# ****一概念理解****

## 1．性能测试目的?

答：验证软件系统是否能够达到用户提出的性能指标。

性能测试是通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。

1） 评估系统的能力----测试中得到的负荷和响应时间数据可被用于验证所计划的模型的能力，并帮助作出决策。

2） 识别体系中的弱点----受控的负荷被增加到一个极端水平，并突破它，从而修复体系的瓶颈或薄弱的地方。

3） 系统调优---重复运行测试，验证调整系统的活动得到了预期的结果，从而改进性能。检测软件中的问题，长时间的测试执行可导致程序发生由于内存泄漏引起的失败，揭示程序中的隐含问题或冲突。

4） 验证稳定性，可靠性---在一个生产负荷下执行测试一定的时间是评估系统稳定性和可靠性是否满足要求的唯一方法。

## 2．系统实际用户数，系统在线用户数含义?

用户数：是指计费系统所能允许记录的不同名称用户数量的最大值。

这个数值取决于计费系统硬件存储器容量和软件的支持能力

**系统实际用户数**：系统额定的用户数量，如一个OA系统，可能使用该系统的用户总数是2000个，那么这个数量，就是系统用户数

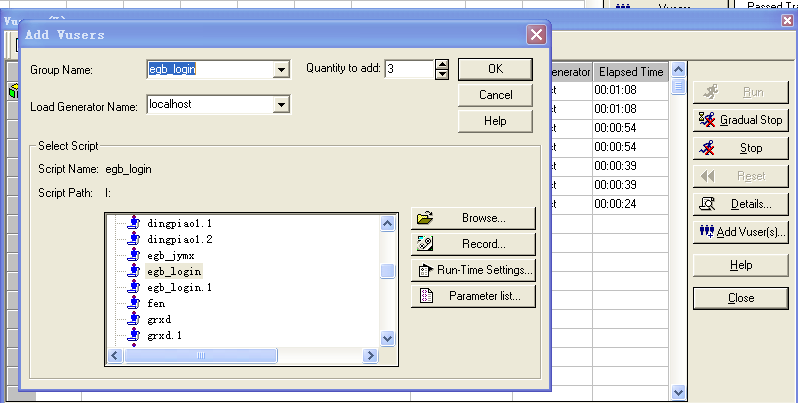
**系统在线：**在一定的时间范围内，同时在线用户数量

## 3．并发概念？

答：并发是同时执行一个操作（同时像服务器提交申请）。主要指当测试多个用户并同时访问同一个应用程序、同一个模块数据记录时是否存在死锁或其他性能问题，几乎所有的性能测试都会涉及并发测试。

## 4．理解负载测试，压力测试，容量测试，配置测试，基准测试，并发测试，疲劳测试的含义和区别

答：**负载测试（Load testing）**，负载测试是模拟实际软件系统所承受的负载条件的系统负荷，通过不断加载（如逐渐增加模拟用户的数量）或其它加载方式来观察不同负载下系统的响应时间和数据吞吐量、系统占用的资源（如CPU、内存）等，以检验系统的行为和特性，以发现系统可能存在的性能瓶颈、内存泄漏、不能实时同步等问题。直接添加用户数双击Down

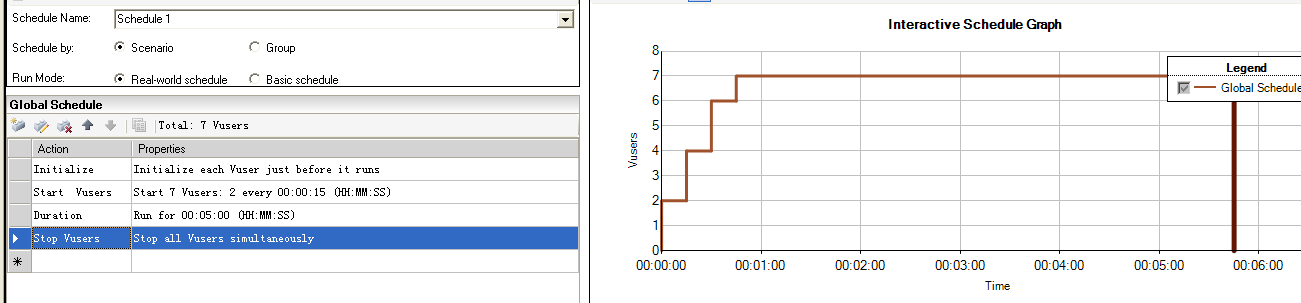
-点击Add Vuser(s)-点击Quantity to add输入框输入要添加的用户数，在原基础上添加用户。

**压力测试：**压力测试是在强负载（大数据量、大量并发用户等）下的测试，查看应用系统在峰值使用情况下操作行为，从而有效地发现系统的某项功能隐患、系统是否具有良好的容错能力和可恢复能力。（压力测试就是不断的向服务器增加用户数，直到服务器返回响应时间过长。）Initialize 5 Vusers every 00:00:15 (HH:MM:SS)（初始化每5个Vuser00:00:15（HH：MM：SS）），

Start Vusers Start 7 Vusers:2every 00:00:15(HH:MM:SS)（开始Vuser的启动7个用户：每15秒增加2个用户

Duration Run for 00:05:00（程序运行时间00:05:00）

Stop Vusers Stop all Vusers simultaneously（停止Vuser的Vuser的同时停止所有）

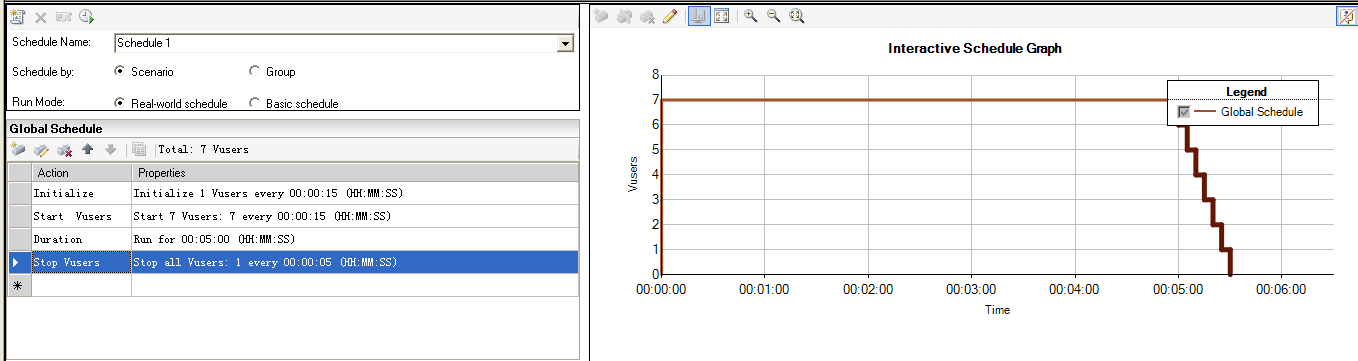


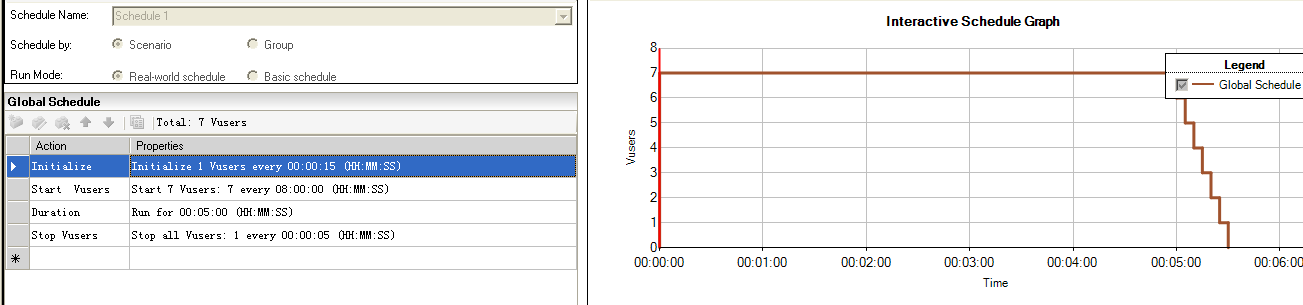
**容量测试（界限或极限值）**：通过测试预先分析出反映软件系统应用特征的某项指标的极限值（如最大并发用户数、数据库记录数等），系统在其极限值状态下没有出现任何软件故障或还能保持主要功能正常运行。容量测试还将确定测试对象在给定时间内能够持续处理的最大负载（负载和或压力）的来的，和工作量。

**配置测试：**就是查看配置项的内容是不是符合标准，也就是说是不是配置项起到了先前所预期的功能。个人计算机；部件；外设；接口；可选项和内存；设备驱动程序。

**基准测试：**是指通过设计科学的测试方法、测试工具和测试系统，实现对一类测试对象的某项性能指标进行定量的和可对比的测试。

**并发测试：**主要指当测试多个用户并同时访问同一个应用程序、同一个模块数据记录时是否存在死锁或其他性能问题，几乎所有的性能测试都会涉及并发测试。



**疲劳测试：**通常是采用系统稳定运行情况下能够支持的最大并发用户数或者日常运行用户数，持续执行一段时间业务，通过综合分析交易执行指标和资源监控指标来确定系统处理最大工作量强度性能的过程（内存溢出）场景结束，没有释放内存（前提要清理内存）。

**总结**压力测试可以看作是容量测试，不是直接的测试目标。压力测试的重点在于发现功能性测试所不易发现的系统方面的缺陷。而容量测试是系统测试的主要目标内容，也就是确定软件产品或系统的非功能性方面的质量特征，包括具体的特征值。容量测试更着力于提供性能与容量方面的数据，为软件系统部署、维护、质量改进服务，并可以帮助市场定位、销售人员对客户的解释、广告宣传等服务。压力测试、容量测试，测试的方法相似、相通，在实际测试工作中，往往结合起来进行，以提高测试效率。一般会设置专门的性能测试实验室，完成这些工作。即使用虚拟的手段模拟实际操作，所需要的客户端有时还是很大的，所以性能测试实验室的投资较大。（并发测试（获得最佳用户数和最大用户数）是为疲劳测试做准备）

## 5．性能测试指标包括哪些，每项的含义是什么？

答：I/O（vmstat、iostat）、CPU(top)、内存（sar）、事务通过数、事务的响应时间（ART）、交易成功率、事务点击数、用户数）

## 6．怎么理解响应时间是从什么时候开始什么时候结束？

答：（是通过记录用户请求的开始时间和服务器返回内容到客户时间的差值来计算用户操作响应时间）响应时间是提交请求和返回该请求的响应之间使用的时间。

## 7．每秒处理请求数，响应时间，点击率的缩写？

答：事务的请求数（TPS),平均响应时间（ART),点击率（HPS）

## 8．什么是事务？

答：事务是指用户在客户端做一种或多种业务的操作集，通过事务函数可以标记完成该业务所需要的操作内容（响应时间等）就是把多件事情当做一件事情来处理叫做事物.

