DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Relatório Técnico

RT-MAC-2005-06

IMPLEMENTAÇÃO DE UM REPOSITÓRIO SEGURO DE APLICAÇÕES BASEADO EM GSS – PROJETO TAQUARA

JOSÉ DE RIBAMAR BRAGA PINHEIRO JR., ALEXANDRE CÉSAR TAVARES VIDAL AND FABIO KON.

agosto de 2005

Implementação de um Repositório Seguro de Aplicações baseado em GSS Projeto TAQUARA*

Jose de R. B. Pinheiro Júnior
Alexandre César Tavares Vidal
Fabio Kon
Departamento de Ciência da Computação
Instituto de Matemática e Estatística
Universidade de São Paulo

29 de agosto de 2005

Abstract

This work describes the secure implementation of the InteGrade Application Repository under the TAQUARA Project pesperctive. The system supports application signature and user authentication in Grid Computing environments.

Resumo

Este trabalho descreve a implementação do repositório seguro de aplicações do InteGrade dentro do contexto do Projeto TAQUARA. O sistema desenvolvido permite a assinatura de aplicações e autenticação de usuários em ambientes de grade computacional.

1 Introdução

Os computadores e demais recursos que participam de grades computacionais [Berman et al., 2003, Foster and Kesselman, 2003, de Camargo et al., 2004]

*Este trabalho é financiado pelo FUNTTEL/FINEP - Projeto GIGA - Chamada RNP 01/2004.

são particulamente vulneráveis a problemas com a segurança. Os sistemas operacionais e o middleware da grade podem não ser seguros o bastante, permitindo o acesso a recursos indevidos ou códigos maliciosos podem ser executados de forma a se beneficiar de alguma falha no sistema para obter estes recursos[1]. Um problema importante nos sistemas de computação em grade é como garantir que as aplicações submetidas não causem prejuízos aos seus participantes. Em consideração a esses fatores, as aplicações passam a ser um ponto crucial com relação à segurança nas grades. Se uma aplicação sofrer modificações, cada instância sua em um nó da grade, executa o seu código malicioso. Os sistemas em grade existentes possuem táticas diferentes para tratar a segurança de suas aplicações, como veremos logo a seguir.

O InteGrade (`http://gsd.ime.usp.br/integrade`) é um sistema de grades motivado principalmente pela necessidade de aproveitamento de recursos computacionais compartilhados (computadores pessoais, estações de trabalho), em geral subutilizados – com predomínio de períodos ociosos – para execução de aplicações com grande demanda por tais recursos. Em particular, as características do InteGrade incluem suporte para uma gama abrangente de aplicações paralelas e mecanismos que buscam minimizar a percepção, pelos proprietários dos recursos compartilhados, de qualquer possível perda de qualidade de serviço [Goldchleger et al., 2004]. O InteGrade necessita, como os demais sistemas de grade, de uma atenção adicional quanto à segurança dos recursos cedidos à grade.

Uma Grade InteGrade se constitui de aglomerados (*clusters*) de computadores organizados de forma hierárquica e escalável (ver Figura 1). Os nós de um aglomerado do InteGrade podem ser de quatro tipos: Gerenciador do Aglomerado, Dedicado, Compartilhado e de Usuário. O nó gerenciador do aglomerado suporta a coordenação do aglomerado e a comunicação com gerenciadores de outros aglomerados(estas atividades podem ser distribuídas para balanceamento de carga ou replicadas para tolerância a falhas). O nó dedicado é exclusivo para execução de aplicações da Grade. O nó compartilhado disponibiliza tempo ocioso para execução de aplicações dos usuários da grade. E, finalmente, o nó de usuário pertence ao usuário da grade que submete aplicações à grade.

O módulo *Global Resource Manager* (GRM) executa no nó gerenciador do aglomerado. Ele usa as informações sobre o estado dos demais nós, coletadas e enviadas pelos módulos *Local Resource Manager* (LRMs), para escalonamento das aplicações na grade, com base nos requisitos das aplicações e na

[1]Técnicas bastante comuns, como o estouro de buffer [Garfinkel and Spafford, 1996]podem ser utilizadas para obter acesso irrestrito aos recursos das máquinas.

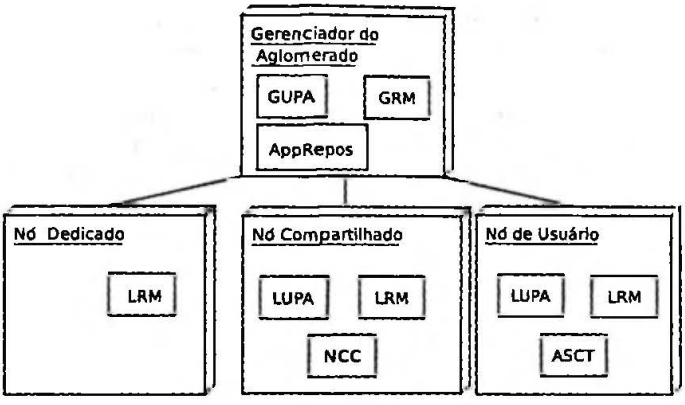

Figura 1: Arquitetura do InteGrade.

disponibilidade de recursos. O LRM também é responsável pela execução das aplicações nos nós da Grade. O módulo *Application Submission and Control Tool* (ASCT) permite aos usuários do InteGrade registrar aplicações armazenadas em um repositório e submetê-las para execução na Grade.

Os módulos *Local Usage Pattern Analyzer* (LUPA) e *Global Usage Pattern Analyzer* (GUPA) trabalham cooperativamente na análise de padrões de uso das máquinas para contribuir para um escalonamento eficiente na grade. O módulo *Node Control Center* (NCC) permite aos proprietários de recursos configurar as condições de compartilhamento, como o percentual de CPU e memória física disponibilizados para a grade, ou definir quando tais recursos podem ser considerados ociosos e, portanto, disponíveis.Este relatório descreve uma implementação básica de segurança para proteção contra usuários maliciosos que desejem alterar as aplicações executadas no InteGrade.

