Luiz Marcelo de Oliveira

Estudo sobre serviço de diretórios distribuídos para instituições acadêmicas

São José – SC

dezembro / 2010

Luiz Marcelo de Oliveira

Estudo sobre serviço de diretórios distribuídos para instituições acadêmicas

Monografia apresentada à Coordenação do Curso Superior de Tecnologia em Sistemas de Telecomunicações do Instituto Federal de Santa Catarina para a obtenção do diploma de Tecnólogo em Sistemas de Telecomunicações.

Orientador:
Prof. Emerson Ribeiro de Mello, Dr.

CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS DE TELECOMUNICAÇÕES
INSTITUTO FEDERAL DE SANTA CATARINA

São José – SC

dezembro / 2010

Monografia sob o título *"Estudo sobre serviço de diretórios distribuídos para instituições acadêmicas"*, defendida por Luiz Marcelo de Oliveira e aprovada em 23 de dezembro de 2010, em São José, Santa Catarina, pela banca examinadora assim constituída:

Prof. Emerson Ribeiro de Mello, Dr.
Orientador

Prof. Marcelo Maia Sobral, M. Sc.
IFSC

Prof. Ederson Torresini, M. Sc.
IFSC

No meio da dificuldade encontra-se a oportunidade.

A. Einstein

Agradecimentos

Dedico o esforço na realização desse trabalho, as pessoas que eu amo. Ao meus pais, pelo incentivo e valores ensinados durante minha vida e minha namorada, companheira que sempre esteve ao meu lado.

Agradeço ao professor Emerson Ribeiro de Mello pela orientação e auxílio na organização desse trabalho. Por final, agradeço aos profissionais e entusiastas participantes do grupo de discussão oficial do *OpenLDAP*[1], pela troca de experiências que auxiliaram na implementação prática desse trabalho.

[1] `http://www.openldap.org/lists/mm/listinfo/openldap-software`

Resumo

Um serviço de diretórios é uma base de dados hierárquica otimizada para fornecer com alto desempenho, informações a múltiplos serviços em uma rede. Por meio do (*Lightweight Directory Access Protocol* – LDAP), esses serviços são integrados ao diretório, resultando na centralização das informações em uma única fonte de dados. Dentro dos principais propósitos de um serviço de diretórios, destaca-se o armazenamento de identidades de usuários para autenticação em diversos serviços e aplicações. Um serviço de diretórios pode operar dentro de um único domínio de rede ou em ambientes inter-domínios, nesse caso, o diretório caracteriza-se pelo seu comportamento distribuído. Quando distribuído, o serviço de diretórios oferece disponibilidade de acesso e escalabilidade nas informações que armazena. Nesse trabalho é apresentado as técnicas existentes para implementação de um serviço de diretórios distribuídos, bem como os conceitos relacionados a sua implementação. O estudo tem o foco em apresentar o serviço de diretórios adequado a instituições acadêmicas, levando em consideração as principais finalidades do serviço de diretórios nesse contexto.

Abstract

A directory service is a hierarchical database optimized to provide high performance, information to multiple services on a network. Through (*Lightweight Directory Access Protocol* – LDAP), these directory services are integrated, resulting in the centralization of information in a single data source. Within the main purpose of a directory service, highlight the storage of user identities for authentication to various services and applications. A directory service can operate within a single network domain or inter-domain environments, in this case, the directory is characterized by its distributed behavior. When distributed, the directory service provides access availability and scalability of the information it stores. This paper shows the existing techniques for implementing a distributed directory service, as well as the concepts related to its implementation. The study has focused on presenting the proper directory service to academic institutions, taking into account the main purpose of the directory service in this context.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de códigos

1 Introdução

O conceito elementar de diretórios faz parte de nossa vida sem ao menos percebermos. Quando usamos um dicionário para buscar o significado de uma palavra, ou então uma lista telefônica para encontrar um número específico, ou ainda, quando utilizamos um computador para salvar nossas fotos, músicas e arquivos pessoais nas suas respectivas pastas, somos beneficiados pela forma como as informações estão organizadas. O diretório na sua essência é isso: armazenar e organizar informações de forma que as buscas sejam simples e rápidas.

No campo das redes de computadores, um diretório atua como um centralizador de informações. A princípio, qualquer informação pertinente a uma rede pode ser armazenado em um diretório. O exemplo mais adequado de atuação do diretório e que predominantemente é aplicado na prática, é o armazenamento de informações de usuários. O diretório é capaz de fornecer informações de usuários para diferentes serviços na rede.

1.1 Motivação

A ideia inicial desse trabalho surgiu a partir de uma análise de como o serviço de diretórios está implementado no (Instituto Federal de Santa Catarina – IFSC). A partir dessa análise, iniciou-se um levantamento da demanda de acesso ao diretório bem como suas qualidades e carências dentro do contexto da instituição. Foi pensado por exemplo, como um aluno poderia autenticar-se na rede de outros campi com o mesmo *logon* de rede utilizado no domínio da rede de seu campus origem. O diretório da forma como se apresenta não é capaz de atender esse requisito. Justamente pelo fato de que, cada campus possui seu diretório próprio, sem qualquer relação com os demais.

No entanto, existem situações em que a disponibilidade compartilhada das informações de usuários entre os campi torna-se interessante, por exemplo, um aluno matriculado em um curso ministrado em um determinado campus da instituição tem acesso à rede de seu campus por meio de uma identidade de usuário. Esse mesmo aluno também está matriculado em uma

bolsa de pesquisa oferecida em outro campus da instituição. Consequentemente, o aluno possui uma outra identidade de usuário específica para o contexto desse campus. Nesse caso, o aluno possui duas identidades distintas dentro da instituição, cada uma específica para o acesso em cada campus.

Se a instituição aplicasse um modelo de armazenamento de identidades que através de uma infraestrutura de diretórios distribuídos, integrasse a base de informações de todos campi, o aluno precisaria apenas de uma única identidade para firmar seu acesso no domínio da rede de qualquer campus.

Paralelamente a essa motivação, o trabalho ainda considerou a ideia de como uma instituição acadêmica poderia fazer parte da infra estrutura de federação acadêmica promovida pela (Rede Nacional de Pesquisa – RNP). Uma federação acadêmica é um ambiente que fornece o intercâmbio de serviços entre instituições de ensino. No Brasil esse intercâmbio entre instituições é promovido pela (Comunidade Acadêmica Federada – CAFe).

As instituições participantes da Federação CAFe, através de uma relação de confiança inter domínios, permitem que os usuários vinculados a uma instituição, usufruam de serviços oferecidos pelas instituições participantes. Tudo isso por meio de uma única identidade de acesso.

Os requisitos básicos para promover essa Infraestrutura de autenticação federada, envolve a implementação de um serviço de diretórios, cuja função é disponibilizar informações pertinentes a usuários, como nome, data de matricula, tipo de vínculo com a instituição, entre tantas outras.

1.2 Objetivos

Esse trabalho tem o objetivo de apresentar um estudo sobre serviço de diretórios distribuídos, aplicado a instituições acadêmicas, levando em consideração, os requisitos técnicos que envolvem a sua implementação. No decorrer do desenvolvimento desse trabalho, os seguintes tópicos serão abordados:

- Fundamentação teórica sobre serviço de diretórios. Sua definição, conceitos e benefícios de sua aplicação;
- Modelo de informação do diretório, contendo informações de usuários para autenticação em sistemas baseados em *Linux* e para atender aos requisitos mínimos exigidos pela Federação CAFe;

- Formas de organizar as informações no diretório de acordo com um propósito específico;
- Modelos de distribuição do diretório utilizando o *software* de código aberto *OpenLDAP*;
- Questões relacionadas a segurança. Métodos de autenticação, autorização de acesso e confidencialidade no tráfego de informações pela rede;
- Experimentos práticos envolvendo os conceitos apresentados, a fim de simular a aplicação do serviço de diretório em um ambiente real de uma instituição acadêmica.

1.3 Organização do texto

Esse trabalho está organizado da seguinte forma: no Capítulo 2 é realizada uma abordagem sobre as principais formas de armazenar informações em uma rede. É dado ênfase no modelo de armazenamento baseado em diretórios e toda fundamentação do (*Lightweight Directory Access Protocol* – LDAP), protocolo responsável por oferecer um serviço de diretórios.

Uma breve descrição é apresentada sobre a (Comunidade Acadêmica Federada – CAFe) apontando suas características e requisitos técnicos exigidos para o ingresso de uma instituição de ensino na federação. Por final, é apresentado uma introdução ao *OpenLDAP* que será a ferramenta de implementação do ambiente proposto nesse trabalho.

O Capítulo 3 apresenta os modelos existentes para a implementação de um serviço diretórios distribuídos aplicado a instituições acadêmicas. São apresentadas todas etapas envolvidas no planejamento de um diretório distribuído e no final, uma conclusão apresenta uma análise comparativa entre os modelos apresentados no decorrer do desenvolvimento.

No Capítulo 4 são apresentados os experimentos práticos. O objetivo desses experimentos é avaliar o comportamento do serviço de diretórios nos modelos de distribuição apresentados no capítulo 3.

2 *Fundamentação teórica*

2.1 Modelos de armazenamento

Armazenar informações e recuperá-las para uma leitura posterior faz parte de nosso cotidiano. Uma lista de compras escrita no papel, um caderno com notas de aula ou uma agenda telefônica com todos os contatos, são exemplos comuns de como as informações podem ser armazenadas e a forma como elas podem ser organizadas para facilitar as consultas. No campo da computação, armazenar informações segue o mesmo princípio. A seguir são apresentadas algumas formas de armazenamento e organização das informações.

2.1.1 Arquivos texto

Armazenamento baseado em arquivo texto é o método mais simples de armazenar informações. Normalmente as informações são organizadas utilizando delimitadores específicos como por exemplo vírgulas, aspas ou dois pontos. A utilização dos delimitadores formam no arquivo texto uma estrutura tabelar composta por linhas e colunas. Um bom exemplo desse método de organização é o formato (*Comma-Separated Values* – CSV) (SHAFRANOVICH, 2005). O formato CSV utiliza vírgulas como delimitador de colunas e o caractere de quebra de linha para separar as linhas. A Figura 2.1(a) ilustra um exemplo de arquivo texto no formato CSV.

```
Camila,28,São Paulo,Jornalista
Rogério,54,Rio de Janeiro,Eng Civil
Mauro,37,Curitiba,Arquiteto
Alex,41,Florianópolis,Clínico Geral
```

(a) Arquivo texto no formato CSV

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
```

(b) Trecho de um arquivo passwd

Figura 2.1: Arquivos textos utilizando delimitadores de campo.

Os sistemas baseados em Unix em geral armazenam as informações de contas de usuários locais no arquivo */etc/passwd* (Figura 2.1(b)). Cada linha desse arquivo representa um registro

de usuário cujos campos são delimitados pelo caractere dois pontos. Esse formato permite o acesso a determinado campo utilizando ferramentas exclusivas para filtro de texto como o *grep* e o *cut*.

A desvantagem de armazenar informações em arquivos texto é que a aplicação responsável por consultar o arquivo, deverá ler linha por linha para encontrar a informação desejada. Em situações que exijam armazenar centenas de milhares de registros, o tempo de processamento para uma consulta pode tornar-se indesejável.

2.1.2 Banco de dados relacional

O conceito de banco de dados relacional surgiu na década de 70 através de pesquisas de funções de automação de escritório. O primeiro modelo de banco de dados relacional foi criado por um pesquisador da IBM, Ted Codd, que através de cálculo e álgebra relacional, desenvolveu um modelo de armazenamento que permitia o acesso aos dados armazenados em tabelas, através de uma série de comandos (SHELDON, 2003). Três elementos compõem um banco de dados relacional:

- **Campos:** Um campo ou também chamado de atributo é um espaço reservado para armazenar uma unidade de informação. Essa informação deve ser de um tipo específico (número inteiro, palavra *string* ou um binário (*boolean*), por exemplo);
- **Registros:** Um registro é um conjunto de campos que dá características a um objeto qualquer (um produto, um local, uma pessoa);
- **Tabelas:** Uma tabela abriga um conjunto de registros de natureza comum, ou seja, uma tabela armazena registros de pessoas, enquanto outra tabela armazena registros de produtos, por exemplo. Os registros representam as linhas da tabela enquanto os campos representam as colunas.

O princípio do modelo relacional é permitir o relacionamento entre tabelas e consequentemente o relacionamento entre registros. A Figura 2.2 ilustra essa relação utilizando um exemplo genérico de um banco de dados que armazena as vendas de uma empresa.

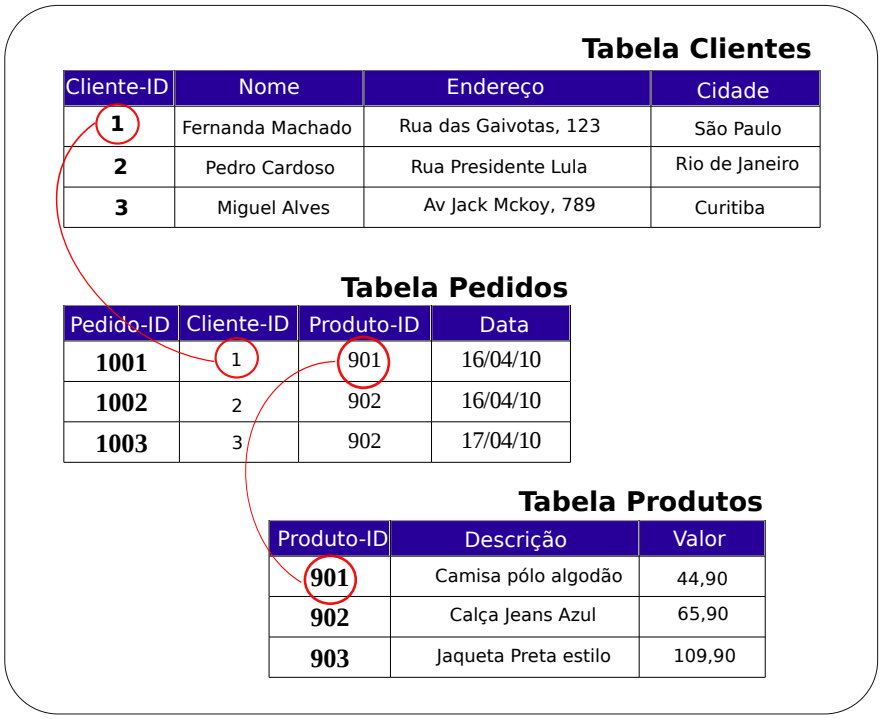


Figura 2.2: Relacionamento entre tabelas.

Toda tabela no modelo relacional deve obrigatoriamente possuir um campo, cujo valor armazenado seja utilizado para identificar, de forma única, os registros nela armazenados. Esse campo é chamado de chave primária. Na tabela Clientes da Figura 2.2, a chave primária é representada pelo campo `Cliente-ID`. É possível observar que na tabela Pedidos, o campo `Cliente-ID` armazena valores do campo `Cliente-ID` da tabela Clientes. Com isso, registros da tabela Pedidos criam um relacionamento com os registros da tabela Clientes. O campo responsável por armazenar um valor, cuja representação é a chave primária de outra tabela, chama-se chave estrangeira.

A consulta aos dados de um banco de dados relacional se dá com a linguagem (*Structured Query Language* – SQL) (GROFF; WEINBERG, 2002). O SQL define dezenas de operações para manipulação na base de dados, entre as principais estão: *SELECT*, *INSERT*, *DELETE* e *UPDATE*. Utilizando o exemplo da Figura 2.2, supondo que desejamos consultar todas as compras (registros) realizadas pela cliente Fernanda Machado, uma consulta SQL para esse caso seria algo a apresentada pela Listagem 2.1.

```
1 SELECT * FROM Tabela_Pedidos WHERE Cliente_ID=1
```

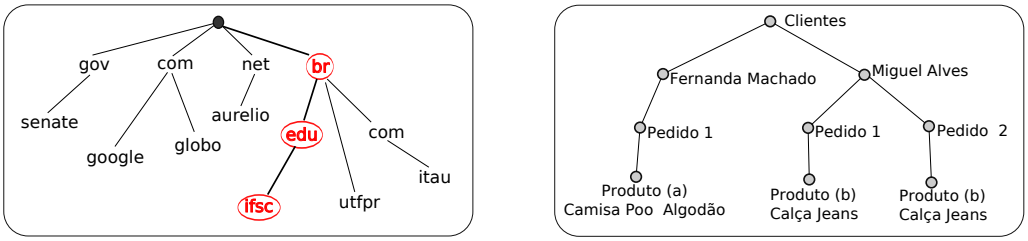
Listagem 2.1: Consulta SQL

A tarefa de relacionar os registros entre múltiplas tabelas, de acordo com os comandos SQL é realizado pelo (Sistema Gerenciador de Banco de Dados – SGBD). O SGBD é o software que

conta com uma série de mecanismos que permitem armazenar e recuperar as informações de forma organizada, além de garantir integridade no resultado das operações. Existem muitas soluções de SGBD, proprietárias e de código aberto. Dentre as proprietárias destacam-se o Oracle [1] e o DB2 da IBM [2]. De código aberto destacam-se o MySQL [3] e o PostgreSQL [4].

2.1.3 Diretórios hierárquicos

Nos diretórios hierárquicos, as informações são dispostas em uma árvore hierárquica, sendo que cada nó dessa árvore, representa um objeto, também chamado de entrada. As entradas podem representar qualquer coisa do mundo real. Uma empresa, uma pessoa, uma lista de endereços ou um arquivo de configuração, são exemplos práticos de entradas que compõem uma árvore de diretórios hierárquicos. O (*Domain Name System* – DNS) (ALBITZ; LIU, 2006) é um bom exemplo de organização em árvore hierárquica.



(a) Hierarquia do DNS. Ramos percorridos para resolver o nome de domínio `ifsc.edu.br`.

(b) Diretório hierárquico que armazena as vendas de uma empresa.

Figura 2.3: Disposição dos dados em árvore hierárquica.

A estrutura hierárquica, oferece agilidade no processo de pedido e resposta, pois ao realizar uma consulta, de acordo com a referência dada como parâmetro de busca, o diretório acessa a informação precisa na árvore. Por outro lado essa estrutura não é otimizada para gerir grandes volumes de atualização, portanto, os diretórios hierárquicos são aplicados em situações em que as informações são relativamente estáticas, ou seja, situações onde se exijam poucas operações de escrita, comparadas ao número de operações de leitura (ARKILLS, 2003).

A Figura 2.3(b) apresenta o mesmo exemplo apresentado na seção 2.1.2 para armazenar as vendas de uma empresa. Nos diretórios hierárquicos, não existe o conceito de relacionamento de registros. Enquanto que o modelo relacional permitia instanciar um produto em uma tabela, uma única vez e relacioná-lo em múltiplos pedidos, utilizando a chave estrangeira, o modelo

[1] `http://www.oracle.com/us/products/database`
[2] `http://www-01.ibm.com/software/data/db2/`
[3] `http://www.mysql.com/products/enterprise/server.html`
[4] `http://www.postgresql.org/`

hierárquico instancia o mesmo produto como um objeto único em cada pedido. Nesse caso, utilizar o modelo hierárquico para o propósito do exemplo, resultaria em um maior consumo de espaço físico para armazenar as informações, além de comprometer o seu desempenho, pois o fluxo de escrita para objetos vendas e produtos, seria constante.

A escolha de um modelo lógico de armazenamento adequado, depende de uma série de exigências que um determinado propósito impõe, para que as consultas e o gerenciamento das informações sejam eficientes e consistentes. Os diretórios hierárquicos ganharam destaque no propósito de concentrar informações de uma rede, como informações de usuários, informações sobre computadores, listas de email, arquivos de configuração, certificados digitais. Essas informações não são frequentemente alteradas, no entanto, são constantemente recuperadas por serviços e aplicações dentro de uma rede local.

Ao centralizar todas essas informações em uma única fonte de armazenamento, os administradores da rede passam a gerenciar e realizar a manutenção das informações de forma mais organizada. Atualmente é raro encontrar uma rede corporativa que não utilize uma solução de diretórios baseado no protocolo (*Lightweight Directory Access Protocol* – LDAP) (ZEILENGA, 2006b).

2.2 *Lightweight Directory Access Protocol* – LDAP

Em meados de 1980, a (*International Telegraph and Telephone Consultative Committee* – CCITT), atualmente chamada de ITU-T, com incentivo das grandes companhias telefônicas, estudava uma forma de criar um diretório universal para pesquisas de números telefônicos e endereços de correio eletrônico. Ao mesmo tempo, a (*International Organization for Standardization* – ISO) procurava uma solução de diretórios para servir nomes dentro de um ambiente de redes. A união de esforços entre esses dois órgãos resultou em 1988 em um conjunto de recomendações técnicas, denominadas como padrão X.500 de diretórios (HOWES; SMITH; GOOD, 2003).

O serviço de diretórios X.500 é baseado no modelo cliente/servidor. Clientes consultam as informações armazenadas no servidor através do protocolo de acesso (*Directory Access Protocol* – DAP) (ITU-T, 2005). Ao contrário das dezenas de operações complexas, permitidas por uma linguagem de acesso como o SQL, o DAP define um conjunto reduzindo de operações para consulta e gerenciamento das informações contidas em um diretório.

O modelo de diretórios X.500 desde o início apresentou dificuldades de operação, devido a complexividade de implementação do protocolo DAP. O DAP foi desenvolvido baseado no

modelo de referência OSI. Isso exigia que as máquinas tivessem uma carga significante de recursos computacionais, excluindo a possibilidade de estações clientes com menores recursos, utilizarem o serviço. Com o surgimento do modelo TCP/IP (SOCOLOFSKY; KALE, 1991) no final da década de 80, o serviço de diretórios passou a apresentar incompatibilidades de operação com a nova arquitetura de comunicação em rede. Diante das dificuldades de integrar o protocolo DAP em redes TCP/IP, pesquisadores da universidade de Michigan deram início no desenvolvimento de um protocolo compatível com a vigente arquitetura de redes, surgiu então o (*Lightweight Directory Access Protocol* – LDAP) (ZEILENGA, 2006b).

O protocolo leve de acesso a diretórios surgiu com a intenção de substituir o antigo e complexo protocolo DAP. O LDAP foi desenvolvido para ser utilizado nativamente sobre TCP/IP. Isso resultou em uma arquitetura de diretórios mais simples, onde permitiu que estações clientes com menor poder computacional, pudessem acessar o serviço de forma eficiente. Com isso, serviços de e-mail, compartilhamento de arquivos, autenticação, entre outros serviços de rede, passaram a implementar APIs [5] para acesso ao serviço de diretórios. A especificação do LDAP é descrita em quatro modelos básicos (HOWES; SMITH; GOOD, 2003).

2.2.1 Modelo de nomes

A árvore de diretórios é composta por entradas as quais possuem um nome absoluto, denominado (*Distinguished Name* – DN) ou nome distinto (ZEILENGA, 2006a). O nome distinto faz com que cada entrada seja representada de forma única e exclusiva no diretório, evitando ambiguidade de nomes. O conjunto das entradas formam a (*Directory Information Tree* – DIT). A Figura 2.4 ilustra um exemplo de DIT.

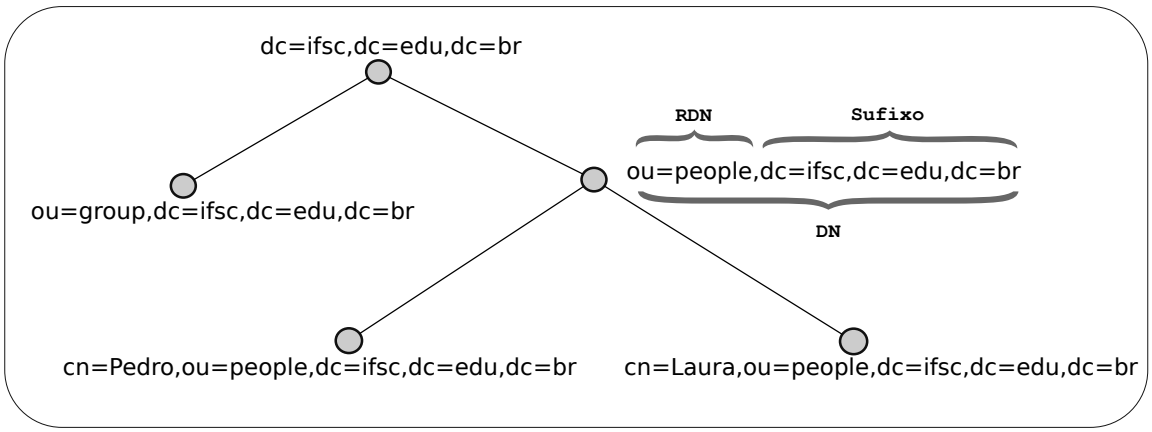


Figura 2.4: (*Directory Information Tree* – DIT) de um diretório.

[5] *Application Programming Interface* (API). É Um conjunto de funções e sub-rotinas usadas para ativar um determinado dispositivo no programa.

Na Figura 2.4 a entrada raiz do diretório é representada pelo DN (`dc=ifsc,dc=edu,dc=br`), que define o sufixo do diretório. A medida que as entradas são dispostas abaixo da raiz, o DN de cada entrada é composto pelo nome da entrada superior mais o (*Relative Distinguished Name* – RDN) ou nome distinto relativo, que representa o identificador único no nível hierárquico em que a entrada se encontra. Essa disposição é semelhante a hierarquia que compõe o (*Domain Name System* – DNS). A composição dos nomes das entradas, define o espaço de nomes do diretório.

2.2.2 Modelo de informação

A unidade básica de armazenamento no diretório é denominada de atributo. Uma entrada é composta por um conjunto de atributos, pertinentes a um objeto específico. Uma entrada, cujo propósito é armazenar informações de um indivíduo por exemplo, pode conter atributos como nome, sobrenome, endereço, telefone e e-mail A Figura 2.5 ilustra um entrada que representa a descrição de um indivíduo no diretório.

Figura 2.5: Atributos de uma entrada que representam um indivíduo.

Os atributos que uma entrada armazena são definidos pelas classes de objetos (*Object-Class*). As classes de objetos caracterizam a natureza da entrada no diretório. Existem classes de objetos para representar pessoas, computadores, organizações, configurações de serviços, enfim, para cada tipo de entrada armazenada no diretório, uma classe de objeto é designada com seus respectivos atributos.

As classes de objetos bem como os atributos utilizados no diretório, são definidos nos esquemas de dados (*schema*). Os esquemas de dados contêm as declarações dos atributos que o diretório utiliza, respeitando os padrões de sintaxe (LEGG, 2006). Através do esquema de dados, o diretório é capaz de interpretar que o atributo (*Common Name* – cn) armazena um

valor referente a um nome, assim como o atributo *telephoneNumber* armazena um número de telefone. Através da sintaxe dos atributos, as aplicações clientes quando acessam o diretório sabem exatamente quais informações desejam recuperar.

O modelo de informação do LDAP possui a capacidade de representar as entradas do diretório através de arquivos texto. Essa representação é denominada (*LDAP Data Interchange Format* – LDIF) (GOOD, 2000). Um arquivo LDIF, permite transportar o conteúdo do diretório entre diferentes servidores e mantê-los como uma forma de *backup*.

Os arquivos no formato LDIF também são utilizados nas tarefas de adicionar e modificar entradas no diretório. Os administradores do serviço utilizam o formato LDIF para declarar novas entradas e inseri-las através de comandos de inserção. A Listagem 2.2 mostra a representação LDIF para o diretório da Figura 2.5.