## 9．什么是性能调优，什么情况下需要调优？

答：通过指标的监控发现系统存在的性能缺陷，利用分析工具定位并修正性能问题。系统响应速度太慢。 每秒所完成的系统输出/入低于预期。 相同的环境，但每秒钟所完成的批操作较先前少。 系统资源（如CPU、内存、硬盘或网络等）长时间处于耗尽的状态。（环境、基准负载、性能指标响应时间多少S）

## 10．性能测试通用流程？

答：性能测试需求(性能指标)

环境搭建（模拟现实环境）数据、

工具选择（性能

性能方案计划文档

场景设计

脚本设计

测试执行

结果分析（page/sec内存），CPU（top),内存（sar),I/O(iostat)

调优方案(代码级调优，数据库调优，架构调优)

性能测试报告

在本公司写好的脚本，到现场要改脚本的地方是（1.脚本请求的地址，证书（握手证书和加密证书）象EWP人员要）

## 11．怎么理解协议，http是指什么协议？

答：协议：计算机通信网络中两台计算机之间进行通信所必须共同遵守的规定或规则。httP:超文本传送协议 (HTTP) 是一种通信协议，它允许将超文本标记语言 (HTML) 文档从 Web 服务器传送到 Web 浏览器。（双方共用的协议）

## 12．分析简单http请求和http应答的报文体？

答：http请求是从客户端到服务器端的请求消息包括，消息首行中，对资源的请求方法、资源的标识符及使用的协议。http应答的报文体是xml格式的utf-8编码的字节流报文具体的内容包含在某个标签内

web\_custom\_request("EMP\_HandShake", "Method=POST",

"URL=http://{HostAdress}/user/handshake?ota\_version=AD-UMP-1.1.1-000000&app={AppName}&o=i",

"RecContentType=application/octet-stream",

"Body={ClientHelloBody}",

LAST);

**协议：通行双方共同约定的规则。**

**http协议（超文本传输协议）：**HTTP是一个属于应用层的面向对象的协议，由于其简捷、快速的方式，适用于分布式超媒体信息系统。

**HTTP协议的主要特点可概括如下：**

1.支持客户/服务器模式。

2.简单快速：客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有GET、HEAD、POST。每种方法规定了客户与服务器联系的类型不同。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快。

3.灵活：HTTP允许传输任意类型的数据对象。正在传输的类型由Content-Type加以标记。

4.无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。

5.无状态：HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。

**HTTP报文结构**：

http请求报文体：方法-URL-协议/版本、请求头、请求正文

http应答报文体：协议-状态代码-描述、应答头、应答正文

**HTTP请求流程：**

建立连接（客户端与服务器建立TCP连接）

发送请求：（打开一个连接后，客户端把请求信息发送到服务器的相应端口上，完成请求动作提交）

发送响应：（服务器在处理完客户端请求之后，要向客户端发送响应消息）

关闭连接：(客户端和服务器端都可以关闭套接字来结束TCP/IP对话)

**请求方法（所有方法全为大写）有多种，各个方法的解释如下：**

GET 请求获取Request-URI所标识的资源

POST 在Request-URI所标识的资源后附加新的数据

HEAD 请求获取由Request-URI所标识的资源的响应消息报头

PUT 请求服务器存储一个资源，并用Request-URI作为其标识

DELETE 请求服务器删除Request-URI所标识的资源

TRACE 请求服务器回送收到的请求信息，主要用于测试或诊断

CONNECT 保留将来使用

OPTIONS 请求查询服务器的性能，或者查询与资源相关的选项和需求

**GET和POST请求的区别**

（1） 在客户端，Get方式在通过URL提交数据，数据在URL中可以看到；POST方式，数据放置在HTML HEADER内提交。

（2）GET方式提交的数据最多只能有1024字节，而POST则没有此限制。

（3）安全性问题。正如在（1）中提到，使用 Get 的时候，参数会显示在地址栏上，而 Post 不会。所以，如果这些数据是中文数据而且是非敏感数据，那么使用 get；如果用户输入的数据不是中文字符而且包含敏感数据，那么还是使用 post为好。

（4） 安全的和幂等的。所谓安全的意味着该操作用于获取信息而非修改信息。幂等的意味着对同一 URL 的多个请求应该返回同样的结果。

**HTTP的状态响应码**

100——客户必须继续发出请求

101——客户要求服务器根据请求转换HTTP协议版本

200——交易成功

201——提示知道新文件的URL

202——接受和处理、但处理未完成

203——返回信息不确定或不完整

204——请求收到，但返回信息为空

205——服务器完成了请求，用户代理必须复位当前已经浏览过的文件

206——服务器已经完成了部分用户的GET请求

300——请求的资源可在多处得到

301——删除请求数据

302——在其他地址发现了请求数据

303——建议客户访问其他URL或访问方式

304——客户端已经执行了GET，但文件未变化

305——请求的资源必须从服务器指定的地址得到

306——前一版本HTTP中使用的代码，现行版本中不再使用

307——申明请求的资源临时性删除

400——错误请求，如语法错误

401——未授权

HTTP 401.1 - 未授权：登录失败

HTTP 401.2 - 未授权：服务器配置问题导致登录失败

HTTP 401.3 - ACL 禁止访问资源

HTTP 401.4 - 未授权：授权被筛选器拒绝

HTTP 401.5 - 未授权：ISAPI 或 CGI 授权失败

402——保留有效ChargeTo头响应

403——禁止访问

HTTP 403.1 禁止访问：禁止可执行访问

HTTP 403.2 - 禁止访问：禁止读访问

HTTP 403.3 - 禁止访问：禁止写访问

HTTP 403.4 - 禁止访问：要求 SSL

HTTP 403.5 - 禁止访问：要求 SSL 128

HTTP 403.6 - 禁止访问：IP 地址被拒绝

HTTP 403.7 - 禁止访问：要求客户证书

HTTP 403.8 - 禁止访问：禁止站点访问

HTTP 403.9 - 禁止访问：连接的用户过多

HTTP 403.10 - 禁止访问：配置无效

HTTP 403.11 - 禁止访问：密码更改

HTTP 403.12 - 禁止访问：映射器拒绝访问

HTTP 403.13 - 禁止访问：客户证书已被吊销

HTTP 403.15 - 禁止访问：客户访问许可过多

HTTP 403.16 - 禁止访问：客户证书不可信或者无效

HTTP 403.17 - 禁止访问：客户证书已经到期或者尚未生效

404——没有发现文件、查询或URl

405——用户在Request-Line字段定义的方法不允许

406——根据用户发送的Accept拖，请求资源不可访问

407——类似401，用户必须首先在代理服务器上得到授权

408——客户端没有在用户指定的饿时间内完成请求

409——对当前资源状态，请求不能完成

410——服务器上不再有此资源且无进一步的参考地址

411——服务器拒绝用户定义的Content-Length属性请求

412——一个或多个请求头字段在当前请求中错误

413——请求的资源大于服务器允许的大小

414——请求的资源URL长于服务器允许的长度

415——请求资源不支持请求项目格式

416——请求中包含Range请求头字段，在当前请求资源范围内没有range指示值，请求也不包含If-Range请求头字段

417——服务器不满足请求Expect头字段指定的期望值，如果是代理服务器，可能是下一级服务器不能满足请求长。

HTTP 500 - 内部服务器错误

HTTP 500.100 - 内部服务器错误 - ASP 错误

HTTP 500-11 服务器关闭

HTTP 500-12 应用程序重新启动

HTTP 500-13 - 服务器太忙

HTTP 500-14 - 应用程序无效

HTTP 500-15 - 不允许请求 global.asa

Error 501 - 未实现

HTTP 502 - 网关错误

[HTTP协议之长连接与短连接](http://blog.csdn.net/zhaozheng7758/article/details/6251796)

分类： [网络知识](http://blog.csdn.net/zhaozheng7758/article/category/447961) 2011-03-15 20:41 638人阅读 [评论](http://blog.csdn.net/zhaozheng7758/article/details/6251796#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/zhaozheng7758/article/details/6251796#report)

HTTP协议之长、短连接

一、长连接与短连接：

长连接：client方与server方先建立连接，连接建立后不断开，然后再进行报文发送和接收。

这种方式下由于通讯连接一直存在。此种方式常用于P2P通信。

短连接：Client方与server每进行一次报文收发交易时才进行通讯连接，交易完毕后立即断开连接。

此方式常用于一点对多点通讯。C/S通信。

二、长连接与短连接的操作过程：

短连接的操作步骤是：

建立连接——数据传输——关闭连接...建立连接——数据传输——关闭连接

长连接的操作步骤是：

建立连接——数据传输...（保持连接）...数据传输——关闭连接

三、长连接与短连接的使用时机：

长连接：短连接多用于操作频繁，点对点的通讯，而且连接数不能太多的情况。

每个TCP连接的建立都需要三次握手，每个TCP连接的断开要四次握手。

如果每次操作都要建立连接然后再操作的话处理速度会降低，所以每次操作后，下次操作时直接发送数据就可以了，不用再建立TCP连接。例如：数据库的连接用长连接，如果用短连接频繁的通信会造成socket错误，频繁的socket创建也是对资源的浪费。

短连接：web网站的http服务一般都用短连接。因为长连接对于服务器来说要耗费一定的资源。像web网站这么频繁的成千上万甚至上亿客户端的连接用短连接更省一些资源。试想如果都用长连接，而且同时用成千上万的用户，每个用户都占有一个连接的话，可想而知服务器的压力有多大。所以并发量大，但是每个用户又不需频繁操作的情况下需要短连接。总之：长连接和短连接的选择要根据需求而定。

**LoadRunner中web\_custom\_request 和 web\_submit\_data的差别**

web\_custom\_request方法可以发送POST和GET类型的请求（只能看到请求获得的数据显示出来）不能显示

web\_submit\_data只能发送POST类型的请求（加密的数据也能显示出来）

所有web\_submit\_data方法发送的请求都可以使用web\_custom\_request来实现

web\_custom\_request可以实现web\_submit\_data无法实现的请求，

比如“查询所有邮件并删除”这个案例中，查询时我们使用关联把所有邮件对应的标识抓取成一个数组，如果使用web\_submit\_data来完成这个删除的请求，需要很多个web\_submit\_data请求才能完成，但使用web\_custom\_request就可以通过一个请求完成，方法是自己写代码拼一个web\_custom\_request 方法POST请求的Body值。

## 13．什么是关联，关联怎么应用？

答：关联是指将客户端的数据与服务器端的数据建立联系，以获取服务器

端返回的动态数据。

web\_reg\_save\_param\_ex(

"ParamName=session",

"LB=<input type=hidden name=userSession value=",

"RB=>",

SEARCH\_FILTERS,

"Scope=Body",

"RequestUrl=http://127.0.0.1:1080/WebTours/nav.pl?in=home",

LAST);

## 14．添加事务的目的是什么？

答：lr\_start\_transaction（”事务名称”）,lr\_end\_transaction("事务名称", LR\_AUTO);

（通过事务监控响应时间，需要做的就是在请求的发出前添加事务开始的计数器，在请求结束的地方添加一个事务的结束计数器，VuGen会自动计算函数间的时间差。）系统操作的响应时间,验证某个功能承受压力的能力,事物来界定该功能在程序中的位置。

## 15．事务结果的状态有哪些，手工设置事务结果和自动返回事务结果是怎么设置，自动返回事务结果什么时候是正确的什么时候是错误的？

答：事务开始lr\_start\_transaction(“事务名”)；lr\_end\_transaction("事务名称", LR\_AUTO);

自动添加事务点击Insert Start Transaction（Ctrl+T）和Insert End Transaction（Ctrl+D）来添加事务。

LR\_AUTO：事物的状态被自动设置，如果事务执行成功，状态设置为PASS，如果执行失败，状态设置为FAIL,如果由于异常中断，状态被设置成STOP.

