Um Servidor de Nomes Distribuído

Antônio Augusto Fröhlich, Thadeu Botteri Corso & Luis Fernando Fausto

Universidade Federal de Santa Catarina
Centro Tecnológico
Pós Graduação em Ciências da Computação
88.049.970 - Florianópolis - SC - Brasil
Tel.:(048)231-9738 Fax:(048)231-9770
E-mail: {guto,thadeu,fausto}@inf.ufsc.br

Resumo

Este artigo descreve um servidor de nomes distribuído que permite mapear nomes de objetos em identificadores, independentemente de suas localizações. Esse servidor permite definir espaços de nomes locais ou global, conforme as necessidades do usuário. Ele incorpora um mecanismo de autenticação que, após validar uma solicitação, fornece ao cliente uma chave que pode ser usada por outros servidores como critério de segurança.

A versão atual do servidor de nomes deve ser utilizada especificamente como servidor de nomes de arquivos no sistema de arquivos distribuído PYXIS, que adota o mecanismo de autenticação para validar requisições a todos os seus servidores componentes. Essa versão suporta um espaço hierárquico de nomes distribuído que é apresentado aos usuários como uma árvore de diretórios.

Abstract

This paper describes a distributed name server, which maps object names (independently of its location) into identifiers. This server dynamicly supports global or local name spaces, according to system requirements. The server incorporates an authentication engine that, after checking a request, suply the client with an authentication key. Such key can be adopted by other servers as a security criteria for further communications with clients.

At this time, an operating version is available. This version acts as the file name server for the PYXIS distributed file system, which uses the server authentication capability to validate requests to all other component servers. In that environment, the name server is in charge of supporting a distributed hierarchic naming space visible to the users like a directory tree.

1 Introdução

A crescente distribuição do *hardware* dos sistemas computacionais, vem motivando o surgimento de inúmeras idéias inovadoras na área dos sistemas operacionais distribuídos. A especialização dos serviços entre vários servidores é uma destas idéias que, apesar de se adaptar perfeitamente a sistemas distribuídos, ainda é pouco explorada. Agrupando tarefas relacionadas em servidores específicos, podemos ter módulos mais funcionais, autônomos e que possivelmente aproveitarão melhor o paralelismo do *hardware*. Com esta modularização, torna-se possível a integração de vários servidores diferentes, cada qual responsável por uma classe de serviços, para a formação de um conjunto adequado às necessidades do sistema.

Uma especialização freqüente é o servidor de nomes, responsável pelo mapeamento de nomes de objetos em identificadores. O servidor de nomes pode ser implementado de forma genérica, possibilitando o mapeamento de qualquer tipo de nome em qualquer tipo de identificador. Duas utilizações comuns do servidor de nomes envolvem o mapeamento de nomes de arquivos e de *hosts*, mas outras são possíveis.

Este artigo descreve um servidor de nomes, enfatizando suas características inovadoras que incluem a possibilidade da definição de espaços de nomes locais ou de um único espaço de nomes global e a capacidade de autenticação dos acessos aos objetos. Ele descreve também uma versão do servidor especificada para atuar como servidor de nomes de arquivos do sistema de arquivos distribuídos PYXIS [FRO 94].

2 Objetivos do Projeto

O projeto do servidor de nomes distribuído aqui descrito é suficientemente versátil para contemplar os seguintes objetivos:

- Distribuição do espaço de nomes: É importante em um sistema distribuído que os usuários possam designar objetos remotos, transparentemente, através de nomes locais, ou seja, sem explicitar a sua localização.
- Generalidade: A especificação do servidor não define os tipos dos nomes mapeados. O usuário provê um nome, que perante o servidor é apenas uma cadeia de caracteres de tamanho variável. Da mesma forma, os identificadores associados aos nomes somente tem sentido para o processo que os definiu. Assim o servidor consegue manter-se genérico, pois não precisa considerar o significado dos dados (nomes e identificadores) que manipula.
- Autenticação de Mensagens: O primeiro contato do usuário com o objeto que ele quer acessar é através servidor de nomes, pois para acessá-lo ele deve antes obter seu identificador. Assim sendo, o servidor de nomes é o ponto mais apropriado para a implementação de um mecanismo de autenticação. Se a autenticação fosse postergada para uma etapa posterior a obtenção do identificador, seria impossível impedir que o identificador fosse utilizado de maneira indevida. A autenticação

executada pelo servidor de nomes gera uma chave de acesso que pode ser utilizada pelos demais servidores como critério de segurança.