2 Trabalhos Relacionados

O projeto Globus[2] é responsável pelo desenvolvolvimento do *Globus Toolkit* [Foster and Kesselman, 1997], uma ferramenta de software baseada em padrões da indústria que permite a construção de sistemas para grade de modo incremental. Ele é a base da infraestrutura para integrar recursos computacionais geograficamente dispersos.

O Globus disponibiliza um serviço de segurança denominado GSI (*Globus Security Infrastructure*) [Foster et al., 1998] que implementa a autenticação

[2]http://www.globus.org

única, comunicação segura e delegação. Na autenticação única, após uma autenticação inicial, o usuário usa os recursos da grade sem a necessidade de novas autenticações. Para tanto, são criados representantes (*Proxyes*) que realizam as autenticações subseqüentes. Um representante de usuário (*User Proxy*) é um processo que possui credenciais que o permitem representar o usuário por um determinado período de tempo. O representante pode então realizar requisições em nome do usuário neste intervalo. Já o representante de recursos (*Resource Proxy*) é responsável pelos aspectos de segurança de um determinado recurso. Sua função é fazer um mapeamento entre as operações de segurança da grade e as operações de segurança de um determinado domínio administrativo. O GSI utiliza SSL (*Secure Sockets Layer*) para prover comunicação segura entre as aplicações dos usuários. Finalmente, com a delegação é possível repassar parte das permissões de um usuário a uma outra entidade, por exemplo, uma aplicação. Dessa maneira, é permitido a uma aplicação requisitar mais recursos em nome do usuário que solicitou a sua execução.

O GSI implementa uma extensão do GSS (*Generic Security Services*) [Meder et al., 2001], um padrão que define uma API para oferecer ao programador transparência sobre os mecanismos de segurança implementados, possibilitando uma maior portabilidade dos sistemas de segurança. A principal finalidade desta extensão é tratar algumas deficiências presentes no GSS, como exportação, importação e manipulação de credenciais, assim como a delegação fora do estabelecimento do contexto inicial.

O Legion[3] [Grimshaw et al., 1997] é um middleware que integra diversos recursos computacionais para prover aos usuários, de maneira transparente, a visão de um único e poderoso computador; ele é composto por objetos ativos, persistentes e independentes que se comunicam. O Legion emprega uma série de medidas de segurança para garantir diversos objetivos como autenticação, autorização, integridade e confiabilidade de dados [Ferrari et al., 1999]. Um usuário no Legion é representado por um objeto de autenticação que contém a chave secreta criptografada do usuário e informações adicionais. O Legion é flexível quanto ao processo de autenticação, podendo utilizar simplesmente usuário e senha ou um serviço de autenticação como o Kerberos [Kohl and Neuman, 1993]. O processo de autenticação produz uma credencial assinada que pode ser usada a posteriori.

A autorização do Legion é feita determinando quais métodos de um objeto (qualquer entidade dentro do Legion é representada por objeto) podem ser acessados pelos demais. Quando uma chamada de método é recebida, a credencial do chamador, propagada juntamente com o pedido, é verificada

[3]http://www.cs.virginia.edu/ legion

através de listas de controle de acesso, podendo ser negada ou não. Para permitir integridade de dados e confiabilidade, o Legion criptografa as mensagens utlizando-se das credenciais conseguidas durante a autenticação.

O OurGrid é um sistema de computação em grade baseado em redes *peer-to-peer* (P2P). Seu foco principal é a execução de aplicações definidas como *bag-of-tasks*[Cirne et al., 2003]; uma aplicação *bag-of-tasks* é aquela que pode ser dividida em tarefas que não necessitam de comunicação entre si. Para impedir que uma aplicação maliciosa use os recursos de uma forma indevida (inclusive a própria rede), o OurGrid possui o SWAN [Andrade et al., 2005], uma implementação de *sandbox* que permite a execução de aplicações em uma máquina virtual isolada. Detsch [Detsch et al., 2004] propõe um arcabouço de segurança para redes P2P que foi implementado no OurGrid. Este arcabouço define serviços de autenticação e confidencialidade para o sistema.

O objetivo do repositório de aplicações do InteGrade é tratar as aplicações dos clientes como recursos computacionais disponibilizados através de serviços de armazenamento, busca, recuperação e gerenciamento de versões. Além disso, o InteGrade provê um ambiente de suporte para iniciar e encerrar a execução dessas aplicações. O papel do repositório de aplicações no InteGrade aproxima-se da abordagem adotada pelo Legion e não encontra similaridade direta no Globus Toolkit ou no Ourgrid.

O gerenciamento de binários e a disponibilização de aplicações para execução são parte dos princípios de projeto do Legion. O Globus Toolkit considera esses requisitos como funções de alto nível, não sendo, portanto, parte direta dessa tecnologia [Grimshaw et al., 2004]. O Ourgrid limita-se a fornecer mecanismos para descrição e execução de *jobs*, não tendo o gerenciamento de binários como objetivo definido.

A abordagem adotada pelo repositório de aplicações, dando às aplicações dos clientes status de recursos computacionais, permite que políticas de proteção possam ser definidas explicitamente, propiciando gerenciamento e compartilhamento seguro das aplicações. Os serviços implementados permitem adicionar uma camada de segurança transparente ao usuário do sistema de modo a garantir autenticidade, confidencialidade e integridade das aplicações.

3 Descrição da Implementação

A implementação do repositório seguro de aplicações oferece as seguintes características para segurança e proteção das informações: autenticação, confidencialidade, integridade, autorização e registros de eventos (*logging*).

A autenticação permite restringir o uso do repositório a usuários e pro-

cessos conhecidos, sendo a base das demais características de segurança. No contexto do repositório de aplicações, cada usuário precisa ser autenticado antes de usar o serviço. Essa autenticação garantirá que todas as operações realizadas no repositório, tais como registro de aplicações e armazenamento e recuperação de arquivos executáveis, sejam feitas apenas por usuários reconhecidos pelo sistema.

