificação dos agentes na plataforma definida, o teste e a validação do cenário composto por estes agentes no sistema de detecção de intrusões em questão.

A integração com outros trabalhos em desenvolvimento no Laboratório Intermídia e a submissão de trabalhos para publicações técnicas também está incluída como uma atividade para este período, como foi feito no período inicial de trabalho, seguindo o cronograma mostrado no início deste relatório.

7. Atividades Extracronograma Realizadas

- 24 a 26 de Outubro/2000 - Participação do II Simpósio Segurança em Informática, no ITA/São José dos Campos, evento que contou com apresentações de artigos, explanações de grupos de pesquisas e empresas de segurança em informática brasileiras e estrangeiras.
- Janeiro/2001 - artigo *"Mobilidade na Segurança Corporativa"* escrito e aprovado para publicação na *Developers' Magazine CEO*, (http://www.developers.com.br) de fevereiro de 2001.
- Atuação, desde o início do projeto, como um dos administradores da rede e dos equipamentos do Laboratório Intermídia.

8. Referências

[BARRUS & ROWE 1998]	BARRUS, J.; ROWE, N.C. *A Distributed Automous-Agent Network-Intrusion Detection and Response System*. In: Proceedings of the 1998 Command and Control Research and Technology. Monterrey CA, Junho-Julho 1998.
[BERNARDES, 1999]	BERNARDES, M.C. *Avaliação do uso de agentes móveis em segurança computacional*. São Carlos : ICMC/USP, 1999. 105p. (Dissertação de Mestrado). Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo.
[BERNARDES, 2000]	BERNARDES, M. C.; MOREIRA, E.S.*; An Architecture for an Intrusion Detection System Based on Mobile Agents*, International Symposium on Advanced Distributed Systems, Guadalajara, Mexico, Mar/2000
[BERNARDES, 2000a]	BERNARDES, M.C.& MOREIRA, E.S. *Implementation of an Intrusion Detection System Based on Mobile Agents*. Trabalho aceito para publicação e apresentação no 5th International Symposium on Software Engineering for Parallel and Distributed Systems and 22nd International Conference on Software Engineering (ICSE2000). June 10-11-2000, Limerick, Ireland. Patrocínio: IEEE.
[BONIFÁCIO Jr, 1998]	BONIFÁCIO Jr., J.M. *Sistemas de segurança distribuído: integração de firewalls com sistemas de detecção de intrusão*. São Carlos : ICMC/USP, 1998. P. (Dissertação de Mestrado). Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo.
[BONIFÁCIO JR. et al., 1998a]	BONIFÁCIO Jr., J.M., CANSIAN, A.M., CARVALHO, A.C.P.L., MOREIRA, E.S. *Neural networks applied in intrusion detection systems*. In: IEEE International Joint Conference on Neural Networks - IJCNN '98. Proceedings**...** Anchorage, Alaska : IEEE, 1998. P.
[BONIFÁCIO JR. et al., 1998b]	BONIFÁCIO Jr., J.M., CANSIAN, A.M., CARVALHO, A.C.P.L., MOREIRA, E.S. *Um ambiente de segurança distribuído para a integração de firewalls com sistemas de detecção de intrusão.* In: Brazilian Symposium on Computer Networks – SBRC'98, XVI, 1998. Proceedings**...** Rio de Janeiro : ed., 1998. P.
[CANSIAN et al., 1997a]	CANSIAN, A.M., MOREIRA, E.M., CARVALHO, A.C.P.L., BONIFÁCIO Jr., J.M. *Network intrusion detection using neural networks*. In: International Conference on Computational Inteligence and Multimedia Applications, ICCIMA'97. Proceedings... Gold Coast, Australia : ed., 1997. p. 276-280.
[CANSIAN et al., 1997b]	CANSIAN, A.M., MOREIRA, E.M., MOURO, R.B., MORISHITA, F.T., CARVALHO, A.C.P.L. *An adaptative system for detecting intrusion in networks*. In: International Congress on Information Engineering, III. Proceedings**.** Buenos Aires, Argentina : 1997. p. 96-105.
[CANSIAN ET AL., 1997c]	CANSIAN, A.M., MOREIRA, E.M., CARVALHO, A.C.P.L., BONIFÁCIO Jr., J.M. *Um modelo adaptativo para detecção de comportamento suspeito em redes de computadores*. In: Brazilian Symposium on Computer Networks, SBRC'97, XV. Proceedings... São Carlos : SBRC, 1997. p. 51-60.