```
1  dn: dc=ifsc,dc=edu,dc=br
2  objectClass: dcObjetc
3  objectClass: organization
4  dc: ifsc
5  o: ifsc
6
7  dn: ou=people,dc=ifsc,dc=edu,dc=br
8  objectClass: organizationalUnit
9  ou: people
10
11 dn: cn=Laura,ou=people,dc=ifsc,dc=edu,dc=br
12 objectClass: inetOrgPerson
13 cn: Laura
14 sn: Santos
15 telephoneNumber: 04897865342
16 mail: laurasantos@ifsc.edu.br
17 homePostalAddress: Rua Albert Einstein, 1955
```

Listagem 2.2: Representação LDIF das entradas de um diretório.

Para cada entrada no diretório, é declarado seu DN, as classes de objetos que representam a entrada e seus respectivos atributos. Na Listagem 2.2, a representação LDIF contém a declaração de três entradas, cada uma com suas respectivas classes de objetos e atributos.

2.2.3 Modelo funcional

O modelo funcional do LDAP (SERMERSHEIM, 2006) define um conjunto de nove operações relacionadas a tarefas de conexão, consulta e atualização no diretório, além de algumas operações adicionais, denominadas operações estendidas. Essas operações são descritas a seguir:

- ***Bind***: É o comando enviado pelo cliente para abrir uma conexão com o servidor. A conexão pode ocorrer por meio de autenticação com uso de credenciais ou por meio de uma conexão anônima;
- ***Unbind***: Mensagem originada pelo cliente para fechar uma conexão com o servidor;
- ***Abandon***: Mensagem originada pelo cliente para abortar uma operação pendente;
- ***Search***: Operação para localizar entradas no diretório. Uma série de parâmetros podem ser utilizados para refinar a busca de uma informação como filtros de pesquisa, expressões regulares e parâmetros de escopo no diretório;
- ***Compare***: Operação que permite o cliente testar se um atributo possui um valor específico;
- ***Add***: Operação que permite adicionar entradas no diretório;
- ***Delete***: Operação que remove uma entrada no diretório;
- ***Modify***: Operação que permite modificar o valor de um atributo;
- ***Modifyrdn***: Modifica o nome distinto relativo (*Relative Distinguished Name* – RDN) de uma entrada. Ao modificar o RDN de uma entrada, todos nomes distintos (DN) das entradas abaixo da modificada devem ser alterados;
- ***Operações Estendidas***: São operações não definidas originalmente pelo protocolo LDAP, porém, permitem o cliente realizar pedidos e receber respostas a operações, cuja sintaxe e semântica são preestabelecidas. Alguns exemplos de operações estendidas são as mensagens entre cliente e servidor para iniciar uma conexão utilizando criptografia, a qual a operação *StartTLS* (HODGES; MORGAN; WAHL, 2000) é uma das mais utilizadas. O *StartTLS* permite a utilização de criptografia na comunicação utilizando o (*Transport Layer Security* – TLS) (DIERKS; ALLEN, 1999). Outra operação estendida é capacidade de alterar valores armazenados no atributo *userPassword*, referente ao armazenamento de senhas (ZEILENGA, 2001) pois a operação *modify* na sua forma básica não suporta esse tipo de alteração.

As operações listadas acima sempre são iniciadas pelo cliente LDAP com destino ao servidor de diretórios. O processo de conexão inicia-se com o cliente enviando uma mensagem de abertura de conexão *bind*. Na abertura de conexão o cliente pode vincular-se de forma anônima ou autenticada. O servidor responde a abertura de conexão com sucesso ou erro. Realizada

a conexão, o cliente solicita qualquer operação no diretório. Após receber os resultados da solicitação, o cliente fecha a conexão com a mensagem *unbind*. A Figura 2.6 ilustra a troca de mensagens entre cliente e servidor para uma operação de consulta (*search*).

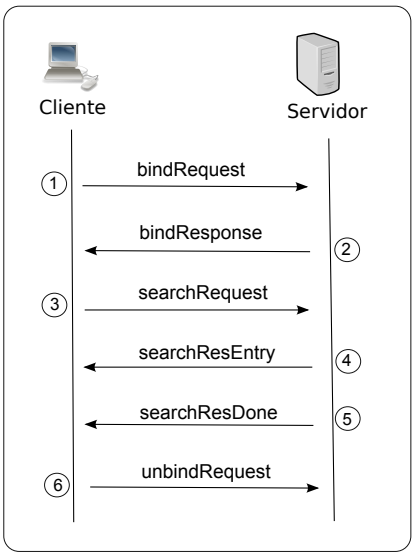


Figura 2.6: Diagrama de mensagens de conexão e consulta entre cliente e servidor.

2.2.4 Modelo de segurança

O modelo de segurança do LDAP define como os dados do diretório podem ser protegidos contra acessos indevidos. A segurança abrange duas etapas: o processo de autenticação de um cliente e a autorização de acesso a determinada informação no diretório. A autenticação ocorre durante a abertura de conexão. Existem quatro métodos de autenticação de um cliente em um servidor LDAP:

- Autenticação anônima: Na autenticação anônima o cliente não envia credenciais para autenticação. Qualquer solicitação anônima é respondida pelo servidor de acordo com a autorização de acesso. O método anônimo é interessante quando clientes solicitam um catálogo público de endereços por exemplo;
- Autenticação simples: A autenticação simples baseia-se no envio de um DN de usuário e uma senha. A desvantagem do método simples de autenticação é que a credencial trafega pela rede em texto claro;
- Autenticação utilizando criptografia: Um método seguro sobre a autenticação simples que evita o tráfego de senhas em texto claro. Nesse caso, utiliza-se o (*Secure Sockets Layer* – SSL) ou (*Transport Layer Security* – TLS) (HARRISON, 2006) nas conexões entre cliente e servidor.

- Autenticação utilizando módulos SASL: O (*Simple Authentication and Security Layer* – SASL) (MYERS, 1997) é um mecanismo externo de segurança que permite ao LDAP utilizar métodos de segurança adicionais, como por exemplo o Kerberos v5 (MELNIKOV, 2006).

A autorização de acesso ao diretório, se dá através das (*Access Control List* – ACL) ou listas de controle de acesso. As ACLs são regras que definem quais informações do diretório um usuário tem permissão de acesso, seja para leitura ou para escrita. O LDAP não define um padrão específico para composição de ACLs. Geralmente cada fornecedor oferece uma forma de implementação particular.

2.2.5 Diretórios distribuídos

O serviço de diretórios LDAP, possui a capacidade de distribuição das informações entre diferentes locais. A distribuição de um diretório torna-se necessária quando as informações precisam ser acessadas em diferentes locais, por exemplo: uma empresa composta pela matriz e uma filial separadas geograficamente, utilizam um serviço de diretórios para armazenar informações de usuários e serviços. Supondo que apenas um servidor de diretórios, localizado na matriz da empresa detêm essas informações, então a filial precisaria consultar remotamente as informações na matriz, utilizando a internet como meio de acesso. Nesse caso, o desempenho do diretório em responder com agilidade as solicitações estaria comprometido.

Através da distribuição do diretório as informações poderiam estar dispostas localmente na filial, eliminando a necessidade de utilizar a internet para consultar o diretório. O acesso local ao diretório aumentaria o desempenho do serviço.

Há duas maneiras básicas de distribuir um diretório em diferentes locais. Por meio do particionamento ou através da replicação. O particionamento consiste em delegar parcelas da estrutura do diretório entre diferentes servidores. Utilizando o exemplo da empresa, a matriz poderia armazenar a parcela raiz do diretório, bem como todas as entradas destinadas apenas a matriz, enquanto a filial poderia armazenar o ramo com as entradas referentes a sua própria unidade, conforme mostra a Figura 2.7.

A união das partições, forma uma única visão lógica do diretório. Se um cliente LDAP localizado na matriz, realizasse uma consulta a alguma informação na partição de sua unidade, a consulta seria local. Entretanto, se a consulta fosse destinada a uma informação localizada na partição da filial, o servidor local encaminharia essa consulta para o servidor de diretórios remoto.

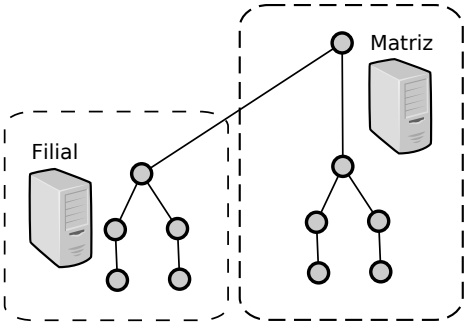


Figura 2.7: Particionamento do diretório.

Outra forma de distribuir o diretório é através da replicação. Diferentemente do particionamento, a replicação permite manter cópias de um mesmo diretório em inúmeros locais distintos. As vantagens de manter uma cópia do diretório em mais de um local, implicam desde a disponibilidade local das informações até a possibilidade de manter um backup do diretório.

No exemplo da empresa, o uso da replicação manteria todas as informações tanto na matriz como na filial eliminando a necessidade de realizar qualquer consulta remota, como ilustra a Figura 2.8

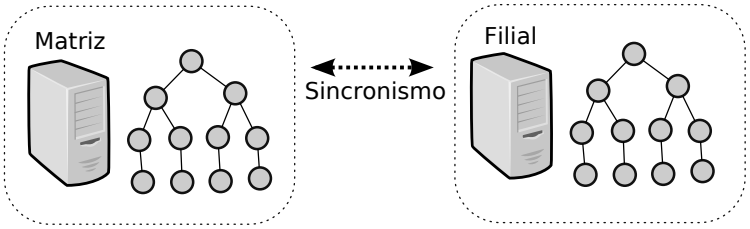


Figura 2.8: Replicação do diretório.

2.3 *OpenLDAP*

O *OpenLDAP* [6] é um *software* de código aberto que oferece um serviço de diretórios. Seu desenvolvimento ocorreu paralelamente ao desenvolvimento e padronização do protocolo LDAP. Atualmente, o *OpenLDAP* é mantido por uma comunidade de código livre, que conta com a participação de diversos desenvolvedores interessados em aprimorar o funcionamento do *software*. Entre as principais características do *OpenLDAP* destacam-se (OPENLDAP, 2010):

- Suporte a versão 3 do protocolo;
- Suporte a mecanismos de autenticação com (*Simple Authentication and Security Layer* – SASL);

[6] `http://openldap.org`

- Suporte a criptografia (*Secure Sockets Layer* – SSL) e (*Transport Layer Security* – TLS);
- Controle de acesso através de (*Access Control List* – ACL);
- Suporte a diversos tipos de *backends* [7];
- Suporte ao particionamento do diretório;
- Suporte a replicação do diretório.

2.3.1 Controle de acesso no *OpenLDAP*

As listas de controle de acesso (*Access Control List* – ACL) permitem gerenciar a autorização de acesso às entradas no diretório. As ACLs são declaradas e lidas sequencialmente, ou seja, regras mais restritivas devem ser declaradas primeiro. A sintaxe básica de declaração possui o seguinte formato:

access to **[o quê]** *by* **[quem] [tipo de acesso] [controle]**

- **o quê**: pode ser uma entrada, um atributo ou um conjunto deles;
- **quem**: uma entrada específica que pode representar uma identidade de usuário ou um grupo de usuários;
- **tipo de acesso**: define se a entrada tem permissões de leitura, escrita ou nenhum acesso;
- **controle**: Um parâmetro opcional que controla o fluxo de processamento das ACLs. Por padrão se uma regra atender, as demais não são processadas.

Utilizando como exemplo a entrada `cn=Laura,ou=people,dc=ifsc,dc=edu,dc=br`, ilustrada na Figura 2.5, uma regra nas ACLs para controlar o acesso aos valores dessa entrada teria a seguinte sintaxe:

access to dn.exact=“ `cn=Laura,ou=people,dc=ifsc,dc=edu,dc=br`***”***
by dn.exact=“`cn=adminstrador,dc=ifsc,dc=edu,dc=br`***” write***
by self read
by * none

[7] O *backend* do diretório refere-se ao método de armazenamento físico das informações, que podem variar desde o armazenamento em arquivo texto até métodos mais abstratos otimizados para essa finalidade.

A regra aplica três cláusulas à entrada do usuário. A primeira, representada pela sintaxe "*dn.exact*", indica que o usuário administrador tem a permissão de modificar (*write*) a entrada em questão. A segunda cláusula, representada pela sintaxe "*self*", indica que o próprio usuário pode ler (*read*) os atributos de sua entrada. A última cláusula, representada pelo caractere asterisco, restringe qualquer acesso (*none*) a qualquer outra identidade ou grupo de usuário.

A sintaxe de declaração das ACLs é bem flexível, pode-se usar expressões regulares, parâmetros avançados para determinar escopos da árvore a qual se deseja aplicar uma regra, enfim, em (BUTCHER, 2007) pode-se encontrar uma referência ampla de como aplicar ACLs no diretório.

2.4 (Comunidade Acadêmica Federada – CAFe)

Uma federação acadêmica tem como objetivo, estabelecer um ambiente de compartilhamento de recursos e serviços entre instituições de ensino. Um usuário vinculado a uma instituição, através da relação de confiança entre as instituições, possui a capacidade de usufruir de serviços oferecidos por outras instituições. Serviços como acesso a bibliotecas digitais, cursos à distância ou até acesso a recursos computacionais compartilhados (CPU, armazenamento) para projetos de pesquisa (MOREIRA et al., 2010).

Muitos países já adotam um ambiente de instituições federadas. Nos EUA por exemplo, a federação *INCommon*[8] é composta por aproximadamente 107 instituições que integradas compartilham recursos e serviços para mais de dois milhões de usuários. No Brasil, sob coordenação da (Rede Nacional de Pesquisa – RNP) e com participação de um grupo de instituições de ensino superior, em 2007 deu início ao projeto de infraestrutura de autenticação e autorização, denominado (Comunidade Acadêmica Federada – CAFe) [9]. A federação CAFe é o projeto brasileiro para oferecer um ambiente de intercâmbio de serviços entre instituições de ensino brasileiras.

A infraestrutura da federação CAFe é composta por dois elementos básicos: provedores de serviço e provedores de identidade. Um provedor de serviço, caracteriza-se por oferecer qualquer recurso aos usuários da federação. Um provedor de identidade por outro lado, caracteriza-se por armazenar informações de usuários que são recuperadas pelos provedores de serviços para autorizar o acesso aos recursos por eles oferecidos.

O acesso aos serviços pelos usuários, ocorre por meio da identidade única, ou seja, no

[8] `http://www.incommonfederation.org/`
[9] `http://www.cafe.rnp.br`

provedor de identidade da instituição, é armazenada a identidade de usuário, válida em todo âmbito da federação. Para isso, no provedor de identidades são exigidos alguns requisitos técnicos como a implementação de um serviço de diretórios, adequado com um esquema de dados comum. Na federação CAFe, o esquema de dados utilizado é o *brEduPerson* [10]. O *brEduPerson* permite ao serviço de diretórios armazenar informações pessoais como nome, sobrenome, data de nascimento, CPF, número de passaporte e informações referentes aos vínculos que um usuário possui com sua instituição, como o tipo do vínculo firmado, data de início e encerramento do vínculo.

2.5 Conclusões

Esse capítulo apresentou uma breve análise de algumas formas de armazenar informações bem como suas características, padrões e vantagens de implementação dentro de um propósito específico. Essa análise serviu principalmente para diferenciar o modelo relacional do modelo de diretórios hierárquicos, justificando porque um serviço de diretórios é mais adequado para centralizar informações de usuários de uma rede.

O LDAP recebeu uma abordagem objetiva com todos os aspectos técnicos e conceituais necessários para a sua compreensão. Os modelos que descrevem o serviço foram apresentados como na maioria das literaturas sobre o assunto, porém, essa abordagem também sugere os aspectos que devem ser levados em consideração no planejamento de um serviço de diretórios.

No planejamento de um serviço de diretórios, deve-se analisar como o espaço de nomes pode ser elaborado de forma que as consultas percorram o menor número de ramos possíveis, ao mesmo tempo que a árvore seja modelada a fim de organizar entradas de natureza comum. No modelo de informação, deve-se levar em consideração qual esquema de dados o serviço precisa adotar para que sejam armazenados todos atributos que as aplicações clientes necessitam recuperar no diretório. E quanto a segurança, quais mecanismos são necessários para garantir confidencialidade no tráfego das informações e como pode ser gerenciado o acesso ao diretório para cada identidade.

A federação CAFe no aspecto técnico de seu funcionamento está acima do escopo desse trabalho, entretanto, a filosofia, os benefícios e o requisitos básicos referentes a um provedor de identidade, foram apresentados a fim de justificar a inclusão do esquema *brEduPerson*, na implementação de um serviço de diretórios aplicado a instituições acadêmicas.

[10] `http://www.cafe.rnp.br/cafewiki/images/breduperson-20080917-0.0.6.schema`

3 *Modelos de diretórios distribuídos para instituições acadêmicas.*

Esse capítulo apresenta um estudo sobre os modelos existentes para implementação de um serviço de diretórios distribuídos, aplicado a instituições acadêmicas multi campi. O principal objetivo desse ambiente de diretórios é permitir que informações oriundas de um determinado campus, sejam compartilhadas entre os demais campi da instituição. A abordagem do estudo dá ênfase as técnicas de particionamento e replicação do diretório, técnicas disponíveis com o uso do *OpenLDAP*.

O estudo ainda leva em consideração, a capacidade do serviço de diretórios adequar-se como uma base de dados, responsável por armazenar identidades de usuários, de acordo com os requisitos técnicos exigidos pela federação CAFe.

A abordagem adotada nesse capítulo, divide o planejamento do serviço de diretórios em tópicos distintos. Cada tópico analisa sob um aspecto particular, as etapas necessárias para o planejamento completo de um serviço de diretórios. Os seguintes aspectos são abordados:

- Definição de um esquema de dados genérico, compatível com a realidade de uma instituição acadêmica;
- Composição do espaço de nomes e a organização hierárquica do diretório;
- Modelos existentes para implementação de um serviço de diretórios distribuídos;
- Recursos de segurança disponíveis para autenticação, autorização e confidencialidade no tráfego das informações;
- Conclusões gerais do capítulo e uma síntese de qual modelo é mais adequado para o propósito desse trabalho.

Os modelos apresentados no decorrer do capítulo baseiam-se na viabilidade oferecida pelo

OpenLDAP, solução de diretórios escolhida como objeto de estudo e utilizada nos experimentos práticos do Capítulo 4.

3.1 Escolha do esquema de dados

O esquema de dados do diretório está diretamente relacionado com os serviços que utilizam esse diretório como meio de armazenamento. Na maioria das situações cotidianas, um diretório desempenha principalmente a função de armazenar informações sobre indivíduos, como nome, sobrenome, telefone, email e endereço, por exemplo.

O esquema de dados mais comum para representar indivíduos no diretório, é composto pela classe de objeto *inetOrgPerson* [1]. A classe de objeto *inetOrgPerson* fornece uma grande quantidade de atributos para essa finalidade. A Tabela 3.1 lista os principais atributos da classse de objeto *inetOrgPerson*.

Atributos	Descrição
cn	Nome
sn	Sobrenome
homePostalAddress	Endereço
telephoneNumber	Telefone
mail	Endereço de email

Tabela 3.1: Principais atributos da classe de objeto *inetOrgPerson*.

A entrada que representa um indivíduo no diretório, pode conter atributos que o identificam como usuário de um determinado serviço ou aplicação dentro do contexto de uma rede. Normalmente, o principal serviço que utiliza o serviço de diretórios para recuperar informações de usuários, refere-se a autenticação em estações dentro do domínio de rede.

Se a rede local é composta por estações com o sistema operacional *Linux*, a classe de objeto *posixAccount* [2] em conjunto com a classe de objeto *shadowAccount* [3] são utilizadas. A classe *posixAccount* fornece os atributos básicos de uma identidade de usuário no *Linux*. Para representação de grupos de usuários no diretório, a classe de objeto *posixGroup*[4] fornece os atributos para essa finalidade. Através dessas três classes de objetos, é possível utilizar um diretório LDAP para autenticar usuários em uma rede composta por estações *Linux*. A Tabela 3.2 resume os principais atributos exigidos na autenticação do usuário.

[1] `http://ldap.akbkhome.com/index.php/objectclass/inetOrgPerson.html`
[2] `http://ldap.akbkhome.com/index.php/objectclass/posixAccount.html`
[3] `http://ldap.akbkhome.com/index.php/objectclass/shadowAccount.html`
[4] `http://ldap.akbkhome.com/index.php/objectclass/posixGroup.html`

Atributos	Descrição
uid	ID do usuário
userPassword	senha
cn	Nome completo
gecos	Nome apresentado no sistema
uidNumber	número do ID de usuário
gidNumber	número do ID do grupo principal do usuário
homeDirectory	local do diretório pessoal
loginShell	shell de sessão

Tabela 3.2: Atributos de um usuário *Linux*, fornecidos pela classe de objeto *posixAccount*.

Em redes heterogêneas (compostas por estações *Linux* e *Windows*), através de um servidor de arquivos Samba[5] por exemplo, o serviço de diretórios precisa adotar um esquema de dados específico para armazenar as propriedades de usuários, grupos e máquinas requeridas pelo serviço. Nesse caso, a classe de objeto *sambaAccount*[6] fornece os atributos exigidos. Neste trabalho, o modelo de informação genérico adotado considera que o serviço de diretórios atende apenas a autenticação de usuários em sistemas *Linux*.

Para que uma instituição faça parte da federação CAFe, é exigido que a mesma, implemente um serviço de diretórios contendo um modelo de informação específico. O modelo de informação adotado é composto pelo esquema de dados *brEduPerson*[7]. A Tabela 3.3 lista os principais atributos do esquema *brEduPerson*.

Atributos	Descrição
brcpf	Número de CPF
brpassport	Número de passaporte
cn	Nome
sn	Sobrenome
schacDateOfBirth	Data de nascimento
schacGender	Sexo
schacCountryOfCitizenship	Nacionalidade
homePostalAddress	Endereço completo
telephoneNumber	Número de Telefone
mail	email

Tabela 3.3: Esquema *brEduPerson*. Atributos gerais de um indivíduo.

O esquema *brEduPerson* oferece também, atributos para indicar os vínculos que uma pessoa possui com sua instituição, por exemplo: estudante, professor, colaborador. Esses atributos são de suma importância para a infraestrutura de autenticação e autorização na federação, afinal,

[5] `http://www.samba.org`
[6] `http://www.cs.dixie.edu/ldap/server/ldap/samba.schema.txt`
[7] `http://www.cafe.rnp.br/cafewiki/images/breduperson-20080917-0.0.6.schema`

baseada nelas, os provedores de serviços determinam se autorizam ou não, o acesso ao usuário a um determinado serviço. A Tabela 3.4 lista os atributos referentes ao vínculo do usuário.

Atributos	Descrição
braff	ID do vínculo
brafftype	Tipo de vínculo
brentr	Data que o vínculo inicia
brexit	Data que o vínculo é finalizado

Tabela 3.4: Esquema *brEduPerson*. Atributos que armazenam o vínculo do usuário com a instituição.

Dentro da hierarquia do diretório, as informações de cada vínculo de usuário são dispostas abaixo da entrada principal do usuário. Essa disposição permite que um usuário possua múltiplos vínculos distintos. A Figura 3.1 ilustra essa disposição.

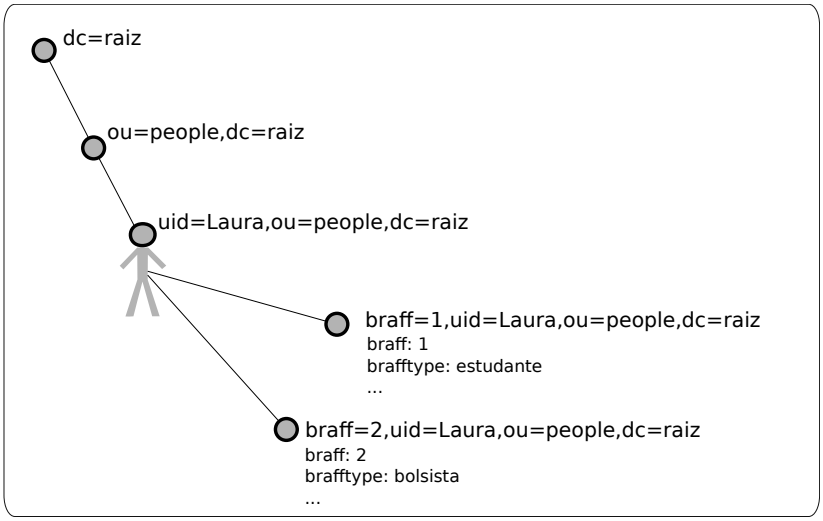


Figura 3.1: Disposição das informações de vínculo do usuário dentro do diretório.

O esquema de dados *brEduPerson* ainda disponibiliza atributos para representar números de telefones *VoIP* e dados biométricos de usuários. Através dos atributos referentes a um número *VoIP*, pode-se armazenar essas informações para dispô-las em um ambiente de telefonia IP, como por exemplo o projeto *FONE@RNP* [8]. Atributos como número do telefone, duração total das conversações e créditos de uso foram criados para atender esse serviço. E para um suposto serviço de identificação de usuários por dados biométricos, o esquema de dados disponibiliza atributos adequados para armazenar esse tipo de informação.

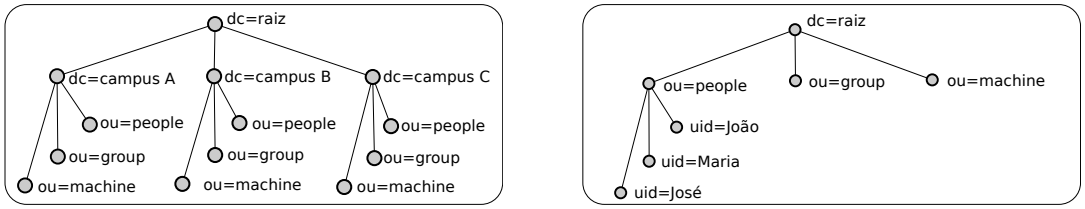
[8] `http://www.rnp.br/voip/`

3.2 O espaço de nomes do diretório

O espaço de nomes define a disposição das entradas no diretório. A composição do (*Distinguished Name* – DN) de cada entrada é definida a medida que a hierarquia da árvore do diretório é formada. Um espaço de nomes bem planejado, deve ser organizado de forma mais simples possível para facilitar a sua compreensão e para maximizar o desempenho do serviço.

Normalmente, entradas de mesma natureza são armazenadas abaixo de um ramo comum. Se o propósito do diretório é armazenar identidades de usuários, essas entradas devem ser alocadas sob um ramo específico, enquanto entradas referentes a grupos de usuários são alocadas em outro ramo específico e assim por diante.

Tratando-se de um diretório aplicado a uma instituição multi campi, uma ideia de disposição das entradas no diretório é apresentada pelas Figuras 3.2(a) e 3.2(b).



(a) Disposição das entradas baseada na divisão por campi.

(b) Disposição das entradas de acordo com a sua natureza.

Figura 3.2: Exemplos de disposição das entradas no diretório.

A Figura 3.2(a) sugere a composição hierárquica do diretório, baseando-se na divisão de informações por campi. Essa divisão é interessante para delegar administrativamente parcela do diretório para cada campus, onde os administradores da rede teriam privilégios sobre as entradas abaixo do ramo de seu respectivo campus. Entretanto, a composição do espaço de nomes da Figura 3.2(a) apresenta algumas desvantagens. A maior delas refere-se a possibilidade de haver duas ou mais entradas de usuários com a mesma identificação, como mostra a Figura 3.3.

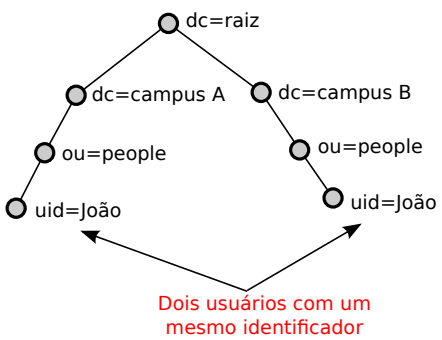


Figura 3.3: Duplicidade de nomes no diretório.

Na Figura 3.3, duas entradas tem o mesmo valor para o atributo (*uid*). O atributo *uid* representa o nome utilizado no *logon* do usuário na rede. Do ponto de vista do diretório, essas entradas são distintas, no entanto, para o serviço responsável pela autenticação, ao realizar uma consulta no diretório em busca de um atributo com o nome "*João*", o diretório retornará dois valores, resultando em conflito de identidades.

A identificação de usuário deve ser única dentro de um contexto de rede. Se o espaço de nomes do diretório não garante essa premissa, a única forma de evitar entradas de usuários duplicadas seria por intervenção do administrador, através de uma política para atribuição de nomes. Uma prática inviável.

Diante disso, o diretório deve ser planejado para garantir unicidade nos nomes de usuários. A Figura 3.2(b) ilustra uma disposição adequada para garantir unicidade de nomes. Todas entradas de mesma natureza são alocadas abaixo de um ramo específico, logo, o próprio espaço de nomes do diretório consegue através da composição do DN do usuário, garantir que não exista nomes duplicados. Dentre as duas formas apresentadas, a disposição da Figura 3.2(b) é a mais simplificada e portanto, adotada nesse trabalho.

3.3 Modelos de diretórios distribuídos

Essa seção apresenta os métodos existentes para distribuição do diretório. No contexto de uma instituição acadêmica multi campi, a distribuição caracteriza-se por disponibilizar as informações do diretório entre servidores locais, em cada um dos campi da instituição. A seguir são apresentados os modelos de implementação de um serviço de diretórios distribuídos, utilizando os recursos existentes no *OpenLDAP*.

3.3.1 Modelo baseado em particionamento

O *OpenLDAP* oferece uma forma de distribuir a árvore do diretório em partições independentes. Cada partição é armazenada e controlada em um servidor de diretórios distinto. Esses servidores respondem pelas informações de sua partição. Para o acesso às outras partições, o serviço é configurado para indicar referências (*referrals*) . A Figura 3.4 ilustra um espaço de nomes particionado.

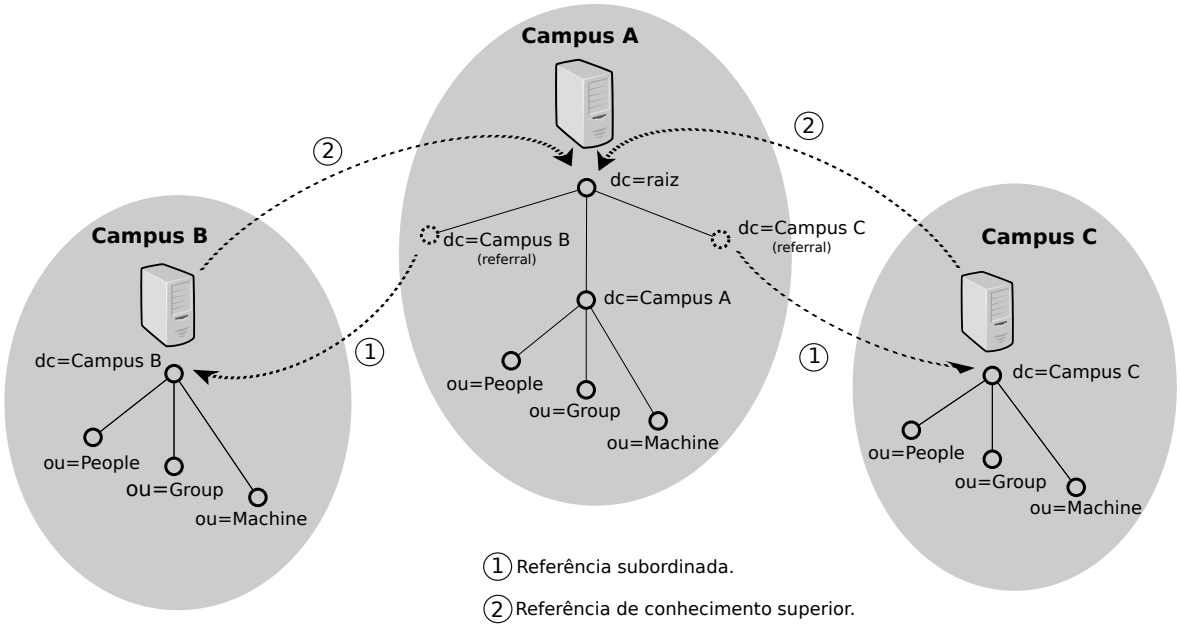


Figura 3.4: Particionamento do diretório acadêmico.

O diretório da Figura 3.4 é particionado em três ramos. As informações de cada partição são armazenadas nos servidores dos respectivos campi. A visão lógica total do diretório é realizada por meio da utilização das referências. As referências indicam os locais onde as demais partições estão armazenadas. As referências podem ser de dois tipos:

- **Referência subordinada:** A referência subordinada (rótulo 1 da Figura 3.4) é representada na forma de uma entrada, composta por um atributo, cujo valor é o endereço no formato URL (HOWES; SMITH, 1997) de uma partição subordinada a estrutura do diretório.

- **Referência de conhecimento superior:** A referência de conhecimento superior (rótulo 2 da Figura 3.4) é declarada na configurarão do serviço como uma diretiva que indica o endereço URL do servidor que detêm a partição hierarquicamente superior da estrutura do diretório.

Quando o cliente realiza uma consulta a uma determinada partição, se a entrada solicitada possuir um (*Distinguished Name* – DN) pertencente ao espaço de nomes da partição em questão, a consulta é realizada localmente. Por outro lado, se o serviço identificar que a entrada solicitada não corresponde ao seu espaço de nomes, ele é capaz de devolver ao cliente uma referência (*referral*), indicando o endereço de uma partição remota. Um exemplo de como ocorre uma consulta envolvendo referências, é ilustrada na Figura 3.5.

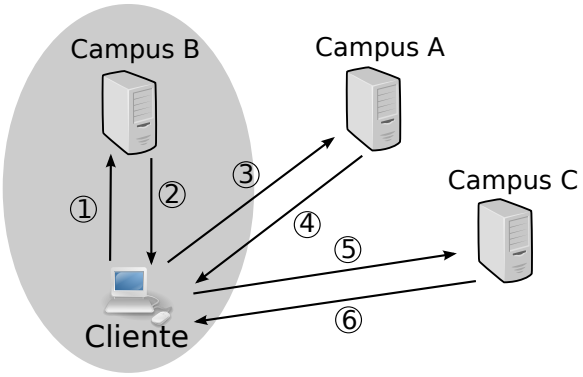


Figura 3.5: Troca de mensagens por busca interativa.

A Figura 3.5 ilustra as trocas de mensagens de um cliente localizado no Campus B, que solicita informações de um usuário, cuja identidade está armazenada na partição do Campus C. As etapas desse processo são descritas a seguir:

1. O cliente realiza uma busca ao diretório local;
2. O servidor local processa o pedido e verifica que a entrada solicitada está fora do seu escopo. Baseado nisso, o servidor retorna uma mensagem indicando onde talvez a entrada possa ser encontrada. Essa mensagem é composta por uma referência de conhecimento superior e seu formato é o endereço do servidor que abriga a partição do diretório imediatamente superior ao seu escopo, no caso, a URL do servidor do Campus A;
3. O cliente ao receber a referência, realiza uma nova consulta. Dessa vez com destino ao endereço do servidor localizado no Campus A;
4. No campus A, o servidor processa o pedido e verifica que apesar de não conter a entrada solicitada, reconhece que ela pertence a uma partição remota componente do diretório. Dessa forma, devolve ao cliente uma mensagem contendo uma referência subordinada, indicando exatamente qual servidor atende pelo nome da entrada solicitada;
5. O cliente realiza uma nova consulta, dessa vez ao servidor do campus C que contém a entrada solicitada;
6. O servidor do campus C ao receber o pedido, responde a solicitação de acordo com os parâmetros e filtros utilizados pelo cliente.

No exemplo ilustrado pela Figura 3.5, pode-se diferenciar os dois tipos de referências que compõem o particionamento do diretório. No passo 2, a referência de conhecimento superior é

devolvida ao cliente, em qualquer situação onde o DN da entrada solicitada, não é conhecido pelo servidor.

As referências de conhecimento superior podem ser comparadas ao comportamento de um *Default Gateway* na rede. Se o endereço (a entrada) não corresponde a rede local (a partição), o destino é entregue ao próximo roteador (o diretório hierarquicamente superior).

No Passo 4, o servidor do Campus A analisa a entrada solicitada e baseado no DN da entrada, responde com uma referência subordinada, indicando o endereço do servidor C, no qual está armazenada a conta do usuário. Uma referência subordinada é entregue ao cliente, apenas se a entrada solicitada possuir um sufixo conhecido pelo servidor.

O processo realizado pelo cliente envolveu três operações de consulta, cada uma em um servidor diferente. Esse processo que envolve o cliente por conta própria, realizar novas consultas a partir de uma referência recebida é denominado como consulta interativa (Figura 3.5). A responsabilidade de seguir referências fica por conta do cliente. Entretanto, muitas vezes as aplicações clientes não possuem a capacidade de processar referências e quando recebidas o cliente não interpreta esse tipo de mensagem e sua operação se dá como perdida. Diante disso, é possível implementar um mecanismo capaz de fazer com que o servidor de diretórios realize as consultas em nome dos clientes que as solicitam. Esse processo é denominado como busca recursiva e seu funcionamento é ilustrado na Figura 3.6.

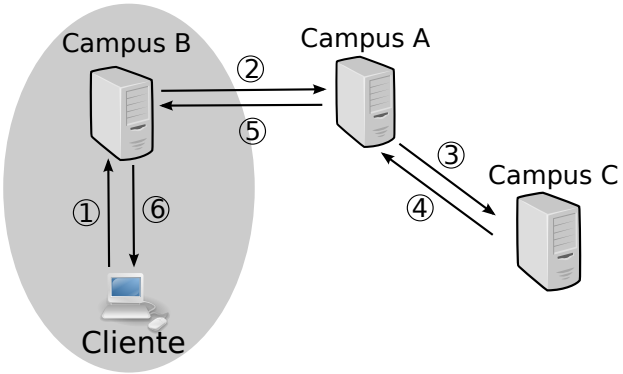


Figura 3.6: Troca de mensagens por busca recursiva.

Em uma operação por busca recursiva, o cliente faz a consulta e recebe o resultado sem ter conhecimento das etapas envolvidas para resolver as referências. A resposta ao pedido, do ponto de vista do cliente, ocorre de forma transparente. No *OpenLDAP* o processo recursivo é realizado utilizando o módulo *chain* [9].

Uma das desvantagens do modelo baseado em particionamento, é a sua composição do espaço de nomes. De acordo com a Figura 3.4, para permitir que o diretório fosse particionado,

[9] `http://www.openldap.org/software/man.cgi?query=slapo-chain`

foi necessário compor o espaço de nomes do diretório, baseando-se na divisão por campi. Essa divisão deve ser evitada, justamente por não fornecer em sua estrutura hierárquica, a unicidade de nomes de usuários (valores únicos para o atributo "`uid`").

3.3.2 Modelo baseado em replicação *Single Master*

A replicação diferentemente do particionamento, consiste em manter cópias de uma mesma árvore de diretórios em locais distintos. A replicação é capaz de resolver os dois principais pontos críticos existentes no modelo baseado em particionamento: a incapacidade do diretório prevenir que duas ou mais entradas tenham o mesmo valor para o atributo *uid* e a necessidade de realizar consultas remotas para recuperar informações de usuários pertinentes a outros campi.

A árvore do diretório é disposta por completo em cada servidor local e o espaço de nomes é modelado, para que todas entradas de mesma natureza sejam dispostas abaixo de um ramo específico. A Figura 3.7 ilustra a disposição do diretório no modelo baseado em replicação.

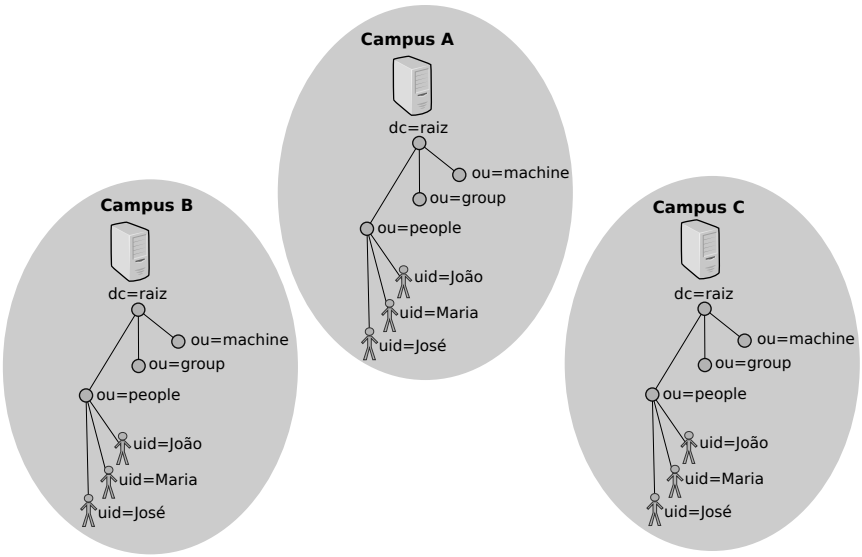


Figura 3.7: Espaço de nomes no modelo baseado em replicação.

De acordo com a Figura 3.7, uma cópia completa do diretório é mantida em cada servidor e portanto todas as consultas são realizadas localmente. Todas as entradas de usuários são dispostas abaixo do ramo `ou=people`, evitando que haja entradas duplicadas (com o mesmo valor para o atributo `uid`).

O *OpenLDAP* fornece dois modelos de replicação do diretório, o modelo de replicação *Single Master* e o modelo *Multi Master*. A diferença básica entre os dois modelos, refere-se a capacidade dos servidores envolvidos receberem operações de leitura e escrita. Um servidor que recebe tanto operações de escrita como operações de leitura é denominado como provedor.

Já um servidor que recebe apenas operações de leitura é denominado como consumidor.

O modelo *Single Master* fornece um ambiente de replicação composto por um servidor provedor e múltiplos consumidores. O modelo *Multi Master* caracteriza-se pela capacidade de todos servidores envolvidos na replicação, atuarem como provedores e consumidores entre si. Essa sessão detalha o modelo de replicação *Single Master* fornecido pelo *OpenLDAP*. Na sessão 3.3.3 é apresentado o modelo de replicação *Multi Master*.

No modelo baseado em replicação *Single Master*, o provedor é o principal servidor de diretórios. Através dele, as entradas são inseridas, modificadas ou removidas. As alterações no diretório são repassadas para todos os consumidores que dependem do provedor para manterem seus diretórios locais atualizados. A Figura 3.8 ilustra essa característica.

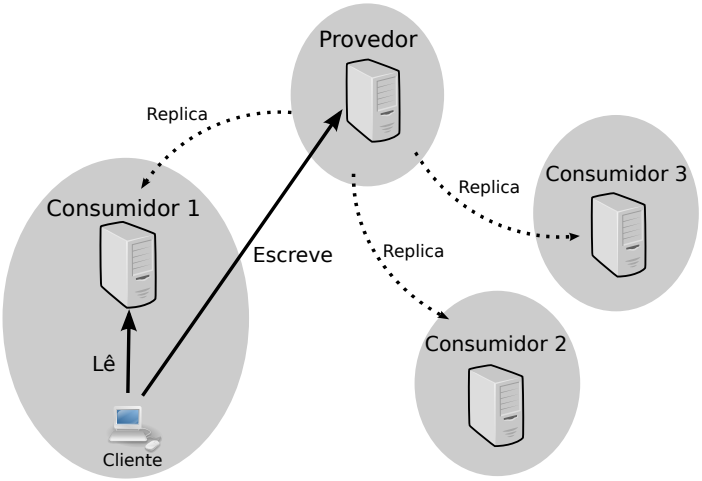


Figura 3.8: Replicação *Single Master*.

O processo de replicação ocorre através do protocolo *syncrepl* (CHOI; ZEILENGA, 2006). O princípio de funcionamento do *syncrepl* consiste nos consumidores manterem suas cópias do diretório, baseadas no diretório do provedor. Cada consumidor é configurado para consultar o estado do diretório provedor de acordo com um ciclo de replicação pré-definido. Para isso, o consumidor troca *cookies* de sincronismo com o provedor. Esses *cookies* de sincronismo contém um número identificador denominado (*Change Sequence Number* – CSN) utilizado para o controle de estado de atualização entre provedor e consumidor.

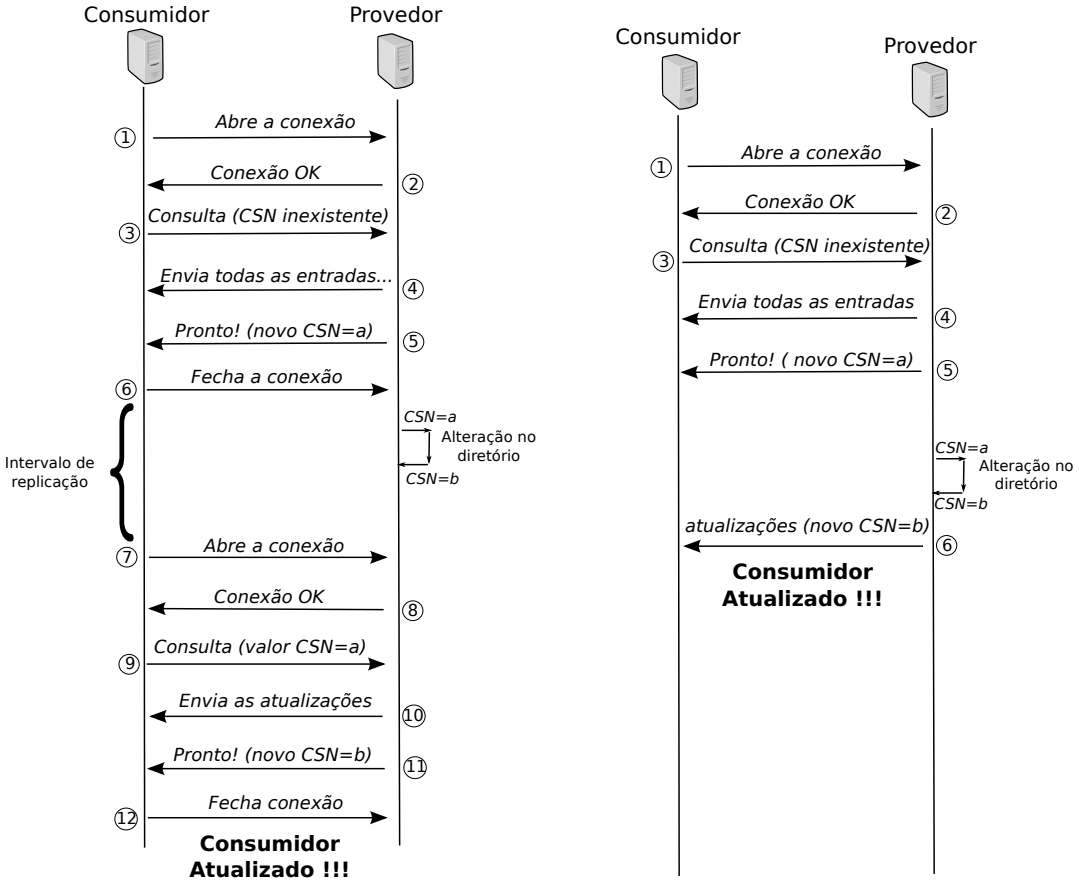
O valor do (*Change Sequence Number* – CSN) é armazenado em um atributo operacional, chamado de *contextCSN*, na entrada raiz do diretório. Além do atributo *contextCSN*, o protocolo cria em cada entrada do diretório outros dois atributos utilizados para o controle do mecanismo de replicação, o atributo *entryUUID* que serve como um identificador único da entrada e o atributo *entryCSN* que mantém um índice de atualização particular para cada entrada.

O intervalo de replicação entre um consumidor e o provedor é definido pelo ciclo de

replicação. O *syncrepl* oferece dois modos de operação quanto ao ciclo de replicação, o modo de operação *PULL* e o modo *PUSH*. No modo de operação *PULL*, o consumidor tem a responsabilidade de periodicamente realizar uma consulta ao diretório provedor para buscar as atualizações. O intervalo entre cada consulta é definido em dias, horas, minutos ou em segundos. A cada ciclo de replicação, o consumidor abre uma conexão com o provedor, realiza a consulta e em seguida fecha a conexão.

No segundo modo de operação, denominado *PUSH*, também chamado de modo persistente, o consumidor abre a conexão com o provedor, realiza uma consulta em busca de atualizações e em seguida mantém a conexão aberta permanentemente. Se houver alguma alteração no diretório provedor, o próprio provedor se encarrega de enviar imediatamente as atualizações para o consumidor. Nesse caso o sincronismo entre provedor e consumidor ocorre em tempo real.

Partindo do princípio que um consumidor não realizou ainda nenhuma operação de sincronismo e portanto não contém nenhuma informação, a Figura 3.9(a) ilustra o processo básico de replicação para o modo *PULL* e a Figura 3.9(b) ilustra o processo para o modo *PUSH*.



(a) Modo de operação *PULL*.

(b) Modo de operação *PUSH*.

Figura 3.9: Troca de mensagens na replicação *Single Master*.

Nos dois modos de operação, o consumidor inicia o processo de replicação através de uma conexão com o provedor. Essa conexão é autenticada e a identidade de acesso deve possuir autorização de leitura para todas as entradas que serão replicadas.

Estabelecida a conexão, o consumidor realiza uma operação de consulta estendida ao provedor. Uma operação de consulta estendida consiste em enviar como parâmetros da consulta, o sufixo do diretório, o escopo da consulta e um filtro baseado em classes de objetos. Em anexo a mensagem da consulta, o consumidor envia o (*Change Sequence Number* – CSN) que indica o estado atual de sua réplica. No passo 3 dos diagramas das Figuras 3.9(a) e 3.9(b), os valores do CSN não são enviados pois eles ainda não existem no consumidor.

O provedor ao receber a solicitação, verifica que através do CSN inexistente, o consumidor encontra-se com o diretório vazio, portanto envia todas as entradas solicitadas de acordo com os parâmetros de busca passados pelo consumidor. Ao final do processo de replicação, o provedor envia uma mensagem com um novo valor CSN, indicando o estado de atualização do diretório.

O primeiro ciclo de replicação é idêntico para os dois modos de operação. Após finalizado o primeiro ciclo de replicação, o consumidor passa a comportar-se de forma distinta nos dois modos de operação. No modo *PULL* o consumidor fecha a conexão enquanto que no modo *PUSH* o consumidor mantêm a conexão em aberto.

Na Figura 3.9, após passado um período de tempo, é realizada uma alteração no diretório provedor. No passo 6 do diagrama 3.9(b) as atualizações do diretório são enviadas imediatamente ao consumidor enquanto que no diagrama da Figura 3.9(a) a atualização será realizada apenas no passo 12, após finalizado o próximo ciclo de replicação periódico.

O *syncrepl* é um mecanismo de replicação baseado em entradas. Quando o atributo de uma entrada é alterado, o provedor ao enviar a atualização para o consumidor não envia apenas o atributo alterado, mas sim, todo conjunto de atributos que compõem a entrada modificada. Se uma entrada contém dezenas de atributos, todos atributos são enviados durante o processo de replicação, nesse caso, atualizações desnecessárias.

Uma alternativa disponível no *OpenLDAP* capaz de reduzir o tráfego gerado, enviando apenas os atributos modificados e não a entrada por completa é o mecanismo denominado *Delta Syncrepl* (OPENLDAP, 2010). O servidor utiliza um módulo do serviço para armazenar na forma de *logs*, as alterações no diretório. Esses *logs* auxiliam na troca de mensagens de replicação entre provedor e consumidor.

Como mencionado anteriormente, um servidor de diretórios consumidor atende apenas a demanda de leitura das informações. Qualquer operação de escrita direcionada a um con-

sumidor, a resposta será uma mensagem, contendo uma referência (*referral*) para o servidor provedor, como ilustra a Figura 3.10.

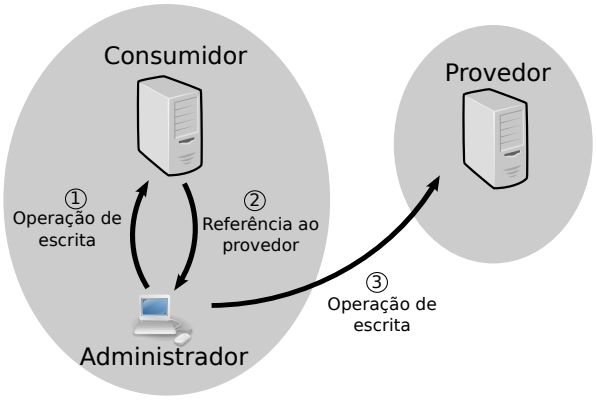


Figura 3.10: Operações de escrita no consumidor são retornadas referências ao provedor.

No *OpenLDAP* a referência pode ser processada utilizando o módulo *chain* e o consumidor pode ser configurado para que ele apenas processe a referência se a identidade a qual solicitou a operação de escrita, equivalha à identidade administrativa do diretório. Isso pode reduzir significativamente o tráfego na rede WAN, pois evita que qualquer identidade que não possua autorização de escrita no diretório, realize uma operação remota desnecessária ao provedor.

O fato de apenas um servidor provedor de diretórios atender a demanda de escrita, faz com que esse provedor esteja amparado por uma infraestrutura adequada, com eficientes recursos computacionais e um enlace WAN eficiente. Essa responsabilidade do provedor ficar disponível interruptamente, apresenta um ponto crítico no modelo de replicação *Single Master*. O modelo, caracteriza-se nesse aspecto por apresentar um ponto único de falha.

Tomando como exemplo o cenário de diretórios distribuídos para uma instituição acadêmica, se cada campus mantêm um servidor de diretórios consumidor e para que os administradores de rede de cada campus realizem suas operações no diretório, o provedor deverá estar sempre disponível, afinal, caso o provedor fique fora de operação, nenhuma manutenção no diretório será realizada.

Uma solução para o ponto único de falha é a implementação conjunta do método de replicação *Mirror Mode* (OPENLDAP, 2010) disponível no *OpenLDAP*. A replicação *Mirror Mode* conta com dois servidores provedores que replicam informações entre si, porém apenas um pode atuar como provedor para os consumidores enquanto o segundo permanece em *standby*. Se o provedor em operação ficar fora do ar é possível encaminhar as operações de escrita para o segundo provedor que nesse caso torna-se o provedor principal.

Apesar do método *Mirror Mode* amenizar o problema do ponto único de falha, os adminis-

tradores continuam dependendo de operações remotas para realizar a escrita no diretório. Uma forma de estabelecer as escritas localmente é alcançada com a implementação do modelo de replicação *Multi Master* (OPENLDAP, 2010).

3.3.3 Modelo baseado em replicação *Multi Master*

No modelo de replicação *Mulit Master* fornecido pelo *OpenLDAP*, cada servidor componente do cenário de replicação tem um comportamento híbrido, ou seja, ao mesmo tempo que um servidor é provedor ele também atua como um consumidor perante aos demais servidores de diretórios. A maior vantagem desse modelo é a alta disponibilidade do serviço. Se um dos servidores sair de operação, os demais continuarão operando normalmente.

No modelo *Multi Master* todas operações são realizadas localmente. O administrador não precisa recorrer a um servidor remoto para inserir entradas no diretório. A Figura 3.11 ilustra o cenário de replicação *Multi Master*.

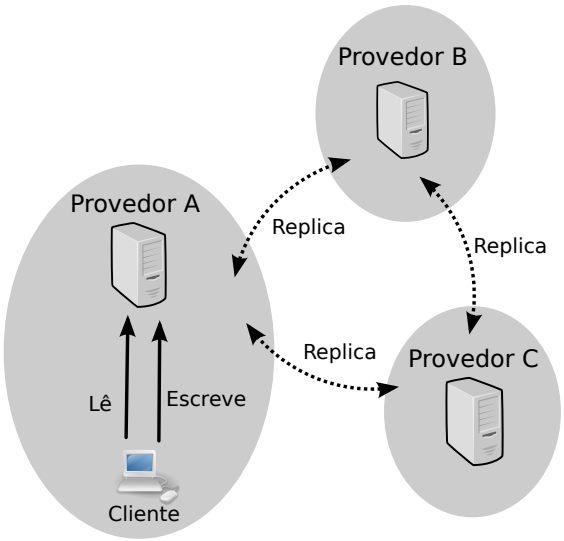


Figura 3.11: Comportamento dos servidores na replicação *Multi Master*.

O processo básico de controle de replicação é semelhante ao apresentado no modelo *Single Master*. Os servidores trocam *cookies* de sessão contendo o (*Change Sequence Number* – CSN) de estado de sincronismo. A replicação opera necessariamente em modo *PUSH* para que o sincronismo entre os servidores seja em tempo real. Esse é um requisito técnico essencial para manter os diretórios estáveis. A Figura 3.12 ilustra a troca de mensagens entre três provedores quando uma entrada é inserida em um dos provedores.

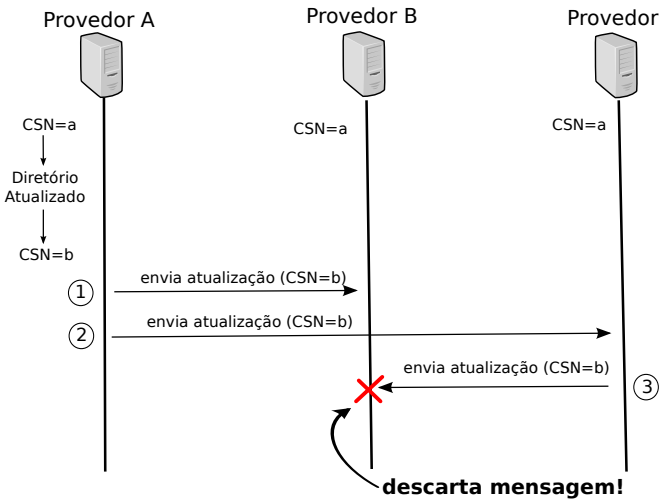


Figura 3.12: Troca de mensagens na replicação *Multi Master*.

Os provedores ilustrados na Figura 3.12 estão inicialmente em sincronismo (CSN=a). Em um determinado momento, o diretório do provedor `A` é alterado e seu estado é atualizado (CSN=b). Através do modo de operação *PUSH*, o provedor `A`, imediatamente envia a atualização para os demais provedores que se comportam como consumidores.

O provedor `C` ao receber a mensagem de atualização do provedor `A` (mensagem 2), a repassa para o provedor `B` (mensagem 3). Entretanto, o provedor `B` já contém o último estado do diretório e portanto, descarta a atualização enviada pelo provedor `C`.

Qualquer provedor ao receber uma atualização, repassa essa informação para os demais provedores. A medida que cada provedor recebe uma atualização, as mensagens de atualizações subsequentes são descartadas e assim permanecem até que todos provedores estabeleçam o sincronismo do diretório. Nesse aspecto, pode-se afirmar que a quantidade de tráfego gerado na replicação é diretamente proporcional a quantidade de provedores envolvidos no ambiente de replicação.

No modelo *Multi Master*, todos servidores recebem operações de escrita. Em uma eventual circunstância, é possível que ao mesmo tempo, uma mesma entrada seja inserida em diferentes provedores. Se a replicação dessas entradas ocorrer no mesmo ciclo de replicação, não há garantia de qual entrada permanecerá no diretório. Uma circunstância dessa natureza pode ser considerada rara, no entanto, quando ocorrida, a consistência dos dados no diretório é comprometida (ZEILENGA, 2004). A Figura 3.13 ilustra uma situação onde uma mesma entrada é inserida ao mesmo tempo, em diferentes provedores.

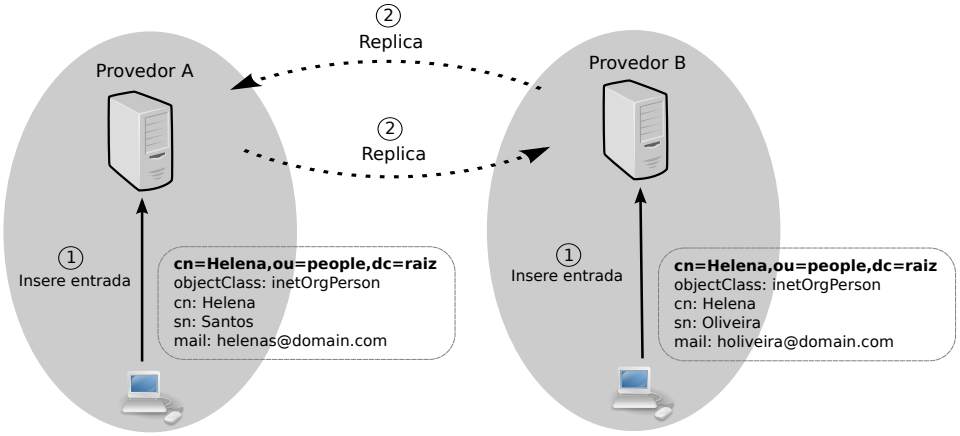


Figura 3.13: Inconsistência no diretório. Uma mesma entrada inserida em diferentes provedores.

Apesar das entradas ilustrada na Figura 3.13 parecerem diferentes, elas contém o mesmo nome distinto (*Distinguished Name* – DN). No momento que cada provedor replicar a sua atualização, o resultado final será incerto pois apenas uma das entradas pode existir no diretório.

3.4 Questões de segurança

3.4.1 Autenticação no diretório

Como mencionado na sessão 2.2.4 do capítulo 2, a segurança do diretório é dividida em dois aspectos: a autenticação e a autorização. A autenticação é o processo de estabelecer uma conexão com o diretório, seja por meio de uma identidade de usuário ou através da conexão anônima. A conexão autenticada ainda pode ser dividida em dois métodos: a autenticação simples ou a autenticação baseada em mecanismos de autenticação (*Simple Authentication and Security Layer* – SASL) (MYERS, 1997). O SASL não será apresentado nesse texto, no entanto, uma abordagem sobre o assunto pode ser encontrada em (OPENLDAP, 2010).

Na autenticação simples, é feito uso de um par contendo um (*Distinguished Name* – DN) válido e uma senha, armazenada no atributo *userPassword*. Qualquer entrada que contenha o atributo *userPassword* é válida para autenticação no diretório. Já na conexão anônima não há necessidade de qualquer identidade.

Um detalhe importante é diferenciar a autenticação no diretório de qualquer autenticação referente a um serviço de rede, como por exemplo autenticação no *Linux*. Um serviço de rede comporta-se como um cliente LDAP e usa uma identidade pré-definida para abrir a conexão com o diretório. Através de uma conexão autenticada, busca no diretório as informações que serão necessárias para autenticar o usuário no serviço.

Uma boa política de segurança é adotar uma identidade de autenticação no diretório para cada serviço ou propósito específico. Para uma instituição acadêmica por exemplo, dois serviços tem a função de buscar no diretório informações de uma identidade específica. O serviço de autenticação em estações *Linux* e o provedor de identidades da Federação CAFe. Se uma identidade de autenticação no diretório é designada para cada serviço, o diretório pode gerenciar de forma mais flexível o controle de autorização sobre cada identidade. A tabela 3.5 lista um exemplo de delegação de identidades para cada serviço ou propósito.

Propósito	Descrição
Replicação do diretório	Uma identidade específica para abrir a conexão entre servidores na replicação.
Autenticação *Linux*	Uma identidade específica para buscar os atributos de um usuário Linux.
Federação CAFe	Uma identidade específica para o provedor de identidades buscar a identidade de usuário federado.
*A*dministrador	Uma identidade específica para um administrador realizar manutenções no diretório.

Tabela 3.5: Autenticação no diretório. Uma identidade para cada propósito.

A Tabela 3.5 apresenta uma ideia geral para delegação de identidades. A medida que outros serviços ou funções são criados para utilizar o serviço de diretórios, outras identidades específicas podem ser atribuídas.

3.4.2 Autorização de acesso

A autorização no diretório é o processo que determina o nível do controle de acesso a uma informação aplicado sobre uma identidade autenticada ou a uma conexão anônima. O nível da autorização varia desde a capacidade de realizar operações de escrita até a negação total do acesso. O papel da autorização no diretório é realizado pelas listas de controle de acesso (*Access Control List* – ACL). O *OpenLDAP* fornece formas flexíveis de implementar ACLs no diretório. A tabela 3.6 lista os níveis de autorização possíveis para uma identidade.

Autorização	Descrição
none	Nenhum acesso
disclose	Permite apenas receber mensagens de erro.
auth	Permite apenas autenticar no diretório.
search	Permite realizar consultas.
compare	Comparar se um valor enviado corresponde ao armazenado.
read	Permite ler determinado valor.
write	Permite a escrita no diretório.

Tabela 3.6: Níveis de controle das (*Access Control List* – ACL).

Os níveis de autorização são cumulativos, ou seja, o nível que autoriza operações de escrita também permite realizar todas as demais operações. Geralmente os níveis mais utilizados nas declarações de ACLs são para leitura (*read*), escrita (*write*) ou nenhum acesso (*none*).

Para o serviço de diretórios de uma instituição multi campi, as ACLs devem controlar o acesso às informações de acordo com o propósito do serviço que utiliza o diretório. A identidade designada a buscar informações de autenticação de usuários *Linux*, deve pelo menos ter acesso de leitura sobre os atributos da classe de objeto *posixAccount*, *shadowAccount* e *posixGroup* de todas as entradas. Já a identidade designada a buscar informações para autenticação na federação CAFe, a autorização deve permitir que a identidade tenha permissão de leitura sobre todos atributos contidos no esquema *brEduPerson*.

Para a identidade responsável pela replicação do diretório, a permissão de acesso deve ser de leitura total sobre o diretório. E se um usuário através de seu (*Distinguished Name* – DN) realiza uma operação de leitura sobre o diretório, as ACLs devem garantir que o acesso seja restrito apenas aos atributos referentes a sua entrada.

Quanto a autorização de acesso para conexões anônimas, o diretório deve proibir qualquer acesso. Há casos que uma conexão anônima é utilizada para buscar os atributos de nome e telefone de usuários para compor um catálogo de contatos. Nesse caso, é recomendável que a consulta seja autenticada por alguma identidade específica para essa finalidade, cuja autorização seja adequada apenas para o acesso os atributos nome e telefone por exemplo.

3.4.3 Administração do diretório no ambiente distribuído

Uma identidade administrativa a princípio, é capaz de realizar qualquer operação no diretório. Através dela, um administrador de rede insere, modifica ou remove informações do diretório. No *OpenLDAP* existem duas formas de atribuir uma identidade administrativa. A primeira é por meio da declaração de um DN genérico e uma senha no arquivo de configuração do serviço. Essa identidade tem a capacidade de gerenciar todo diretório, independente de ACLs. É através dela que o administrador cria a estrutura inicial do diretório.

A segunda forma de atribuir uma identidade administrativa, é através da criação de uma entrada de usuário e aplicando regras ACLs sobre ela, de forma que permita essa entrada realizar operações de escrita. Esse método é mais flexível, pois permite que as ACLs imponham limites à identidade.

No cenário de diretório distribuídos para instituições multi campi, a administração do diretório é caracterizada pela multiplicidade de administradores. Cada administrador de cada

campus necessita inserir e realizar a manutenção das informações pertinentes ao seu campus. Nesse aspecto, é fundamental adotar uma política de acesso que garanta a organização do diretório.

No modelo particionado (ver sessão 3.3.1), a política de administração é mais simples, pois cada partição do diretório é administrada de forma isolada pelo administrador local da partição. As ACLs aplicadas na partição podem ser configuradas para que um único administrador gerencie as informações.

No modelo baseado em replicação (ver sessões 3.3.2 e 3.3.3), a administração do diretório apresenta alguns fatores mais sensíveis quanto a delegação de privilégios sobre o diretório, pois na replicação, a administração é compartilhada. Cada administrador gerencia as informações do diretório junto aos demais. Se todos possuem privilégios de escrita sobre todo o diretório, um administrador de rede de um campus pode remover ou modificar entradas pertinentes a outro campus, por exemplo. As informações inseridas por um administrador não devem ser modificadas ou removidas pelos demais. Se todos administradores tem a capacidade de gerenciar todo o conteúdo do diretório, sua organização e consistência são comprometidas.

Uma boa política, é restringir o papel dos administradores sobre o diretório. A autorização de acesso pode por exemplo, permitir que todos administradores do diretório tenham permissão para inserir novas entradas abaixo de um ramo específico. Entretanto, a política ainda pode determinar que essas entradas inseridas são apenas modificadas ou removidas pelo administrador que as criou. Dessa forma o diretório é compartilhado por cada campus e a manutenção das entradas de cada campus é realizada apenas pelo seu respectivo administrador.

Um exemplo prático, disponível pelos recursos das ACLs no *OpenLDAP*, capaz de satisfazer a restrição de privilégios sobre a administração do diretório, é apresentada pela Listagem 3.1.

