INSTITUTO MILITAR DE ENGENHARIA

FABIO LOPES LICHT

FORNECIMENTO AUTOMATIZADO DE CERTIFICADOS DE CURTA DURAÇÃO PARA DISPOSITIVOS MÓVEIS EM GRADES COMPUTACIONAIS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências em Sistemas e Computação.

Orientador: Prof. Bruno Schulze - D. Sc.

Co-orientador: Prof. Edison Ishikawa - D. Sc.

Rio de Janeiro 2006

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro - RJ CEP: 22290-270

L699f Licht, Fabio Lopes
Fornecimento Automatizado de Certificados de Curta Duração para Dispositivos Móveis em Grades Computacionais - Rio de Janeiro : Instituto Militar de Engenharia, 2006.
169p.: il., graf., tab.
Dissertação (mestrado) - Instituto Militar de Engenharia - Rio de Janeiro, 2006.
1. Certificados de Curta Duração. 2. Autoridade Certificadora. 3. Java. 4. Grades Computacionais.
I. Institulo Militar de Engenharia. II. Título.

CDD 005.8

INSTITUTO MILITAR DE ENGENHARIA

FABIO LOPES LICHT

FORNECIMENTO AUTOMATIZADO DE CERTIFICADOS DE CURTA DURAÇÃO PARA DISPOSITIVOS MÓVEIS EM GRADES COMPUTACIONAIS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências em Sistemas e Computação.

Orientador: Prof. Bruno Schulze - D. Sc.

Co-orientador: Prof. Edison Ishikawa - D. Sc.

Aprovada em 20 de dezembro de 2006 pela seguinte Banca Examinadora:

Prof. Bruno Richard Schulze - D. Sc. do LNCC - Presidente

Prof. Edison Ishikawa - D. Sc. do IME

Prof. Antônio Tadeu Azevedo Gomes - D. Sc. do LNCC

Prof. Claudio Gomes de Mello - D. Sc. do IME

Rio de Janeiro

2006

À Deus acima de tudo por ter me dado a capacidade e força para estudar e chegar até aqui. À meus pais por terem me ensinado a nunca desistir de um sonho e sempre lutar com dignidade. À minha querida irmã por sempre ter me apoiado em todos os momentos. À meu orientador e amigo Bruno por ter confiado em mim e sempre me apoiado, desde a graduação e até agora. Sem ele, não teria conseguido. À minha filha Nathalia por ser um anjo sempre presente, por ser minha melhor amiga e por ter sempre me apoiado. À minha namorada Príscila por me apoiar, por sorrir nos momentos de alegria e tornar os momentos de tristeza breves o suficiente para não causarem danos.

AGRADECIMENTOS

À meu orientador, Prof. Bruno Schulze pela orientação e apoio desde 2001 no início da graduação até hoje e por estar sempre presente nas conquistas acadêmicas como ninguém. Por ter sido um grande amigo nas horas em que mais precisava.

À meu co-orientador Prof. Edison Ishikawa pela força nos momentos onde "tudo" parecia estar caindo.

Ao Prof. Paulo Rosa por ter me dado a chance de fazer esse mestrado e por ter confiado em meu potencial. Sem ele, este trabalho não seria possível.

Ao Prof. Jauvane por ter apoiado minha entrada no Instituto Militar de Engenharia e pelo apoio durante todo o curso.

À Secretaria do IME, especialmente à Emília por ser a pessoa maravilhosa que é e por estar sempre disposta à ajudar, principalmente nas horas em que parecia não existir solução.

Ao Instituto Militar de Engenharia (IME) por ter me aceito no mestrado e por ter dado condições da conclusão.

Ao Laboratório Nacional de Computação Científica (LNCC) por ter cedido a instituição para minha pesquisa e desenvolvimento.

À CAPES por financiar meus estudos e de tantos alunos que sem essa ajuda não poderiam concluir uma pós-graduação.

À todos os amigos do IME e do LNCC que nas horas difíceis estavam presentes, especialmente Luís César, Luis Rodrigo, Matheus Bandini, Thais Mello, Anolan, Márcio, Gadelha e todos os outros que mesmo não tendo seus nomes aqui estarão gravados no coração.

Sumário

Lista de Figuras

RESUMO

Este trabalho apresenta a proposta de extensão de um serviço existente de fornecimento de certificados confiáveis de curta duração, com o intuito de autorizar nós móveis a ingressarem temporariamente em grades computacionais. Dentre as atividades deste trabalho, inclui-se a utilização de chaves públicas, privadas, fornecimento de certificados confiáveis e dinâmicos além de criptografia e utilização de redes móveis. O uso destes padrões em conjunto com a ferramenta MyProxy, responsável por fornecimento de certificados dinâmicos de curta duração, irá propiciar maior agilidade na inclusão de novos usuários e a sua extensão para inclusão de nós móveis que quiserem associar-se à grade computacional, modificando um padrão atualmente manual de inclusão de novos nós, em um sistema automático e funcional. Fortalecendo o uso da grade por dispositivos móveis e por usuários temporários, sem abrir mão da segurança.

ABSTRACT

This work presents a proposal to extend an existing service for short lived certificates (SLCs), with the purpose of allowing mobile nodes to enter computational grids. Amongst the parts of this work, it is cited, the use of public and private keys, supply of trustworthy certificates and dynamic SLCs. MyProxy is the tool responsible for supplying dynamic certificates and increase agility in the inclusion of new users and also the extension to nodes that want to enter a computational grid, modifying a currently manual standard of inclusion of new nodes, in an automatic and functional system. Fortifying the use of the grid for mobile devices and temporary users, without opening hand of the security.

Capítulo 1

Introdução

A necessidade sempre crescente de capacidade computacional tem levado empresas durante anos a criarem cada vez mais dispositivos com maior poder de processamento, seguindo a "Lei de Moore"[1] que afirma que a capacidade computacional tenderia a dobrar sua capacidade a cada 18 meses. Hoje em dia essa "lei" tem suas exceções e é o próprio Gordon Moore quem em uma entrevista em 2003, questiona ela, afirmando:

> *O que a dirige desde sua formulação em 1965 é a capacidade de criar coisas menores e menores, e eventualmente o fato de que materiais feitos de átomos são um limite real. Começamos a ver efeitos dessa mecânica já em aparelhos que produzimos hoje. Creio que teremos mais duas ou três gerações em direção ao caminho que estivemos sempre. Então, teremos que mudar. Podemos produzir chips maiores. Pode não ser tão rápido quanto foi até agora - algo que dobre a cada quatro anos, em vez de dois anos, é quase sem precedentes. Vai diminuir bastante a velocidade, e depende muito de fatores que são difíceis de prever, como muito investimento. Isso é mais fácil em um mercado em crescimento do que em um mercado estável. Espero que o mercado continue a crescer, mas todas essas coisas estão amarradas.*

[1] Lei criada em 1965 por Gordon Moore, um dos fundadores da Intel.

Tomando a base desta afirmação, podemos supor que para que possamos aumentar a capacidade computacional, ou mudamos a arquitetura de hardware ou usamos a arquitetura atual de outra forma. Enquanto na primeira opção dependemos de novas idéias e novos conceitos, a segunda nos permite usar estruturas já definidas como as grades computacionais em que todo processamento pode ser dividido entre diversas máquinas, ao invés de uma só.

Partindo deste presuposto, admitimos a utilização de grades computacionais para processamento de larga escala e que necessite de grande capacidade computacional. Mas como fazer com que as máquinas consigam interagir entre si? E como garantir segurança nesse caso? Essas entre outras questões surgem automaticamente quando pensamos em soluções baseadas em grades computacionais. Neste trabalho propõe-se exatamente uma solução a algumas das questões relacionadas ao aspecto de segurança.

Se avançarmos em direção ao tema segurança, veremos que exatamente esse é um dos pontos mais vulneráveis de uma grade. Assim como ocorre em uma LAN[2] em que se procura proteger toda a estrutura de possíveis invasões, em uma grade isso também é um ponto a ser visto e repensado. Enquanto que em uma rede local, em geral, as possíveis vulnerabilidades se encontram em um único ponto de acesso externo, em uma grade por se tratar de estrutura distribuída geograficamente, os pontos de vulnerabilidade tornam-se maiores e mais difíceis de serem controlados, podendo comprometer não somente uma máquina que possa ser "invadida" e sim toda a estrutura da grade.

Em uma grade a idéia de segurança na inclusão de dispositivos ou usuários consiste em fornecer a estes permissão para o uso e essa "permissão" é fornecida, normalmente, pelo gerente de uma Autoridade Cerificadora (CA).

O procedimento de fornecer permissão de uso de recursos computacionais pode ser sumarizado como se segue: primeiro o usuário gera a sua chave privada e uma chave pública baseando-se em um certificado fornecido pela CA. A chave pública é enviada à unidade certificadora da grade para ser assinada pelo gerente de certi-

[2]Local Area Network - Uma rede de alcance local, geralmente em um ambiente fechado.

ficados. A chave assinada, ou seja, o certificado com o prazo de validade definido pela CA e assinado pela mesma é então enviado ao usuário que a requisitou. Um problema desse procedimento é que tanto os certificados de máquina quanto o certificados de usuários, bem como as chaves privadas associadas, ficam em poder do indivíduo que fez o pedido e dependente dos requisitos de segurança desse indivíduo, e assim, podendo ficar essas informações contidas nos certificados expostas a outros indivíduos, a princípio não autorizados. Outro problema é utilização de senhas não seguras, com poucos caracteres e por vezes até senhas somente numéricas. Segundo [OESCHSLIN06] uma senha numérica de seis dígitos de um sistema operacional como Windows, por exemplo, pode ser "quebrada" na média entre 13 e 101 segundos, ou seja, senhas ditas fracas (em geral senhas apenas compostas de números), podem comprometer a segurança de um sistema complexo como uma grade computacional. Nessa situação, uma pessoa mal intencionada pode, de posse de um certificado, quebrar a senha e se utilizar dos benefícios da grade ou até mesmo usá-la para ataques.

Uma solução para os problemas citados acima é a utilização de chaves temporárias. Quando há a necessidade de uso por uma entidade (homem ou máquina), pré-cadastrada, uma chave temporária seria passada ao solicitante através um sistema confiável para distribuição de chaves.

Outras soluções para a segurança das chaves não abordadas diretamente neste trabalho, mas citadas em alguns capítulos também envolvem a utilização de criptografia, exigência de senhas seguras e geração de certificados com tempo de duração curto. Essas medidas podem ser utilizadas para uma melhora do nível de segurança e integridade tanto em redes cabeadas quanto em redes sem fio. A utilização de Firewall[3] é sem dúvida uma necessidade. Podería-se com o uso de um firewall controlar quem pode e quem não pode usar uma grade, mas este não fornece uma solução definitiva na utilização de redes sem fio, pois estas não permitem um controle centralizado de acesso à rede já que dispositivos móveis como PDAs, Celulares, PCs ou LapTops, por exemplo, com rede sem fio tendem a gerar maior preocupação

[3] É o nome dado ao dispositivo de rede que tem por função regular o tráfego de rede entre redes distintas e impedir a transmissão de dados nocivos ou não autorizados de uma rede a outra.

na área de segurança, pois estes sistemas são pervasivos, trocam o endereço e mesmo o domínio com certa frequência e necessitam de mecanismos de segurança mais flexíveis e seguros que os disponíveis em sistemas de firewall em redes cabeadas, por exemplo.

1.1 O Problema

Em um ambiente de grade a idéia de fornecer permissão de uso para máquinas torna-se uma tarefa de difícil administração. Centralizar a autenticação usando certificados com chaves públicas e privadas através dos mecanismos convencionais pode ocasionar demora, pois existe a necessidade de assinatura das mesmas pelo gerente da Autoridade Certificadora em particular. Essa demora é particularmente problemática para dispositivos móveis. Como a tarefa de autenticação pode não ser automatizada, se a rede estiver com problemas (demora na entrega do email com a chave pública, por exemplo), ou se o responsável pela assinatura das chaves não estiver disponível no momento da requisição da assinatura, um certificado pode demorar muito tempo, o que pode comprometer o uso da grade.

Usualmente, quando se deseja interagir com a grade em um processo de visualização gráfica como proposto por este trabalho, por exemplo, é necessário a requisição de certificados de máquina e de usuário. O certificado de usuário pode ser requerido utilizando-se o serviço de solicitação de certificados temporários MyProxy (vide capítulo 5), mas o certificado de máquina tem que ser gerado localmente, em geral pelo administrador da máquina e enviado para o gerente de certificação para ser assinado pela entidade certificadora. Esse processo pode requerer tempo não tolerável por dispositivos móveis e/ou visitantes[4].

O fornecimento de certificados de máquinas dinâmicos de curta duração é uma proposta para a automatização desse serviço. Além de facilitar o uso da grade, essa proposta possibilita um aumento no nível de segurança, já que o certificado gerado

[4]Dispositivos, em geral móveis de usuários em visita a uma instituição, não fazendo parte das máquinas interligadas à grade.

dinamicamente tem um tempo curto para utilização. Uma vez expirado esse tempo é necessário que se faça uma nova requisição por um usuário válido, impedindo que esse certificado permaneça válido e seja utilizado por outras entidades não autorizadas.

1.2 Solução Proposta

A proposta deste trabalho é a criação de uma estrutura que forneça de uma maneira automatizada, um cadastro que permita a usuários da grade fazerem uma requisição de certificados de uso, sem abrir mão da segurança, disponibilizando não somente uma arquitetura funcional, mas principalmente segura tanto para redes móveis quanto para redes cabeadas.

Esta solução é para uma aplicação específica de fornecimento de certificados de curta duração para dispositivos que desejam ingressar em grades computacionais por um curto período de tempo, para interagir com a mesma para interação entre os processos submetidos. Também é desenvolvido para o trabalho com um middleware[5] específico chamado Globus Tookit, que é um dos middlewares mais utilizados atualmente em grades computacionais no mundo. A idéia principal, tema desta dissertação é a criação de um serviço que permita que um usuário, desde que com certificado de usuário válido, assinado e não expirado faça a requisição de um certificado de máquina para um dispositivo móvel ou visitante, para que esse dispositivo se associe à grade por um período curto de tempo, interagindo com seus trabalhos submetidos à grade.

O objetivo é criar um serviço onde usuários possam fazer de uma forma simples e funcional a requisição de um certificado de uso da grade. Após esta requisição, um usuário válido poderia, de posse de seu certificado temporário, requisitar também um certificado para seu dispositivo, principalmente para que este possa interagir diretamente com a grade (em um processo de renderização de imagens, por exemplo). É importante salientar que o uso de certificados temporários não exclui a idéia de

[5] Camada de software responsável por equalizar arquiteturas de grades computacionais independente do hardware utilizado

uma unidade certificadora, já que esta é quem forneceria certificados que ficariam em um repositório protegido pela segurança do sistema de arquivos local, para que sejam fornecidos a usuários e máquinas, quando estes forem requisitados e apesar de não suspender a autoridade certificadora, torna automatizada a tarefa desempenhada por esta.

1.3 Estrutura do Documento

O restante desta dissertação está organizada da seguinte forma: O Capítulo 2 trata do assunto grades computacionais e do middleware necessário para estas, o Capítulo ?? trata específicamente do middleware globus toolkit o qual é utilizado pela implementação deste trabalho. O Capítulo 4 trata diretamente de certificados digitais, de seu funcionamento, da necessidade de uso e da assinatura e criptografia dos mesmos, o Capítulo 5 trata de MyProxy, um servidor de certificados proxy temporários para usuários do qual foi tirada a idéia de geração de certificados temporários para máquinas. No capítulo 6 é tratado o capítulo principal desta dissertação, ou seja, o fornecimento dinâmico de certificados temporários para máquinas. No Capítulo 7 mostramos os trabalhos relacionados e no Capítulo 8 são mostradas as conclusões e trabalhos futuros seguidas dos anexos do trabalho.

Capítulo 2

Grade

Neste capítulo são apresentados os conceitos básicos para entendimento de uma grade. Na seção 2.1 são definidas grades computacionais, na Seção 2.1.1 será definido *middleware*, na seção 2.1.2 serão definidas as organizações virtuais. Na seção 2.1.3 é mostrada a arquitetura de uma grade e são definidas cada uma destas camadas da arquitetura. Na seção 2.1.4, Sistemas de Grade onde são mostrados os requisitos fundamentais das grades computacionais e alguns conceitos que se fazem necessários para o entendimento dos capítulos seguintes, por fim, na seção 2.1.5 é apresentada uma breve conclusão deste capítulo.

2.1 Grades Computacionais

Com o passar dos anos a informatização de diversas áreas de conhecimento vem se tornando um fato, junto a esse crescimento no uso de computadores vem a necessidade de cada vez mais capacidade computacional. Computadores com cada vez mais memória, processadores mais rápidos, mais quantidade de recurso de armazenamento e assim por diante, só que esses recursos apesar de disponíveis acarretam um custo monetário alto e contínuo de atualização.

Nem sempre convém comprar um supercomputador, além deste ter geralmente um custo elevado para aquisição, também pode ser útil por um curto período de

tempo. Se durante o desenvolvimento de uma aplicação for avaliada a necessidade de mais recursos computacionais, já que em geral supercomputadores são escaláveis somente até certo ponto, quando esta escalabilidade do supercomputador chegar ao fim seria necessária a aquisição de uma nova máquina e assim o custo elevado destes equipamentos pode comprometer um projeto.

A computação em grade vem para tentar minimizar os gastos com necessidade de maior capacidade computacional por se basear na idéia de utilização de recursos existentes para aumento das próprias capacidades computacionais. Segundo Foster:

> *"Uma Grade Computacional é uma infra-estrutura de hardware e software que provê acesso seguro, consistente, de forma dispersa e a custo baixo à potenciabilidade computacional máxima".*[FOSTER02]

Quando se pensa em uma grade, por muitas vezes a idéia que se tem é de um sistema heterogêneo, em que diversas máquinas processando uma determinada aplicação retornam em um tempo tolerável um resultado esperado, entretanto, a idéia de grade é um tanto mais complexa que isso, ou seja, supor uma grade como se fosse um cluster não é correto. Em um cluster as máquinas ficam, em geral, confinadas em um determinado ambiente e executam sistemas operacionais iguais ou equivalentes. Numa grade, os sistemas operacionais podem ser diferentes, podem existir diversas arquiteturas, diversos tipos de redes e até mesmo as máquinas podem estar distribuídas geograficamente.

Toda essa idéia de grade torna a estrutura de uma grade computacional algo de difícil administração, dependendo principalmente de sistemas de controle e protocolos específicos e quando se consegue unir esta idéia à necessidade, uma grade pode:

1. Coordenar recursos de diferentes finalidades (aplicações científicas, comerciais, desktop, etc.);
2. Usar interfaces e protocolos padronizados, abertos e de propósito geral;

3. Oferecer Qualidade de serviço(QoS) não triviais: segurança, autenticação, escalonamento de tarefas, disponibilidade, tempo de resposta, entre outras.

Em resumo, grade computacional é a união de diversos recursos computacionais em prol da resolução de um problema computável somente por supercomputadores, isto é, uma grade computacional é uma coleção de recursos interligados diretamente ou indiretamente, usufruindo e/ou oferecendo serviços de forma que se possa explorar a potenciabilidade destes. É válido lembrar que nem sempre a união de muitos recursos faz com que uma determinada aplicação tenha o efeito esperado, é necessário que a aplicação tenha seu código escrito para este tipo de arquitetura, ou seja, é necessário que se use computação distribuída de forma adequada.

Uma grade computacional é definida como um sistema que coordena recursos, que não estão submetidos a um controle centralizado, através de interfaces e protocolos de propósito geral, padronizados e abertos que disponibilizam qualidades de serviço não triviais de serem conseguidas. Uma grade busca possibilitar a existência das organizações virtuais, controlando o compartilhamento dos recursos entre os vários nós.

John MacCarty do MIT já previa em 1961 o que hoje temos como grade, ele cita:

> *"Se os computadores do tipo que eu imagino, se tornarem os computadores do futuro, então a computação poderá algum dia ser organizada como um serviço público, assim como a telefonia o é... Esse serviço poderá se tornar a base da nova e importante indústria"*

Governos, empresas e organizações de pesquisa estão trabalhando para criar redes de super-computação para disponibilizar estes recursos computacionais agregados de um grupo para qualquer computador conectado. As entidades que nos últimos anos investiram em clusters de alta performance estão dando o próximo passo para a futura computação em grades computacionais (*grid computing*[1]), e os pesquisadores

[1] O termo *grid computing* foi criado a partir de uma proposta computacional que se baseia na similaridade às malhas de energia elétrica

estão trabalhando para criar uma interface padronizada de Web para que diversas pessoas em várias áreas distintas possam utilizar a super-computação em grade como um serviço público tradicional, como telefone, luz ou internet, por exemplo.

Apesar de toda potencialidade que a grade pode prover, alguns cuidados devem ser tomados, por exemplo, que tipo de software deve ser usado ou como será a comunicação entre as diversas máquinas da grade, dentre esses cuidados podemos citar: *Middleware*, Organização Virtual, Computação em Grade e Sistemas de Grade, os quais serão discriminados a seguir.

2.1.1 Middleware

Middleware é um conjunto de serviços especializados, compartilhados pelas aplicações e pelos usuários, que tem componentes básicos capazes de auxiliar todos os aplicativos. Em nosso contexto, *middleware* é uma camada que facilita o desenvolvimento de aplicações abertas em sistemas distribuídos. Já que a grade pode possuir diversas arquiteturas e diversos sistemas operacionais, sem o middleware seria difícil, senão impossível usar a idéia de processamento paralelo em grade.

A função do *middleware* é a de possibilitar que as aplicações possam ser escritas de modo o mais independente possível do hardware e do sistema operacional, permitindo assim que um mesmo código de aplicação possa ser carregado e executado em diferentes equipamentos receptores. Em resumo, o middleware é um software capaz de interpretar os aplicativos e traduzí-los na linguagem do sistema operacional em que ele reside.

Existem dois tipos de *middleware* que podem ser usados em uma grade computacional, *middleware* baseado em orientação objeto, que suporta infra-estrutura de orientação objeto como, por exemplo, herança[2] e polimorfismo[3] e *middleware* baseado em serviços que suporta containers de serviço com infra-estrutura e aplicações coexistindo.

[2]mecanismo importante quando um grupo de classes apresenta a mesma interface, mas a implementação interna dos métodos é diferente.

[3]característica em orientação a objetos que usa a hierarquia de objetos.

2.1.2 Organizações Virtuais

Uma organização virtual é uma entidade, geralmente geograficamente distribuída que pode se beneficiar ou participar de uma grade. O foco principal é compartilhamento de recursos. O compartilhamento entre os participantes de uma organização virtual diz respeito ao acesso direto a computadores, softwares, dados e outros recursos necessários para a solução de algum problema computacional.

Organizações virtuais podem ser desenvolvidas nas seguintes situações:

1. Quando uma organização singular se divide em várias organizações distribuídas;

2. Quando o trabalho é distribuído entre várias organizações diferentes;

3. Quando a organização virtual é "oportunista", isto é, quando ela é desenvolvida apenas para aproveitar uma oportunidade, sem senso de permanência.

"Em qualquer situação, um ambiente informacional cooperativo é condição para o sucesso da organização virtual" como define Goranson[GORANSON00].

Para ficar mais clara a idéia de organização virtual, a figura 2.1 mostra a disposição de 3 entidades, das quais duas organizações virtuais podem ser vistas. No exemplo, as 3 entidades compartilham seus discos. Entre as entidades 1 e 2 a linha tracejada mostra a formação de uma organização virtual com 4 discos, e entre as entidades 2 e 3 uma nova organização com 3 discos. Vale lembrar que as organizações virtuais podem compartilhar outros tipos de recursos, bem como podem existir as entidades oportunistas que simplesmente utilizam os recursos sem fornecerem nenhum tipo de cooperativismo.

2.1.3 Arquitetura da Grade

Como pode ser visto na figura 2.2 a arquitetura da grade é composta de quatro camadas, sendo que estas camadas podem ser comparadas ao modelo da arquitetura

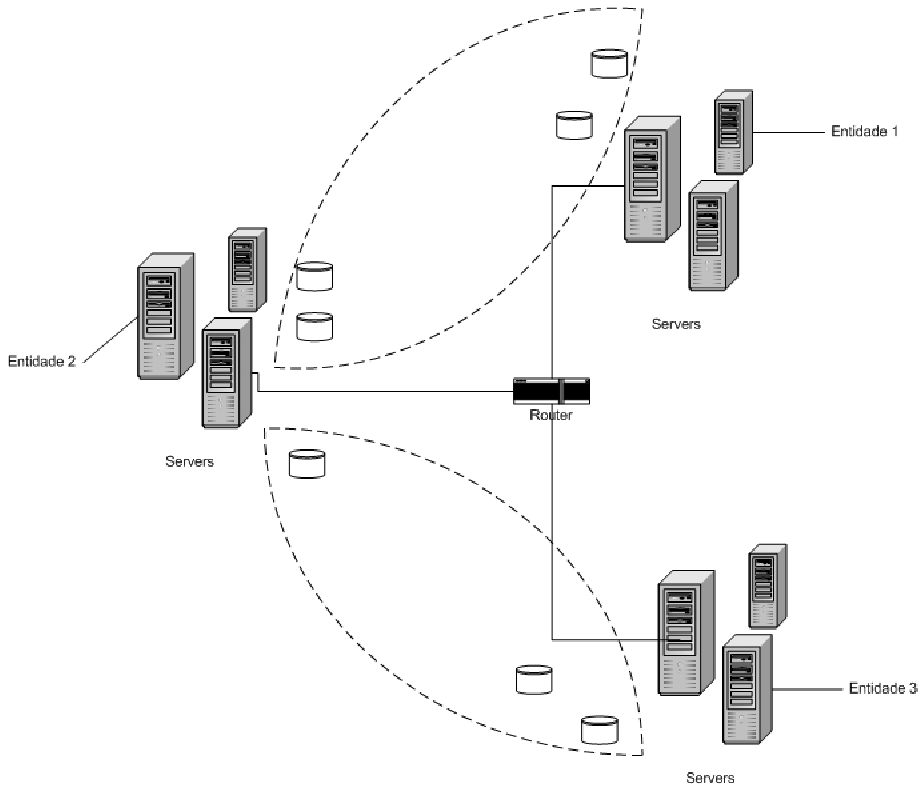

Figura 2.1: Entidades formando duas organizações diferentes.

internet, na qual a camada *fabric* corresponderia à camada de enlace, serviços da grade corresponderia às camadas de rede e transporte e por fim, as camadas superiores seriam similares à camada de aplicação do modelo internet. Já que podemos definir os modelos da arquitetura da grade e da arquitetura internet como modelos equivalentes, isto é, podemos fazer um mapeamento entre as duas arquiteturas, podemos também utilizar a internet para uso da grade já que a idéia desta é utilização de recursos distribuídos geograficamente e a internet pode ser o meio para que isso aconteça.

As camadas da arquitetura da grade podem ser descritas como a seguir:

1. **Fabric:** Nesta camada são definidos os recursos que se deseja acessar. Nesta camada devem-se definir os recursos que serão acessíveis e para cada recurso deve-se fornecer e implementar serviços que garantam QoS.