LR\_PASS：事务如果执行成功，代码的返回状态就是PASS。

LR\_FAIL：事务如果执行失败，代码的返回状态就是FAIL。

运行自动事务时，怎么变成手动设置事务函数lr\_set\_transaction\_status（）；事务结束之前。

## 16．参数化的目的是什么？

答：在脚本中，如果不使用参数，那么所有的测试数据是跟脚本绑定在一起的，如果需要测试不同的数据，需要运行一次，改一下，再运行。如果使用了参数化，可以把多个测试数据保存起来，测试时脚本自动选择测试数据运行。是向服务器批量提交符合业务逻辑的数据。

## 17．手动关联和自动关联区别，分别都怎么做？

答：**关联是**指将客户端的数据与服务器端的数据建立联系，以获取服务器端返回的动态数据(自动关联是我们为关联设置规则，可以是具体的应用程序服务，这里的数据由所创建的规则替代。在手工关联中，我们想要关联的数值被扫描并且编写关联函数完成关联。VuGen内建自动关联引擎（auto-correlation engine）,可以自动找出需要关联的值，并可以利用关联函数自动建立关联。

手动关联需要自行查找关联的对象，然后自行插入关联函数。)

**1.自动关联的**方法是,你在tools-recording options的correlation中,不选中enable correlation during recording的复选框的情况下进行自动关联,你可以先录制一个脚本,然后选择vuser-scan action for correlation或者直接ctrl+f8来运行一下脚本,这样也能自动找到不同的地方而进行自动关联，此时你只需要钩选你需要关联的地方后点击关联即可。

2.**手工关联**,需要录制2个相同业务流程的虚拟用户脚本,然后通过tools-compare with vusers来打开wdiff.exe对2个脚本进行比较,找到需要关联的部分,然后指定左右边界,然后将脚本中原来的值替换成参数即可.

## 18．怎么理解检查点的作用？

答：在进行Web应用的压力测试时，经常会有页面间数据传递的操作，如果做性能测试时传递次数逐渐增多，页面间就会发生传递混乱的情况，或者客户端与服务端数据传输中断或不正确的现象。为了解决这些问题，LoadRunner提供了在脚本中插入检查点的方法，就是检查Web服务器返回的网页是否正确。在每次脚本运行到此检查点时，自动检查该处的网页是否正确，省去执行结束后人工检查的步骤和时间，进而加快了测试进度。

## 19. 什么时候需要使用集合点，作用是什么？

答：如果想让多个vuser同时进行某操作，那就要创建集合点了；集合点是为了衡量在加重负载的情况下服务器的性能情况lr\_rendezvous("集合点名称");集合点函数可以帮助我们生成有效可控的并发操作。

**集合点设置的策略有三种：1种所有用户达到100%的加载的虚拟用户后才释放;2种是当到达集合点的用户是正在运行用户的100%时集合点释放;3种是到达多少用户释放，剩下的那个是超时时间。**

## 20． cookies和session的含义和区别？

答：用session,和cookie同时保存用户的信息,Session数据存于服务器，Cookie数据存于用户机器 （1.session中保存的是对象，cookie中保存的是字符串。2.session默认需要借助cookie才能正常工作） cookie的内容主要包括：名字，值，过期时间，路径和域。路径与域一起构成cookie的作用范围，服务器通过在HTTP的响应头中加上一行特殊的指示以提示浏览器按照指示生成相应的cookie。

## 21．思考时间是什么？

答：思考时间lr\_think\_time（）; 就是一个事务要开始时思考的时间；（加入思考时间可以更真实的模拟系统的真实情况，压力也不一样）

## 22．动态链接库，头文件基本概念？

答：动态链接库（Dynamic Link Library，缩写为DLL）是一个可以被其它应用程序共享的程序模块，其中封装了一些可以被共享的例程和资源。动态链接库文件的扩展名一般是dll，

头文件被大量使用的文件。#include <>1头文件开头处的版权和版本声明；(2)预处理块；(3)函数和类结构声明等。

**22.跟客户怎么说压力测试和负载测试、并发测试（压力测试是找最大用户数，负载测试是梯度式加压，并发测试同时像服务器发出请求。）**

**问题：什么是拐点测试？系统性能在某一点变化过大（某一项指标的数值）**

# ****二、熟悉工具****

## 1. LR做性能测试时通常用到哪几个工具？

答：脚本编写 VuGen

场景执行 Controller

性能分析器 Analysis

CPU、内存监控 Top/topas、sar,查看日志ssh， putty

## 2. 当同时设了迭代次数和持续运行时间，哪个设置生效？

答：运行时间设置”的时间是 LR会持续连续运行脚本的时间，运行次数可以在LR中查看。

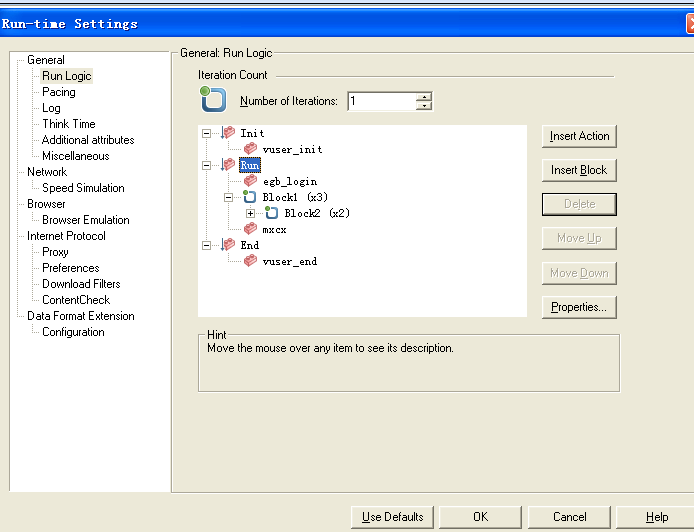
“迭代次数设置”的次数 就是以脚本为主，设置了多少次，脚本就会运行多少次；此时运行时间一般设置为运行完成。

如果同时设置迭代次数和持续时间, 那么当时间结束后, 不管迭代次数是否已经到达设定值,此虚拟用户都会结束。

## 3. 怎么通过block块控制脚本执行流程？

答：Block是取代 Action,block运行的次数,是每个业务运行的次数（业务迭代的总次数=该Block迭代次数×Number of Iterations） 在Run右键属性设置properties-lterations(比例)

在每个业务右键属性设置对应要设置的比例点击确认即可。



## 4. 运行时设置的每个选项的含义？

答：Run Logic（运行逻辑）设置迭代次数，

Pacing设置迭代运行时间，

Log查看日志设置，

Think Time(运行时设置的思考时间)

,Additional attributes(附加属性)添加不是系统本来就有的属性参数,

Miscellaneous(其他的设置:Error Handing错误交接(错误处理），Multithreading多线程，

Automatic Transactions（选择自动交易方式）），

Speed Simulation（网速选择）速度模拟，

Browser Emulation（浏览器仿真）模拟浏览器的性质，

Internet Protocol（互联网协议）（Proxy代理，Preferences，Download Filters（下载过滤器），ContentCheck内容检查）

Configuration组态

## 5. 参数化的多种组合结果是什么？

**Sequential**

Sequential- each iteration

如2个用户分别第一次循环取第一行值，第二次循环取第二行值。

Sequential- each occurrence

如2个用户分别第一次循环取第一行值, 本次循环中当脚本中出现要使用参数的话，参数值取第二行值。第二次循环取取第三行值,本次循环中当脚本中出现要使用参数的话，参数值取第四行值，依次类推。