O sistema descrito neste relatório possibilita que o repositório de aplicações utilize técnicas de criptografia para obter confidencialidade em operações tais como o armazenamento e a recuperação de arquivos executáveis. Os dados criptografados na origem são ilegíveis durante o transporte, tornando-se acessíveis no destino pela ação dos mesmos mecanismos de segurança. Essa característica é opcional em nossa implementação por motivos de desempenho.

Prover a integridade de dados consiste em impedir que arquivos, por exemplo executáveis, sejam modificados de forma maliciosa durante a transferência ou enquanto armazenados no repositório.

Níveis diferentes de autorização são necessários no repositório de aplicações. Através da autenticação, o usuário tem a visibilidade restrita do espaço de nomes no repositório de aplicações. Ao registrar uma aplicação, ele o fará somente em uma área para a qual está autorizado ou em uma área de domínio público.

Finalmente, o registro de eventos (*logging*) fornece uma base de informações para auditorias futuras. A seguir descrevemos os componentes básicos da implementação do repositório de aplicações seguro.

3.1 O Repositório de Aplicações

Para que o usuário do InteGrade não seja obrigado a resubmeter uma aplicação sempre que deseje executá-la, o repositório de aplicações do InteGrade permite registrar e recuperar uma aplicação. Além de persistência, ele oferece outras facilidades para uso e controle das aplicações da grade por usuários e administradores. Essas facilidades envolvem, por exemplo, aspectos relacionados à segurança e ao gerenciamento de versões.

A implementação de persistência no repositório de aplicações consiste no armazenamento em disco das aplicações permitindo que sejam identificadas univocamente na grade e utilizando nomes compostos para referenciá-las. Essa funcionalidade permite a implementação da interface gráfica de usuário para navegação pelo espaço de nomes, de forma similar a diretórios em um sistema de arquivos, cujo conteúdo abrange as aplicações registradas e informações associadas a essas aplicações. O gerenciamento de versões pelo repositório de aplicações provê suporte para tratamento de variantes das

aplicações em diferentes plataformas.

A interface do repositório permite a manipulação do espaço de nomes do repositório incluindo funcionalidades que facilitam o gerenciamento e reutilização das aplicações na grade. As principais funcionalidades dessa interface incluem registrar uma aplicação e diferentes versões de códigos binários dessa aplicação associadas a diferentes plataformas computacionais e recuperar esses arquivos binários para execução na grade.

Um esquema de metadados é utilizado para manter informações sobre as aplicações e sobre suas versões de código binário dessas aplicações. Ao registrar uma aplicação, o sistema cria no repositório de aplicações um diretório com o nome da aplicação e dentro desse diretório um arquivo chamado *.AppDescription*. Esse arquivo contém metainformações sobre a aplicação, incluindo quantidade e nome das versões de códigos binários existentes para essa aplicação. Ao armazenar uma nova versão de código binário para uma aplicação, o sistema atualiza o arquivo de metadados. Esse arquivo também é atualizado se uma versão de código binário for excluída.

As informações armazenadas no arquivo de metadados são utilizadas no escalonamento de uma aplicação quando de sua execução. Ao submeter uma aplicação, o usuário pode escolher se deseja executar uma versão específica do código binário ou se a aplicação pode ser executada por qualquer das versões armazenadas no repositório. No primeiro caso o escalonador deverá selecionar apenas as máquinas disponíveis que atendam a essa restrição sobre a plataforma. No segundo caso, o escalonador seleciona pares compatíveis de máquinas e versões ao distribuir a tarefa entre os nós da grade.

A Figura 2 mostra a interface gráfica do módulo ASCT navegando sobre o espaço de nomes do repositório de aplicações. Cada usuário tem acesso a um espaço de nomes próprio e também ao espaço de nomes sob o diretório *public* (ver figura), o qual também é visível pelos demais usuários. Nesse exemplo, a aplicação "Alinhamento de Seqüência de DNA" foi registrada com três tipos de executáveis correspondentes a três plataformas de hardware diferentes: Linux/86, Linux/AMD64 e MacOSX/PowerPC Macintosh. O menu de contexto apresentado na figura pode ser utilizado para submeter aplicações à grade.

O módulo de segurança do InteGrade possibilita ao repositório de aplicações incorporar informações associando aplicações registradas a permissões de usuários.

3.2 A API GSS

O repositório seguro de aplicações foi implementado sobre a API GSS. Esta API provê serviços de segurança de alto nível que permitem aos desen-

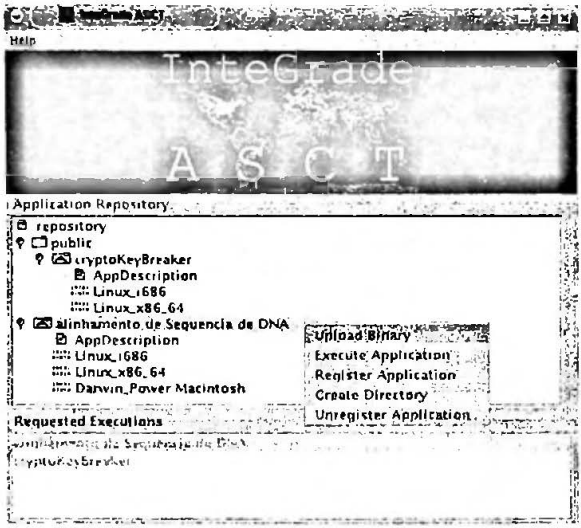

Figura 2: Visão do Repositório de Aplicações via ASCTGui.

volvedores ignorarem detalhes de segurança da rede, favorecendo fortemente a portabilidade dos sistemas [Linn, 1997]. Desse modo, diferentes mecanismos e tecnologias de segurança como o Kerberos [Kohl and Neuman, 1993] ou chaves privadas/públicas podem ser utilizados, de modo transparente, sobre o GSS.