2. **Serviços da Grade:** Nesta camada são definidos os protocolos para comunicação e autenticação via rede. Os protocolos de comunicação devem possi-

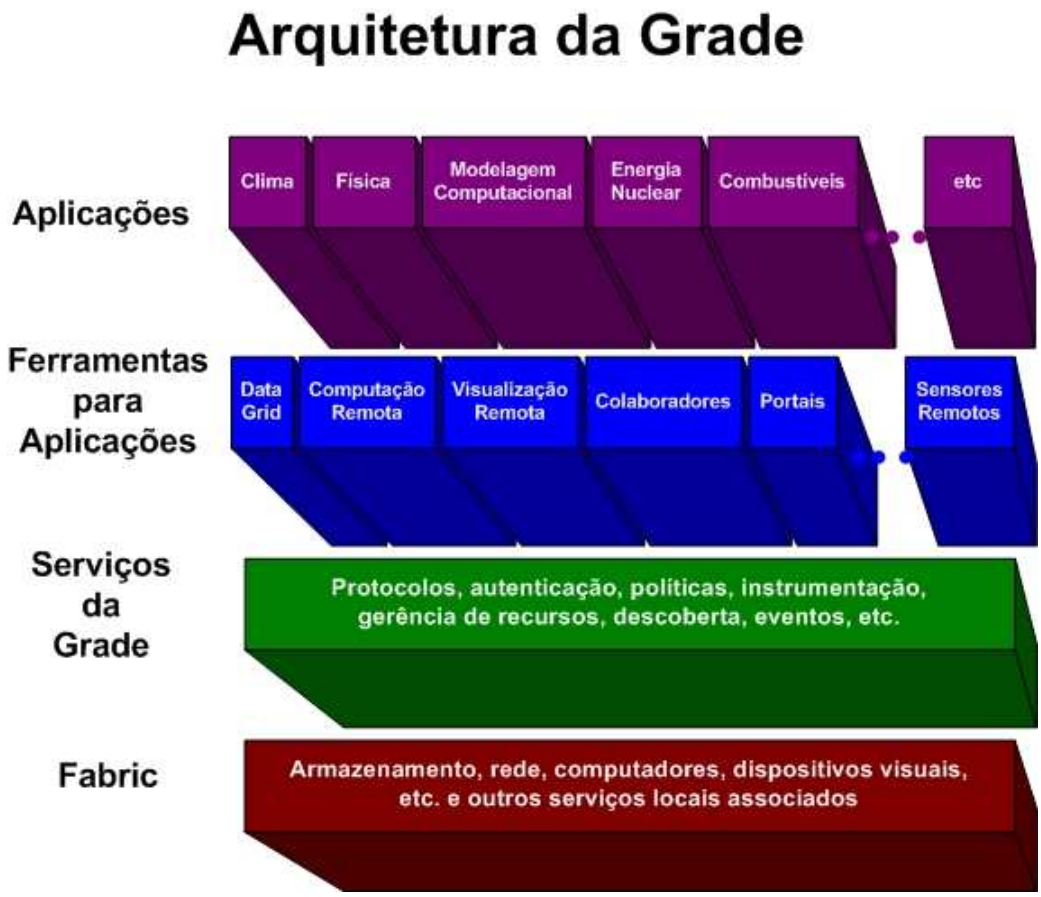

Figura 2.2: Arquitetura da Grade.

bilitar a troca de dados entre os recursos da camada *Fabric*. Os protocolos já existentes da arquitetura internet podem ser utilizados, como por exemplo: IP, UDP, TCP, etc., mas isso não garante que estes protocolos sejam ideais para essa arquitetura, sendo necessário a utilização de outros protocolos da grade para problemas específicos.

3. **Ferramentas para aplicações:** Nesta camada são definidas segurança, conectividade com aplicações, monitoração e gerência. Também podem ser vistos protocolos e serviços destinados à realização de um controle mais global da grade e a captura das interações realizadas em um conjunto de recursos.

4. **Aplicações:** Camada onde estão as aplicações das organizações virtuais que são controladas pela grade.

2.1.4 Sistemas de Grade

Em uma grade, do objetivo final pode depender o sistema a ser trabalhado. Dentre sistemas de grade, alguns tendem a dificultar a criação de padrões que possam ser usados em todos os casos sem prejudicar o desempenho. Nesta seção são citados alguns sistemas e características de cada um de forma a oferecer uma visão mais abrangente de quais recursos podem ser disponibilizados em uma grade.

1. **Sistemas de Grades Computacionais:** É uma extensão dos sistemas distribuídos e grandes sistemas paralelos. Neste tipo de sistema existe um conjunto de computadores disponíveis e um conjunto de usuários que usufruem destes através de um único computador, podendo solicitar a execução de um processo que necessite de mais de um computador para ser executado. A idéia principal deste tipo de sistema é fornecer computação de alto desempenho a baixo custo.

2. **Sistemas de Grades de Acesso:** Possuem o objetivo de construir localmente um ambiente, no qual usuários possam usufruir dos serviços e recursos específicos da grade dentro de um sistema distribuído, como se fosse um ambiente virtual.

3. **Sistemas de Grade de dados:** O objetivo é permitir que grande quantidade de dados seja movida entre repositórios, como se fossem pequenos arquivos e desta forma, os usuários conseguiriam acessar estes dados como se estivessem em um único lugar.

4. **Sistemas de Grades Centrada em Dados:** O objetivo destas é permitir computações sobre grandes repositórios de dados distribuídos que não possam ser armazenados em um único local. A idéia principal é executar os processos onde os dados estão e não trazer os dados para o lugar onde estão os processos.

2.1.5 Conclusão do Capítulo

A computação em grade veio para suprir a necessidade cada vez maior de capacidade computacional, como foi visto neste capítulo, ao contrário de um cluster

em que máquinas ficam confinadas em um ambiente homogêneo local, a grade pode se utilizar dos recursos da Internet formada por sistemas totalmente heterogêneos, fracamente acoplados e geograficamente distribuídos. Podemos imaginar neste momento quantos computadores estão sendo sub-utilizados e conectados à internet. Estes poderiam ser usados para computação de auto desempenho em uma grade para o bem comum (pesquisa de doenças ou meteorologia, por exemplo).

Na próxima seção será mostrado o Globus um dos *Middlewares* mais usados atualmente em ambientes de grade. Este middleware tem grande aceitação entre as comunidades de pesquisa, sendo este um dos motivos por ter sido escolhido para os testes objeto deste trabalho. O bom entendimento deste *middleware* é de suma importância para o entendimento de outros capítulos. Por esse motivo Globus ganhou um capítulo exclusivo que será visto a seguir.

Capítulo 3

Globus

Neste capítulo é apresentado um dos middlewares descritos no capítulo anterior chamado Globus. A opção por este middleware se relaciona diretamente à quantidade de material e suporte disponível na internet. Na seção 3.1 é descrita a história do Globus junto a uma introdução que ajudará a entender melhor os conceitos das seções seguintes. Na seção 3.1.1 é mostrado o paradigma para a formação da equipe de desenvolvedores do Globus até chegar à idéia do Globus Tookit. Na seção 3.1.2 discrimina-se o Globus Tookit, suas funcionalidades, seus conceitos e sua estrutura. Na seção 3.1.3 é mostrada a arquitetura do GT4 e na 3.1.4 define-se um dos componentes da arquitetura que trata da importante distribuição e gerência dos recursos de dados. Na seção 3.1.5 trata-se da segurança no globus e, mais especificamente, na seção 3.1.6 descreve-se o GSI, importante serviço de segurança do Globus e base para que este exista. Na seção 3.1.7 Define-se MDS, um importante mecanismo de obtenção de informação sobre a grade, seus recursos e disponibilidades. Por fim será mostrada uma breve conclusão deste capítulo na seção 3.2

3.1 O Middleware Globus

O Globus é um projeto de pesquisa e desenvolvimento de middleware para aplicações em grades computacionais. Este projeto é desenvolvido por organizações e

indivíduos que buscam desenvolver tecnologias fundamentais para o funcionamento de grades computacionais. Essa comunidade de desenvolvedores é chamada "Globus Alliance" e o principal foco desta comunidade são as aplicações científicas e de engenharia da computação [FOSTER97]. O projeto Globus gerou o middleware Globus Toolkit. Através das bibliotecas contidas no Globus Toolkit são implementados módulos funcionais de gerência de recursos, gerência de dados e gerência de acesso que controlam o uso da grade computacional.

A primeira versão do Globus Toolkit (versão 1.0) foi lançada em 1998, o que mostra que o estudo sobre grades computacionais e suas aplicações não é recente. Contudo, o aumento da quantidade de desenvolvedores e conseqüentemente aumento da produção de módulos é mais recente. Em 2002 a versão 2.0 foi lançada e somente em 2005 sairam as versões 3 e 4. Atualmente a versão estável é a 4.0.2[1] e esta foi a escolhida para o desenvolvimento de mais esse módulo, objeto deste trabalho que é suportado pelo Globus Tookit.

A função do Globus Toolkit é permitir que diversas pessoas utilizem alta capacidade computacional, compartilhamento de dados ou outros recursos indisponíveis em máquinas comuns, mas facilmente encontradas em sistemas de grade. A idéia de distribuição geográfica da grade não prejudica o desempenho nem agride a autonomia de uma instituição e isso é um dos focos principais do globus.

O projeto Globus teve tanta aceitação junto às comunidades de desenvolvedores que em 2002 ganhou o R&D 100[2][ReD100]. Ganhar um R&D 100 fornece a uma marca um título de excelência, como a prova de que o produto é uma das idéias mais inovadoras do ano.

[1] referência da versão: junho de 2006

[2] "Oscar da invenção" da Tribuna de Chicago que por 44 anos têm concedido prestigio e ajudado a indústrias, governo e instituições acadêmicas a fornecer o impulso inicial muito importante para um novo produto.

3.1.1 Desenvolvedores Globus

A idéia da criação de uma comunidade onde diversos pesquisadores, desenvolvedores, estudantes e usuários pudessem trocar experiências foi uma das alavancas da criação do Globus Alliance. A infra-estrutura suportada por essa comunidade inclui: repositório de códigos, lista de email, acompanhamento de problemas e assim por diante, tudo isso disponível em globus.org.

Diversas instituições promovem o desenvolvimento do Globus Tookit a fim de disseminar seu uso tanto dentro de unidades de ensino, pesquisa e desenvolvimento quanto para aplicações comerciais. Dentre essas instituições, algumas que podemos citar são:

1. Laboratório Nacional de Argonne, Universidade de Chicago;
2. EPCC, Universidade de Edimburgo;
3. Centro Nacional para Aplicações de Supercomputação, NCSA;
4. Universidade do Norte de Illinois, Laboratório de Computação de Alto Desempenho;
5. Instituto de Tecnologia Real, Sweden;
6. Univa Corporation
7. Instituto de Ciência da Informação da Universidade do Sul da Califórnia.

Outras instituições podem promover o desenvolvimento do Globus Toolkit através da criação de novos módulos ou correção de possíveis bug's. Por ser um sistema Open Source[3] é possível baixar todo o código e fazer as modificações ou inclusões de módulos que forem necessários à aplicação específica, assim como MyProxy capítulo 5 desta dissertação ou mesmo o objeto principal deste trabalho descrito no capítulo 6.

[3]Sistema de código aberto onde é possível visualisar todo o código fonte

3.1.2 Globus Toolkit

Globus Toolkit é um conjunto de serviços e bibliotecas de software para suportar aplicações da grade. O Toolkit inclui software para segurança, infra-estrutura de informação, gerenciamento de recursos e de dados, comunicação, detecção de falhas e portabilidade. É desenvolvido como um pacote com todas as funcionalidades, que podem ser usadas juntas ou separadas, para o desenvolvimento de aplicações.

O globus Tookit foi concebido para remover obstáculos que impedem a colaboração entre instituições, seus serviços, protocolos e relações permitindo que um usuário execute suas aplicações como se estivesse em uma rede local. O software Globus tem em foco permitir que aplicações utilizem recursos distribuídos, como: computadores, armazenamento, dados, serviços, redes, ou sensores.

Inicialmente, o trabalho com Globus era motivado pelas demandas das organizações virtuais. Mais recentemente, as aplicações comerciais tornaram-se cada vez mais importantes, por um fato simples: o de que o comércio e a ciência freqüentemente, mas não sempre, têm interesses similares.

O software fornece uma variedade de componentes e potencialidades, incluindo os seguintes:

1. Ferramentas para construção de novos serviços Web, em Java, Perl ou Pynthon;

2. Boa infra-estrutura de segurança;

3. API's[4] implementadas em diversas linguagens e acesso a diversos serviços e componentes através de linha de comando;

4. Documentação detalhada destes vários componentes, suas relações, e como podem ser usados para construir aplicações.

[4]Application Programming Interface - conjunto de rotinas e padrões estabelecidos por um software, isto é, programas que não querem envolver-se em detalhes da implementação do software, mas apenas usar seus serviços

3.1.3 Arquitetura do Globus

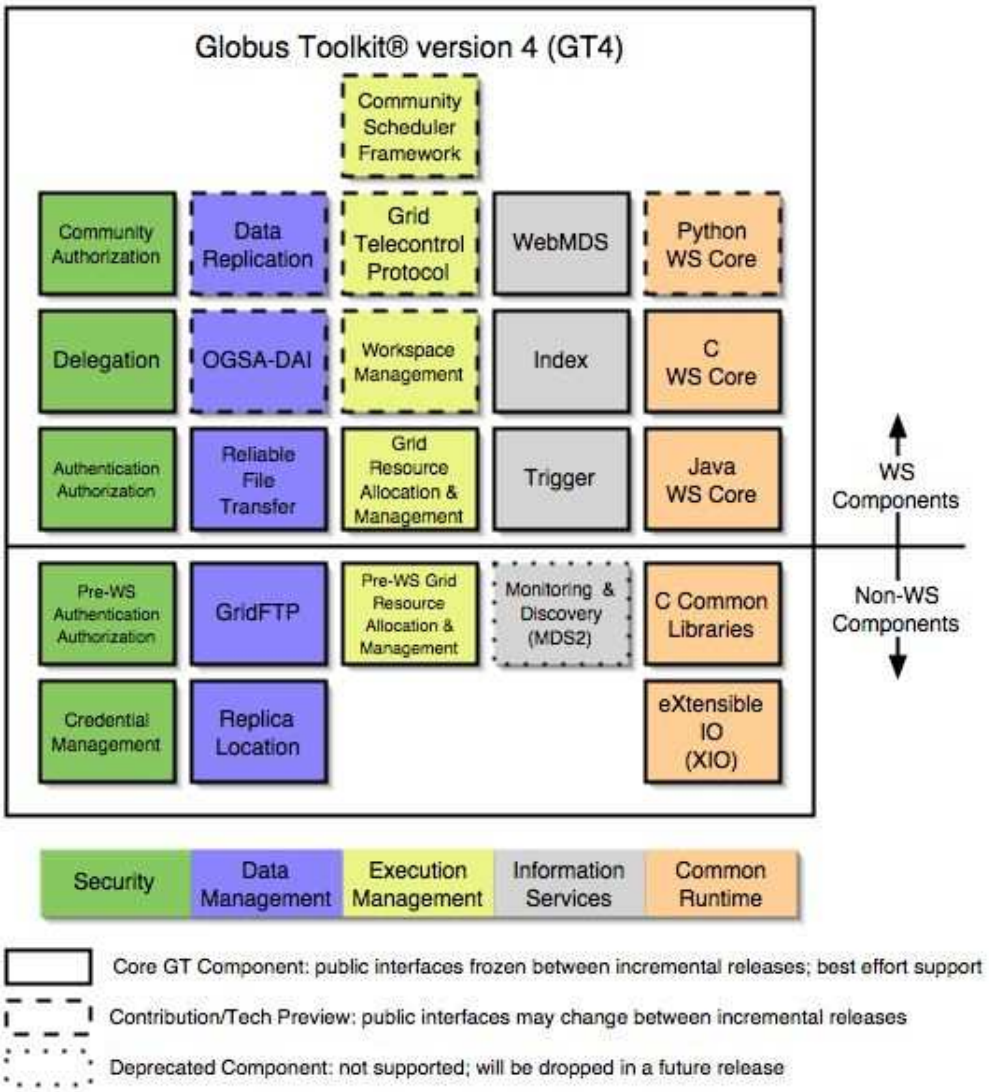

Figura 3.1: Arquitetura do Globus

Como mostrado na Figura 3.1[GLOBUS06], GT4[5] compreende um conjunto de serviços e bibliotecas associadas a estes que provêm componentes que utilizam e componentes que não utilizam a tecnologia de Web Services. Dentre os serviços pré-definidos no GT4, destacam-se os seguintes:

1. **Serviços de Informação:** responsável por reunir e disponibilizar informações sobre os recursos disponíveis na grade, denominado MDS (Monitoring and

[5]Globus Toolkit 4

Discovery Service). No GT4, este componente é um Grid Service (WebMDS). Tal serviço define uma série de entidades e protocolos que cooperam de maneira a prover informações sobre o estado da Grade.

2. **Serviço de Segurança:** denomina-se GSI (Globus Security Infrastructure) seção 3.1.6. O GSI trata das questões de segurança do Globus, tais como:

   (a) **Autenticação única:** o usuário se autentica uma única vez no sistema, o que permite a utilização de diversos recursos sem a necessidade de novas autenticações;

   (b) **Comunicação segura:** impede que uma entidade não autorizada obtenha acesso às comunicações das aplicações do usuário;

   (c) **Delegação:** é possível delegar parte das permissões de um usuário a uma outra entidade, por exemplo, uma aplicação.

   Dessa maneira, é possível que uma aplicação requisite mais recursos em nome do usuário que a executou.

3. **Gerenciamento e Alocação de Recursos:** permite que usuários-clientes possam submeter programas para serem executados em diversos recursos computacionais da grade. Além disso, estes programas podem ser monitorados ou cancelados. Este componente denomina-se GRAM (Grid Resource Allocation and Management) ou mais especificamente WS GRAM para Web Services no GT4.

4. **Acesso e Gerenciamento de Dados:** são utilizados para viabilizar um uso mais simples dos protocolos disponíveis. A recuperação de falhas, escolha do protocolo adequado, e o gerenciamento da cópia são realizados pelo serviço e o cliente final simplesmente realiza uma solicitação.

3.1.4 Acesso e Gerência de dados do Globus

O GT4 tem algumas funcionalidades e padrões que permitem uma gerência centralizada dos recursos distribuídos, assim como alguns protocolos criados para fins

específicos que permitem acesso a arquivos distribuídos. Dentre estes protocolos, podemos destacar os seguintes:

1. **GridFTP:** define extensões ao protocolo FTP permitindo a transferência de dados de uma forma segura, confiável e com um alto desempenho. Uma implementação do serviço está disponível no Globus Toolkit 4 e é usado pelo "Reliable File Transfer (RTF) Service" no Globus. O FTP foi escolhido como base para o sistema por várias razões, dentre elas:

   (a) A existência de canais de controle separados dos canais de dados;

   (b) Pela ampla difusão do protocolo;

   (c) Pela capacidade de definição de um conjunto de extensões previsto na especificação do protocolo

   . As principais funcionalidades do GridFTP são:

   - **Controle de Transferência Externo:** Um cliente, devidamente autenticado, é capaz de mediar a transferência de dados entre dois servidores de dados. Desta forma, o cliente realiza operações remotamente e é capaz de iniciar, monitorar e controlar a transferência de dados.
   - **Autenticação, Integridade e Confidencialidade:** Utiliza o "Generic Secutiry Service" e suporta a definição de vários níveis de confidencialidade e integridade dos dados. A autenticação também é usada para validar as partes envolvidas na comunicação, ou seja, o cliente e os servidores de dados.
   - **Transferência Particionada:** Essa técnica visa fazer um uso otimizado da rede, onde, mesmo que uma máquina não consiga utilizar toda a capacidade da rede, um conjunto de máquinas faz o uso compartilhado desta rede, enviando em vários canais e resultando em um alto desempenho na operação como um todo.
   - **Transferência Paralela:** Utiliza vários canais TCP paralelos entre a mesma fonte e destino, de forma a obter uma taxa agregada de transfe-

rência maior que a taxa de um único canal. Essa técnica pode ser usada junto à transferência particionada.

- **Transmissão de Dados Parciais:** Além da função de resume existente no FTP, o GridFTP também permite a transferência de regiões arbitrárias do arquivo. Permitindo que apenas um determinado conjunto de informações seja transferido. Devido aos erros que podem acontecer na transferência, a continuação do trabalho de forma otimizada é importante.
- **Negociação automática do buffer TCP:** O uso de um tamanho de buffer adequado pode aumentar o desempenho da comunicação em redes de longa distância.

2. **Reliable File Transfer (RFT) Service:** Embora o GridFTP ofereça suporte para a recuperação de falhas, o controle da operação é feito pelo cliente da transferência e uma falha no cliente pode resultar em uma operação parcialmente concluída entre os servidores. Caso o cliente não possua um mecanismo de persistência para a continuação da operação, os dados permanecerão inconsistentes e a operação como um todo terá falhado. Além disso, o canal de controle deve permanecer aberto durante toda a transferência, o que dificulta clientes móveis, ou com uma conectividade instável [SANTOS05]. Nesse sentido, a criação de um serviço de transferência de arquivos é necessária, onde clientes realizam a solicitação e o servidor é responsável por manter o canal de controle e o estado das transferências.

   O Reliable File Transfer Service atua como um proxy, permitindo que os usuários solicitem as operações de transferência através do recebimento de descrições do trabalho a ser realizado. Diversas operações podem ser realizadas pelo serviço, mapeando as solicitações em comandos a serem executados pelo RFT usando o GridFTP. O RFT é responsável pelo gerenciamento dessas solicitações, representando o cliente.

3. **Global Access to Secondary Storage (GASS) Service:** O GASS define um espaço de nomes global através de URLs e permite que as aplicações aces-

sem os dados usando as interfaces padrão de entrada e saída através da cópia dos dados para a área local (staging). O objetivo do GASS não é disponibilizar um sistema de arquivos distribuído de propósito geral, e sim funcionalidades para o uso comum em sistemas de computação em grade, como transferência de executáveis e leitura de arquivos de configuração. Os padrões de acesso aos dados considerados são:

(a) acesso somente de leitura a um arquivo;

(b) escrita compartilhada de um arquivo, sem controle de concorrência;

(c) acesso para concatenação de informações, como em arquivos de log;

(d) acesso aleatório de leitura e escrita, também sem controle de concorrência.

Dessa forma, as operações podem ser realizadas sem levar em consideração outras aplicações que estejam acessando o arquivo. A abertura e o fechamento dos arquivos são feitos através de funções especiais, como: globus_gass_fopen e globus_gass_fclose [BESTER04]. Essas funções ativam o mecanismo de cache do GASS e de acordo com a política estabelecida pelo administrador da grade realiza a cópia do arquivo remoto.

4. **Replica Location Service (RLS):** a utilização de réplicas na grade reduz a latência no acesso aos dados, melhorando o desempenho das aplicações distribuídas. O serviço de localização de réplicas mantém um mapeamento entre nomes lógicos e nomes de arquivos reais armazenados na grade. O serviço visa substituir o catálogo global de réplicas existente nas versões anteriores do Globus Toolkit. A versão distribuída do serviço provê um maior desempenho e escalabilidade.

> *A semântica da palavra "réplica" deve ser observada atentamente, visto que na grande maioria dos casos, não existe um controle de versões mantendo a integridade dos arquivos, garantindo que todos eles possuem o mesmo conteúdo.*[SANTOS05]

3.1.5 Segurança no Globus

Como visto no capítulo 2, Grades Computacionais são mecanismos de computação de alto desempenho, distribuídos geograficamente e por esse motivo frágeis no quesito segurança. Da comunicação entre diversas entidades depende o bom funcionamento da Grade e isso inclui uma política de segurança que permita troca de arquivos, processamento ou mesmo informações de recursos.

Em uma estrutura de redes, deve-se atentar aos objetivos da segurança, ou seja, tem-se que criar uma política de acesso que não comprometa a grade. É necessário que se conheça e utilize as auditorias e conseqüentemente entenda-se os seguintes mecanismos como definem Carrião [CARRIAO01]:

1. **Autenticação**

   A autenticação é o processo de estabelecer a validade de uma identidade reivindicada. A autenticação é o primeiro passo na segurança de um sistema computacional; junto à confidencialidade e à integridade, consiste num dos pilares da segurança.

2. **Não Repudiação**

   A Não Repudiação consiste em obter provas (ou fortes indícios) de ações realizadas no passado de forma que um indivíduo não possa negar ações que tenha realizado no sistema.

3. **Confidencialidade**

   A confidencialidade impede que os dados sejam lidos ou copiados por usuários que não possuem o direito de fazê-lo.

4. **Integridade de Dados**

   A integridade de dados protege a informação de ser removida ou alterada sem a autorização do dono.

5. **Disponibilidade**

   A disponibilidade é a proteção dos serviços para que eles não sejam degradados

ou fiquem indisponíveis sem autorização. Isto implica em dados e sistemas prontamente disponíveis e confiáveis.

6. **Controle**

   Um sistema computacional pode possuir diversos recursos. O controle permite que somente usuários conhecidos e que têm direitos de acesso em períodos determinados possam, devidamente, dispor dos recursos.

7. **Prevenção**

   A prevenção é um dos elementos fundamentais em todo sistema de segurança [GARFINKEL96]. José Ribamar [PINHEIRO05] cita que a segurança de computadores consiste em uma série de soluções técnicas para problemas não técnicos. Pode-se gastar grandes quantias de dinheiro, tempo e esforço em sistemas de segurança sem nunca resolver problemas como os defeitos desconhecidos nos softwares (bug's) ou funcionários maliciosos. Assim sendo, precaver-se em relação a possíveis problemas é uma boa prática de segurança.

8. **Auditoria**

   Ainda não se conhece um sistema de segurança perfeito, sempre é possível que usuários não autorizados possam tentar acessar o sistema, ou usuários legítimos tenham efetuado ações erradas ou até mesmo atos maliciosos possam ser praticados. Nesses casos é necessário determinar o que aconteceu, quando, quem foi o responsável e o que foi afetado pela ação. A auditoria deve ser um registro incorruptível de principais eventos de segurança. Através da auditoria podemos nos resguardar das ações dos usuários e até mesmo utilizar mecanismos que impossibilitem que os usuários neguem os seus atos ao utilizarem o sistema (não repudiação)[CARRIAO01].

Globus Toolkit possui uma estrutura bem definida de segurança, para isso utiliza o GSI (*Globus Security infrastructure*)[FOSTER98]. GSI permite uma autenticação única do usuário na grade. A partir desta autenticação, o GRAM (*Globus Resource Allocation Manager*) verifica se o usuário pode executar no recurso em questão. Caso o usuário tenha o acesso permitido, é criado um Job Manager, que é responsável por iniciar e monitorar a tarefa submetida. As informações sobre o estado da tarefa

e do recurso são constantemente reportados ao serviço de informação e diretório do Globus, o *Monitoring and Discovery Service* (MDS).

3.1.6 GSI - Grid Security Infrastructure

O Globus Security Infrastructure é uma infra-estrutura que foi construída e melhorada tendo como principal objetivo fazer com que os serviços disponibilizados pela grade, e que estiverem disponíveis na rede, não precisassem de privilégios locais especiais para executarem, reduzindo assim o risco de ataques. Foster [FOSTER98] define GSI como uma implementação da arquitetura de segurança baseada no OGSA[6] (Open Grid Services Architecture).

Para disponibilizar credenciais e fazer autenticação o GSI utiliza certificados digitais X.509 para cuidar da identificação dos usuários e dos serviços disponibilizados pelas grades. Os acessos aos recursos compartilhados são realizados através de um arquivo chamado grid-mapfile. Este arquivo contém uma lista fazendo o mapeamento dos nomes de usuários e serviços da grade para nomes de usuários locais nas máquinas que estão com recursos compartilhados. Dessa forma, um usuário ou um dado serviço, acessará um recurso compartilhado através de uma conta local da máquina.

Além da autenticação X.509, presente também nas versões anteriores do Globus Toolkit, a versão 4 permite a autenticação de usuários utilizando identificador de usuário e senha. Porém quando este método é usado perde-se a possibilidade de usar mecanismos de segurança que incluem confidencialidade e integridade de dados. Para permitir a delegação dinâmica o GSI estende o conceito de proxy do X.509 que permite ao usuário atribuir uma nova identidade X.509 ao usuário e então delegar alguns de seus atributos de segurança a esta nova identidade. Este mecanismo permite a criação de novas identidades e credenciais sem a intervenção do administrador da rede.

[6] É a definição de uma arquitetura de serviços básicos para a construção de uma infra-estrutura de Grades de serviços. Define mecanismos, padrão para criação, nomeação e descoberta de serviços da grade.

A vantagem de se estender o padrão X.509 é a possibilidade de utilização de bibliotecas já existentes com uma modificação bem pequena já que o GSI disponibiliza um conjunto de ferramentas e bibliotecas que possibilitam o acesso seguro de aplicações e usuários aos recursos de uma grade computacional [GLOBUS06] além de oferecer duas formas de autenticação, a única e a mútua. Na autenticação única, o usuário autentica-se apenas uma única vez para poder utilizar os recursos da grade, não importando o domínio administrativo em que ele se encontra.

Na autenticação mútua, existe a idéia de troca de certificados entre os elementos da grade e para garantir a segurança na comunicação, o GSI implementa um conceito de assinaturas digitais, responsáveis pela autenticidade e integridade dos dados de origem.

A figura 3.2 apresenta a arquitetura do GSI, onde pode-se verificar que esta arquitetura é composta de cinco camadas, a camada superior (primeira camada) é responsável pela autorização, a segunda camada é responsável pela Delegação, na terceira camada é feita a Autenticação, na quarta é feita a proteção de mensagens e na última camada (SOAP) é feita a formatação de mensagens.

Figura 3.2: Arquitetura do GSI.

Para determinar as permissões dos usuários, através do GSI, a identidade Globus

do solicitante do recurso é mapeada para um usuário local e isso é feito na camada superior como pode ser visto na Figura 3.2 através do grid-mapfile. Por exemplo, um recurso poderia mapear:
"/O=Grid/OU=LNCC/OU=ComCiDis-CA/OU=lncc.br/CN=Globus User" para globus. Há de se notar que a identidade global será mapeada para diferentes usuários em recursos que estejam em diferentes domínios.

