Mecanismos de Segurança para Plataformas de Agentes Móveis, baseados em Redes de Confiança SPKI/SDSI

Michelle Silva Wangham*, Joni da Silva Fraga

Departamento de Automação e Sistemas - LCMI-DAS-UFSC

Universidade Federal de Santa Catarina - C.P. 476 – CEP 88040-900– Florianópolis

{wangham, fraga}@das.ufsc.br

Resumo. *Este trabalho define um esquema de segurança para proteção das plataformas de agentes móveis baseado em redes de confiança SPKI/SDSI, considerando sistemas distribuídos e de larga escala. O esquema é composto por um protocolo de autenticação mútua para as plataformas envolvidas, por um autenticador de agentes móveis e por um mecanismo para geração de domínios de proteção. Devido à flexibilidade dos mecanismos de delegação de certificados SPKI/SDSI adotados, o esquema proposto fornece um controle descentralizado de autorização e de autenticação.*

Abstract. *This work defines a security scheme for protecting mobile agents plataforms, which is based on SPKI/SDSI trust networks in a large-scale distributed systems context. The scheme comprises a mutual authentication protocol for the platforms involved, a mobile agent authenticator and a mechanism for generation of protection domains. As the mechanisms for certificate delegation in SPKI/SDSI are very flexible, the proposed scheme effectively provides decentralized authorization and authentication control.*

1. Introdução

Novas técnicas, linguagens e paradigmas têm surgido para facilitar a criação de aplicações distribuídas em diversas áreas. Talvez o mais promissor entre os novos paradigmas seja o que incorpora a noção de agente móvel. Um agente móvel em uma rede de larga escala pode ser visto como um agente de software que viaja através de uma rede heterogênea, atravessando diversos domínios de segurança sendo executados de maneira autônoma no seu destino.

O movimento de um agente em um sistema distribuído é caracterizado pela transferência de seu código e estado (OMG 2000). Neste contexto, o termo estado do agente é o seu estado de execução e valores de atributos do agente que determinam o que deve ser feito quando a execução for retomada no ambiente computacional destino. A utilização de agentes móveis implica a necessidade de serviços extras nas máquinas onde esses códigos irão ser executados. Cada máquina necessita de um ambiente computacional, normalmente identificada como plataforma de agentes, cujo propósito é dar suporte às aplicações para realocar dinamicamente seus componentes de software sob diferentes sítios. Esta plataforma é usada para criar, interpretar, transferir e retomar agentes.

Apesar das suas vantagens, a ampla adoção do paradigma de agentes móveis vem sendo retardada devido à preocupação com a **segurança**. Os mecanismos disponíveis hoje para contornar os riscos desta tecnologia não cobrem com eficiência todas as ameaças

* Bolsista CNPq - Brasil

existentes, além de introduzirem restrições de desempenho, que, muitas vezes, negam os benefícios do uso de agentes móveis para certas aplicações (Jansen e Karygiannis 2000).

Devido às características do paradigma de agentes móveis e às ameaças a que este está exposto, os mecanismos de segurança são voltados para a proteção do suporte de comunicação, da plataforma de agentes e dos próprios agentes. O trabalho descrito neste artigo se concentra na problemática da segurança das plataformas de agentes, considerando sistemas distribuídos de larga escala.

Diante dos possíveis ataques e ações falhas de agentes móveis, os recursos e funcionalidades das plataformas de agentes devem ser protegidos. O estabelecimento de domínios isolados para cada agente, combinado a um controle de acessos entre domínios, é uma abordagem que vem sendo comumente adotada com a finalidade de oferecer esta proteção. Em termos tradicionais, o conceito de gerenciar o acesso é conhecido como um monitor de referência. Implementações do conceito de monitor de referência usam diversas técnicas de segurança que são também aplicáveis a ambientes de agentes. Porém, a proteção contra agentes maliciosos não se restringe à limitação de seus acessos a domínios próprios de execução em plataformas de agentes. Outras questões precisam ser tratadas quando sistemas distribuídos de larga escala são considerados. Por exemplo, a geração destes domínios depende da autenticação e dos atributos de privilégios (credenciais) dos correspondentes agentes. Logo, quando estes sistemas de larga escala são considerados, a implantação de mecanismos de autenticação e de autorização se caracteriza como um grande desafio.

Uma plataforma, para se proteger de um agente, depende não só da verificação da autenticidade do proprietário do agente, mas também do grau de confiança das plataformas já visitadas pelo agente, já que um agente móvel pode se tornar malicioso em virtude do seu estado ter sido corrompido por plataformas visitadas anteriormente. Garantir a confiabilidade e a segurança dos agentes *multi-hop* com itinerários livres, já que não se conhece previamente o seu destino e as plataformas visitadas, não é tarefa fácil e ainda não possui uma solução efetiva (Ordille 1996).

Baseado em um controle descentralizado de autorização que se vale dos mecanismos de delegação de certificados SPKI/SDSI (Ellison 1999), este trabalho tem por objetivo definir um esquema para proteger as plataformas de agentes contra agentes móveis maliciosos para sistemas de larga escala. Este esquema inclui técnicas de prevenção, autenticação mútua de plataformas e esquema para geração de domínios de proteção, e técnica de detecção, autenticação de agentes móveis *multi-hop*. Além disso, para atender às necessidades específicas de aplicações, este esquema pretende ser flexível de modo que o mesmo possa ser especializado através da seleção de um subconjunto de mecanismos e possa também ser próprio para as funcionalidades das aplicações[1].

2. Segurança das Plataformas de Agentes Móveis

A literatura aponta um conjunto de ameaças a que as plataformas de agentes móveis estão sujeitas, entre elas estão (Jansen e Karygiannis 2000): a **personificação,** quando um agente se passa por um agente autorizado com o intuito de obter acesso a serviços e recursos locais

[1] Os resultados obtidos com o esquema de segurança proposto estão sendo absorvidos em dois projetos de pesquisa; um no programa Conteúdos Digitais, financiado pelo CNPq (processo nº 552175/2001-3) , e outro no projeto Instituto Fábrica do Milênio (IFM), financiado pelo Ministério de Ciência e Tecnologia. Mais informações sobre estes projetos e seus objetivos podem ser obtidas nas páginas http://www.ifm.org.br e http://www.das.ufsc.br/seguranca, respectivamente.

que não são atribuídos a este; a **negação de serviço**, quando agentes lançam ataques para consumir uma quantidade excessiva de recursos computacionais de uma plataforma de agentes (p.ex. espaço em disco, CPU e portas de rede); e o **acesso não-autorizado**, por exemplo, quando um agente obtém acesso de escrita e leitura dos dados nos quais este não tem autorização.

