



Red Hat Directory Server 12

调整红帽目录服务器的性能

提高服务器和数据库性能

提高服务器和数据库性能

法律通告

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

摘要

要提高目录服务器性能，您可以调整锁定、资源限值和事务日志的数量。如果系统中的磁盘空间不足，您还可以监控本地磁盘并关闭 Directory 服务器。

目录

对红帽文档提供反馈	4
第 1 章 监控数据库活动	5
1.1. 使用命令行监控数据库活动	5
1.2. 使用 WEB 控制台监控数据库活动	5
1.3. 数据库监控属性	5
第 2 章 提高视图的性能	8
2.1. 使用命令行创建索引以提高视图的性能	8
2.2. 使用 WEB 控制台创建索引以提高视图的性能	10
第 3 章 设置索引扫描限制以便在载入长 ID 列表时提高性能	13
3.1. 使用命令行设置全局索引扫描限制	13
3.2. 使用 WEB 控制台设置全局索引扫描限制	14
3.3. 使用命令行将索引扫描限制设置为数据库	14
第 4 章 调优锁定数量	17
4.1. 通过监控空闲数据库锁定来避免数据崩溃	17
4.2. 手动监控锁定数量	18
4.3. 使用命令行设置锁定数	18
4.4. 使用 WEB 控制台设置锁定数	19
第 5 章 设置目录服务器线程数量	21
5.1. 使用命令行启用自动线程调整	21
5.2. 使用 WEB 控制台启用自动线程调整	22
5.3. 使用命令行手动设置线程数量	23
5.4. 使用 WEB 控制台手动设置线程数量	23
第 6 章 调优资源限制	25
6.1. 使用命令行更新资源限制设置	25
6.2. 使用 WEB 控制台更新资源限值设置	26
6.3. 禁用 TRANSPARENT HUGE PAGES 功能	28
第 7 章 每个搜索操作日志记录统计信息	30
第 8 章 监控本地磁盘以在低磁盘空间中关闭 DIRECTORY 服务器	32
8.1. DIRECTORY 服务器的行为取决于可用磁盘空间的数量	32
8.2. 使用命令行配置本地磁盘监控	32
8.3. 使用 WEB 控制台配置本地磁盘监控	33
第 9 章 调整事务日志记录	35
9.1. 使用命令行更改数据库检查点间隔	35
9.2. 使用 WEB 控制台更改数据库检查点间隔	35
9.3. 禁用持久事务	36
第 10 章 管理缓存设置	38
10.1. CACHE-AUTOSIZE 和 CACHE-AUTOSIZE-SPLIT 参数如何影响数据库和条目缓存大小	38
10.2. 所需的缓存大小	40
10.3. 使用命令行设置数据库缓存大小	42
10.4. 使用 WEB 控制台设置数据库缓存大小	42
10.5. 使用命令行设置 DN 缓存大小	43
10.6. 使用 WEB 控制台设置 DN 缓存大小	44
10.7. 使用命令行设置条目缓存大小	45
10.8. 使用 WEB 控制台设置条目缓存大小	46

第 11 章 提高导入性能	48
11.1. 使用大型数据库导入和导入的大型数据库导入和导入，使用大型属性值调整目录服务器	48
第 12 章 调优服务器连接管理	49
12.1. 使用命令行管理连接监听程序线程数量	49

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。请让我们了解如何改进文档。要做到这一点：

- 要通过 JIRA 提交反馈（需要帐户）：
 1. 登录到 [Jira](#) 网站。
 2. 在顶部导航栏中点 **Create**
 3. 在 **Summary** 字段中输入描述性标题。
 4. 在 **Description** 字段中输入您对改进的建议。包括到文档相关部分的链接。
 5. 点对话框底部的 **Create**。
- 要通过 Bugzilla 提交反馈（需要帐户）：
 1. 进入 [Bugzilla](#) 网站。
 2. 在 Component 中选择 **Documentation**。
 3. 在 **Description** 中输入您要提供的信息。包括文档相关部分的链接。
 4. 点 **Submit Bug**。

第 1 章 监控数据库活动

管理员应监控数据库活动，以确保正确配置缓存等调优设置。

1.1. 使用命令行监控数据库活动

要使用命令行显示监控活动，请显示动态更新的只读属性，该属性存储在 `cn=monitor,cn=monitor,cn=name,cn=ldbm` 数据库, `cn=plugins,cn=config` 中。

流程

- 要显示数据库的当前活动，请输入：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com monitor backend
userRoot
```

此命令显示 `userRoot` 数据库的活动。

其他资源

- [数据库监控属性](#)

1.2. 使用 WEB 控制台监控数据库活动

在 web 控制台中，Directory 服务器显示来自 `cn=monitor,cn=db_name,cn=ldbm` 数据库, `cn=plugins,cn=config` 的从 `cn=monitor,cn=config` 的动态更新只读监控属性的值。

流程

1. 导航到 **Monitoring** → **Database** → **数据库名称**。
2. 在 **条目缓存** 和 **DN 缓存** 标签页中显示缓存值。

其他资源

- [数据库监控属性](#)

1.3. 数据库监控属性

表 1.1. 继承设置

属性	Description
<code>readonly</code>	指明数据库是否处于只读模式(1)还是读写模式 () 。
<code>entrycachehits</code>	成功条目缓存查找的总数。该值是服务器可以从条目缓存检索条目的次数，而不必从数据库重新加载条目。
<code>entrycachetries</code>	开始您启动实例后的条目缓存查找总数。该值为总数，因为实例已启动，Directory 服务器尝试从条目缓存中检索条目。

属性	Description
entrycachehitratio	<p>条目缓存数量会尝试成功条目缓存查找。此数字基于您上次启动实例之后的总查找和命中。接近条目缓存命中率越接近 100%，这越好。</p> <p>每当操作试图找到条目缓存中不存在的条目时，服务器都需要访问数据库来获取条目。因此，随着此比率降至零，磁盘访问数量会增加，目录搜索性能会降低。要提高此比率，增大数据库的条目缓存的大小。</p> <p>要提高此比率，请通过增加 <code>cn= database_name ,cn=database_name,cn=plugins,cn=plugins,cn=config</code> 条目中的 <code>nsslapd-cachememsize</code> 属性的值来增加条目缓存的大小。</p>
currententrycachesize	<p>条目缓存中当前存在的目录条目的总大小（以字节为单位）。</p> <p>要增大缓存中可以存在的条目的大小，请增大 <code>cn=database_name,cn=ldbm database,cn=plugins,cn=config</code> 条目中的 <code>nsslapd-cachememsize</code> 属性的值。</p>
maxentrycachesize	<p>Directory 服务器可在条目缓存中维护的目录条目的最大大小，以字节为单位。</p> <p>要增大缓存中可以存在的条目的大小，请增大 <code>cn=database_name,cn=ldbm database,cn=plugins,cn=config</code> 条目中的 <code>nsslapd-cachememsize</code> 属性的值。</p>
currententrycachecount	<p>存储在给定后端条目缓存中的条目数量。</p>
maxentrycachecount	<p>数据库条目中存储的最大条目数。</p> <p>要调整这个值，请增大 <code>cn=database_name,cn=ldbm database,cn=plugins,cn=config</code> 条目中的 <code>nsslapd-cachesize</code> 属性的值。</p>
dncachehits	<p>服务器通过从 DN 缓存获取规范化名称(DN)来处理请求的次数，而不是再次规范化。</p>
dncachetries	<p>启动了该实例后的 DN 缓存访问总数。</p>
dncachehitratio	<p>缓存尝试成功进行 DN 缓存命中率。接近这个数值比 100%，越好。</p>
currentdncachesize	<p>DN 缓存中当前存在的 DN 的总大小（以字节为单位）。</p> <p>要增大 DN 缓存中可以存在的条目的大小，增大 <code>cn=database_name,cn=ldbm database,cn=plugins,cn=config</code> 条目中的 <code>nsslapd-dncachememsize</code> 属性的值。</p>

属性	Description
maxdnccachesize	Directory 服务器可在 DN 缓存中维护的 DN 的最大大小（以字节为单位）。 要增大缓存中可以存在的条目的大小，增大 cn=database_name,cn=ldbm database,cn=plugins,cn=config 条目中的 nsslapd-dnccachememsize 属性的值。
currentdnccachecount	当前存在于 DN 缓存中的 DN 数。
maxdnccachecount	DN 缓存中允许的最大 DN 数。

第 2 章 提高视图的性能

基于视图的层次结构的性能取决于层次结构本身的结构以及目录树(DIT)中的条目数量。

一般来说，如果您使用虚拟 DIT 视图，性能可能会有细微变化（在标准 DIT 上具有同等搜索的几分点）。如果您没有在搜索中调用视图，则不会产生性能影响。根据预期的搜索模式测试虚拟 DIT 视图，并在部署前加载。

如果您打算将视图用作组织中通用导航工具，红帽建议在查看过滤器中使用属性。

另外，您可以配置一个虚拟列表视图(VLV)索引，以便在视图中评估子过滤器。

不需要专门调优目录的任何其他部分，以供视图使用。

2.1. 使用命令行创建索引以提高视图的性能

视图是基于给定过滤器的搜索结果衍生。过滤器的一部分是 `nsViewFilter` 中明确给定的属性；过滤器的其余部分基于条目层次结构，查找在视图中包括的实际条目的 `entryid` 和 `parentid` 操作属性。

```
!(parentid=search_base_id)(entryid=search_base_id)
```

如果搜索的任何属性 - `entryid`、`parentid` 或 `nsViewFilter` - 中的属性没有索引，那么搜索就会成为部分未索引，并且目录服务器会搜索整个目录树中的匹配条目。

要提高查看性能，请按如下所示创建索引：

- 为 `entryid` 创建 *相等的索引* (`eq`)。 `parentid` 属性默认在系统索引中索引。
- 如果 `nsViewFilter` 测试存在(`attribute=*`)中的过滤器，为正在测试的属性创建 *presence index* (`pres`)。您应该只在目录条目中以次要方式显示的属性使用此索引类型。
- 如果在 `nsViewFilter` 测试相等度(`attribute=value`)中的过滤器，为正在测试的属性创建 *相等的索引* (`eq`)。
- 如果 `nsViewFilter` 中的过滤器测试子字符串(`attribute=value*`)，为要测试的属性创建 *子字符串索引* (子)。
- 如果 `nsViewFilter` 测试 approximation 的过滤器(`属性~value`)，为正在测试的属性创建大约 *索引* (大约)。

例如，当您使用以下视图过滤器时：

```
nsViewFilter: (&(objectClass=inetOrgPerson)(roomNumber=*66))
```

您应该使用相等的 *index 进行索引*（默认为）和带有 *子字符串 index* 的 `roomNumber`。

先决条件

- 您了解您在视图过滤器中使用的属性。

流程

1. 可选：列出后端以确定数据库进行索引：

```
# dsconf -D "cn=Directory Manager" instance_name backend suffix list
dc=example,dc=com (userroot)
```

请注意所选数据库名称（在括号中）。

2. 使用所选后端数据库的 `dsconfig` 实用程序创建索引配置。

指定属性名称、索引类型以及可选的匹配规则来设置 collation order (OID)，特别是在国际化实例时。

```
# dsconf -D "cn=Directory Manager" instance_name backend index add --attr
roomNumber --index-type sub userroot
```

对 `view` 过滤器中使用的每个属性重复此步骤。

3. 重新索引数据库以应用新索引：

```
# dsconf -D "cn=Directory Manager" instance_name backend index reindex userroot
```

验证

1. 使用您在视图中使用的相同过滤器执行基于标准目录树的搜索：

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -b
dc=example,dc=com (&(objectClass=inetOrgPerson)(roomNumber=*66))
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -b
dc=example,dc=com "(&(objectClass=inetOrgPerson)(roomNumber=*66))"
```

2. 查看 `/var/log/dirsrv/slapd-instance_name/ access` 中的访问日志。

您搜索的 `RESULT` 应该不包含详情中的 `note=U` 或 `Partially Unindexed Filter`。

其他资源

- [管理索引](#)

2.2. 使用 WEB 控制台创建索引以提高视图的性能

视图是基于给定过滤器的搜索结果衍生。过滤器的一部分是 `nsViewFilter` 中明确给定的属性；过滤器的其余部分基于条目层次结构，查找在视图中包括的实际条目的 `entryid` 和 `parentid` 操作属性。

```
(!(parentid=search_base_id)(entryid=search_base_id)
```

如果搜索的任何属性 - `entryid`、`parentid` 或 `nsViewFilter` - 中的属性没有索引，那么搜索就会成为部分未索引，并且目录服务器会搜索整个目录树中的匹配条目。

要提高查看性能，请按如下所示创建索引：

- 为 `entryid` 创建 *相等的索引* (`eq`)。 `parentid` 属性默认在系统索引中索引。
- 如果 `nsViewFilter` 测试存在(`attribute=*`)中的过滤器，为正在测试的属性创建 *presence index* (`pres`)。您应该只在目录条目中以次要方式显示的属性使用此索引类型。
- 如果在 `nsViewFilter` 测试相等度(`attribute=value`)中的过滤器，为正在测试的属性创建 *相等的索引* (`eq`)。
- 如果 `nsViewFilter` 中的过滤器测试子字符串(`attribute=value*`)，为要测试的属性创建 *子字符串索引* (`子`)。
- 如果 `nsViewFilter` 测试 *approximation* 的过滤器(属性 `~value`)，为正在测试的属性创建 *大约索引* (`大约`)。

例如，当您使用以下视图过滤器时：

```
nsViewFilter: (&(objectClass=inetOrgPerson)(roomNumber=*66))
```

您应该使用相等的 `index` 进行索引（默认为）和带有 *子字符串 index* 的 `roomNumber`。

先决条件

- 在 **web 控制台** 中登录到实例。
- 您了解您在视图过滤器中使用的属性。

流程

1. 在 **Database** 下，从要为其创建索引的配置树中选择一个后缀。
2. 导航到 **Indexes** 和 **Database Indexes**。
3. 点 **Add Index** 按钮。
4. 键入属性的名称，再选择 属性。
5. 选择应该为此属性创建的 **Index Types**。
6. 另外，还可添加 **匹配规则** 来指定并发顺序(OID)，特别是在国际化的实例时。
7. 选择 在创建后的 **Index** 属性来 重新构建索引。
8. 点 **Create Index**。
9. 对要索引的每个属性重复上述步骤。

验证

- 通过键入添加的属性的名称来过滤索引。
- 新索引属性应显示在结果中。

其他资源

- [管理索引](#)

第 3 章 设置索引扫描限制以便在载入长 ID 列表时提高性能

在大型目录中，搜索结果列表可能非常大。例如，一个带有 `inetorgperson` 属性的条目的目录将以过滤器（如 `(objectclass=inetorgperson)`）返回搜索中的所有条目。

从数据库加载长 ID 列表会显著降低搜索性能。ID 列表扫描限制设置在键被视为与整个主索引匹配前的 ID Directory Server 读取次数的限制。这意味着目录服务器将搜索视为非索引搜索，并设置不同的资源限值。

对于大型索引，实际上，处理与索引搜索匹配的任何搜索效率更高。搜索操作只需要查看整个目录，以处理结果，而不是搜索几乎是目录大小的索引，以及目录本身。

您可以在全局范围内或特定数据库设置索引扫描限制。

3.1. 使用命令行设置全局索引扫描限制

默认情况下，Directory 服务器中的 ID 列表扫描限值为 4000。在大多数情况下，这个值可以为常见的数据库大小和访问模式提供良好的性能，您不需要更改默认值。如果数据库索引比 4000 条目稍大，但仍然比整个目录小得多，增加 ID 列表扫描限制可提高搜索。

另一方面，降低限制可能会显著加快搜索速度，否则点击 4000 条目限制，但不需要扫描每个条目的位置。

流程

1.

更新 ID 列表扫描限制：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --idlistscanlimit=8000
```

这个命令将限制设置为 8000 条目。

2.

重启实例：

```
# dsctl instance_name restart
```

3.2. 使用 WEB 控制台设置全局索引扫描限制

默认情况下，Directory 服务器中的 ID 列表扫描限值为 4000。在大多数情况下，这个值可以为常见的数据库大小和访问模式提供良好的性能，您不需要更改默认值。如果数据库索引比 4000 条目稍大，但仍然比整个目录小得多，增加 ID 列表扫描限制可提高搜索。

另一方面，降低限制可能会显著加快搜索速度，否则点击 4000 条目限制，但不需要扫描每个条目的位置。

流程

1. 导航到 Database → Global Database Configuration。
2. 更新 ID List Scan Limit 字段。
3. 点 Save Config。
4. 点击右上角的 Actions，然后选择 Restart Instance。

3.3. 使用命令行将索引扫描限制设置为数据库

在某些情况下，对于为特定索引定义限制或根本不使用 ID 列表，这很有用。您可以为不同类型的搜索过滤器配置 ID 列表扫描限值。

例如，在一个大型数据库中，有 1,000 万条目包含对象类 inetOrgPerson， (& (objectClass=inetOrgPerson) (uid=user) 首先创建一个 ID 列表，其中包含与 objectClass=inetOrgPerson 匹配的所有 1,000 万 ID。当数据库应用过滤器的第二部分时，它将搜索结果列表以获取与 uid=user 匹配的对象。在这种情况下，对于为特定索引定义限制或根本不使用 ID 列表，这很有用。

此流程演示了如何配置 Directory 服务器来记录 AND 子句中的 objectClass=inetOrgPerson 条件创建 ID 列表。

流程

- 设置 `nsIndexIDListScanLimit` 参数：

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x
dn: cn=objectclass,cn=index,cn=userRoot,cn=ldb database,cn=plugins,cn=config
changetype: modify
replace: nsIndexIDListScanLimit
nsIndexIDListScanLimit: limit=0 type=eq flags=AND values=inetOrgPerson
```

使用这些设置，Directory 服务器不会在 AND 子句中为 `objectClass=inetOrgPerson` 条件创建任何 ID 列表。在所有其他情况下，Directory 服务器应用全局 ID 列表扫描限制值。

`nsIndexIDListScanLimit` 参数使用以下语法：

```
nsIndexIDListScanLimit: limit=NNN [type=eq[,sub,...]] [flags=AND[,XXX,...]]
[values=val[,val,...]]
```

-

limit：设置 ID 列表的最大值。有效值为：

- **-1: unlimited**
- **0**：不使用索引
- **1 到 32 位整数的最大(2147483647)**：最大 ID 数

-

类型：可选：设置更改扫描限制行为的标记。有效值为：

- **AND**：仅应用扫描限制，以搜索 AND 子句中的属性。
- **OR**：应用扫描限制，仅要搜索 OR 子句中显示的属性。

-

值：可选：一个以逗号分隔的值列表，值必须与搜索过滤器匹配，才能应用限制。由于每次都匹配一次匹配，因此如果有任何值匹配，这些值将匹配。

一次只使用一个类型的值。值必须与索引类型和应用索引的属性的语法对应。例如，如果您指定了基于整数的属性 `uidNumber`，并为 `eq` 类型进行索引，则无法使用 `type=eq values=abc`。

如果值包含需要转义的空格、commas、NULL 或其他值，请使用 LDAP 过滤器转义语法：A反斜杠(\)后跟字符的 2 个十六进制代码。在以下示例中，DN 值中的逗号使用 `\2C` 进行转义：

```
nsIndexIDListScanLimit: limit=0 type=eq
values=uid=user\2Cou=People\2Cdc=example\2Cdc=com
```

第 4 章 调优锁定数量

Directory 服务器中的锁定机制控制目录服务器进程可以同时运行多少个副本。例如，在导入作业过程中，Directory 服务器在 `/run/lock/dirsrv/slaped-instance_name/imports/` 目录中设置一个锁定，以防止 ns-slaped Directory Server 进程、另一个导入或导出操作运行。

如果服务器没有可用的锁定，Directory 服务器会在 `/var/log/dirsrv/slaped-instance_name/errors` 文件中记录以下错误：

```
libdb: Lock table is out of available locks
```

但是，Directory 服务器默认设置会尝试防止服务器没有锁定，以避免数据崩溃。详情请参阅 [监控免费数据库锁定的数据崩溃](#)

4.1. 通过监控空闲数据库锁定来避免数据崩溃

退出数据库锁定可能会导致数据崩溃。要避免这种情况，Directory 服务器会监控每 500 毫秒的剩余空闲数据库锁定数量，如果活跃数据库锁定数等于或大于 90%，Directory 服务器会中止所有搜索。

这个过程将间隔改为 600 毫秒，阈值更改为 85%。



注意

如果您设置了太高的间隔，则服务器可在下一次监控检查发生前退出锁定。设置太短的间隔可能会减慢服务器的速度。

流程

1. 设置间隔和阈值：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --locks-monitoring-enabled on --locks-monitoring-pause 600 --locks-monitoring-threshold 85
```

2. 重启实例：

```
# dsctl instance_name restart
```

验证

- 显示锁定监控设置：

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com backend config get |
grep "nsslapd-db-locks-monitoring"
nsslapd-db-locks-monitoring-enabled: on
nsslapd-db-locks-monitoring-threshold: 85
nsslapd-db-locks-monitoring-pause: 600
```

4.2. 手动监控锁定数量

目录服务器跟踪 `nsslapd-db-current-locks` 和 `nsslapd-db-max-locks` 属性中的当前锁定数量，`cn=database,cn=monitor,cn=ldbm database,cn=plugins,cn=config`。

流程

- 要显示锁定数量，请输入：

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -s sub -b
"cn=database,cn=monitor,cn=ldbm database,cn=plugins,cn=config" nsslapd-db-
current-locks nsslapd-db-max-locks
...
nsslapd-db-current-locks: 37
nsslapd-db-max-locks: 39
```

4.3. 使用命令行设置锁定数

使用 `dsconf backend config set` 命令来更新 Directory Server 可以使用的锁定数量。

流程

1. 设置锁定数量：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
locks=20000
```

此命令将锁定数设置为 20000。

2. 重启实例：

```
# dsctl instance_name restart
```

验证

- 显示 `nsslapd-db-locks` 参数的值：

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com backend config get |  
grep "nsslapd-db-locks:"  
nsslapd-db-locks: 20000
```

4.4. 使用 WEB 控制台设置锁定数

您可以在 web 控制台的全局数据库配置中设置锁定目录服务器使用的数量。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 导航到 Database → Global Database Configuration。
2. 单击 Show Advanced Settings。
3. 选择 Enable Monitoring，并输入 threshold 百分比和暂停毫秒。
4. 点 Save Config。
5. 点 Actions → Restart Instance。

验证

1. 导航到 **Database** → **Global Database Configuration**。
2. 单击 **Show Advanced Settings**。
3. 验证锁定监控设置。

第 5 章 设置目录服务器线程数量

用于处理同时连接的线程目录服务器数量会影响服务器的性能。例如，如果所有线程都忙于处理耗时的任务，如 添加操作，则会排队新的传入连接，直到空闲线程可以处理请求。

如果服务器提供少量的 CPU 线程，配置更多线程可以提高性能。但是，在具有大量 CPU 线程的服务器上，设置太高的值不会进一步提高性能。

默认情况下，目录服务器使用自动调整设置来计算线程数量。这个数字基于实例启动时服务器的硬件资源。



警告

避免手动设置线程数量。改为使用 auto-tuning 设置。

启用自动线程调整后，Directory 服务器使用以下优化的线程数量：

CPU 线程号	目录服务器线程号
1-16	16
17-512	目录服务器线程号与系统中的 CPU 线程号匹配。例如，如果您的系统有 24 个 CPU 线程，则目录服务器使用 24 个线程。目录服务器线程的最大数量为 512。
512 及更多	512.目录服务器应用推荐的线程数。

5.1. 使用命令行启用自动线程调整

默认情况下，目录服务器根据可用硬件自动设置线程数量。然而，在某些情况下，您可以使用命令行手动启用此自动调整功能。

流程

- 要启用自动调整功能，请使用以下命令将 `nsslapd-threadnumber` 属性值设为 `-1`：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-threadnumber="-1"
```

```
Successfully replaced "nsslapd-threadnumber"
```

验证

- 使用以下命令验证 Directory 服务器现在使用的 `tread` 数量：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get nsslapd-threadnumber
```

```
nsslapd-threadnumber: 16
```



注意

命令检索目录服务器根据正确的硬件资源计算的线程数量。

其他资源

- [nsslapd-threadnumber 属性描述](#)。

5.2. 使用 WEB 控制台启用自动线程调整

默认情况下，目录服务器根据可用硬件自动设置线程数量。然而，在某些情况下，您可以使用 Web 控制台手动启用此自动调整功能。

先决条件

- 在 web 控制台中登录到实例。如需了解更多详细信息，[请参阅使用 Web 控制台登录到目录服务器](#)。

流程

1. 导航到 **Server** → **Tuning & Limits**。

2. 在 **Number Of Worker Threads** 字段中，将线程数量设置为 **-1**。
3. 点 **Save Settings**。

其他资源

- [nsslapd-threadnumber 属性描述](#)。

5.3. 使用命令行手动设置线程数量

在某些情况下，需要手动设置固定数量的目录服务器线程。例如，如果您不使用 **auto-tuning** 设置并更改虚拟机中的 CPU 内核数，调整 Directory 服务器线程数量可能会提高性能。

如果您之前设置了特定数量的线程，您还可以使用此流程重新启用自动调整设置。

流程

- 设置线程目录服务器应使用的数量：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace  
nsslapd-threadnumber="64"
```

```
Successfully replaced "nsslapd-threadnumber"
```

将 **nsslapd-threadnumber** 参数设置为 **-1** 以启用 **auto-tuning** 设置。

5.4. 使用 WEB 控制台手动设置线程数量

在某些情况下，需要手动设置修复多个 Directory Server 线程。例如，如果您不使用 **auto-tuning** 设置并更改虚拟机中的 CPU 内核数，调整 Directory 服务器线程数量可能会提高性能。

请注意，如果您之前设置了特定数量的线程，您可以使用 Web 控制台重新启用自动调整设置。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 导航到 **Server** → **Tuning & Limits**。
2. 在 **Number Of Worker Threads** 字段中，设置线程数量。
3. 点 **Save Settings**。

第 6 章 调优资源限制

目录服务器提供多个设置来调优实例使用的资源数量。您可以使用命令行或 Web 控制台进行更改。

6.1. 使用命令行更新资源限制设置

本节提供了如何更改资源限值设置的一般步骤。根据您的环境调整设置。

流程

1.

更新性能设置：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace  
parameter_name=value
```

您可以设置以下参数：

- **nsslapd-threadnumber:** 设置 worker 线程数量。
- **nsslapd-maxdescriptors:** 设置文件描述符的最大数量。
- **nsslapd-timelimit:** 设置搜索时间限制。
- **nsslapd-sizelimit:** 设置搜索大小限制。
- **nsslapd-pagedsizelimit:** 设置页面搜索大小限制。
- **nsslapd-idletimeout:** 设置闲置连接超时。
- **nsslapd-ioblocktimeout:** 设置输入/输出(I/O)块超时。

- **nsslapd-ndn-cache-enabled:** 启用或禁用规范化 DN 缓存。
- **nsslapd-ndn-cache-max-size:** 设置规范化 DN 缓存大小（如果启用了 **nsslapd-ndn-cache-enabled**）。
- **nsslapd-outbound-ldap-io-timeout:** 设置出站 I/O 超时。
- **nsslapd-maxbersize** : 设置最大基本编码规则(BER)大小。
- **nsslapd-maxsasliosize:**设置最大简单验证和安全层(SASL) I/O 大小。
- **nsslapd-listen-backlog-size** : 设置可用于接收传入连接的最大插槽数。
- **nsslapd-max-filter-nest-level:** 设置最大嵌套过滤器级别。
- **nsslapd-ignore-virtual-attrs:** enables 或禁用虚拟属性查找。
- **nsslapd-connection-nocanon:** enables 或禁用反向 DNS 查找。
- **nsslapd-enable-turbo-mode:** enables 或禁用 turbo 模式功能。

详情请查看 [Configuration](#) 和 [schema](#) 引用中的参数描述

2.

重启实例：

```
# dsctl instance_name restart
```

6.2. 使用 WEB 控制台更新资源限值设置


本节提供了如何更改资源限值设置的一般步骤。根据您的环境调整设置。

先决条件

- 在 **web 控制台** 中登录到实例。

流程

1. 导航到 **Server** → **Tuning & Limits**。
2. 更新设置。（可选）点击 **Show Advanced Settings** 显示所有设置。

Tuning & Limits 

Number Of Worker Threads	-	16	+	The number of worker threads that handle database operations. Set to '-1' for enable auto tuning. (nsslapd-threadnumber).
Search Time Limit	-	3600	+	
Search Size Limit	-	2000	+	
Paged Search Size Limit	-	0	+	
Idle Connection Timeout	-	3600	+	
I/O Block Timeout	-	10000	+	
▼ Hide Advanced Settings				
Outbound IO Timeout	-	300000	+	
Maximum BER Size	-	2097152	+	
Maximum SASL IO Size	-	2097152	+	
Listen Backlog Size	-	128	+	
Maximum Nested Filter Level	-	40	+	
<input checked="" type="checkbox"/> Disable Reverse DNS Lookups				
<input checked="" type="checkbox"/> Enable Connection Turbo Mode				
<input checked="" type="checkbox"/> Disable Virtual Attribute Lookups				
<input checked="" type="checkbox"/> Enable Normalized DN Cache				
NDN Max Cache Size	-	20971520	+	

3. 点 **Save Settings**。

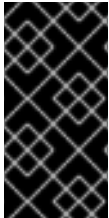
4.

点 Actions → Restart Instance。

6.3. 禁用 TRANSPARENT HUGE PAGES 功能

透明大内存页(THP)是 Linux 中的内存管理功能，通过使用较大的内存页面，可加快对具有大量内存的机器上的 Translation Lookaside Buffer (TLB)检查。RHEL 系统上默认启用 THP 功能，并支持 2 MB 内存页。

但是，THP 功能在大型连续分配模式上启用时效果最佳，并可以在涉及红帽目录服务器的小型稀疏分配模式中降低性能。进程的常驻内存大小可能会最终超过限制和影响性能，或者由内存不足(OOM)终止程序终止。



重要

为了避免性能和内存消耗问题，请在安装了 Red Hat Directory Server 的 RHEL 系统上禁用 THP。

流程

1.

检查 THP 的当前状态：

```
# cat /sys/kernel/mm/transparent_hugepage/enabled
```

2.

如果透明大内存页功能是活跃的，请在引导时禁用它或运行时：

•

在引导时通过在 `grub.conf` 文件中的内核命令行中添加以下内容来禁用透明大内存页：

```
transparent_hugepage=never
```

•

运行以下命令，在运行时禁用透明大内存页：

```
# echo never > /sys/kernel/mm/transparent_hugepage/enabled
# echo never > /sys/kernel/mm/transparent_hugepage/defrag
```

其他资源

- **RHDS 上的 Transparent Huge Pages (THP)的负面影响**
- **在 RHEL 7 中配置透明巨页.**

第 7 章 每个搜索操作日志记录统计信息

在一些搜索操作中，特别是带有 (cn=user*) 等过滤器，服务器用于接收任务的时间，然后发送结果返回 (etime) 可能会非常长。

使用搜索操作中使用的索引信息扩展访问日志有助于诊断为什么 etime 值的资源代价。

使用 nsslapd-statlog-level 属性启用收集统计信息，如多个索引查找（数据库读取操作）和每个搜索操作的索引查找的整体持续时间，对服务器的影响最少。

先决条件

- 启用了访问日志记录。

流程

1. 启用搜索操作指标：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-statlog-level=1
```

2. 重启实例：

```
# dsctl instance_name restart
```

验证

1. 执行搜索操作：

```
# ldapsearch -D "cn=Directory Manager" -H ldap://server.example.com -b "dc=example,dc=com" -s sub -x "cn=user*"
```

2. 查看访问日志文件并找到搜索统计记录：

```
# cat /var/log/dirsrv/slapd-instance_name/access
...
[16/Nov/2022:11:34:11.834135997 +0100] conn=1 op=73 SRCH
base="dc=example,dc=com" scope=2 filter="(cn=user)*" attrs=ALL
```

```
[16/Nov/2022:11:34:11.835750508 +0100] conn=1 op=73 STAT read index:  
attribute=objectclass key(eq)=referral --> count 0  
[16/Nov/2022:11:34:11.836648697 +0100] conn=1 op=73 STAT read index: attribute=cn  
key(sub)=er_ --> count 25  
[16/Nov/2022:11:34:11.837538489 +0100] conn=1 op=73 STAT read index: attribute=cn  
key(sub)=ser --> count 25  
[16/Nov/2022:11:34:11.838814948 +0100] conn=1 op=73 STAT read index: attribute=cn  
key(sub)=use --> count 25  
[16/Nov/2022:11:34:11.841241531 +0100] conn=1 op=73 STAT read index: attribute=cn  
key(sub)=^us --> count 25  
[16/Nov/2022:11:34:11.842230318 +0100] conn=1 op=73 STAT read index: duration  
0.000010276  
[16/Nov/2022:11:34:11.843185322 +0100] conn=1 op=73 RESULT err=0 tag=101  
nentries=24 wtime=0.000078414 optime=0.001614101 etime=0.001690742  
...
```

其他资源

- [TBA 链接到 nsslapd-statlog-level 描述](#)

第 8 章 监控本地磁盘以在低磁盘空间中关闭 DIRECTORY 服务器

当系统中的磁盘空间变得太小时，Directory 服务器进程会终止。因此，损坏数据库或丢失数据的风险。要防止这个问题，您可以配置 Directory Server，以监控包含配置、事务日志和数据库目录的文件系统上的可用磁盘空间。如果可用空间达到配置的阈值，Directory 服务器会关闭该实例。

8.1. DIRECTORY 服务器的行为取决于可用磁盘空间的数量

在配置监控时，Directory 服务器的行为取决于剩余空间的大小：

- 如果可用磁盘空间达到定义的阈值，Directory 服务器：
 - 禁用详细日志记录
 - 禁用访问日志
 - 删除归档的日志文件



注意

目录服务器始终继续写入错误日志，即使达到阈值也是如此。

- 如果可用磁盘空间低于配置的阈值的一半，Directory 服务器会在定义的宽限期内关闭。
- 如果可用磁盘空间低于 4 KB，目录服务器会立即关闭。

如果释放磁盘空间，Directory 服务器会中止关闭过程并重新启用所有之前禁用的日志设置。

8.2. 使用命令行配置本地磁盘监控

目录服务器可以监控包含配置、事务日志和数据库目录的文件系统上的可用磁盘空间。根据剩余的可用空间，Directory 服务器禁用特定的日志记录功能或关闭。

流程

1. 启用磁盘监控功能，设置阈值和一个宽限期：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-disk-monitoring=on nsslapd-disk-monitoring-threshold=3221225472 nsslapd-
disk-monitoring-grace-period=60
```

此命令将可用磁盘空间的阈值设置为 3 GB (3,221, 225, 472 字节)，并将宽限期设置为 60 秒。

2. 可选：配置 Directory Server 不禁用访问日志记录或删除归档的日志：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-disk-monitoring-logging-critical=on
```

3. 重启实例：

```
# dsctl instance_name restart
```

8.3. 使用 WEB 控制台配置本地磁盘监控

目录服务器可以监控包含配置、事务日志和数据库目录的文件系统上的可用磁盘空间。根据剩余的可用空间，Directory 服务器禁用特定的日志记录功能或关闭。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 导航到 **Server** → **Server Settings** → **Disk Monitoring**。
2. 选择“启用磁盘空间监控”。以字节为单位设置阈值和宽限期（以分钟为单位）：

General Settings	Directory Manager	Disk Monitoring	Advanced Settings	
<input checked="" type="checkbox"/> Enable Disk Space Monitoring				
Disk Monitoring Threshold		-	3221225472	+
Disk Monitoring Grace Period		-	60	+
<input type="checkbox"/> Preserve Logs Even If Disk Space Gets Low				

本例将监控阈值设置为 3 GB (3,221、225、472 字节), 以及 Directory 服务器在达到阈值 60 分钟后关闭实例的时间。

3. 可选：选择 **Preserve Logs Even If Disk Space Gets Low**
4. 点 **Save Settings**。
5. 点击右上角的 **Actions**, 然后选择 **Restart Instance**。

第 9 章 调整事务日志记录

每个 **Directory Server** 实例包含一个事务日志，用于记录它管理的数据库更新。每当执行目录数据库操作（如修改操作）时，服务器会为作为该 **LDAP** 操作调用的所有数据库操作创建一个数据库操作。这包括在数据库文件中更新条目记录，其中包含条目并更新所有属性索引。如果所有操作都成功，服务器会提交事务，将操作写入事务日志并验证整个事务是否已写入磁盘。如果有任何操作失败，服务器会回滚事务，并且丢弃所有操作。这种全体式方法保证更新操作是原子的。整个操作可以永久成功，且不可恢复，或者它会失败。

Directory 服务器定期清除事务日志的内容到实际数据库索引文件，并检查事务日志是否需要修剪。

如果服务器遇到故障（如关机），并正常关闭，有关最近目录更改的信息仍由事务日志保存。当服务器重启时，服务器会自动检测到错误条件，并使用数据库事务日志恢复数据库。

虽然数据库事务日志记录，清除数据库、修剪和数据库恢复是无需干预的自动进程，但建议调整一些数据库事务日志记录属性以优化性能。

9.1. 使用命令行更改数据库检查点间隔

在常规的间隔中，**Directory** 服务器将事务日志记录的事务写入数据库文件，并在数据库事务日志中记录一个检查点条目。通过指示哪些更改已经写入数据库，检查点条目表示要从事务日志开始恢复，从而加快恢复过程。

默认情况下，**Directory** 服务器每 60 秒向数据库事务日志发送检查点条目。增加检查点间隔可能会提高目录写入操作的性能。但是，它还可以在不排序关闭后增加恢复目录数据库所需的时间，并需要更多磁盘空间，因为大型数据库事务日志文件。

流程

- 将检查点间隔（例如，更改为 120 秒）：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --  
checkpoint-interval=120
```

9.2. 使用 WEB 控制台更改数据库检查点间隔

在常规的间隔中，**Directory** 服务器将事务日志记录的事务写入数据库文件，并在数据库事务日志中记录一个检查点条目。通过指示哪些更改已经写入数据库，检查点条目表示要从事务日志开始恢复，从而加

快恢复过程。

默认情况下，Directory 服务器每 60 秒向数据库事务日志发送检查点条目。增加检查点间隔可能会提高目录写入操作的性能。但是，它还可以在不排序关闭后增加恢复目录数据库所需的时间，并需要更多磁盘空间，因为大型数据库事务日志文件。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 导航到 Database → Global Database Configuration。
2. 单击 Show Advanced Settings。
3. 更新 Database Checkpoint Interval 字段的值。
4. 单击 Save Configuration。

9.3. 禁用持久事务

Durable 事务日志记录意味着每个 LDAP 更新操作（由事务中的一系列数据库操作组成）实际写入磁盘。尽管每个 LDAP 操作都可以由多个数据库更新组成，但每个 LDAP 操作都将被视为单个数据库事务。每个 LDAP 操作都是 atomic 和 durable。



警告

关闭持久化交易可以提高目录服务器的写入性能，同时降低数据丢失。

当您禁用持久的事务日志记录时，Directory 服务器会将每个目录数据库操作写入数据库事务日志文件中，但可能不会立即写入磁盘。如果目录被写入了逻辑数据库交易日志文件，但在系统崩溃时不会物理写

入磁盘，则更改将无法被恢复。当禁用持久事务时，恢复的数据库会一致，但不会反映在系统崩溃之前完成的所有 LDAP 写入操作的结果。

请注意，如果 Directory Server 正在运行，则无法更改 `nsslapd-db-durable-transaction` 参数。

流程

1.

停止实例：

```
# dsctl instance_name stop
```

2.

编辑 `/etc/dirsrv/slapd-instance_name/dse.ldif` 文件，并将 `cn=config,cn=ldbm database,cn=plugins,cn=config` 条目的 `nsslapd-db-durable-transaction` 参数设置为 `off`：

```
dn: cn=config,cn=ldbm database,cn=plugins,cn=config
...
nsslapd-db-durable-transaction: off
...
```

3.

启动实例：

```
# dsctl instance_name start
```

第 10 章 管理缓存设置

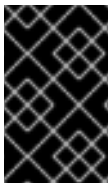
目录服务器使用以下缓存：

- 条目缓存，其中包含单独的目录条目。
- 可区分名称(DN)缓存用于将 DN 和相对区分名称(RDN)与条目关联。
- 包含数据库索引文件 *.db 文件的数据库缓存。

为了实现最高的性能改进，所有缓存大小都必须能够存储其所有记录。如果不使用推荐的自动大小功能且没有足够的 RAM 可用，请将可用内存分配给缓存，按照前面显示的顺序将可用内存分配给缓存。

10.1. CACHE-AUTOSIZE 和 CACHE-AUTOSIZE-SPLIT 参数如何影响数据库和条目缓存大小

默认情况下，Directory 服务器使用自动大小功能优化服务器硬件资源上的数据库和条目缓存的大小。



重要

红帽建议使用自动大小功能，而不手动设置缓存大小。

cn=config,cn=ldbm database,cn=plugins,cn=config 条目中的以下参数控制自动大小：

nsslapd-cache-autosize

这些设置控制是否为数据库和条目缓存启用自动大小。启用自动大小：

- 对于数据库和条目缓存，如果 nsslapd-cache-autosize 参数设置为大于 0 的值。
- 对于数据库缓存，如果 nsslapd-cache-autosize 和 nsslapd-dbcachesize 参数被设置为 0。
-

对于条目缓存，如果 `nsslapd-cache-autosize` 和 `nsslapd-cachememsize` 参数被设置为 0。

`nsslapd-cache-autosize-split`

- 值设定 Directory 服务器用于数据库缓存的 RAM 百分比。服务器使用条目缓存的剩余百分比。
- 将 1.5 GB RAM 用于数据库缓存不会提高性能。因此，Directory 服务器会将数据库缓存限制为 1.5 GB。

默认情况下，Directory 服务器使用以下默认值：

- `nsslapd-cache-autosize: 25`
- `nsslapd-cache-autosize-split: 25`
- `nsslapd-dbcachesize: 1,536 MB`

使用这些设置，使用系统的可用 RAM 的 25%(`nsslapd-cache-autosize`)。在这个内存中，服务器使用 25% 的数据库缓存(`nsslapd-cache-autosize-split`)，剩余的 75% 用于条目缓存。

根据可用 RAM，这会导致以下缓存大小：

表 10.1. 如果 `nsslapd-cache-autosize` 和 `nsslapd-cache-autosize-split` 使用默认值，缓存大小

GB 可用 RAM	数据库缓存大小	条目缓存大小
1 GB	64 MB	192 MB
2 GB	128 MB	384 MB
4 GB	256 MB	768 MB
8 GB	512 MB	1,536 MB

GB 可用 RAM	数据库缓存大小	条目缓存大小
16 GB	1,024 MB	3,072 MB
32 GB	1,536 MB	6,656 MB
64 GB	1,536 MB	14,848 MB
128 GB	1,536 MB	31,232 MB

10.2. 所需的缓存大小

`dsconf monitor dbmon` 命令可让您监控运行时的缓存统计信息。要显示统计，请输入：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com monitor dbmon
DB Monitor Report: 2022-02-24 10:25:16
-----
Database Cache:
- Cache Hit Ratio: 50%
- Free Space: 397.31 KB
- Free Percentage: 2.2%
- RO Page Drops: 0
- Pages In: 2934772
- Pages Out: 219075

Normalized DN Cache:
- Cache Hit Ratio: 60%
- Free Space: 19.98 MB
- Free Percentage: 99.9%
- DN Count: 100000
- Evictions: 9282348

Backends:
- dc=example,dc=com (userroot):
  - Entry Cache Hit Ratio: 66%
  - Entry Cache Count: 50000
  - Entry Cache Free Space: 2.0 KB
  - Entry Cache Free Percentage: 0.8%
  - Entry Cache Average Size: 8.9 KB
  - DN Cache Hit Ratio: 21%
  - DN Cache Count: 100000
  - DN Cache Free Space: 4.29 MB
  - DN Cache Free Percentage: 69.8%
  - DN Cache Average Size: 130.0 B
```

(可选) 将 `-b` 后端或 `-x` 选项传递给命令，以显示特定后端或索引的统计信息。

如果您的缓存的大小足够大小，DN Cache Count 中的数字与 Cache Count 后端条目中的值匹配。另外，如果所有条目和 DN 都出现在其对应缓存中，则条目缓存计数 值与 DN Cache Count 的值匹配。

示例输出显示：

- 只有 2.2% 可用的数据库缓存被保留：

```
Database Cache:
...
- Free Space:      397.31 KB
- Free Percentage: 2.2%
```

但是，为了高效操作，至少需要 15% 免费的数据库缓存。要确定数据库缓存的最佳大小，请计算 `/var/lib/dirsrv/slaped-instance_name/db/` 目录（包括子目录和 changelog 数据库）的所有 *.db 文件的大小，并为开销添加 12%。

要设置数据库缓存，请参阅 [使用命令行设置数据库缓存大小](#)。

- userroot 数据库的 DN 缓存很好：

```
Backends:
- dc=example,dc=com (userroot):
...
- DN Cache Count:      100000
- DN Cache Free Space: 4.29 MB
- DN Cache Free Percentage: 69.8%
- DN Cache Average Size: 130.0 B
```

数据库的 DN 缓存包含 100000 记录，69,8% 的缓存是空闲的，内存中的每个 DN 需要 130 字节。

要设置 DN 缓存，请参阅 [使用命令行设置 DN 缓存大小](#)。

- userroot 数据库的条目缓存统计表示应提高条目缓存值以提高性能：

```
Backends:
- dc=example,dc=com (userroot):
...
- Entry Cache Count:   50000
```

- Entry Cache Free Space: 2.0 KB
- Entry Cache Free Percentage: 0.8%
- Entry Cache Average Size: 8.9 KB

条目缓存在此数据库 50000 记录中包括，仅保留 2 Kilobytes 的可用空间。要启用 Directory 服务器缓存所有 100000 DN，缓存必须增加到 890 MB (100000 DN * 8.9 KB 平均条目大小)。但是，红帽建议将所需的最小大小放在下一个最高 GB，并加倍结果。在这个示例中，条目缓存应设置为 2 千兆字节。

要设置条目缓存，请参阅 [使用命令行设置条目缓存大小](#)。

10.3. 使用命令行设置数据库缓存大小

数据库缓存包含数据库的 Berkeley 数据库索引文件，即数据库用于属性索引的所有 *.db 和其他文件。这个值传递给 Berkeley DB API 功能 `set_cachesize ()`。这个缓存大小对 Directory 服务器性能的影响小于条目缓存大小，但如果设置了条目缓存大小后有可用的 RAM，则增加分配给数据库缓存的内存量。

流程

1. 禁用自动缓存调整

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
cache-autosize=0
```

2. 手动设置数据库缓存大小：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
dbcachesize=268435456
```

指定数据库缓存大小（以字节为单位）。在本例中，命令将数据库缓存设置为 256 MB。

3. 重启实例：

```
# dsctl instance_name restart
```

10.4. 使用 WEB 控制台设置数据库缓存大小

数据库缓存包含数据库的 Berkeley 数据库索引文件，即数据库用于属性索引的所有 *.db 和其他文件。这个值传递给 Berkeley DB API 功能 `set_cachesize ()`。这个缓存大小对 Directory 服务器性能的影响小于条目缓存大小，但如果设置了条目缓存大小后有可用的 RAM，则增加分配给数据库缓存的内存量。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 导航到 Database → Global Database Configuration。
2. 取消选择 自动缓存调整。
3. 点 Save Config。
4. 在 Database Cache Size 字段中输入数据库缓存大小（以字节为单位），如 268435456 for 256 MB。
5. 点 Save Config。
6. 点击右上角的 Actions，然后选择 Restart Instance。

10.5. 使用命令行设置 DN 缓存大小

目录服务器使用 `entryrdn` 索引将可分辨名称(DN)和 `relative distinguished name (RDN)`与条目关联。它使服务器能够有效地重命名子树、移动条目和执行 `moddn` 操作。服务器使用 DN 缓存缓存 `entryrdn` 索引的内存中表示，以避免昂贵的文件 I/O 和转换操作。

如果您不使用自动调整功能（尤其是对于最佳性能），但没有限制重命名条目和移动操作，请将 DN 缓存设置为启用目录服务器在数据库中缓存所有 DN 的大小。

如果 DN 不在缓存中，Directory 服务器会从 `entryrdn.db` 索引数据库文件读取 DN，并将 DN 从磁盘上的格式转换为内存中格式。存储在缓存中的 DNS 可让服务器跳过磁盘 I/O 和转换步骤。

流程

1.

显示后缀及其对应的后端：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com suffix list  
dc=example,dc=com (userroot)
```

这个命令显示每个后缀旁的后端数据库名称。下一步需要后缀的数据库名称。

2.

设置 DN 缓存大小：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend suffix set --  
dncache-memsize=20971520 userRoot
```

这个命令将 userRoot 数据库的 DN 缓存设置为 20MB。

3.

重启实例：

```
# dsctl instance_name restart
```

10.6. 使用 WEB 控制台设置 DN 缓存大小

目录服务器使用 `entryrdn` 索引将可分辨名称(DN)和 `relative distinguished name (RDN)`与条目关联。它使服务器能够有效地重命名子树、移动条目和执行 `moddn` 操作。服务器使用 DN 缓存缓存 `entryrdn` 索引的内存中表示，以避免昂贵的文件 I/O 和转换操作。

如果您不使用自动调整功能（尤其是对于最佳性能），但没有限制重命名条目和移动操作，请将 DN 缓存设置为启用目录服务器在数据库中缓存所有 DN 的大小。

如果 DN 不在缓存中，Directory 服务器会从 `entryrdn.db` 索引数据库文件读取 DN，并将 DN 从磁盘上的格式转换为内存中格式。存储在缓存中的 DNS 可让服务器跳过磁盘 I/O 和转换步骤。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 导航到 Database → es suffix_name。
2. 在 DN Cache Size 字段输入 DN 缓存大小（以字节为单位）。
3. 单击 Save Configuration。
4. 点击右上角的 Actions，然后选择 Restart Instance。

10.7. 使用命令行设置条目缓存大小

目录服务器使用条目缓存来存储搜索和读取操作期间使用的目录条目。将条目缓存设置为可让目录服务器存储所有记录的大小对搜索操作具有最高的性能。

如果没有配置条目缓存，Directory 服务器会从 id2entry.db 数据库文件中读取条目，并将区分名称 (DN) 从 on-disk 格式转换为内存中格式。存储在缓存中的条目可让服务器跳过磁盘 I/O 和转换步骤。

流程

1. 禁用自动缓存调整：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend config set --
cache-autosize=0
```

2. 显示后缀及其对应的后端：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com suffix list
dc=example,dc=com (userroot)
```

这个命令显示每个后缀旁的后端数据库名称。下一步需要后缀的数据库名称。

3. 为数据库设置条目缓存大小（以字节为单位）：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend suffix set --  
cache-memsize=2147483648 userRoot
```

这个命令将 `userRoot` 数据库的条目缓存设置为 2GB。

4. 重启实例：

```
# dsctl instance_name restart
```

10.8. 使用 WEB 控制台设置条目缓存大小

目录服务器使用条目缓存来存储搜索和读取操作期间使用的目录条目。将条目缓存设置为可让目录服务器存储所有记录的大小对搜索操作具有最高的性能。

如果没有配置条目缓存，Directory 服务器会从 `id2entry.db` 数据库文件中读取条目，并将区分名称 (DN) 从 `on-disk` 格式转换为内存中格式。存储在缓存中的条目可让服务器跳过磁盘 I/O 和转换步骤。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 导航到 `Database` → `es suffix_name` → `Settings`。
2. 禁用自动 缓存调优 设置。
3. 单击 `Save Configuration`。
4. 点击右上角的 `Actions`，然后选择 `Restart Instance`。

5. 导航到 **Database** → **es suffix_name** → **Settings**。
6. 在 **'Entry Cache Size** 字段中设置数据库缓存的大小。
7. 单击 **Save Configuration**。
8. 点击右上角的 **Actions**，然后选择 **Restart Instance**。

第 11 章 提高导入性能

在导入操作过程中，非常大的属性大小或大量条目可能会对服务器性能造成负面影响。本节论述了如何调整目录服务器设置以提高导入性能。

11.1. 使用大型数据库导入和导入的大型数据库导入和导入，使用大型属性值调整目录服务器

为操作使用导入缓存自动大小功能，如下所示：

- 导入非常大的数据库
- 导入具有大型属性的数据库，如存储证书链或镜像的二进制属性



注意

离线导入比在线导入要快。考虑尽可能使用离线导入。

您可以使用 `nsslapd-import-cache-autosize` 属性配置导入缓存自动功能。默认情况下，Directory 服务器只启用导入缓存自动调整 `ldif2db` 操作，自动为导入缓存分配 50% 的空闲物理内存。

详情请查看 [Configuration, Command, and File Reference](#) 文档中的 [nsslapd-import-cache-autosize](#) 属性的描述。

第 12 章 调优服务器连接管理

`nsslapd-numlisteners` 属性指定目录服务器可用于监控已建立的连接的监听程序线程数量。您可以通过增加属性值来提高服务器遇到大量客户端连接时的响应时间。

12.1. 使用命令行管理连接监听程序线程数量

您可以使用命令行管理连接监听程序线程数量。默认值为 1。

流程

1. 列出连接监听程序线程的数量：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get nsslapd-  
numlisteners
```

2. 修改连接监听程序线程的数量：

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace  
nsslapd-  
numlisteners=4
```

3. 重启实例：

```
# dsctl instance_name restart
```