O GSS disponibiliza, basicamente, dois conceitos: contexto e serviços de segurança. O contexto de segurança pode ser tratado como um estado de confiança entre grupo de aplicações. Os serviços de segurança disponibilizados pelo GSS são a integridade e a confidencialidade. Enquanto o primeiro garante que os dados não sejam modificados, o último certifica que eles não sejam interceptados.

A Figura 3 mostra os passos necessários para criar um contexto no GSS. Inicialmente, o servidor se autentica e recebe credenciais do serviço de autenticação (1), por exemplo, de um servidor Kerberos. Em seguida, o cliente, para iniciar um contexto de segurança, se autentica no serviço de autenticação (2) e cria, através do GSS, um pacote especial (*token*) de início de contexto, enviando-o ao servidor através da rede (3,4). Na seqüência, o servidor verifica este *token* (5), gerando um outro *token* que corresponde à reposta do servidor à requisição do cliente (6). O cliente recebe este *token* e verifica se o contexto foi criado, repetindo os passos anteriores caso o processo não

seja finalizado (7). Sob este contexto, o cliente e o servidor poderão assinar e criptografar as mensagens que trocarem durante toda a sessão do serviço a ser utilizado.

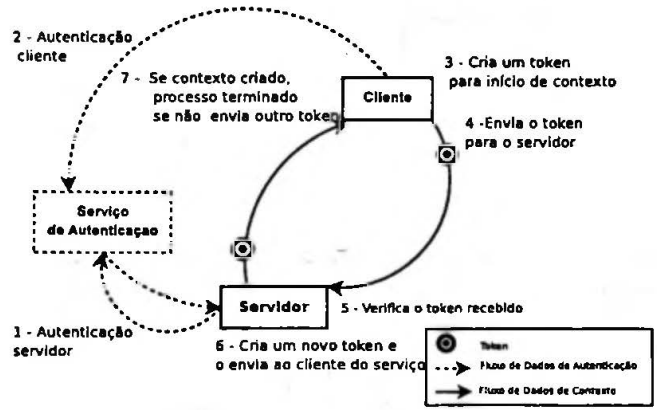

Figura 3: Criação do contexto no GSS.

3.3 Módulos de Segurança

Os módulos que implementam os serviços de segurança são o LSM (*Local Security Manager*) e o GSM (*Global Security Manager*), vistos na Figura 4. Ambos utilizam o GSS para obter os contextos de segurança entre o repositório de aplicações e os módulos que com ele interagem. O GSS, por sua vez, tem seus serviços implementados através do Kerberos. A implementação atual do repositório seguro utiliza a versão 5 do Kerberos desenvolvida pelo MIT[4] para a linguagem C. Para a linguagem Java, essa implementação utiliza a API GSS Java.

O GSM é responsável por iniciar e gerenciar os contextos. Cada cliente, através do LSM, possui um contexto de segurança com o GSM. Todas as trocas de mensagens entres os módulos são feitas utilizando esses contextos. O LSM pode efetuar quatro operações básicas: assinar e verificar uma mensagem enviada com o contexto que ela possui (operações requestSignature() e checkSignature()), ou ainda, assinar e verificar uma mensagem

[4]http://web.mit.edu/kerberos/www

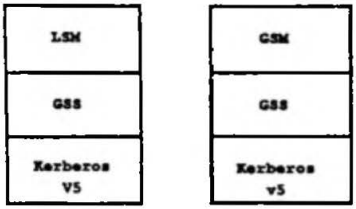

Figura 4: Novos módulos do InteGrade.

enviada sob um outro contexto (requestForeignSignature() e checkForeignSignature()). Mais especificamente, o repositório usa o GSM (via o LSM) para verificar e assinar os arquivos executáveis das aplicações dos seus clientes, enquanto estes clientes usam o LSM para assinar e verificar arquivos durante o armazenamento e recuperação no repositório.

A Figura 5 nos fornece a visão da localização dos novos módulos na arquitetura do InteGrade. O GSM, localizado em um nó Gerenciador do Aglomerado, é responsável pelo gerenciamento dos contextos de segurança entre os diversos módulos. O LSM (presente nos módulos AppRepos, ASCTGui e LRM) autentica cada módulo e cria contextos de segurança com o GSM.

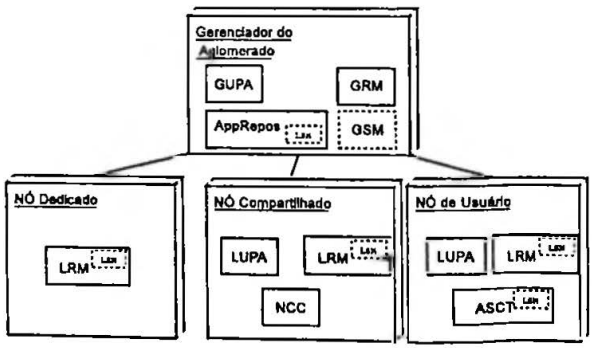

Figura 5: Localização dos módulos de segurança na arquitetura do InteGrade.

3.4 Protocolos seguros

O protocolo de armazenamento de uma aplicação do InteGrade é apresentado na Figura 6. O ASCT utiliza o LSM para assinar o arquivo e solicitar ao repositório de aplicações o seu armazenamento (1,2,3). O repositório de aplicações verifica a assinatura deste binário através do LSM (4), que o faz através do GSM (4.1). Sendo o arquivo verificado com sucesso, o repositório de aplicações calcula um resumo do binário através de uma função *hash* conhecida[5] (5), por exemplo o MD5, e o armazena a seguir no sistema de arquivos (6). Ao final do processo de armazenamento em disco, o repositório de aplicações envia a identificação (ID) assinada da aplicação (7), que por sua vez também é verificada (8). No caso de qualquer uma das operações falhar é gerada uma exceção que é devidamente tratada e registrada em arquivo de *log*.