Como já comentado, a abordagem de estabelecer domínios isolados para os agentes é a técnica mais comum para proteger os recursos e funcionalidades das plataformas de agentes contra agentes maliciosos. As entradas ou mesmo trocas entre domínios são mediados por um monitor de referência. Além desta abordagem, outras técnicas foram propostas tomando como base técnicas de segurança convencionais. Algumas destas técnicas são citadas a seguir:

- **Interpretação de Código Seguro**. A idéia chave é que os comandos considerados nocivos em um agente devam ser executados em um local seguro ou podem ser negados ao agente. Como exemplos, pode-se citar o Interpretador Seguro do Java (Sun 2002) e o Safe Tcl (Levy 1998).
- **Assinatura Digital**. Esta técnica serve como um meio para confirmar a autenticidade de um objeto, de sua origem e de sua integridade. Tipicamente, o assinante do código é o criador do agente. Como um agente opera em favor de uma organização ou usuário final, sistemas de agentes móveis comumente usam a assinatura do usuário como uma indicação da autoridade e nome de quem o agente opera.
- **Histórico do Caminho** - *Path Histories* (Ordille 1996). A idéia básica desta técnica é manter um registro autenticável das plataformas visitadas, a priori, por um agente, para que a próxima plataforma visitada possa determinar se processará o agente e quais as restrições de recursos serão aplicadas.
- ***Proof-Carrying Code*** ou **PCC** (Necula e Lee 1998). Esta técnica segue a abordagem baseada em verificadores de código e associa pedaços de código do agente a provas formais que garantem a corretude do código e que são verificadas antes da execução do código. Nesta abordagem o produtor do código (p.ex.: o autor de um agente) deve fornecer provas de que o programa possui propriedades de corretude (*safety*), previamente acordadas com os consumidores do código .

Os mecanismos aqui citados oferecem, para algumas classes de aplicações, uma segurança efetiva para as plataformas de agentes e seus recursos, em especial, quando as técnicas podem ser combinadas. A técnica de criação de domínios isolados, combinada com a assinatura de código - como suportada pela plataforma Java 2 - possibilita a implantação de um controle de acesso[2], em tempo de execução, baseado na autoridade de um agente. Porém, estas técnicas citadas acima precisam ainda ser aprimoradas, visando garantir um controle de acesso mais adequado para aplicações em sistemas de larga escala como a Internet. Além disso, o controle de acesso baseado somente na autoridade do agente não se mostra adequado quando agentes *multi-hop* com itinerários livres são considerados já que a confiança em um agente não depende só da verificação de autenticidade do proprietário do agente, mas também da confiança nas plataformas visitadas anteriormente pelo mesmo.

Em outras palavras, um esquema de autenticação e de autorização efetivo para um sistema de larga escala deve estar baseado nos atributos de privilégios (credenciais do

[2] Limitações do modelo de controle de acesso do Java 2 estão descritas na seção 7.

agente), na autenticidade do principal[3] responsável pelo agente e na autenticidade das plataformas visitadas pelo mesmo. Mecanismos flexíveis de autenticação e de autorização que atendam aos requisitos de escalabilidade inerentes a ambientes como a Internet não são evidentes na literatura.

3. Infra-estrutura SPKI/SDSI

Motivadas pela complexidade do esquema global de nomeação usado no X.509, duas iniciativas relacionadas, porém independentes, surgiram: o SPKI (*Simple Public Key Infrastructure*), padrão IETF (1999), proposto por Carl Ellison e outros membros, e o SDSI (*Simple Distributed Security Infrastructure*), projetado no MIT por Ronald Rivest e Butler Lampson (1996). Como iniciativas complementares, em 1998, as mesmas foram integradas em uma só especificação surgindo o SPKI/SDSI, que é citado por alguns autores como apenas SPKI.

O SPKI/SDSI define um modelo igualitário de confiança. Os sujeitos (ou principais) são chaves públicas e cada chave pública é uma entidade certificadora (Clarke 2001). Não há infra-estrutura global hierárquica como no X.509. Dois tipos de certificados são definidos no SPKI/SDSI: os certificados de nomes e os de autorização. Um certificado de nomes define um nome local no espaço de nomes do emissor e liga esse nome a uma chave pública ou, ainda, a outro nome. O emissor do certificado é sempre identificado por sua chave pública.

O certificado de autorização concede autorizações específicas do emissor do certificado para o sujeito do certificado e essas autorizações são ligadas a um nome, a um grupo de nomes ou a uma chave. O emissor de um certificado pode ainda permitir que o principal delegue as permissões recebidas a outros principais. A delegação permite uma distribuição controlada de autorizações. Um certificado de autorização SPKI deve possuir os seguintes campos: a chave do emissor; o sujeito do certificado[4]; o bit de delegação (*boolean*); uma *tag* que especifica uma autorização e uma validade para o certificado.

O modelo do SPKI/SDSI possibilita a construção de cadeias de confiança que partem do administrador de um serviço terminando em chaves de principais. Quando um determinado sujeito ou chave pública deseja obter acesso a algum recurso, este deve apresentar uma requisição assinada e uma cadeia de certificados que permita a checagem das autorizações concedidas (Clarke 2001). Um dos principais objetivos da checagem de cadeias de certificados é assegurar que mesmo que o emissor no topo da cadeia conceda o direito de delegações subseqüentes nenhum sujeito pode obter autorizações maiores que as do emissor de origem.

O SPKI/SDSI tem avançado bastante no sentido de se firmar como uma plataforma de suporte à autenticação e à autorização em sistemas distribuídos. O SPKI segue uma abordagem totalmente descentralizada para o controle da autenticação e da autorização. Esta abordagem adotada pelo SPKI/SDSI se mostra mais adequada para a implementação destes controles de segurança em sistemas distribuídos de larga escala. A dificuldade deste modelo está em identificar, entre os certificados de um cliente, caminhos de uma cadeia de confiança que levem ao servidor desejado. Em alguns casos, um cliente e um servidor podem nem mesmo estar conectados por uma cadeia de confiança. Este problema é de

[3] Entidades autorizadas pela política de segurança a efetuar acessos sobre informações mantidas pelo sistema.
[4] O sujeito do certificado é a chave de um principal ou o grupo que irá receber a autorização

difícil solução, porém diversos trabalhos se dedicam à determinação de cadeias de certificados que viabilizem acessos (Santin 2002).

4. Abordagem Proposta para Autenticação e Autorização Baseada em Redes de Confiança

Os mecanismos de segurança aqui propostos estão fundamentados em um modelo de agentes que tem como premissa itinerários livres e múltiplos saltos (*multi-hop*). A especificação MAF (*Mobile Agent Facility*) (OMG 2000) serve de guia neste trabalho, visando conseguir a interoperabilidade entre sistemas de agentes.

O esquema aqui proposto, visando possibilitar a sua implantação em aplicações distribuídas em sistemas de larga escala, como a Internet tem um controle da autenticação e da autorização totalmente descentralizado. Para isto, este esquema faz uso das redes de confiança SPKI/SDSI.