Sequential- once

所有的用户在所循环中，只用同一个值。

**Random**

Random - each iteration

如2个用户，在不同的循环次数中，各自随机取值。

Random - each occurrence

如2个用户，在不同的循环次数中，各自随机取值，当每次循环中每遇到一次参数名，也各自随机取值。

Random - once

如2个不同的用户，不管循环多少次，各自只随机取值一次。

**unique**

unique - each iteration

如2个用户，分别在不同的循环中取不同参数值。

unique - each occurrence

只能手工分配用户参数数据值。

unique – once

按照用户数给每个用户顺序分配一个固定参数，以后的循环这个用户就使用这个固定的参数。

## 6.怎么查看每个用户的运行时日志？

答：打开日志后，运行脚本在passed右键点击show Vuser log.或在脚本文件夹\res\log里

在场景中查看日志时，要把脚本和场景的日志都要打开，

## 7. 脚本运行时设置和场景运行时设置的区别？

答：脚本设置时看脚本是否正确，场景设置是监控查看性能指标。

1、thinktime：在VuGen中运行时会忽略，但是在场景中，会按照脚本的lr\_think\_time()函数进行运行。

2、系统日志设置会更改为不出现错误就不记录日志

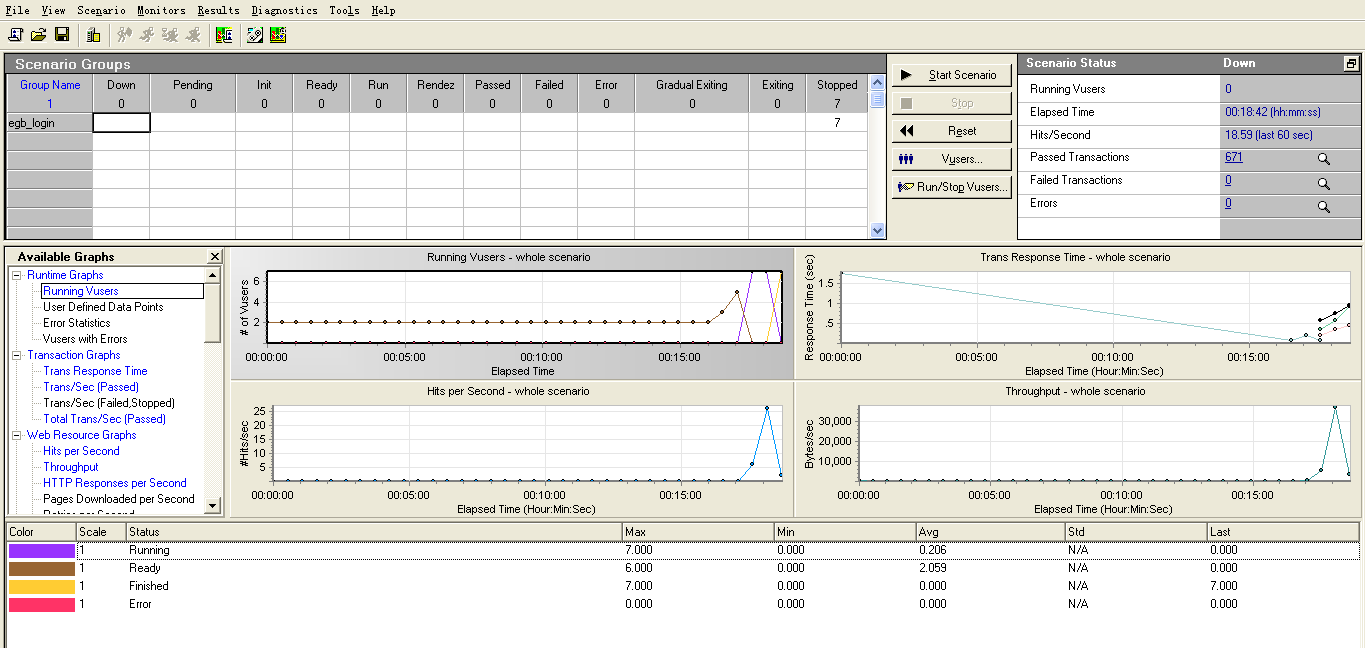
3、需要关闭自动化事务，为了避免多余数据影响

4、带宽模拟，需要为用户设置一个合理的带宽来得到更真实的用户访问时间

5、集合点策略，如果脚本中有集合点，则需要根据需求对集合点的策略进行设置

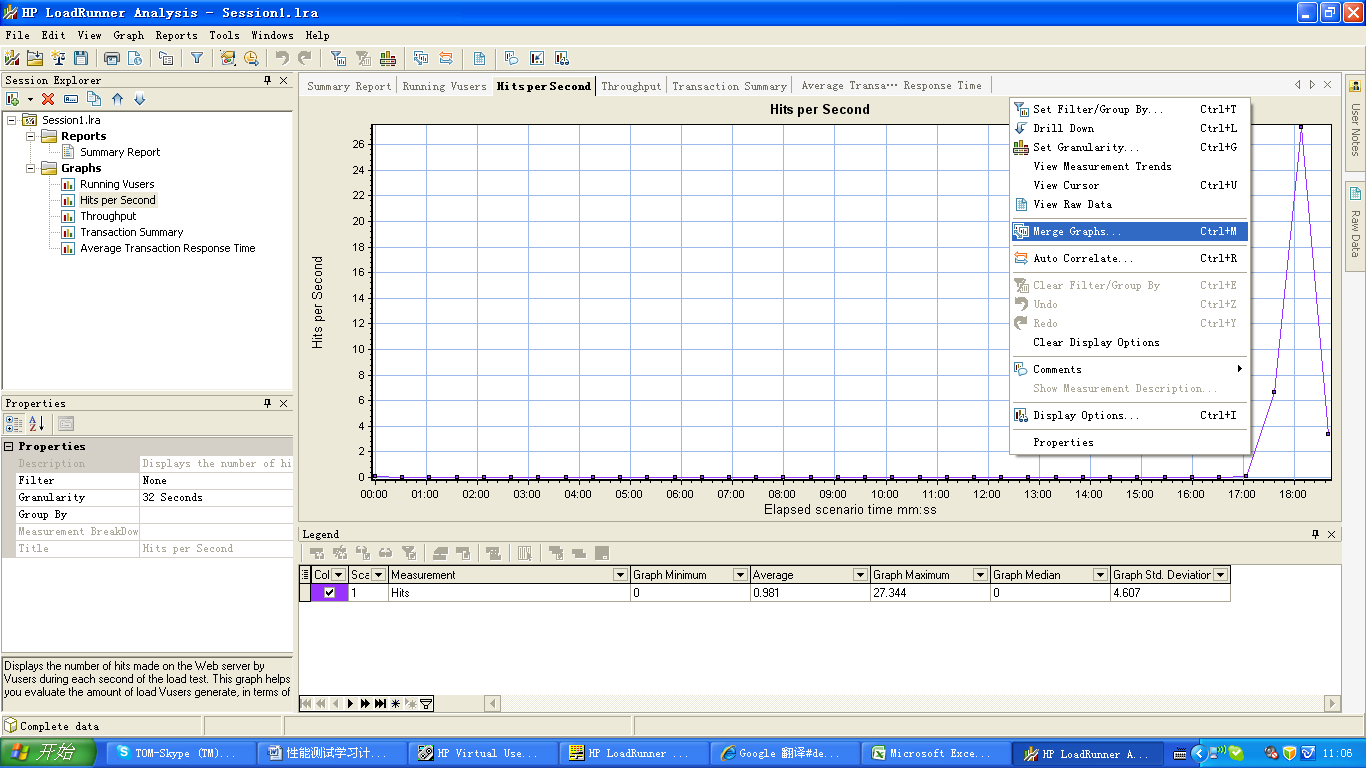
## 8. 我们需要重点监控哪几个图？

答：Running Vusers(用户数），Trans Response Time(事务响应时间），Throughput(吞吐量），Hits per Second(每秒点击数），查看TPS要用的图Trans/Sec（Passed）和Trans/Sec（Failed ,Stopped）,服务器 Web Resource Graphs（Hits per Second） TPS公式：TPS\_1 ≥ 70000 \* 80% / (24 \* 20% \* 3600) = 3.2 笔/秒（根据统计数据，恒丰银行网银系统当前生产环境高峰日交易总量为7万笔。根据二八原则（80%的交易量发生在20%的时间段内），当前生产环境对主机的交易吞吐量指标要求为：

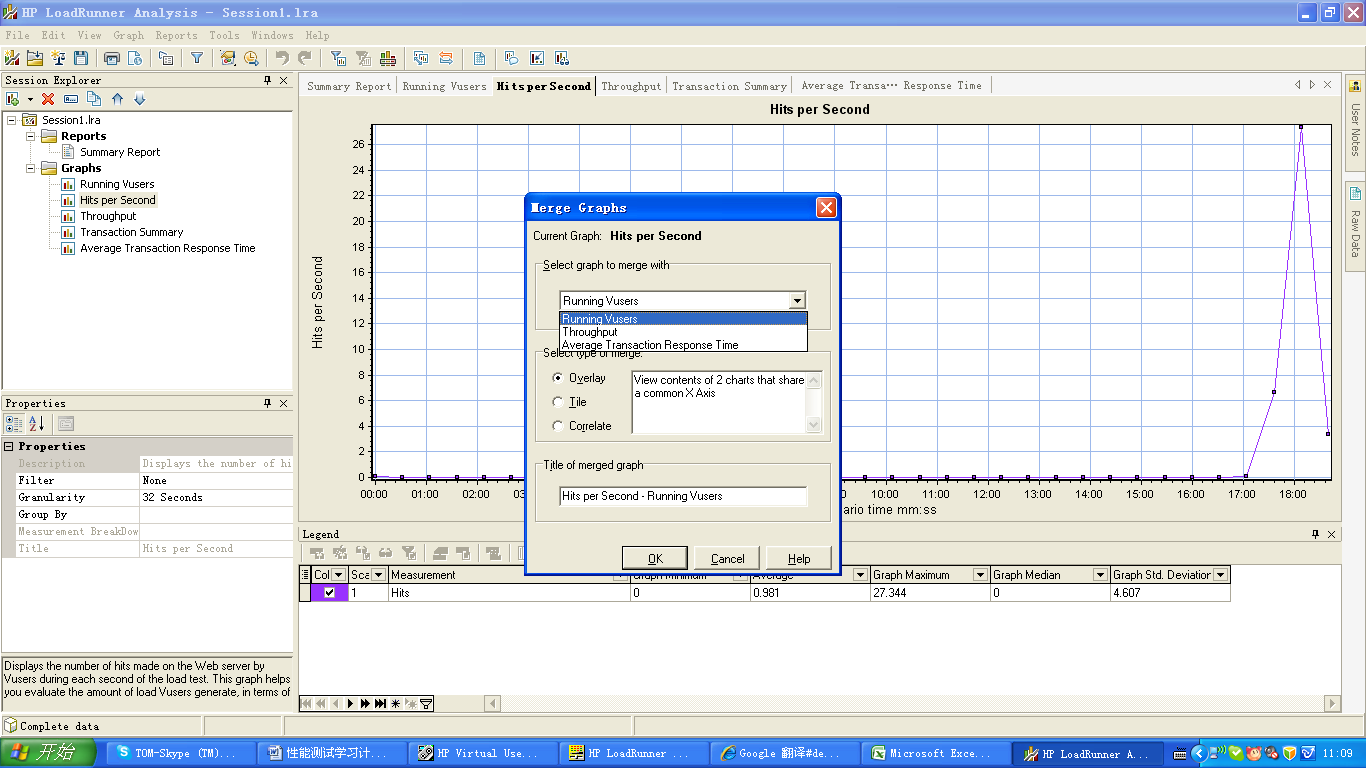


## 9. 在分析结果时怎么进行结果图的叠加？

答：选中要跌加的一个图点击右键Merge Graphs(Ctrl+M),在Select graph to merge with 现在要叠加图的名称



选择要合并的图的名称



## 10.怎么分析结果的时间，服务器还是网络资源耗时多？

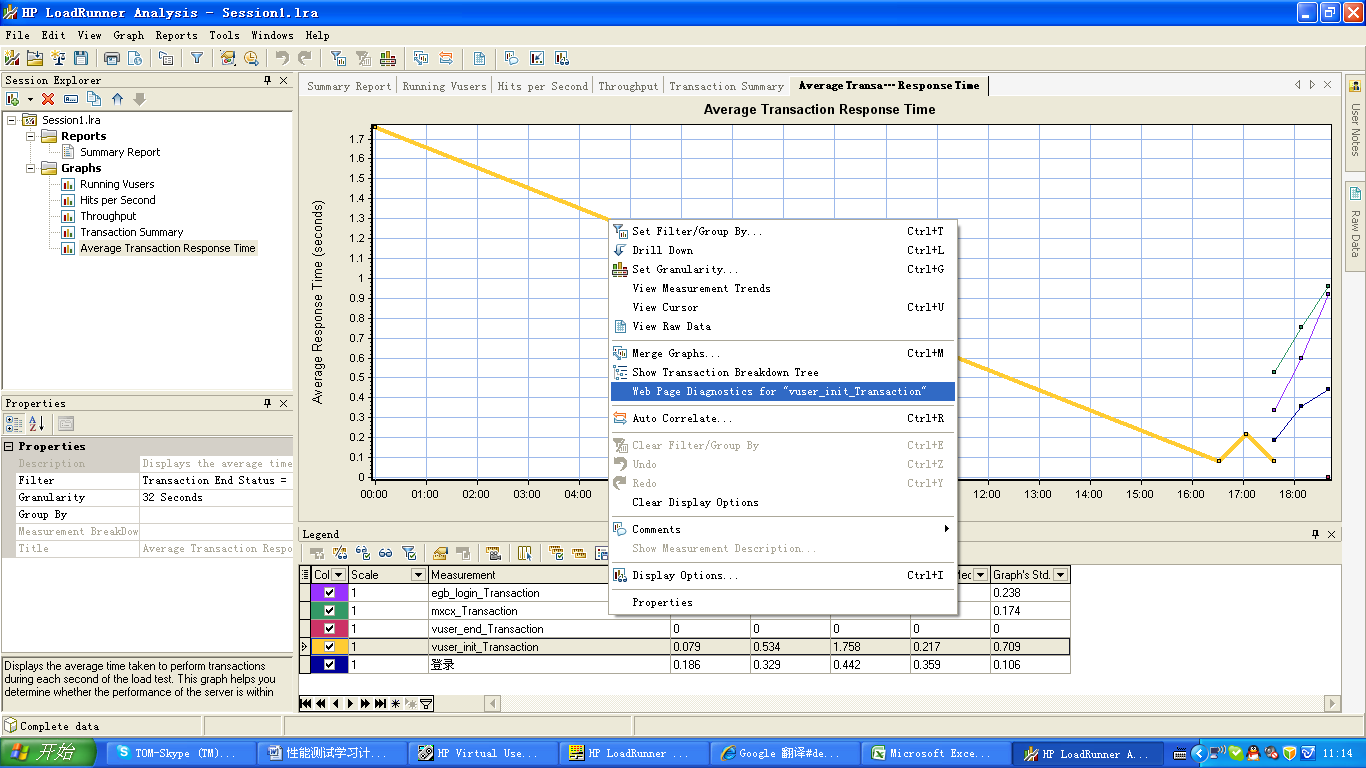
选择吞吐量试图（Average Transaction Response Time)右键-Web Page Diagnostics for Download Time 下载时间