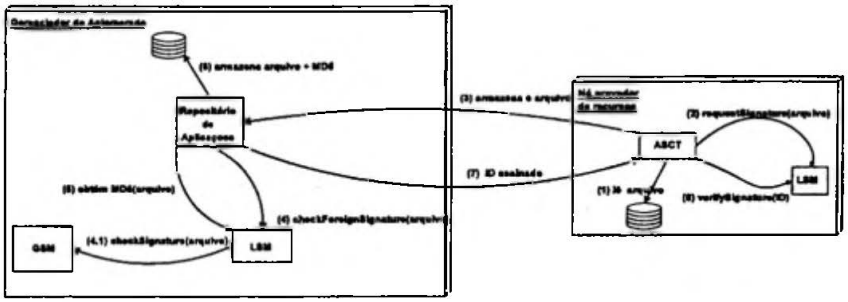

Figura 6: Protocolo de armazenamento de um executável de uma aplicação.

A Figura 7 apresenta o protocolo de recuperação de um arquivo. Ao receber a solicitação de execução, um determinado LRM deverá requerer o binário compatível com a sua plataforma. De posse do ID da aplicação, o LRM o assina e indica ao repositório de aplicações o arquivo executável desejado (1,2). O Repositório obtém o arquivo desejado e verifica sua integridade através da função de *hash* (3,4). Antes de enviar o arquivo executável ao LRM, o repositório assina o binário através do LSM (5), o qual repassa essa função para o GSM (5.1), pois a assinatura deve ser feita através do

[5]Para garantir que esse resumo não possa ser gerado pelo atacante, o GSM adiciona ao arquivo uma chave que é obtida pela API GSS. Essa é uma técnica bastante utilizada em sistemas de segurança [Terada, 2000].

contexto criado para o LRM que fez a requisição. Ao receber o arquivo, o LRM verifica sua assinatura (6,7) e o armazena em disco para execução (8).

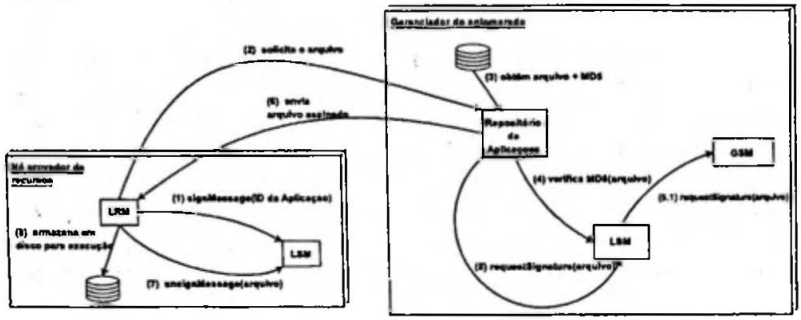

Figura 7: Protocolo de recuperação de um binário de uma aplicação.

Os protocolos acima descritos garantem a autenticidade e a integridade dos arquivos executáveis submetidos à grade. Todas as mensagens trocadas entre o repositório de aplicações e seus clientes são devidamente assinadas e criptografadas. Pessoas mal intencionadas terão mais dificuldade para interferir no uso da grade ao utilizar técnicas de ataque conhecidas para modificar, interceptar ou fabricar dados. O uso de função de *hash* no sistema de arquivos é útil para tentar impedir a modificação das aplicações através de falhas de segurança no sistema de arquivos do gerenciador do aglomerado.

Para avaliar o desempenho dos mecanismos de segurança descritos, foram realizados experimentos que consistiram na transferência de arquivos de diferentes tamanhos entre o repositório e um nó da grade. Os experimentos foram executados sobre um *hardware* AMD Athlon XP 2800+, com 900MB de memória *RAM* utilizando os algoritmos de criptografia DES triplo e SHA1 [Terada, 2000]. Os tempos de execução obtidos com o repositório de aplicações com os módulos de segurança desativados foram comparados com os obtidos com os mecanismos de segurança ativados. Ambos os experimentos foram realizados sobre uma mesma amostra de dez binários executáveis, cujos tamanhos variavam, em ordem crescente, de aproximadamente 100 KB (menor tamanho) até 1000KB (maior tamanho), diferindo de 100 KB do próximo.

Os resultados na Figura 8 mostram um acréscimo de tempo na execução da implementação do repositório seguro em comparação com a execução da implementação sem ativação do mecanismo de segurança. Isso se deve às re-

petidas operações de criptografia e descriptografia na movimentação de cada arquivo no primeiro caso. Um outro motivo é o uso do DES um algoritmo sabidamente lento [Terada, 2000].

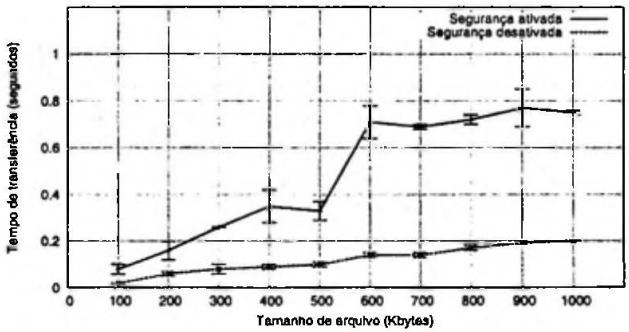

Figura 8: Resultados experimentais.