4.1- Federações de Plataformas

Neste esquema, as plataformas de agentes móveis podem se agrupar de acordo com suas classes de serviços, constituindo federações de serviços e definindo relações de confiança entre si. Por exemplo, a Federação de Empresas de Moldes agrupa as plataformas de agentes móveis das empresas que fabricam moldes e matrizes. A constituição de Federações de Serviços SPKI, introduzida em (Santin 2002), tem por objetivo facilitar o acesso de clientes a serviços, através do estabelecimento dinâmico de redes de confiança. Estas redes de confiança entre cliente e servidor são estabelecidas, de forma rápida e eficiente, a partir de certificados SPKI de nomes e de autorização disponíveis nos repositórios de certificados das federações de serviços. Nesta forma de organização, um membro de uma federação pode participar de outras federações e diferentes federações podem ter vínculos entre si (relações de confiança), formando assim teias de federações de alcance global. A principal função da teia de federação é ajudar um cliente, através de seus agentes na busca de privilégios de acesso que o liguem ao guarda de um serviço (outra plataforma).

As federações devem fornecer repositórios de certificados e suporte de busca de cadeias de certificado. Através da colocação dos certificados de nomes e de autorização nestes repositórios, os serviços disponíveis nas plataformas associadas podem ser divulgados. A inclusão de uma plataforma em uma destas federações deverá ser negociada com a associação que controla este serviço de disponibilidade de certificados. Maiores detalhes sobre o conceito de federações podem ser obtidos em (Santin 2002).

4.2- Esquema de Autenticação e Autorização

Durante o processo de criação de um agente móvel, o proprietário, na qualidade da autoridade que o agente representa, emite um certificado de nome SPKI/SDSI para o agente, associando o mesmo à sua chave pública, e, repassa um conjunto de certificados de autorização SPKI/SDSI definindo atributos de privilégio para o agente (credenciais). Listas, indicando as federações de serviço cujas plataformas membros estão autorizadas para executar o agente, podem ainda ser definidas e anexadas ao agente. Por fim, o proprietário do agente deve assinar o código do agente e os dados definidos pelo programador como somente-leitura, para então criá-lo em sua plataforma de origem e poder despachá-lo pela rede.

A Fig. 1 ilustra os procedimentos definidos no esquema de segurança, compostos de técnicas de prevenção e de detecção, que enfatizam principalmente a proteção das plataformas de agentes e de seus recursos.

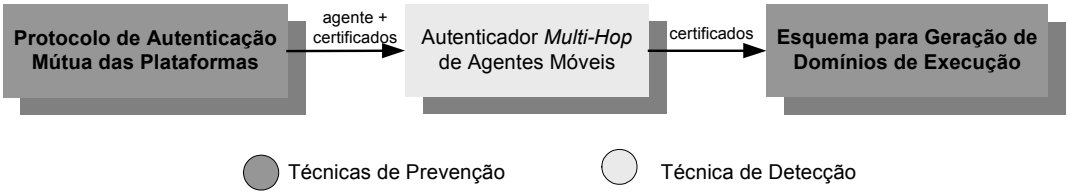

Fig. 1 – Esquema de Segurança para Proteção de Plataformas de Agentes

Neste esquema, primeiramente, as plataformas fonte (emissora de agentes) e destino (receptora de agentes) devem passar por um protocolo de autenticação mútua para que um canal seguro seja estabelecido para o envio de agentes. Em seguida, o agente será enviado com as suas credenciais para ser autenticado pela plataforma destino e então o seu domínio de execução poderá ser criado. Ou seja, quando um agente chega a uma plataforma, este apresenta a sua chave pública e um ou mais certificados SPKI/SDSI (cadeia de certificados) para que o verificador da plataforma realize a checagem dos certificados. A partir destas informações, este verificador deverá gerar as permissões necessárias ao agente para sua execução na plataforma destino. A particularidade da geração dos conjuntos de permissões necessários para a execução de agentes na plataforma destino torna o esquema flexível e dentro da ótica do "princípio de menos privilégio" (Saltzer 1975). As redes de confiança contribuem ainda com a escalabilidade necessária em aplicações na Internet.

A seguir, serão analisados alguns aspectos referentes aos mecanismos que constam no esquema proposto.

4.2.1- Autenticação Mútua

Segundo a especificação MAF (OMG 2000), as plataformas de agentes, antes de receberem agentes ou de interagirem com outros agentes ou outras plataformas, devem aplicar mecanismos de autenticação e de autorização para conceder ou restringir acessos a seus recursos. No esquema proposto, antes que uma transferência de agentes ocorra, a autenticação mútua entre as plataformas envolvidas deve ser estabelecida, tendo como resultado a criação, na infra-estrutura de comunicação, de um canal seguro entre as plataformas autenticadas para o envio de agentes. A autenticação mútua das plataformas também deve ser realizada quando uma plataforma solicita uma criação remota de um agente ou quando esta deseja invocar remotamente métodos de um agente.

De acordo com o modelo SPKI/SDSI, a identificação não é feita através de nomes, mas através de chaves públicas, e o mecanismo de autenticação é a assinatura digital. Assim, na autenticação de plataformas, para que uma assinatura digital seja verificada, a chave pública e o elemento de verificação da assinatura digital (por exemplo, uma requisição assinada digitalmente), devem chegar até o destinatário. No modelo igualitário do SPKI/SDSI, as chaves públicas para verificação de assinaturas são encontradas em cadeias de autorização apresentadas pelas plataformas fonte às plataformas destino. No trabalho proposto, a plataforma fonte monta um elemento de verificação a partir de uma requisição da operação “estabelecimento de canal seguro”.

A Fig. 2 ilustra a autenticação mútua realizada com o uso de um protocolo *Challenge/Response,* baseado em certificados SPKI/SDSI dos proprietários (ou

administradores) das plataformas envolvidas. A autenticação é verificada no SPKI/SDSI, sempre tomando como base cadeias de certificados de autorização (Elisson 1999).

No primeiro passo da Fig. 2, a plataforma fonte envia uma requisição (*establish_trust*) mais um *nonce*[5] (*noncePF*), assinados, sem qualquer certificado SPKI/SDSI, para o estabelecimento de um canal seguro. De posse da requisição, a plataforma destino monta um desafio (*challenge*) para que a plataforma fonte prove a posse das permissões para a operação e o manda assinado para a plataforma fonte (passo 2). O desafio é composto com informações da ACL (*access control list*) do recurso, pelo *noncePF* e por um *nonce* gerado pela plataforma destino (*noncePD*). No passo 3, a plataforma fonte verifica a assinatura do *challenge* para confirmar a autenticidade da plataforma destino[6]. Na seqüência, a plataforma fonte envia uma resposta (*response*) contendo a requisição e o *noncePD* assinados mais os certificados de autorização para a operação desejada. A partir da cadeia de certificados, a plataforma destino pode verificar a assinatura do emissor, terminando o processo de autenticação (passo 4).

