

深入理解 LoadRunner 的测试结果

李 康

概要

本文的目的是为了使 LoadRunner 用户能够深入了解 LoadRunner 测试结果,并为通过了解性能数据的存储方式,从而为第三方集成 LoadRunner,扩展 LoadRunner 使用而提供帮助,这里所提到的信息主要包括 3 部分内容:

- a) 测试场景信息
- b) 虚拟用户的交易统计
- c) 服务器监控数据统计

本文读者

本文所面向的主要是对 LoadRunner 的性能数据采集有深入了解的应用性能工程师,或者希望把 LoadRunner 的测试结果集成到其企业自身的数据表现平台的开发人员。

在阅读本文之前,确认您已经具备以下的相关知识:

- LoadRunner 的架构和测试执行流程
- LoadRunner 脚本开发和交易类型
- LoadRunner 实时监控器和相关设定
- LoadRunner 交易类数据收集和相关设定
- LoadRunner 用户性能数据收集方法

假设

- 为了方便说明,作者假设 LoadRunner 的测试结果已被存放在了 *C:\Sample\Results* 目录中,并被取名为 *Result1*。
- 同样 LoadRunner 的安装目录为 *C:\Mercury\LoadRunner*,在下文中以 %LR% 说明。

更多信息

请参考 LoadRunner 在线帮助文档,或浏览 <http://www.mercury.com>。或直接联系作者 (E-Mail: kang.li@mercury.com ; MSN:expatriate_lee@msn.com)

测试场景信息

显然，LoadRunner 测试场景的结果将被存放在一个目录中。

这个目录就是以测试结果的名字而命名：

RESULT_DIR=*C:\Sample\Results\Result1*

测试结果的主文件名为：

RESULT_FILE=*C:\Sample\Results\Result1\Result1.lrr*

所有的 LoadRunner 测试结果的主文件都是以 **.lrr** 为文件扩展名的，这个文件是一个类 INI 文件的格式，其中包括了测试场景执行时的重要信息：

```
[Scenario]
Time_Zone=#
Start_time=#
Daylight_Bias=#
Stop_time=#
```

下面是一个具体的例子：

```
[Scenario]
Time_Zone=28800
Start_time=1075066329
Daylight_Bias=0
Stop_time=1075066555
```

请注意其中时间都是以 C 语言的 `time_t` 所兼容的长整型所定义，即从 1970 年 1 月 1 日 0 点以来的秒数，在这里表示为测试开始时间是：2004 年 1 月 25 日 13:32:09，测试结束时间是：2004 年 1 月 25 日 13:35:55。

虚拟用户交易性能统计

在 LoadRunner 的测试结果中，所有的虚拟用户的执行结果都被保存在了以 *.EVE* 为扩展名的文件中。这些文件被列举在了 Remote_results.txt 中（如本文的例子即为 *C:\Sample\Results\Result1\remote_results.txt*）。关于 Remote_result.txt 的文件格式为：

```
[Hosts]
<Hostname1>=<Path_to_eve_file>
<Hostname2>=<Path_to_eve_file>
...
<HostnameN>=<Path_to_eve_file>
```

下面是一个实例：

```
[Hosts]
localhost=C:\Sample\Results\Result1\localhost_1.eve
ibmt40= C:\Sample\Results\Result1\ibmt40_1.eve
```

所有的 .EVE 文件被存放在 *%RESULT_DIR%* 中，比如上面的例子中，表示在“Localhost”上执行的虚拟用户结果被保存在了 *%RESULTDIR%\localhost_1.eve* 中。这些文件是以二进制形式存放的。在 LoadRunner 的预处理过程中，它们将会被转换成平文本格式。同样可以利用在本文中所附的工具做这样的转换，具体过程如下：

安装转换工具（笔者在 LoadRunner 7.8 环境中验证通过，不确定在 8.0 及其后版本中的可用性）

- 拷贝相关文件至 %LR% 所对应的目录中

- 转换 EVE 文件为文本格式

通过命令行方式（CMD.EXE）执行 *%LR%/BIN* 下的 *cmd_reader_eve_file*，命令行格式如下：

```
cmd_reader_eve_file    <EVE_FILE_NAME>    <OUTPUT_DIR>    <FLUSH_FLAG>
<VERTICAL_FLAG>
```

具体实例如下：

```
Cmd_reader_eve_file          C:\Sample\Results\Result1\localhost1.eve
C:\Sample\Result\Result1\eve_output 1 0
```

值得注意的是，LoadRunner 没有对其中的参数（Flag）作出解释，在此处的 VERTICAL_FLAG 为 0 表示每一数据行对应一个交易（Transaction）或一个数据点（Data Point）入口。

作为 *cmd_reader_eve_file* 的输出，LoadRunner 将在 *<OUTPUT_DIR>* 目录中产生 *eve_text.log* 和 *read_eve.log* 文件：

- 1) *eve_text.log* 包括了所有在输入的 .EVE 文件中的交易性能统计数据或数据点信息。

2) read_eve.log 则是 cmd_reader_eve_file 的执行日志，包括了其状态。

- 进一步理解 eve_text.log 文件

该文件的基本格式可能有 2 种，并且这 2 种格式都是有效格式。

格式 1: <type> <starttime> <endtime> <transaction_name> <trans_handle>
<parent_trans_handle> <think_time> <wasted_time> <status> <vuser_id> <group_name>
<scenario_id> <script_name> <trans_type>

各字段的意义如下：

<type>	=1 固定，交易数据类型
<starttime>	交易开始时间
<endtime>	交易结束时间
<transaction_name>	交易名称
<trans_handle>	交易句柄
<parent_trans_handle>	父交易句柄
<think_time>	总思考时间（由用户定义的 lr_think_time 或 sleep 函数产生）
<wasted_time>	总耗费时间（用于 LoadRunner 生成请求和处理响应时间等耗损时间）
<status>	交易状态，正常结束时为 0
<vuser_id>	虚拟用户 ID
<group_name>	用户组名
<scenario_id>	测试场景 ID
<script_name>	测试脚本名称
<trans_type>	交易类型，其中 1 为用户定义交易，2 为自动交易，3 为脚本模块或 LoadRunner 函数交易

其中典型的交易响应时间为：

交易时间=交易结束时间-交易开始时间-总思考时间-总耗费时间

- 由 lr_start_transaction/lr_end_transaction 函数生成的交易的交易类型<trans_type>=1
- 由脚本 API 定义或通过运行时设定的交易的交易类型<trans_type>=2
- 由脚本或 LoadRunner 函数所定义交易（如 vuser_init, vuser_end 等）的交易类型<trans_type>=3

以下是这类数据格式的实例：

```
1 1075066330.631081 1075066330.631413 vuser_init_Transaction 1,1 0,0 0.000000
0.000000 0 1 sample_user 1 sample_user 3
1 1075066330.636429 1075066331.944108 LOGON 1,3 1,2 1.294478 0.000000 0 1
sample_user 1 sample_user 1
1 1075066340.345697 1075066340.348619 SUBMIT_REQUEST 1,12 1,2 0.000000 0.000000 0
1 sample_user 1 sample_user 1
```

```

1 1075066343.224992 1075066343.227754 homepage_Action_21 1,16 1,15 0.000000
0.000000 0 1 sample_user 1 sample_user 2
1 1075066343.224884 1075066343.227792 LOGOFF 1,15 1,2 0.000000 0.000000 0 1
sample_user 1 sample_user 1
1 1075066330.636378 1075066343.227842 Action_Transaction 1,2 0,0 12.572037
0.000000 0 1 sample_user 1 sample_user 3

```

格式 2: <type> <timestamp> <datapoint_name> <handle> <parent_handle>
<value> <vuser_id> <group_name> <scenario_id> <script_name> <dpoint_type>

各字段的意义如下表

<type>	=2 固定，数据点类型
<timestamp>	数据点采集的时间戳
<name>	数据点名称
<handle>	数据点句柄
<parent_handle>	父数据句柄
<value>	数据点值
<vuser_id>	虚拟用户 ID
<group_name>	用户组名称
<scenario_id>	测试场景 ID
<script_name>	脚本名称
<dpoint_type>	数据点类型

数据点可以有 LoadRunner 的其他协同模块收集，也可以由比如 lr_user_data_point 等函数产生。

以下是该格式的实例：

```

2 1075066343.227884 HTTP_200 1,18 0,0 2.000000 1 sample_user 1 sample_user 1
2 1075066345.350112 mic_recv 1,19 0,0 14705.000000 1 sample_user 1 sample_user 1

```

服务器监控性能统计

在 LoadRunner 中，几乎所有的服务器信息是通过 LoadRunner Realtime Monitor 来采集的。这些采集的信息最终将被存放在 %RESULT_DIR%\offline.dat 中。

offline.data 的一般格式为：<datapoint_alias> <timestamp> <value>

具体实例如下：

```
offline_datapoint_0 1075066332 69705728.000
offline_datapoint_1 1075066330 4094.388
offline_datapoint_2 1075066330 412168.350
offline_datapoint_11 1075066330 22.000
offline_datapoint_3 1075066330 0.000
```

这些数据点所代表的实际监控指标被定义在了统一目录（%RESULT_DIR%）中的各自的 offl_*.def 文件中。每个 offl_*.def 都代表着一个性能指标。Offl_*.def 是一个 INI 格式的文件，其中的 [Graph definition] 定义块详细描述了该性能指标。

GraphGroupMenuTitle	性能图表（组）的名称
GraphTitle	性能指标名称
XAxisTitle	X 轴名称
YAxisTitle	Y 轴名称
XAxisIsElaspsedTime	TRUE/FALSE, X 轴是否为时间
LineType	NOSTEP, 固定
BuildUnderLoadGraph	TRUE/FALSE
AggregateFunction	汇总类型，如 SUM，AVG 等
MissingData	
GranularityMode	粒度模型，比如 NOTINCLUDEZERO
NoDataBehavior	
GraphType	
Count=#	数据点数量
DataPointLabel_#	数据点别名
LineTitle_#	Host 名称，数据点源
LineGroup_#	
DataPointDescr_#	数据点的描述

下面是一个 offl_*.def 文件的实例：

```
[Graph definition]
count=1
GraphTitle=% Processor Time (Process aspnet_wp)
DataPointLabel_1=offline_datapoint_3
LineTitle_1=localhost
```

```
LineGroup_1=0
YAxisTitle=% Processor Time (Process aspnet_wp)
GraphGroupMenuTitle=Resource Monitoring
XAxisTitle=Elapsed Scenario Time (seconds)
XAxisIsElapsedTime=TRUE
LineType=NOSTEP
BuildUnderLoadGraph=TRUE
AggregateFunction=AVG
MissingData=PREVIOUS
GranularityMode=NOTINCLUDEZERO
NoDataBehavior=RemoveGraph
GraphType=es_rm_svr_res_nt
```

有关 GraphType 的定义,请参考 LoadRunner 安装目录下的 lr_ext 目录中的相关信息。比如, es_rm_svr_res_nt 对应 Windows Resources; es_rm_svr_rstat_unix 对应 UNIX Resources 等。