3.1.7 MDS - Monitoring and Discovery Service

O MDS é responsável por prover informações estáticas e dinâmicas sobre os recursos existentes dentro do ambiente da grade. Esse serviço está baseado em um servidor LDAP (Lightweight Directory Access Protocol) capaz de armazenar informações sobre potência e disponibilidade dos recursos[CZAJKOWSKI01].

Na versão 4 do Globus Toolkit este serviço foi transformado em um Web Service, o que, apesar de tê-lo transformado em um serviço mais estável, fez com que seu funcionamento dependesse diretamente da existência de um banco de dados, em geral, Postgresql[7].

O MDS é um serviço tão útil que este pode ser usado, por exemplo, por um escalonador de tarefas para definir que recursos utilizar por cada um dos processos a serem submetidos à grade.

3.2 Conclusão do Capítulo

Apesar de resolver satisfatoriamente diversas questões do aspecto operacional, o Globus não ataca o problema do grid do ponto de vista gerencial. Utilizando o mecanismo de acesso hoje disponível, são necessárias negociações com os donos de recursos para obter o acesso a estes e há necessidade do mapeamento dos clientes para usuários locais, fatos que limitam a escalabilidade deste mecanismo.

[7]Sofisticado sistema gerenciador de banco de dados relacional e orientado a objetos, de livre distribuição e código-fonte aberto. Desenvolvido na Universidade da Califórnia apartir de 1985

No próximo capítulo serão mostrados conceitos de autenticações, utilização de chaves públicas e privadas e dos certificados X.509 através da Autoridade Certificadora (AC), ou seja, a base da segurança na grade e neste caso específico, da utilização do GSI no Globus.

Capítulo 4

Certificados

Quando pensamos em uma grade computacional, pensar em segurança torna-se uma meta a ser alcançada através da definição de políticas de uso e de acesso, por exemplo, definir segurança no entanto é um ponto complexo e difícil de ser alcançado. Definidas as políticas a serem tomadas, deve-se buscar sistemas que permitam que estas sejam feitas. Neste capítulo serão mostrados os serviços disponíveis no âmbito da grade que permitem ao sistema como um todo dispor de certa quantidade de segurança.

Na seção 4.1 trataremos do uso de certificados digitais, na seção 4.2 definiremos Criptografia e a importância da utilização desta, na seção 4.2.1 trataremos da criptografia com chaves simétricas e na 4.2.2 com chaves assimétricas. Na seção 4.3 definiremos segurança no uso de certificados digitais tratando dos sistemas e arquiteturas desta segurança dentro de estruturas de CA's, SPKI/SDSI e certificados X.509 nas seções seguintes. Por fim, na seção 4.4 trataremos do uso dos certificados, seguido na seção 4.5 da conclusão deste capítulo.

4.1 Certificados Digitais

Implementados hoje em dia em diversas instituições, estes são muito comuns quando se pensa em garantia de identidade, em outras palavras, podemos considerar

que certificado digital é uma carteira de identidade virtual, já que em um certificado digital há informações pessoais ou institucionais, mas a principal informação presente em um certificado é a sua chave pública descrita na seção 4.2.2.

Em se tratando de certificados digitais, vale lembrar que estes precisam ser assinados e para tanto é necessário que se crie a idéia de uma assinatura digital. Similar a uma assinatura comum, a assinatura digital permite dar a idéia de autenticidade da informação ali prestada.

Quando uma pessoa assina um documento, esta tem que estar ciente de que sua assinatura atesta a validade do que ali está escrito, ninguém deve assinar um documento rasurado, rasgado ou em branco. A assinatura digital deve seguir os mesmos preceitos.

A assinatura convencional, procura oferecer garantias de identificação da autoria do documento, como também da integridade de seu conteúdo desde o ato de sua assinatura. Ainda assim, comparar assinatura digital e convencional basicamente se torna uma comparação incompleta. Precisamos saber a quem, e como, tais garantias são oferecidas, antes de nos deixarmos levar pelas promessas virtuais. Algumas pessoas podem mencionar riscos da assinatura convencional ser falsificada ou roubada, e que esses riscos não existiriam para a assinatura digital, no entanto, o contrário seria muito mais normal de acontecer.

Segundo Rezende [REZENDE00], só teria sentido o "roubo" de assinatura convencional, à caneta e em papel, para reuso. Isto é, sua extração de um documento legítimo para autenticar um outro. O roubo literal produz rasura ou emenda no suporte físico da assinatura reusada no papel que a vincula ao conteúdo autenticado. Mas rasuras ou emendas são facilmente detectáveis por inspeção deste suporte. Entretanto, para a assinatura digital não há suporte material, pois o documento eletrônico é apenas uma seqüência binária, que representa símbolos. Além de codificar seu conteúdo, esta seqüência terá que servir também como suporte para sua própria assinatura.

Para documentos eletrônicos cópias digitais são indistinguíveis de "originais". Sua

assinatura digital deverá então ser calculada, a partir da seqüência binária que lhe dá suporte e de uma outra seqüência binária que servirá para identificar o assinante, denominada chave de assinatura. A seqüência de bits resultante deste cálculo é concatenada a tal documento[REZENDE00]. Para eficácia deste processo, tal chave precisa ser mantida em sigilo por seu titular, e por isso é também chamada de chave privada, mas para isso, deve-se considerar uma política de segurança do sistema de arquivos local, onde esta chave privada estará "guardada". O equivalente ao sigilo da chave privada na assinatura convencional é a exigência legal de que sua impressão seja cursiva, ou seja, de próprio punho.

4.2 Criptografia

Ainda em se tratando de certificados, precisamos pensar em criptografia[1]. Não adianta criar a idéia de autenticação se a mensagem a ser enviada em uma rede, por exemplo, está sendo enviada abertamente, ou seja, texto simples.

A criptografia transforma um texto compreensível, denominado texto original ou texto em claro, em uma informação transformada, chamada de texto cifrado ou texto código ou simplesmente cifra, que tem a aparência de um texto gerado aleatoriamente e totalmente incompreensível.

O ato de transformar os dados para uma forma ilegível é denominado cifrar, e procura garantir a privacidade, mantendo a informação escondida de pessoas não autorizadas, mesmo que estas possam visualizar os dados criptografados.

O processo de cifrar é conhecido a mais de 2000 anos e um dos primeiros processos foi criado por César e era chamado "Cifra de César" e é um dos mais simples métodos para cifrar mensagens. Trata-se de uma cifra de substituição[2] na qual cada letra do texto a cifrar é substituída por outra situada na ordem do alfabeto um número fixo

[1] (do grego kriptós que significa escondido, oculto mais grápho que significa grafia, escrita)Arte de escrever em cifra, em alfabeto oculto

[2] cifra de substituição é um método de encriptação que opera de acordo com um sistema pré-definido de substituição

de posições. Por exemplo, com um número fixo de 3 posições, "A" é substituído por "D", "B" por "E", e assim sucessivamente. O nome do método tem origem em Júlio César[3], que se sabe ter usado este esquema para comunicar com os seus generais.

O processo inverso é conhecido por decifrar e ao cifrarmos ou decifrarmos uma mensagem, precisamos de informações confidenciais, denominadas chaves e os algoritmos de criptografia podem ser classificados utilizando-se de dois tipos de chaves: chave simétrica e chave assimétrica.

4.2.1 Criptografia com Chave Simétrica

Também conhecida por chave única, utiliza a mesma chave tanto para a cifragem como para a decifragem. Este método é bastante limitado, pois o emissor e receptor devem conhecer antecipadamente a chave, e é bastante difícil de se conseguir um meio seguro de se passar a chave secreta e gerenciá-las (se fosse simples, poderíamos simplesmente enviar os dados por esse meio). Esse é um tipo de chave mais simples. Existem vários algoritmos que usam chaves simétricas, alguns deles são descritos abaixo:

1. DES (Data Encryption Standard): criado pela IBM em 1977, faz uso de chaves de 56 bits. Isso corresponde a 72 quadrilhões de combinações. É um valor aparentemente alto, mas não para um computador potente. Em 1997, ele foi quebrado por técnicas de força bruta[4] em um desafio promovido na internet;

2. IDEA (International Data Encryption Algorithm): criado em 1991 por James Massey e Xuejia Lai, o IDEA é um algoritmo que faz uso de chaves de 128 bits e que tem uma estrutura semelhante ao DES. Sua implementação em software é mais fácil do que a implementação deste último;

3. RC (Ron's Code ou Rivest Cipher): criado por Ron Rivest na empresa RSA

[3] (em Latim: Gaius Julius Caesar) (viveu entre 13 de Julho, 100 a.C. e 15 de Março, 44 a.C.) foi um líder militar e político da República de Roma.

[4] tentativa e erro. Trata-se da técnica de projeto de algoritmos de mais fácil codificação, porém quase sempre de complexidade elevada, a^n, por exemplo

Data Security, esse algoritmo é muito utilizado em e-mails e faz uso de chaves que vão de 8 a 1024 bits. Possui várias versões: RC2, RC4, RC5 e RC6. Essencialmente, cada versão difere da outra por trabalhar com chaves maiores.

A Figura 4.1 mostra um exemplo de criptografia usando chaves simétricas, ou seja, chaves iguais para cifrar e decifrar. Na figura, um indivíduo deseja enviar uma mensagem e utiliza para isso uma chave para cifrar a mensagem, um possível observador que conseguisse ter acesso à essa mensagem não conseguiria decifrá-la pois não tem conhecimento da chave usada para cifrar. Ao chegar no destino, a mensagem deve ser decifrada para retornar ao formato original, para isso, é importante que tenha havido um compartilhamento da chave simétrica entre o emisor e o receptor da mensagem.

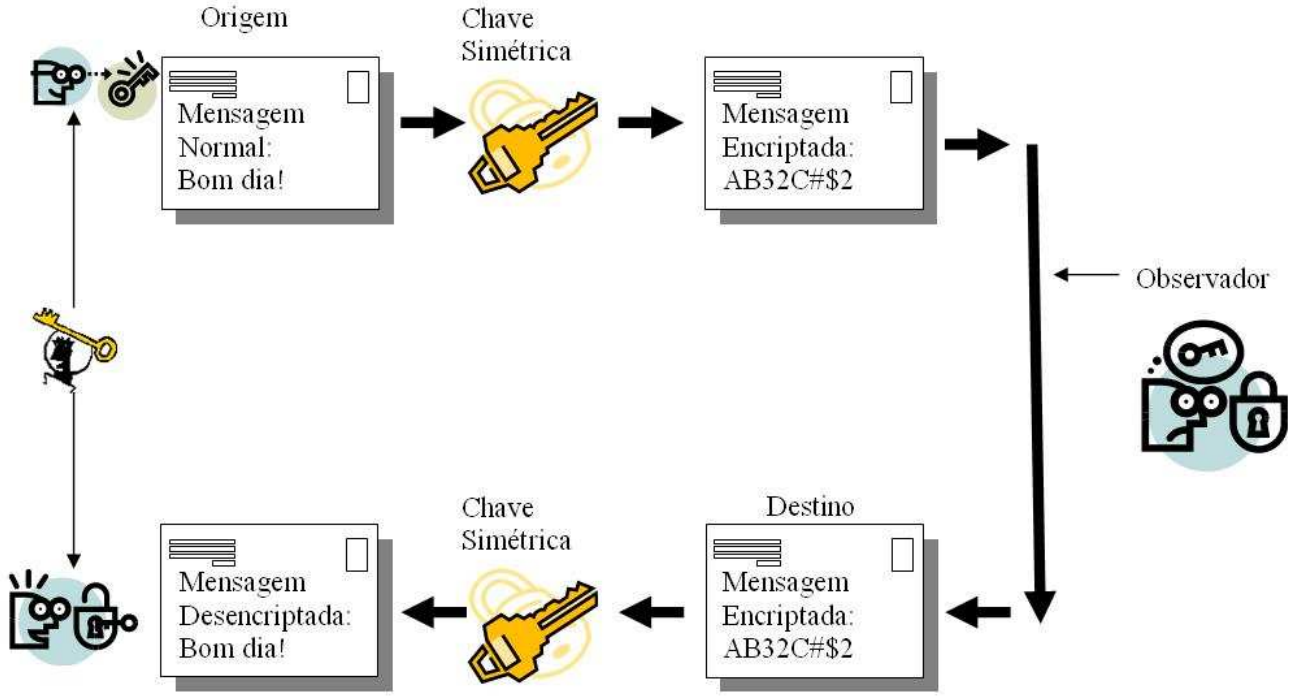

Figura 4.1: Criptografia com Chave Simétrica.

O uso de chaves simétricas tem algumas desvantagens, fazendo com que sua utilização não seja adequada em algumas situações como:

1. Onde a informação é muito valiosa.

2. Quando existem muitas pessoas envolvidas, um projeto por exemplo, nesse

caso, é necessário usar uma grande quantidade de chaves, uma para cada usuário.

3. Tanto o emissor quanto o receptor precisam conhecer a chave usada, o que complica a sua distribuição, tornando o processo de certa forma falho o que nos leva a situação 4.

4. A transmissão desta chave de um usuário para outro pode não ser segura, e em geral não o é, e cair em "mãos erradas" (de pessoas maliciosas, por exemplo).

4.2.2 Criptografia com Chave Assimétrica

Também chamados de algoritmos de criptografia de chaves públicas e privadas, utilizam chaves diferentes para cifrar e decifrar os dados. Em um sistema de chave assimétrica cada pessoa tem duas chaves: uma chave pública que pode ser divulgada e outra privada que deve ser mantida em segredo, "guardada" em local seguro, onde pessoas mal intencionadas não tenham acesso.

Mensagens cifradas com a chave pública só podem ser decifradas com a chave secreta e vice versa. Em outras palavras, uma das chaves dizemos ser a chave privada, a ser mantida em sigilo pelo usuário, em seu exclusivo poder, e a outra, a chave pública, que, como sugere o nome pode e deve ser livremente distribuída. Estas duas chaves são dois números que se relacionam de tal modo que uma desfaz o que a outra faz. Cifrando a mensagem com a chave pública, geramos uma mensagem cifrada que não pode ser decifrada com a própria chave pública que a gerou. Só com o uso da chave privada poderemos decifrar a mensagem que foi codificada com a chave pública. E o contrário também é verdadeiro: o que for cifrado com o uso da chave privada, só poderá ser decifrado com a chave pública.

A Figura 4.2 mostra o funcionamento básico de uma mensagem criptografada usando chaves assimétricas, ou seja, chaves diferentes para cifrar e decifrar. O funcionamento é simples: A pessoa que desejar que as mensagens recebidas de um destino qualquer garantam certo nível de segurança, pode gerar duas chaves. A chave privada, que ficará em sua posse e a chave pública que será destinada às pessoas que

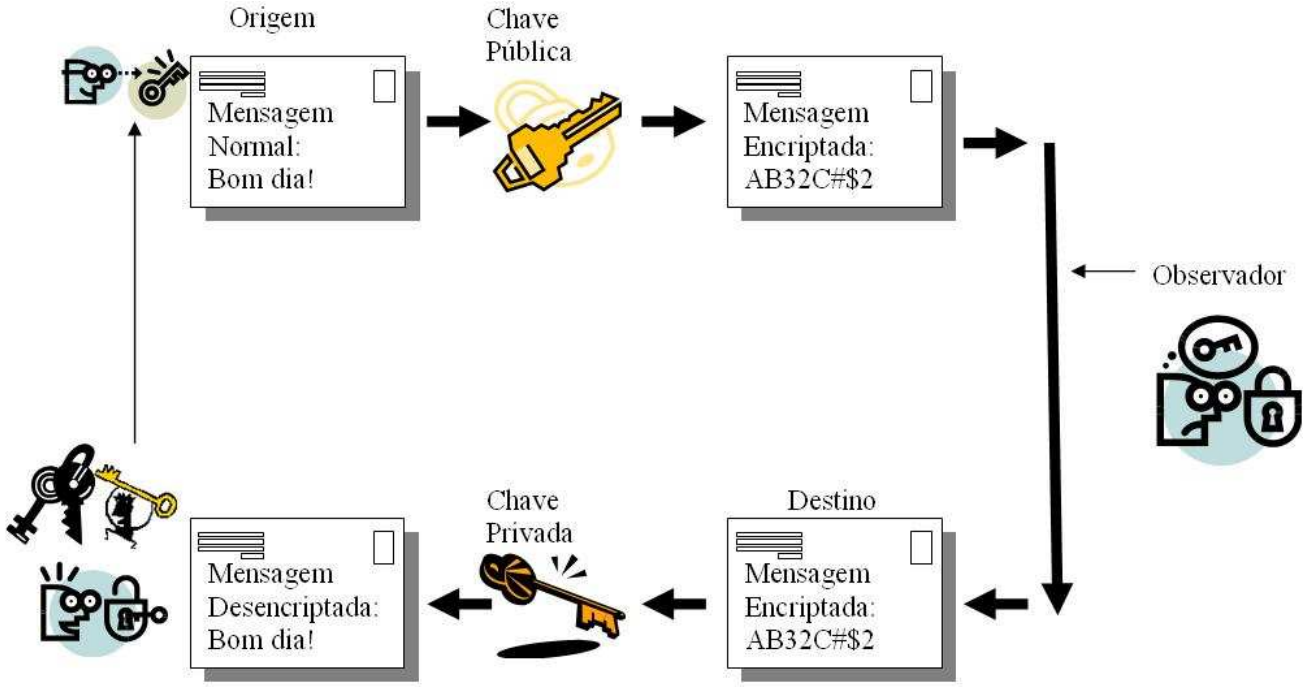

Figura 4.2: Criptografia com Chave Assimétrica.

desejarem enviar-lhe mensagem, assim sendo, a pessoa após criar a mensagem a ser enviada se utilizará da chave pública do destinatário para cifrar esta. A diferença em relação às chaves convencionais (chave simétrica) é que somente o destinatário real da mensagem poderá decifrá-la, já que, teóricamente, somente este possui a chave que decifra a mensagem, ou seja, a chave privada.

Vale lembrar que neste tipo de criptografia, é excencial que a chave privada seja bem protegida e que, ao contrário, a chave pública deve ser livre para as pessoas que se propuserem a enviar uma mensagem deste tipo.

4.2.3 Assinatura Digital X Criptografia com Chave Assimétrica

Além da utilização das técnicas descritas acima, sempre é possível utilizar-se de mais de uma dessas para criar uma estrutura mais funcional e mais segura. É comum que isto seja feito utilizando-se chaves públicas e privadas associadas à assinatura digital de forma a garantir a autenticidade do destinatário e do emisor.

Para um melhor entendimento do funcionamento deste tipo de estrutura será mostrado um exemplo de utilização descrito em diversas páginas na Internet, este foi retirado de [VABSOLUTA06] como segue:

Supondo que Frank quer enviar uma mensagem para todos os seus contatos informando-os que mudou de emprego. Na verdade, ele não se importa com quem lerá a mensagem, mas quer ter certeza de garantir a seus contatos que a mensagem realmente é dele, e não de outra pessoa.

1. Frank escreve a mensagem e a cifra utilizando sua chave privada.
2. Frank envia a mensagem a seus contatos através da Internet.
3. Os contatos recebem a mensagem e a decifram utilizando a chave pública de Frank.

O fato de a chave pública de Frank ter decifrado a mensagem garante aos contatos que a mensagem realmente é de Frank. Qualquer mensagem decifrada com a chave pública de Frank só poderia ter sido criada com sua chave privada.

Isso é muito importante. Na criptografia com chave pública, cada par de chaves é único. Só existe apenas uma chave pública para cada chave privada e vice-versa. Se isso não fosse verdade, a assinatura digital não seria possível e um impostor poderia utilizar outra chave privada para criar uma mensagem que pudesse ser lida pela chave pública fornecida.

Podemos observar que a assinatura digital assegura aos contatos que a mensagem não foi alterada (integridade) e que veio de Frank (autenticidade). Além disso, Frank é o único com acesso a sua chave privada.

Agora combinando os dois métodos, Frank pode enviar uma mensagem privada e assinada para Andrea, como segue:

1. Frank escreve a mensagem e a cifra utilizando sua chave privada (assinatura da mensagem).

2. Em seguida, ele cifra a mensagem com a chave pública de Andrea (tornando-a privada)

3. Frank envia a mensagem duplamente cifrada para Andrea através da Internet.

4. Andrea recebe a mensagem.

5. Ela decifra a mensagem duas vezes. Primeiro, ela utiliza sua chave privada e, depois, a chave pública de Frank.

6. Agora Andrea pode ler a mensagem e tem certeza que ela é secreta e veio de Frank. Ela tem certeza de que a mensagem não foi modificada, para alterá-la, o observador teria de acessar a chave privada de Frank.

Outro exemplo pode ser visto abaixo e define-se quando se propõe a utilização de mais métodos, ou seja, criptografia com chave simétrica e criptografia com chave assimétrica podem ser combinados codificando-se a mensagem com o método da chave simétrica e cifrando-se a chave simétrica com o método de chave pública como segue:

Novamente vamos considerar Frank e Andrea. Frank quer enviar uma mensagem à Andrea, mas dessa vez ele utilizará uma combinação de criptografia com chave pública e de criptografia com chave simétrica:

1. Frank escreve a mensagem e a codifica utilizando a criptografia com chave simétrica com uma chave que ele cria aleatoriamente apenas para essa mensagem. Isso é conhecido como chave de mensagem ou chave de sessão.

2. Frank cifra essa chave de sessão com a chave pública de Andrea.

3. Frank envia a mensagem cifrada e a chave de sessão cifrada à Andrea.

4. Andrea decifra a chave de sessão utilizando sua chave privada.

5. Em seguida, Andrea decifra a mensagem usando a chave de sessão que acabou de receber.

6. Andrea agora pode ler a mensagem.

Esse método se beneficia da força dos dois tipos de criptossistemas[5]: a velocidade da criptografia simétrica e a facilidade dos mecanismos de distribuição de chave do sistema de criptografia com chave pública.

4.3 Segurança em Certificados Digitais

Como acabamos de ver, a criptografia com chave pública pode ser usada para proporcionar confidencialidade, integridade e não-repudiação. A assinatura digital exige que o verificador esteja certo de que tem uma chave pública pertencente à pessoa que assinou a mensagem. A confiança do verificador na assinatura deverá ser igual à sua confiança no proprietário da chave pública. Por exemplo, se um observador quiser forjar documentos eletrônicos, uma estratégia seria criar um par de chaves, pública e privada, e divulgar a chave pública com o nome de outra pessoa. Qualquer documentos assinados com a chave privada do observador serão verificados com a chave pública correspondente, ou seja, uma chave pública que tenha sido anunciada como pertencente a outra pessoa.

Existem diversas estratégias para solucionar esse problema. Uma delas é trocar chaves públicas através de um meio direto, como uma reunião física. Infelizmente, esse método não funciona bem quando temos muitas pessoas envolvidas, ou quando as pessoas estão distantes geograficamente umas das outras. Uma opção melhor seria a utilização de certificados e de autoridades de certificação.

Um certificado é um documento digital contendo informações de identificação e uma chave pública. Em geral, os certificados têm um formato comum, normalmente baseados no padrão ITU-T X.509[SILVA06]. Mas ainda assim, não podemos garantir que o certificado é genuíno e não falso. Uma forma de descobrir isso seria utilizar autoridades de certificação, CAs.

Uma autoridade de certificação assina certificados de chave pública digitalmente.

[5]sistemas que dispõe de algoritmos e funções de criptografia, projetados para resistir à ataques

Ao assinar um certificado, a CA garante sua validade. No entanto um problema persiste de como a chave pública da CA seria distribuída. Também existem muitas estratégias para esse problema, em uma delas, se a CA for muito conhecida, ele poderá divulgar amplamente sua chave pública. Outro método seria que a CA tivesse seu próprio certificado assinado por outra CA, também conhecida pelo destinatário.

Esta idéia de encadeamento de certificação pode avançar ainda mais, com várias CAs organizadas em uma hierarquia onde cada CA subordinada valida sua assinatura com a assinatura de uma CA mais alta na hierarquia. Obviamente, as CAs de nível mais alto deverão reverter para o método de divulgação direta.

4.3.1 Autoridade Certificadora - CA

Imprecisões são comuns em explicações leigas sobre o processo de certificação, como define Pedro Rezende[REZENDE00]. São freqüentes as afirmações de que, para ter um par de chaves assimétricas, o internauta deve primeiro se cadastrar numa certificadora digital, após o qual passará a contar com uma função a mais em seu navegador, entretanto, não é bem assim que isto funciona. A geração do seu par de chaves é o primeiro passo, que precisa ser executado em seu próprio ambiente.

O que as autoridades certificadoras (CA) procuram hoje oferecer, valendo-se dos navegadores, é uma infra-estrutura global para o uso interoperável de chaves criptográficas assimétricas, uma PKI (*Public Key Infrastructure*). No caso dos navegadores, o processo obedece aos padrões adotados pelo protocolo de segurança neles implementados, o SSL (*Secure Sockets Layer*), já adaptado ao TCP/IP como TSL[6]. Ao pedir um certificado, o usuário gera um par de chaves assimétricas. A chave privada será armazenada no seu disco e a chave pública submetida à certificação pela CA escolhida, juntamente com os dados do titular, conforme irão constar no certificado x509 que a distribuirá. Esta CA assina tal certificado mediante cobrança,

[6] O Protocolo TSL (Camada de transporte segura), foi desenvolvido pela IETF (Força Tarefa de Engenharia para Internet) em 1999 baseado no protocolo SSL (Camada de conexão segura) 3.0 da empresa Netscape Communications. O TSL tem como finalidade prover comunicação segura pela Internet, utilizando métodos de criptografia simétrica e assimétrica.

devolvendo-o assinado ao solicitante. Apenas certificados assinados são aceitos pelo SSL.

Não é a pessoa quem é cadastrada na CA, mas a sua chave pública, desta forma, podemos cadastrar várias chaves públicas em um único nome, ou cadastrá-las com qualquer outro nome, mesmo sendo de um único usuário. As CAs podem até verificar a identidade civil do titular dos certificados que assina. Mas normalmente se eximem desta responsabilidade. Num certificado x509 o titular deste é apenas uma seqüência de letras, e cabe a quem for usá-lo interpretá-la como identificação de alguém ou de algo, assim sendo, a certificação não garante a identidade de ninguém, mas apenas a integridade léxica[7] de uma chave pública e de um nome, a ela associado no ato de certificação por quem a apresentou.

O certificado garante que o titular é quem diz ser, é um figura de linguagem para efeito de marketing. A questão de alguém ser o que diz ser não tem nada a ver com criptografia. A criptografia é constituída de procedimentos sintáticos, e a identificação de uma entidade física ou jurídica é um procedimento semântico, um processo cultural que se torna bem mais complexo em redes e principalmente na internet. Por exemplo: Quem é "licht@lncc.br"? Quem é "www.google.com.br"? De que forma cada uma dessas seqüências de letras poderiam garantir ser quem diz ser?

Certificados assinados por CAs são necessários a um browser, por exemplo, porque este implementa o SSL. No SSL, uma cadeia de autenticação é percorrida, onde as chaves públicas destas entidades são usadas para verificar assinaturas em certificados transmitidos ao browser no momento da abertura de uma conexão protegida[8].

Um certificado enviado ao SSL contém a chave pública para estabelecimento de sigilo com seu titular, ou para verificação de sua assinatura. A integridade do conteúdo deste certificado é verificada pela chave pública da CA que o assinou.

[7] processo de analisar a entrada de linhas de caracteres (tal como o código-fonte de um programa de computador) e produzir uma seqüência de símbolos chamados "símbolos léxicos"

[8] Mostram um cadeado fechado na tela, em geral no canto inferior direito do navegador

4.3.2 SPKI (Simple Public Key Infrastructure) / SDSI (Simple Distributed Security Infrastructure)

O desenvolvimento do SDSI (*Simple Distributed Security Infrastructure*) e do SPKI (*Simple Public Key Infrastructure*) foi motivado pela limitação e pela complexidade da infra-estrutura de chaves publicas baseada na hierarquia global de nomes, X.509. O SPKI é uma infra-estrutura de chaves públicas que tem como características o uso de *namespaces* locais. O SPKI [CARRIAO01] foi desenhado com a intenção de ser um modelo de autorização simples e flexível, muito bem definido e de fácil implementação. A união do SPKI e SDSI resultou em um sistema de autenticação e autorização para aplicações distribuídas, (grades, por exemplo). Com a criação desta infra-estrutura os *namespaces* **principais**[9] tornam-se locais e o modelo baseado em cadeias de confiança é simples e flexível.

Em SPKI / SDSI existem dois tipos distintos de certificados: para nomes e para autorizações com as seguintes características:

1. Os certificados de nomes são responsáveis por associar nomes a chaves públicas ou a outros nomes.

2. O sistema de nomeação é adotado do SDSI que induz ao uso de nomes locais mesmo no sentido global de um ambiente distribuído.