Os suportes necessários para a implementação deste protocolo são providos pelo objeto *SPKI/SDSIResolver* disponíveis em todas as plataformas. E é ainda através deste objeto, que a infra-estrutura SPKI/SDSI se torna disponível em cada plataforma.

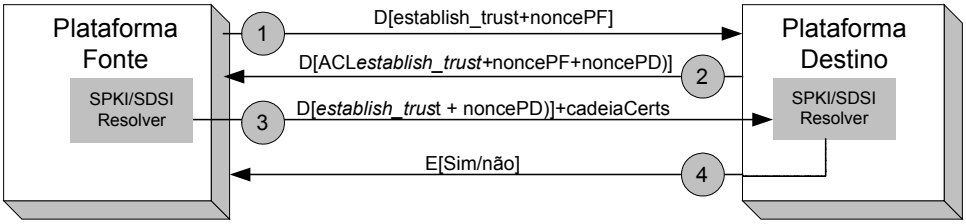

Fig. 2- Protocolo de Autenticação Mútua das Plataformas

É importante frisar que o processo de autenticação mútua das plataformas termina com o estabelecimento de um canal seguro que permanecerá válido nas interações posteriores, permitindo o envio de agentes entre estas plataformas de forma segura. Ou seja, um canal seguro é criado para ser usado na emissão de agentes sem a necessidade da repetição dos passos da Fig. 2, a cada envio de agentes entre as plataformas.

No estabelecimento do canal seguro, uma tecnologia de segurança subjacente (o SSL) é usada, onde chaves de sessão são negociadas e geradas, visando garantir a confidencialidade e a integridade das comunicações entre as plataformas de agentes. Quando um canal seguro já estiver estabelecido, a plataforma fonte envia o agente para a plataforma destino juntamente com os seus certificados de autorização SPKI/SDSI.

4.2.2- Autenticação de Agentes Móveis

Após a autenticação mútua das plataformas, mas antes de instanciar uma *thread* para um agente, a plataforma destino deve então executar o processo de autenticação do agente recebido. Para autenticação de agentes móveis, o mecanismo de assinatura digital não pode ser usado devido à limitação que impede os agentes de carregarem consigo suas chaves privadas. Neste caso, a especificação MAF sugere o uso de algoritmos, chamados autenticadores, para a verificação da autenticidade dos agentes. Porém, quando aplicações envolvendo agentes *multi-hop* são consideradas, esta especificação é omissa. Como

[5] O objetivo deste *nonce* é proteger contra ataques de mensagem antiga.
[6] A plataforma fonte obtém a chave publica a partir do certificado de nome da plataforma destino que esta possui. Caso não tenha este certificado, esta pode ir busca-lo nas federações SPKI/SDSI.

contribuição deste trabalho, propõe-se a criação de um autenticador *multi-hop* que, baseado na autenticidade do proprietário do agente, na autenticidade das plataformas que o agente visitou e ainda nas listas de federações definidas pelo proprietário do agente, estabeleça a confiança no agente.

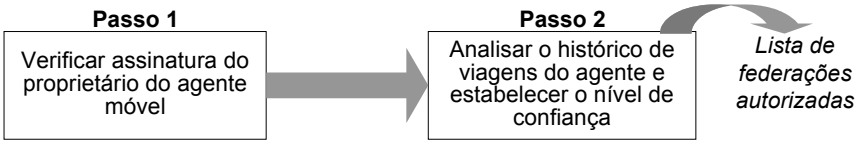

Fig. 3- Autenticador de agentes *multi-hop*

No autenticador ilustrado na Fig. 3, uma plataforma, ao receber um agente móvel, deve, primeiramente, através da verificação da assinatura do agente (do código e dos dados somente-leitura), comprovar que este agente permanece íntegro e confirmar a sua associação a um principal, seu proprietário (**passo 1**). Desta forma, modificações feitas por plataformas maliciosas podem ser facilmente detectadas por qualquer plataforma visitada pelo agente. Quando uma plataforma detectar uma modificação, esta deve avisar à plataforma *de origem* para que providências sejam tomadas.

Para agentes *one-hop*, a técnica proposta no passo 1 garante a integridade dos agentes, porém, para agentes *multi-hop* esta técnica não é suficiente. Visando detectar possíveis modificações e conferir o percurso de viagens do agente, a plataforma de agentes destino deve ainda analisar o histórico do caminho do agente (**passo 2**). Para isto, cada plataforma visitada deve adicionar em um objeto (histórico das plataformas visitadas) uma assinatura indicando sua identidade e a identidade da próxima plataforma a ser visitada, formando a história do caminho percorrido pelo agente. Além disso, baseado em Karnik (1998), propõe-se ainda que os dados sensíveis, gerados em uma plataforma, sejam armazenados em um "contêiner", que será carregado pelo agente, e sejam assinados pela plataforma geradora dos mesmos para que possíveis modificações destes dados possam ser detectadas. Vale ressaltar que o uso do contêiner pode impor um custo ao desempenho das aplicações e que esta degradação deve ser considerada quando o conjunto de mecanismos a ser empregado na proteção da plataforma for selecionado.

As plataformas visitadas deverão estar associadas às federações autorizadas do agente que estão definidas na lista que o agente carrega consigo. Se for desejado, é possível verificar a autenticidade das assinaturas das mesmas em cada entrada do histórico e a integridade do contêiner. Um inconveniente é que a verificação do caminho e do contêiner de dados sensíveis tem um custo que cresce à medida que o histórico do caminho aumenta. A partir dessa análise, é possível avaliar o nível de confiança nas plataformas visitadas pelo agente e o nível de risco em aceitar o agente. Estas técnicas permitem a detecção de modificações no agente.

4.2.3- Procedimentos para a Geração dos Domínios de Proteção

Após estabelecida a confiança no agente (resultado dos procedimentos anteriores) e com base nos certificados de autorização SPKI/SDSI do agente e nos níveis de confiança e de risco determinados a partir da lista de federações autorizadas, são definidos os domínios de proteção em que estes serão executados e quais as permissões serão atribuídas a estes domínios (ver Figura 1). As cadeias de autorizações que o agente carrega consigo e que representam suas credenciais (ou atributos de privilégios) necessitam ser verificadas pelo guarda do serviço para que, em seguida, o conjunto de permissões seja definido e os domínios de proteção do agente sejam gerados. Neste esquema, teve-se a preocupação de desacoplar os atributos de privilégios concedidos a principais (credenciais do agente) dos

atributos requeridos ou necessários (atributos de controle ou políticas) para acessar os recursos protegidos da plataforma, proporcionando, assim, um controle de acesso mais flexível, dinâmico e adequado para sistemas de larga escala[7].