- Descentralização das informações de controle: As informações necessárias para a execução das tarefas do servidor de nomes não estão concentradas nele. Cada cliente mantém as informações de controle necessárias para realizar as suas solicitações aos servidores. Isto facilita o processo de reativação de um servidor após uma falha, além de simplificar a manutenção das tabelas internas do servidor.

3 Espaço de Nomes

Para nomear um número grande de objetos de forma não ambígua, uma estrutura linear de entradas provavelmente implicaria em grandes dificuldades administrativas. Uma organização hierárquica de objetos, além de ser mais facilmente gerenciada, é mais próxima da forma como os objetos se encontram organizados na natureza.

A finalidade de uma estrutura de nomes é prover acesso a objetos que existem em algum universo. Um objeto pode ser qualquer coisa naquele universo, desde que possa ser nomeado.

Um nome é uma cadeia de caracteres que identifica um objeto particular no conjunto de todos os objetos. Um nome não deve ser ambíguo, isto é, deve indicar somente um objeto. Entretanto, um nome não precisa ser único, isto é, mais de um nome pode indicar o mesmo objeto sem ambigüidade.

A hierarquia de nomes é composta de entradas organizadas em tabelas, cada qual contendo informações sobre um único objeto. Para cada objeto particular existe uma ou mais entradas que formam o conjunto primário das informações sobre aquele objeto. Esse conjunto contém, no mínimo, o nome do objeto, mas pode também conter outras informações quaisquer. O conjunto dessas tabelas hierarquicamente acopladas define um espaço de nomes.

Um espaço de nomes é então o conjunto de todos os nomes conhecidos em um determinado domínio. Se um domínio estiver restrito a um único *host*, então diz-se que o espaço de nomes é local aquele *host*. Caso um domínio agregue vários *hosts*, então diz-se que o espaço de nomes é global aquele conjunto de *hosts*.

Ao contrário de um espaço de nomes local, um espaço de nomes global agrega nomes de objetos distribuídos por vários *hosts*. Da maneira como está definido, todos os objetos referenciados em um espaço de nomes global são acessíveis, a partir de qualquer *host*, sempre com o mesmo nome, independentemente de sua localização. A figura 1 apresenta um exemplo de espaço de nomes global, onde "Y" é uma tabela contida no *host* "1" e que faz referência a objetos ("Z" e "d") contidos no *host* "2".

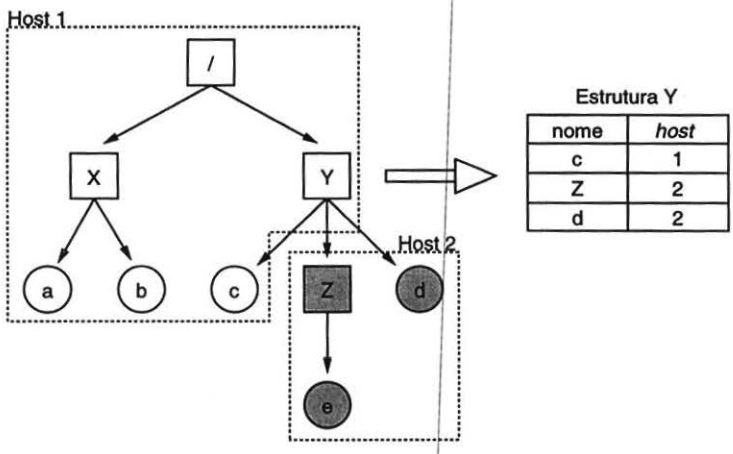

Figura 1: Um espaço de nomes global

4 Segurança dos Sistemas

As informações armazenadas em um sistema (dados e código), bem como os componentes físicos do sistema computacional, precisam ser protegidos de acessos não autorizados, destruição ou alteração maliciosa, e introdução de inconsistências acidentais. Dizemos que um sistema é seguro se seus componentes são utilizados e acessados conforme o previsto em todas as circunstâncias [SIL 94]. Infelizmente não é possível prover segurança total aos objetos do sistema, mas deve-se utilizar mecanismos para garantir que quebras da segurança do sistema sejam exceções, e não norma.

Violações da segurança do sistema podem ser categorizadas como intencional ou acidental. É mais fácil proteger um sistema contra um acesso indesejável acidental do que intencional. As principais formas de acesso indesejáveis intencionais são: leitura não autorizada de um objeto, modificação não autorizada de um objeto e destruição não autorizada de um objeto.