4 Conclusão e próximos passos

A implementação do repositório seguro de aplicações disponibiliza aos seus usuários três dos quatro fundamentos básicos na implementação de sistemas de segurança: confidencialidade, integridade e uso legítimo. A confidencialidade e a integridade são oferecidas através do uso de serviços disponíveis na API GSS. Estes dois fundamentos são reforçados pelo fato de todas as mensagens entre clientes e o repositório de aplicações serem protegidas por assinatura e criptografia. A sessão estabelecida, com autenticação prévia dentro de um contexto de segurança, acrescida da visão limitada de cada usuário às aplicações registradas, possibilita o uso legítimo dos recursos. No entanto, o repositório seguro de aplicações não oferece o fundamento da disponibilidade, uma vez que um usuário malicioso poderia impossibilitar a sua utilização sobrecarregando-o através de ataques de negação de serviços, por exemplo. Devido à natureza dos potenciais usuários (i.e., computação de alto desempenho a baixo custo), a proteção contra ataques deste tipo não é nossa prioridade no momento. Como recurso adicional, o repositório seguro implementa o armazenamento de registros (*logs*) para auxiliar na análise de incidentes de segurança que possam ocorrer.

Os resultados experimentais obtidos poderão ser melhorados em uma fase posterior. Espera-se que a utilização de outros algoritmos de criptografia, tais como o RSA, possa afetar de forma positiva o desempenho da solução.

Na próxima fase do projeto, durante o segundo semestre do ano em curso, o sistema será testado sobre o ambiente de rede de alta velocidade. Nesse ambiente, nós do InteGrade presentes em redes distantes (UFRJ e USP) se comunicão sobre a rede GIGA e o impacto da rede sobre o sistema será analisado.

Será considerado também o uso de dois servidores Kerberos em sítios diferentes e com relações de confiança entre eles. A idéia é verificar qual das duas soluções terá melhor qualidade: a distribuída ou a totalmente centralizada, ambas sobre uma rede de alta velocidade.

Uma implementação totalmente distribuída baseada em redes de confiança será usada num fase posterior dentro do projeto Taquara. Esta solução possui seu desenvolvimento em curso e visa implementar um sistema de segurança mais abragente, totalmente distribuído, baseado em uma extensão de redes de confiança (SDSI - *Simple Distributed Security Infrastructure* e SPKI - *Simple Public Key Infrastructure*)[Ellison et al., 1999].

Referências

[Andrade et al., 2005] Andrade, N., Costa, L., Germóglio, G., and Cirne, W. (2005). Peer-to-peer grid computing with the ourgrid community. In *SBRC 2005 - IV Salão de Ferramentas*, pages 1191–1198. Simpósio Brasileiro de Redes de Computadores (SBRC).

[Berman et al., 2003] Berman, F., Fox, G., Hey, A. J. G., and Hey, T. (2003). *Grid Computing: Making the Global Infrastructure a Reality*. John Wiley & Sons, Inc.

[Cirne et al., 2003] Cirne, W., Paranhos, D., Costa, L., Santos-Neto, E., Brasileiro, F., Sauvé, J., Silva, F. A. B., Barros, C. O., and Silveira, C. (2003). Running Bag-of-Tasks Applications on Computational Grids: The MyGrid Approach. In *Proceedings of the 2003 International Conference on Parallel Processing*, pages 407–416.

[de Camargo et al., 2004] de Camargo, R. Y., Goldchleger, A., Carneiro, M., and Kon, F. (2004). Grid: An Architectural Pattern. In *The 11th Conference on Pattern Languages of Programs (PLoP'2004)*, Monticello, Illinois, USA.

[Detsch et al., 2004] Detsch, A., Gaspary, L. P., Barcellos, M. P., and Cavalheiro, G. G. H. (2004). Towards a flexible security framework for peer-to-peer based grid computing. In *Proceedings of the 2nd workshop on*

Middleware for grid computing, pages 52–56, New York, NY, USA. ACM Press.

[Ellison et al., 1999] Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Bell, S., and Ylonen, T. (1999). SPKI Certificate Theory. Internet RFC #2693.

[Ferrari et al., 1999] Ferrari, A., Knabe, F., Humphrey, M., Chapin, S. J., and Grimshaw, A. S. (1999). A Flexible Security System for Metacomputing Environments. In *Proceedings of the 7th International Conference on High-Performance Computing and Networking*, pages 370–380. Springer-Verlag.

[Foster and Kesselman, 1997] Foster, I. and Kesselman, C. (1997). Globus: A Metacomputing Infrastructure Toolkit. *International Journal of Supercomputer Applications*, 2(11):115–128.

[Foster and Kesselman, 2003] Foster, I. and Kesselman, C. (2003). *The Grid 2: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers Inc.

[Foster et al., 1998] Foster, I., Kesselman, C., Tsudik, G., and Tuecke, S. (1998). A Security Architecture for Computational Grids. In *Proceedings of the 5th ACM Conference on Computer and Communications Security*, pages 83–92.

[Garfinkel and Spafford, 1996] Garfinkel, S. and Spafford, G. (1996). *Practical UNIX & Internet Security.* O Reilly & Associates, Inc.

[Goldchleger et al., 2004] Goldchleger, A., Kon, F., Goldman, A., and Finger, M. (2004). InteGrade: Object-Oriented Grid Middleware Leveraging Idle Computing Power of Desktop Machines. In *ACM/IFIP/USENIX International Workshop on Middleware for Grid Computing.*

[Grimshaw et al., 2004] Grimshaw, A. S., Humphrey, M. A., and Natrajan, A. (2004). A philosophical and technical comparison of legion and globus. *IBM J. Res. Dev.*, 48(2):233–254.

[Grimshaw et al., 1997] Grimshaw, A. S., Wulf, W. A., and Team, T. L. (1997). The Legion Vision of a Worldwide Virtual Computer. *Communications of the ACM*, 40(1):39-45.

[Kohl and Neuman, 1993] Kohl, J. and Neuman, C. (1993). The Kerberos network authentication service (v5). Internet RFC #1510.

[Linn, 1997] Linn, J. (1997). The Generic Security Service Application Program Interface (GSS API). Technical Report Internet RFC 2078, Network Working Group.

[Meder et al., 2001] Meder, S., Welch, V., Chicago, U., Tuecke, S., and Engert, D. (2001). Gss-api extensions. http://www.gridforum.org/2_SEC/GSI.htm.