```
1  access to dn.exact="ou=people,dc=raiz" attrs=children
2          by group.exact="cn=admins,ou=group,dc=raiz" write
3          by * none
4
5  access to dn.children="ou=people,dc=raiz" attrs=entry,@extensibleObject
6          by set="this/creatorsName & user" write
7          by group.exact="cn=admins,ou=group,dc=raiz" read
8          by group.exact="cn=leitores,ou=group,dc=raiz" read
9          by set="this/-* & user" read
10         by * none
```

Listagem 3.1: Exemplo de declaração de uma (*Access Control List* – ACL) para restringir privilégios sobre o diretório.

A Figura 3.1 apresenta um exemplo de aplicação de ACL para restringir privilégios sobre o ramo `ou=people` do diretório. A composição da ACL utiliza duas regras que trabalham em conjunto. A primeira regra (linhas 1 a 3), determina que o grupo administradores pode realizar operações de escrita, incluindo operações de inserir, modificar e remover, em todas entradas abaixo do ramo `ou=people`.

A segunda regra (linhas 5 a 10), complementa a primeira regra, aplicando de forma mais restritiva o acesso por determinada identidade. Na linha 6, uma cláusula determina que uma entrada abaixo do ramo `ou=people` pode ser alterada apenas pelo seu criador. O (*Distinguished Name* – DN) da identidade na qual inseriu uma entrada no diretório, é armazenado em um atributo operacional chamado *creatorsName* e seu valor é insubstituível.

As linhas restantes, são cláusulas que controlam o acesso para as demais identidades. Na linha 8, uma cláusula determina que as identidades do grupo leitores, possuem acesso somente leitura em todas as entradas abaixo do ramo `ou=people`. Na linha 9, a regra determina que uma identidade de usuário tem acesso, apenas às suas respectivas informações. E a última cláusula na linha 10, restingue qualquer acesso a identidades que não correspondam qualquer uma das cláusulas anteriores, inclusive acessos anônimos.

3.4.4 Confidencialidade no tráfego

No processo de autenticação simples, o cliente através da mensagem *bind*, envia o nome distinto (*Distinguished Name* – DN) e a senha armazenada no atributo *userPassword* da entrada. O serviço ao receber a credencial, procura pela entrada e compara a senha recebida com o valor armazenado no atributo. O valor armazenado no atributo é o *hash* da senha da identidade. O *OpenLDAP* permite a utilização de uma série de algorítimos de *hash* para armazenar o valor de senha correspondente, sendo que o mais usual é o algorítimo (*Salted Secure Hash Algorithm* – SSHA) (GUTIERREZ; GALLAGHER, 2008). O diretório ao armazenar apenas o *hash* correspondente oferece um nível seguro quanto a leitura desses valores, afinal, não é possível recuperar um valor de senha desconhecido, por meio do seu *hash*.

Apesar disso, um cliente ao realizar uma conexão com o servidor, envia sua credencial em texto claro. As informações ficam sujeitas a interceptação por meio de escuta de tráfego na rede. Essa situação fica mais crítica quando a conexão envolve tráfego pela (*Wide Area Network* – WAN) como nas trocas de mensagens de replicação entre servidores. Se as informações trafegam em texto claro, todo conteúdo do diretório fica vulnerável.

O (*Secure Sockets Layer* – SSL) e o (*Transport Layer Security* – TLS) (DIERKS; ALLEN, 1999) podem ser usados como mecanismos criptográficos para garantir confidencialidade das informações trafegadas pela rede. O SSL e o TLS estabelecem um meio seguro de tráfego nas conexões através da criptografia de chave pública e certificados X.509 (HOUSLEY et al., 1999).

O *OpenLDAP* trabalha com dois métodos para estabelecer uma conexão criptografada entre cliente e servidor. A primeira, utilizada nas versões mais antigas do serviço, baseia-se na conexão via (*Secure Sockets Layer* – SSL). Os clientes são configurados para acessar endereço do servidor através de uma porta segura (normalmente a 636) utilizada para conexões cifradas, enquanto as conexões sem mecanismos seguros são aceitas pela porta padrão do serviço (389).

O segundo método de conexão cifrada, utiliza a operação estendida *StartTLS* (HARRISON, 2006). Ao abrir uma conexão com o servidor LDAP, o cliente envia primeiramente uma operação estendida *StartTLS*, indicando que a conexão será cifrada. Para isso, o cliente deve possuir um certificado válido do servidor que contém a chave para cifragem dos dados. A partir da abertura de conexão via *StartTLS*, todas mensagens trafegam de forma segura por meio de criptografia simétrica. A vantagem desse método é que a conexão é realizada através da porta padrão do serviço, sem a necessidade de uma porta extra para abrir conexões seguras.

Através da utilização de SSL e TLS, os dados podem trafegar de forma segura pela rede, evitando a interceptação de informações por terceiros, principalmente no tráfego entre servidores que utilizam a internet como meio de conexão para a replicação do diretório.

3.5 Conclusões

Nesse capítulo foram apresentados os modelos para distribuição de um diretório. Pôde-se observar que cada modelo possui características particulares quanto aos recursos técnicos e ao comportamento do serviço, quando aplicado ao propósito de prover a integração de informações entre diferentes servidores, representados pelos campi que compõe uma instituição de ensino.

O modelo baseado em particionamento por exemplo, permite que o diretório de cada campus seja independente dos demais. A união das partições independentes é realizada por meio das referências. Por outro lado, os modelos baseados em replicação, fornecem uma única estrutura de diretórios que é compartilhada e disponível para cada campus. Ao realizar uma análise comparativa dos prós e contras de cada modelo de distribuição, os modelos baseados em replicação apresentam-se como os mais indicados para a distribuição do serviço de diretórios em uma instituição acadêmica. A Tabela 3.7 fornece argumentos para essa afirmação.

Modelos	Vantagens	Desvantagens
Particionamento	• Cada partição é administrada isoladamente, evitando ACLs complexas em cada partição.	• Consultas remotas; • Espaço de nomes baseado na divisão de campi. Não garante unicidade de nomes de usuários; • Configuração e capacidade de extensão mais complexas.
Replicação *Single Master*	• As operações de consulta são realizadas localmente; • Simplicidade de configuração e extensão; • Permite replicação *delta*; • Opera em modo *PUSH* ou *PULL*.	• Ponto único de falha; • Operações de escrita realizadas remotamente.
Replicação *Multi Master*	• Alta disponibilidade; • Todas as operações são realizadas localmente.	• Maior tráfego de mensagens na replicação; • Possibilidade de inconsistência dos dados se duas escritas são realizadas em uma mesma entrada dentro do mesmo ciclo de replicação; • Configuração e capacidade de extensão mais complexas.

Tabela 3.7: Vantagens e desvantagens de cada modelo de distribuição.

A Tabela 3.7 resume as vantagens e desvantagens de cada um dos modelos de distribuição. O modelo baseado em replicação, tanto na configuração *Single Master* quanto na *Multi Master*, atendem satisfatoriamente o propósito de distribuir as informações do diretório entre múltiplos campi. As informações ficam dispostas localmente e o espaço de nomes é modelado de forma mais simples e funcional.

A escolha de qual modelo de replicação adequa-se melhor em um cenário real, depende de fatores relacionados infraestrutura de rede disponível em cada servidor e a rotina usual do diretório, por exemplo: se os administradores apenas realizam eventualmente alguma manutenção no diretório, não faria sentido configurar os servidores para replicar os dados em tempo real, pois a conexão por meio do enlace WAN da rede permaneceria em aberto sem nenhum fluxo de dados sendo trafegado.

Outro aspecto determinante na escolha de um dos modelos baseados em replicação é a quantidade de campi que compõem a instituição de ensino. Considerando uma instituição composta por dez campi, a configuração baseada em replicação *Multi Master* exigiria que cada servidor mantivesse nove conexões em aberto. Por outro lado, na replicação *Single Master*, seria necessário apenas uma conexão com o provedor e ainda seria possível escolher entre os

modos de operação *PUSH* ou *PULL*.

A configuração do *OpenLDAP* e a facilidade de extensão (a criação de um novo campus por exemplo), também é um fator diferencial entre os dois modelos de replicação. No modelo *Single Master*, configurar um novo servidor de um novo campus para o ambiente de diretórios distribuídos é mais fácil, comparado à configuração necessária no modelo *Multi Master*, afinal, para o *Multi Master*, seria necessário alterar a configuração de todos os servidores para comportarem-se como consumidores do novo provedor de diretório. Na replicação *Single Master*, a configuração seria resumida em configurar o novo servidor consumidor para realizar as atualizações no provedor de diretórios.

4 Experimentos e Resultados

Este capítulo tem por objetivo realizar experimentos em cada modelo de replicação, levando em consideração os conceitos apresentados no capítulo 3. Para realizar os experimentos, montou-se um cenário composto por um serviço de diretórios, distribuídos por três diferentes servidores, simulando assim, uma instituição acadêmica composta por três campi, como mostra a Figura 4.1.

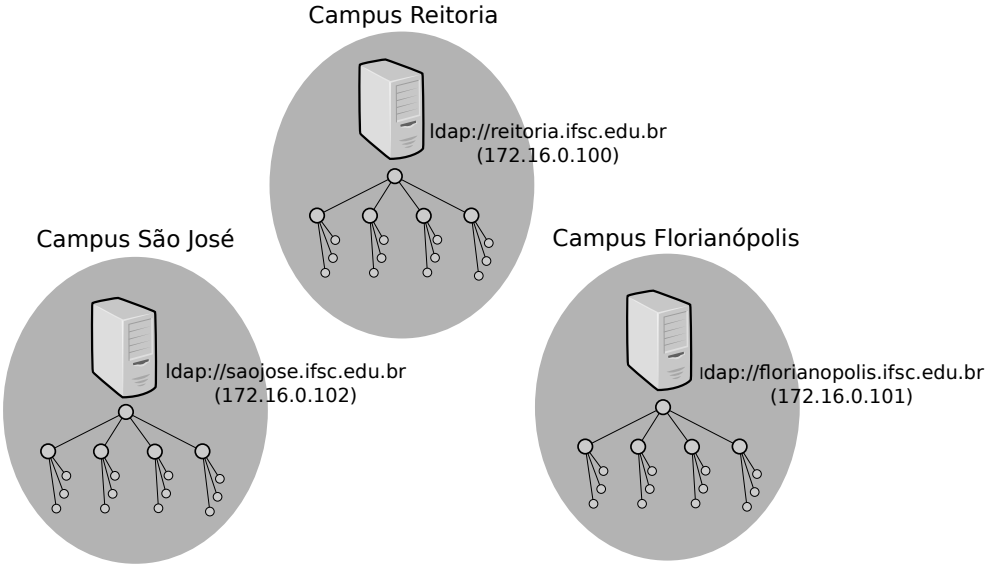


Figura 4.1: Composição do cenário para os experimentos.

Para a composição do espaço de nomes, a árvore do diretório desse cenário foi planejado para armazenar entradas de usuários, grupos e computadores do domínio da rede, conforme a organização baseada em entradas de natureza comum apresentadas na seção 3.2. A Figura 4.2 ilustra a composição da árvore do diretório.

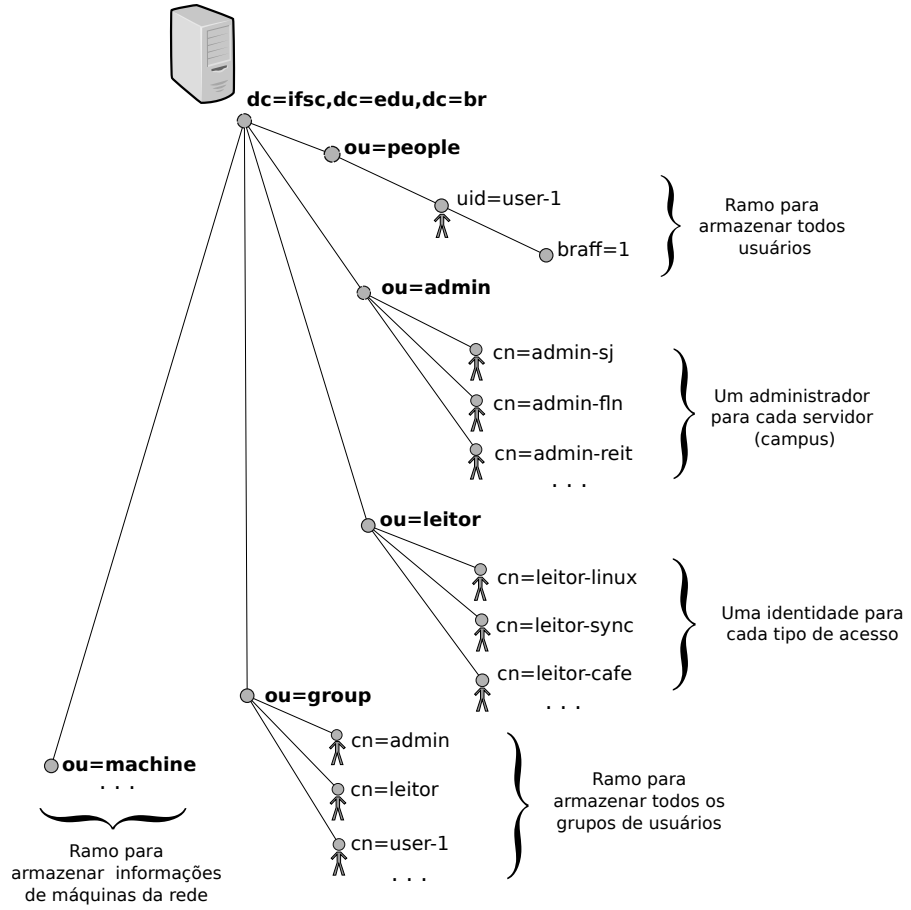


Figura 4.2: Espaço de nomes do diretório adotado no cenário de experimentos.

A Figura 4.2, ilustra a organização das entradas do diretório. Baseado nos conceitos de segurança apresentados na seção 3.4, foram criadas identidades de autenticação para cada serviço ou propósito específico. Na Figura 4.2, o ramo `ou=leitor` abriga as identidades responsáveis pela abertura de conexão para um determinado serviço e o ramo `ou=admin` abriga as identidades administrativas. A representação (*LDAP Data Interchange Format* – LDIF) da estrutura básica da árvore da Figura 4.2 é apresentada na Listagem D.1 do Anexo D. O modelo de entrada de usuário utilizado é apresentado na Listagem D.2 do Anexo D.

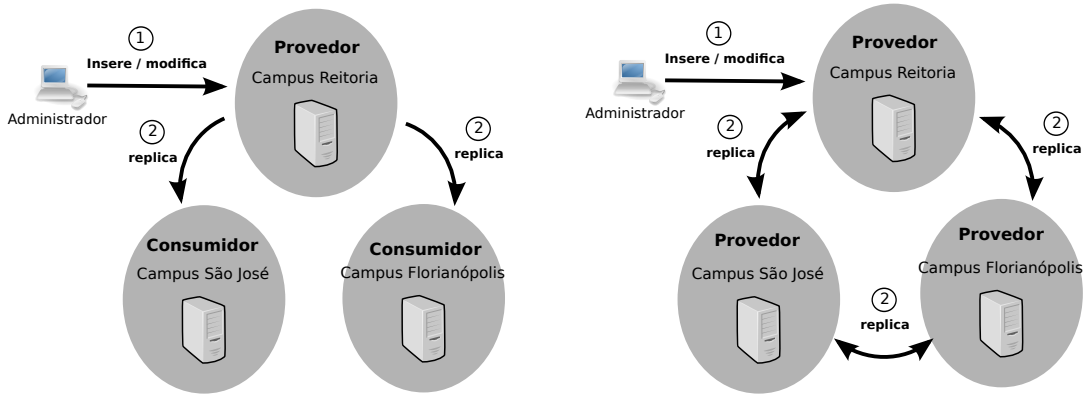
O cenário de experimentos foi configurado e aplicado nos seguintes modelos de replicação:

- ***Single Master***. Um servidor opera como provedor do diretório e dois servidores como consumidores. O ciclo de replicação opera em modo *PULL*, ou seja, periodicamente;
- ***Delta***. Semelhante ao *Single Master*. Diferencia-se apenas por algumas configurações adicionais no *OpenLDAP* para permitir que a replicação ocorra apenas nos atributos modificados e não na entrada por completa;
- ***Multi Master***. Os três servidores do cenário comportam-se como provedores e consu-

midores entre si. A replicação opera em modo *PUSH*, ou seja, os servidores mantêm a conexão permanentemente aberta e a replicação ocorre em tempo real.

4.1 Tráfego gerado durante a replicação.

Este experimento avalia a quantidade de tráfego envolvido durante a replicação de informações para cada modelo de replicação estudado. O processo do experimento é ilustrado nas Figuras 4.3(a) e 4.3(b).



(a) Processo realizado na captura de tráfego na replicação *Single Master e Delta*.

(b) Processo realizado na captura de tráfego na replicação *Multi Master*.

Figura 4.3: Processos envolvidos na captura de tráfego na replicação.

O tráfego gerado na replicação foi analisado nas seguintes situações:

- Tráfego gerado ao replicar uma nova entrada de usuário: O administrador insere uma entrada de usuário no provedor. A entrada de usuário contém informações que ao todo somam 841 *Bytes* de informação. A entrada de usuário é apresentada no formato (*LDAP Data Interchange Format* – LDIF) na Listagem D.2 do Anexo D;

- Tráfego gerado ao replicar uma alteração na entrada do usuário: o administrador altera o valor do atributo `schacGender`, resultando em um total de 14 *Bytes* de informação. Para isso, foi utilizado um arquivo LDIF ilustrado pela Listagem D.3 do Anexo D;

- Tráfego gerado ao replicar 100 novas entradas de usuários: O administrador insere em lote, 100 novas entradas de usuários. Para isso, foi utilizado um *Script shell* que gera um arquivo LDIF contendo 100 entradas de usuários, somando ao total 85,11 *KBytes* de informação. O *script* utilizado é apresentado na Listagem D.4 do Anexo D;

- Tráfego gerado ao replicar a alteração em cada uma das 100 entradas de usuários: O administrador altera em lote o atributo `schacGender` de cada um dos 100 usuários inseridos, resultando em um total de 140 *Bytes* de informação. Para isso, foi utilizado um *script shell* para gerar um arquivo LDIF contendo as alterações. O *script* é apresentado na Listagem D.5 do anexo D.

De acordo com as Figuras 4.3(a) e 4.3(b), o administrador realiza uma operação de escrita no diretório provedor para cada uma das situações descritas acima. No momento em que as informações são replicadas para os demais servidores, é realizada a captura de pacotes utilizando o *tcpdump*[1] na interface do servidor representado pelo campus São José, capturando todo tráfego recebido por esse servidor. Para avaliar a quantidade de tráfego recebido, foi utilizado o *software Wireshark*[2]. Foram realizadas cinco tentativas em cada configuração e obtida a média de tráfego total gerado durante a replicação. Os resultados obtidos são listados na Tabela 4.1.

	Single Master	*Delta*	*Multi Master*
Tráfego ao replicar uma entrada inserida	**2,4 KB**	**3,7 KB**	**5,6 KB**
Tráfego ao replicar um atributo alterado	**1,6 KB**	**1,3 KB**	**2,5 KB**
Tráfego ao replicar 100 entradas inseridas	**154,8 KB**	**280,5 KB**	**491,3 KB**
Tráfego ao replicar um atributo alterado em cada uma das 100 entradas	**84,4 KB**	**49,7 KB**	**247,7 KB**

Tabela 4.1: Resultados obtidos na captura de tráfego na replicação.

De acordo com a Tabela 4.1 os resultados obtidos para os cenários baseados em replicação *Single Master* e *Delta*, tiveram pouca diferença. A replicação *Delta*, durante a replicação das entradas inseridas, gerou mais tráfego devido ao fluxo de mensagens de controle necessárias para a consulta ao provedor. Por outro lado, a configuração *Delta* gerou menos tráfego durante a replicação das alterações dos atributos, justamente por replicar apenas o atributo alterado, ao invés da entrada por completa.

Nos resultados obtidos na configuração *Multi Master*, pôde-se observar que a quantidade total de tráfego gerado durante replicação, praticamente dobrou. Isso se deve ao fato de que um servidor recebe mensagens de atualização de todos os servidores envolvidos no cenário, (ao contrário da replicação *Single Master* onde as mensagens da replicação são estabelecidas exclusivamente entre consumidor e provedor). Se o cenário de replicação *Multi Master* desse

[1] `http://www.tcpdump.org/`
[2] `http://www.wireshark.org`

experimento, fosse composto por mais provedores, o tráfego total recebido por um dos servidores, aumentaria proporcionalmente.

4.2 Concorrência ao inserir entradas idênticas simultaneamente.

No modelo baseado em replicação *Multi Master*, todos servidores do cenário comportam-se como provedores, logo, qualquer servidor aceita receber operações de escrita. Esse experimento tem por objetivo verificar o comportamento do serviço de diretórios quando dois administradores inserem nos servidores de seus respectivos campi, uma entrada com o mesmo (*Distinguished Name* – DN), na mesma fração de tempo.

Para automatizar a tarefa de inserir as duas entradas na mesma fração de tempo, foi utilizado a agendador de tarefas *crontab* do *Linux* que executará um mesmo *script* nos dois servidores. O relógio dos servidores foram configurados com uma mesma fonte de horário, para garantir que os *scripts* sejam executados simultaneamente. O *script* executado é semelhante ao da Listagem D.4, apenas com diferença de gerar uma única entrada de usuário, ao invés de 100.

Ao executar os *scripts*, foi analisado os *logs* gerados pelo *OpenLDAP* no processo de inserção das entradas, como mostra a Listagem 4.1.