Se proteção absoluta contra o mal uso do sistema não é possível, o custo para o invasor deve ser suficientemente alto para deter a maioria dos acessos não autorizados às informações. Para proteger um sistema deve-se tomar medidas de segurança em dois níveis:

- Material: Os locais que contém os computadores que compõem o sistema devem estar fisicamente seguros contra entradas não autorizadas de invasores.
- Humano: Os usuários devem ser cuidadosamente autorizados para reduzir as chances de um usuário dar acesso à um invasor em troca de um prêmio ou favor.

A segurança nestes dois níveis deve ser mantidas para que se possa garantir a segurança

global do sistema. Uma fraqueza em um dos dois níveis (material ou humano) possivelmente possibilitará acessos indevidos aos componentes do sistema.

4.1 Autenticação de mensagens

Os objetos manipulados por um sistema possuem características próprias que devem ser respeitadas por todos os que queiram acessá-los. Nem todos os clientes tem os mesmos direitos de acesso aos objetos, donde surgem problemas de como barrar acessos não autorizados. A proteção do sistema depende da habilidade de identificar os processos que estão executando e seus respectivos donos (usuários). Geralmente, a autenticação é baseada em uma combinação de três elementos: uma informação de posse do usuário (chave ou cartão), a identificação do usuário e a impressão digital ou assinatura do usuário.

A abordagem mais comum para autenticação de usuários é através de senhas. Quando um usuário fornece um identificador ou um nome de conta é solicitada uma senha. Se esta senha é igual à armazenada (criptografada) no sistema, assume-se então que o usuário é realmente quem diz ser. Além de autenticar usuários, senhas podem também ser utilizadas para controlar acessos a objetos. Por exemplo, uma senha pode ser associada a cada objeto. Se for necessário um nível de granularidade de acesso mais fino, é possível associar-se diferentes senhas para diversos tipos de acesso. Por exemplo, pode-se ter senhas diferentes para leitura, atualização ou remoção de um objeto.

Embora hajam alguns problemas associados às senhas, elas são freqüentemente usadas, por serem de fácil emprego. O problema com as senhas está relacionado à dificuldade de guardá-las secretamente. Elas podem ser comprometidas quando expostas acidentalmente ou por transferência de um usuário autorizado para outro não autorizado.

Existem dois meios comuns de se descobrir uma senha. Um deles é o instruso conhecer o usuário e obter informações sobre ele: freqüentemente as pessoas utilizam informações óbvias, tais como a data de nascimento, como suas senhas. O outro modo é a força bruta, ou seja, tentando todas as possíveis combinações de letras, números e pontuação. Senhas curtas não são difíceis de se obter através de repetidas tentativas. Por exemplo, uma senha de 4 dígitos pode gerar apenas 10.000 variações. Em média, em 5000 tentativas a senha é descoberta. Se algum programa for capaz de tentar uma senha a cada milisegundo, não levará mais do que 5 minutos para descobri-la. Senhas grandes não são tão suscetíveis a serem descobertas por enumeração, mas com um pouco mais de trabalho elas serão igualmente descobertas.

Com isto surge a necessidade de se implementar um outro mecanismo para autenticar o acesso aos objetos a fim de torná-los menos suscetíveis a acessos indevidos. A solução adotada pelo servidor de nomes está baseada na geração de chaves para serem usadas pelos demais servidores do sistema na autenticação de solicitações.

4.2 Criação da chave

A chave gerada pelo servidor é função do próprio identificador do objeto, da data de sua criação e do identificador do usuário. A figura 2 mostra o processo de criação da chave. A

UFRGS
INSTITUTO DE INFORMÁTICA
BIBLIOTECA

função de geração da chave não possui inversa, o que inviabiliza a descoberta do método utilizado para gerar as chaves de autenticação.

A inclusão da data de criação do objeto na composição da chave de autenticação deve-se ao fato de que um usuário pode obter um identificador de um objeto e a respectiva chave e ficar algum tempo sem acessá-lo. Durante este tempo, o objeto pode ser destruído e seu identificador reutilizado para designar um novo objeto. Isto é matematicamente possível, uma vez que identificadores são recursos finitos. Nesse caso, o usuário do objeto destruído acessaria um objeto que não lhe pertence. A inclusão da data de criação do objeto na chave, elimina este problema.