3. Os nomes SPKI / SDSI são sempre locais, correspondendo ao espaço de nomes de quem emitiu o certificado. O emissor do certificado é sempre identificado pela sua chave pública.

4. A combinação chave pública mais nome local forma um identificador global único.

No SPKI / SDSI é usado um modelo igualitário, os **principais**, que são chaves públicas que podem assinar e divulgar certificados, como uma CA do X.509, assim,

[9] Entidades ativas que possuem um par de chaves (privada e pública) e através disto podem executar assinaturas digitais.

qualquer **principal** pode criar seu par de chaves (privada e pública), e então, associar à chave pública do par a um nome no seu espaço local de nomes e divulgá-los através de certificados, o que exclui a necessidade de uma entidade centralizadora que faça o registro de chaves públicas e emita certificados como a CA da PKI X.509. Assim, cada principal define da maneira que lhe parecer mais intuitiva em seu espaço de nomes, os nomes atribuídos a um outro principal.

Um certificado de nomes pode fazer referência a um nome publicado num certificado, no espaço de nomes de outro **principal** e assim sucessivamente, de modo a formar uma cadeia de certificados de nome [CARRIAO01]. Assim, a divulgação de nomes no SPKI / SDSI é feita através de redes de confiança formadas por certificados de nomes ligados por encadeamento de referências. Estas cadeias de nomes devem ser reduzidas a uma chave pública que representa o **principal** sendo referenciado quando se deseja a identificação do mesmo.

Além disso, os certificados de autorização SPKI / SDSI ligam autorizações a um nome ou a uma chave. Através destes certificados, o emissor delega permissões de acesso a outros **principais** no sistema. Na infra-estrutura SPKI / SDSI os certificados de autorização são construídos a partir das ACL's (*Access Control List*) do guardião[10]. O conteúdo do certificado pode ser o mesmo da ACL, porém ao certificado é acrescido o campo do emissor assinando o certificado e a ACL não possui este campo porque é local ao guardião do serviço.

4.3.3 Certificados X.509

Modelo hierárquico[11] de confiança muito comum que tem em foco utilizar um nome que seja único e que identifique globalmente o proprietário do certificado, na prática infelizmente não funcionam como inicialmente proposto. Cristian e Luis [SOUZA01] afirmam que as Autoridades Certificadoras (CAs) trabalham de forma individualizada, não havendo nenhuma relação de confiança entre elas. Como cada Autoridade Certificadora trabalha de forma individual, cada uma com seus próprios

[10] monitor de referência

[11] Hierarquia de certificação onde é necessária a autenticação por uma pessoa

critérios, não há como estabelecer um "Distinguished Name" que seja realmente único e identifique de maneira global o proprietário do certificado.

No SPKI/SDSI não existe mais o "Distinguished Name", os certificados passam a ser identificados pela chave pública e o sujeito é a própria chave pública, neste caso, o mapeamento entre a chave pública e o proprietário do certificado passa a existir no espaço local de nomes e os nomes não precisam mais ser globalmente únicos. Entretanto eles precisam ser únicos e significativos para a pessoa que os mantém, semelhantes aos nomes que cadastramos em nossas agendas telefônicas ou aos apelidos em nossa lista de contato de e-mails. Neste caso, cada usuário é livre para que de acordo com o seu grau de confiança aceite ou emita certificados para quem desejar, dispensando o papel da Autoridade Certificadora, criando a idéia de uma unidade certificadora hierárquica de confiança.

Características de um Certificado X.509

1. Todos os certificados X.509 obedecem o padrão internacional ITU-T X.509, assim, teoricamente certificados X.509 criados para uma aplicação podem ser usados por qualquer aplicação que obedece seu padrão. Em prática, porém, companhias diferentes criaram as próprias extensões delas para certificados X.509.

2. Um certificado X.509 é uma coleção de um conjunto padrão de campos contendo informações sobre um usuário ou dispositivo e sua correspondente chave pública.

3. O padrão X.509 define qual informação vai ser colocada no certificado, e descreve como codificar o formato dos dados. Todos certificados X.509 têm os seguintes dados:

   (a) O número da versão do X.509. Isto identifica qual padrão é aplicado na versão do X.509 para este certificado, o que afeta qual informação pode ser especificada neste.

(b) A chave pública do possuidor do certificado junto com um algoritmo de identificação que especifica qual sistema de criptografia pertence a chave e quaisquer parâmetros associados.

(c) O número de série do certificado e a entidade (aplicação ou pessoa) que criou o certificado. Neste existe um número de série para distinguir este de outros certificados que ele emite. Esta informação é usada de várias maneiras, por exemplo, quando um certificado é revogado seu número serial é colocado em uma Lista de Revogação de Certificado ou CRL.

(d) A identificação única do possuidor de certificado ou DN. Este nome tem que ser único pela Internet. Um DN consiste em múltiplas subseções e pode parecer com algo do tipo: CN="Nome Comum do assunto", OU="Unidade Organizacional", O="Organização", C="País", ou mais claramente:
"/O=Grid/OU=LNCC/OU=ComCiDis-CA/OU=lncc.br/CN=Globus User"

(e) O período de validade do certificado a data do começo do certificado e data de vencimento para indicar quando o certificado expira.

(f) A assinatura digital do emissor e a assinatura que usa a chave privada da entidade que emitiu o certificado.

(g) A identificação do algoritmo de assinatura.

4. Certificados X.509 suportam só um único nome para o dono de chave

5. Certificados X.509 suportam só uma única assinatura digital para atestar a validez da chave.

6. Para obter um certificado X.509, você tem que pedir para uma CA que emita um certificado para você. Assim, você providencia sua chave pública, prova que você possui a chave privada correspondente, e alguma informação específica a seu respeito. Então, digitalmente assina a informação e envia o pedido de certificado para a CA. A CA então executa alguma devida verificação que a informação que você providenciou está correta, e neste caso, gera o certificado e retorna ele a quem o pediu.

É possível pensar em um certificado X.509 como parecendo um certificado de papel padrão com uma chave pública gravada nele. Este teria seu nome e um pouco de informação sobre você, mais a assinatura da pessoa que emitiu isto para você.

4.4 Uso dos Certificados

Na prática, o uso de certificados melhora o nível de segurança em uma rede, mas vale lembrar que para que isso seja um fato existem restrições que devem ser seguidas, como as que se seguem:

1. **Tamanho da chave:** A criptoanálise[12], se baseia no fatoramento de números grandes. Conseqüentemente, quanto maior a chave, mais difícil decifrá-la.

2. **Revogação de certificado:** O que acontece quando uma chave privada é comprometida, ou uma chave pública passa a ser inválida? Nesse caso, os certificados deixam de ser confiáveis, pois as informações que eles estão certificando não são mais verdadeiras. No entanto existe uma certa dificuldade em impedir que os certificados continuem sendo usados. Muitas autoridades de certificação divulgam periodicamente listas de certificados que não devem mais ser considerados válidos, as CRLs (certificate revocation lists). Os certificados dessas listas podem ter expirados, ou o par de chaves associados ao certificado pode ter sido decifrado. As CRLs são muito eficientes para verificar a precisão de um certificado em um determinado momento. No entanto, as CRLs não lhe dão qualquer garantia de que um certificado não foi revogado desde sua divulgação. Por essa razão, muitas organizações estão procurando outras soluções para lidar com a verificação de certificados.

3. **Estrutura de CA:** Grandes organizações podem optar por ter dentro delas várias CAs (uma para o departamento de pesquisa, outra para o departamento de bolsas, outra para o setor de computação e assim por diante) e neste caso, ter uma única CA de nível mais alto para certificar as CAs dos departamentos.

[12] Ciência da decodificação de cifras

No entanto, manter vários níveis de CAs pode ser complicado. Além disso, ter uma única CA na empresa cria um "único ponto de falha", pois o comprometimento do par de chaves da CA corporativa pode resultar na perda das chaves de todos os funcionários da empresa. Outras empresas optam por uma estrutura plana de CAs, na qual cada CA departamental tem muitas outras CAs parceiras, que correspondem ao outros departamentos da empresa. Em qualquer caso é bom ter o conhecimento da estrutura de uma CA e optar pela estrutura que melhor atenda ao que for proposto, ou seja, o comprometimento de uma única CA pode resultar no comprometimento de todos os certificados da empresa, ou é melhor ter uma estrutura plana, na qual várias CAs devem ser controladas e devem trocar mensagens umas com as outras?[VABSOLUTA06]. Esse é o primeiro passo para que uma CA possa ser criada atendendo não somente uma empresa como citado acima, mas também a estrutura da CA que será instalada em uma grade computacional.

4. **Fazer ou não fazer caução:** Esse é um dos assuntos mais debatidos no que se refere à criptografia. Muitas empresas afirmam que, como a comunicação dos funcionários é propriedade da empresa, a organização deverá ter acesso às chaves dos funcionários, recuperar mensagens (quando houver desligamento de funcionários ou quando eles perderem suas chaves, por exemplo). Por outro lado, a maior parte dos defensores da privacidade é terminantemente contra essa idéia.

5. **O que fazer com todas essas informações:** O arquivamento de chaves e de dados cifrados (mensagens de correio eletrônico cifradas, ordens de compra assinadas, certificados de utilização da grade,...) é uma questão extremamente difícil. Muitas empresas têm de armazenar informações durante muito tempo, devido a regulamentos ou outras restrições. Mas, com freqüência, armazenar mensagens ou ordens de compra pode envolver muito mais do que simples manutenção de informações. Considere, por exemplo, o armazenamento de ordens de compras assinadas. Como as assinaturas só podem ser verificadas com a chave pública apropriada, todas as chaves públicas (e suas cadeias de certificação) devem ser guardadas para sempre. Mais uma vez, para grandes

empresas, isso pode ser complicado e, no caso de uma grade, pode deixar o sistema vulnerável quanto a possibilidade de quebra de chaves por força bruta, por exemplo, caso estas venham a cair em "mãos erradas".

4.5 Conclusão do capítulo

Neste capítulo foram vistos diversos métodos para se garantir a integridade, segurança e autenticidade de um documento. No próximo capítulo serão mostrados os mecanismos descritos neste capítulo incidindo principalmente na utilização de MyProxy que é uma ferramenta que tem a função de fornecer certificados com tempo de vida relativamente curto, possibilitando um maior controle dos certificados. Trataremos de toda a estrutura deste serviço que é essencial para o entendimento desta dissertação, bem como de todo o processo exposto desde o início do trabalho.

Capítulo 5

Certificados Proxy

Neste capítulo serão mostradas algumas definições referentes à utilização de certificados de curta duração, a seção 5.1 trata diretamente deste ponto. Na seção 5.1.1 é mostrado MyProxy que tem como função fornecer certificados de curta duração, na seção 5.1.2 é tratado do funcionamento do servidor MyProxy. Na seção 5.1.3 são tratados alguns assuntos como Delegação, Autenticação Mútua e Certificados Proxy. Na seção A.2 são mostrados alguns comandos do MyProxy seguido na seção 5.1.4 pela definição de algumas premissas do uso de MyProxy e por fim na seção 5.1.5 uma breve conclusão deste capítulo.

5.1 Certificados de Curta Duração

Com o aparecimento das grades computacionais, a preocupação com segurança deixou de ser um problema local e tornou-se uma meta inter-entidades de diversos domínios. Fornecer direitos de uso dos recursos da grade é complexo, isso sem falar na dificuldade em se gerar um certificado para os usuários como descrito no Capítulo 4.

5.1.1 MyProxy

MyProxy é um software de código aberto para gerenciamento de infra-estrutura de chaves públicas X.509. MyProxy combina um repositório de credencial on-line com uma autoridade certificadora (CA) para permitir que os usuários obtenham facilmente credenciais quando e onde necessitarem[NCSA06].

MyProxy tem como preceito fornecer certificados temporários para os usuários que desejarem submeter seus trabalhos. Uma das vantagens neste tipo de serviço é que é possível criar a idéia de "segurança temporária", ou seja, um usuário ganha o direito de usar a grade por um período curto de tempo, e, acabado esse tempo seu certificado expira e é necessário que se renove para continuar usando a grade.

Como comentado anteriormente nesta dissertação, quando um usuário deseja utilizar a grade computacional é necessário que se emita um certificado de uso, estes certificados devem ser compatíveis com a Autoridade Certificadora da grade, em especial pelo Globus, e MyProxy pode fornecer estes certificados. O processo para que isso seja efetuado utilizando o *middleware globus tookit*, segue os seguintes passos, impreterivelmente nessa ordem:

1. O usuário solicita o modelo do certificado à Autoridade Certificadora.

2. O usuário compila esse certificado usando o *Globus Tookit* em sua máquina usando o seguinte comando:

   $GLOBUS_LOCATION/sbin/gpt-build globus_simple_ca_<...>.tar.gz gcc32dbg

3. O usuário após esse passo, deve definir este certificado como padrão, pois, uma máquina pode possuir diversos certificados de diversas autoridades certificadoras, entretanto, somente um destes certificados pode ser usado por vez como padrão:

   $GLOBUS_LOCATION/setup/gpt-build globus_simple_ca_<...>/setup-gsi -default

4. O próximo passo é requisitar seu certificado de usuário da grade, gerando a chave pública e a chave privada com o seguinte comando:

   grid-cert-request (neste passo será pedida a senha para o certificado pessoal)

5. Agora deve-se enviar a chave pública à Autoridade Certificadora da grade para ser assinado:

   cat /home/globus/.globus/usercert_request.pem | mail licht@lncc.br

6. O certificado agora tem que ser assinado pela Autoridade Certificadora com o comando:

   grid-ca-sign -in usercert_request.pem -out usercert.pem

   Onde usercert_request.pem é a chave pública da pessoa que requisitou a assinatura e usercert.pem é a chave assinada que será reenviada ao solicitante.

7. O certificado assinado é enviado ao solicitante e este deve colocar o certificado assinado em:

   /home/globus/.globus

   Vale lembrar que este diretório citado acima é o diretório padrão em sistemas operacionais derivados do unix, como linux, por exemplo, e para aplicações com o *Middleware Globus Toolkit*.

Como pôde ser visto nos passos acima, não é tão simples submeter um trabalho à grade. É fato a necessidade de utilização de permissões devido à necessidade de segurança, também é fato que devido a estas necessidades, pode ocorrer que a inclusão de um novo usuário leve tempo elevado e por muitas vezes intolerável. Isto ocorre, por exemplo, quando da assinatura do certificado depende um gerente de certificação que pode não estar disponível naquele momento, sem levar em conta a possibilidade da rede estar indisponível ou mesmo o email com a chave pública não chegar ao destino.

Quando se pensa nos problemas citados acima, notamos que até que se perceba a demora no retorno do certificado assinado e a necessidade de reenvio, pode ter

passado muito tempo, que pode prejudicar o uso da grade (dias, por exemplo). Em geral, quando usamos a grade, percebemos a necessidade de que tudo seja mais automatizado, ou seja, em poucos minutos possa estar usando o serviço, entretanto, se tiver que esperar um tempo maior que esse, pode causar perda do interesse no uso da grade ou mesmo perda da necessidade de uso, já que pode-se não estar mais disponível para submissão de trabalhos. Um exemplo que pode ser visto é quando em um *workshop*, em que o período normal é em torno de uma semana e, se o certificado demorar tal tempo, pode ter prejudicado os testes e uso da grade.

MyProxy veio com o intuito de fornecer certificados de uma maneira mais simples e funcional, para que se consiga uma permissão temporária de uso da grade,neste caso, basta que se solicite um certificado ao servidor MyProxy e este o enviará dinâmicamente, (desde que o certificados esteja disponível em um repositório), tudo isso sem abrir mão da segurança.

Em um processo normal de criação de certificados, é definido o tempo em que esse certificado estará valido, em uma instalação normal da CA esse tempo é de 5 anos, mas é comum entre os gerentes de certificação definir tempos menores que este (um ano, por exemplo).

Quando é usado o servidor MyProxy, este "tempo de validade" do certificado costuma ser bem menor. Normalmente este tempo é de sete dias podendo ser aumentado ou diminuido de acordo com as políticas de cada CA. Vale lembrar que quando da expiração do certificado, caso os trabalhos submetidos à grade, ou por outro motivo um usuário desejar que seu certificado tenha um prazo maior, basta que este faça uma nova requisição, renovando o mesmo. Para não se perder o controle disto, o gerente de certificação, pode a qualquer tempo, tirar os direitos de uso se assim achar necessário, por considerar que o certificado foi violado, por exemplo.

5.1.2 Servidor MyProxy

MyProxy teve sua criação baseada no livro de A.Menezes, P.Oorschot, e S.Vanstone [MENEZES96], também citado por Jim Basney do NCSA, um dos criadores do My-

Proxy, apresentando *MyProxy: A Multi-Purpose Grid Authentication Service* no *IV WCGA Workshop de Computação em Grid e Aplicações* junto ao 24° Simpósio Brasileiro de Redes de Computadores que aconteceu em Curitiba-PR entre 29 de maio a 02 de junho de 2006 onde abordou aspectos relacionados á segurança de *ciberinfraestruturas e ciberambientes*. Especificamente o capítulo 8 do livro trata da criptografia de chaves públicas, também citadas no Capítulo 4 desta dissertação.

MyProxy é um repositório de credenciais, ou seja, é possível armazenar credenciais do proxy X.509[1] no repositório MyProxy, protegido por uma senha, para uma recuperação futura quando se fizer necessário. Isto elimina a necessidade de copiar manualmente as chaves (pública e privada) entre as máquinas, as quais se desejar submeter um trabalho.

O Globus 4 foi a primeira versão do globus toolkit a implantar no código o MyProxy, e no site do Globus Alliance [GLOBUSSEC06] é possível verificar suas características e funcionalidades.

Funcionamento do Servidor MyProxy

A Figura 5.1 mostra o funcionamento do servidor MyProxy e cada passo é descrito através das setas numeradas como segue:

1. A autoridade certificadora possui um modelo de certificado e, pode estar diretamente ligada ao servidor MyProxy assinando os certificados que ficarão disponíveis em um repositório deste. Outra possibilidade é instalar o Servidor MyProxy na própria autoridade certificadora e manter o repositório de certificados nesta.

2. A comunicação entre o servidor MyProxy e o Servidor Web é importante, pois a partir deste é possível que um usuário, através de um portal faça sua submissão de tarefas com um certificado temporário criado através da própria interface web de uma maneira transparente para o usuário.

[1] formato comum de certificados digitais especificado pelo padrão ITU X.509 na RFC 2459

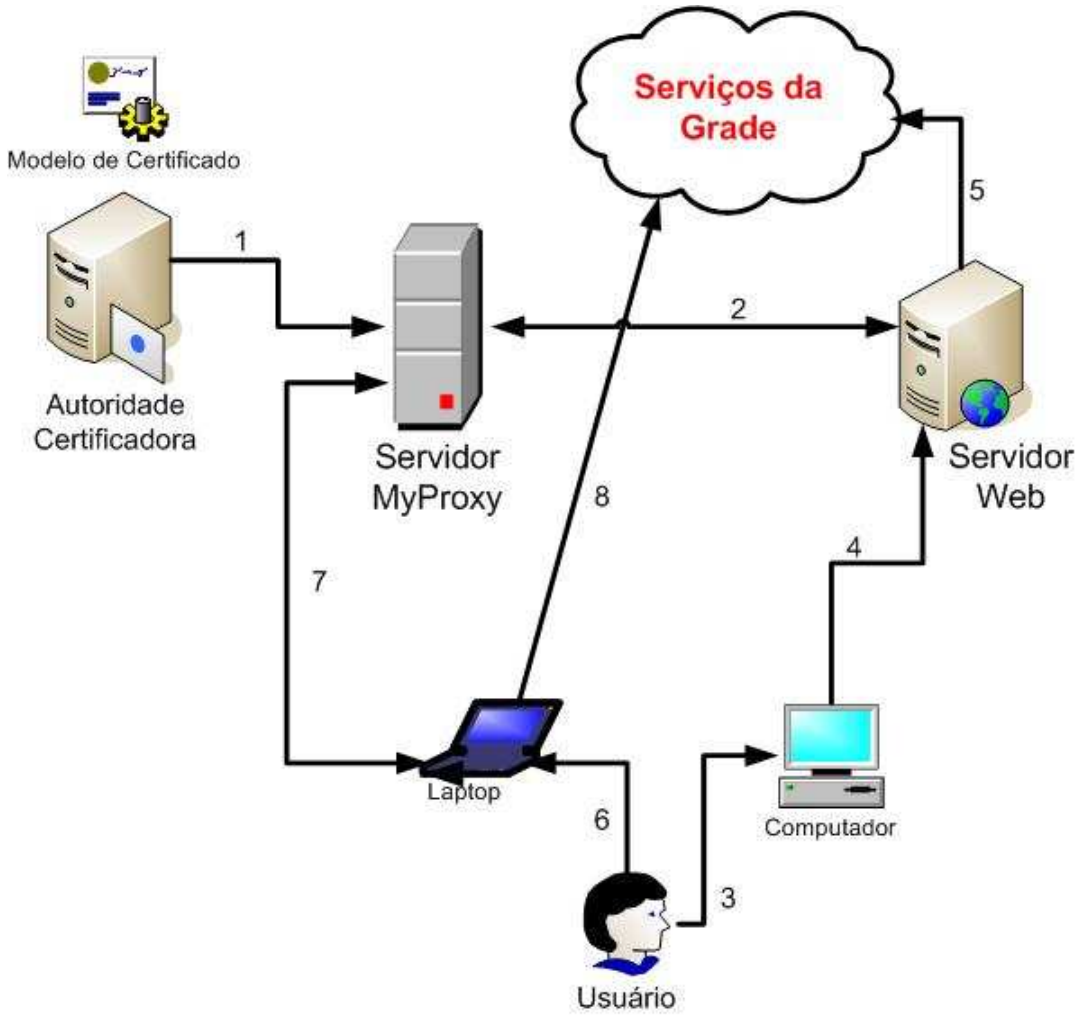

Figura 5.1: Funcionamento Servidor MyProxy.

3. Um usuário pode através de seu computador, mediante o uso de qualquer sistema operacional se conectar via navegador de Internet ao servidor web onde está instalado o portal da grade computacional.

4. A conexão com o servidor Web é feita sem que haja certificados de uso da grade no computador do usuário.

5. Uma vez aberta a conexão com o servidor web, o proxy do usuário é criado no servidor.

6. Com o proxy criado, o usuário pode utilizar os serviços da grade.

7. Outra possibilidade é o usuário usar um computador com certificado local.

8. Se o usuário possuir uma credencial no servidor MyProxy, este pode solicitar, via terminal, ou interface, a criação de um proxy em seu computador.

9. Uma vez criado o proxy, o usuário pode utilizar diretamente aos serviços da grade sem a necessidade de uso de um portal.

5.1.3 Delegação, Autenticação Mútua e Certificados Proxy

GSI (Grid Security Infrastructure) fornece certa potencialidade na delegação de direitos de uso. Por ser uma extensão do protocolo padrão do SSL (Secure Sockets Layer)[2], este reduz o número de vezes que o usuário deve entrar com sua senha. Se para executar uma aplicação na grade fossem requeridos diversos recursos em diversos computadores ou se houver a necessidade de ter agentes (locais ou remotos) para a aplicação executar, seria requerida uma autenticação em cada máquina, como é padrão nos sistemas operacionais derivados do Unix, assim, pedir estes recursos em nome de um usuário iria requerer que a senha fosse usada para criação de cada entrada. Tudo isso pode ser evitado criando-se um proxy.

Um proxy consiste em um certificado e em uma chave privada. O par de chaves que é usado para o proxy, isto é, a chave pública e a chave privada, podem ser regeradas para cada proxy ou ser obtida por outros meios. O certificado contém a identidade do proprietário como visto no Capitulo 4, modificada ligeiramente para indicar que é um proxy. O certificado é assinado e inclui uma notação de tempo, depois do qual o proxy não mais será aceito, ou seja, perderá a validade.

A chave privada do proxy deve ser mantida em certo nível de segurança[3], entretanto, devido ao proxy não ser válido por muito tempo, não tem a necessidade de ser mantido tão seguro quanto a chave privada do proprietário. Assim, é possível armazenar a chave privada do proxy no sistema de arquivos local sem ser cifrada, contanto que as permissões do arquivo impeçam que qualquer indivíduo consiga vi-

[2] tecnologia de segurança que é comumente utilizada para codificar os dados trafegados entre o computador do usuário e um website.

[3] esta chave difere da chave privada do usuário e em geral fica localizada no diretório temporário da máquina da qual foi instanciado.

sualizar seu conteúdo. Uma vez que um proxy é criado e armazenado, o usuário pode usar o certificado do proxy e sua chave privada para a autenticação mútua sem incorporar uma senha.

Quando os proxies são usados, o processo mútuo de autenticação difere ligeiramente. O servidor remoto recebe não somente o certificado proxy (assinado pelo proprietário), mas também o certificado do proprietário. Durante a autenticação mútua, a chave pública do proprietário (obtida de seu certificado) é usada para validar a assinatura no certificado do proxy. A chave pública do CA é então usada para validar a assinatura no certificado do proprietário. Isto estabelece uma corrente da confiança do CA com o proxy através do proprietário.

5.1.4 Premissas do MyProxy

A combinação das exigências como, por exemplo, autenticação e delegação e limitações como dependência de um *Middleware* em comum, Globus Toolkit, por exemplo, na grade causou a necessidade de um sistema para se conseguir alguns destes objetivos citados e assim dependem para o funcionamento algumas premissas como segue [NOVOTNY01]:

- Deve permitir que os usuários alcancem suas credenciais em qualquer lugar da grade, mesmo se estiverem em um sistema sem software da grade e sem acesso seguro a suas credenciais de longo prazo.

- Deve permitir que deleguem credenciais aos recursos a que não normalmente podem, desde que as aplicações envolvidas não suportem o mecanismo do delegação do GSI [4], por exemplo, de um navegador de Internet a um portal.

- Deve remover, tanto quanto possível, todas as credenciais do portal exceto quando são realmente necessárias a fim de baixar o risco do portal ser comprometido.

[4] Globus Security Infrastructure

- Deve dar ao usuário tanto controle de suas credenciais e de credenciais do *proxy* quanto possível. Portais devem somente conter credenciais de usuário.

5.1.5 Conclusão do Capítulo

MyProxy se mostra uma ferramenta essencial na implantação de uma grade, a partir de sua idéia foi desenvolvido este trabalho, usando suas bibliotecas e comandos foi possível implementar o objeto desta dissertação.

Manter credenciais de curta duração disponíveis em um repositório, garante, até certo ponto uma segurança melhor que credenciais sendo manipuladas por tempo indeterminado e até mesmo sendo usadas por pessoas que não fazem parte dos usuários cadastrados.

No próximo capítulo será mostrado o trabalho desenvolvido, ou seja, todo o processo de criação de certificados de curta duração voltados à utilização por dispositivos móveis, usando o mesmo preceito agregado aos certificados de curta duração para usuários comuns.

Capítulo 6

Certificados Host de Curta Duração

Neste capítulo será mostrado o trabalho principal desenvolvido nesta dissertação. A idéia principal do tema, bem como a criação do código fonte todo implementado na linguagem de programação Java[JAVA06] teve como princípio a necessidade de se ter em uma grade computacional a automatização do sistema de inclusão de dispositivos móveis. Na seção 6.1 serão mostrados quais os requisitos de certificação para que dispositivos móveis possam se associar à grade. Na seção 6.1.1 será mostrada a descrição do trabalho proposto. Na seção 6.1.2 é trabalhada a idéia da necessidade de que somente usuários válidos possam requisitar certificados de host e na mesma seção é mostrado o funcionamento desta parte do código. Na seção 6.1.3 são mostrados os passos para se chegar a uma autoridade certificadora automatizada e na mesma seção o funcionamento do código. Na seção 6.2 são mostradas as classes confeccionadas e a função de cada uma. Na seção 6.3 são apresentadas algumas das tarefas que dispositivos móveis podem fazer associando-se à grade bem como outras vantagens e por fim, na seção 6.4.1 é apresentada uma breve conclusão deste capítulo.

6.1 Certificados para Dispositivos Móveis

Quando pensamos em uma grade computacional para dispositivos que permanecerão por pouco tempo associados à ela, é comum pensar que estes podem causar

uma falha de segurança na grade e na rede em que essa grade está conectada, e assim, utiliza-se a idéia de autenticação para uso da mesma. Se pensarmos como administradores da segurança de uma rede, a idéia é sem dúvida atrativa, mas também complexa do ponto de vista de desenvolvimento e administração dos recursos.

Se pararmos para pensar na associação de servidores, que não se moverão, nem trocarão seus endereços IPs, por exemplo, com certeza a idéia é atrativa, mas pensando pelo lado do cliente que deseja submeter determinado trabalho para processamento, este teria que fazer uma requisição de certificados como mostrado no capítulo 4 desta dissertação. Vale salientar que neste caso existe a necessidade de uma pessoa para assinar os certificados, se esta pessoa não estiver disponível, o certificado pode demorar para ser assinado, podendo tornar o uso da grade inviável para este solicitante.