É importante ressaltar que as cadeias de autorização iniciais que o agente carrega consigo podem não ser suficientes para garantir ao agente o acesso a certos recursos de uma plataforma. Novos certificados podem ser concedidos ao agente durante suas visitas às plataformas de agentes. Por exemplo, supondo que um agente precisa visitar a empresa transportadora de produtos conveniada a uma empresa de moldes. O agente pode não ter os certificados necessários para ser recebido nesta plataforma. A plataforma que representa a empresa de moldes pode então delegar certificados a este agente para que o mesmo possa acessar a transportadora conveniada. O modelo proposto no SPKI/SDSI determina que o cliente deve ser o responsável por encontrar os certificados que o ligue a um serviço. Logo, o próprio agente pode ir atrás desses certificados nas teias de federações e negociá-los quando estes forem encontrados[8].

5. Aspectos de Implementação do Esquema Proposto

Um protótipo, envolvendo o esquema de segurança para proteção das plataformas de agentes, foi definido e está sendo implementado visando comprovar a viabilidade de sua utilização em aplicações distribuídas que embutem a noção de mobilidade de código, e, por conseguinte, validar o esquema aqui proposto.

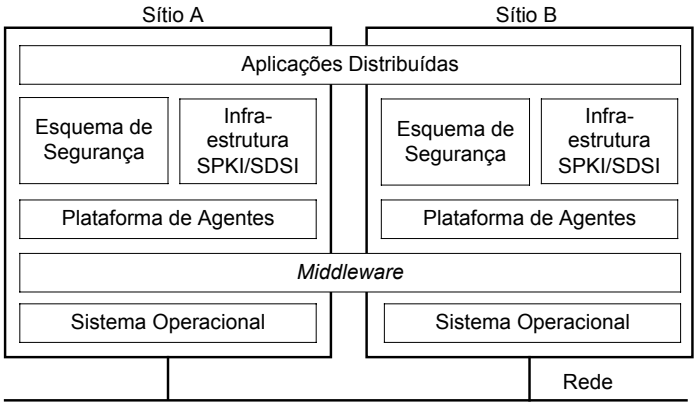

Fig. 4- Arquitetura do Protótipo

As escolhas tecnológicas necessárias para compor a arquitetura do protótipo, ilustrada na Fig. 4, têm como base o uso de padrões abertos e componentes de pratelerias (*Commercial Off-the-Shelf* - COTS) para que os requisitos de portabilidade, escalabilidade e interoperabilidade sejam preservados. Outros padrões no contexto da Internet também estão sendo considerados.

Plataforma de Agentes

Para compor a camada do ambiente computacional que suporta os agentes móveis, foi adotada a plataforma *Aglets* da IBM (1996), primeiramente por usar o Java como linguagem de código móvel e também por ser uma iniciativa de código aberto. A linguagem Java, além de ser considerada um padrão de fato para a programação de aplicações distribuídas, possui diversas propriedades que a tornam uma boa linguagem para programação de agentes móveis, tais como: portabilidade, interpretação segura de código, programação *multithread*, serialização de objetos, reflexão estrutural, suporte para programação distribuída, carregamento dinâmico de código e assinatura de código. Embora a linguagem Java seja

[7] Mais detalhes desse procedimento estão descritos em (Wangham et al 2002).
[8] Em (Nikander and Partanen 1999), os autores propõem que a busca para formação de novas cadeias seja de responsabilidade do servidor. Porém, como já mencionado, isto foge do modelo SPKI/SDSI.

muito conveniente para criação de agentes móveis, suas limitações e desvantagens precisam e estão sendo levadas em conta.

O Kit de Desenvolvimento de Software *Aglets* (ASDK) fornece uma interface de programação orientada a objetos, mecanismos para mobilidade de código, de dados e de informações de estado e um ambiente computacional padrão, chamado *Tahiti*, que suporta criação, clonagem, execução e envio de agentes. A plataforma *Aglets*, apesar de seus autores terem participado ativamente da definição da especificação MAF, não implementa os serviços dessa especificação. A conformidade do *Aglets* com o padrão MAF está sendo tratada pelo grupo de pesquisa dos autores, porém não será abordada neste artigo. Esta conformidade possibilita que esta plataforma use como infra-estrutura de comunicação, não só o ATP (*Agent Transfer Protocol*) e o Java RMI, mas também o CORBA.

Infra-estrutura SPKI/SDSI

A infra-estrutura SPKI/SDSI, integrante da arquitetura do protótipo, é responsável pela criação dos certificados SPKI/SDSI das plataformas de agentes e dos agentes móveis. Os certificados dos agentes móveis serão emitidos pelos proprietários dos agentes durante o seu processo de criação (ver seção 4). Uma biblioteca de códigos Java que implementa o SDSI 2.0 (Morcos 1998) e que provê essas funcionalidades está sendo utilizada para esta finalidade. Novos objetos estão sendo implementados para que a infra-estrutura SPKI/SDSI se torne mais flexível, eficiente e disponível para as plataformas de agentes.

A biblioteca SDSI 2.0 também é usada na implementação das federações SPKI/SDSI. Os repositórios de certificados são implementados com auxílio da base de dados *Apache Xindice*[9] com XML nativo. Os certificados armazenados estarão no formato XML visando facilitar a busca para formação de novas cadeias.

Middleware

O padrão CORBA (*Common Object Request Broker Architecture*) (OMG 2001) foi escolhido para compor a camada de *middleware*. Esta escolha se deu em razão da grande aceitação desta especificação, que também é um padrão ISO, e por esta introduzir uma arquitetura distribuída e orientada a objetos, que segue conceitos de sistemas abertos e define padrões para ambientes heterogêneos, permitindo a construção de aplicações independentes de linguagem, de sistema operacional, de *hardware* e de tecnologia de rede. O padrão CORBA incorpora ainda os protocolos Internet para comunicação (pilha TCP/IP).

5.1- Esquema de Segurança: técnicas de prevenção e de detecção

Como a plataforma de agentes escolhida para compor o protótipo está baseada na linguagem Java, a interpretação segura do código dos agentes e a definição dos domínios de proteção para os agentes móveis são garantidas, em parte, pelo modelo de segurança do Java 2, através dos seguintes mecanismos: verificadores de *bytecode*, carregadores dinâmicos de códigos, gerenciadores de segurança (classe *SecurityManager*), controladores de acesso (classe AccessController) e das APIs JCA (*Java Crytography Architecture*) e JCE (*Java Cryptographic Extension*).

A implementação do esquema de segurança proposto, conforme definido na Fig. 1, será analisada a seguir.