O identificador do usuário foi envolvido na geração da chave para evitar que um usuário obtenha uma chave de acesso a um objeto e depois passe-os para outro usuário não autorizado. Desta forma impede-se, por exemplo, que um usuário autorizado a ler um determinado objeto, passe o identificador e a chave do mesmo a um outro usuário sem tal permissão, gerando então uma falha na proteção dos objetos.

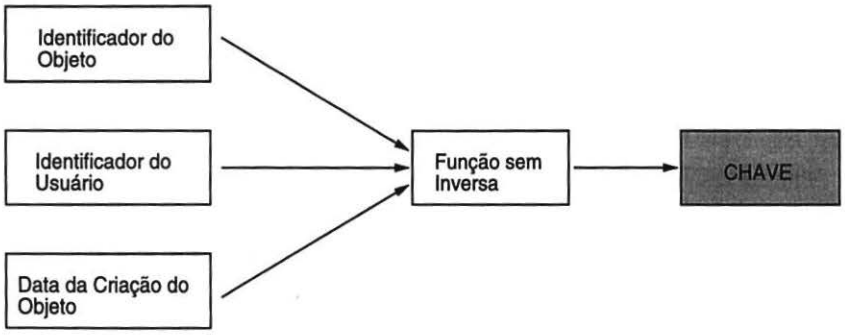

Figura 2: Processo de geração da chave de autenticação

5 Realização do Servidor

A versão atual do servidor trata os nomes de arquivos no projeto do sistema de arquivos PYXIS, que introduz uma proposta inovadora para esses sistemas, baseada numa visão, projetada sobre um futuro próximo, onde um ambiente computacional possuirá um número de unidades processadoras equiparável ao número de tarefas e onde as entidades comunicantes disporão de canais de comunicação com velocidades hoje encontradas apenas em barramentos internos de computadores.

O PYXIS faz uso do servidor de nomes para permitir o acesso transparente a arquivos remotos, para mapear nomes de arquivos em identificadores internos e para gerar chaves que garantirão a autenticação das mensagens entre os seus servidores e clientes. A transparência de localização dos objetos é alcançada através da adoção de identificadores globais de objetos, capazes de designar qualquer objeto no domínio da INTERNET.

O servidor de nomes não mantém informações contextuais a respeito do estado dos seus clientes. Cada solicitação deve conter todas as informações necessárias para a sua realização. Isto também torna mais fácil o processo de reativação do servidor no caso de uma falha. Uma vez que ele não mantém nenhuma informação contextual, o processo de reativação é igual ao de inicialização.

O servidor de nomes comunica-se com o servidor de arquivos do PYXIS para que o mesmo valide o acesso aos arquivos, verificando as permissões que o usuário possui sobre eles. A utilização de funções de outros servidores visa manter o servidor de nomes o mais genérico possível, de tal forma que os métodos de validação de acesso a um determinado objeto sejam implementados pelo servidor responsável pela classe do objeto.

Em um nível superior, o servidor esconde toda a sua estrutura interna, mapeando externamente os seus serviços para que sejam compatíveis com o UNIX [BAC 87]. Isso possibilita que aplicações desenvolvidas em conformidade com essa norma acessem o sistema de arquivos sem qualquer adaptação.

Os serviços que o servidor de nomes provê aos demais servidores do PYXIS e aos clientes são os seguintes:

- `translate`: Este é o principal serviço do servidor de nomes, pois é através dele que os nomes de arquivos são mapeados em identificadores. Além disso, é este serviço que provê ao usuário a chave de autenticação. Embora seja o serviço mais importante do servidor, ele não é visível ao usuário final, pois aparece sempre encapsulado em funções de biblioteca (*stubs*).
- `link`: Este serviço possibilita a anexação de nomes de arquivos, possivelmente remotos, à arvore de diretórios.
- `opendir`: Abre um diretório
- `chdir`: Muda o diretório corrente
- `closedir`: Fecha um diretório
- `readdir`: Lê uma entrada de um diretório
- `rmdir`: Remove um diretório
- `rewinddir`: Altera a posição do ponteiro de entradas para o início do diretório.

O processo de inicialização do servidor de nomes pressupõe o fornecimento do identificador do arquivo correspondente à raiz da árvore de diretórios. No caso de se ter apenas um servidor de nomes no domínio, ou se todos os servidores do domínio forem inicializados com a mesma raiz, teremos um espaço de nomes único. Se, por outro lado, cada servidor for inicializado com uma raiz própria poderão existir espaços de nomes distintos no domínio. Nada impede que árvores de diretórios independentes tenham sub-árvores comuns. Assim sendo, cabe aos usuários decidir se desejam ter um espaço de nomes único e compartilhado por todos os *hosts* ou independente para cada *host*.