A idéia deste trabalho então é permitir que a criação e assinatura de certificados possa ser feita de uma maneira automatizada, sem que o gerente de certificação da autoridade certificadora (CA) precise tomar conhecimento da requisição e assinatura. Mas como impedir que pessoas mal intencionadas se utilizem desta facilidade?

1. Uma necessidade prevista foi a de garantir a solicitação de certificados de máquina somente por usuários válidos para a grade, ou seja, aqueles usuários que possuem certificados de usuários válidos, não expirados e com senha para criação do *proxy* na máquina que for usada para a solicitação.

2. O segundo problema foi com respeito à assinatura do certificado e envio das chaves públicas e privadas para os hosts.

3. O terceiro problema foi a definição de tempo de validade máximo para os certificados solicitados, para este caso, foi considerado que sete dias de validade seria suficiente, como o que ocorre com o servidor MyProxy, pois, se um usuário tentar burlar a segurança da grade e do programa, isto poderia ser feito por um período de uma semana e como uma grade tem um monitoramento, em geral contínuo, podería-se avaliar prejuizos, e neste caso, cancelar o certificado do usuário que fez a requisição e tomar as atitudes definidas pelo administrador

da grade.

4. Outro ponto importante foi a confecção de um código que pudesse ser usado em qualquer plataforma operacional. Mesmo sendo a linguagem Java (independente de plataforma) a utilizada, alguns comandos do sistema operacional e do próprio *middleware* seriam necessários.

Manter a grade em condições "dinâmicas" de uso passou a ser um dos focos principais deste trabalho e este ponto será trabalhado nas seções seguintes.

6.1.1 Descrição do Trabalho

Este trabalho tem a idéia de resolver um problema específico da grade computacional baseada no *middleware Globus Toolkit*, que é o fornecimento dinâmico de certificados de curta duração para dispositivos, que se deseja que sejam interligados à grade por um curto período de tempo, em geral para interagir com esta grade, ou seja, não é uma solução definitiva, mas em associação com outros aplicativos, pode melhorar a velocidade na geração de certificados e inclusão de novos nós na grade.

A Figura **??** mostra o funcionamento da solução proposta, para melhor identificar os passos seguidos, a figura foi enumerada como segue:

1. Uma grade pode ter diversos usuários associados a ela;

2. Estes usuários estão interligados a diversas máquinas como pode ser visto na figura 6.1, as máquinas podem fazer parte de um domínio ou serem máquinas individuais em uma rede conectada à grade;

3. Todos os usuários que desejam submeter suas tarefas à grade devem ter seu certificado, válido e não expirado, este certificado fica em posse de cada usuário com as chaves públicas e privadas associadas à ele;

4. As máquinas que fazem parte da grade devem possuir certificados também válidos, no mesmo padrão dos certificados de usuários;

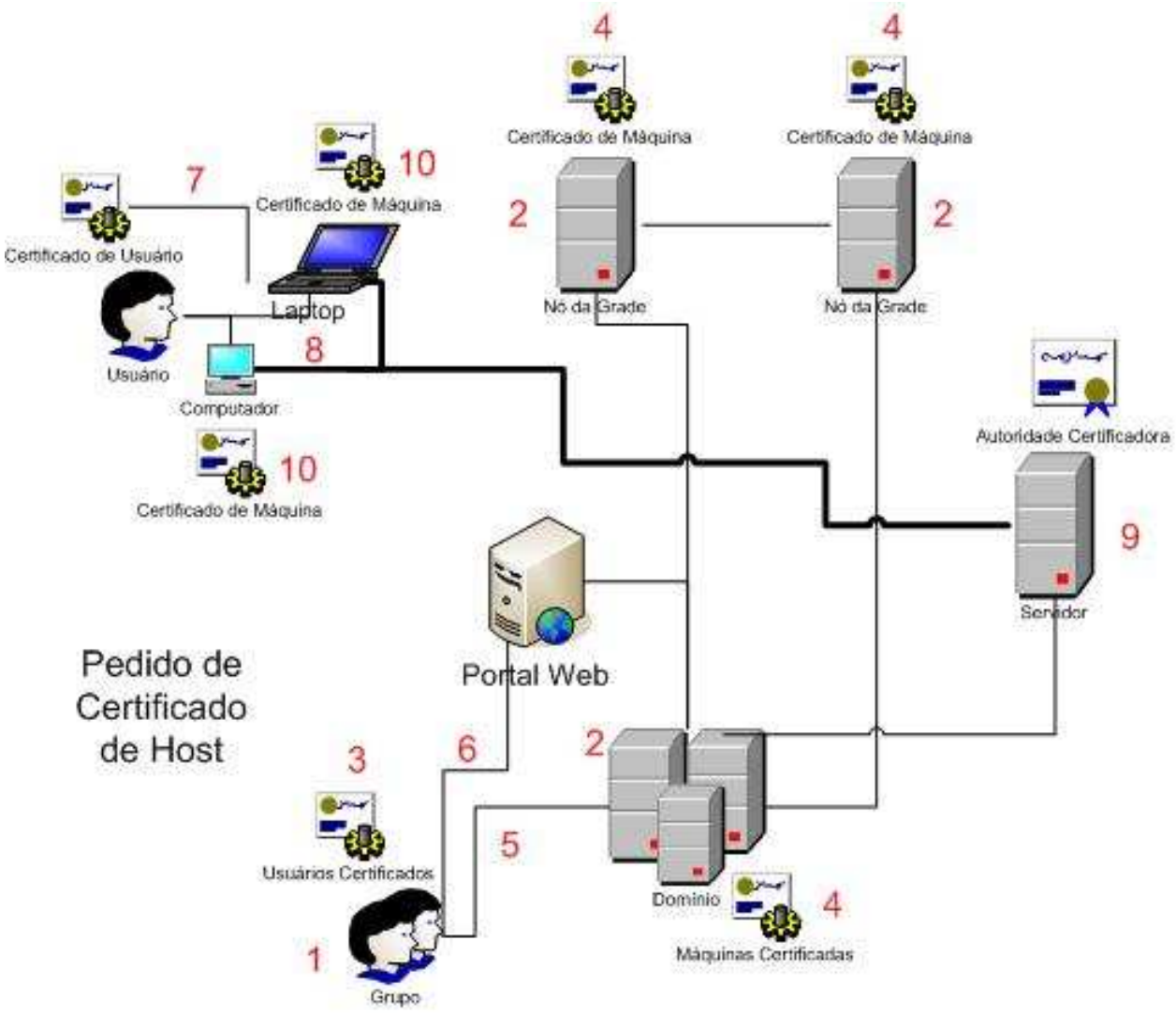

Figura 6.1: Descrição do funcionamento.

5. Para submeter uma tarefa para ser executada na grade, os usuários podem se conectar às máquinas certificadas e, de posse de seu certificado de usuário fazer a submissão;

6. As submissões também podem ser feitas através de portais *web* interligados à grade computacional;

7. Neste trabalho, um usuário, de posse de seu certificado válido, pode criar um *proxy* na máquina a qual fará a requisição de um certificado temporário de máquina, podendo ser esta máquina móvel ou não;

8. Através da interface criada na implementação deste trabalho, o usuário pode

fazer a requisição de certificados temporários de host pela interface gráfica ou texto;

9. O software se conectará à autoridade certificadora da grade usando o protocolo RMI(Remote Method Invocation)[RMI06] que gerará o certificado com o nome da máquina cliente, o assinará e retornará para o solicitante usando um outro canal de dados que usa SSL(Secure Sockets Layer) para criptografar a mensagem. Para que a conexão SSL seja concluída, é necessário que o usuário digite a senha de seu certificado, que será validada no proxy local e no certificado no servidor remoto;

10. Quando o programa recebe o certificado temporário, ele o grava no diretório atual e dá as instruções para que o usuário proceda a colocação deste certificado assinado no diretório correto.

6.1.2 Solicitação de Certificados por Usuários Válidos

A garantia de solicitação por usuários válidos de certificados de hosts foi um ponto importante para não abrir mão da segurança. Como passo fundamental neste caso pensou-se que quando um usuário deseja submeter uma tarefa à grade, este precisa de um certificado de usuário. Este certificado passou então a ser o ponto fundamental para permitir o uso deste serviço pelos usuários.

O certificado de usuário pode ser passado ao mesmo diretamente através de uma assinatura da CA pelos meios manuais como foi mostrado no capítulo 4 desta dissertação, ou solicitada através do servidor MyProxy mostrado no capítulo 5. Indiferente da maneira como foi solicitado, é necessário que para que o usuário se utilize dos recursos da grade, que este crie um *proxy* que é a criação de uma instância de abertura da grade para as máquinas remotas.

Quando este *proxy* é criado, independente da maneira, ou seja, através do *middleware Globus Toolkit* ou através do MyProxy, ele possui algumas informações sobre o usuário que solicitou o recurso, estas informações são consideradas como o identificador do usuário e para que este usuário possa acessar outras máquinas na grade,

não basta que ele tenha criado o proxy, é necessário que a permissão esteja explícita em um arquivo denominado *grid-mapfile*. Esse arquivo é responsável nas máquinas remotas por fazer um link entre o identificador do usuário e um usuário válido na máquina remota. Esta foi a chave pensada para resolver o problema da garantia de que o usuário solicitante era mesmo quem dizia ser e que o mesmo era um usuário válido para efeito de criação e assinatura do certificado. Entretanto uma possível tentativa de quebra da segurança poderia ser aplicada aqui por um usuário mal intencionado.

Como o identificador do usuário é uma *string* de um formato padrão, outros usuários poderiam de posse desta informação criar um janela para requerer um certificado como sendo um usuário válido. A solução foi criar um servidor que usa SSL para a transmissão do certificado no canal de dados, ou seja, mesmo que um usuário se passasse por um usuário válido, este só conseguiria gerar o certificado, mas este certificado ficaria no repositório aguardando que novamente seja feita uma autenticação para enviar ao solicitante. Se este não o fizer no prazo de 30 segundos o certificado é apagado do repositório e é necessário refazer a requisição. Este passo é importante também por que assim, caso aconteça problemas com a conexão, por exemplo, o certificado não ficaria exposto.

Funcionamento

Tomando como base a idéia anterior, podemos mostrar o funcionamento definido como segue:

1. Primeiramente o usuário cria seu proxy na máquina local, independente de ser a máquina para a qual deseja solicitar o certificado;

2. Após a criação do proxy, o mesmo fica normalmente válido por um período de 12 horas e possui o seguinte formato:

   subject : O=Grid/OU=org/OU=AC/OU=domain/CN=fulano/CN=99999
   issuer : /O=Grid/OU=org/OU=AC/OU=domain/CN=fulano
   identity : /O=Grid/OU=org/OU=AC/OU=domain/CN=fulano

type : Proxy draft (pre-RFC) compliant impersonation proxy
strength : 512 bits
path : /tmp/x509up_u1010
timeleft : 12:00:00

3. De posse da entrada acima, é possível pegar o *identity* que é o identificador do usuário e submeter ao servidor de assinatura de certificados de máquina;

4. Quando este identificador chega ao servidor, o mesmo é comparado com as entradas do grid-mapfile que é um arquivo que contém os identificadores de todos os usuários que podem usar a máquina onde está e o usuário local para o qual os processos serão executados, e, somente se a entrada for idêntica à recebida o processo se completa;

5. Caso contrário o processo é encerrado e é enviado um erro ao solicitante.

6. Uma vez o usuário sendo considerado válido, a senha que o mesmo digitou no cliente é usada agora para criar uma instância de uma conexão com o servidor usando SSL, se o certificado for válido, o servidor enviará o certificado temporário de máquina para o solicitante usando esse canal de dados.

6.1.3 Autoridade Certificadora Automatizada

Outro problema encontrado e tratado neste trabalho foi a assinatura dos certificados usados pela grade. O primeiro passo foi o uso de assinatura com os comandos do próprio middleware globus[GLOBUS06]. A dificuldade encontrada nesse ponto foi que a utilização das ferramentas do globus, primeiro tornava o software dependente da arquitetura, o segundo problema é que o *Globus*, apesar de ser de código aberto, modificar seu conteúdo torna-se uma tarefa árdua e possivelmente insegura, devido a possíveis falhas que podem ser incluídas no código sem que se perceba, isto acontece normalmente dada a dificuldade de se modificar código do qual não participamos do desenvolvimento, mesmo estando este muito bem documentado como é o caso do *Globus*.

A solução a este problema surgiu da idéia de deixar o *middleware* de lado e passar a utilizar openssl[1][OPENSSL06], que também é usado pelo *Globus*, para assinar os certificados diretamente, sem a intervenção do *middleware*, por exemplo, esta implementação *open source* do SSL, permite que sejam passados parâmetros, que através do *Globus* não seriam possíveis. Vale lembrar de situações que também serão favorecidas como, por exemplo, a independência de arquitetura, *middleware* ou sistema operacional.

Funcionamento

Todo o processamento foi feito diretamente no servidor MyProxy e o desenvolvimento e funcionamento foram feitos como segue:

1. O usuário válido, ou seja, com certificado válido, não expirado e protegido por senha faz a requisição via terminal através da classe Cliente ou graficamente através da classe FrameCliente, implementadas em java;

2. Na requisição, além de seu identificador vão o nome do servidor MyProxy, o nome da máquina cliente para o qual deseja-se solicitar o certificado e a quantidade de dias para o certificado expirar;

3. No campo nome da máquina cliente é possível que se coloque o endereço IP do mesmo caso a máquina não esteja registrada no DNS[2];

4. No campo quantidade de dias do certificado, vale lembrar que o prazo máximo para a requisição é de sete dias, período esse considerado seguro do ponto de vista de uso da grade, já que dispositivos móveis para os quais esta implementação se destina raramente utilizam a grade por tempo maior que esse, e, caso necessite de mais tempo, de acordo com o administrador da grade é possível modificar esse tempo ou então fazer uma nova requisição sempre que for necessário, com isso garante-se de certa forma que, no caso de dispositivos

[1] Uma implementação livre do protocolo SSL. Para maiores detalhes veja anexo D

[2] Domain Name System - Responsável por fazer a resolução de nomes

móveis, o certificado não estaria válido por um tempo considerado grande o suficiente para prejudicar a segurança da rede;

5. Uma vez feita a requisição, todo o trabalho fica por conta do servidor no qual MyProxy está executando, este que fará a geração do certificado de host, assinatura da chave pública, limpeza do lixo[3] e devolução ao solicitante da chave privada e do certificado através do protocolo SSL, além de armazenar um log com o identificador do usuário solicitante, o número de dias solicitados, o nome da host cliente e a data e hora, com isso, seria possível manter uma certa administração sobre as criações de certificados e punir um usuário que usar este serviço de forma ilícita ou fora do permitido pelas políticas de uso;

6. Quando o cliente recebe os certificados, estes são gravados em arquivo e ao cliente basta colocar no local especificado pelo *middleware* utilizado (em geral em /etc/grid-security) trocando as permissões dos arquivos para as permissões padrão (em geral 644 para o certificado e 400 para a chave privada). Isso levando em consideração o sistema operacional derivado do Unix (linux, por exemplo).

7. Concluidos esses passos a máquina está associada à grade e pode submeter seus trabalhos diretamente à mesma, receber tarefas para processar e até mesmo interagir diretamente.

6.2 Desenvolvimento do Código

O código fonte deste projeto foi todo desenvolvido na linguagem de programação Java, sua escolha foi principalmente devido à necessidade de utilização do servidor de geração de certificados para hosts independente da plataforma de hardware e software, já que essa é uma das premissas da utilização e funcionamento das grades. Java é por natureza uma linguagem que oferece este tipo de suporte.

O código foi dividido em dez classes e o método de comunicação usado na rede

[3] Arquivos de requisição dos certificados que não serão mais usados

para a requisição foi o RMI[4] e para a distribuição dos certificados foi usado SSL. As classes foram divididas da seguinte forma:

1. **Classe Servidor** - Classe que cria uma instância do RMI na máquina e aguarda requisições. Esta classe é usada para implementar a interface que será lida pelos clientes;

2. **Classe Interface** - É um tipo de classe onde existem os métodos que poderão ser acessados pelos clientes e implementados pelo servidor;

3. **Classe Implementacao** - Classe responsável pelos métodos que lerão as requisições dos clientes e de acordo com elas, implementará a interface através da classe servidor, é ela também a responsável por comparar as permissões de uso e gravar o log de uso do serviço de solicitação de certificados temporários de host. É a partir desta classe que são gerados os esqueletos de classe necessários ao protocolo RMI[RMI06] que serão de conhecimento tanto do cliente quanto do servidor;

4. **Classe Limpeza** - Esta classe é responsável por fazer backup de arquivos utilizados durante a criação e assinatura dos certificados de hosts e após isso, fazer a exclusão de arquivos que não são mais necessários. Quando se faz a requisição de um certificado de host, este gera três arquivos, um é a chave pública que é o pedido propriamente dito, o segundo é a chave privada do host e o terceiro é o certificado assinado pela autoridade certificadora. Se estes arquivos permanecerem no diretório local e como o pedido de certificado pode ser feito por cada usuário para diversos hosts, isso pode gerar grande quantidade de arquivos que não serão mais usados após terem sido passados a seus solicitantes. Com essa classe é possível fazer uma limpeza deste diretório, excluindo os arquivos e retornando os arquivos de backup ao estado original da autoridade certificadora da grade.

[4]Invocação remota de métodos baseada na linguagem Java que usa para comunicação entre processos uma interface também em Java

5. **Classe Cliente** - Esta classe é a que junto ao esqueleto da classe Implementacao gerada pelo comando rmic[5] fará com que o a máquina local se conecte ao servidor MyProxy e faça a requisição de um certificado de host. Esta classe pede três parâmetros, o primeiro o endereço ou IP do servidor, o segundo o nome ou IP do cliente e o terceiro o número de dias que se deseja que o certificado tenha validade. Quando a classe é instanciada ela consulta se o proxy do cliente está ativo e pega o identificador deste cliente para comparar com um cliente válido no servidor, caso não seja válido é retornado um erro.

6. **Classe FrameCliente** - Esta classe é similar à Cliente, só que neste caso é possível se utilizar de uma interface gráfica para a requisição do certificado, facilitando o entendimento e uso por usuários não especializados.

7. **Classe SSLCertHost** - É a classe que cria um servidor SSL que irá fazer a entrega do certificado assinado ao solicitante, desde que este possua um certificado válido.

8. **Classe SSLKeyHost** - É a classe que cria um servidor SSL similar ao anterior, o que diferencia é que este envia a chave privada ao solicitante, desde que este possua um certificado válido.

9. **Classe sslClientCert** - Esta classe é a instanciada pelos clientes para que seja recebida a chave assinada através do servidor SSLCertHost. Esta classe também é responsável por gravar o arquivo hostcert.pem no diretório.

10. **Classe sslClientKey** - Esta classe é a instanciada pelos clientes para que seja recebida a chave privada através do servidor SSLKeyHost. Esta classe também é responsável por gravar o arquivo hostkey.pem no diretório.

[5] o compilador rmic cria os stubs e os skeletons. Ele pega o arquivo como um argumento e produz arquivos .class da forma arq_SKEL.class e arq_STUB.class

6.3 Vantagens em Fazer Parte da Grade

Se considerarmos uma grade computacional "comum", ou seja, com certificados de longa duração e com a tarefa de criação de certificados não automatizada, tomando como exemplo um usuário que resolvesse fazer uma submissão de tarefas para a grade, sendo um usuário válido e com certificado, este só poderia submeter suas tarefas, caso a sua máquina não pertença à grade, a partir de um terminal, conectando-se à uma máquina com certificado de host válido, pertencente à grade e só aí poderia executar suas aplicações.

Em um processo de renderização de imagens, por exemplo, este usuário precisaria incluir sua máquina na grade para interagir com os processos, se esse usuário não possuísse mais interesse em usar a grade, ou seja, essa submissão e interação poderiam ser feitas uma única vez, e como os certificados seriam de longa duração, este certificado poderia ficar sem uso mas continuando válido por um longo tempo. Ou em outra hipótese, o usuário decidir de última hora incluir um novo nó na grade para submeter seus trabalhos diretamente e interagir com os processos, mas se o gerente da autoridade certificadora não estiver disponível, a demora na assinatura da chave pública poderia ser intolerável à utilização deste usuário.

Com a criação e assinatura automátizada de certificados de host temporários, os dispositivos móveis se beneficiariam desta estrutura, bem como os usuários que poderiam incluir seus dispositivos sempre que necessário para apresentar em uma palestra os resultados de um processamento, por exemplo.

Outra vantagem no uso destes certificados seria, que poderia haver um sistema implementado de forma que verificaria com qual frequência um usuário dos certificados temporários está se associando à grade e com quais máquinas, e desta forma prover um serviço *vip* a este, desde que o mesmo permita que sua máquina seja usada por outros usuários quando esta máquina estivesse disponível (estas consultas de recursos e disponibilidade poderiam ser adquiridas através de softwares de monitoramento ou mesmo do MDS mostrado na seção 3.1.7). Similar ao que ocorre com

o BOINC[6] (Berkeley Open Infrastucture For Network Computing).

Se um usuário estivesse se utilizando da grade com suas máquinas de maneira ilícita ou contradizendo o proposto pelo "contrato de uso" da grade, este poderia ter seu certificado de usuário cancelado e conseqüentemente de todas as suas máquinas, isso seria fácil de ser feito pois o servidor de certificados temporários de host possiu a geração de log de uso que pode ser exportado para uma planilha, onde constam: Identificador do solicitante do certificado, máquina cliente, quantidade de dias (firmando que o máximo seriam sete dias) além de data e hora do pedido. Em uma planilha, poder-se-ia inclusive gerar um gráfico de uso da grade por dispositivos temporários monitorando o crescimento na utilização da grade para levantamentos estatísticos, por exemplo.

Com a validade dos certificados por apenas sete dias, se o usuário decidir ou suspeitar que o tempo de execução de sua tarefa será maior que este prazo o mesmo pode fazer uma nova solicitação e apenas substituir o certificado sem que sua tarefa seja interrompida, já que o processo de verificação das chaves é feito somente na hora que um serviço é submetido. Isso principalmete ajuda na segurança, pois os certificados "esquecidos" perdem a validade em um prazo muito curto e se o Administrador da grade decidir que o tempo é grande ou pequeno demais, este pode alterar o prazo de validade dos certificados de acordo com seu interesse.

6.4 Implementação

Durante o desenvolvimento deste código, pensou-se nas facilidades que deveriam conter e como garantir a segurança na grade, definindo que uma das políticas adotadas deveria ser a de que somente usuários com certificados válidos pudessem fazer a requisição de certificados de máquina.

Dentre as diversas possibilidades de se conseguir uma autenticação, a adotada foi a de que um usuário só é válido na grade se possui um certificado e consegue

[6] Uma plataforma, na forma de framework, que visa facilitar a implementação de sistemas de computação distribuídos, funcionando através de um grid computacional de dimensões mundiais.

gerar um proxy usando a senha do certificado para submissão de suas tarefas, assim sendo, antes que o usuário faça a requisição de seu certificado, este deve gerar um proxy (através do comando grid-proxy-init ou via MyProxy mostrado no Capítulo 5). Após o proxy ser criado, o programa verifica o identificador único do usuário que fez a requisição e compara no servidor com os usuários que podem usar a grade. Se este for um usuário válido, ele poderá requisitar o certificado.

O aplicativo servidor roda na máquina remota, onde está a autoridade certificadora na porta 7513 enquanto que os servidores SSL para envio das chaves, ou seja, SSLCertHost e SSLKeyHost são executados nas portas 7514 e 7515 respectivamente. Sempre que o servidor é colocado em funcionamento é pedida a senha da CA. É este servidor, desenvolvido nesta dissertação, que fará a implementação da interface que será lida pelo cliente.

Dois clientes foram implementados, um para os usuários que quiserem fazer suas requisições de certificados temporários de host através de uma interface em modo texto, e neste caso, o usuário deverá passar os parâmetros da requisição, como endereço do servidor, nome do cliente e quantidade de dias que o certificado será válido. Esta quantidade é verificada tanto no cliente quanto no servidor, para evitar que uma modificação do código cliente acarrete em uma falha de segurança, caso o cliente requisite um certificado com mais de sete dias de validade (valor considerado seguro). As Figuras 6.2 e 6.3 mostram o funcionamento da interface texto para solicitação de certificados temporários de máquinas e a Figura 6.4 mostra uma requisição de certificado temporário de máquina bem sucedido.

No anexo E podem ser vistos os certificados gerados nesta solicitação bem como os certificados de usuário e o proxy de usuário, bem como todas as chaves públicas e privadas associadas a estes.

O segundo cliente é uma interface gráfica com diversas funcionalidades, como por exemplo um botão de ajuda e um indicador de status do programa. O funcionamento desta interface pode ser visto nas Figuras 6.5, 6.6, 6.7 e 6.8. Na Figura 6.9 é mostrada uma requisição de certificado temporário de máqina bem sucedido. Na Figura 6.10 é mostrada a tela de ajuda após ter sido pressionado o botão de ajuda e na Figura

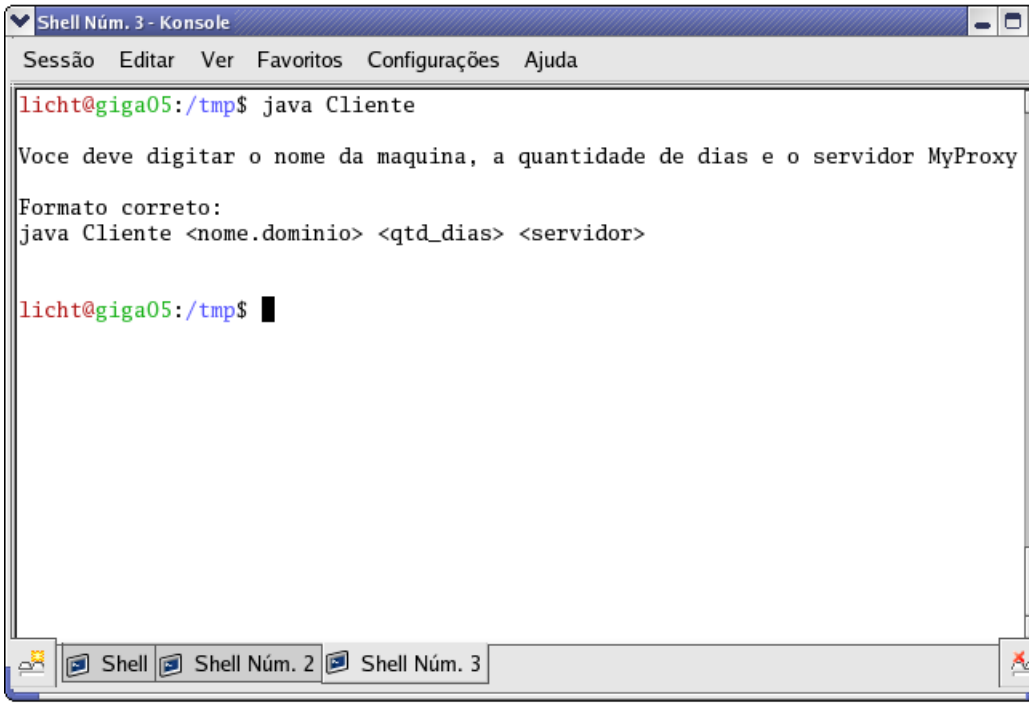

Figura 6.2: Cliente em modo texto (tela 1).

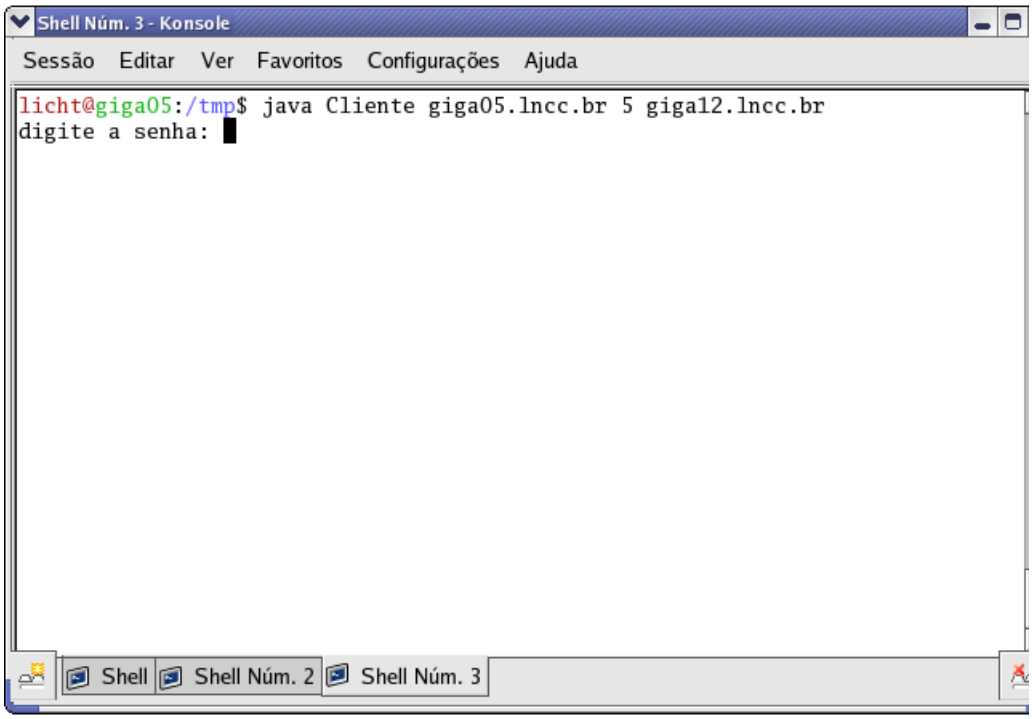

Figura 6.3: Cliente em modo texto (tela 2).