```
1  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 fd=21 ACCEPT from IP=[::1]:40814 (IP=[::]:389)
2  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=0 BIND dn="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=
       br" method=128
3  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=0 BIND dn="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=
       br" mech=SIMPLE ssf=0
4  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=0 RESULT tag=97 err=0 text=
5  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=1 ADD dn="uid=user-1,ou=people,dc=ifsc,dc=edu,dc=
       br"
6  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=1 RESULT tag=105 err=0 text=
7  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=2 ADD dn="brEduAffiliation=2,uid=user-1,ou=people,
       dc=ifsc,dc=edu,dc=br"
8  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=3 ADD dn="cn=user-1,ou=group,dc=ifsc,dc=edu,dc=br"
9  Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=2 RESULT tag=105 err=0 text=
10 Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=3 RESULT tag=105 err=0 text=
11 Nov 2 01:32:01 saojose slapd[7627]: conn=1002 op=4 UNBIND
12 Nov 2 01:32:01 saojose slapd[7627]: conn=1002 fd=21 closed
13
14
15 Nov 2 01:32:01 florianopolis slapd[12115]: conn=1002 fd=30 ACCEPT from IP=[::1]:49162 (IP=[::]:389)
16 Nov 2 01:32:01 florianopolis slapd[12115]: conn=1002 op=0 BIND dn="cn=admin-fln,ou=admin,dc=ifsc,dc
       =edu,dc=br" method=128
17 Nov 2 01:32:01 florianopolis slapd[12115]: conn=1002 op=0 BIND dn="cn=admin-fln,ou=admin,dc=ifsc,dc
       =edu,dc=br" mech=SIMPLE ssf=0
18 Nov 2 01:32:01 florianopolis slapd[12115]: conn=1002 op=0 RESULT tag=97 err=0 text=
```