O servidor faz uso de *links* remotos, que são implementados no PYXIS para prover a transparência de localidade ao usuário. Com isto, pode-se anexar um arquivo de um

nodo remoto, ou até mesmo uma sub-ávores remota, à árvore local, fazendo com que estes arquivos e diretórios sejam mapeados localmente.

Nome Local *(link)*	Nome Remoto	Tipo
Deluana	venus.inf.ufsc.br:/usr/Deluana	diretório
Ledir	apolo.inf.ufsc.br:/home/vesta/elerson	diretório
mosaic	marte.inf.ufsc.br:/usr/local/www/mosaic	arquivo
Schiffer	venus.inf.ufsc.br:/usr/local/www/pics/Claudia	arquivo

Figura 3: Exemplo de mapeamento local de nomes remotos

A utilização do mecanismo de *links* remotos pode degenerar a árvore de diretórios em um grafo cíclico, o que acarretaria problemas em aplicações como "tar" e "find", que percorrem toda a árvore. Todavia, a maneira como os diretórios são implementados no PYXIS permite às aplicações diferenciar arquivos locais de remotos, tornando possível a prevenção de laços infinitos e minimizando os efeitos negativos da ocorrência de ciclos no grafo de diretório.

6 Conclusões

Este artigo apresentou um servidor de nomes distribuído que provê mecanismos para autenticação de mensagens e a transparência de localidade de objetos. A versão atual comprova a funcionalidade do servidor quanto aos objetivos propostos. Os resultados já obtidos são muito satisfatórios, o que incentiva a continuidade do projeto no sentido de conseguir um servidor de nomes ainda mais genérico e com maior capacidade de distribuição.

O servidor de nomes é acessado, normalmente, apenas uma vez para cada objeto que um processo referencia. A baixa freqüência das requisições de serviço não impõe grandes restrições de tempo ao servidor de nomes. Contudo, resultados preliminares mostram que o tempo de mapeamento de um objeto e a conseqüente geração de uma chave não é muito maior, no caso do servidor de arquivos, que a validação de acesso pelo servidor associado ao objeto.

Uma avaliação mais precisa do desempenho do servidor depende da migração do PYXIS para sua plataforma alvo: um multicomputador com rede de interconexão dinâmica hoje em desenvolvimento [COR 93].

Referências

[BAC 87] BACH, M., *The Design of the UNIX Operating System*, Englewood Cliffs: Prentice-Hall, 1987.

[RFC 1630] BERNERS-LEE, T., *Universal Resource Identifiers in WWW*, Menlo Park: Network Information Center, 1994 (Internet Request for Comment).

[CCI 88] THE INTERNATIONAL TELEGRAPH AND TELEPHONE CONSULTATIVE COMMITTEE, *Data Comunication Networks Directory*, CCITT: Recommendations X.500 - X.521, 1994.

[COR 93] CORSO, T., *Ambiente para Programação Paralela em Multicomputador*, Florianópolis: UFSC/CTC/INE, 1993 (Relatório Técnico).

[FRO 94] FROHLICH, A., *PYXIS:Um Sistema de Arquivos Distríbuido.*, Florianópolis: UFSC, 1994 Dissertação(Mestrado).

[GAR 93] GARFINKEL, S. & SPAFFORD, G., *Pratical Unix Security*, Sebastopol: O'Reilly & Associates, 1993.

[RFC 1034] MOCKAPETRIS, P., *Domain Names - Concepts and Facilities*, Menlo Park: Network Information Center, 1987 (Internet Request for Comment).

[SIL 94] SILBERSCHATZ, A. & GALVIN, P., *Operating Systems Concepts*, Reading: Addison-Wesley, 1994.

[TAN 87b] TANENBAUM, A., *Operating Systems: Design and Implementation*, Englewood Cliffs: Prentice-Hall, 1987.

[TAN 92] TANENBAUM, A., *Modern Operating Systems*, Englewood Cliffs: Prentice-Hall, 1992.

[TAN 92b] TANENBAUM, A., *The Amoeba Distributed Operating System*, Amsterdam: Vrije Universiteit, 1992 (Relatório Técnico).

[TAN 92c] TANENBAUM, A., *Using Sparse Capabilities in a Distributed Operating System*, Amsterdam: Vrije Universiteit, 1992 (Relatório Técnico).