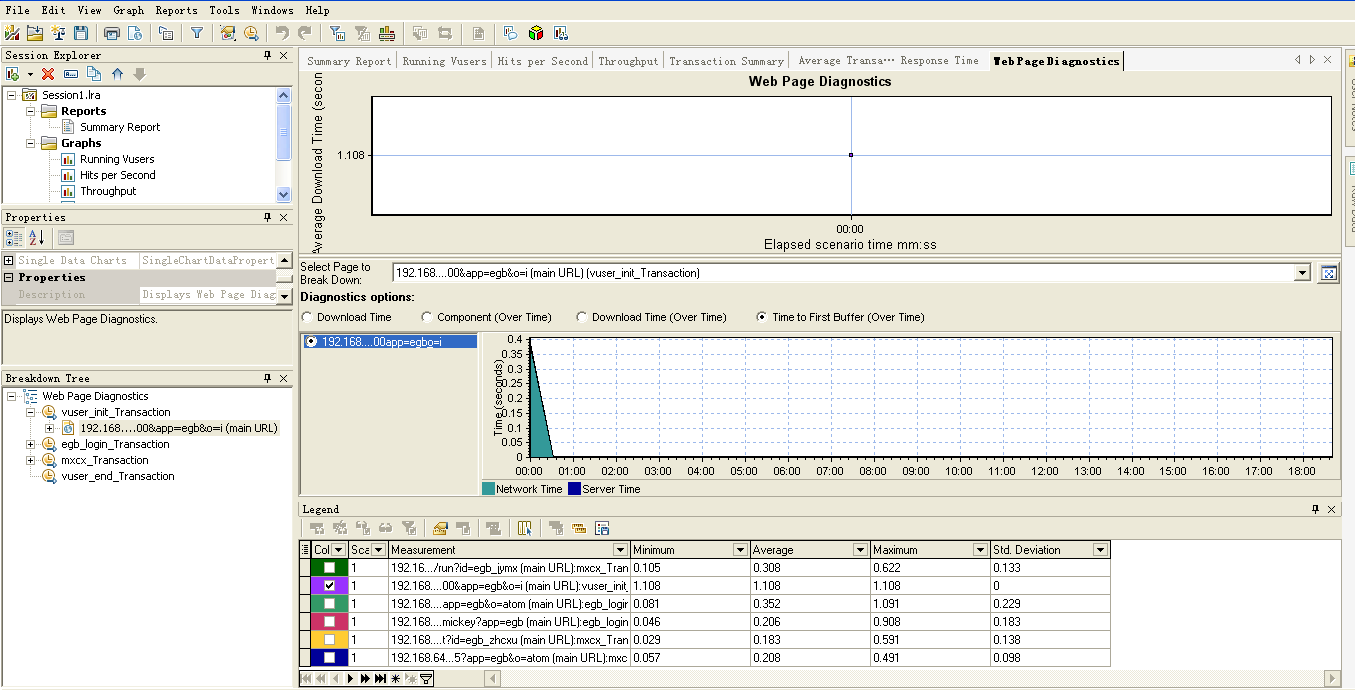
Component(Over Time)组件（超过时间）

Download Time 下载时间

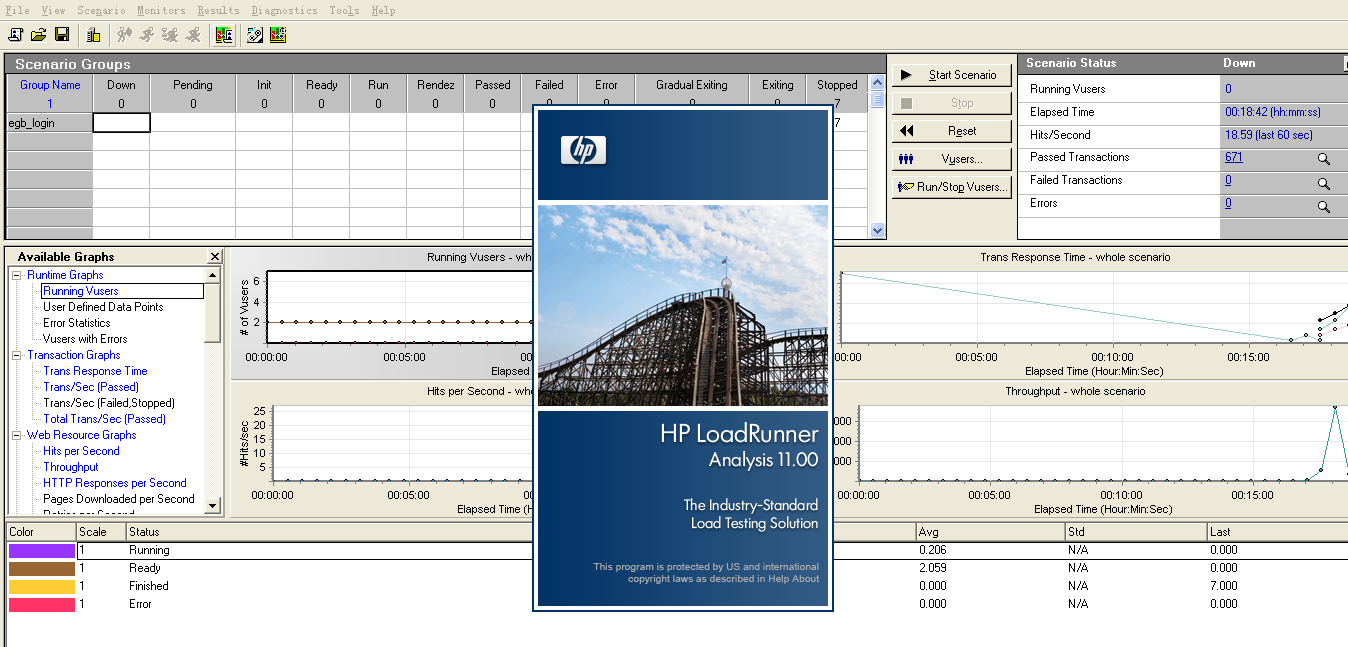
Time to First Buffer 时间第一个缓冲区

network 网络 Server服务器

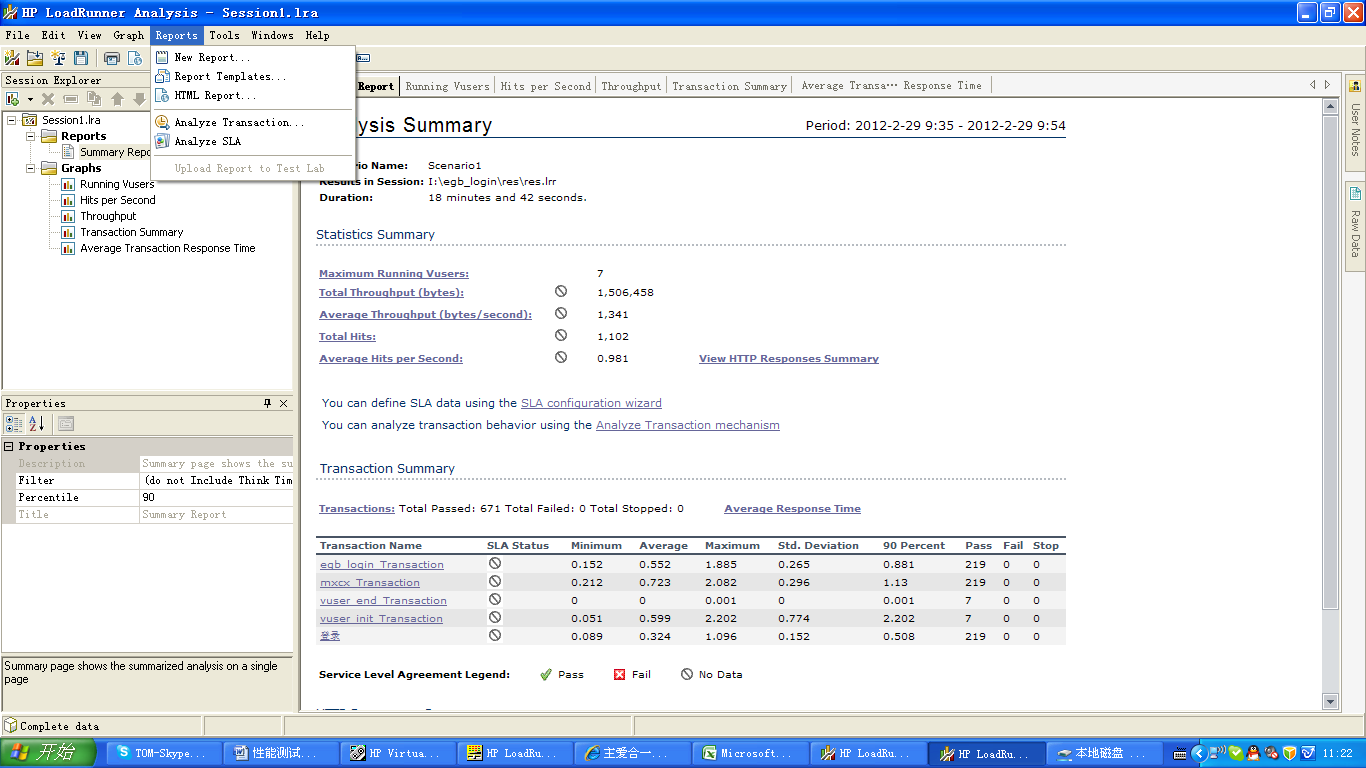




## 11．怎么生成报告，或者保存为网页报告形式？

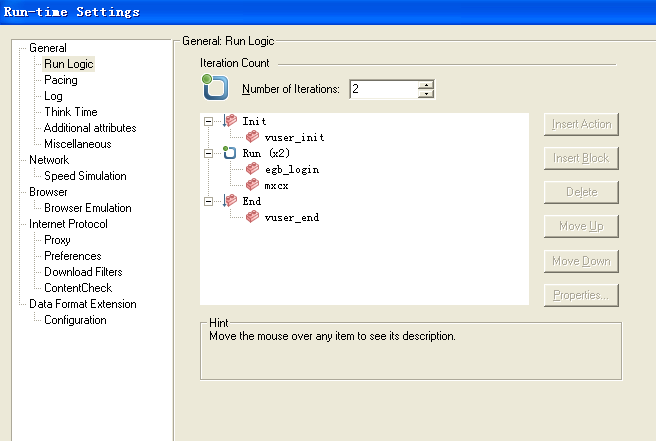
答：Results-Analyze Results生成报告，生成文档报告（Reports-Nen Reports建报告名-Generate.自动生成报告。

选择生成报告格式



## 12. 脚本结构，哪个部分用户循环？

答：Action部分用户循环



## 13. 脚本运行时日志中的各选项含义？

1、 Enable logging启用日志记录

如果选中该选项Loadrunner在执行脚本时，进行日志的记录，否则不记录日志

2、 Send messages only when an error occurs 仅在出错时发送消息

也称为 JIT (实时)消息传递，仅当错误发生时才写入日志，选择该选项后则可以设置高级选项，指明日志缓存的大小，loadrunner默认的日志到小为1k

3、 Always send messages

始终发送消息

4、 Standard log

标准日志：创建在脚本执行期间发送的函数和消息的标准日志，供调试时使用。

对于大型负载测试场景、优化会话或配置文件禁用此选项。

如果日志记录级别设置为“标准”，当把脚本添加到场景、会话步骤或配置文件中时，日志记录模式将被自动设置为“Send messages only when an error occurs”。但是，如果日志记录模式被禁用或者设置为“扩展”，则将脚本添加到场景、会话步骤或配置文件中将不会影响其日志记录设置。

5、 Extended log-----Parameter substitution

参数替换：选择此选项可以记录指定给脚本的所有参数及其相应的值

当脚本进行参数化、插入事务、关联等优化后，在执行脚本过程中，参数化的值、事务所耗时间、关联函数取出的变量值均会在日志中输出，这个选项对调试脚本查看参数化取值、关联取值是否正确有着重要的作用