```
19 Nov  2 01:32:01 florianopolis slapd[12115]: conn=1002 op=1 ADD dn="uid=user-1,ou=people,dc=ifsc,dc=
       edu,dc=br"
20 Nov  2 01:32:01 florianopolis slapd[12115]: conn=1002 op=1 RESULT tag=105 err=68 text=
21 Nov  2 01:32:01 florianopolis slapd[12115]: conn=1002 op=2 UNBIND
22 Nov  2 01:32:01 florianopolis slapd[12115]: conn=1002 fd=30 closed
```

Listagem 4.1: *Logs* gerados ao inserir entrada concorrentes.

Na Listagem 4.1, os *logs* exibem o registro completo das duas operações de inserção. As linhas 1 a 12, representam o *log* gerado pelo servidor representado pelo campus São José, enquanto as linhas 15 a 22, representam o *log* do servidor representado pelo campus Florianópolis.

Acompanhando as mensagens de *log*, nas linhas 5 e 19, os servidores processam a operação de inserção e nas linhas 6 e 20 são apresentadas as respostas dos servidores para a operação. Enquanto que na linha 6 a mensagem indica sucesso na inserção da entrada, a linha 20 indica que a entrada já existe no diretório (`err=68`). Pode-se concluir que o servidor de São José ao receber a nova entrada, replicou essa informação antes do servidor de Florianópolis processar o pedido de inserção de uma nova entrada no seu diretório.

É possível observar que todos os registros nos *logs* dos servidores foram gerados na mesma fração de segundo. Em um ambiente real, onde o tráfego da replicação utiliza a rede WAN para enviar as atualizações para todos servidores envolvidos, a probabilidade da operação ocorrer na mesma fração de segundos é muito baixa. Logo, não é possível garantir que a replicação ocorra antes do processamento da segunda inserção da entrada, como foi observado no experimento.

4.3 Restabelecimento crítico do sincronismo no *Multi Master*.

A replicação *Multi Master* opera em modo *PUSH*, ou seja, qualquer alteração realizada em um dos provedores, é enviada imediatamente aos demais. Existem situações em que um provedor pode perder a conexão com os servidores, como por exemplo, quando o enlace WAN é interrompido. Apesar do servidor perder a conexão com os provedores envolvidos na replicação, ele ainda opera normalmente dentro da rede local em que atua, aceitando operações de leitura e escrita. Se esse provedor receber qualquer alteração no diretório durante a falta de conexão com os demais provedores, a replicação será realizada apenas quando o enlace for restabelecido.

Para esse experimento, será avaliado o comportamento do processo de replicação *Multi Master* quando um dos provedores perde a conexão com os demais servidores durante um período de tempo. Nesse período, duas entradas distintas, porém compostas com o mesmo (*Distinguished Name* – DN) são inseridas no diretório. Uma delas é inserida no provedor sem

conexão, enquanto a outra é inserida em um outro provedor, cuja replicação ocorre imediatamente. A entrada inserida é idêntica a apresentada pela Listagem D.2. A Figura 4.4 ilustra o cenário do experimento 4.

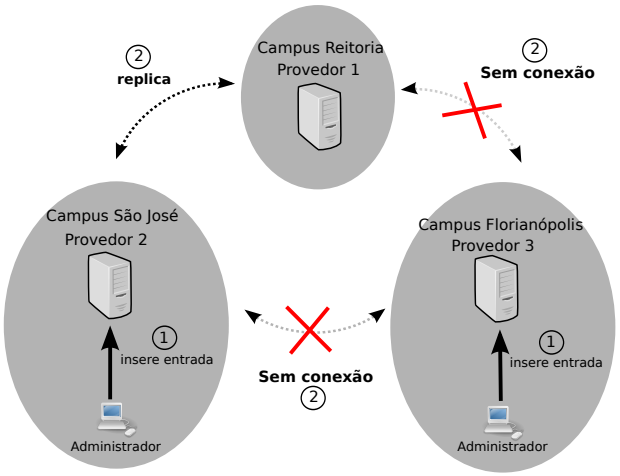


Figura 4.4: Sincronismo crítico no *Multi Master*.

Na Figura 4.4, o provedor do campus Florianópolis durante um período de tempo perde a conexão com os demais servidores, logo, qualquer alteração realizada em Florianópolis não é replicada de imediato aos provedores da Reitoria e de São José. Nesse período de tempo, uma entrada é inserida no provedor de Florianópolis. Uma entrada como o mesmo (*Distinguished Name* – DN) também é inserida no provedor de São José. A entrada inserida em São José é imediatamente replicada para o provedor da Reitoria.

No momento que o provedor de Florianópolis restabelecer a conexão com os demais provedores, ele tentará replicar a entrada inserida durante o período em que permaneceu sem conexão. No entanto, os provedores já possuem uma entrada com um mesmo (*Distinguished Name* – DN), inserida anteriormente pelo provedor de São José.

Essa situação caracteriza-se como a pior situação existente no cenário de replicação *Multi Master*, pois no momento em que o provedor *off-line* restabelecer a conexão com os demais provedores, ocorrerá a replicação mútua das duas entradas. Ao final do processo de replicação, apenas uma poderá existir no diretório de todos provedores.

Foram realizadas cinco tentativas para esse experimento. Todas elas resultaram na inconsistência do diretório. O Provedor de Florianópolis ao restabelecer a conexão, enviou suas atualizações para os demais provedores . A entrada que antes existia no provedor de São José, foi sobrescrita com a nova entrada enviada pelo provedor de Florianópolis. E no servidor da Reitoria, por alguma razão desconhecida, a entrada deixou de existir. Portanto, o sincronismo do diretório entre os servidores ficou comprometido.

4.4 Conclusões

Esse capítulo apresentou experimentos com os modelos baseados em replicação, para um ambiente de distribuição do diretório. De acordo com os resultados obtidos, em síntese, a replicação *Single Master* e a replicação *Delta* apresentaram vantagens sobre o modelo de replicação *Multi Master*.

Na seção 4.1, a carga de tráfego gerada na replicação *Multi Master* foi praticamente o dobro da gerada pela replicação *Single Master* e *Delta*. E esse valor é aumentado a medida que mais provedores de diretório são instalados no ambiente de replicação. E nas seções 4.2 e 4.3, a replicação *Multi Master* foi testada nos piores casos, onde o resultado obtido no experimento da seção 4.3 mostrou a possibilidade de inconsistência dos dados.

Entretanto, as vantagens dos modelos *Single Master* e *Delta* sobre o modelo *Multi Master* são relativas quando analisada a função do serviço de diretórios em um cenário real. O tráfego gerado pela replicação *Multi Master* ao replicar 100 entradas de usuários, resultou ao total 491,3 *KBytes*. Esse valor pode ser considerado baixo para o tráfego de informações pela WAN.

No experimento da seção 4.2 por exemplo, a chance da condição imposta ocorrer em uma situação prática, é mínima, visto que o serviço de diretórios para uma instituição acadêmica, cujo propósito principal é armazenar informações de usuários, a frequência de operações de escrita é eventual.

No experimento da seção 4.3 onde um dos provedores perde a conexão com os demais e nesse momento, uma mesma entrada é inserida em provedores distintos, a chance de ocorrer a perda de conexão em uma situação real é possível e até comum. No entanto, a chance de dois administradores inserirem ao mesmo tempo, uma mesma entrada, no intervalo em que um dos provedores está sem conexão, também pode ser considerada mínima.

Pode-se concluir que a escolha do modelo de distribuição do diretório entre os campi de uma instituição, depende apenas de outros fatores além das características técnicas dos modelos de replicação. A escolha do modelo de distribuição adequado, deve levar em consideração a frequência de operações de escrita, os recursos de processamento do servidor, a qualidade de conexão do enlace WAN da rede e a necessidade de redimensionar o conjunto de servidores que compõem o cenário de replicação.

5 *Considerações finais*

Esse trabalho apresentou um estudo sobre serviço de diretório distribuídos, aplicado a instituições acadêmicas multi campi. Dentro desse contexto, foi dado ênfase aos modelos de distribuição do diretório baseados em replicação, implementados pelo *OpenLDAP*.

Cada aspecto do planejamento do serviço de diretórios foi abordado levando em consideração dois propósitos específicos: oferecer uma infraestrutura de autenticação inter campi, onde usuários de um campus sejam capazes de autenticarem-se dentro da rede de outros campi, e a capacidade de uma instituição acadêmica fazer parte da Federação CAFe, atendendo aos requisitos técnicos mínimos, exigidos no fornecimento de identidades de usuários.

O resultado final apresentado nesse trabalho, pode ser considerado como um ponto de partida para pesquisadores, estudantes e entusiastas realizarem pesquisas e projetos relacionados ao tema em questão. Espera-se que o conteúdo organizado nesse trabalho, sirva de referência para futuros estudos sobre serviço de diretórios em geral.

Como sugestão para trabalhos futuros, pode-se citar alguns estudos relacionados ao tema serviço de diretórios, ou ainda, serviços de diretórios aplicados a instituições acadêmicas, são eles:

- **Integração de serviços utilizados em instituições acadêmicas**: O modelo de informação apresentado nesse trabalho, levou em consideração a utilização do serviço de diretórios para fornecer apenas informações de usuários para Federação café e para autenticação em sistemas baseados em *Linux*. Uma sugestão interessante para continuidade desse trabalho é realizar um estudo e implementação de serviços integrados ao diretório, como por exemplo serviço de autenticação em redes heterogêneas, utilizando controlador de domínio *Samba*[1], integração do serviço de email (*Postfix*[2] por exemplo), integração do serviço de telefonia VoIP (*Asterisk*[3] por exemplo), além de outros possíveis serviços,

[1] `http://www.samba.org/`
[2] `http://www.postfix.org/`
[3] `http://www.asterisk.org/`

existentes na rede de uma instituição de ensino;

- **Estudo sobre mecanismos de autenticação avançados**: Nesse trabalho, foi apresentado apenas o mecanismo de autenticação simples no diretório. No entanto, o *OpenLDAP* tem a capacidade de utilizar mecanismos robustos de autenticação. O (*Simple Authentication and Security Layer* – SASL) é um objeto de estudo interessante, pois oferece um conjunto de métodos de autenticação, dentre os quais o *Kerberos* [4] é um dos mais disseminados.

[4] `http://web.mit.edu/Kerberos`

ANEXO A – Configuração básica do OpenLDAP.

O Anexo A apresenta os passos para a instalação e configuração básica do *OpenLDAP*. A versão do *OpenLDAP* utilizada é a **2.4.21**. O sistema operacional utilizado nos servidores é o ***Ubuntu Server* 10.04**.

A.1 Instalação

Utilizando o gerenciador de pacotes *apt*, no terminal, instale os seguintes pacotes.