[9] http://xml.apache.org/xindice/

Autenticação Mútua

O protocolo para autenticação mútua (ver Fig. 2) foi implementado com o auxílio da biblioteca SDSI 2.0 e das ferramentas criptográficas do Java 2. Devido a limitações da plataforma *Aglets,* esse protocolo foi implementado em um agente móvel (*Authenticator Agent*) que é acionado sempre que uma plataforma de agentes tenta se comunicar com outra plataforma. Se um canal seguro já está estabelecido, a comunicação ocorre normalmente. Senão, o protocolo se inicia para a estabelecer a autenticação mútua entre as plataformas através de troca de mensagens entre *Authenticator agents* das plataformas.

Para a geração das chaves de sessão, necessárias para garantir integridade e confidencialidade do canal de comunicação, utilizou-se, como tecnologia de segurança subjacente, o protocolo SSL, conforme implementado pela ferramenta iSaSiLk. Esta interação com o SSL foi implementada e integrada à plataforma *Aglets*. Os certificados SSL necessários foram gerados a partir dos próprios certificados SPKI que as plataformas de agentes móveis já possuem.

Autenticação de Agentes Móveis

O autenticador *multi-hop*, conforme descrito na seção 4.2.2, está sendo implementado com o auxílio das ferramentas criptográficas do Java 2 e da biblioteca SDSI 2.0. Todas as entradas do histórico das plataformas visitadas estão sendo verificadas, porém, somente dois níveis de confiança foram definidos de acordo com a lista de federações: plataforma autorizada ou não autorizada, ou seja, a plataforma é membro ou não da federação presente na lista de federações previamente definidas para a missão do agente. O algoritmo *multi-hop* definido até este momento está representado na Fig. 5.

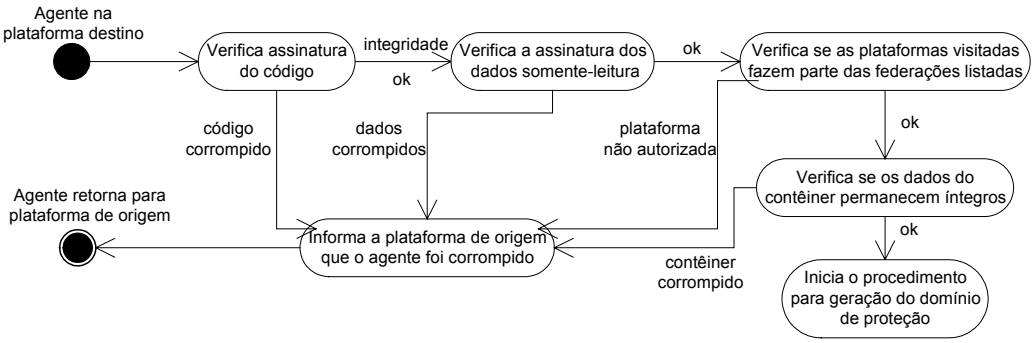

Fig. 5- Autenticador *Multi-hop*

Geração de Domínios de Proteção

Esta etapa, que tem como resultado a definição dos domínios de proteção, foi implementada em parte, porém todas as definições já foram realizadas. O processo para geração do conjunto de permissões foi definido visando contornar as limitações com relação à abordagem centralizada e estática do esquema de controle de acesso do Java 2. Os domínios de proteção dos agentes são definidos a partir de certificados de autorização SPKI/SDSI que formam suas credenciais (atributos de privilégios) e que são carregados pelos mesmos.

As extensões necessárias no modelo de segurança do Java 2 para a geração do domínio de proteção onde um agente irá se executar (domínio de aplicação), baseado nos certificados SPKI/SDSI do agente, representados nos objetos em cinza da Fig. 6, são: *SPKISecureClassLoader,* necessário para que os certificados possam ser extraídos do agente e para que o domínio de proteção para a *thread* seja criado; *SPKIPolicy,* objeto que representa a política SPKI e que define, a partir dos certificados que o agente carrega

consigo, quais permissões Java serão associadas ao domínio de aplicação; e o *SPKIVerifier*, necessário para a checagem dos certificados SPKI.

Seguindo a dinâmica da Figura 6, o administrador da plataforma descreve a política de segurança da plataforma de agentes mapeando a autorização concedida a partir dos certificados SPKI/SDSI em permissões Java, definindo para isto um arquivo de configuração de política (*policy file*). Quando um agente é recebido na plataforma, suas credenciais (cadeia de certificados) são repassadas pelo *SPKISecureClassLoader* para o objeto *SPKIPolicy* que as interpreta. Quando as permissões SPKI são mapeadas em permissões Java, o suporte Java gera o domínio de proteção correspondente para a execução da *thread* que se ocupará do agente. As permissões do Java ficam disponíveis no objeto *PermisssionCollection*.

Se a *thread* (agente) faz uma solicitação de acesso a um recurso fora do seu domínio de aplicação, ou seja, em um domínio de sistema, o controlador de acesso (classe *AccessController*) é acionado para verificar se o acesso deve ser permitido. Este controlador de acesso deve verificar no domínio de proteção criado se o solicitante tem o objeto *Permission* correspondente na sua coleção de permissões. Em caso afirmativo, a *thread* pode trocar de domínio, entrando no domínio de sistema.

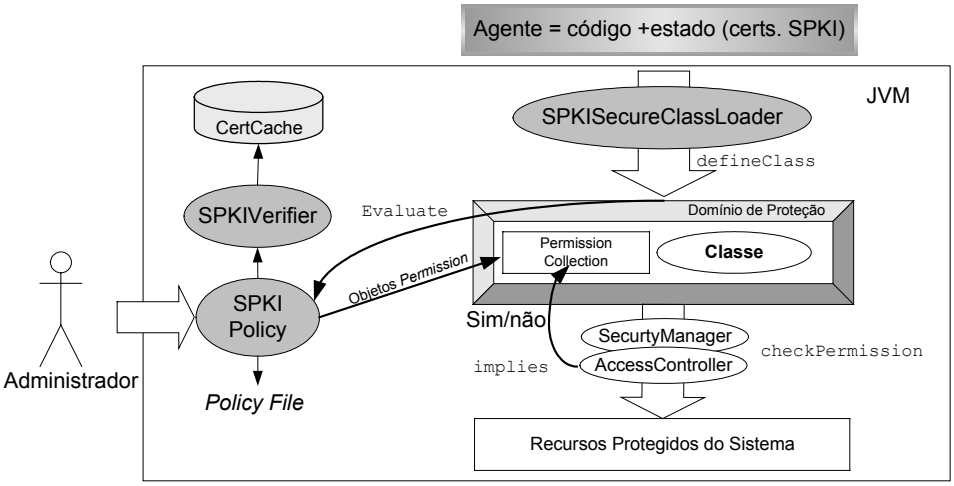

Fig. 6- Dinâmica para Geração do Domínio de Proteção

6- Integração do Protótipo a uma Aplicação Distribuída

O protótipo apresentado na seção 5 está sendo utilizado em uma aplicação de busca e seleção de parceiros na formação de empresas virtuais baseada em agentes móveis (EVs), como parte da contribuição do Departamento de Automação e Sistemas ao projeto IFM. Estas empresas virtuais correspondem a uniões temporárias de empresas participantes de uma organização virtual (OV) que se agregam visando responder a oportunidades de negócios. No contexto da definição das possíveis combinações de empresas, os agentes móveis possuem inúmeras características que trazem a eficiência desejada para formação dinâmica de EVs.