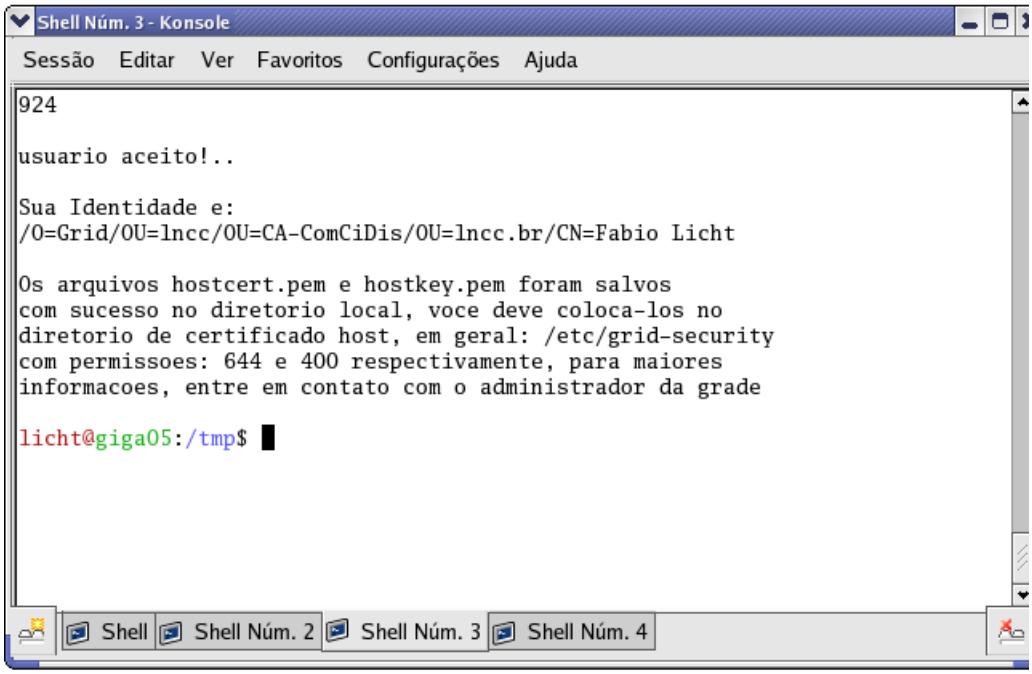

Figura 6.4: Cliente em modo texto (requisição).

6.11 o resultado do pressionamento do botão sobre.

Como pode ser visto nas Figuras, tanto a interface texto quanto a interface gráfica possuem tratamento de erro, bem como telas de ajuda que facilitam a operação por usuários que não possuem um bom conhecimento da grade, estas facilidades podem ser adicionadas a outras soluções para prover outras funcionalidades.

6.4.1 Conclusão do Capítulo

Um ponto importante desta implementação foi a autenticação do usuário que solicita o certificado de máquina por diversas vezes, o que fez com que uma requisição mal intecionada ou a tentativa de quebra da segurança do sistema fosse mais complexa de ser atingida. Vale lembrar que esta não é uma solução definitiva, mas como os módulos trabalhados estão bem documentados, é possível que outros desenvolvedores utilizem as classes associadas ao projeto para desenvolvimento de novas ferramentas, que aprimorem a segurança e automação das grades computacionais.

Figura 6.5: Cliente em modo gráfico (tela 1).

Figura 6.6: Cliente em modo gráfico (tela 2).

Figura 6.7: Cliente em modo gráfico (tela 3).

Figura 6.8: Cliente em modo gráfico (tela 4).

Figura 6.9: Cliente em modo gráfico (requisiçao).

Figura 6.10: Cliente em modo gráfico (tela de ajuda).

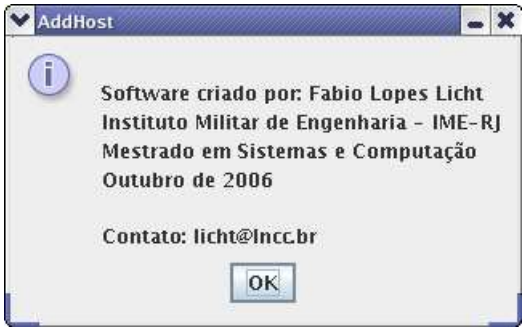

Figura 6.11: Cliente em modo gráfico. (tela sobre).

No capítulo seguinte serão mostrados alguns trabalhos relacionados seguido do Capítulo de conclusão onde também serão mencionados alguns trabalhos futuros e as contribuições deste projeto.

Capítulo 7

Trabalhos Relacionados

Devido ao recente crescimento do uso de grades computacionais e ao estudo destas estar em fase inicial, pouco material relacionado foi encontrado, que desse base na utilização de certificados de curta duração para dispositivos móveis que utilizam globus toolkit como middleware para inclusão nas grades. Entretanto alguns trabalhos se assemelham por tratar de assuntos como certificação temporária para usuários, estes poucos trabalhos encontrados estão descritos como segue:

MyProxy foi a base para a idéia da criação deste serviço descrito nesta dissertação, este serviço descrito no capítulo 5 trata diretamente do fornecimento de certificados proxy temporários para usuários que desejam submeter tarefas à grade. MyProxy é descrito em [NCSA06] e sua similaridade com o desenvolvimento desta dissertação inclui somente o fornecimento de certificados temporários, pois, enquanto que no servidor MyProxy as credenciais de usuário precisam existir no repositório e apenas um proxy temporário é criado na máquina local, no servidor proposto nesta dissertação não é necessário que as máquinas sejam pré-cadastradas para terem acesso à grade como nó de submissão ou interação. O certificado proposto é criado no instante em que for necessário e desta forma torna-se um serviço mais automatizado. Entretanto não se pode descartar a importância de MyProxy para esta dissertação, tanto que este teve um capítulo destinado a seu intendimento, pois dele, além de vir a idéia inicial, pode ser usado para criar o proxy que torna possível a requisição de um certificado temporário de máquina, mesmo tendo os caminhos

seguidos para a conclusão diferentes.

Outro trabalho que tem certa influência nesta dissertação, mas que tem muito mais similaridade com MyProxy que com essa dissertação é descrito em [WELCH05], onde também é tratada a criação de proxys temporários para usuários. Este trabalho cita inclusive o globus tookit e o próprio MyProxy para criação dos proxys.

Em [BHATIA05] outra abordagem é dada ao mesmo serviço de criação de proxys temporários, entretanto neste trabalho é descrita uma ferramenta em que se pode fazer a requisição e criação de proxys através de portais Web, como: portais desenvolvidos em java ou mesmo Portlets[1] para uso em aplicações cliente específicas. Uma vez criados os proxys nas máquinas remotas, a submissão de tarefas podem ser feitas diretamente do portal, entretanto a máquina cliente continua não fazendo parte da grade para interação entre os trabalhos em execução.

[1] Componentes padronizáveis que aparecem como pequenas janelas nas páginas dos portais web, conectando os usuários a aplicações. Funcionam como elementos básicos dos portais, incorporando ampla variedade de conteúdos, podendo ser manipulados pelos usuários com facilidade.

Capítulo 8

Conclusão

Este trabalho teve como princípio a idéia de automatização dos serviços da grade, principalmente da geração de certificados, que têm sido um dos "calcanhares de aquires" do uso da grade computacional. Sempre que se pensava em associação à grade notava-se uma grande dificuldade, principalmente para incluir máquinas ou dispositivos móveis por curto período de tempo. Para cada nó que quisesse aderir à grade era necessário que um gerente de certificação da autoridade certificadora assinasse o certificado solicitado. Como a tarefa, neste caso é uma tarefa "manual", ou seja dependente diretamente de uma pessoa, se este não estivesse disponível, ou se a rede ou meios de comunicação (email, por exemplo) estivessem "fora do ar"(indisponíveis), o certificado poderia demorar tempo intolerável para ser assinado. Imaginando uma entidade móvel (um laptop ou outros) que decidisse usar a grade para interagir com uma aplicação durante uma apresentação em um evento, por exemplo. Quando este estivesse com o certificado em mãos, poderia não ser mais útil, isso, sem imaginar o constrangimento que poderia causar.

Outro problema que havia é que os certificados gerados, para as máquinas que seriam temporariamente usadas ficavam à disposição do solicitante e como o mesmo não usaria mais a grade, poderiam os certificados cair em mãos erradas e serem usados para fins ilícitos.

Com esse trabalho, foi possível criar uma estrutura mais automatizada onde,

apesar de ter um controle menos pessoal e mais dependente de administração, seja possível visualisar em qualquer tempo o crescimento no uso da grade, verificando o log de utilização do serviço de geração e fornecimento de certificados temporários de host. Com esse log gerado é possível até mesmo, com o auxílio de uma planilha, visualisar o crescimento gráfico ou mesmo saber quem são os usuários que mais solicitaram certificados e assim, se estes contribuirem para o crescimento da grade, tornem-se usuários privilegiados (VIP's).

A utilização do protocolo RMI (Remote Method Invocation) foi outro ponto discutido como sendo importante, pensa-se no futuro em utilizar autoridades certificadoras hierárquicas de forma distribuída, ou seja, ter diversas entidades que pudessem assinar os certificados, e assim sendo é possível através do protocolo RMI fazer uma programação paralela em Java mais facilmente, transportando o código desenvolvido sem ter que modificar toda a estrutura.

É importante destacar que esta aplicação foi desenvolvida para utilização em uma arquitetura bem definida e para uma aplicação fechada, ou seja, foi projetada uma solução para fornecimento de certificados de curta duração para dispositivos móveis na inclusão destes em uma grade computacional baseada no middleware globus toolkit para interação com aplicações submetidas para processamento, por exemplo, quando se deseja interagir em uma renderização de imagens.

Quando uma máquina precisa interagir diretamente com as tarefas submetidas usando o globus toolkit é imprescindível que esta faça parte da grade computacional, exemplos de visualização gráfica de andamento dos processos são comuns. Em casos normais, em que os processos não interagem com uma máquina na qual se deseja verificar o andamento do processo ou resultado deste é necessário que se colete todos os arquivos decorrentes do processamento e faça o cálculo sobre estes. Com esta aplicação, por tornar possível a inclusão de dispositívos móveis ou visitantes por um curto período de tempo, é possível fazer parte do processamento ou interagir com ele diretamente, desde que este tenha o middleware globus instalado e um certificado temporário válido.

A interação direta com aplicações clientes foi o foco deste trabalho, a inclusão de

dispositivos temporários mostrou-se como uma necessidade crescente devido principalmente à visualização dos resultados. Esperar que uma aplicação termine seu processamento para somente neste ponto verificar o resultado pode demandar muito tempo, já que a aplicação pode estar errada e mostrar resultados não satisfatórios. Através de verificações constantes, ou seja, interagindo com a aplicação, é possível corrigir erros, reformular problemas, ou simplesmente usar parte do processamento para cálculo ou estimativa do tempo de execução total. Verificar todo o andamento graficamente é outro ponto muito interessante que esta aplicação pode permitir.

Vale lembrar que a aplicação desenvolvida neste trabalho não é a solução definitiva, alguns trabalhos ainda podem ser incluídos para que se possa torná-la mais segura do ponto de vista de segurança. Algumas modificações do código podem ser feitas com o intúito de prover uma maior privacidade das chaves pública e privada.

Como esta aplicação foi desenvolvida para que se possa fazer requisições de certificados temporários de máquinas, independente de arquitetura (hardware ou software) de uma maneira automatizada, pensou-se em fornecer os certificados através da rede, assim sendo, um observador monitorando a rede, poderia coletar os certificados e de posse destes usá-los de forma não apropriada. Vale lembrar que neste caso, como são certificados temporários, com validade máxima de sete dias, em um curto período de tempo estes perderiam a validade tornando-se inúteis. Como para uma nova requisição de certificados é necessário que se utilize do certificado de usuário, válido e para isso é necessário a senha do usuário, caso um certificado seja utilizado ilícitamente, por exemplo, é possível punir o responsável pela requisição, já que o registro de requisição estará disponível para consuta pelo administrador da grade.

Tomando os prós e contras da aplicação, podemos pensar em novas alternativas para os problemas mencionados, entre eles a inclusão desta aplicação em pacotes de instalação do globus toolkit, e assim sendo, seria possível fazer a requisição do certificado na própria máquina para a qual se deseja o certificado temporário, e ao invés de requisitar os certificados com suas chaves públicas e privadas, seria possível gerar a chave pública e privada na máquina local e enviar somente a chave pública para a assinatura, tornando a aplicação muito mais segura.

8.1 Contribuições

As contribuições deste trabalho atualmente estão direcionadas para fins acadêmicos e estudantes da área que quiserem aprofundar-se em certificação digital para grades baseadas no *middleware Globus Toolkit*, além de fornecimento de recursos e delegação para estas grades, entretanto um projeto da RNP (Rede Nacional de Pesquisa) incluirá este trabalho em uma proposta de prover de uma maneira automatizada o acesso à grades computacionais à instituições e pessoas que desejarem processar seus trabalhos.

O trabalho denominado VCG (*Virtual Community Grid*) [VCG07] um dos GTs (Grupos de Trabalho) da RNP 2006/2007 tem prazo de um ano para ser concluído e visa tornar a grade um serviço mais acessível a usuários que possuem a necessidade de processamento paralelo de alta capacidade a um baixo custo. A implementação deste trabalho de geração de certificados temporários para máquinas será anexado a outros códigos de forma a prover um sistema seguro, automatizado e funcional para entidades que desejam fazer parte da grade.

8.2 Trabalhos Futuros

Como trabalhos futuros, pretende-se desenvolver uma ferramenta similar a esta, para geração de certificados de usuários e não somente de hosts, a diferença é que enquanto esta ferramenta se baseia no certificado de usuário válido para gerar certificados de host, a próxima basear-se-á em uma estrutura de banco de dados onde os usuários poderão preencher cadastros e após aprovados manualmente, estes teriam uma cota para solicitar certificados de usuários ou máquinas, para uma instituição, por exemplo.

Além disso, pretende-se usar o log de uso dos certificados temporários para criar uma interface web, onde possa ser monitorado de maneira mais facilitada, toda a estrutura da grade, como usuários mais ativos, usuários com certificado expirado, quantidade de usuários logados atualmente, entre outras informações que se fizerem

necessárias para que se tenha um controle mais centralizado e mais funcional.

Outro ponto a ser trabalhado é o descrito na conclusão em que se propõe a inclusão desta aplicação nos pacotes de instalação do globus, tornando possível a criação de uma geradora de chaves locais que apenas serão assinadas na autoridade certificadora e não mais transmitidas pela rede diretamente.

Por fim, um trabalho também a ser estudado é o fornecimento automatizado para dispositivos fixos com certificados de longa duração e neste caso, aliar ao monitoramento de certificados. Desta forma será possível monitorar quando um certificado estiver vencendo sua validade e de uma forma análoga ao DHCP (Dynamic Host Configuration Protocol) que é um protocolo que fornece dinâmicamente IP‘s em uma rede e renova a concessão automaticamente, fazer a renovação da assinatura dos certificados.

Apêndice A

Funcionamento MyProxy

Este anexo tem a função de prover informação necessária ao bom funcionamento e utilização de MyProxy citado no Capítulo 5, este servirá como um manual de uso deste servidor, importante não somente para o uso do servidor em questão, mas também para o uso da solução proposta nesta dissertação.

A.1 Funcionamento

1. Os usuários podem armazenar e recuperar credenciais do proxy X.509 usando o myproxy-init e o myproxy-logon.

2. Os usuários podem armazenar e recuperar múltiplas credenciais a partir de "entidades finais"[1] usando myproxy-store e myproxy-retrieve.

3. Os administradores podem carregar o repositório com as credenciais X.509 no interesse dos usuários usando o comando myproxy-admin-load-credential.

4. Os administradores podem usar o comando myproxy-admin-adduser para criar credenciais de usuário e carregá-las no repositório MyProxy.

5. Os usuários e os administradores podem ajustar políticas de controle de acesso nas credenciais do repositório.

6. Se permitido pela política, os gerentes podem renovar credenciais antes que expirem.

A.2 Comandos MyProxy

Para um usuário e mesmo administrador é importante conhecer os comandos úteis ao serviço MyProxy e estes estão descritos como segue:

1. myproxy-init - armazenar uma credencial proxy para uma recuperação futura.

2. myproxy-info - informação sobre credenciais.

[1] entidades que fazem parte da grade mas não são autoridades certificadoras CA

3. myproxy-logon - recuperar uma credencial.
4. myproxy-store - armazenar a credencial em uma "entidade final" para uma recuperação futura.
5. myproxy-retrieve - recuperar uma credencial da "entidade final".
6. myproxy-destroy - remover uma credencial do repositório.
7. myproxy-change-pass-phrase - mudar a senha das credenciais no repositório.
8. myproxy-admin-adduser - adicionar uma nova credencial de usuário.
9. myproxy-admin-change-pass - mudar a senha de uma credencial.
10. myproxy-admin-query - mostrar índices do repositório.
11. myproxy-admin-load-credential - carregar diretamente do repositório
12. myproxy-server - armazenar credenciais em um repositório.

Apêndice B

Funcionamento do Trabalho

Este anexo tem a função de ajuda para a utilização do servidor e dos clientes foco desta dissertação fazendo um passo a passo do uso das classes implementadas.

Primeiramente é bom focar que para que o servidor de certificados temporários para host funcione adequadamente, é necessário que na máquina onde se deseja executá-lo haja uma autoridade certificadora em execução.

Passado o primeiro passo a próxima tarefa é compilar as classes com o comando:
javac *.java -classpath swing-layout-1.0.jar

Agora deve-se gerar os esqueletos de classe como segue:
rmic Implementacao
Este comando vai gerar dois arquivos Implementacao_Skel.class e Implementacao_Stub.class que deverão ser enviados para o cliente junto às classes Cliente.class ou FrameCliente.class.

Antes de iniciar o servidor é necessário que se inicialize o rmiregistry que é o serviço que gerencia os objetos registrados e é implementado por um programa que deve ser rodado em todos os servidores que disponibilizam objetos. Toda a ligação

é realizada através da classe Naming, que faz parte da API padrão do Java. Para tanto, execute o comando na porta em que se deseja rodar o servidor como segue:

rmiregistry 7513 &

A porta 7513 foi usada para seguir a ordem da porta do servidor MyProxy que está em execução na 7512. O & (e comercial) é usado para que o terminal onde foi executado não fique travado, forçando o programa a rodar em background.

Agora pode-se iniciar o servidor com o comando:

java Servidor

O programa solicitará que seja digitada uma senha, esta senha tem que ser a mesma usada pela autoridade certificadora. A exigência da senha aqui é para que não seja digitada no terminal como parâmetro e conseqüentemente entre para o histórico de comandos usados, colocando a senha exposta.

No cliente use o seguinte comando para importar o certificado gerado através do proxy:

keytool -import -alias SSLCertificate -keystore SSLStoreServer -file /tmp/x509up_u1010

Agora é possível que se execute o cliente de duas maneiras, ou via terminal ou então via interface gráfica. Se fosse executado pelo terminal texto, o seguinte comando seria necessário:

java Cliente <nome.dominio do cliente ou ip> <quantidade de dias> <nome.dominio do servidor ou ip>

É importante verificar que para que seja possível executar este comando você deve criar um proxy na máquina local com o comando grid-proxy-init ou usando um servidor MyProxy. Após executar o comando, será pedida a senha do proxy criado para certificar que o usuário está usando um proxy válido.

Na inteface gráfica o trabalho fica mais simples:

java FrameCliente -classpath swing-layout-1.0.jar
Assim como no cliente em modo texto, antes de executar o cliente, crie um proxy com o comando grid-proxy-init ou através do MyProxy.
Após executado o cliente, uma interface gráfica será iniciada e nela é pedido cada um dos parâmetros, caso um destes falte o próprio software se encarrega de avisar ao usuário das dependências ou erros.

Vale lembrar que em ambos os clientes (texto ou gráfico) a requisição gerará dois arquivos, um é o hostcert.pem (certificado host) e o outro é o hostkey.pem (chave privada do host) estes arquivos devem ser colocados no diretório /etc/grid-security (normalmente) e então, deve-se verificar as permissões dos mesmos de forma que fiquem com as seguintes:

-rw-r–r– 1 root root 2603 Sep 25 11:19 hostcert.pem
-rw——- 1 root root 887 Sep 25 11:18 hostkey.pem

Os seguintes comandos do linux podem ser usados para que se consiga essas permissões:
chmod 644 hostcert.pem (permissão de leitura e escrita para o dono e somente leitura para outros)
chmod 400 hostkey.pem (leitura para o dono e nenhuma permissão para outros)

Terminados estes passos sua máquina estará fazendo parte da grade através do *middleware Globus Tookit* por um período igual ou inferior a sete dias.

Apêndice C

Código Fonte

C.1 Servidor.java

```
import java.io.BufferedReader;
import java.io.FileWriter;
import java.io.InputStreamReader;
import java.rmi.Naming;
/*
 * Servidor.java
 *
 * Created on 18 de Outubro de 2006, 14:37
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */
/**
 *
 * @author licht
 */
public class Servidor {

    /** Creates a new instance of Servidor */

    public Servidor() {
        try {
            byte password[] = new byte[10];

            System.out.print("digite a senha: ");

            BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
```

```
            String senha = reader.readLine();
//Le a senha do terminal

            /*Como java nao possui um metodo para limpeza de tela, nem
     *utiliza goto, por exemplo, entao e necessario que se utilize
     *um metodo para limpeza que no caso abaixo desloca uma tela.
     *o for se faz necessario para deslocar diversas telas fazendo
     *com que a senha deixa de ficar exposta.
             */
            for(int i = 0; i < 50; i++){
                System.out.print( "\033[H\033[2J" );
// Limpeza de tela
            }

            System.out.print("Servidor MyProxyHost Ativo");

            Interface m = new Implementacao(senha);
            Naming.rebind("rmi://localhost:7513/Server", m);
                // Cria um servidor rmi na porta 7513
            System.out.println("\nAguardando requisicoes de Clientes\n");
        } catch( Exception e ) {
            System.out.println( "Exception: " + e );
        }
    }

    public static void main(String[] args) {
        new Servidor();
    }
}
```

C.2 Implementacao.java

```
/*
 * implementacao.java
 *
 * Created on 16 de Outubro de 2006, 14:42
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
import java.util.Date;
import javax.net.ssl.SSLKeyException;
import javax.net.ssl.SSLServerSocket;
import javax.net.ssl.SSLServerSocketFactory;
import javax.net.ssl.SSLSocket;
//import org.apache.catalina.Host;

public class Implementacao extends UnicastRemoteObject
implements Interface {
```

```
    /*A classe implementacao e responsavel por todos os metodos que
     *implementao a interface atraves do servidor rmi.
     *E ela que cria os certificados, assina e chama a classe de limpeza
     *para evitar o excesso de lixo da requisicao.
     *Alem disso, e nela que e criado o arquivo de log de utilizacao que
     *pode servir para verificacao de usuarios que usaram o servico.
     */

    String senha = "";
    public Implementacao(String senha) throws RemoteException {
        this.senha = senha;
        //super();
    }

    public void criaCertHost( String identidade, String host, int dias )
    throws RemoteException, IOException, InterruptedException{
        /* Esse metodo cria no diretorio local o certificado de host,
         * caso o usuariario seja
         * valido, caso contrario a criacao sera negada
         */

        FileWriter registroDeUso = new FileWriter("registro.dat",true);
        /*Gravacao do registro de usuarios que pediram certificados de host
         *o valor true depois de registro e usado para fazer um append ao
         *inves de recriar o arquivo ele permite gravar sempre no final
*do arquivo.
         */

        Date dt = new Date();
        String registro = "\n"+identidade+" - "+host+" - "+dias+" - "+dt;
        registroDeUso.write(registro); //Grava no final do registro
```

```
        registroDeUso.flush();

        String dirLocal = System.getProperty("user.dir");
//Busca o diretorio atual
        String dirUser = System.getProperty("user.home");
//Busca o diretorio home do usuario
        String saida2 = "", saida1 = "";
        boolean userValido = false;
//Armazena a situacao do usuario (valido ou nao)
        int j = 0;
        if(dias > 7){
// Se a requisicao for para mais de 7 dias nao fazer nada
        } else{
// Caso contrario implemente a interface
            File gridMap = new File("/etc/grid-security/grid-mapfile");
//Abre o arquivo mapfile local
            int tam = (int)gridMap.length();
            FileInputStream fis = new FileInputStream(gridMap);
            byte map[] = new byte[tam];
            fis.read(map);
            String mapFile = new String(map);
        /* O proximo passo tenta localizar dentro do arquivo grid-mapfile
 * se existe uma entrada igual a identidade passada ao metodo,
 * caso exista, a variavel userValido recebe true, caso contrario
 * o usuario nao conseguira criar um certificado de host.
         */
            for(int i = 0; i < mapFile.length(); i++){
                if("\"".equals(mapFile.substring(i, i+1))){
                    i++;
                    j = i;
                    while(!("\"".equals(mapFile.substring(j, j+1)))){
```

```
                        j++;
                    }
                    //System.out.println(identidade);
                    //System.out.println(mapFile.substring(i, j));

                    if(identidade.equals(mapFile.substring(i, j))){
                        userValido = true;
                        i = mapFile.length();
                    }else{
                        i = j+1;
                    }

                }
            }
            if(userValido){
                Runtime rt = Runtime.getRuntime();
                Process p = rt.exec(
"grid-cert-request -service host -host "+host+
                        " -dir /tmp -force");

                /*InputStream is = p.getInputStream();
                BufferedReader br = new BufferedReader(
new InputStreamReader(is));
                while ((saida1 = br.readLine()) != null){
                    saida2 += (saida1+"\n");
                }
                System.out.println(saida2);
                saida2 = "";

                 */
                Limpeza limpar = new Limpeza();
```

```
                limpar.armazenarConf();

                Thread.sleep(10000);
                rt = Runtime.getRuntime();

                p = rt.exec("openssl ca -passin pass:"+senha+
" -batch -config " +
                        "/root/.globus/simpleCA//grid-ca-ssl.conf -in " +
                        "/tmp/hostcert_request.pem " +
                        "-out /tmp/hostcert.pem -days "+dias);

                // Executa um comando opessl para assinar o certificado
// de host
                /*is = p.getInputStream();
                br = new BufferedReader(new InputStreamReader(is));
                while ((saida1 = br.readLine()) != null){
                    saida2 += (saida1+"\n");
                }
                System.out.println(saida2);
                saida2 = "";
                 */
            }else
                System.out.println("usuario invalido");
        }
    }

    public String retornaCertHost( String Host ) throws RemoteException,
            FileNotFoundException, IOException, InterruptedException{
        /* Esse metodo retorna o certificado de host assinado
   ao solicitante */
        //String str[] = new String[1];
```

```
        //SSLCertHost certHost = new SSLCertHost(host);
        //certHost.main(str);
        return "concluido";

    }

    public String retornaKeyHost( String host ) throws RemoteException,
            FileNotFoundException, IOException, InterruptedException{
        /* Esse metodo retorna a chave privada ao solicitante */
        //String str[] = new String[1];
        //SSLKeyHost keyHost = new SSLKeyHost(host);
        //keyHost.main(str);
        Thread.sleep(5000);
        Limpeza limpar = new Limpeza();
        limpar.apagarArqs(host);
        return "concluido";
    }

}
```

C.3 Interface.java

```
/* Interface.java
 *
 * Created on 16 de Outubro de 2006, 14:46
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */
/**
 *
 * @author licht
 */
import java.io.FileNotFoundException;
import java.io.IOException;
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface Interface extends Remote {
    /* Classe interface com os métodos implementados pelo servidor */
    public void criaCertHost( String identidade, String host, int dias )
throws RemoteException,
            IOException, InterruptedException;
    public String retornaCertHost( String host ) throws RemoteException,
            FileNotFoundException, IOException, InterruptedException;
    public String retornaKeyHost( String host ) throws RemoteException,
            FileNotFoundException, IOException, InterruptedException;
}
```