```
1 apt-get install slapd ldap-utils
```

Listagem A.1: Pacotes para instalação do *OpenLDAP*.

O pacote *slapd* é o *daemon* do serviço. O pacote *ldap-utils* é um conjunto de ferramentas para gerenciar o serviço. Após instalado, os arquivos do serviço estão localizados em `/etc/ldap/`.

A.2 Configuração inicial

As últimas versões do *OpenLDAP* utilizam uma nova forma de configuração, denominada *cn=config*. Esse método de configuração armazena todas configurações em um diretório virtual, cujas alterações são realizadas via comandos LDAP. Entretanto, nesse trabalho foi utilizado método antigo de configuração, baseado em arquivo (`slapd.conf`). A seguir, são apresentados os procedimentos iniciais para configuração do serviço.

Crie o arquivo `slapd.conf`.

```
1 touch /etc/ldap/slapd.conf
```

Listagem A.2: Criação do arquivo `slapd.conf`.

Determine adequadamente as propriedades e permissões no arquivo.

```
chown openldap\: /etc/ldap/slapd.conf ; chmod 644 /etc/ldap/slapd.conf
```

Listagem A.3: Definição das permissões do arquivo `slapd.conf`.

Configure o método de configuração. Para isso, altere no arquivo `/etc/default/slapd`, a opção, exatamente como abaixo.

```
SLAPD-CONF=/etc/ldap/slapd.conf
```

Listagem A.4: Alterando a configuração do serviço para utilizar o arquivo `slapd.conf`.

Remova o diretório utilizado pelo método de configuração `cn=config`.

```
rm -rf /etc/ldap/slapd.d
```

Listagem A.5: Removendo arquivos desnecessários do serviço.

Uma boa prática de administração é separar as mensagesn de *log* do *OpenLDAP* das demais redirecionadas para o arquivo `/var/log/syslog`. Para isso, crie um arquivo vazio (ex: `/var/log/openldap.log`) e edite o arquivo `/etc/rsyslog.conf`, adicionando a seguinte linha.

```
LOCAL4.* /var/log/openldap.log
```

Listagem A.6: Configurando o *syslog*.

Reinicie o serviço *syslog*.

```
service rsyslog restart
```

Listagem A.7: Reiniciando o *syslog*.

O conteúdo do arquivo `slapd.conf` é apresentado nos Anexos B e C, de acordo com o modelo de replicação adotado.

A.3 Configuração do (*Transport Layer Security* – TLS).

Para implementar criptografia nos dados trafegados na rede com (*Transport Layer Security* – TLS), é preciso gerar para cada servidor um certificado auto assinado. Para isso, os seguintes passos devem ser seguidos.

Instale o *OpenSSL*

```
apt-get install openssl
```

Listagem A.8: Instalação do *OpenSSL*.

Crie um diretório para o armazenamento dos arquivos e acesse o diretório.

```
mkdir /etc/ldap/tls
cd /etc/ldap/tls
```

Listagem A.9: Criando um diretório para os certificados.

Dentro do diretório corrente, crie uma agência certificadora através do *script* disponível pelo *OpenSSL*.

```
/usr/lib/ssl/misc/CA.sh -newca
```

Listagem A.10: Criando uma agência certificadora.

Nessa etapa, duas observações são importantes. No processo de criação da agência certificadora, será pedido algumas informações. Dentre as informações solicitadas, duas são obrigatoriamente requeridas: a senha e o (*Fully Qualifield Domain Name* – FQDN), ou nome de domínio totalmente qualificado do servidor.

```
Enter PEM pass phrase: UMA SENHA PARA ASSINATURA DE CERTIFICADOS
...
Common Name (ed, YOUR name) []: O FQDN DO SERVIDOR
```

Listagem A.11: Determinando a senha e o nome da agência certificadora.

Crie o certificado

```
openssl req -new -nodes -keyout newreq.pem -out newreq.pem
```

Listagem A.12: Criando um certificado.

Na geração do certificado, também será solicitado algumas informações das quais o `Common Name` (que representa FQDN do servidor) deve ser obrigatoriamente dado.

Assine o certificado com a agência certificadora

```
/usr/lib/ssl/misc/CA.sh -sign
```

Listagem A.13: Assinando o certificado.

No final do processo de assinatura do certificado, é solicitado a senha criada para a agência certificadora.

Após assinar o certificado, foram gerados dois arquivos. O certificado `newcert.pem` e a chave privada `newreq.pem`. Esses arquivos podem ser renomeados para melhor compreensão. Outro arquivo importante é o certificado da agência certificadora, que pode ser copiado e renomeado para o diretório corrente, para melhor localização.