Os agentes móveis terão como função deslocar-se pelas empresas em busca de informações para a formação de Evs, conforme ilustrado na Fig. 7. Esta figura ilustra um cenário, para a seleção das possíveis empresas virtuais, usando uma arquitetura híbrida de agentes. Depois de obter as informações necessárias, o agente móvel deve retornar para o

broker trazendo consigo as alternativas de empresas virtuais, devidamente escalonadas, a partir de indicadores de desempenho previamente definidos para avaliá-las.

O esquema de segurança proposto está sendo integrado à aplicação descrita na Fig. 7, possibilitando assim analisar a sua viabilidade para aplicações na Internet[10].

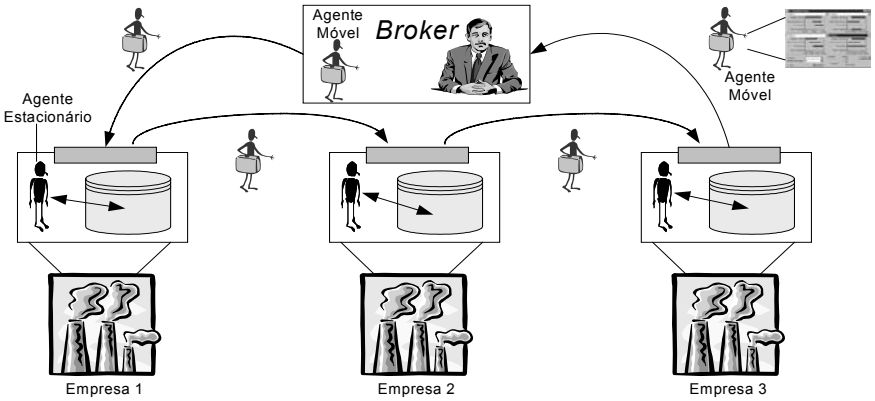

Fig. 7- Cenário para a Seleção de Empresas Virtuais

7. Trabalhos Relacionados

A maioria das plataformas de agentes móveis, baseadas na linguagem Java, utiliza o modelo de segurança do próprio Java, em especial o do Java 2, para implementar parte de seus mecanismos de segurança. Entre estas, podem ser citadas as plataformas comerciais *Aglets* da IBM, o *Grasshopper* da GMDFokus e as plataformas acadêmicas *SOMA* e *Ajanta*, desenvolvidas na Universidade de Bologna e na Universidade de Minesota, respectivamente.

Estas plataformas de agentes móveis estendem o gerenciador de segurança do Java (*Security Manager*) para fornecer uma solução mais flexível e adequada para as plataformas de agentes e para implementar domínios de proteção que isolam os agentes móveis prevenindo que ataques maliciosos dos mesmos ocorram. Em algumas plataformas, a diferença entre os esquemas de autorização está na informação usada para determinar o conjunto de direitos de acesso de um agente. As plataformas *Aglets* e *Ajanta* usam somente a identidade do proprietário do agente. A plataforma *Grasshopper* faz uso de políticas de controle de acesso baseadas na identidade do proprietário ou no nome do seu grupo (*group membership*). Na plataforma SOMA, os principais são associados com papéis (*roles*) que identificam as operações permitidas sobre os recursos do sistema. Todos esses casos se mostram inadequados para sistemas de larga escala.

Além disso, é importante destacar ainda que o modelo de controle de acesso do Java 2 possui algumas limitações que precisam ser analisadas. Ao invés de seguir a natureza distribuída do seu modelo de execução, o modelo de segurança do Java 2 está fundamentado em um esquema de autorização centralizado. Essa centralização refere-se ao fato de que todo o controle de acesso é executado a partir de um arquivo único de configuração que define toda a política de segurança de uma máquina. Portanto, tem-se uma única ACL relacionando todos os sujeitos e objetos do sistema. Durante a sua execução, cada código é rotulado como pertencente a um ou mais domínios de proteção. Cada

10 A integração do esquema de segurança proposto à aplicação descrita está detalhada no artigo intitulado "*Improving the Creation of Virtual Enterprises through Secure Mobile Agents*" (submetido para publicação).

domínio possui um conjunto de permissões associadas a partir de um arquivo de configuração de política. Este arquivo define portanto um mapeamento estático entre cada componente móvel e as permissões concedidas ao mesmo para a sua execução em um ambiente local. Além de numerosas dificuldades práticas em termos de programação, as limitações resultantes da confiança das decisões de controle de acesso sobre o arquivo de configuração local impedem o desenvolvimento de um ambiente distribuído e dinâmico, exigindo uma definição estática de todos os componentes distribuídos e seus atributos de segurança de uma só vez.

Para se implantar um esquema de autorização, em sistemas de agentes móveis efetivo, a autenticação dos agentes se torna imprescindível. As plataformas *Aglets* e *Grasshopper* não possuem mecanismos para autenticação de agentes móveis. Na plataforma SOMA, os agentes são autenticados com base em uma série de informações contidas em suas credenciais, são elas: os nomes do domínio e do lugar de origem; o nome da classe que implementa o agente e o nome do usuário responsável pelo agente. Antes da migração, estas informações, o estado inicial do agente e o código do agente são digitalmente assinados pelo usuário que criou o agente. Quando o agente chega em um sítio remoto, suas credenciais são verificadas em relação à sua autenticidade (assinatura do proprietário do agente). Já a plataforma *Ajanta* usa como protocolo para autenticação o *Challenge*/*Response* com geração aleatória de *nonces* para prevenir ataques de mensagem antiga, baseado na assinatura do proprietário do agente.

Em comparação com o modelo estático do Java 2 e com as plataformas de agentes móveis citadas acima, o esquema proposto neste trabalho, apesar de usar algumas funcionalidades do modelo de segurança Java, ao fazer uso de certificados de autorização SPKI, desacopla os atributos de privilégios (credenciais) dos atributos de controle (políticas). Isto significa que, embora se possa continuar definindo estaticamente a política no arquivo de configuração, o mecanismo proposto ganha em flexibilidade com as possíveis delegações oferecidas pelos certificados SPKI. Ou seja, os domínios são definidos dinamicamente conforme o agente vá agregando certificados delegados às suas credenciais em seus itinerários.

Além disso, no processo de autenticação do agente, conforme descrito na seção 5.2, a informação usada para determinar o conjunto de direitos de acesso de um agente não está baseada somente na identidade do proprietário do agente, mas também em cima das chaves públicas do proprietário e das plataformas visitadas, o que facilita a gestão de nomes em sistemas de larga escala.