[Terada, 2000] Terada, R. (2000). *Segurança de Dados. Criptografia em Redes de Computadores.* Editora Edgard Blücher Ltda.

RELATÓRIOS TÉCNICOS

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
Instituto de Matemática e Estatística da USP

A listagem contendo os relatórios técnicos anteriores a 2000 poderá ser consultada ou solicitada à Secretaria do Departamento, pessoalmente, por carta ou e-mail (mac@ime.usp.br).

Douglas Moreto and Markus Endler
EVALUATING COMPOSITE EVENTS USING SHARED TREES
RT-MAC-2001-01, janeiro 2001, 26 pp.

Vera Nagamura and Markus Endler
COORDINATING MOBILE AGENTS THROUGH THE BROADCAST CHANNEL
RT-MAC-2001-02, janeiro 2001, 21 pp.

Júlio Michael Stern
THE FULLY BAYESIAN SIGNIFICANCE TEST FOR THE COVARIANCE PROBLEM
RT-MAC-2001-03, fevereiro 2001, 15 pp.

Marcelo Finger and Renata Wassermann
TABLEAUX FOR APPROXIMATE REASONING
RT- MAC-2001-04, março 2001, 22 pp.

Julio Michael Stern
FULL BAYESIAN SIGNIFICANCE TESTS FOR MULTIVARIATE NORMAL STRUCTURE MODELS
RT-MAC-2001-05, junho 2001, 20 pp.

Paulo Sérgio Naddeo Dias Lopes and Hernán Astudillo
VIEWPOINTS IN REQUIREMENTS ENGINEERING
RT-MAC-2001-06, julho 2001, 19 pp.

Fabio Kon
O SOFTWARE ABERTO E A QUESTÃO SOCIAL
RT- MAC-2001-07, setembro 2001, 15 pp.

Isabel Cristina Italiano, João Eduardo Ferreira and Osvaldo Kotaro Takai
ASPECTOS CONCEITUAIS EM DATA WAREHOUSE
RT – MAC-2001-08, setembro 2001, 65 pp.

Marcelo Queiroz , Carlos Humes Junior and Joaquim Júdice
ON FINDING GLOBAL OPTIMA FOR THE HINGE FITTING PROBLEM
RT- MAC –2001-09, novembro 2001, 39 pp.

Marcelo Queiroz , Joaquim Júdice and Carlos Humes Junior
THE SYMMETRIC EIGENVALUE COMPLEMENTARITY PROBLEM
RT- MAC-2001-10, novembro 2001, 33 pp.

Marcelo Finger, and Fernando Antonio Mac Cracken Cezar
BANCO DE DADOS OBSOLESCENTES E UMA PROPOSTA DE IMPLEMENTAÇÃO.
RT- MAC - 2001-11- novembro 2001, 90 pp.

Flávio Soares Correa da Silva
TOWARDS A LOGIC OF PERISHABLE PROPOSITIONS
RT- MAC- 2001-12 - novembro 2001, 15 pp.

Alan M. Durham
O DESENVOLVIMENTO DE UM INTERPRETADOR ORIENTADO A OBJETOS PARA ENSINO DE LINGUAGENS
RT-MAC-2001-13 – dezembro 2001, 21 pp.

Alan M. Durham
A CONNECTIONLESS PROTOCOL FOR MOBILE AGENTS
RT-MAC-2001-14 – dezembro 2001, 12 pp.

Eugênio Akihiro Nassu e Marcelo Finger
O SIGNIFICADO DE "AQUI" EM SISTEMAS TRANSACIONAIS MÓVEIS
RT-MAC-2001-15 – dezembro 2001, 22 pp.

Carlos Humes Junior, Paulo J. S. Silva e Benar F. Svaiter
SOME INEXACT HYBRID PROXIMAL AUGMENTED LAGRANGIAN ALGORITHMS
RT-MAC-2002-01 – Janeiro 2002, 17 pp.

Roberto Speicys Cardoso e Fabio Kon
APLICAÇÃO DE AGENTES MÓVEIS EM AMBIENTES DE COMPUTAÇÃO UBÍQUA.
RT-MAC-2002-02 – Fevereiro 2002, 26 pp.

Julio Stern and Zacks
TESTING THE INDEPENDENCE OF POISSON VARIATES UNDER THE HOLGATE BIVARIATE DISTRIBUTION: THE POWER OF A NEW EVIDENCE TEST.
RT- MAC – 2002-03 – Abril 2002, 18 pp.

E. N. Cáceres, S. W. Song and J. L. Szwarcfiter
A PARALLEL ALGORITHM FOR TRANSITIVE CLOSURE
RT-MAC – 2002-04 – Abril 2002, 11 pp.

Regina S. Burachik, Suzana Scheimberg, and Paulo J. S. Silva
A NOTE ON THE EXISTENCE OF ZEROES OF CONVEXLY REGULARIZED SUMS OF MAXIMAL MONOTONE OPERATORS
RT- MAC 2002-05 – Maio 2002, 14 pp.

C.E.R. Alves, E.N. Cáceres, F. Dehne and S. W. Song
A PARAMETERIZED PARALLEL ALGORITHM FOR EFFICIENT BIOLOGICAL SEQUENCE COMPARISON
RT-MAC-2002-06 – Agosto 2002, 11pp.

Julio Michael Stern
SIGNIFICANCE TESTS, BELIEF CALCULI, AND BURDEN OF PROOF IN LEGAL AND SCIENTIFIC DISCOURSE
RT- MAC – 2002-07 – Setembro 2002, 20pp.

Andrei Goldchleger, Fabio Kon, Alfredo Goldman vel Lejbman, Marcelo Finger and Siang Wun Song.
INTEGRADE: RUMO A UM SISTEMA DE COMPUTAÇÃO EM GRADE PARA APROVEITAMENTO DE RECURSOS OCIOSOS EM MÁQUINAS COMPARTILHADAS.
RT-MAC – 2002-08 – Outubro 2002, 27pp.