```
1 Copiando o certificado a agencia certificadora
2 cp demoCA/cacert.pem ./
3
4 Renomenando os arquivos
5 mv newcert.pem certificado.pem
6 mv newreq.pem chave_privada.pem
7 mv cacert.pem certificado_CA.pem
```

Listagem A.14: Renomeando os arquivos do certificado criado.

Com os arquivos em mãos, pode-se utilizá-los na configuração do *OpenLDAP* para implementação do (*Transport Layer Security* – TLS) nas conexões.

A.4 Configuração do *Linux* como cliente *LDAP*.

A autenticação *Linux* por meio de um serviço de diretórios, ocorre em duas etapas: primeiramente o sistema busca as informações do usuário no diretório e em seguida compara o par *login*/senha passado pelo usuário com as informações recuperadas no diretório.

A tarefa de buscar as informações no diretório é realizada pelo módulo *nsswitch*[1] do *Linux*, enquanto a autenticação propriamente dita é realizada pelos módulos do (*Pluggable Authentication Modules* – PAM)[2].

Para autenticar um usuário no *Linux*, através do serviço de diretórios como base de dados, utilizando o sistema operacional Ubuntu *Desktop* 10.10, os seguintes passos devem ser seguidos.

Instale o pacote `libnss-ldap`:

```
1 apt-get install libnss-ldap
```

Listagem A.15: Instalação do pacote `libnss-ldap`.

[1] `http://linux.die.net/man/5/nsswitch.conf`
[2] `http://www.ibm.com/developerworks/br/library/l-pam`

Junto com o pacote `libnss-ldap` o gerenciador de pacotes instalará outros pacotes dependentes. Durante a instalação, é aberto um menu interativo que permite uma configuração facilitada. No entanto, será a apresentado a configuração manual.

Edite o arquivo `/etc/ldap.conf`, alterando as seguintes diretivas:

```
#  Configuracao minima!
#  Sufixo do diretorio
base dc=ifsc,dc=edu,dc=br
#  Endereco do servidor
uri ldap://NOMEDOSERVIDOR
#  Versao do protocolo
ldap_version 3
#  Identidade de autenticacao
binddn cn=leitor-linux,ou=leitor,dc=ifsc,dc=edu,dc=br
#  Senha da identidade
bindpw leitor
#  Ativa TLS
ssl start_tls
```

Listagem A.16: Configuração mínima do arquivo `/etc/ldap.conf`

Altere o arquivo `/etc/nsswitch.conf`, da forma como mostra a Listagem A.17

```
passwd:         ldap compat
group:          ldap compat
shadow:         ldap compat
```

Listagem A.17: Configuração do arquivo `/etc/nsswitch.conf`

Verifique se o sistema está acessando o diretório corretamente.

```
getent passwd
```

Listagem A.18: Comando para verificar se o sistema acessa o diretório corretamente.

Ao realizar o comando da Listagem A.18, o sistema deve retornar as identidades de todos os usuários armazenadas no diretório.

O (*Pluggable Authentication Modules* – PAM) é serviço responsável pela autenticação no sistema. Para o correto funcionamento do *PAM*, verifique se as seguintes diretivas estão declaradas nos respectivos arquivos de configuração:

Arquivo `/etc/pam.d/common-account`

```
account [success=1 default=ignore] pam_ldap.so
```

Listagem A.19: Diretiva necessária no arquivo `/etc/pam.d/common-account`.

Arquivo `/etc/pam.d/common-auth`

```
1 auth    [success=1 default=ignore]    pam_ldap.so use_first_pass
```

Listagem A.20: Diretiva necessária no arquivo `/etc/pam.d/common-auth`.

Arquivo `/etc/pam.d/common-password`

```
1 password      [success=1 user_unknown=ignore default=die] pam_ldap.so use_authtok try_first_pass
```

Listagem A.21: Diretiva necessária no arquivo `/etc/pam.d/common-password`.

Arquivo `/etc/pam.d/common-session`

```
1 session optional pam_ldap.so
```

Listagem A.22: Diretiva necessária no arquivo `/etc/pam.d/common-session`.

ANEXO B – Configuração do OpenLDAP para replicação Single Master e Delta.

O Anexo B apresenta os passos para configuração do *OpenLDAP* nos modelos baseados em replicação *Single Master* e *Delta*. As configurações referem-se ao arquivo `slapd.conf`, para os servidores operando como provedor e consumidor.

Devido a grande quantidade de parâmetros no arquivo `slapd.conf`, o arquivo foi dividido em três arquivos distintos, para melhor organização e compreensão. Os arquivos foram organizados da seguinte forma:

- Um arquivo que contém as configurações globais do serviço;
- Um arquivo que contém as configurações do *backend*, bem como as configurações de replicação;
- Um arquivo que contém as declarações de (*Access Control List* – ACL). O arquivo contendo as declarações de ACLs, (Figura B.3) foi utilizado na configuração de todos os modelos de replicação.

As diretivas referentes ao nome do servidor para o diretório provedor, foram baseadas no nome de domínio `reitoria.ifsc.edu.br`, enquanto que, para o servidor consumidor, essas diretivas foram baseadas no nome de domínio `saojose.ifsc.edu.br`.

B.1 Configuração para replicação *Single Master*

B.1.1 Configuração do provedor

Para a configuração do servidor provedor, a Figura B.1, B.2 e B.3 ilustram as diretivas utilizadas no arquivo `slapd.conf`. Para cada diretiva, é apresentado um comentário resumido

que explica seu propósito. Detalhes de cada diretiva, podem ser encontradas em (OPENLDAP, 2010) e (BUTCHER, 2007).

```
#   :: slapd.conf - Configuracoes Globais ::

# Suporte a versao antiga do protocolo
allow bind_v2

# Local dos esquemas utilizados pelo servidor
include         /etc/ldap/schema/core.schema
include         /etc/ldap/schema/cosine.schema
include         /etc/ldap/schema/nis.schema
include         /etc/ldap/schema/inetorgperson.schema
include         /etc/ldap/schema/brEduPerson.schema
include         /etc/ldap/schema/eduperson.schema
include         /etc/ldap/schema/schac.schema

# Local e Modulos carregados dinamicamente
modulepath      /usr/lib/ldap
moduleload      back_hdb
moduleload      syncprov
moduleload      back_monitor

#       Local do PID do processo.
pidfile         /var/run/slapd/slapd.pid
#       Local dos parametros de execucao.
argsfile        /var/run/slapd/slapd.args
#       Nivel de informacoes que devem ser armazenadas no arquivo de log.
loglevel        256 16384
#       Limite de entradas retornadas por uma consulta
sizelimit unlimited
#       Tempo maximo para o cliente obter respostas do servidor
timelimit unlimited
#       Quantidades de processos do servico
tool-threads 2
#       Tempo para anular uma conexao perdida
idletimeout 60

#       ::: Seguranca :::
#       Nega conexoes anonimas
disallow bind_anon
require bind

#       Configuracoes de certificado. (TLS)
TLSCertificateFile /etc/ldap/tls/certificado_SRV.pem
TLSCertificateKeyFile /etc/ldap/tls/chave_Privada.pem
TLSCACertificateFile /etc/ldap/tls/reitoria_CA.pem
TLSVerifyClient never

#       Determina o minimo de seguranca aplicada nas conexoes
security ssf=1 update_ssf=112 simple_bind=64

#       Backend MONITOR. Para armazenamento de estatisticas do servico
database monitor
```

```
52 access to dn.subtree="cn=Monitor"
53         by dn.exact="cn=admin-reit,ou=admin,dc=ifsc,dc=edu,dc=br" read
54         by * none
55
56 #       O arquivo abaixo continua com a configuracoes de backend para o diretorio
57 include /etc/ldap/backend_reit_singlemaster.conf
```

Listagem B.1: Configurações globais do arquivo `slapd.conf` para um provedor na replicação *Single Master*.

```
1  #       ::: backend do diretorio :::
2
3  # Cria uma instancia de um backend hdb
4  database       hdb
5
6  #       sufixo do diretorio
7  suffix         "dc=ifsc,dc=edu,dc=br"
8  #       Administrador do servico
9  rootdn         "cn=admin,dc=ifsc,dc=edu,dc=br"
10 rootpw {SSHA}NxruY48AQTXLq2EeG+s54EmR1Cugp1Su
11
12 #       Local de armazenamento dos dados
13 directory      "/var/lib/ldap/backend_reit_singlemaster"
14
15 #       Parametros especificos do backend hdb
16 dbconfig set_cachesize 0 2097152 0
17 dbconfig set_lk_max_objects 1500
18 dbconfig set_lk_max_locks 1500
19 dbconfig set_lk_max_lockers 1500
20
21 #       Indexacao de atributos
22 index      objectClass eq
23 index      entryCSN,entryUUID eq
24 index      uid,uidNumber,gidNumber eq
25 index      memberUID,mail,givenname eq
26
27 #       Permite registrar mudancas na entrada
28 lastmod   on
29 #       Checagem periodica da base
30 checkpoint     512 30
31
32 #       Ativa modo provedor de replicacao
33 overlay syncprov
34 syncprov-checkpoint 100 10
35 syncprov-sessionlog 1000
36
37 #       Arquivo contendo as ACLs
38 include /etc/ldap/acls_provedor.conf
```

Listagem B.2: Configurações de *backend* no arquivo `slapd.conf` para um provedor no modelo *Single Master*.

```
#       ::: ACLs :::
# ACL para o acesso aos atributos de senhas
access to attrs=userPassword,shadowLastChange
          by anonymous auth
          by set="this/-* & user" write
          by self write
          by set="this/creatorsName & user" write
          by group.exact="cn=leitor,ou=group,dc=ifsc,dc=edu,dc=br" read
          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" read
          by * none

# ACL para o ramo (ou=leitor)
access to dn.subtree="ou=leitor,dc=ifsc,dc=edu,dc=br"
        by group.exact="cn=leitor,ou=group,dc=ifsc,dc=edu,dc=br" read
        by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" read
        by * none

# ACL para o ramo (ou=admin)
access to dn.subtree="ou=admin,dc=ifsc,dc=edu,dc=br"
          by self write
          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" read
          by group.exact="cn=leitor,ou=group,dc=ifsc,dc=edu,dc=br" read
          by * none

# ACL para o ramo (ou=people)
access to dn.exact="ou=people,dc=ifsc,dc=edu,dc=br" attrs=children
          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" write
          by * none

# ACL para determinar que apenas o criador da entrada pode modifica-la ou remove-la
access to dn.children="ou=people,dc=ifsc,dc=edu,dc=br" attrs=entry,@extensibleObject
          by set="this/creatorsName & user" write
          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" read
          by group.exact="cn=leitor,ou=group,dc=ifsc,dc=edu,dc=br" read
          by set="this/-* & user" read
          by * none

# ACL para o ramo (ou=group)
access to dn.exact="ou=group,dc=ifsc,dc=edu,dc=br" attrs=children
          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" write
          by * none

# ACL para determinar que apenas o criador da entrada pode modifica-la ou remove-la
access to dn.children="ou=group,dc=ifsc,dc=edu,dc=br" attrs=entry,@extensibleObject
          by set="this/creatorsName & user" write
          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" read
          by group.exact="cn=leitor,ou=group,dc=ifsc,dc=edu,dc=br" read
          by * none

# ACL para o ramo (ou=machine)
access to dn.exact="ou=machine,dc=ifsc,dc=edu,dc=br" attrs=children
          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" write
          by * none
```

```
54
55 # ACL para determinar que apenas o criador da entrada pode modifica-la ou remove-la
56 access to dn.children="ou=machine,dc=ifsc,dc=edu,dc=br" attrs=entry,@extensibleObject
57          by set="this/creatorsName & user" write
58          by group.exact="cn=admin,ou=group,dc=ifsc,dc=edu,dc=br" read
59          by group.exact="cn=leitor,ou=group,dc=ifsc,dc=edu,dc=br" read
60          by * none
61
62 # ACL geral de leitura do diretorio apos aplicadas todas as regras cima
63 access to *
64          by users read
65          by * none
```

Listagem B.3: Configurações das ACLs.

B.1.2 Configuração do consumidor

As Figuras B.4 e B.5 ilustram as configurações do `slapd.conf` para o consumidor.

```
1  #      Suporte a versao antiga do protocolo
2  allow bind_v2
3  #      Local dos esquemas utilizados pelo servidor
4  include        /etc/ldap/schema/core.schema
5  include        /etc/ldap/schema/cosine.schema
6  include        /etc/ldap/schema/nis.schema
7  include        /etc/ldap/schema/inetorgperson.schema
8  include        /etc/ldap/schema/brEduPerson.schema
9  include        /etc/ldap/schema/eduperson.schema
10 include        /etc/ldap/schema/schac.schema
11 #      Local e modulos carregados dinamicamente
12 modulepath            /usr/lib/ldap
13 moduleload            back_ldap
14 moduleload            back_hdb
15 moduleload            back_monitor
16 #      Local do PID do processo.
17 pidfile        /var/run/slapd/slapd.pid
18 #      Local dos parametros de execucao.
19 argsfile       /var/run/slapd/slapd.args
20 #      Nivel de informacoes que devem ser armazenadas no arquivo de log.
21 loglevel       256 16384
22 #      Limite de entradas retornadas por uma consulta
23 sizelimit unlimited
24 #      Tempo maximo para o cliente obter respostas do servidor
25 timelimit unlimited
26 #      Quantidades de processos do servico
27 tool-threads 2
28 #      Tempo para anular uma conexao perdida
29 idletimeout 60
30
31 #      Modulo responsavel por seguir referencias (referrals)
32 overlay        chain
33 chain-uri      ldap://reitoria.ifsc.edu.br
```

```
chain-idassert-bind   bindmethod=simple
        binddn="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br"
        credentials="admin"
        mode=self
chain-return-error   true
chain-rebind-as-user true
#       Nega conexoes anonimas
disallow bind_anon
require bind
#       Configuracoes de certificado. (TLS)
TLSCertificateFile /etc/ldap/tls/certificado_SRV.pem
TLSCertificateKeyFile /etc/ldap/tls/chave_Privada.pem
TLSCACertificateFile /etc/ldap/tls/saojose_CA.pem
TLSVerifyClient never
#       Determina o minimo de seguranca aplicada nas conexoes
security ssf=1 update_ssf=112 simple_bind=64

#       Backend MONITOR. Para armazenamento de estatisticas do servico
database monitor
access to dn.subtree="cn=Monitor"
        by dn.exact="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br" read
        by * none
#       O arquivo abaixo continua com a configuracoes de backend para o diretorio
include         /etc/ldap/backend_sj_singlemaster.conf
```

Listagem B.4: Configurações globais do arquivo `slapd.conf` para o consumidor na replicação *Single Master*.

```
#       ::: backend do diretorio :::
database hdb

#       sufixo
suffix          "dc=ifsc,dc=edu,dc=br"
#       Administrador do servico
rootdn          "cn=admin,dc=ifsc,dc=edu,dc=br"
#       Local de armazenamento
directory       "/var/lib/ldap/backend_sj_singlemaster"

#       Parametros especificos do backend hdb
dbconfig set_cachesize 0 2097152 0
dbconfig set_lk_max_objects 1500
dbconfig set_lk_max_locks 1500
dbconfig set_lk_max_lockers 1500

#       Indexacao de atributos
index       objectClass eq
index       entryCSN,entryUUID eq
index       uid,uidNumber,gidNumber eq
index       memberUID,mail,givenname eq

#       Alterar os atributos operacionais da entrada
lastmod   off
#       Checagem periodica da base
```

```
26 checkpoint     512 30
27
28 #       REPLICACAO - Configuracao do comsumidor
29 syncrepl rid=1
30         provider=ldap://reitoria.ifsc.edu.br:389
31         type=refreshOnly
32         interval=00:00:15:00
33         retry="60 +"
34         searchbase="dc=ifsc,dc=edu,dc=br"
35         filter="(objectClass=*)"
36         scope=sub
37         attrs="*"
38         bindmethod=simple
39         binddn="cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br"
40         credentials=leitor
41         schemachecking=on
42         starttls=yes
43         tls_cacert=/etc/ldap/tls/reitoria_CA.pem
44         tls_cert=/etc/ldap/tls/certificado_SRV.pem
45         tls_key=/etc/ldap/tls/chave_Privada.pem
46         tls_reqcert=demand
47
48 #       Endereco do Provedor
49 updateref   ldap://reitoria.ifsc.edu.br
50 # Arquivo contendo as ACLs
51 include        /etc/ldap/acls_consumidor.conf
```

Listagem B.5: Configurações de *backend* do arquivo `slapd.conf` para o consumidor na replicação *Single Master*.

Nas configurações das (*Access Control List* – ACL)s para o consumidor, foram utilizadas as mesmas regras ilustradas na Figura B.3. É importante observar que as regras ilustradas na Figura B.3, envolvem controles em operações de escrita. Esses controles não são aplicados em um consumidor, afinal, o diretório consumidor atende apenas a operações de leitura.

B.2 Configuração para replicação *Delta*

A configuração do *OpenLDAP* para a replicação *Delta* difere-se da configuração *Single Master* apenas nas diretivas relacionadas à replicação, portanto, a seguir, são ilustradas apenas os trechos das configurações, dos quais devem ser alterados dos arquivos, ilustrados na sessão B.1.

B.2.1 Configuração do provedor

Nas configurações globais do arquivo `slapd.conf` para o provedor, utiliza-se a mesma configuração apresentada na Figura B.1, adicionando apenas a seguinte diretiva:

```
moduleload accesslog
```

Listagem B.6: Diretiva adicional do arquivo `slapd.conf` para configuração do provedor *Delta*.

Para o *backend* otimizado para replicação *Delta*, a Figura B.7 ilustra as configurações do `slapd.conf`.

```
#       ::: backend do diretorio Delta:::

#       ACL para acesso aos logs da replicacao delta
access to *
        by dn.base="cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br" read
        by * break

#       Backend Accesslog (para replicacao delta)
database        hdb
suffix cn=accesslog
directory /var/lib/ldap/backend_reit_accesslog
rootdn cn=accesslog
index default eq
index entryCSN,objectClass,reqEnd,reqResult,reqStart

#       Ativa modo provedor de replicacao (para o backend accesslog)
overlay syncprov
syncprov-nopresent TRUE
syncprov-reloadhint TRUE
limits dn.exact="cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br"
 time.soft=unlimited time.hard=unlimited size.soft=unlimited size.hard=unlimited

database        hdb
#       sufixo
suffix          "dc=ifsc,dc=edu,dc=br"
#       Administrador do servico
rootdn          "cn=admin,dc=ifsc,dc=edu,dc=br"
rootpw {SSHA}NxruY48AQTXLq2EeG+s54EmR1Cugp1Su
#       Local de armazenamento dos dados
directory       "/var/lib/ldap/backend_reit_delta"

#       Parametros especificos do backend hdb
dbconfig set_cachesize 0 2097152 0
dbconfig set_lk_max_objects 1500
dbconfig set_lk_max_locks 1500
dbconfig set_lk_max_lockers 1500

#       Indexacao de atributos
```

```
index       objectClass eq
index       entryCSN,entryUUID eq
index       uid,uidNumber,gidNumber eq
index       memberUID,mail,givenname eq

#       Alterar os atributos operacionais da entrada
lastmod   on
#       Checagem periodica da base
checkpoint     512 30

#       Ativa modo provedor de replicacao
overlay syncprov
syncprov-checkpoint 100 10
syncprov-sessionlog 100

#       Parametros do log de replicacao delta
overlay accesslog
logdb cn=accesslog
logops writes
logsuccess TRUE
logpurge 07+00:00 01+00:00
limits dn.exact="cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br"
 time.soft=unlimited time.hard=unlimited size.soft=unlimited size.hard=unlimited

# Arquivo contendo as ACLs
include /etc/ldap/acls_provedor.conf
```

Listagem B.7: Configurações do *backend* do arquivo `slapd.conf` para o provedor na replicação *Delta*.

B.2.2 Configuração do consumidor

```
#       Suporte a versao antiga do protocolo
allow bind_v2
#       Local dos esquemas utilizados pelo servidor
include        /etc/ldap/schema/core.schema
include        /etc/ldap/schema/cosine.schema
include        /etc/ldap/schema/nis.schema
include        /etc/ldap/schema/inetorgperson.schema
include        /etc/ldap/schema/brEduPerson.schema
include        /etc/ldap/schema/eduperson.schema
include        /etc/ldap/schema/schac.schema
include        /etc/ldap/schema/samba.schema
#       Local e Modulos carregados dinamicamente
modulepath            /usr/lib/ldap
moduleload            back_ldap
moduleload            back_hdb
moduleload            back_monitor

#       Local do PID do processo.
pidfile        /var/run/slapd/slapd.pid
```

```
#       Local dos parametros de execucao.
argsfile        /var/run/slapd/slapd.args
#       Nivel de informacoes que devem ser armazenadas no arquivo de log.
loglevel        256 16384
#       Limite de entradas retornadas por uma consulta
sizelimit unlimited
#       Tempo maximo para o cliente obter respostas do servidor
timelimit unlimited
#       Quantidades de processos do servico
tool-threads 2
#       Tempo para anular uma conexao perdida
idletimeout 60

#       Modulo responsavel por seguir referencias (referrals)
overlay         chain
chain-uri       ldap://reitoria.ifsc.edu.br
chain-idassert-bind   bindmethod=simple
        binddn="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br"
        credentials="admin"
        mode=self
chain-return-error true
chain-rebind-as-user true

#       Nega conexoes anonimas
disallow bind_anon
require bind
#       Configuracoes de certificado. (TLS)
TLSCertificateFile /etc/ldap/tls/certificado_SRV.pem
TLSCertificateKeyFile /etc/ldap/tls/chave_Privada.pem
TLSCACertificateFile /etc/ldap/tls/saojose_CA.pem
TLSVerifyClient never

#       Determina o minimo de seguranca aplicada nas conexoes
#security ssf=1 update_ssf=112 simple_bind=64

#       Backend MONINTOR. Para armazenamento de estatisticas do servico
database monitor
access to dn.subtree="cn=Monitor"
        by dn.exact="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br" read
        by * none

#       O arquivo abaixo continua com as configuracoes de backend para o diretorio
include         /etc/ldap/backend_sj_delta.conf
```

Listagem B.8: Configurações globais do arquivo `slapd.conf` para um consumidor na replicação *Delta*.