C.4 Limpeza.java

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
/*
 * Limpeza.java
 *
 * Created on 20 de Outubro de 2006, 09:52
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

/**
 *
 * @author licht
 */
public class Limpeza {

    /**
     * Creates a new instance of Limpeza
     */
    public Limpeza() {
    }

    public void armazenarConf() throws IOException{
        String dirLocal = System.getProperty("user.dir");
            //Busca o diretorio atual
```

```
String dirUser = System.getProperty("user.home");
    //Busca o diretorio home do usuario
Process p;
Runtime rt = Runtime.getRuntime();
InputStream is;
BufferedReader br;
String saida1, saida2 = "";
p = rt.exec("cp "+dirUser+"/.globus/simpleCA/index.txt .");
    /*Faz um backup do arquivo
     *de link dos certificados
    */
is = p.getInputStream();
br = new BufferedReader(new InputStreamReader(is));
while ((saida1 = br.readLine()) != null){
    saida2 += (saida1+"\n");
}
System.out.println("\n\ncopiando index.txt\n"+saida2);
saida2 = "";

rt = Runtime.getRuntime();
p = rt.exec("cp "+dirUser+"/.globus/simpleCA/serial .");
    /*Faz um backup do arquivo
    *de serial dos certificados
    */
is = p.getInputStream();
br = new BufferedReader(new InputStreamReader(is));
while ((saida1 = br.readLine()) != null){
    saida2 += (saida1+"\n");
}
System.out.println("\n\ncopiando serial\n"+saida2);
saida2 = "";
```

```
}

public void apagarArqs(String arquivo) throws IOException{
    /* Metodo para apagar os arquivos criados durante a
     * requisicao dos certificados (Limpeza de lixo)
     * Retorna tambem os arquivos de backup para seus locais
     */
    String dirLocal = System.getProperty("user.dir");
    String dirUser = System.getProperty("user.home");
    Process p;
    Runtime rt = Runtime.getRuntime();
    InputStream is;
    BufferedReader br;
    String saida1, saida2 = "";
    p = rt.exec("cp "+dirLocal+"/index.txt "+dirUser+
            "/.globus/simpleCA -f");
    is = p.getInputStream();
    br = new BufferedReader(new InputStreamReader(is));
    while ((saida1 = br.readLine()) != null){
        saida2 += (saida1+"\n");
    }
    System.out.println("\n\nmovendo index.txt\n"+saida2);
    saida2 = "";

    rt = Runtime.getRuntime();
    p = rt.exec("cp "+dirLocal+"/serial "+dirUser+
            "/.globus/simpleCA -f");
    is = p.getInputStream();
    br = new BufferedReader(new InputStreamReader(is));
    while ((saida1 = br.readLine()) != null){
        saida2 += (saida1+"\n");
```

```
        }
        System.out.println("\n\nmovendo serial\n"+saida2);
        saida2 = "";

        rt = Runtime.getRuntime();
        p = rt.exec("rm /tmp/hostcert.pem");
        p = rt.exec("rm /tmp/hostkey.pem");
        p = rt.exec("rm /tmp/hostcert_request.pem");
        is = p.getInputStream();
        br = new BufferedReader(new InputStreamReader(is));
        while ((saida1 = br.readLine()) != null){
            saida2 += (saida1+"\n");
        }
        System.out.println("\n\napagando arquivos\n"+saida2);
        saida2 = "";
    }

}
```

C.5 Cliente.java

```
/*
 * Cliente.java
 *
 * Created on 17 de Outubro de 2006, 13:27
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

/**
 *
 * @author licht
 */

import java.io.BufferedReader;
import java.io.FileWriter;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;

public class Cliente extends Thread implements Runnable {
    /* Classe Cliente em modo texto caso o usuario pretenda
     * fazer a requisicao de certificados via terminal.
     */
    public static void main( String args[] ) {
```

```
        String host = "";

        String dirLocal = System.getProperty("user.dir");
//Busca o diretorio atual
        String dirUser = System.getProperty("user.home");
//Busca o diretorio home do usuario
        String dirTemp = System.getProperty("java.io.tmpdir");
//Busca o diretorio temporário
        int dias = 0;
        String servMyProxy = "";

        if(args.length < 3){
// Caso falte argumento, e retornado um erro
            System.out.println("\nVoce deve digitar o nome da maquina, a "+
                    "quantidade de dias e o servidor MyProxy");
            System.out.println(
"\nFormato correto:\njava Cliente <nome.dominio> " +
                "<qtd_dias> <servidor>\n\n");
            System.exit(1);
        } else{
            host = args[0];
            dias = Integer.parseInt(args[1]);
            if(dias > 7){
                System.out.println(
"Voce pode pedir certificados com prazo maximo "+
                        "de 7 dias.");
                System.exit(1);
            } else{
                servMyProxy = args[2];
            }
        }
```

```
        try {
            byte password[] = new byte[10];
            System.out.print("digite a senha: ");
            BufferedReader reader = new BufferedReader(
new InputStreamReader(System.in));
            String senha = reader.readLine();
//Le a senha do terminal

            /*Como java nao possui um metodo para limpeza de tela,
     *nem utiliza goto, por exemplo, entao e necessario que
     *se utilize um metodo para limpeza que no caso abaixo
     *desloca uma tela. o for se faz necessario para
             *deslocar diversas telas fazendo com que a senha deixa
     * de ficar exposta.
            */

            for(int i = 0; i < 50; i++){
                System.out.print( "\033[H\033[2J" ); // Limpeza de tela
            }
            Process p;
            Runtime rt = Runtime.getRuntime();
            InputStream is;
            BufferedReader br;
            String saida1, saida2 = "", identidade = "";
            //System.out.println(System.getProperties());

            String cmd = "openssl x509 -in "+dirTemp+
"/x509up_u1010 -signkey "+
                dirUser+"/.globus/userkey.pem -passin pass:"+senha;
```

```
            p = rt.exec(cmd);
            is = p.getInputStream();
            br = new BufferedReader(new InputStreamReader(is));

            while ((saida1 = br.readLine()) != null){
                saida2 += (saida1+"\n");

            }

            System.out.println(saida2.length());
            if(saida2.length() < 700){
                System.out.println("Senha invalida ou proxy inexistente\n" +
                        "Tente novamente ou contate o administrador da "+
"grade!..\n\n");
            }else{ //Usuário digitou a senha correta
                saida1 = "";
                saida2 = "";
                System.out.println("\nusuario aceito!..\n");

                p = rt.exec("grid-proxy-info");
// Busca informacao do certificado de usuario
                is = p.getInputStream();
                br = new BufferedReader(new InputStreamReader(is));
                while ((saida1 = br.readLine()) != null){
                    if("id".equals(saida1.substring(0,2))){
                        identidade = saida1;
                    }
                    saida2 += (saida1+"\n");
                }

                for(int i = 2; i < identidade.length(); i++){
```

```
                    if("/".equals(identidade.substring(i, i+1))){
                        identidade = identidade.substring(
i, identidade.length());
                        i = identidade.length();
                    }
                }
                System.out.println("Sua Identidade e:\n"+identidade);

                Interface i = (Interface) Naming.lookup(
"rmi://"+servMyProxy+":7513/Server" );
                 //Criada uma conexao rmi com o servidor

                i.criaCertHost(identidade, host, dias);
/*implementa a interface
                 *e cria o certificado de host
                 */
                Thread.sleep(15000);
// Contador de tempo para evitar erros

                if(i.retornaCertHost(host)==null){
                    System.out.println(
"Voce nao e um usuario valido, \n"+
"entre em contato com o " +
                        "administrador da grade\nou crie seu proxy "+
"e tente novamente");
                } else{

                    //String dirTemp = System.getProperty("java.io.tmpdir");
                    String local = System.getProperty("user.dir");
                    cmd = "java -Djavax.net.ssl.trustStore=SSLStoreServer " +
                            "-Djavax.net.ssl.trustStorePassword="+senha+" " +
```

```
                        "sslClientCert "+servMyProxy;

                    //System.out.println(cmd);

                    p = rt.exec(cmd);
                    /*is = p.getInputStream();
                    br = new BufferedReader(new InputStreamReader(is));
                    while ((saida1 = br.readLine()) != null){
                        saida2 += (saida1+"\n");

                    }
                    */

                    Thread.sleep(5000);

                    cmd = "java -Djavax.net.ssl.trustStore=SSLStoreServer "+
                          "-Djavax.net.ssl.trustStorePassword="+senha+" "+
                          "sslClientKey "+servMyProxy;

                    p = rt.exec(cmd);

                    System.out.println(
"\nOs arquivos hostcert.pem e hostkey.pem foram "+
"salvos \ncom sucesso no diretorio local, voce "+
"deve coloca-los no \ndiretorio de certificado "+
"host, em geral: /etc/grid-security \n" +
                        "com permissoes: 644 e 400 respectivamente, para"+
" maiores \ninformacoes, entre em contato com o "+
"administrador da grade\n");
                }
            }
```

```
        } catch( MalformedURLException e ) {
            System.out.println();
            System.out.println( "MalformedURLException: " + e.toString() );
        } catch( RemoteException e ) {
            System.out.println();
            System.out.println( "RemoteException: " + e.toString() );
        } catch( NotBoundException e ) {
            System.out.println();
            System.out.println( "NotBoundException: " + e.toString() );
        } catch( Exception e ) {
            System.out.println();
            System.out.println( "Exception: " + e.toString() );
            System.out.println("Voce nao e um usuario valido,
\nentre em contato com o " +
                "administrador da grade\nou crie seu proxy e tente "+
"novamente");
        }
    }
}
```

C.6 FrameCliente.java

```
import java.awt.Color;
import java.awt.Font;
import java.io.BufferedReader;
import java.io.FileWriter;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import javax.swing.JOptionPane;
//import cliente.sslClient;
/*
 * FrameCliente.java
 *
 * Created on 23 de Outubro de 2006, 16:08
 */
/**
 *
 * @author  licht
 */
public class FrameCliente extends javax.swing.JFrame {
    /*Esta classe é uma implementacao grafica do cliente para requisicao
     * de cetificados de host. Esta, se conecta via rmi do java e
     *requisita a criacao de um certificado
     *host por vez por um prazo maximo de 7 dias.
     */
    String dirLocal = System.getProperty("user.dir");
```

```
        //Busca o diretorio atual
    String dirUser = System.getProperty("user.home");
        //Busca o diretorio home do usuario
    String dirTemp = System.getProperty("java.io.tmpdir");
        //Busca o diretorio temporário

    public FrameCliente() {
        initComponents();
    }

    private void initComponents() {
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jTextField1 = new javax.swing.JTextField();
        jTextField2 = new javax.swing.JTextField();
        jTextField3 = new javax.swing.JTextField();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jLabel5 = new javax.swing.JLabel();
        jPasswordField1 = new javax.swing.JPasswordField();

        setDefaultCloseOperation(
javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setResizable(false);
        jLabel1.setFont(new java.awt.Font("Dialog", 1, 14));
```

```
        jLabel1.setText(
    "AddHost - Pedido de Certificados Tempor\u00e1rios para Host");

        jLabel2.setText("Servidor MyProxy:");
        jLabel3.setText("Nome da M\u00e1quina Cliente:");
        jLabel4.setText("Quantidade de Dias max. 7:");

        jButton1.setText("Requisitar Certificado de Host");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jButton2.setText("Sobre");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jButton3.setText("Sair");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jButton4.setText("Ajuda");
        jButton4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                jButton4ActionPerformed(evt);
            }
        });

        jTextArea1.setColumns(20);
        jTextArea1.setEditable(false);
        jTextArea1.setRows(5);
        jScrollPane1.setViewportView(jTextArea1);

        jLabel5.setText("Senha do Certificado:");

        org.jdesktop.layout.GroupLayout layout =
new org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.LEADING)
                    .add(
org.jdesktop.layout.GroupLayout.TRAILING, jScrollPane1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 475,
Short.MAX_VALUE)
                    .add(layout.createSequentialGroup()
                        .add(jButton2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 134,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .add(38, 38, 38)
                        .add(jButton3,
```

```
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 133,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(
org.jdesktop.layout.LayoutStyle.RELATED, 37,
Short.MAX_VALUE)
                        .add(jButton4,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 133,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(jButton1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 475,
Short.MAX_VALUE)
                    .add(layout.createSequentialGroup()
                        .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jLabel2)
                            .add(jLabel3)
                            .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.TRAILING, false)
                                .add(
org.jdesktop.layout.GroupLayout.LEADING, jLabel5,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                                .add(
org.jdesktop.layout.GroupLayout.LEADING, jLabel4,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)))
                        .add(10, 10, 10)
                        .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.LEADING)
```

```
                            .add(jTextField3,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 294,
Short.MAX_VALUE)
                            .add(jTextField2,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 294,
Short.MAX_VALUE)
                            .add(jTextField1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 294,
Short.MAX_VALUE)
                            .add(jPasswordField1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 294,
Short.MAX_VALUE)))
                    .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabel1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 424,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(jLabel1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 32,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(18, 18, 18)
                .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jLabel2)
                    .add(jTextField1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 21,
```

```
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(
org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jLabel3,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 15,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(jTextField2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jLabel4)
                    .add(jTextField3,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jLabel5)
                    .add(jPasswordField1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(
org.jdesktop.layout.LayoutStyle.RELATED, 17, Short.MAX_VALUE)
                .add(jButton1)
```

```
                .add(17, 17, 17)
                .add(layout.createParallelGroup(
org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jButton2)
                    .add(jButton4)
                    .add(jButton3))
                .add(19, 19, 19)
                .add(jScrollPane1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 164,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        pack();
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

        Font f = new Font("Times",1,14);
        jTextArea1.setText("");
        jTextArea1.setFont(f);
        jTextArea1.setText("" +
"Antes de  requisitar um  certificado de  host você  deve\n" +
        "criar  seu  proxy  local, em  geral, usando  o  comando:\n" +
        "grid-proxy-init  ou caso  você esteja  usando o servidor\n" +
        "MyProxy,  usando  o comando:  myproxy-logon ( leia  o\n" +
        "manual  do MyProxy  para  maiores  informaçôes). Após\n" +
        "criar o  proxy, você deve preencher todos  os campos do\n" +
        "formulário.  Atente  para  o  a  utilização  do  endereço\n" +
        "do servidor. O  certificado de  host pode ser solicitado\n" +
        "pelo  nome  do  host ou  pelo  ip  do  mesmo. O  prazo\n" +
        "máximo para requisição de um certificado  é de 7 dias,\n" +
```

```
        "caso  você  necessite de  mais tempo, faça  uma  nova\n" +
        "requisição, mesmo  que o  prazo de 7  dias não  tenha\n" +
        "sido concluído, isso te dará mais 7 dias ou a quantidade\n" +
        "de dias solicitado. A senha passada no formulário deve ser\n" +
        "a mesma do proxy criado na máquina local\n");
        // TODO add your handling code here:
        //Color c = new Color(233);
        jTextArea1.setForeground(Color.BLUE);
        jTextArea1.append(

"Para maiores informações envie email para licht@lncc.br");
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

        jTextArea1.setForeground(Color.RED);
        jTextArea1.setText("");
        Font f = new Font("Times",1,16);
        jTextArea1.setFont(f);
        if("".equals(jTextField1.getText())){
            jTextArea1.setText("");
            jTextArea1.setText(
"\n\nO servidor MyProxy não pode ser vazio\n" +
                    "Digite o nome do Servidor");
        }else{
            if("".equals(jTextField2.getText())){
                jTextArea1.setText(" ");
                jTextArea1.setText("\n\nDigite o nome do Cliente ");
            }else{

                if("".equals(jTextField3.getText())){
```

```
                    jTextArea1.setText("");
                    jTextArea1.setText(
"\n\nO número de dias não pode ser vazio\n" +
                        "Digite o número de dias para uso do\n" +
                        "certificado de host");
                }else{
                    if((Integer.parseInt(jTextField3.getText()))>7){
                        jTextArea1.setText("");
                        jTextArea1.setText(
"\n\nVocê só pode solicitar um certificado\n" +
                        "de host por até 7 dias ");
                        // TODO add your handling code here:
                    } else{
                        String senha = jPasswordField1.getText();
                        String servMyProxy = jTextField1.getText();
                        String host = jTextField2.getText();
                        int dias = Integer.parseInt(jTextField3.getText());

                        solicitaCert(servMyProxy, host, dias, senha);
                        /*Chama o metodo que solicita
                         * ao servidor MyProxyHost o
                         * certificado, passando como
                         * parametro o endereco do servidor
                         * o nome do host cliente e o
                         * numero de dias para validar o
                         * certificado
                         */
                    }
                }
            }
        }
```

```
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        JOptionPane.showMessageDialog(null, "Obrigado por usar a Grade\n" +
                "Tenha um bom dia");
        System.exit(0);// TODO add your handling code here:
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        /*Botao sobre. Mostra informacoes do desenvolvedor*/
        JOptionPane.showMessageDialog(null,
"\nSoftware criado por: Fabio Lopes Licht \n" +
                "Instituto Militar de Engenharia - IME-RJ\n" +
                "Mestrado em Sistemas e Computação\n" +
                "Outubro de 2006\n\n" +
                "Contato: licht@lncc.br","AddHost",1);
    }

    private void solicitaCert(String servMyProxy,
String host, int dias, String senha){

/*Metodo principal da classe, responsavel por criar uma conexao
 *com o servidor MyProxyHost receber os certificados assinados
 *e salva-los em arquivo.
         */

        String identidade = "";
        try {
            //System.out.println("Servidor myproxy: "+servMyProxy);
```

```
            //System.out.println("Cliente: "+host);
            Process p;
            Runtime rt = Runtime.getRuntime();
            InputStream is;
            BufferedReader br;
            String saida1, saida2 = "";

            jTextArea1.setText(senha);

            String cmd = "openssl x509 -in "+dirTemp+
"/x509up_u1010 -signkey "+dirUser+
                "/.globus/userkey.pem -passin pass:"+senha;

            p = rt.exec(cmd);
            is = p.getInputStream();
            br = new BufferedReader(new InputStreamReader(is));

            while ((saida1 = br.readLine()) != null){
                saida2 += (saida1+"\n");

            }
            //System.out.println(saida2.length());
            if(saida2.length() < 700){
                Font f = new Font("Times",1,14);
                jTextArea1.setText("");
                jTextArea1.setFont(f);
                jTextArea1.setText(
    "\nSenha invalida ou proxy inexistente\n" +
                    "Tente novamente ou contate o administrador"+
    "da grade!..\n\n");
            }else{ //Usuário digitou a senha correta
```

```
                saida1 = "";
                saida2 = "";
                Font f = new Font("Times",1,14);
                jTextArea1.setText("");
                jTextArea1.setFont(f);
                jTextArea1.setText("\nusuário aceito!..\n");

                p = rt.exec("grid-proxy-info");
/* Busca informacao do certificado de usuario
                 * local para ser passado como parametro ao
                 * servidor MyProxyHost para comparar se o
                 * usuario e valido.
                 */
                is = p.getInputStream();
                br = new BufferedReader(new InputStreamReader(is));
                while ((saida1 = br.readLine()) != null){
/*Busca o identificador do usuario*/

                    if("id".equals(saida1.substring(0,2))){
/*Identifica na string o ID do usuario.*/
                        identidade = saida1;
                    }
                    saida2 += (saida1+"\n");
                }

                for(int i = 2; i < identidade.length(); i++){
//Pega somente o id
                    if("/".equals(identidade.substring(i, i+1))){
                        identidade = identidade.substring(i,
identidade.length());
                        i = identidade.length();
```

```
                    }
                }

                Interface i = (Interface) Naming.lookup(
"rmi://"+servMyProxy+":7513/Server" );

//Criada uma conexao rmi com o servidor

                i.criaCertHost(identidade, host, dias);
/*implementa a interface
                  *e cria o certificado de host
                  */

                Thread.sleep(25000); // Contador de tempo para evitar erros

                if(i.retornaCertHost(host)==null){
                    f = new Font("Times",1,14);
                    jTextArea1.setText("");
                    jTextArea1.setFont(f);
                    jTextArea1.setText(
"\nSua Identidade é:\n"+identidade+
                        "\nVocê não e um usuário válido, \nentre em "+
"contato com o administrador da grade\nou crie"+
"seu proxy e tente novamente");
                } else{

                    String dirTemp = System.getProperty("java.io.tmpdir");
                    String local = System.getProperty("user.dir");
                    cmd = "java -Djavax.net.ssl.trustStore=SSLStoreServer " +
                            "-Djavax.net.ssl.trustStorePassword="+senha+" " +
                            "sslClientCert "+servMyProxy;
```

```
                    //System.out.println(cmd);
                    p = rt.exec(cmd);
                    /*is = p.getInputStream();
                    br = new BufferedReader(new InputStreamReader(is));
                    while ((saida1 = br.readLine()) != null){
                        saida2 += (saida1+"\n");

                    }
                    */

                    Thread.sleep(5000);

                    cmd = "java -Djavax.net.ssl.trustStore=SSLStoreServer " +
                            "-Djavax.net.ssl.trustStorePassword="+senha+" " +
                            "sslClientKey "+servMyProxy;

                    p = rt.exec(cmd);

                    /*is = p.getInputStream();
                    br = new BufferedReader(new InputStreamReader(is));
                    while ((saida1 = br.readLine()) != null){
                        saida2 += (saida1+"\n");

                    }
                    */

                    jTextArea1.setText(
"\nSua Identidade é:\n"+identidade+"\n" +
                        "\nOs arquivos hostcert.pem e hostkey.pem foram"+
"salvos \n com sucesso no diretorio"+dirTemp+
```

```
", você deve colocá-los no \ndiretório de certificado"+
" host, em geral: /etc/grid-security \ncom permissões"+
": 644 e 400 respectivamente. Para maiores \n" +
"informações, entre em contato com o administrador da"+
"grade\n\n"+ i.retornaKeyHost(""));
                }
            }

        } catch( MalformedURLException e ) {
            jTextArea1.setText("");
            jTextArea1.setText( "MalformedURLException: \n" + e.toString() );
        } catch( RemoteException e ) {
            jTextArea1.setText("");
            jTextArea1.setText( "RemoteException: \n" + e.toString() );
        } catch( NotBoundException e ) {
            jTextArea1.setText("");
            jTextArea1.setText( "NotBoundException: \n" + e.toString() );
        } catch( Exception e ) {
            jTextArea1.setText("");
            //System.out.println( "Exception: " + e.toString() );
            jTextArea1.setText("Sua Identidade é:\n"+identidade+
               "\n\nException:\nVocê não e um " +
               "usuário válido, \nentre em contato com o " +
               "administrador da grade\nou crie seu proxy e tente novamente");
        }
    }

    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new FrameCliente().setVisible(true);
```

```
            }
        });
    }
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JPasswordField jPasswordField1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
}
```

C.7 SSLCertHost.java

```
/*
 * SSLCertHost.java
 *
 * Created on 23 de Dezembro de 2006, 16:08
 */

/**
 *
 * @author  licht
 */
import java.io.*;
import javax.net.ssl.*;

public class SSLCertHost {

    private SSLServerSocket serverSocket;


    public SSLCertHost() throws Exception {

        SSLServerSocketFactory socketFactory =
                ( SSLServerSocketFactory )
                SSLServerSocketFactory.getDefault();

        serverSocket = ( SSLServerSocket )
        socketFactory.createServerSocket( 7514 );

    }
```

```
private void runServer() throws InterruptedException{

    while ( true ) {

        try {

            System.err.println( "Aguardando conneccoes..." );

            SSLSocket socket = ( SSLSocket ) serverSocket.accept();

            System.err.println( "Connectado...");

            BufferedReader input = new BufferedReader(
                    new InputStreamReader( socket.getInputStream() ) );

            String line = input.readLine();
            System.out.println(line);

            Thread.sleep(5000);
            File hostcert = new File("/tmp/hostcert.pem");
            int tam = (int)hostcert.length();
            FileInputStream fis = new FileInputStream(hostcert);
            byte cert[] = new byte[tam];
            fis.read(cert);
            String certificado = new String(cert);
            PrintStream out = new PrintStream(
                    socket.getOutputStream() );
            out.println(certificado);

            input.close();
```

```
                out.close();
                socket.close();

            }

            catch ( IOException ioException ) {
                ioException.printStackTrace();
            }

        }

    }

    public static void main( String args[] ) throws Exception {
        SSLCertHost server = new SSLCertHost();
        server.runServer();
    }
}
```

C.8 SSLKeyHost.java

```
/*
 * SSLKeyHost.java
 *
 * Created on 23 de Dezembro de 2006, 16:08
 */

/**
 *
 * @author  licht
 */
import java.io.*;
import javax.net.ssl.*;

public class SSLKeyHost {

    private SSLServerSocket serverSocket;

    public SSLKeyHost() throws Exception {
        SSLServerSocketFactory socketFactory =
                ( SSLServerSocketFactory )
                SSLServerSocketFactory.getDefault();

        serverSocket = ( SSLServerSocket )
        socketFactory.createServerSocket( 7515 );

    }

    private void runServer() throws InterruptedException {
```

```
while ( true ) {

    try {

        System.err.println( "Aguardando conneccoes..." );

        SSLSocket socket = ( SSLSocket ) serverSocket.accept();

        System.err.println( "Connectado...");

        BufferedReader input = new BufferedReader(
                new InputStreamReader( socket.getInputStream() ) );

        String line = input.readLine();
        System.out.println(line);

        String dirLocal = System.getProperty("user.dir");
        File hostkey = new File("/tmp/hostkey.pem");
        int tam = (int)hostkey.length();
        FileInputStream fis = new FileInputStream(hostkey);
        byte key[] = new byte[tam];
        fis.read(key);
        String chave = new String(key);
        PrintStream out = new PrintStream(
                socket.getOutputStream() );
        out.println(chave);

        Thread.sleep(2000);
        Limpeza lp = new Limpeza();
        lp.apagarArqs("");
```

```
                input.close();
                out.close();
                socket.close();

            }


            catch ( IOException ioException ) {
                ioException.printStackTrace();
            }

        }
    }

    public static void main( String args[] ) throws Exception {
        SSLKeyHost server = new SSLKeyHost();
        server.runServer();
    }
}
```

C.9 sslClientCert.java

```
/*
 * sslClientCert.java
 *
 * Created on 5 de Janeiro de 2007, 13:01
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

/**
 *
 * @author licht
 */

import java.io.*;
import javax.net.ssl.*;

public class sslClientCert {

    public static void main(String args[]) throws InterruptedException{
        clientCert(args[0], 7514);
    }

    public static void clientCert(String servidor, int porta)
    throws InterruptedException {
        String line = "";

        try {
```

```
            SSLSocketFactory socketFactory =
                    ( SSLSocketFactory ) SSLSocketFactory.getDefault();

SSLSocket socket =
                    ( SSLSocket ) socketFactory.createSocket(
                    servidor, porta );

            PrintStream out = new PrintStream(
                    socket.getOutputStream() );
            out.println("Sou o cliente Cert");

            Thread.sleep(5000);
            BufferedReader input = new BufferedReader(
                    new InputStreamReader( socket.getInputStream() ) );
            line = input.readLine();
            String Acumulador = "";
            while(line != null){
                Acumulador += line+"\n";
                line = input.readLine();
            }

            //System.out.println(Acumulador);
            FileWriter hostcert = new FileWriter("hostcert.pem");
            // Cria arquivo local
            hostcert.write(Acumulador);
// Grava o certificado hostcert
            hostcert.flush();
            hostcert.close();

            input.close();
            out.close();
```

```
            socket.close();

        }
        catch ( IOException ioException ) {
            ioException.printStackTrace();
        }

        finally {
            System.exit( 0 );
        }

    }

}
```

C.10 sslClientKey.java

```
/*
 * sslClientKey.java
 *
 * Created on 5 de Janeiro de 2007, 13:01
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */


/**
 *
 * @author licht
 */

import java.io.*;
import javax.net.ssl.*;

public class sslClientKey {

    // sslClient constructor

    public static void main(String args[]){
        clientKey(args[0], 7515);
    }

    public static void clientKey(String servidor, int porta) {
        String line = "";
```

```
        // open SSLSocket connection to server and send login
        try {

            // obtain SSLSocketFactory for creating SSLSockets
            SSLSocketFactory socketFactory =
                    ( SSLSocketFactory ) SSLSocketFactory.getDefault();

            // create SSLSocket from factory
            SSLSocket socket =
                    ( SSLSocket ) socketFactory.createSocket(
                    servidor, porta );

            PrintStream out = new PrintStream(
                    socket.getOutputStream() );
            out.println("Sou o cliente Key");