6、 Extended log-----Data returned by server

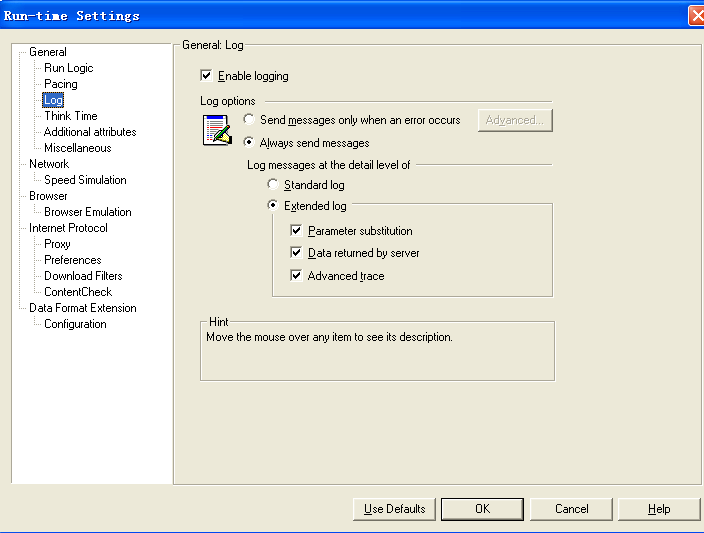
选择此选项可以记录服务器返回的所有数据。

Loadrunner会将所有对服务器发出请求后的response情况记录在日志中，从这个日志中可以查看到服务器对请求的回应是否正确，在使用关联取值时往往需要到该日志中查看需要关联的值，从而确认所取数据左右边界。

7、 Extended log-----Advanced trace 高级跟踪

选择此选项可以记录 Vuser 在会话期间发送的所有函数和消息。

调试 Vuser 脚本时，该选项非常有用。



## 14. 场景执行过程中怎么看需要关心的数据，都有哪些数据？

答：响应时间、点击率、用户数、事务通过数、失败数、吞吐量

cpu：top

内存：sar。sar -r 2 10

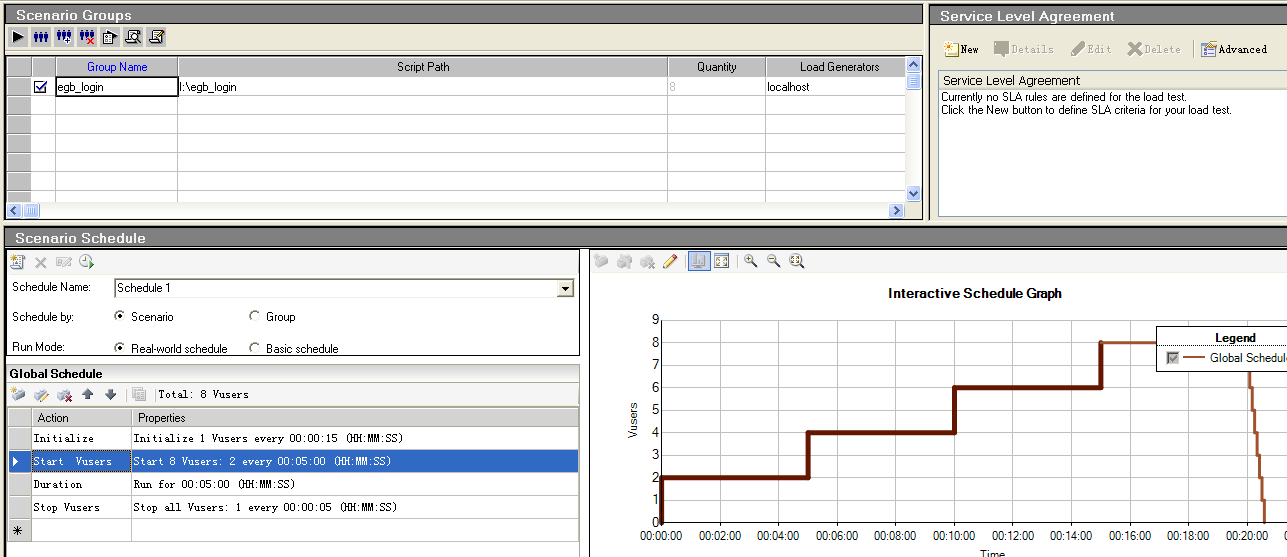
I/O iostat 2 10 vmstat

查看ewp进程ps auwx|grep ewp

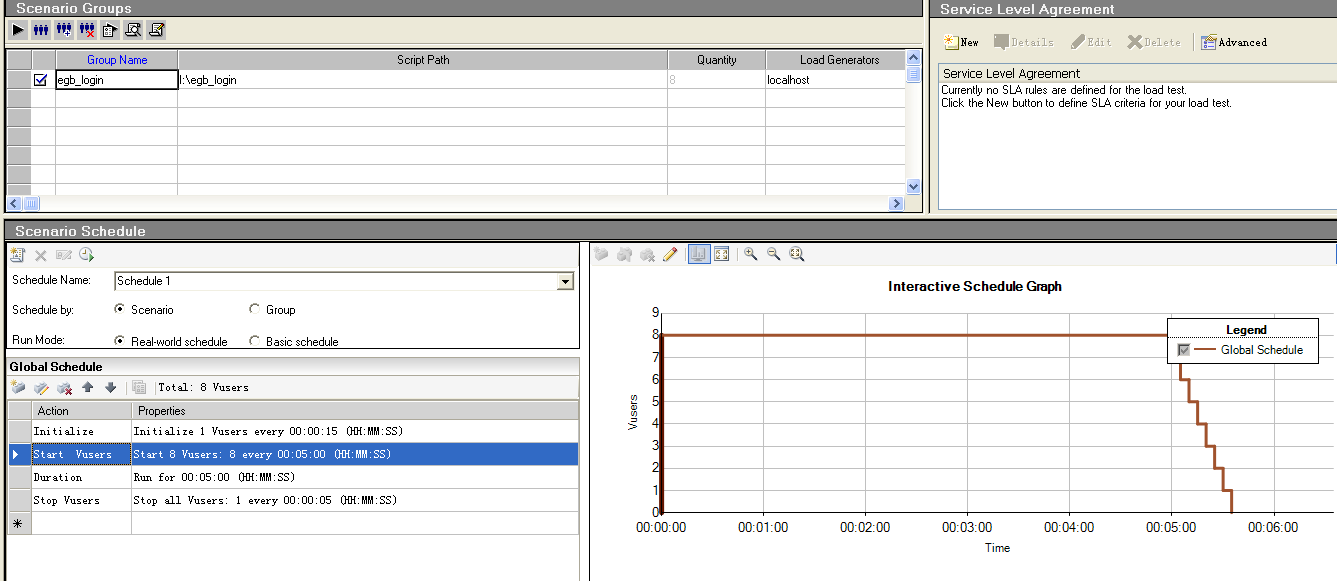
看ewp日志：tail -f /var/www/apps/ewp/log/report.log

## 15. 场景设计中的负载加载方式有哪些，怎么做？

答：阶梯式加压，一次性加压,并发式加载

阶梯式加压

一次性加压,并发式加载



## 16. 协议的选取？

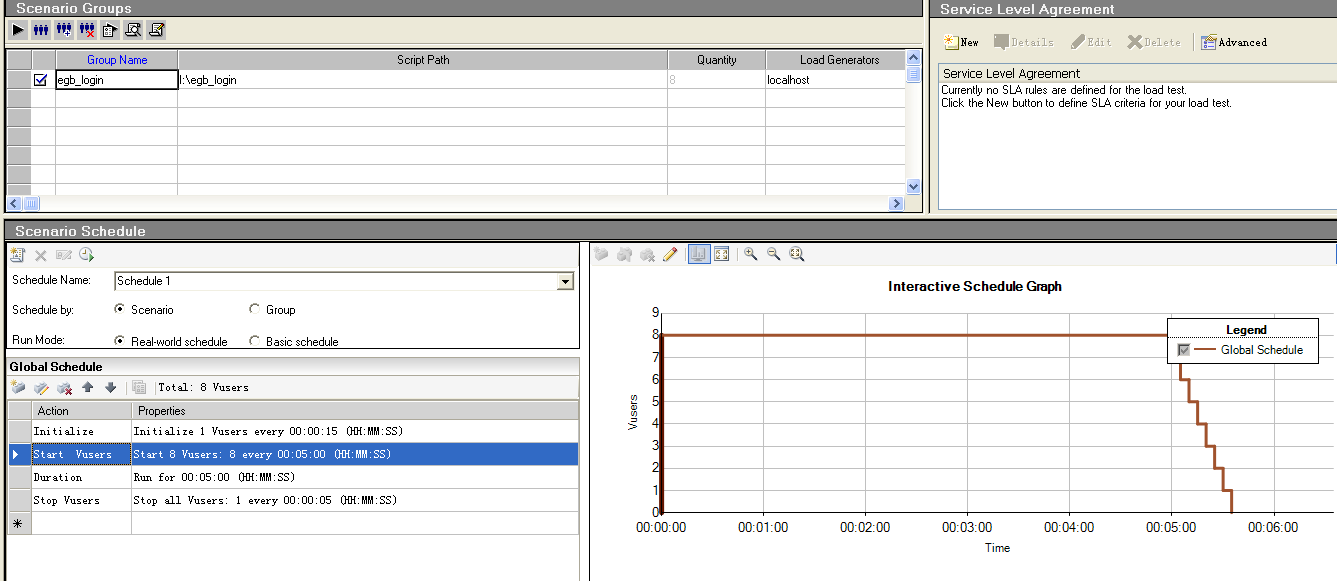
答：1、可以直接确认

2、通过研发人员了解被测系统使用了什么样的协议。

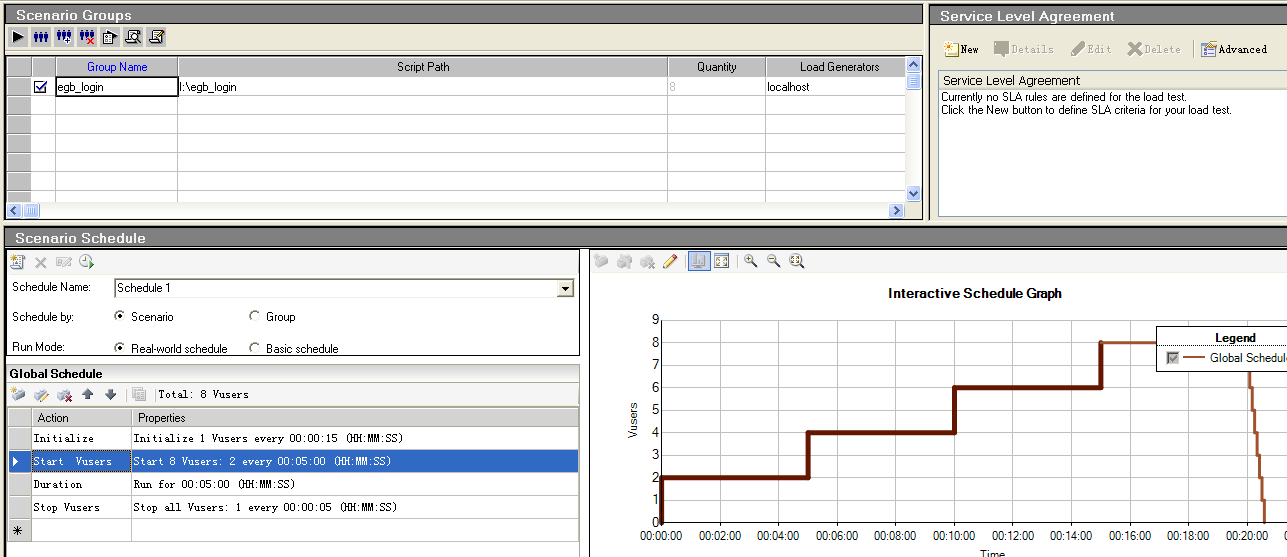
## 17. 几种场景的设计方式?