```
#       ::: backend do diretorio :::
database hdb

#       sufixo
suffix          "dc=ifsc,dc=edu,dc=br"
#       Administrador do servico
```

```
7  rootdn          "cn=admin,dc=ifsc,dc=edu,dc=br"
8  #       Local de armazenamento
9  directory       "/var/lib/ldap/backend_sj_delta"
10
11 #       Parametros especificos do backend hdb
12 dbconfig set_cachesize 0 2097152 0
13 dbconfig set_lk_max_objects 1500
14 dbconfig set_lk_max_locks 1500
15 dbconfig set_lk_max_lockers 1500
16
17 #       Indexacao de atributos
18 index           objectClass eq
19 index           entryCSN,entryUUID eq
20 index           uid,uidNumber,gidNumber eq
21 index           memberUID,mail,givenname eq
22
23 #       Alterar os atributos operacionais da entrada
24 lastmod   off
25 #       Checagem periodica da base
26 checkpoint     512 30
27
28 #       REPLICACAO - Configuracao do comsumidor
29 syncrepl rid=1
30         provider=ldap://reitoria.ifsc.edu.br:389
31         type=refreshOnly
32         interval=00:00:05:00
33         retry="60 +"
34         searchbase="dc=ifsc,dc=edu,dc=br"
35         filter="(objectClass=*)"
36         scope=sub
37         attrs="*"
38         bindmethod=simple
39         binddn="cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br"
40         credentials=leitor
41         logbase="cn=accesslog"
42         logfilter="(&(objectClass=auditWriteObject)(reqResult=0))"
43         schemachecking=on
44         syncdata=accesslog
45         starttls=yes
46         tls_cert=/etc/ldap/tls/certificado_SRV.pem
47         tls_key=/etc/ldap/tls/chave_Privada.pem
48         tls_cacert=/etc/ldap/tls/florianopolis.pem
49         tls_reqcert=demand
50
51 #       Endereco do Provedor
52 updateref   ldap://reitoria.ifsc.edu.br
53 # Arquivo contendo as ACLs
54 include         /etc/ldap/acls_consumidor.conf
55
```

Listagem B.9: Configurações do *backend* do arquivo `slapd.conf` para um consumidor na replicação *Delta*.

ANEXO C – Configuração para replicação Multi Master.

O anexo C apresenta as configurações de um único provedor (representado pelo servidor de São José) para replicação *Multi Master*. As configurações dos demais provedores são idênticas, diferenciando-se apenas em questões de nomes de domínio e nomenclatura de arquivos. As Figuras C.1 e C.2ilustram as configurações do serviço. As configurações de ACLs, são semelhantes às ilustradas pela Figura B.3

```
#  :: Configuracoes Globais :::

#       Suporte a versao antiga do protocolo
allow bind_v2

#       Local dos esquemas utilizados pelo servidor
include         /etc/ldap/schema/core.schema
include         /etc/ldap/schema/cosine.schema
include         /etc/ldap/schema/nis.schema
include         /etc/ldap/schema/inetorgperson.schema
include         /etc/ldap/schema/brEduPerson.schema
include         /etc/ldap/schema/eduperson.schema
include         /etc/ldap/schema/schac.schema

#       Local e modulos carregados dinamicamente
modulepath            /usr/lib/ldap
moduleload            back_hdb
moduleload            syncprov
moduleload            back_monitor

#       Local do PID do processo.
pidfile         /var/run/slapd/slapd.pid
#       Local dos parametros de execucao.
argsfile        /var/run/slapd/slapd.args
#       Nivel de informacoes que devem ser armazenadas no arquivo de log.
loglevel        256 16384
#       Limite de entradas retornadas por uma consulta
sizelimit unlimited
#       Tempo maximo para o cliente obter respostas do servidor
timelimit unlimited
#       Quantidades de processos do servico
```

```
32 tool-threads 2
33 #       Tempo para anular uma conexao perdida
34 idletimeout 60
35
36 #       Nega conexoes anonimas
37 disallow bind_anon
38 require bind
39
40 #       Configuracoes de certificado. (TLS)
41 TLSCertificateFile /etc/ldap/tls/certificado_SRV.pem
42 TLSCertificateKeyFile /etc/ldap/tls/chave_Privada.pem
43 TLSCACertificateFile /etc/ldap/tls/saojose_CA.pem
44 TLSVerifyClient never
45
46 #       Determina o minimo de seguranca aplicada nas conexoes
47 #security ssf=1 update_ssf=112 simple_bind=64
48
49 #       Backend MONINTOR. Para armazenamento de estatisticas do servico
50 database monitor
51 access to dn.subtree="cn=Monitor"
52         by dn.exact="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br" read
53         by * none
54
55 #       O arquivo abaixo continua com as configuracoes de backend para o diretorio
56 include         /etc/ldap/backend_sj_multimaster.conf
57
```

Listagem C.1: Configurações globais do arquivo `slapd.conf` para replicação *Multi Master.*

```
1  #       ::: backend do diretorio :::
2  #       identificador de provedor
3  serverID 003
4
5  database hdb
6  #       sufixo
7  suffix          "dc=ifsc,dc=edu,dc=br"
8  #       Administrador do servico
9  rootdn          "cn=admin,dc=ifsc,dc=edu,dc=br"
10 rootpw      {SSHA}weVg5twqZXuE4yT2e/lr/krA44Dp/wiy
11 #       Local de armazenamento
12 directory       "/var/lib/ldap/backend_sj_multimaster"
13
14 #       Parametros especificos do backend hdb
15 dbconfig set_cachesize 0 2097152 0
16 dbconfig set_lk_max_objects 1500
17 dbconfig set_lk_max_locks 1500
18 dbconfig set_lk_max_lockers 1500
19
20 #       Indexacao de atributos
21 index           objectClass eq
22 index           entryCSN,entryUUID eq
23 index           uid,uidNumber,gidNumber eq
24 index           memberUID,mail,givenname eq
```

```
25 #       Alterar os atributos operacionais da entrada
26 lastmod   on
27 #       Checagem periodica da base
28 checkpoint     512 30
29
30 #       Servidor como comsumidor para cada provedor envolvido
31 syncrepl rid=000
32         provider=ldap://florianopolis.ifsc.edu.br:389
33         type=refreshAndPersist
34         retry="5 5 300 +"
35         searchbase="dc=ifsc,dc=edu,dc=br"
36         attrs="*,+"
37         bindmethod=simple
38         binddn="cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br"
39         credentials=leitor
40         starttls=yes
41         tls_cert=/etc/ldap/tls/certificado_SRV.pem
42         tls_key=/etc/ldap/tls/chave_Privada.pem
43         tls_cacert=/etc/ldap/tls/reitoria_CA.pem
44         tls_reqcert=demand
45
46 syncrepl rid=001
47         provider=ldap://reitoria.ifsc.edu.br:389
48         type=refreshAndPersist
49         retry="5 5 300 +"
50         searchbase="dc=ifsc,dc=edu,dc=br"
51         attrs="*,+"
52         bindmethod=simple
53         binddn="cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br"
54         credentials=leitor
55         starttls=yes
56         tls_cert=/etc/ldap/tls/certificado_SRV.pem
57         tls_key=/etc/ldap/tls/chave_Privada.pem
58         tls_cacert=/etc/ldap/tls/reitoria_CA.pem
59         tls_reqcert=demand
60
61 #      Diretiva obrigatoria para replicacao entre provedores
62 mirrormode TRUE
63 #       Ativa o diretorio como provedor
64 overlay syncprov
65 syncprov-checkpoint 100 10
66
67 # Arquivo contendo as ACLs
68 include /etc/ldap/acls_provedor.conf
```

Listagem C.2: Configurações do *backend* e replicação do arquivo `slapd.conf` para um servidor de diretórios na replicação *Multi Master*

ANEXO D – Arquivos utilizados nos experimentos

D.1 Representação LDIF da esturtura básica do diretório.

```
#  Raiz
dn: dc=ifsc,dc=edu,dc=br
objectClass: dcObject
objectClass: organization
o: ifsc
dc: ifsc

# Ramo People
dn: ou=people,dc=ifsc,dc=edu,dc=br
objectClass: organizationalUnit
ou: people

#  Ramo Group
dn: ou=group,dc=ifsc,dc=edu,dc=br
objectClass: organizationalUnit
ou: group

# Ramo Machine
dn: ou=machine,dc=ifsc,dc=edu,dc=br
objectClass: organizationalUnit
ou: machine

# Ramo Admin
dn: ou=admin,dc=ifsc,dc=edu,dc=br
objectClass: organizationalUnit
ou: admin

#  Ramo Leitor
dn: ou=leitor,dc=ifsc,dc=edu,dc=br
objectClass: organizationalUnit
ou: leitor

#  Administradores
dn: cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin-sj
userPassword: {SSHA}n5wBs8l3J4l54632Bd2u3fQ4fg2pNHZo
```

```
39
40 dn: cn=admin-fln,ou=admin,dc=ifsc,dc=edu,dc=br
41 objectClass: simpleSecurityObject
42 objectClass: organizationalRole
43 cn: admin-fln
44 userPassword: {SSHA}n5wBs8l3J4l54632Bd2u3fQ4fg2pNHZo
45
46 dn: cn=admin-reit,ou=admin,dc=ifsc,dc=edu,dc=br
47 objectClass: simpleSecurityObject
48 objectClass: organizationalRole
49 cn: admin-reit
50 userPassword: {SSHA}n5wBs8l3J4l54632Bd2u3fQ4fg2pNHZo
51
52 #  Leitores
53 dn: cn=leitor-linux,ou=leitor,dc=ifsc,dc=edu,dc=br
54 objectClass: simpleSecurityObject
55 objectClass: organizationalRole
56 cn: leitor-linux
57 userPassword: {SSHA}toYzwWXBCo+1/GYFSfoJTU3w7vBnDm2W
58
59 dn: cn=leitor-cafe,ou=leitor,dc=ifsc,dc=edu,dc=br
60 objectClass: simpleSecurityObject
61 objectClass: organizationalRole
62 cn: leitor-cafe
63 userPassword: {SSHA}toYzwWXBCo+1/GYFSfoJTU3w7vBnDm2W
64
65 dn: cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br
66 objectClass: simpleSecurityObject
67 objectClass: organizationalRole
68 cn: leitor-sync
69 userPassword: {SSHA}toYzwWXBCo+1/GYFSfoJTU3w7vBnDm2W
70
71 # Grupo Admins
72 dn: cn=admin,ou=group,dc=ifsc,dc=edu,dc=br
73 objectClass: groupOfNames
74 cn: admin
75 member: cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br
76 member: cn=admin-fln,ou=admin,dc=ifsc,dc=edu,dc=br
77 member: cn=admin-reit,ou=admin,dc=ifsc,dc=edu,dc=br
78
79 #  Grupo Leitor
80 dn: cn=leitor,ou=group,dc=ifsc,dc=edu,dc=br
81 objectClass: groupOfNames
82 cn: leitor
83 member: cn=leitor-linux,ou=leitor,dc=ifsc,dc=edu,dc=br
84 member: cn=leitor-sync,ou=leitor,dc=ifsc,dc=edu,dc=br
85 member: cn=leitor-cafe,ou=leitor,dc=ifsc,dc=edu,dc=br
```

Listagem D.1: Representação LDIF da esturtura básica do diretório.

D.2 Representação LDIF de uma entrada de usuário.

```
dn: uid=user-1,ou=people,dc=ifsc,dc=edu,dc=br
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
objectClass: brPerson
objectClass: schacPersonalCharacteristics
objectClass: eduPerson
uid: user-1
cn: user-1
sn: user-1
eduPersonPrincipalname: user-1
schacCountryofCitizenship: Brasil
schacDateOfBirth: 00000000
schacGender: 1
brcpf: 00000000000
brpassport: 000000
telephoneNumber: 00 00000000
mail: user-1@ifsc.edu.br
uidNumber: 1000
gidNumber: 1000
homedirectory: /home/user-1
loginshell: /bin/bash
userPassword: {SSHA}iIcfwlYqh5A1S3A+7X1iQsPY8ooNEFRc

dn: braff=1,uid=user-1,ou=people,dc=ifsc,dc=edu,dc=br
objectClass: brEduPerson
braff: 1
braffType: estudante
brentr: 00000000
brexit: 00000000

dn: cn=user-1,ou=group,dc=ifsc,dc=edu,dc=br
objectclass: posixGroup
cn: user-1
gidNumber: 1000
memberUid: user-1
```

Listagem D.2: Representação LDIF de uma entrada de usuário.

D.3 Representação LDIF para alterar um atributo da entrada de usuário.

```
dn: uid=user-1,ou=people,dc=ifsc,dc=edu,dc=br
replace: schacGender
schacGender: 1
```

Listagem D.3: Representação LDIF para alterar um atributo da entrada de usuário

D.4 *Script shell* para gerar 100 entradas de usuários e inserí-las no diretório.

```
#! /bin/bash

LDAP_HOST="localhost"
LDAP_PORT="389"
LDAP_DN="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br"
LDAP_PWD="admin"
USERS_FILE="/etc/ldap/LDIFs/adiciona_100.ldif"
UID_NUMBER="1000"
ID_NAME="user-"

>$USERS_FILE

I=1
while [ "$I" -le "100" ] ; do

 echo "dn: uid="$ID_NAME$I",ou=people,dc=ifsc,dc=edu,dc=br" >> $USERS_FILE
 echo "objectClass: inetOrgPerson" >> $USERS_FILE
 echo "objectClass: posixAccount" >> $USERS_FILE
 echo "objectClass: shadowAccount" >> $USERS_FILE
 echo "objectClass: brPerson" >> $USERS_FILE
 echo "objectClass: schacPersonalCharacteristics" >> $USERS_FILE
 echo "objectClass: eduPerson" >> $USERS_FILE
 echo "uid: $ID_NAME$I" >> $USERS_FILE
 echo "cn: $ID_NAME$I" >> $USERS_FILE
 echo "sn: $ID_NAME$I" >> $USERS_FILE
 echo "eduPersonPrincipalname: "$ID_NAME$I" " >> $USERS_FILE
 echo "schacCountryofCitizenship: Brasil" >> $USERS_FILE
 echo "schacDateOfBirth: 00000000" >> $USERS_FILE
 echo "schacGender: 1" >> $USERS_FILE
 echo "brcpf: 00000000000" >> $USERS_FILE
 echo "brpassport: 000000" >> $USERS_FILE
 echo "telephoneNumber: 00 00000000" >> $USERS_FILE
 echo "mail: $ID_NAME$I@ifsc.edu.br" >> $USERS_FILE
 echo "uidNumber: $UID_NUMBER" >> $USERS_FILE
 echo "gidNumber: $UID_NUMBER" >> $USERS_FILE
 echo "homedirectory: /home/$ID_NAME$I" >> $USERS_FILE
 echo "loginshell: /bin/bash" >> $USERS_FILE
 USER_PWD=‘slappasswd -s $ID_NAME$I‘
 echo "userPassword: $USER_PWD" >> $USERS_FILE
 echo "" >> $USERS_FILE

 echo "dn: braff=1,uid="$ID_NAME$I",ou=people,dc=ifsc,dc=edu,dc=br" >> $USERS_FILE
 echo "objectClass: brEduPerson" >> $USERS_FILE
 echo "braff: 1" >> $USERS_FILE
 echo "braffType: estudante" >> $USERS_FILE
 echo "brentr: 00000000" >> $USERS_FILE
 echo "brexit: 00000000" >> $USERS_FILE
 echo "" >> $USERS_FILE

 echo "dn: cn=$ID_NAME$I,ou=group,dc=ifsc,dc=edu,dc=br" >> $USERS_FILE
```

```
51  echo "objectclass: posixGroup" >> $USERS_FILE
52  echo "cn: $ID_NAME$I" >> $USERS_FILE
53  echo "gidNumber: $UID_NUMBER" >> $USERS_FILE
54  echo "memberUid: $ID_NAME$I" >> $USERS_FILE
55  echo "" >> $USERS_FILE
56
57  I=‘expr $I + 1‘
58  UID_NUMBER=‘expr $UID_NUMBER + 1‘
59 done
60
61 ldapadd -h $LDAP_HOST -p $LDAP_PORT -x -D $LDAP_DN -w $LDAP_PWD -f $USERS_FILE
62
63 rm $USERS_FILE
```

Listagem D.4: *Script shell* para gerar 100 entradas de usuários e inserí-las no diretório

D.5 *Script shell* para gerar um arquivo LDIF com as alterações nas 100 entradas de usuários.

```
1  #! /bin/bash
2
3  LDAP_HOST="localhost"
4  LDAP_PORT="389"
5  LDAP_DN="cn=admin-sj,ou=admin,dc=ifsc,dc=edu,dc=br"
6  LDAP_PWD="admin"
7  USERS_FILE="/etc/ldap/LDIFs/modifica_100.ldif"
8  ID_NAME="user-"
9
10 >$USERS_FILE
11
12 I=1
13
14 while [ "$I" -le "100" ] ; do
15
16  echo "dn: uid="$ID_NAME$I",ou=people,dc=ifsc,dc=edu,dc=br" >> $USERS_FILE
17  echo "replace: schacGender" >> $USERS_FILE
18  echo "schacGender: 0" >> $USERS_FILE
19  echo "" >> $USERS_FILE
20
21  I=‘expr $I + 1‘
22
23  ldapmodify -h $LDAP_HOST -p $LDAP_PORT -x -D $LDAP_DN -w $LDAP_PWD -f $USERS_FILE
24 done
25
26 rm $USERS_FILE
```

Listagem D.5: *Script shell* para alterar um atributo de cada uma das 100 entradas de usuários.

Lista de Abreviaturas

ACL *Access Control List*

CAFe Comunidade Acadêmica Federada

CCITT *International Telegraph and Telephone Consultative Committee*

cn *Common Name*

CSN *Change Sequence Number*

CSV *Comma-Separated Values*

DAP *Directory Access Protocol*

DIT *Directory Information Tree*

DN *Distinguished Name*

DNS *Domain Name System*

FQDN *Fully Qualifield Domain Name*

IFSC Instituto Federal de Santa Catarina

ISO *International Organization for Standardization*

LDAP *Lightweight Directory Access Protocol*

LDIF *LDAP Data Interchange Format*

PAM *Pluggable Authentication Modules*

RDN *Relative Distinguished Name*

RNP Rede Nacional de Pesquisa

SASL *Simple Authentication and Security Layer*

SGBD Sistema Gerenciador de Banco de Dados

SQL *Structured Query Language*

SSHA *Salted Secure Hash Algorithm*

SSL *Secure Sockets Layer*

TLS *Transport Layer Security*

WAN *Wide Area Network*

Referências Bibliográficas

ALBITZ, P.; LIU, C. *DNS and BIND*. 5. ed. [S.l.]: O'Reilly, 2006.

ARKILLS, B. *LDAP Directories Explained: An Introduction and Analysis*. [S.l.]: Addison Wesley, 2003.

BUTCHER, M. *Mastering OpenLDAP - Configuring, Securing, and Integrating Directory Services*. 1. ed. [S.l.]: Packt Publishing, 2007.

CHOI, J.; ZEILENGA, K. *The Lightweight Directory Access Protocol (LDAP) Content Synchronization Operation*. [S.l.], 2006. Disponível em: <http://www.ietf.org/rfc/rfc4533>.

DIERKS, T.; ALLEN, C. *The TLS Protocol - Version 1.0*. [S.l.], 1999. Disponível em: <http://www.ietf.org/rfc/rfc2246.txt>.

GOOD, G. *The LDAP Data Interchange Format (LDIF) - Technical Specification*. [S.l.], 2000. Disponível em: <http://www.ietf.org/rfc/rfc2849.txt>.

GROFF, J. R.; WEINBERG, P. N. *SQL: The Complete Reference, Second Edition*. 2. ed. [S.l.]: McGraw-Hill/Osborne, 2002.

GUTIERREZ, C. M.; GALLAGHER, P. *Secure Hash Standard (SHS)*. [S.l.], 2008.

HARRISON, R. *LDAP: Authentication Methods and Security Mechanisms*. [S.l.], 2006. Disponível em: <http://tools.ietf.org/id/draft-ietf-ldapbis-authmeth-19.txt>.

HODGES, J.; MORGAN, R.; WAHL, M. *Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security*. [S.l.], 2000. Disponível em: <http://www.ietf.org/rfc/rfc2830.txt>.

HOUSLEY, R. et al. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. [S.l.], 1999. Disponível em: <http://www.ietf.org/rfc/rfc2459.txt>.

HOWES, T.; SMITH, M. *The LDAP URL Format*. [S.l.], 1997. Disponível em: <http://www.ietf.org/rfc/rfc2255>.

HOWES, T. A.; SMITH, M. C.; GOOD, G. S. *Understanding and Deploying LDAP Directory Services*. [S.l.]: Addison Wesley, 2003.

ITU-T. *The Directory: Protocol specifications*. [S.l.], 2005. Disponível em: <http://www.itu.int/rec/T-REC-X.511-200508-I>.

LEGG, S. *Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules*. [S.l.], 2006. Disponível em: <http://www.ietf.org/rfc/rfc4517.txt>.

MELNIKOV, A. *"The Kerberos V5 GSSAPI Simple Authentication and Security Layer (SASL) Mechanism"*. [S.l.], 2006. Disponível em: <http://www.ietf.org/rfc/rfc4752.txt>.

MOREIRA, E. Q. et al. *Federação CAFe: Implantação do Provedor de Identidade*. [S.l.]: Escola Superior de Redes RNP, 2010.

MYERS, J. *Simple Authentication and Security Layer (SASL)*. [S.l.], 1997. Disponível em: <http://www.ietf.org/rfc/rfc2222.txt>.

OPENLDAP. *OpenLDAP Software 2.4 Administrator's Guide*. [S.l.], 2010. Disponível em: <http://www.openldap.org/doc/admin24/guide.html>.

SERMERSHEIM, J. *Lightweight Directory Access Protocol (LDAP): The Protocol*. [S.l.], 2006. Disponível em: <http://www.ietf.org/rfc/rfc4511.txt>.

SHAFRANOVICH, Y. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. [S.l.], 2005. Disponível em: <http://www.ietf.org/rfc/rfc4180.txt>.

SHELDON, R. *SQL: A Beginner's Guide*. 2. ed. [S.l.]: McGraw-Hill/Osborne, 2003.

SOCOLOFSKY, T.; KALE, C. *A TCP/IP Tutorial*. [S.l.], 1991. Disponível em: <http://www.ietf.org/rfc/rfc1180.txt>.

ZEILENGA, K. *LDAP Password Modify Extended Operation*. [S.l.], 2001. Disponível em: <http://www.ietf.org/rfc/rfc3062.txt>.

ZEILENGA, K. *Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names*. [S.l.], 2006. Disponível em: <http://www.ietf.org/rfc/rfc4514.txt>.

ZEILENGA, K. *Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map*. [S.l.], 2006. Disponível em: <http://www.ietf.org/rfc/rfc4510.txt>.

ZEILENGA, K. D. *LDAP Multi-master Replication Considered Harmful*. [S.l.], 2004. Disponível em: <http://www.watersprings.org/pub/id/draft-zeilenga-ldup-harmful-02.txt>.