Visando aprimorar o controle de acesso do Java 2, dois trabalhos relacionados propõem a utilização de certificados SPKI/SDSI. O primeiro trabalho proposto por Nikander e Partanen (1999) usa certificados de autorização SPKI para delegar permissões Java que descrevem diretamente as possíveis permissões associadas a um domínio de proteção. Nesta proposta, a *tag* de autorização do certificado SPKI foi estendida para expressar as permissões Java. Esta solução é aplicável apenas para *applets* e traz a desvantagem de usar certificados SPKI modificados. O segundo trabalho (Molva e Roudier 2000) propõe dois aprimoramentos ao modelo de controle de acesso do Java 2: associação de informação de controle de acesso com cada código móvel (*applet*) na forma de atributos e introdução de elementos intermediários no esquema de controle de acesso para auxiliar a configuração da informação de controle de acesso em ambientes dinâmicos. Nesta proposta, o mecanismo de grupo do SPKI/SDSI, implementado através de certificados de nomes, é quem possibilita esses aprimoramentos. Este trabalho não detalha como implementar esta proposta e como combiná-la com o modelo de segurança atual do Java 2.

Os dois trabalhos comentados acima não estão diretamente voltados para proteção de agentes móveis, mas sim para *applets* Java. Estes não tratam a autenticação mútua entre as plataformas fonte e destino e não analisam o histórico das plataformas visitadas para estabelecer a confiança no código móvel. Somente a primeira proposta possui as características de flexibilidade semelhante ao esquema proposto onde os domínios são formados segundo os certificados delegados ao agente na sua criação e em seu itinerário.

8. Considerações Finais

A aceitação do paradigma de agentes móveis para o desenvolvimento de aplicações distribuídas ainda é prejudicada devido a questões de segurança. As limitações das técnicas apontadas nas seções 2 e 7 comprovam que estas ainda não apresentam resultados satisfatórios que garantam a segurança para as plataformas de agentes móveis. Tais limitações são ainda maiores quando se leva em conta aplicações em sistemas abertos que anseiam por escalabilidade, portabilidade e interoperabilidade, já que estas aplicações envolvem vários domínios administrativos e tecnologias heterogêneas. Nesse contexto, as soluções para nomes e segurança devem ser sempre flexíveis, escaláveis e abertas, permitindo que sistemas de grandes dimensões possam evoluir durante a sua dinâmica.

Foi a constatação dessas limitações e a preocupação com estes aspectos específicos da segurança em aplicações de sistemas distribuídos de larga escala que motivaram o esquema de segurança proposto que visa prevenir ataques de agentes móveis contra plataformas, definindo um procedimento que envolve técnicas de prevenção e detecção: a autenticação mútua das plataformas, a autenticação de agentes móveis *multi-hop* e a geração do conjunto de permissões que será associado aos domínios de proteção criados para o agente. Este esquema está baseado em um controle descentralizado de autorização e autenticação que se mostra adequado para sistemas de larga escala, ao se valer de certificados de autorização SPKI/SDSI. O mecanismo de delegação dos certificados de autorização SPKI possibilita um desacoplamento dos atributos de privilégio do agente dos atributos de controle (políticas), resultando assim em um esquema mais flexível para geração de domínios de proteção, quando comparado com os trabalhos correlatos.

O trabalho descrito neste artigo corresponde a um esforço de dois anos que, apesar de não estar totalmente implementado, já apresenta resultados satisfatórios. Assim que for concluído, testes de desempenho serão realizados visando verificar a eficiência das técnicas empregadas. A integração e adequação deste protótipo a aplicações distribuídas na Internet também estão sendo tratadas visando comprovar a sua aplicabilidade. Do ponto vista da proteção das plataformas de agentes e do canal de comunicação, o esquema de segurança proposto contorna as possíveis ameaças de segurança. Como trabalhos futuros pretende-se definir ainda técnicas de prevenção e de detecção para proteção também dos agentes móveis contra plataformas de agentes.

Referências Bibliográficas

Clarke, D. E. (2001). *SPKI/SDSI HTTP server/certificate chain discovery in SPKI/SDSI*, Master's thesis, Massachusetts Institute of Technology – MIT, USA.

Ellison, C. M. e et al (1999). *SPKI Requirements*, The Internet Engineering Task Force. (http://www.ietf.org/rfc/rfc2693.txt).

IBM (1996). Aglets software development kit. (http://www.trl.ibm.co.jp/aglets).

Jansen, W. e Karygiannis, T. (2000). *NIST Special Publication 800-19*-Mobile agent security, National Institute of Standards and Technology, USA.

Karnik, N. (1998). *Security in Mobile Agent System*, PhD thesis, University of Minnesota, USA.

Lampson, B. e Rivest, R. (1996). A simple distributed security infrastructure. (http://theory.lcs.mit.edu/ cis/sdsi.html).

Levy, J. Y., Ousterhout, J. K. e Welch, B. B. (1998). The Safe-Tcl Security Model, *In:* G. Vigna (ed.), *Mobile Agents and Security*, LNCS 1419, Springer-Verlag, p. 217-234.

Molva, R. e Roudier, Y. (2000). A distributed access control model for java, *European Symposium on Research in Computer Security - ESORICS 2000*, France.

Morcos, A. (1998). *A java implementation of simple distributed security infrastructure*, Master's thesis, Massachusetts Institute of Technology, USA.

Necula, G. e Lee, P. (1998). Safe, untrusted agents using proof-carrying code, *In:* G. Vigna (ed.), *Mobile Agents and Security*, LNCS 1419, Springer-Verlag, p. 61–91.

Nikander, P. e Partnen, J (1999). Distributed Policy Management for JDK1.2, Network and Distributed System Security Symposium, USA.

OMG (2000). Mobile agent facility specification, OMG Document 2000-01-02.

OMG (2001). The common object request broker architecture v2.6, OMG Document 01-12-30.

Ordille, J. J. (1996). When agents roam, who can you trust ?, *First Conference on Emerging Technologies and Applications in Communications*.

Saltzer, J. e Schroeder, M. (1975). The protection of information in computer systems, *IEEE*, V.vol.63, p. 1278–1308.

Santin, A., Fraga, J., Mello, E. e Siqueira, F. (2002) Um Modelo de Autorização e Autenticação baseado em Redes de Confiança para Sistemas Distribuídos de Larga Escala, Simpósio de Segurança em Informática (SSI), São José dos Campos-SP.

Sun (2002). Java 2 sdk. v1.4 security documentation. (http://www.java.sun.com/security/ index.html).

Wangham, M., Fraga e Santin, A. (2002). Usando Certificados SPKI/SDSI para Geração de Domínios de Execução de Agentes Móveis, Simpósio de Segurança em Informática (SSI), São José dos Campos-SP.