////////////////////////////////////////////////////////////////
            BufferedReader input = new BufferedReader(
                    new InputStreamReader( socket.getInputStream() ) );
            line = input.readLine();
            String Acumulador = "";
            while(line != null){
                Acumulador += line+"\n";
                line = input.readLine();
            }

            //System.out.println(Acumulador);
            FileWriter hostkey  = new FileWriter("hostkey.pem" );
            // Cria arquivo local
            hostkey.write(Acumulador); // Grava a chave hostkey
            hostkey.flush();
```

```
            hostkey.close();

////////////////////////////////////////////////////////////////

            out.close();
            input.close();
            socket.close();

        } // end try

        // handle exception communicating with server
        catch ( IOException ioException ) {
            ioException.printStackTrace();
        }

        // exit application
        finally {
            System.exit( 0 );
        }

    } // end sslClient constructor

}
```

Apêndice D

OpenSSL

Este anexo foi coletado na integra do trabalho de Hammurabi[MENDES] e serve de base para o entendimento de como foi implementada a classe que assina os certificados desta dissertação. No mesmo trabalho Hammurabi apresenta uma implementação usando as bibliotecas OpenSSL. Vale lembrar que este texto, serve como fonte de consulta para o entendimento do OpenSSL e não diretamente deste trabalho de dissertação.

O protocolo SSL foi criado com o objetivo de proporcionar mecanismos de autenticação e sigilo entre duas aplicações que se comunicam via algum tipo de protocolo de comunicação. Outros aspectos importantes considerados no momento de sua concepção foram: interoperabilidade, permitindo a comunicação com outra aplicação sem que haja a necessidade de entrar em detalhes a respeito de sua implementação; extensiblidade, que permite criar novas rotinas e funcionalidades baseadas em mecanismos pré-existentes do protocolo; por fim, eficiência, tornando o protocolo viável para o uso entre aplicações cliente-servidor via Internet.

A arquitetura do SSL é disposta em camadas, a exemplo do TCP/IP. Uma delas, a chamada Record Layer, recebe informações não encriptadas das aplicações, dispondo-as em blocos numerados seqüencialmente. Estes blocos então passam por uma compressão, seguida da geração de códigos de autenticação (MACs). Em seguida, os blocos são encriptados e enviados. A numeração das mensagens enviadas

é importante para facilitar o trabalho do receptor na detecção de blocos em falta, alterados ou injetados por terceiros.

Os protocolos sobre os quais o SSL é construído incluem um especialmente concebido com o objetivo de sinalizar transações entre estratégias de cifragem usadas na sessão. Este protocolo é denominado Change Cipher Spec Protocol. Outro protocolo importante é o Alert Protocol, usado na sinalização de erros e em notificações de fechamento de conexão.

Ainda há o Handshake Protocol, que estabelece os parâmetros criptográficos da sessão, operando ao topo da Record Layer. No início da sessão, o cliente envia ao servidor uma hello message, informando os algoritmos e protocolos disponibilizados por sua implementação do SSL, sendo eles criptográficos ou não (por exemplo, os algoritmos de compressão associados também são informados). O servidor, baseado nos algoritmos e protocolos que a ele estão disponíveis, escolhe alguns dos parâmetros informados pelo cliente para serem usados no estabelecimento da sessão, notificando-o através de uma outra hello message.

Em seguida, o servidor envia seu certificado (ou informações associadas a um protocolo usado para troca de chaves, caso ele não tenha um certificado ou seu certificado possa ser usado apenas para verificar assinaturas digitais), e o cliente opcionalmente faz o mesmo. Logo após, a chave de sessão é instituída, através dos métodos criptográficos estabelecidos na troca das hello messages (por exemplo, Diffie-Hellmann ou RSA).

O projeto OpenSSL disponibiliza um toolkit em código livre, que implementa o protocolo SSL e vários algoritmos e primitivas criptográficas de uso comum, incluindo algoritmos de troca de chaves, funções de hash, algoritmos simétricos e assimétricos. O toolkit se apesenta na forma de duas bibliotecas e um conjunto de programas que implementam as rotinas por elas disponibilizadas. Os mecanismos do SSL estão implementadas na libssl, e os outros algoritmos estão implmentados na libcrypto.

O OpenSSL é uma implementação de código aberto, largamente utilizada, dos

protocolos Secure Sockets Layer (SSL v2/v3) e Transport Layer Security (TLS v1), além de ser utilizada como biblioteca criptográfica. Os protocolos SSL e TSL são usados para prover conexão segura entre o cliente e servidor em protocolos de alto nível, como o HTTP.

No site do projeto [OPENSSL06] o OpenSSL é tratado como um padrão de uma implementação livre do protocolo SSL (Secure Sockets Layer). O mesmo site trata o projeto OpenSSL como um esforço colaborativo para desenvolver uma biblioteca de criptografia de propósito geral e a implementação dos protocolos Secure Sockets Layer (SSL v2/v3) e Transport Layer Security (TLS v1). O projeto é gerenciado por uma comunidade internacional de voluntários que usam a Internet para se comunicar, planejar e desenvolver o OpenSSL e sua documentação relacionada.

O OpenSSL é baseado na biblioteca SSLeay desenvolvida por Eric A. Young e Tim J. Hudson. Para concluir, OpenSSL é livre e você usa-lo para propósitos comerciais ou não e este foi um dos motivos de ter sido utilizado para esta implementação.

Apêndice E

Modelo de Certificados

E.1 Modelo de Chave Pública de Usuário

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 13 (0xd)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: O=Grid, OU=lncc, OU=CA-ComCiDis, CN=ComCiDis Simple CA
        Validity
            Not Before: Oct  3 20:36:23 2006 GMT
            Not After : Oct  3 20:36:23 2007 GMT
        Subject: O=Grid, OU=lncc, OU=CA-ComCiDis, OU=lncc.br, CN=Fabio Licht
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:c2:39:0f:c3:16:7c:da:d1:ec:7f:35:3d:c9:82:
                    28:b3:3f:2d:75:78:06:92:7d:a1:8b:64:45:be:53:
                    31:a2:87:0e:88:9d:a5:07:59:67:0d:5c:f0:39:d3:
                    ae:6e:84:f6:90:57:0a:07:ec:cd:85:aa:a2:96:02:
```

```
                    4e:bb:2b:7f:1a:5d:d1:2c:02:95:46:85:93:c2:39:
                    fe:c3:2e:d9:f6:f4:df:05:0f:3f:f9:f3:39:a9:0a:
                    9d:7f:a1:d9:48:83:51:fe:90:d3:e9:9f:6d:fc:e4:
                    16:f7:8e:1d:a4:13:78:47:c0:bc:19:11:51:4c:a6:
                    c5:b1:5c:b5:60:38:33:66:57
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            Netscape Cert Type:
                SSL Client, SSL Server, S/MIME, Object Signing
    Signature Algorithm: md5WithRSAEncryption
        ae:2e:6c:b6:a1:3a:10:72:11:b4:0e:71:5c:ed:ea:94:0e:be:
        fb:dc:76:a7:c8:c7:81:d8:8e:74:bb:4e:73:48:83:d6:26:55:
        17:bc:cb:9b:ba:fc:5b:7e:0a:75:c0:bb:4e:77:5a:13:7c:14:
        d2:cd:61:5e:17:02:8c:da:d3:9b:02:13:57:c8:24:e1:fe:7b:
        bd:95:8a:f2:2d:5c:b0:70:bd:69:5b:f5:54:ff:c9:8d:3f:b8:
        1a:ae:49:f1:f2:a1:69:9a:4d:d7:60:4d:63:8b:12:19:9a:cc:
        9b:a3:c3:aa:ef:a2:a1:ac:79:2a:fb:76:4b:cf:f0:20:86:82:
-----BEGIN CERTIFICATE-----
MIICODCCAaGgAwIBAgIBDTANBgkqhkiG9w0BAQQFADBRMQ0wCwYDVQQKEwRHcmlk
MQ0wCwYDVQQLEwRsbmNjMRQwEgYDVQQLEwtDQS1Db21DaURpczEbMBkGA1UEAxMS
Q29tQ2lEaXMgU2ltcGxlIENBMB4XDTA2MTAwMzIwMzYyM1oXDTA3MTAwMzIwMzYy
M1owXDENMAsGA1UEChMER3JpZDENMAsGA1UECxMEbG5jYzEUMBIGA1UECxMLQ0Et
Q29tQ2lEaXMxEDAOBgNVBAsTB2xuY2MuYnIxFDASBgNVBAMTC0ZhYmlvIExpY2h0
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDCOQ/DFnza0ex/NT3JgiizPy11
eAaSfaGLZEW+UzGihw6InaUHWWcNXPA5065uhPaQVwoH7M2FqqKWAk67K38aXdEs
ApVGhZPCOf7DLtn29N8FDz/58zmpCp1/odlIg1H+kNPpn2385Bb3jh2kE3hHwLwZ
EVFMpsWxXLVgODNmVwIDAQABoxUwEzARBglghkgBhvhCAQEEBAMCBPAwDQYJKoZI
hvcNAQEEBQADgYEAri5stqE6EHIRtA5xXO3qlA6++9x2p8jHgdiOdLtOc0iD1iZV
F7zLm7r8W34KdcC7TndaE3wU0s1hXhcCjNrTmwITV8gk4f57vZWK8i1csHC9aVv1
VP/JjT+4Gq5J8fKhaZpN12BNY4sSGZrMm6PDqu+ioax5Kvt2S8/wIIaCVCw=
-----END CERTIFICATE-----
```

E.2 Modelo de Chave Privada de Usuário

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,59D1FFA625F741FB

jC2YSc984iHJtFU+nrqiLVuh/ehKBj+sIzH4sgUmzfAeKE5xV1DjHq0wSuLUO5BV
n1u/3sX5KLeQPzPmedIcyWS/xjaYco4+KJOx9+wVWMSyoeLXfvNe4f5nLiDzLFSB
xSCJeqgoJZDhOCe4pxgJ336/KLI9JhHigiDGgIbuDK3PeOuidNVDp0N1bNf/Agiw
bDbxrh+/0lk3o/n86XYwS52z98eghD73P8zWe1Q6TniH6cVgKyTHpYFR3rpCYnvQ
Pi/fy0EmlSzhADtumH+bDWHpS2LoKZRGW4ZYsO1ZSxu2KP0cCFoX+cHF9sGrnPyL
Fciy+vhD0JCyd9fTPxIvMtwx+3NGDjzZh/xsYjvBtB/w/BXULUMs/FWZs7Pmooqf
lgr507iOz92o0Wco2bCT5He7oNpcZkN3+jLIKgRutAYZceMx7gatChqXIv6gWcLM
ECaAnRIKCQPQYF4RyeTHBgTBTDm2ekyPsdLgCxqHrEBoqG1LGFwwROEZ7FG4h+K6
0ssvdKzoebMh2JsOaqu70+CKZDn7Lr+oMrtfpmnK+btb1uxQGxzU5UYOQ95f2wlI
9KcCclpy45BKQEH8xhZFedpjgjwAaSndR+Jxz+m3cBWacLGhoh+G8ciVdcBd7qdH
N/Z8+hb3BHlOO59O2OsyxgnS7eTA6pVR6VTtq5pNQt8tNLtNfHZXAySyGreAEhKq
ydb7o01lcAgteb5QXrRV2524tC50a/kYk0bVIfpwN5ulZVS/3rdcEqw4S794zW8p
kS9MWgrevK5ce2cbQEwUZEmLNas5LBLyUGdGMYgZQglQslshK705EQ==
-----END RSA PRIVATE KEY-----
```

E.3 Modelo de Chave Pública de Máquina

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 12 (0xc)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: O=Grid, OU=lncc, OU=CA-ComCiDis, CN=ComCiDis Simple CA
        Validity
            Not Before: Jan 10 15:34:21 2007 GMT
            Not After : Jan 15 15:34:21 2007 GMT
        Subject: O=Grid, OU=lncc, OU=CA-ComCiDis, CN=host/giga05.lncc.br
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:bb:23:0e:09:c9:ae:14:9c:c3:2c:8d:cf:ae:27:
                    77:db:ff:ec:41:ca:dc:0a:61:c7:63:01:27:2e:37:
                    17:58:0b:bd:73:e7:ea:72:4d:c5:3d:f3:9c:1e:52:
                    83:0f:3d:21:72:4c:14:78:fd:f2:3b:5e:a6:a3:47:
                    15:aa:46:1e:bf:bb:40:c8:6a:d3:90:7d:25:a2:fe:
                    22:9e:3c:fd:39:f0:d3:67:6f:27:f3:8c:17:4a:43:
                    d3:45:1e:ce:c2:c5:fe:49:8f:53:73:af:ba:4a:3d:
                    b3:64:c3:f6:8f:05:f1:6d:28:2d:12:e2:b3:fb:61:
                    1d:55:a2:80:2a:2d:59:3f:95
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            Netscape Cert Type:
                SSL Client, SSL Server, S/MIME, Object Signing
    Signature Algorithm: md5WithRSAEncryption
        22:c8:44:fb:03:49:fe:f1:67:ed:2a:a3:24:6b:5d:78:9c:bb:
```

```
        55:c2:00:fb:09:82:84:23:76:a8:05:cf:51:2a:0f:6d:4b:0a:
        26:4f:02:68:09:72:d9:24:d9:46:d8:16:d1:c7:ed:2f:71:33:
        ac:34:54:cc:9e:23:44:a9:0b:7d:18:ae:8b:6d:f0:6e:32:10:
        d3:bc:c1:68:4b:15:e0:1e:8e:87:51:dd:11:9a:43:35:b3:51:
        d1:5d:1c:17:50:b7:79:73:05:20:9b:cd:27:13:ba:c3:df:78:
        c1:1b:52:a9:d3:0e:13:2c:d1:1d:2f:1c:f7:26:4a:ba:86:e3:
        c2:ae
-----BEGIN CERTIFICATE-----
MIICLjCCAZegAwIBAgIBDDANBgkqhkiG9w0BAQQFADBRMQ0wCwYDVQQKEwRHcmlk
MQ0wCwYDVQQLEwRsbmNjMRQwEgYDVQQLEwtDQS1Db21DaURpczEbMBkGA1UEAxMS
Q29tQ2lEaXMgU2ltcGxlIENBMB4XDTA3MDExMDE1MzQyMVoXDTA3MDExNTE1MzQy
MVowUjENMAsGA1UEChMER3JpZDENMAsGA1UECxMEbG5jYzEUMBIGA1UECxMLQ0Et
Q29tQ2lEaXMxHDAaBgNVBAMTE2hvc3QvZ2lnYTA1LmxuY2MuYnIwgZ8wDQYJKoZI
hvcNAQEBBQADgY0AMIGJAoGBALsjDgnJrhScwyyNz64nd9v/7EHK3Aphx2MBJy43
F1gLvXPn6nJNxT3znB5Sgw89IXJMFHj98jtepqNHFapGHr+7QMhq05B9JaL+Ip48
/Tnw02dvJ/OMF0pD00UezsLF/kmPU3Ovuko9s2TD9o8F8W0oLRLis/thHVWigCot
WT+VAgMBAAGjFTATMBEGCWCGSAGG+EIBAQQEAwIE8DANBgkqhkiG9w0BAQQFAAOB
gQAiyET7A0n+8WftKqMka114nLtVwgD7CYKEI3aoBc9RKg9tSwomTwJoCXLZJNlG
2BbRx+0vcTOsNFTMniNEqQt9GK6LbfBuMhDTvMFoSxXgHo6HUd0RmkM1s1HRXRwX
ULd5cwUgm80nE7rD33jBG1Kp0w4TLNEdLxz3Jkq6huPCrg==
-----END CERTIFICATE-----
```

E.4 Modelo de Chave Privada de Máquina

-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQC7Iw4Jya4UnMMsjc+uJ3fb/+xBytwKYcdjAScuNxdYC71z5+py
TcU985weUoMPPSFyTBR4/fI7XqajRxWqRh6/u0DIatOQfSWi/iKePP058NNnbyfz
jBdKQ9NFHs7Cxf5Jj1Nzr7pKPbNkw/aPBfFtKC0S4rP7YR1VooAqLVk/lQIDAQAB
AoGBAJsU7uJPn5L3Yos+rlgt3xeTXBAd9AIqi9jRMM4M26ycdhFipRi6uHWht8Qt
Lu1nBJu6ztCB6scmpkPzedHVME7aVMdfZzZALJcNgsTehqjJwkDOh2lglxpAdrke
8Cb11OH6RnceM4weCufFWWQqYQWTGEyd41I5j0tgfhMieTEVAkEA6oJ/qKdQQEca
AvrYL+y0mE1Ua2b7/PHL0Tkm4fxMfuZoNpGmEtpbwOwQVkcQtemOl5vtsmIFBCBY
SJuLDitsNwJBAMxJNZjMg8br+PO32PmHZqTpezbixMZflcNdZ34+2gIMntF79vd7
I6nUwBJUoboYW5ZoX8EYAhiO/SKklGjwxJMCQDUXSDlxb4ytzCjCi3EGILieQS9z
iDbystbyHmhpCxRrq0o3XSbFgqJBj01JF3Jnpt2R8pVA9avCetotRKT3la8CQQC/
yU9XJ1Hj8++iJ4y9HwAffM9fhOAF+QDqDAu4extC50RHGSGPz/x4WbMEwo2e33VE
rLKQ3pBBLCS8XFS0fK2fAkAEi+Coql+545O3kHkJsSMIOQYjgrP9sheifSDbydYp
Ww4wrfnufUjVGBg7QI5S/G7j5lGtPMl2wWxrdlMgZxg6
-----END RSA PRIVATE KEY-----

E.5 Modelo de Proxy Criado


```
-----BEGIN CERTIFICATE-----
MIICJDCCAY2gAwIBAgIENni/2TANBgkqhkiG9w0BAQQFADBcMQ0wCwYDVQQKEwRH
cmlkMQ0wCwYDVQQLEwRsbmNjMRQwEgYDVQQLEwtDQS1Db21DaURpczEQMA4GA1UE
CxMHbG5jYy5icjEUMBIGA1UEAxMLRmFiaW8gTGljaHQwHhcNMDcwMTAzMTMxMDU0
WhcNMDcwMTA0MDExNTU0WjBwMQ0wCwYDVQQKEwRHcmlkMQ0wCwYDVQQLEwRsbmNj
MRQwEgYDVQQLEwtDQS1Db21DaURpczEQMA4GA1UECxMHbG5jYy5icjEUMBIGA1UE
AxMLRmFiaW8gTGljaHQxEjAQBgNVBAMTCTkxMzg4MzA5NzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDTB8zTEGUy3D1DrmJXCXd/3g3+SL+XKF8rR09Cj7DRjqoi2XK1
HuJr9CkCpKmjT278mQN9VZFeJC5UawpAvWwNAgMBAAGjIzAhMB8GCisGAQQBm1AB
gV4BAf8EDjAMMAoGCCsGAQUFBxUBMA0GCSqGSIb3DQEBBAUAA4GBAKdjLIx/kTYg
qkfZKo242xPHq/jO7Px8HaZDa8Q4zk62eupshP2quWIZXbLS4y4LvpJ2pcr6hGGy
S/CqHuBi+9h58UqVB6kRhvYHwggPDTpewAd+dCEKIZzMo6KRj8EfrGA+U1PzAtPL
IKBQQP8y+vLI+n5yUiMThlkgsq/rTPnA
-----END CERTIFICATE-----
-----BEGIN RSA PRIVATE KEY-----
MIIBOgIBAAJBANMHzNMQZTLcPUOuYlcJd3/eDf5Iv5coXytHT0KPsNGOqiLZcrUe
4mv0KQKkqaNPbvyZA31VkV4kLlRrCkC9bA0CAwEAAQJACEqjKpvXThNHOPSOGptd
naGM7TPQi1DjxGCN5bCdou4kz0uRjwU+qfw0VVmfBeYzlhNfn1MFInCkNLaK/Wu0
RQIhAOl/zjFe2f+k3jIVj/g8D9icrlyh8JNQBA0CUVM6gqpXAiEA5122XgKUQJrb
rJsqVDKI6MOi5wLmFxAJZhUVRUta5jsCIBU3gsHjr4VWMsZ0RR+2PW5eVk2efWC1
XZjdBAy+/U6DAiEAxlcJ3XGske2vYDrsgJ2JzLDZ19weT6sJw59TSsIcvL0CIGAz
pa/a6s1ZQYIHhYW6sOIEoucyFB/Mb6caUkmvuE6+
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIICODCCAaGgAwIBAgIBDTANBgkqhkiG9w0BAQQFADBRMQ0wCwYDVQQKEwRHcmlk
MQ0wCwYDVQQLEwRsbmNjMRQwEgYDVQQLEwtDQS1Db21DaURpczEbMBkGA1UEAxMS
Q29tQ2lEaXMgU2ltcGxlIENBMB4XDTA2MTAwMzIwMzYyM1oXDTA3MTAwMzIwMzYy
M1owXDENMAsGA1UEChMER3JpZDENMAsGA1UECxMEbG5jYzEUMBIGA1UECxMLQ0Et
Q29tQ2lEaXMxEDAOBgNVBAsTB2xuY2MuYnIxFDASBgNVBAMTC0ZhYmlvIExpY2h0
```

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDCOQ/DFnza0ex/NT3JgiizPy11
eAaSfaGLZEW+UzGihw6InaUHWWcNXPA5065uhPaQVwoH7M2FqqKWAk67K38aXdEs
ApVGhZPCOf7DLtn29N8FDz/58zmpCp1/odlIg1H+kNPpn2385Bb3jh2kE3hHwLwZ
EVFMpsWxXLVgODNmVwIDAQABoxUwEzARBglghkgBhvhCAQEEBAMCBPAwDQYJKoZI
hvcNAQEEBQADgYEAri5stqE6EHIRtA5xXO3qlA6++9x2p8jHgdiOdLtOc0iD1iZV
F7zLm7r8W34KdcC7TndaE3wU0s1hXhcCjNrTmwITV8gk4f57vZWK8i1csHC9aVv1
VP/JjT+4Gq5J8fKhaZpN12BNY4sSGZrMm6PDqu+ioax5Kvt2S8/wIIaCVCw=
-----END CERTIFICATE-----

Referências Bibliográficas

[FOSTER02] I. Foster, What is the grid? A Three Point Checklist, Argonne National Laboratory & University of Chicago. 2002.

[FOSTER05] I. Foster, Globus Toolkit Version 4: Software for Service-Oriented Systems. IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2005

[FOSTER01] I. Foster, C. Kesselman, S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations International J. Supercomputer Applications, pp 15-17, 2001.

[FOSTER98] I.Foster, C.Kesselman, G.Tsudik, S.Tuecke, A Security Architecture for Computational Grids - Mathematics and Computer Science (Argonne) and Information Sciences Institute University of Southern California (Marina Del Rey). pp 83-92, 1998

[FOSTER97] I.Foster, C.Kesselman. Globus: A metacomputing infrastructure toolkit. International, Journal of Supercomputer Applications, pp 115-128, 1997

[GORANSON00] H. T. Goranson, Infrastructure for the advanced virtual enterprise: a report using a Brazilian-based example., INTERNATIONAL FEDERATION FOR INFORMATION PROCESSING;, 2000, Florianopolis. Proceedings? Massachusetts: Kluwer Academic Publishers. pp 47-62, 2000

[QI06] L. Qi, H. Jin, I. Foster, Jarek Gawor HAND: Highly Available Dynamic Deployment Infrastructure for Globus Toolkit 4, The Globus Alliance, 2006.

[PITANGA04] M.Pitanga, Computação em Grade - Uma Visão Introdutória; Artigo publicado Clube do Hardware, 2004

[PINHEIRO05] J.Pinheiro, F.Kon - Segurança em Grades Computacionais Departamento de Ciência da Computação, Instituto de Matemática e Estatística - Universidade de São Paulo, V Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. pp 65-112, 2005

[CHERVENAK02] A.Chervenak, I.Foster, C.Kesselman, C.Salisbury, S.TueckeThe, Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets, Journal of Network and Computer Applications, 2002

[SANTIN05] A.Santin, J.Fraga, E.Mello, F.Siqueira,Teias de federações como extensões ao modelo de autenticação e autorização SDSI / SPKI, pp 553-568, 2005

[REZENDE00] P.Rezende,Certificados Digitais, Chaves Púbicas e Assinaturas - O que são, como funcionam e como não funcionam, Publicado no Obseratório da Imprensa e-Notícia: "Internet, Riscos e Falácias", 2000

[VABSOLUTA06] verdade@bsoluta, http://www.absoluta.org/cripty/algoritmos.htm, acessada em julho 2006

[SOUZA01] C.Souza L.Mattos, Incorporação de Certificados SPKI/SDSI ao Protocolo SSL, I Workshop em Segurança de Sistemas Computacionais, Florianópolis - SC, 2001

[CARRIAO01] D.Carrião, A.Santin, C.Maziero, INTEGRANDO O MODELO DE SEGURANÇA SPKI/SDSI AO AMBIENTE DE GERÊNCIA WBEM, I Workshop em Segurança de Sistemas Computacionais, Florianópolis - SC, pp 1-12 2001

[GLOBUS06] Página do projeto Globus Alliance, http://www.globus.org/toolkit/docs/4.0/, acessada em julho de 2006

[SANTOS05] M.Santos, Computação em Grade - A Perspectiva do Gerenciamento de Dados, Departamento de Informática - PUC-Rio Programa de Pós-Graduação em Informática Disciplina: Seminários de Sistemas Distribuídos, Monografia, 2005

[BESTER04] J.Bester, I.Foster, C.Kesselmanz, J.Tedescoy, S.Tueckeo, GASS: A Data Movement and Access Service for Wide Area Computing Systems, Sptember 2004

[GARFINKEL96] S.Garfinkel, G.Spafford, Practical UNIX & Internet Security. O Reilly & Associates, 1996

[CZAJKOWSKI01] K.Czajkowski, S.Fitzgerald, I.Foster, C.Kesselman, Grid Information Services for Distributed Resource Sharing. In Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing, pp 181-194 2001.

[SILVA06] A.Silva, M.Stanton, Processamento dinâmico de caminhos de certificação em ambientes distribuídos de grande porte, Instituto de Computação, Instituto Nacional de Tecnologia da Informação, Universidade Federal Fluminense, 2006

[MENEZES96] A.Menezes, P.Oorschot, S.Vanstone, Handbook of Applied Cryptograpy, ISBN: 0-8493-8523-7, 1996

[GLOBUSSEC06] Página sobre segurança do projeto Globus Alliance, http://www.globus.org/toolkit/docs/4.0/security/, acessada em julho em agosto de 2006

[NCSA06] Página dos criadores do MyProxy, National Center for Supercomputing Applications (NCSA), http://grid.ncsa.uiuc.edu/myproxy/, acessada em maio de 2006

[RFC3820] S.Tuecke, V.Welch, D.Engert, L.Pearlman, M.Thompson, "Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile,"IETF RFC3280, June 2004.

[NOVOTNY01] Novotny, S.Tuecke, V.Welch, “An Online Credential Repository for the Grid: MyProxy,” Proceedings of the Tenth International Symposium on High

Performance Distributed Computing (HPDC-10), IEEE Press, August 2001, pp 104-111.

[OESCHSLIN06] P.Oeschslin, Cryptography and Security Laboratory of the Swiss Federal Institute of Technology (EPFL), http://www.netmarkt.com.br/noticia2003/1230.html, acessada em agosto de 2006

[JAVA06] Página do projeto Java da SUN Developer Network(SDN), http://java.sun.com/, acessada em outubro de 2006

[OPENSSL06] Página do projeto OpenSSL, http://www.openssl.org/, acessada em outubro de 2006

[RMI06] Página do Projeto RMI - Remote Method Invocation da SUN, http://java.sun.com/products/jdk/rmi, acessada em outubro de 2006

[MENDES03] Mendes Hammurabi das Chagas, Usando OpenSSL Uma implementação livre do protocolo SSL, Trabalho da Disciplina Segurança de Dados 103, Universidade de Brasília, 13 de Julho de 2003

[WELCH05] V.Welch, I.Foster, C.Kesselman, O.Mulmo, L.Pearlman, S.Tuecke, J.Gawor, S.Meder, F.Siebenlist, "X.509 Proxy Certificates for Dynamic Delegation", 2005

[BHATIA05] K.Bhatia, S.Chandra, A.Lin, K.Mueller, V.Veytser, C.Youn, "Engineering an End-to-End GSI-based Security Infrastructure", 2005

[ReD100] Página onde são feitas eleições anuais dos 100 projetos mais inovadores do ano, também chamada "Oscar da Invenção", http://www.rdmag.com/awards.html, acessada em dezembro de 2006

[VCG07] Página do Virtual Community Grid, grupo de trabalho da Rede Nacional de Pesquisa (GT-VCG/RNP), http://vcg.lncc.br, acessada em janeiro de 2007

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo