

# 基于 Asianux 3 workstation (for x86) 平台的 流媒体点播服务器的设计与实现

**摘要:**随着网络带宽的不断加大和网络性能的不断提高, 基于网络的实时流媒体将成为主流。与传统的物理介质媒体相比, 网络流媒体具有传播快速, 时效性高, 使用方便等特点。跟版权技术相结合, 实时流媒体将会成为人们接收音频、视频的主要方式。我们利用 Real 公司 Helix 作为流媒体服务器, 针对中小型局域网提供视频共享、流媒体点播服务, 比较适合学校、网吧以及生活小区的多媒体点播。在本设计中, 通过在虚拟机上部署三台服务器进行测试, 基本上能够满足视频点播要求。

**关键词:** 流媒体点播; Asianux 3 workstation (for x86) ; Helix Server ; RealPlayer 11

## 1 背景

宽带网络的普及催生了对实时流媒体的需求。网络条件下, 人们可以不必像以往一样被动等待物理介质媒体的传播, 而是可以从网络上, 动态实时选择性地接收自己所需要的视频和音频信息。流媒体是以媒体流的形式发布到本地计算机并进行播放而不需要打包下载。但是这种功能的实现是基于优良的网络性能的, 如果网络拥挤或者服务器宕机, 就有可能造成传输中断。因此在实际应用中都会根据实际情况对服务器数据进行备份或者冗余。

## 2 应用范围

本设计是针对中小型局域网(校园网、小区宽带等)提出的一套视频点播服务器设计方案。但是, Helix Server 是 Real 公司产品的一款优秀的流媒体服务器软件, 就其软件功能而言, 并不受网络范围大小的局限。此方案是在多台虚拟机组成的小型局域网上设计调试的, 受技术水平和硬件环境的限制, 可能有些功能需要进一步优化或者细化。

## 3 特点和设计思路

### 3.1 特点

本设计的主要特点在于强调整个系统的稳定性和软硬件资源的利用率。提高系统稳定性最为主要和常用的手段就是数据的冗余, 通过对 Helix Server 的双机热备份实现在复杂网络和高负荷环境下提供服务的稳定性和连续性。在提高资源利用率方面, 我们采用了传输服务器这一手段来简化主/从服务器的配置和管理, 同时将数据从主/从服务器转移到了传输服务器, 有效降低了整个系统对存储媒体的需求。

### 3.2 设计思路

本着由浅入深的顺序, 首先着手实现单台服务器的服务功能, 为了进一步提高系统稳定性, 我们加入了一台冗余服务器。由于冗余服务器上的数据是主服务器的完全拷贝, 整体的使用效率并不高, 所以考虑使用将主/冗余服务器相互冗余(商业应用中至少需要三台服务器)并同时对外提供服务(均衡负载)的办法来提高资源利用率。最后, 引进一台(也可多台)传送服务器, 通过它向主/冗余服务器广播/单播媒体流, 从而取消了在主/冗余服务器的数据备份, 同时提高了系统的服务质量。

#### 3.2.1 基本功能

考虑到局域网带宽的实际情况和用户需求, 我们假设要在局域网中部署一台 Helix 主服务器, 用于向用户提供基本的视频点播服务。试用版单台服务器提供正常服务所能够承受的最大负载数为 100, 那么最多可以同时为 100 个用户提供在线视频点播服务, 如图 1。

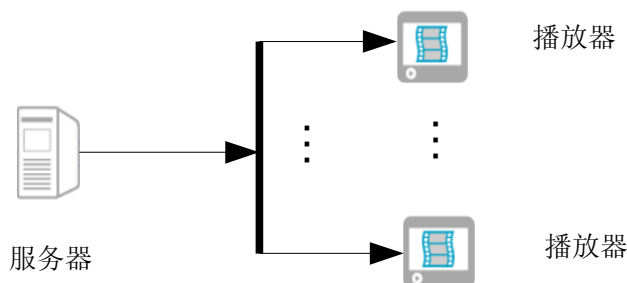


图 1 单台服务器提供服务

#### 3.2.2 功能扩展 1: 冗余服务器

增加一台冗余服务器。在基本功能实现的基础上, 如果用户数进一步增加, 将会造成服务质量下降甚至引起服务器崩溃而无法继续提供服务。基于这种情况, 我们考虑引入一台冗余服务器, 以便在各种不可预知的网络故障造成主服务器不能正常工作时, 冗余服务器能够接管整个网络的多媒体服务: 当

Real Player 通过 RTSP 与主服务器相连时，主服务器会将一个冗余服务器列表发送给 Real Player，一旦 Real Player 与主服务器之间的连接发生中断，Real Player 会根据列表中的条目尝试连接冗余服务器，如果存在多台冗余服务器，那么它将随机选择一台，如图 2。

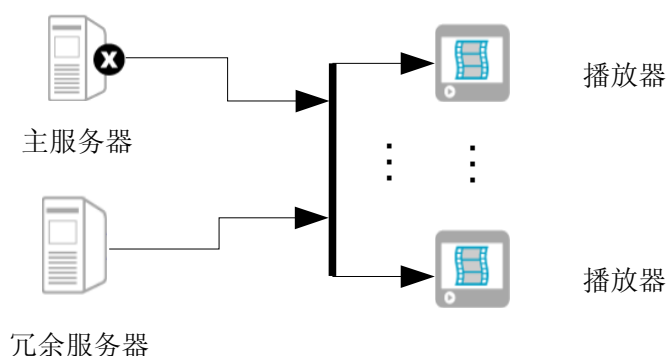


图 2 设置冗余服务器提高系统稳定性

### 3.2.3 功能扩展 2: 数据冗余+负载均衡

实际上扩展 1 虽然通过增加一台冗余服务器来增强系统的稳定性，但是并没有充分利用现有资源。首先，冗余服务器上的多媒体文件是主服务器的完全拷贝，而这些资源只有在主服务器宕机之后才能发挥作用；其次，在主服务器失效后，所有连接用户都将自动转移到冗余服务器上去，冗余服务器的压力依然很大，仍有失效的可能；第三，如果主服务器在暂时性失效后又恢复工作，那么先前转移到冗余服务器上的用户将继续驻留冗余服务器，主服务器可以空余部分资源为后续用户提供服务，这是最好情况，如果主服务器崩溃，后续用户将无法连接，冗余服务器也就不能为新用户提供服务。

因此，我们将主/冗余服务器实现负载均衡：两台服务器不再区分主/冗余服务器，每台服务器在做主服务器提供视频点播服务的同时，还兼做另外一台服务器的冗余服务器（我们称之为服务器 1 和服务 器 2）。用户可以根据服务器当前负载大小选择性地连入负载较少的任何一台服务器：这样既降低了单台服务器的压力，又充分利用了原来冗余服务器上的已有资源，同时还有效提升了可服务的用户数（由 100 提升至 200）。当然，这样仍然有过载的危险：如果一台崩溃，两台服务器的用户将同时加载到一台服务器上如图 3（a, b）。商业应用中应该至少用三台这样的服务器组成集群。

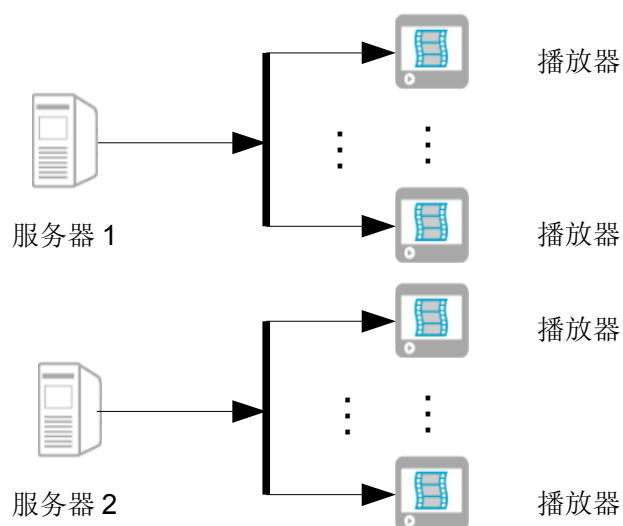


图 3（a）正常情况下两台服务器分别提供服务

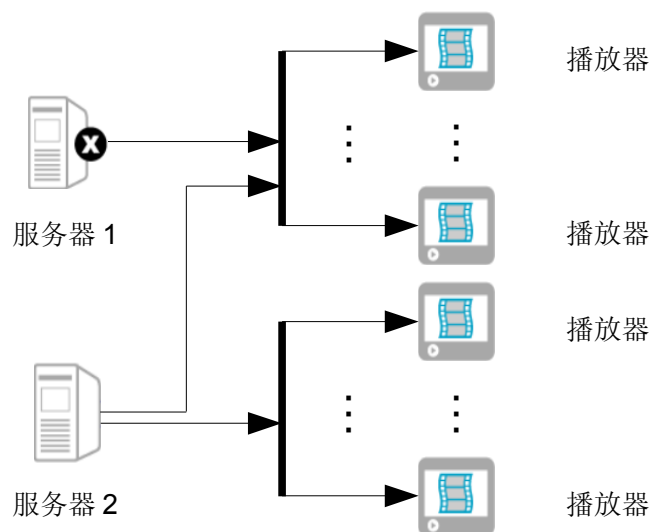


图 3 (b) 一台服务器失效后, 另一台提供冗余

### 3.2.4 功能扩展 3: 传送服务器

为了向用户提供更为清晰、流畅的视频服务, 我们考虑再在系统中增加一台发送服务器, 同时将两台视频服务器设置为接收服务器。

**Helix 的传送服务器的工作原理:** 首先利用解码器 (系统的或者软件自带的) 将媒体文件 (主要是 Real 媒体文件) 解码, 然后传送服务器通过广播或者单播的方式将直播流 (主要是视频流) 播送出去。在传送服务器的接收服务器列表中定义的接收服务器依据端口号来确定所要接收和转发的内容, 接收服务器就不再保存视频文件, 极大地节省了存储空间。一旦传送服务器开始直播流文件, 接收服务器就从发送服务器中接收广播内容。当用户向接收服务器发送连接请求时, 接收服务器替代传送服务器来向客户端提供更快更高质量的广播服务。由于在接收服务器和发送服务器之间已经预先建立好连接, 所以当客户端向接收服务器提出数据请求的时候, 广播内容将被迅速的发布到客户端上, 从而减轻高峰期主服务器的工作压力, 有效利用带宽资源。在整个过程中, 服务器 1 和服务器 2 只起到了接收和分发媒体流的作用, 相当于两台分发服务器, 这就缩短了用户与源服务器之间的距离, 同时极大地减轻了传送服务器的压力。并且传送服务器和服务器 1 及服务器 2 之间的连接是在用户请求之前建立的, 当接收到用户发送的请求时, 服务器 1 和服务器 2 就可以直接转发媒体流, 从而可以向用户提供更为清晰流畅的多媒体服务。

传送服务器可以通过 UDP, TCP 或者 UDP 多播的方式来传送广播, 默认采用 UDP 单播方式。具体工作过程见图 4 (a,b)。

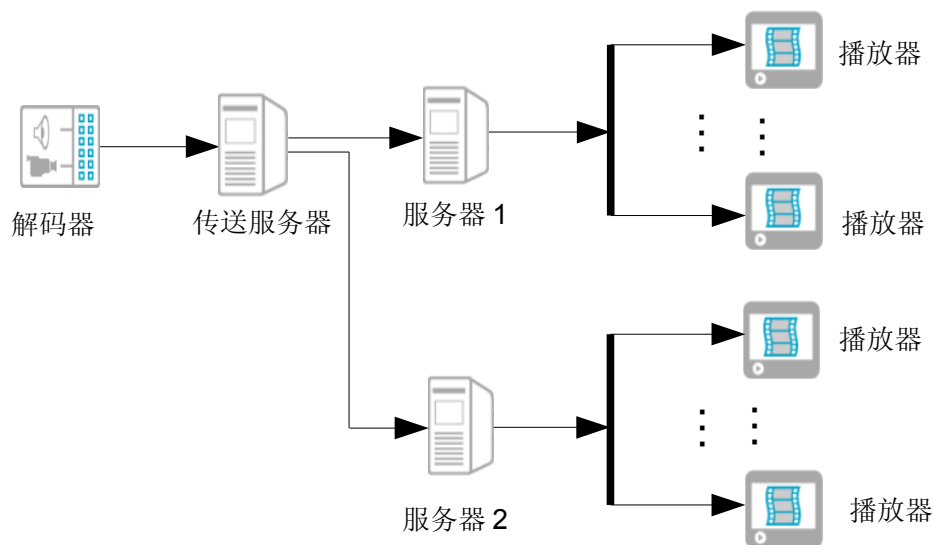


图 4(a) 利用传送服务器改进整体性能

同样，如果其中一台服务器失效，另一台将提供冗余。

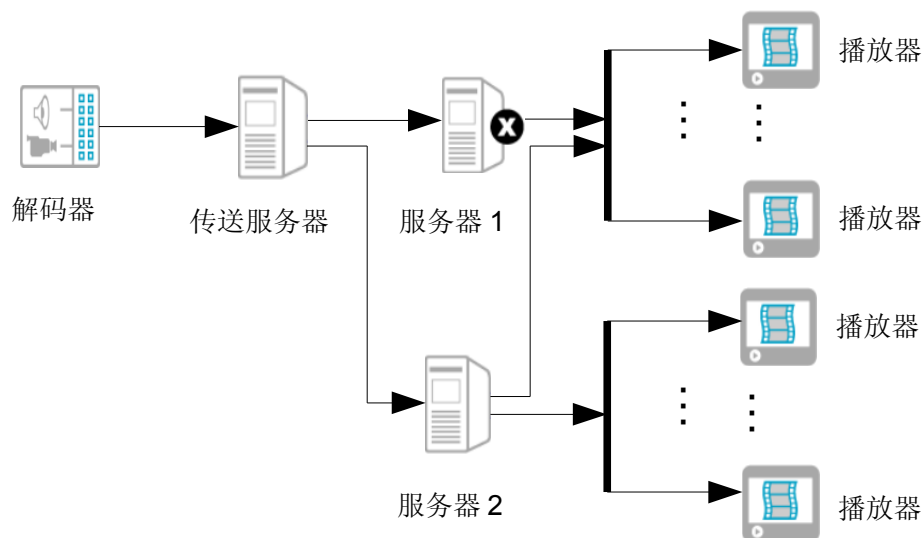


图 4 (b) 其中一台服务器失效后另一台继续提供服务

#### 4 硬件和软件环境要求

硬件平台为 Intel x86-32 位，CPU 为 P4 3.0G，2G 内存，本设计所有的系统设计、调试、测试、验收全部基于本平台展开。

软件基于 Asianux 3 workstation (for x86) 系统平台，为满足设计要求，在不影响系统稳定运行的基础上进行了适量裁减，去掉了部分系统自带而与本设计不相关的服务及软件包，具体内容详见（裁减软件列表）。考虑到系统实现的便捷性与整体性能，我们选择使用 VMware Workstation 虚拟机软件，流媒体服务器软件是 Real 公司提供的 Helix 流媒体服务器 for RedHat EL4，版本号为 rs1200-ga-linux-rhel4，无限制试用版，许可证文件为 RNKey-Helix\_Server\_Unlimited-120-584799885134542.lic，有效期至 2008 年 12 月 03 日。视频播放软件为 RealPlayer 11 for Linux。

#### 5 系统设计

首先，为了简化配置过程，我们先将整个设计过程中使用到的三个 Helix 服务器实例列表。

表 1: 服务器安装

服务器名	安装路径	配置文件(基于安装路径)	执行命令(基于安装路径)
传送服务器 (Helix_transmitter)	/usr/local/Helix_transmitter	rmserver.cfg	/Bin/rmserver rmserver.cfg
主服务器(Helix_master)	/usr/local/Helix_master	rmserver.cfg	/Bin/rmserver rmserver.cfg
冗余服务器 (Helix_redundant)	/usr/local/Helix_redundant	rmserver.cfg	/Bin/rmserver rmserver.cfg

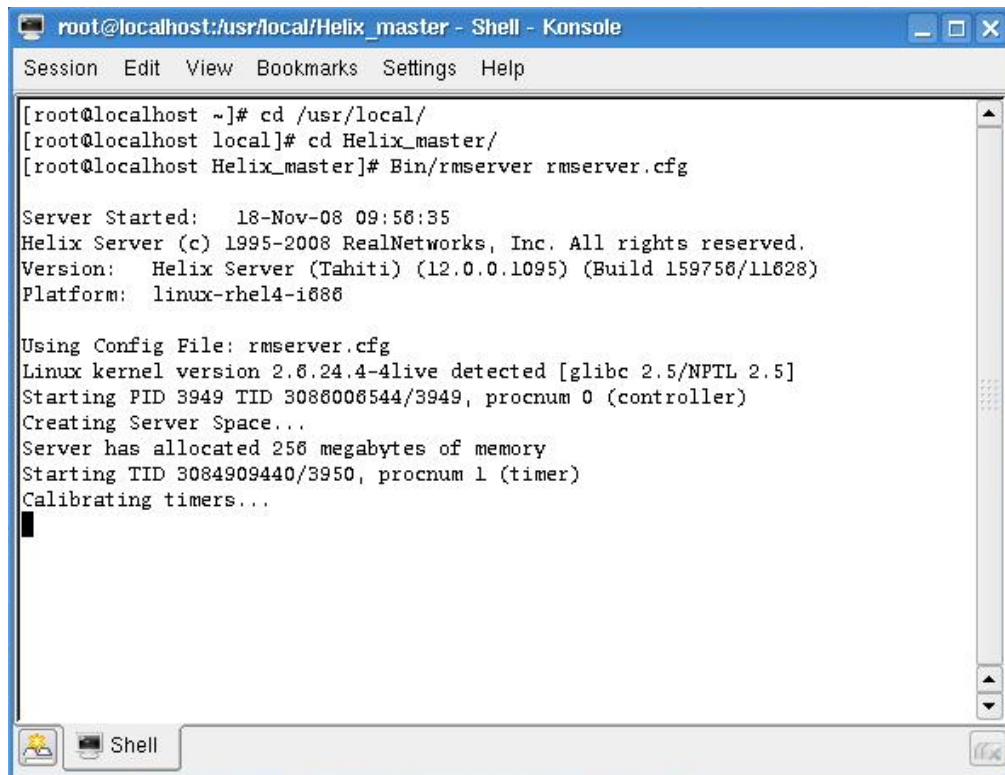
表 2: 服务器常用端口（括号内为默认端口）

服务器名	RTSP(554)	PNA(7070)	HTTP(80)	MMS(1755)	监控端口(9090)	管理端口(随机)
传送服务器 (Helix_transmitter)	558	7070	80	1740	9080	15246
主服务器(Helix_master)	554	7070	8080	1750	9090	10527
冗余服务器 (Helix_redundant)	556	7070	8088	1760	9099	14004

使用的流媒体播放器为 RealPlayer 11，安装路径为 /usr/local/RealPlayer，可以从 Start 菜单栏中选择打开。

##### 5.1 基本功能的配置实现

从控制台启动 Helix\_master 服务器，可得到如下画面，关闭服务器使用 **Ctrl + c**。



```
root@localhost:/usr/local/Helix_master - Shell - Konsole
Session Edit View Bookmarks Settings Help

[root@localhost ~]# cd /usr/local/
[root@localhost local]# cd Helix_master/
[root@localhost Helix_master]# Bin/rmserver rmserver.cfg

Server Started: 18-Nov-08 09:56:35
Helix Server (c) 1995-2008 RealNetworks, Inc. All rights reserved.
Version: Helix Server (Tahiti) (12.0.0.1095) (Build 159756/11628)
Platform: linux-rhel4-i686

Using Config File: rmserver.cfg
Linux kernel version 2.6.24.4-4live detected [glibc 2.5/NPTL 2.5]
Starting PID 3949 TID 3086006544/3949, procnum 0 (controller)
Creating Server Space...
Server has allocated 256 megabytes of memory
Starting TID 3084909440/3950, procnum 1 (timer)
Calibrating timers...
█
```

图5 启动 Helix\_master 服务器

通过在Firefox浏览器的地址栏中输入 <http://127.0.0.1:10527/admin/index.html> 可以打开 Helix 服务器自带的一个控制界面，通过此界面可以实现一些基本功能的配置。借助于软件的默认安装（例如 RTSP 端口 554，PNA 端口 7070，HTTP 端口 80，MMS 端口 1755，监控端口 9090，以及管理端口 10527（随机值））等就可以实现单台服务器功能。下面是控制界面“服务器设置”->“端口”控制台的截图：



## 服务器设置

## 端口

IP绑定

MIME类型

连接控制

冗余服务器

配置加载点

URL 别名

HTTP分发

缓存目录

User/Group Name

媒体演示

## 安全设置

## 日志和监控

## 广播设置

## 广播分发

## 内容管理

## 服务器设置

下面列出的端口号是服务器用于PNA,HTTP,RTSP,监控和管理请求的端口.保留它们的默认设置将是更多的人能访问到你的内容.如果你要更改它们的默认值,你需要更改所有的链接中的端口指向.服务器同样对编码器,分发以及多播设定默认端口.

出于安全因素,管理端口在安装时被设定为一个随机值.请校验您重新定义的端口是否系统现有的端口相冲突.

在UNIX系统中,访问低于1024的端口的时候,你需要以管理员的身份登录.

## → 端口

帮助

RTSP 端口  (默认值为 554)PNA 端口  (默认值为 7070)HTTP 端口  (默认值为 80)MMS 端口  (默认值为 1755)监控端口  (默认值为 9090)管理端口 

警告:如果你更改了管理端口,你必须记住新的端口用以重新登录进入管理界面.同时,你需要更改所有指向本页面的地址.

新建 管理员帐户

启用Ramgen端口发送地址线索 

包括对屏蔽用户开放地址中的HTTP端口

启用ASXGen发送HTTP连接地址 

UDP重发端口范围

 to 

→ 应用 重置

图 6 Helix\_master 服务器的端口配置页面

对应的配置文件内容如下:

```
<!-- PORTS -->
<!--UNIX customers must have root privileges to execute the server -->
<!--with the RTSP port set to 554. -->
<!--The following are the default ports that RealPlayer and -->
<!--RealPlayer Plus clients will connect to for an URL that has -->
<!--no port specified: -->
<!-- RTSP: 554 -->
<!-- HTTP: 80 (...then 8080 if 80 is unavailable) -->
<!-- MMS: 1755 -->
<Var RTSPPort="554"/>
<Var HTTPPort="8080"/>
<Var MMSPort="1750"/>
<Var MonitorPort="9090"/>
<Var AdminPort="10527"/>
<Var ChannelControlPort="8006"/>
```

对应图 6 的端口

通过 Start -> RealPlayer 11, 选择 file -> open location -> rtsp://127.0.0.1:554/11realvideo10.rm , 可以看到如下播放画面:

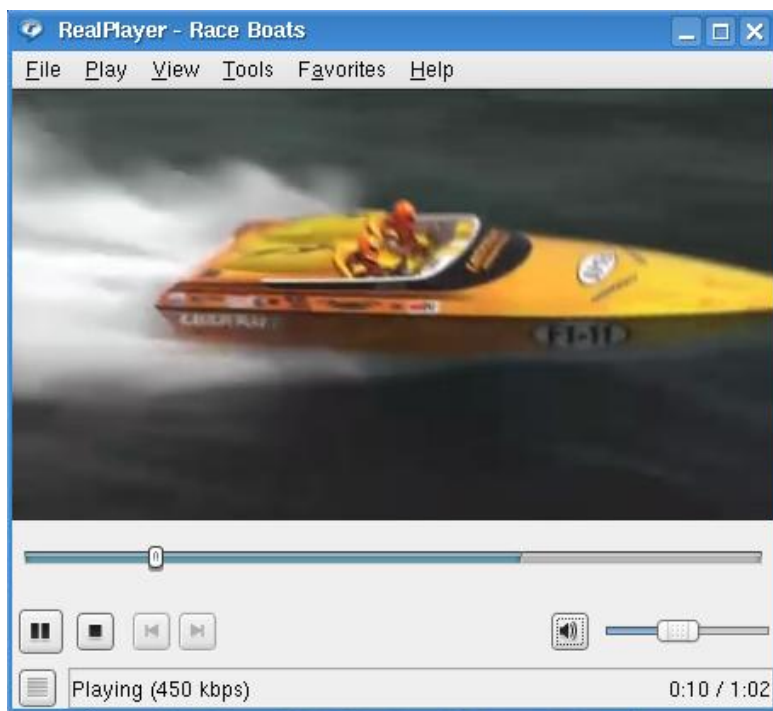


图 7 RealPlayer 11 的播放画面

### 3.2 功能扩展 1 的实现

要实现冗余功能, 我们计划使用两台服务器, 分别命名为主服务器(Helix\_master)和冗余服务器(Helix\_redundant)。正常情况下主服务器提供服务, 当主服务器失效后, 冗余服务器接替主服务器进行工作。主服务器和冗余服务器的设置基本相同, 由于是在一台主机上进行虚拟实现, 我们通过使用不同端口对主/冗余服务器进行区别。主/冗余服务器根目录下的/Content 目录为默认的媒体文件存放目录, 要进行冗余, 主/冗余服务器的此目录下必须有相同的媒体文件。

配置冗余服务器。打开 Firefox 输入 <http://127.0.0.1:14004/admin/index.html> 通过控制界面进行配置。启动方法及端口配置同“基本功能实现”, 具体端口分配见“表 2”。

配置主服务器。打开 Firefox 输入 <http://127.0.0.1:10527/admin/index.html>。要实现冗余功能, 我们要在主服务器的控制页面中“服务器设置”->“冗余服务器”页面进行设置。由于是设置主服务器的冗余服务器, 我们应该在主服务器的控制界面进行设置。

选择“添加冗余服务器”, 服务器描述为: Redundant, 主机: 127.0.0.1, 端口: 556。添加“重定向规则”, 为将所有资源冗余, 我们在“编辑生效路径”填入“/”, 在“用于替换的服务器”下拉菜单中选择我们已经配置好的冗余服务器“Redundant”, “规则例外路径”不予配置。控制页面截图如图 8

。

## 服务器设置

端口

IP绑定

MIME类型

连接控制

冗余服务器

配置加载点

URL别名

HTTP分发

缓存目录

User/Group Name

媒体演示

安全设置

日志和监控

广播设置

广播分发

内容管理

## 服务器设置

当发生网络中断和服务器故障时,RealOne Player (9.0和以上版本)在播放中断时将会自动尝试重新建立连接. Helix 服务器将指派一个或者更多的冗余服务器来继续提供服务. 在建立最初连接时,冗余服务器的信息就被同时传达给RealOne Player, 它将根据下面页面中定义的指派方式重新连接到相应的冗余服务器.

主机条目可以是一个IP地址或者是一个主机名.这个IP地址或主机名将解析到一个DNS转向或者是交换机上所定义的虚拟地址. 任何一台冗余服务器必须可以发布和主服务器一样的直播流,同时拥有和主服务器一样的点播资源.

冗余服务器将被指派应用于某一个资源路径. 如果想对整个主机的内容进行冗余服务,请在下面的规则定义中使用"/"作为生效路径.

冗余服务器

冗余服务器

Redudant

重定向规则

/

规则例外路径

描述

Redudant

主机

127.0.0.1

端口

556

编辑生效路径

/

用于替换的服务器

Redudant

在此规则中增加一个服务!

编辑路径

应用 重置

图8 主服务器中冗余服务器的配置

主服务器配置文件模块如下:

端口部分

```
<!-- PORTS -->
<!--UNIX customers must have root privileges to execute the server -->
<!--with the RTSP port set to 554. -->
<!--The following are the default ports that RealPlayer and -->
<!--RealPlayer Plus clients will connect to for an URL that has -->
<!--no port specified: -->
<!-- RTSP: 554 -->
<!-- HTTP: 80 (...then 8080 if 80 is unavailable) -->
<!-- MMS: 1755 -->
<Var RTSPPort="554"/>
<Var HTTPPort="8080"/>
<Var MMSPort="1750"/>
<Var MonitorPort="9090"/>
<Var AdminPort="10527"/>
<Var ChannelControlPort="8006"/>
```



### 冗余服务器配置部分

```
<!-- REDUNDANT SERVERS -->
<List Name="ServerAlternates">
  <List Name="Alternates">
    <List Name="Redudant">
      <Var Port="556"/>
      <Var Host="127.0.0.1"/>
    </List>
  </List>
</List>
<List Name="RedirectRules">
  <List Name="100">
    <List Name="Alternates">
      <Var 100="Redudant"/>
    </List>
    <Var Rule="/">
  </List>
</List>
<List Name="ExcludePaths">
</List>
</List>
```

冗余服务器配置文件端口模块如下：

```
<!-- PORTS -->
<!--UNIX customers must have root privileges to execute the server -->
<!--with the RTSP port set to 554. -->
<!--The following are the default ports that RealPlayer and -->
<!--RealPlayer Plus clients will connect to for an URL that has -->
<!--no port specified: -->
<!-- RTSP: 554 -->
<!-- HTTP: 80 (...then 8080 if 80 is unavailable) -->
<!-- MMS: 1755 -->
<Var RTSPPort="556"/>
<Var HTTPPort="8088"/>
<Var MMSPort="1760"/>
<Var MonitorPort="9099"/>
<Var AdminPort="14004"/>
<Var ChannelControlPort="8010"/>
```

冗余功能测试：通过系统控制台打开主服务器和冗余服务器。

点□ Start -> RealPlayer 11, 选择 file -> open location -> rtsp://127.0.0.1:554/11realvideo10.rm , 可以看到正常播放画面, 然后在主服务器的系统控制台按 **Ctrl+c** 关闭主服务器, 保留冗余服务器继续运行, 可以看到播放器在经过短暂缓冲后继续播放, 此时再关闭冗余服务器, 那么 RealPlayer 11 将弹出连接中断的警告对话框, 如图 9。

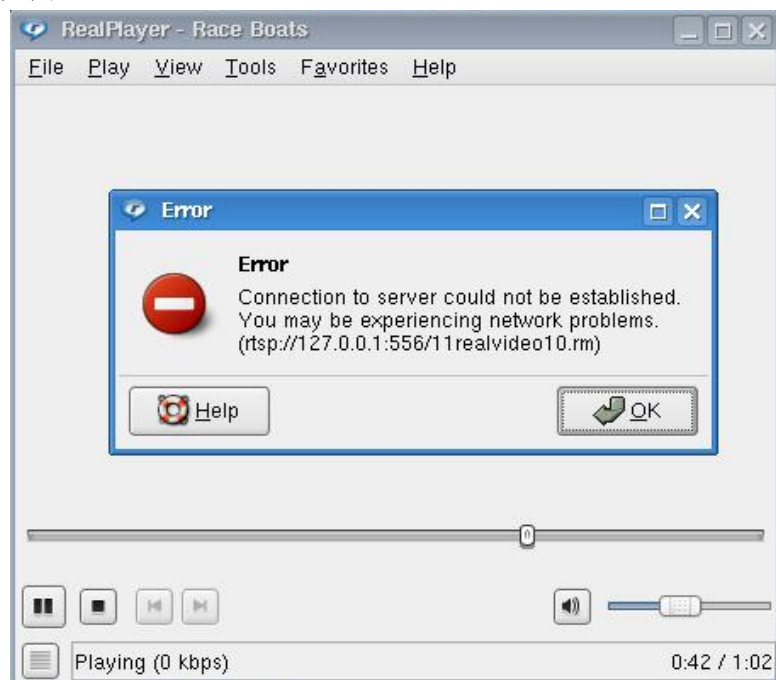


图 9 关闭冗余服务器（端口 556）后 RealPlayer 连接失败

### 3.3 功能扩展 2 的实现

虽然冗余服务器可以提高系统的健壮性，但是要在每台服务器上保存相同的备份，在主服务器进行正常工作时，冗余服务器空置，资源利用率只有 50%，为了提高利用率，我们将主/冗余服务器互相冗余。这不仅可以提高资源利用率，还可以在较多用户访问时，借助于网页计数器技术实现负载均衡功能（选择用户请求较少的一台服务器进行服务）。当然在实际使用中，我们并不建议只用两台，商业应用中，一般至少设置三台服务器，两台提供服务，一台提供冗余，资源利用率是 67% 左右。

由于主服务器冗余功能已经配置完成，下面只配置冗余服务器就可以了。可以通过控制页面也可以直接修改配置脚本来实现。在此，我们使用后者。（以后我们称主服务器为服务器 1，冗余服务器为服务器 2）。

（服务器 2 的冗余功能配置）

```
<!-- REDUNDANT SERVERS -->
<List Name="ServerAlternates">
  <List Name="Alternates">
    <List Name="Master">
      <Var Host="127.0.0.1"/>
      <Var Port="554"/>
    </List>
  </List>
</List>
<List Name="RedirectRules">
  <List Name="100">
    <List Name="Alternates">
      <Var 100="Master"/>
    </List>
    <Var Rule="/" />
  </List>
</List>
<List Name="ExcludePaths">
  </List>
</List>
```

服务器 1 冗余功能测试：重启服务器 1 和服务器 2。

启 RealPlayer 11，选择 file -> open location -> rtsp://127.0.0.1:554/11realvideo10.rm，可以看到正常播放画面，然后在服务器 1 的系统控制台按 **Ctrl+c** 关闭服务器 1，保留服务器 2 继续运行，可以看到播放器在经过短暂缓冲后继续播放，此时再关闭服务器 2，那么 RealPlayer 11 将弹出请求资源不存在的警告对话框，同图 9。

服务器 2 冗余功能测试：重启服务器 1 和服务器 2。

启 RealPlayer 11，选择 file -> open location -> rtsp://127.0.0.1:556/11realvideo10.rm，可以看到正常播放画面，然后在服务器 2 的系统控制台按 **Ctrl+c** 关闭服务器 2，保留服务器 1 继续运行，可以看到播放器在经过短暂缓冲后继续播放，此时再关闭服务器 1，那么 RealPlayer 11 将弹出请求资源不存在的警告对话框。区别服务器 1 还是服务器 2 通过 rtsp 端口号（554/556）。

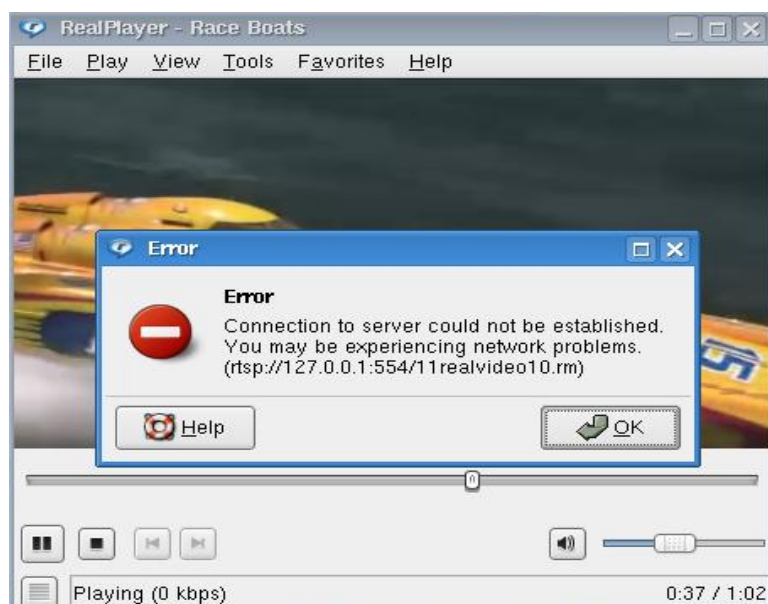


图 10 关闭冗余服务器（端口 554）后 RealPlayer 连接失败

3.4 功能扩展 3 的实现

功能扩展 2 实现了系统冗余和负载均衡，基本上可以满足服务请求了，但是他还面临了一个重要问题就是需要在两台服务器上进行文件存储，这就消耗了大量磁盘空间，这在存储大量媒体文件的情况下是很不经济的。因此我们决定在功能扩展 2 的基础上再利用 Helix 自带的传送服务器功能增加一台传送服务器用来广播或者单播流媒体，同时将服务器 1 和服务器 2 设置为接收服务器。

首先将服务器 1 和服务器 2 设置为接收服务器。接收服务器配置参数见图。

RealNetworks

运行中的服务器: 127.0.0.1

服务器设置

安全设置

日志和监控

广播设置

广播分发

传送服务器

接收服务器

可扩展多播

后台多播

Windows Media多播

会话声明

内容管理

接收服务器

帮助

加载点\*

广播传送服务器

Simulation

编辑传送服务器名

Simulation

传送服务器地址

127.0.0.1

传送服务器子网掩码

32 Bits (255.255.255.255 or /32)

端口范围

30020 to 30040

传输方式

udp/unicast

多播地址

重发请求

No

安全类型

None

设置密码

确认密码

启用Pull 分发请求

No

Pull 分发虚拟路径

连接错误率

20

数据元传输率

30

Pull 分发后台传输

UDP

\*以上字段的更改将在服务器重新启动以后生效.

应用 重置

图 11 接收服务器的配置页面

服务器 1 对应的配置脚本

```
<!-- BROADCAST DISTRIBUTION -->
<!-- TRANSMITTER -->
<List Name="BroadcastDistribution">
  <Var SourceName="localhost.localdomain"/>
</List>
<!-- RECEIVER -->
<List Name="BroadcastReceiver">
  <List Name="Receivers">
```

```

<List Name="Simulation">
  <Var UseTCPForPullBackchannel="0"/>
  <Var Protocol="udp/unicast"/>
  <Var PortRange="30001-30020"/>
  <Var OriginSpec="127.0.0.1/32"/>
  <Var PullSplitEnabled="0"/>
  <Var FECLevel="20"/>
  <Var ResendSupported="0"/>
  <List Name="Security">
    <Var Type="None"/>
  </List>
  <Var AcquisitionDataInterval="30"/>
</List>
</List>
</List>

```

从本地 30001-30020 端口接收单播

服务器 2 对应的配置脚本

```

<!-- BROADCAST DISTRIBUTION -->
<!-- TRANSMITTER -->
<List Name="BroadcastDistribution">
  <Var SourceName="localhost.localdomain"/>
</List>
<!-- RECEIVER -->
<List Name="BroadcastReceiver">
  <List Name="Receivers">
    <List Name="Simulation">
      <Var UseTCPForPullBackchannel="0"/>
      <Var Protocol="udp/unicast"/>
      <Var PortRange="30021-30040"/>
      <Var OriginSpec="127.0.0.1/32"/>
      <Var PullSplitEnabled="0"/>
      <Var FECLevel="20"/>
      <Var ResendSupported="0"/>
      <List Name="Security">
        <Var Type="None"/>
      </List>
      <Var AcquisitionDataInterval="30"/>
    </List>
  </List>
</List>
</List>

```

从本地 30021-30040 端口接收单播

然后是传送服务器的设置。Helix 的传送服务器有两种工作模式：基本模式（Base）和高级模式（Advance）。运行于基本模式的传送服务器只能向一台服务器广播或者单播传送媒体流，高级模式并不限制分发服务器的数目。由于我们需要服务器 1 和服务器 2 实现数据冗余和负载均衡，所以我们选择在高级模式运行传送服务器。数据传输用 udp/unicast 模式。发送服务器配置参数如下：



图 12 传送服务器的配置页面

传送服务器对应的配置脚本  
(端口部分)

```
<!-- PORTS -->
<!--UNIX customers must have root privileges to execute the server -->
<!--with the RTSP port set to 554. -->
<!--The following are the default ports that RealPlayer and -->
<!--RealPlayer Plus clients will connect to for an URL that has -->
<!--no port specified: -->
<!-- RTSP: 554 -->
<!-- HTTP: 80 (...then 8080 if 80 is unavailable) -->
<!-- MMS: 1755 -->
<Var RTSPPort="558"/>
<Var HTTPPort="80"/>
<Var MMSPort="1740"/>
<Var MonitorPort="9080"/>
```



```
<Var AdminPort="15246"/>
<Var ChannelControlPort="8004"/>
```

(传送服务器部分)

```
<!-- BROADCAST DISTRIBUTION -->
<!-- TRANSMITTER -->
<List Name="BroadcastDistribution">
  <Var SourceName="localhost.localdomain"/>
  <List Name="Destinations">
    <List Name="Master">
      <Var AcquisitionDataInterval="30"/>
      <Var FECLevel="20"/>
      <Var PathPrefix="/broadcast"/>
      <Var RelayMode="0"/>
      <Var PortRange="30001-30020"/>
      <Var ResendSupported="1"/>
      <Var Protocol="udp/unicast"/>
      <Var LocalAddress="127.0.0.1"/>
      <Var SureStreamAware="1"/>
      <Var Address="127.0.0.1"/>
      <List Name="Security">
        <Var Type="None"/>
      </List>
      <Var TTL="16"/>
    </List>
    <List Name="Redudant">
      <Var PortRange="30021-30040"/>
      <Var Protocol="udp/unicast"/>
      <Var Address="127.0.0.1"/>
      <List Name="Security">
        <Var Type="None"/>
      </List>
      <Var PathPrefix="/broadcast"/>
      <Var TTL="16"/>
      <Var FECLevel="20"/>
      <Var ResendSupported="1"/>
      <Var LocalAddress="127.0.0.1"/>
      <Var SureStreamAware="1"/>
      <Var RelayMode="0"/>
      <Var AcquisitionDataInterval="30"/>
    </List>
  </List>
</List>
```

向本地 30001-30020 端口发送单播

向本地 30021-30040 端口发送单播

设置好基本的发送服务器以后，并不能直接向接收服务器广播或者单播媒体流，还需要编辑一个配置脚本。首先在传送服务器主目录 /usr/local/Helix\_transmitter/ 下建立一个子目录，命名为 simulate，然后将主目录下的 slta.cfg、其子目录 Bin/ 下的 slta.sh、以及将要传送的媒体文件拷贝到 simulate 目录中，同时在此目录下建立一个 txt 文件，命名为 playlist.txt。商业模式下应该在单独一台主机或者服务器上保存 simulate 目录和执行 slta.sh 命令。Slta.cfg 是一个配置模板，将其重命名为 transmit.cfg，然后编辑。

```
<!-- Example Configuration file for SLTA -->
<!-- Change these entries to support your configuration -->

<List Name="BroadcastDistribution">
  <Var SourceName="Simulation"/>
  <List Name="Destinations">
    <List Name="Master">
      <Var PathPrefix="*/>
      <Var PortRange="30001-30020"/>
      <Var AcquisitionDataInterval="30"/>
      <Var FECLevel="0"/>
      <Var SureStreamAware="0"/>
      <Var BufferlessTransport="1"/>
      <Var LocalAddress="0.0.0.0"/>
      <Var Address="127.0.0.1"/>
      <Var TTL="16"/>
      <Var ResendSupported="0"/>
      <Var Protocol="udp/unicast"/>
      <List Name="Security">
        <Var Type="None"/>
      </List>
      <!-- Var Password="ExamplePassword"/ -->
```

Master 服务器配置部分

```

</List>
</List>

<List Name="Redudant">
  <Var PathPrefix="*" />
  <Var PortRange="30021-30040" />
  <Var AcquisitionDataInterval="30" />
  <Var FECLevel="0" />
  <Var SureStreamAware="0" />
  <Var BufferlessTransport="1" />
  <Var LocalAddress="0.0.0.0" />
  <Var Address="127.0.0.1" />
  <Var TTL="16" />
  <Var ResendSupported="0" />
  <Var Protocol="udp/unicast" />
  <List Name="Security">
    <Var Type="None" />
    <!-- Var Password="ExamplePassword" / -->
  </List>
</List>

</List>
<List Name="Pull Settings">
  <List Name="Transmittter1">
    <List Name="Security">
      <Var Type="None" />
      <!-- Var Password="ExamplePassword" / -->
    </List>
    <Var SureStreamAware="0" />
    <Var ListenPort="2030" />
    <Var PathPrefix="/" />
    <Var LocalAddress="0.0.0.0" />
  </List>
</List>
</List>

```

Redundant 服务器配置部分

编辑播送列表 `playlist.txt`，将要播送的文件列入其中，每行一个项目。要求列表中的项目必须有相同的文件格式和码率，如果文件不在同一个目录中，应该用绝对路径或者相对于 `playlist.txt` 的路径表示其具体位置。例如：

```

CompanyLogo.rm
Welcome.rm
President.rm
Treasurer.rm
Conclusions.rm

```

配置结束后启动传送服务器、服务器 1 和服务器 2，在 `simulate/` 目录下执行如下命令：

```
slta.sh -c transmit.cfg stream_name playlist.txt | clipname
```

```

root@localhost:/media/disk/Helix_transmitter/simulate - Shell - Konsole
Session Edit View Bookmarks Settings Help

[root@localhost simulate]# ./slta.sh -c lshta.cfg live.rm realvideo10.rm
SLTA - Live Broadcast Simulation Utility

Executing SLTA with
config_file=lshta.cfg, live_file=live.rm, playlist_file=realvideo10.rm

SLTA (c) 2001-2008 RealNetworks, Inc. All rights reserved.
Adding TAC event stream
Transmitting file:///media/disk/Helix_transmitter/simulate/realvideo10.rm...
0---1---2---3---4---5---6---7---8---9---10
****

```

图 13 开启 `slta.sh` 进行流媒体传输

至此，整个项目设计完成，在 RealPlayer 11 中输入如下地址：

`rtsp://127.0.0.1:554/broadcast/Simulation/live.rm`

则可播放正在播送的视频节目，如果输入

rtsp://127.0.0.1:556/broadcast/Simulation/live.rm

两台服务器均可以提供服务。

如果 **Ctrl+c** 关闭服务器 1，RealPlayer 会自动连接到服务器 2 进行播放。这里假设用户可以根据服务器的连接请求数自动连接客户较少的服务器实现负载均衡，商业应用中，则可以利用网页计数器实现此功能。

#### 4 相关技术比较和分析

除了 Helix 以外，Linux 下的流媒体服务器还有 gnum3d，由 GNU 组织开发，小巧，稳定，但是跟 Helix 相比，功能并不全面，主要是用来在小型局域网中提供音频服务。

Helix Server 是 Real 公司开发的流媒体服务器软件，功能完备，具有跨平台特性，用 RealPlayer 作为客户端可以发挥其最大优势。但是在本设计中用到的冗余服务器功能不支持 9.0 以前版本的 RealPlayer，也就是说万一主服务器掉线，RealPlayer 9.0 以前版本的客户端是无法续传的。

本设计根据 Helix Server 已有功能进行了扩展配置，在有些功能的实现上，还略显粗糙，有待进一步优化。有些已经考虑加入的功能，由于时间和技术的问题，还没有实现。

#### 5 总结

本次设计作为 2008 “红旗杯” 知识竞赛的复赛作品，是在全组成员的共同努力下完成的。虽然接触 Linux 已经有一段时间了，但是组队完成一个实用项目还尚属首次。感谢队员们对本团队的热情支持和本项目的积极投入，利用自己有限的课余时间，从选材，查阅资料，到整个系统的设计调试，无不体现了他们的智慧和汗水。还要感谢陈光喜老师对本设计在项目的选择、方案的确定以及说明书的编写等方面给予的热情指导。

**未完成功能：**系统通过计算主/从服务器的用户连接数，自动为用户选择负载较低的服务器，实现真正意义上的负载均衡。

#### 参考资料

Helix Server 说明文档，© 2002 RealNetworks, Inc. All rights reserved。

#### 指导老师意见

基本能够实现预定功能，具有一定的实用性，创新点不够突出。

分工明确，在整个设计过程中较好地实现了团队协作。

指导教师签名： 陈光喜

2008 年 11 月 20 日

#### 成员列表

姓名	分工	手机	电子邮件	通信地址
臧运港	系统裁减、设计、制作	13297734576	zangyungang@gmail.com	桂林电子科技大学 ES06 信箱
郑少鹏	系统设计、测试	13737737647	zheng54035403@126.com	桂林电子科技大学 ES06 信箱
许海兵	系统设计、测试	15977346837	xhb_forever@163.com	桂林电子科技大学 A312 信箱

**系统裁减列表**  
(基本安装, 3 张光盘)

RPM 组	裁减内容
Basesystem-optional	defaults
Base-X	defaults
Chiese-support	defaults
Core	defaults
Database-tools	all
Development	alsa-lib-dev auto* cpp cvs cyrus* DB4* dbus* dhcp docbook* fontconfig-devel freetype-devel gcc* gdb gettext* ghostscript gmp gnome* gphoto2 gpgme* jdk* lcm* ldapjdk* kernel-doc libbonbo libgcj-devel libgnat libgnome*-dev libgnome-print net-*-devel
Japanese-support	all
Gnome-libs	defaults
Kdedesktop	defaults
Legacy-software-development	all
Middle-ware-geronimo	all
Middle-ware-resin	all
Middle-tomcat	all
Others	defaults
Printing	all
Server	all
System-tools	defaults
Virtualization	all
Web-browsers	Only left firefox