Flávio Protasio Ribeiro
OTTERLIB – A C LIBRARY FOR THEOREM PROVING
RT- MAC – 2002-09 – Dezembro 2002 , 28pp.

Cristina G. Fernandes, Edward L. Green and Arnaldo Mandel
FROM MONOMIALS TO WORDS TO GRAPHS
RT-MAC – 2003-01 – fevereiro 2003, 33pp.

Andrei Goldchleger, Márcio Rodrigo de Freitas Carneiro e Fabio Kon
GRADE: UM PADRÃO ARQUITETURAL
RT- MAC – 2003-02 – março 2003, 19pp.

C. E. R. Alves, E. N. Cáceres and S. W. Song
SEQUENTIAL AND PARALLEL ALGORITHMS FOR THE ALL-SUBSTRINGS LONGEST COMMON SUBSEQUENCE PROBLEM
RT- MAC – 2003-03 – abril 2003, 53 pp.

Said Sadique Adi and Carlos Eduardo Ferreira
A GENE PREDICTION ALGORITHM USING THE SPLICED ALIGNMENT PROBLEM
RT- MAC – 2003-04 – maio 2003, 17pp.

Eduardo Laber, Renato Carmo, and Yoshiharu Kohayakawa

QUERYING PRICED INFORMATION IN DATABASES: THE CONJUNTIVE CASE
RT-MAC – 2003-05 – julho 2003, 19pp.

E. N. Cáceres, F. Dehne, H. Mongelli, S. W. Song and J.L. Szwarcfiter
A COARSE-GRAINED PARALLEL ALGORITHM FOR SPANNING TREE AND CONNECTED COMPONENTS
RT-MAC – 2003-06 – agosto 2003, 15pp.

E. N. Cáceres, S. W. Song and J.L. Szwarcfiter
PARALLEL ALGORITMS FOR MAXIMAL CLIQUES IN CIRCLE GRAPHS AND UNRESTRICTED DEPTH SEARCH
RT-MAC – 2003-07 – agosto 2003, 24pp.

Julio Michael Stern
PARACONSISTENT SENSITIVITY ANALYSIS FOR BAYESIAN SIGNIFICANCE TESTS
RT-MAC – 2003-08 – dezembro 2003, 15pp.

Lourival Paulino da Silva e Flávio Soares Corrêa da Silva
A FORMAL MODEL FOR THE FIFTH DISCIPLINE
RT-MAC-2003-09 – dezembro 2003, 75pp.

S. Zacks and J. M. Stern
SEQUENTIAL ESTIMATION OF RATIOS, WITH APPLICATION TO BAYESIAN ANALYSIS
RT-MAC – 2003-10 - dezembro 2003, 17pp.

Alfredo Goldman, Fábio Kon, Paulo J. S. Silva and Joe Yoder
BEING EXTREME IN THE CLASSROOM: EXPERIENCES TEACHING XP
RT–MAC – 2004-01-janeiro 2004, 18pp.

Cristina Gomes Fernandes
MULTILENGTH SINGLE PAIR SHORTEST DISJOINT PATHS
RT-MAC 2004-02 – fevereiro 2004, 18pp.

Luciana Brasil Rebelo
ÁRVORE GENEALÓGICA DAS ONTOLOGIAS
RT- MAC 2004-03 – fevereiro 2004, 22pp.

Marcelo Finger
TOWARDS POLYNOMIAL APPROXIMATIONS OF FULL PROPOSITIONAL LOGIC
RT- MAC 2004-04 – abril 2004, 15pp.

Renato Carmo, Tomás Feder, Yoshiharu Kohayakawa, Eduardo Laber, Rajeev Motwani, Liadan O' Callaghan, Rina Panigrahy, Dilys Thomas
A TWO- PLAYER GAME ON GRAPH FACTORS
RT-MAC 2004-05 – Julho 2004

Paulo J. S. Silva, Carlos Humes Jr.
RESCALED PROXIMAL METHODS FOR LINEARLY CONSTRAINED CONVEX PROBLEMS
RT-MAC 2004-06-setembro 2004

Julio M. Stern
A CONSTRUCTIVIST EPISTEMOLOGY FOR SHARP STATISTICAL HYPOTHESES IN SCIENTIFIC RESEARCH
RT-MAC 2004-07- outubro 2004

Arlindo Flávio da Conceição, Fábio Kon
O USO DO MECANISMO DE PARES DE PACOTES SOBRE REDES IEEE 802.11b
RT-MAC 2004-08 – outubro 2004

Carlos H. Cardonha, Marcel K. de Carli Silva e Cristina G. Fernandes
COMPUTAÇÃO QUÂNTICA: COMPLEXIDADE E ALGORITMOS
RT- MAC 2005-01 – janeiro 2005

C.E.R. Alves, E. N. Cáceres and S. W. Song
A BSP/CGM ALGORITHM FOR FINDING ALL MAXIMAL CONTIGUOS SUBSEQUENCES OF A SEQUENCE OF NUMBERS
RT- MAC- 2005-02 – janeiro 2005

Flávio S. Corrêa da Silva
WHERE AM I? WHERE ARE YOU?
RT- MAC- 2005-03 – março 2005, 15pp.

Christian Paz-Trillo, Renata Wassermann and Fabio Kon
A PATTERN-BASED TOOL FOR LEARNING DESIGN PATTERNS
RT- MAC – 2005-04 – abril 2005, 17pp.

Wagner Borges and Julio Michael Stern
ON THE TRUTH VALUE OF COMPLEX HYPOTHESIS
RT- MAC – 2005-05 – maio 2005, 15 pp.

José de Ribamar Braga Pinheiro Jr., Alexandre César Tavares Vida and Fabio Konl
IMPLEMENTAÇÃO DE UM REPOSITÓRIO SEGURO DE APLICAÇÕES BASEADO EM GSS – PROJETO TAQUARA
RT- MAC – 2005-06 – agosto 2005, 21 pp.