答：并发测试、阶梯式加压、疲劳测试、单业务测试、负载测试

并发测试



压力测试



## 18. 怎么设置场景？

答：启动Controller,(Manual Scenario用户完全手动设置，Goal-Oriented Scenario在现在的场景中设置，选择添加执行的脚本，创建场景方案，运行场景。

**创建场景：**

1、可以通过VuGen直接转换当前脚本进入场景，在tools菜单下打开Create Controller Scenario

2、打开Controller新建场景，在弹出的新场景设置窗口设置场景类型和对应脚本，目标场景（Goal Scenario），手工场景（Manual Scenario）

**目标场景设置：**

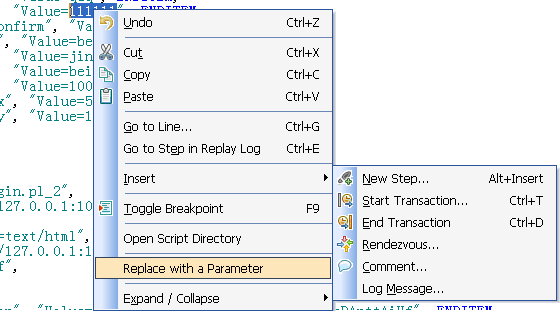
1. Virtual Users表示虚拟用户数，被测试系统所需要支持的用户数
2. Hits per Second 设置每秒点击数目标
3. Transaction per Sceond 设置每秒事务数，一个事务代表一个操作，每秒事务数代表了系统的处理能力。
4. Transaction Response Time事务响应时间，反映了系统的处理速度以及做一个操作所需花费时间。
5. Pages per Minute 每分钟页面的刷新次数，反映了系统在每分钟下所能提供的Page的处理能力。

**Design设计 Run 运行**

## 19. 脚本的参数化怎么做？

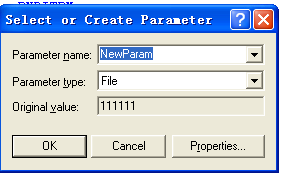
1、将光标定位在要参数化的字符上，点击右键。打开弹出菜单。

2、在弹出菜单中，选择"Replace with a Parameter"。选择或者创建参数的对话框弹出。



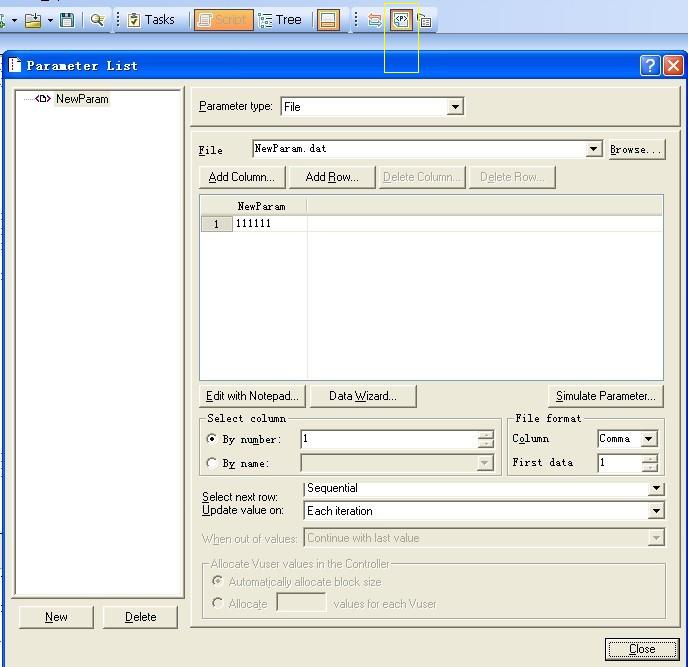
3、在"Parameter name"中输入参数的名称，或者选择一个在参数列表中已经存在的参数。

4、在"Parameter type"下拉列表中选择参数类型。



5、点击"OK"，关闭该对话框。脚本生成器便会用参数中的值来取代脚本中被参数化的字符，参数用一对"{}"括住。

"Name=password", "Value={NewParam}", ENDITEM,

6.在双击 

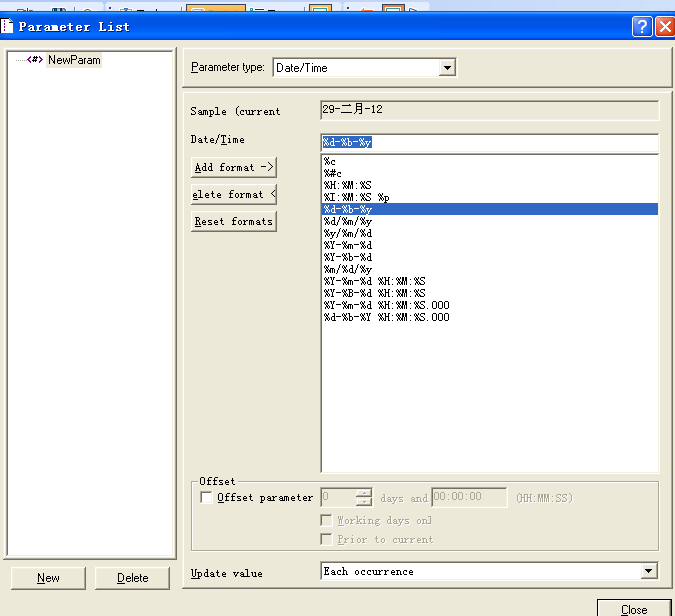
点击Edit with Notepad,打开参数化数据文件，添加数据后，选择By name选择参数名，点击Close。

## 20. 参数化的几种类型是什么，怎么用？

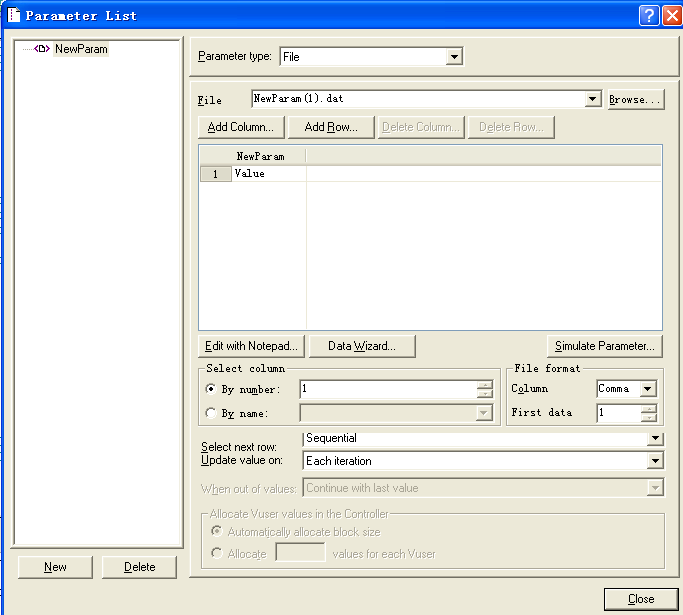
BPT

Custom

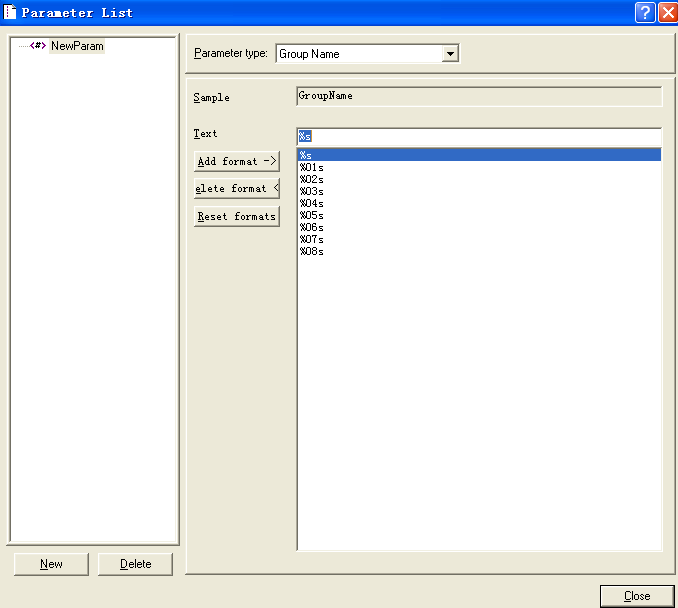
**Date/Time（日期/时间）**参数类型：Date/Time类型用当前日期/时间替换参数



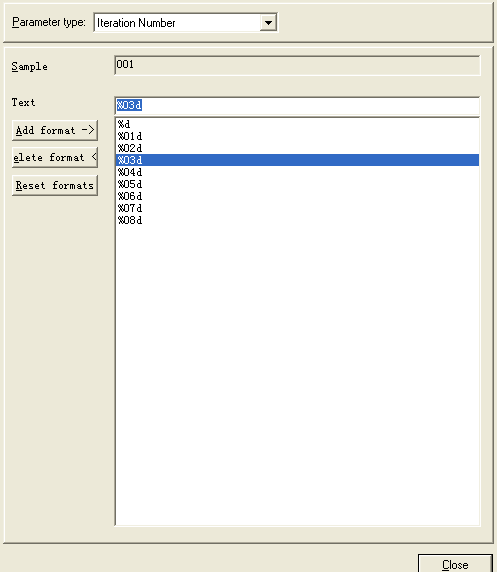
**File（文件）**参数类型：在参数属性编辑参数文件，或直接选择已编译好的参数文件，还可以从现成数据库提取此文件。



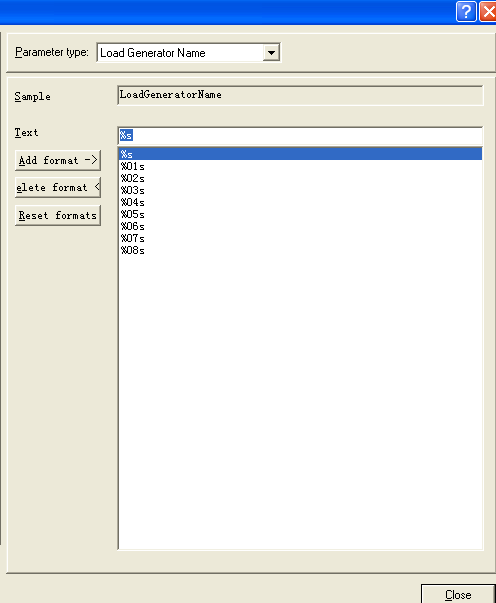
**Group Name（组的名称）**参数类型:用Vuser组的名称替换参数。



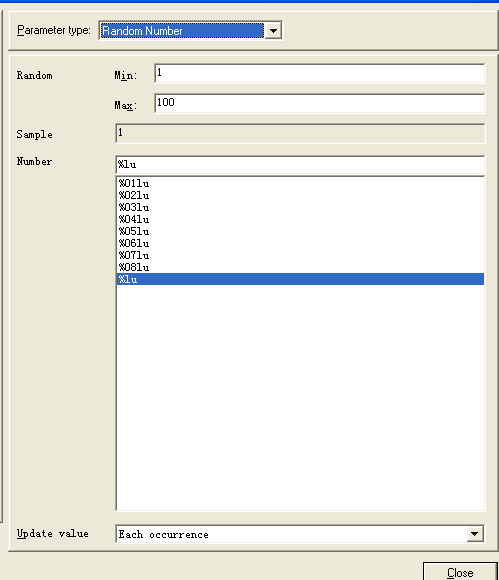
**lteration Number（迭代编号）**参数类型：用当前的迭代编号替换参数。