[CANSIAN, 1997]	CANSIAN, A.M. *Desenvolvimento de um sistema adaptativo de detecção de intrusos em redes de computadores.* São Carlos : IFSC/USP, 1997. 153p. (Tese de Doutorado). Instituto de Física de São Carlos, Universidade de São Paulo.
[CROSBIE & SPAFFORD 1995a]	CROSBIE, M.; SPAFFORD, E.H. *Defending a Computer System using Autonomous Agents*. Department of Computer Sciences, Purdue University, 1995. (Relatório Técnico CSD-TR-95-022; Coast TR 95-02). Disponível em: http://www.cerias.purdue.edu/homes/spaf/tech-reps/9508.ps. Visitado em 29/01/2001.
[CROSBIE & SPAFFORD 1995b]	CROSBIE, M; SPAFFORD, E.H. *Active Defense of a Computer System using Autonomous Agents*. Department of Computer Sciences, Purdue University, 1995. (Relatório Técnico CSD-TR-95-008). Disponível em: http://www.cerias.purdue.edu/homes/spaf/tech-reps/9522.ps. Visitado em 29/01/2001.
[CSI, 2000]	Computer Institute Security Press Release. [on-line]. [Citado em 12/04/2000]. Disponível na internet : http://www.gocsi.com/prelea_000321.htm
[GANE & SARSON, 1994]	GANE, Chris; SARSON, Trish. *Análise Estruturada de Sistemas*. Rio de Janeiro: LTC-Livros Técnicos e Científicos Editora, 1994.
[GANE, 1988]	GANE, Chris. *Desenvolvimento rápido de sistemas*. Rio de Janeiro: LTC-Livros Técnicos e Científicos Editora, 1988
[KARJOTH, 1997]	KARJOTH, G., et. al. *A Security Model for Aglets*. IEEE Internet Computing, Julho/Agosto - 1997. http://computer.org/internet/
[LANGE & OSHIMA, 1998]	LANGE D.B. and Oshima M. *Programing and Deploying Java Mobile Agents with Aglets*. 2nd ed. Lange D.B. and Oshima M.Addison-Wesley, 1998
[MILAGRES, 2001]	MILAGRES, Francisco G. *Mobilidade na Segurança Corporativa*. Developers' Magazine CEO. Fevereiro, 2001.
[MODULO, 2000]	MÓDULO Security Solutions S.A. [on-line] *Pesquisa Nacional de Segurança da Informação*. http://www.modulo.com.br , 2000.
[NWANA, 1996]	NWANA, H.S. *Software agents : An overview*. *Knowledge Engineering Review***,** v. 11, no. 3, p. 205-244, Out./Nov. 1996.
[PEREIRA FILHO, 2000]	PEREIRA FILHO, S. F. *Avaliação de Ambientes Servidores para Agentes Móveis*. Mini-Dissertação de Mestrado. ICMC , USP, 2000.
[PRESSMAN, 1995]	PRESSMAN, Roger S. *Engenharia de Software*. São Paulo: Makron Books, 1995.
[REAMI, 1998]	REAMI, E.R. *Especificação e prototipagem de um ambiente de gerenciamento de segurança apoiado por agentes móveis.* São Carlos : ICMC/USP, 1998. 82p. (Dissertação de Mestrado). Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo.
[SPAFFORD et al., 1998]	SPAFFORD, E., BALASUBRAMANIYAN, J.S., FERNANDEZ, J.O.G., ISACOFF, D., ZAMBONI, D**.** *An architecture for intrusion detection using autonomous agents*. 1998. (COAST Technical Report 98/05).
[ZAMBONI ET AL, 1998]	ZAMBONI, Diego; BALASUBRAMANIYAN, Jai; GARCIA-FERNANDES, Jose Omar and SPAFFORD, E. H.; *An Architecture for Intrusion Detection using Autonomous Agents* Department of Computer Sciences, Purdue University; Coast TR 98-05; 1998. Disponível em: http://www.cerias.purdue.edu/homes/aafid/docs/zamboni9805.pdf Visitado em 29/01/2001.

São Carlos, 29 de Janeiro de 2001.

Bolsista:	*Orientador:*
Francisco Gomes Milagres	Prof. Dr. Edson dos Santos Moreira