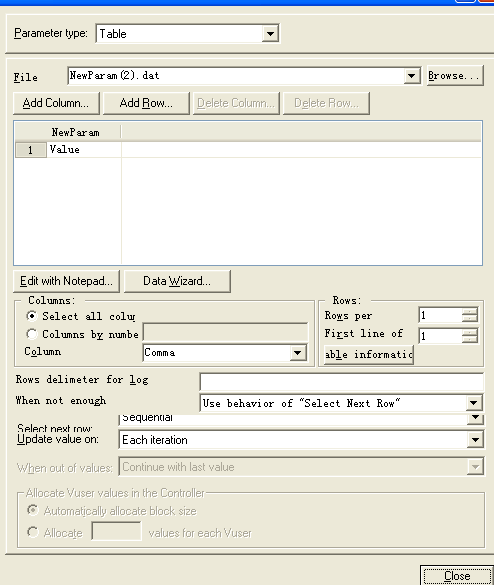
**Load Generator Name（负载生成服务器的名称）**参数类型：用Vuser脚本的负载发生器名替换参数，负载发生器是运行Vuser的计算机。



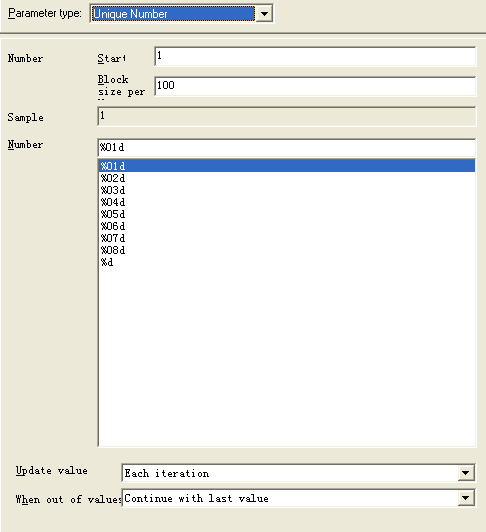
**Random Number随机数（参数化随机数范围）**参数类型：用一个随机生成的整数替换参数，可以通过指定最大值和最小值，设置随机编号范围。



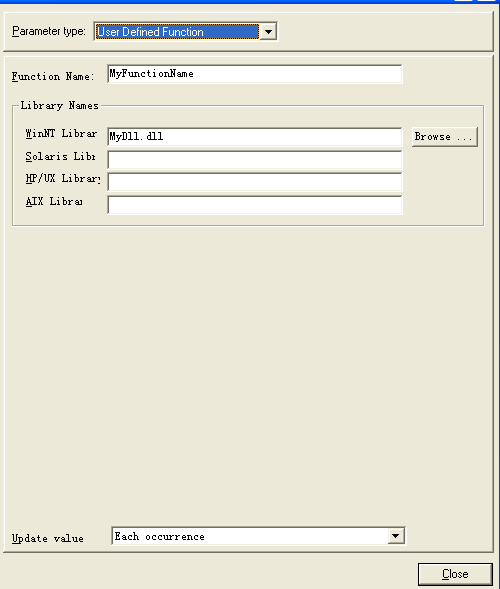
**Table表（参数化数据表）:** 是一个FILE类型增强版，它提供了一些独特的功能。Columns提供了对列的分割方式的设置。

****

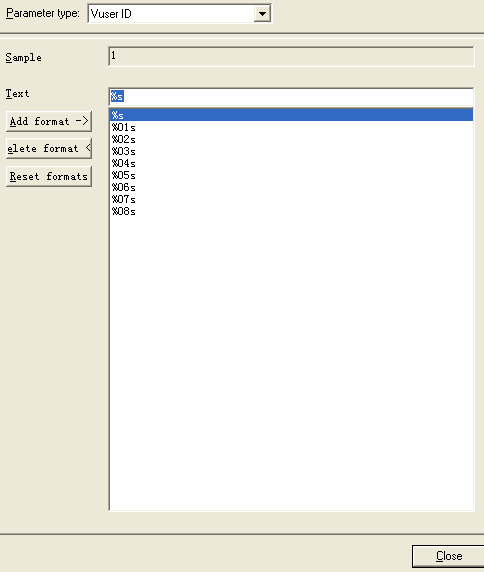
**Unique Number（参数化用户名）唯一的编号数据**参数类型：Block size(块大小)用一个唯一编码替换参数。指明分配给每个Vuser的编号的大小。每个Vuser都从其范围的下限开始，在每次迭代是递增该参数。

****

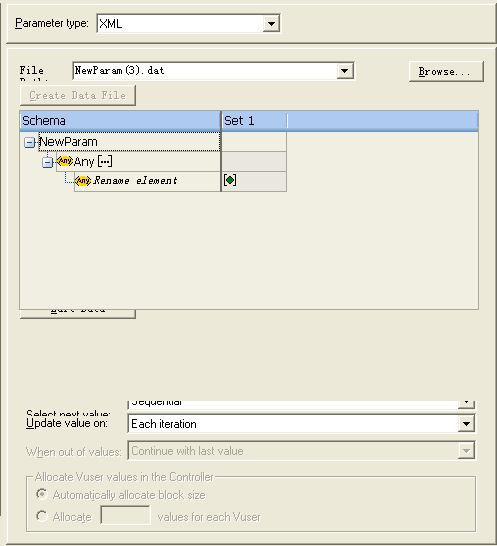
**User Derined Function用户功能Derined（提供外部dll扩展脚本功能）**：这是VuGen提供的扩展功能。当需要通过外部程序生成某些数据的时候，可以通过dll的方式来完成特殊输入。

****

**Vuser ID虚拟用户的编号**参数类型：LR使用该虚拟用户的ID来替换数值，该ID由Controller来控制。在VuGen中运行脚本时，VuGen将会是-1.



**xml（以xml格式参数化）**参数类型：作为一种流行数据格式，XML在业界得到了大量的应用，XML参数类型提供了对XML格式的支持，可以在参数器修改XML内容格式。



## 22.理解线程和进程的含义和区别?

答：进程和线程都是由操作系统所体会的程序运行的基本单元，系统利用该基本单元实现系统对应用的并发性。进程和线程的区别在于：简而言之,一个程序至少有一个进程,一个进程至少有一个线程.

线程的划分尺度小于进程，使得多线程程序的并发性高。

线程在执行过程中与进程还是有区别的。每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口。但是线程不能够独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制。

有2个用户在运行时，进程是在任务管理器出现2个mmdrv.exe,线程是出现一个mmdrv.exe。

## 23怎么插入事务？

答：自动添加事务点击Insert Start Transaction（Ctrl+T）和Insert End Transaction（Ctrl+D）来添加事务，手动添加事务在获取响应时间前添加 事务开始lr\_start\_transaction(“事务名”)；获取响应时间后 lr\_end\_transaction("事务名", LR\_AUTO);事务结束。

## 24.怎么创建场景？

答：1、可以通过VuGen直接转换当前脚本进入场景，在tools菜单下打开Create Controller Scenario

2、打开Controller新建场景，在弹出的新场景设置窗口设置场景类型和对应脚本，目标场景（Goal Scenario），手工场景（Manual Scenario）

## 25.怎么生成结果？

Results-Results settings在场景中保存结果，选择的目录

答：在运行脚本结束后，查看运行情况，Results-Analyze Results（analysis----->report-------->Html Report(word report格式)），查看结果图，和查看并发用户数是负载压力的主要指标，测试重点得到两类并发用户数指标，一类是系统最佳性能的并发用户数，另一类是系统能够承受的最大并发用户数，这两类指标在某种情况下有可能重叠。交易响应时间图， 交易通过率指标，吞吐量图， 点击率图，查看 服务器操作系统资源占用  CPU数据、磁盘管理I/O数据、内存数据进程

1. 创建HTML报告

打开Report菜单下的HTML Report命令，设置报告保存地址，确认后就会得到HTML版的性能测试报告。

1. 创建word报告

打开Report菜单下的Microsoft Word Report命令创建一个word报告。

1. 创建水晶报表

打开Report菜单下的Crystal Report命令，这里提供了Activity report（活动报告）和Performance report（性能报告）

活动报告：提供关于虚拟用户和场景运行的信息，包括开始时间、结束时间等详细信息。又包括场景执行（Scenario Execution）、失败交易（Failed Transaction）、失败的虚拟用户（Failed Vusers）三种。

性能报告：可以分析每个Transaction（事务）所用的时间，其中有Data Point（数据点）、Detailed Transaction（详细事务）和Transaction Performance By User（事务性能）。

## 26.在运行脚本中，发现地址写错了怎么办？

答：在设置场景场景中Design-上边的按钮（Details）-Refresh按钮-选Script 刷新脚本 ( Runtime Settings运行时设置)

## 27.怎么去设置HTTP的请求超时时间设置？

答：选择Run-time Settings –preferences –Options-HTTP-

HTTP -request connect timeout (sec) HTTP请求连接超时（秒）

## 28.在现实脚本把

答：Run-time Settings-Misscellaneous-Error Handing (Continue on error)选上。

## 29.怎么在结果中，添加没有监控的图？

答：打开场景结果图界面Graphs右键-Add NewItem –Add New Graph-选择你要选择的指标图双击后自动打开，生成HTMl和自动保存文件等。

# ****三．搭建项目环境****

## 1.常见证书文件格式，什么时候需要用证书?

答：CACert.crt证书和unicom.cer证书。

CACert.crt签名文件证书（数字证书采用公钥体制，即利用一对互相匹配的密钥进行加密、解密）

unicom.cer证书（手机通信证书）

.pem：隐式增强邮件（Private Enhanced Mail）【openssl用此格式保存私钥（类似keytool中的keystore的作用），也是一种证书格式（不含私钥），BASE64编码】

.csr：证书签名请求（Certificate Signing Request）

.p12/.pfx：PKCS#12标准，一般作为密钥库或信任库使用，有密码保护（pfx-名词释义       
公钥加密技术12号标准（Public Key Cryptography Standards #12，PKCS#12）为存储和传输用户或服务器私钥、公钥和证书指定了一个可移植的格式。它是一种二进制格式，这些文件也称为PFX文件。）

.cer/.crt：客户证书，不含私钥，.cer有两种编码，BASE64编码或DER二进制编码

.ber .per .xer .der .jks：密钥库类型，Java KeyStore，也是默认密钥库类型

## 2.需要哪些头文件？

答：Handshake.h加密握手头文件，diCrPKI.h 宏定义定义所有PKl头文件， Lib.h分配空间的头文件 ，diCryptoSys.h宏定义定义所有的 API头文件，

常用的头文件如下   
#include <assert.h> //设定插入点   
#include <ctype.h> //字符处理   
#include <errno.h> //定义错误码   
#include <float.h> //浮点数处理   
#include <fstream.h> //文件输入／输出   
#include <iomanip.h> //参数化输入／输出   
#include <iostream.h> //数据流输入／输出   
#include <limits.h> //定义各种数据类型最值常量   
#include <locale.h> //定义本地化函数   
#include <math.h> //定义数学函数   
#include <stdio.h> //定义输入／输出函数   
#include <stdlib.h> //定义杂项函数及内存分配函数   
#include <string.h> //字符串处理   
#include <strstrea.h> //基于数组的输入／输出   
#include <time.h> //定义关于时间的函数   
#include <wchar.h> //宽字符处理及输入／输出   
#include <wctype.h> //宽字符分类   
ALLOC.H 　　　说明内存管理函数(分配、释放等)。

ASSERT.H 　 　定义 assert调试宏。

BIOS.H 　　 　说明调用IBM—PC ROM BIOS子程序的各个函数。

CONIO.H 　　　说明调用DOS控制台I/O子程序的各个函数。

CTYPE.H 　　　包含有关字符分类及转换的名类信息(如 isalpha和toascii等)。

DIR.H 　　　　包含有关目录和路径的结构、宏定义和函数。

DOS.H 　　　　定义和说明MSDOS和8086调用的一些常量和函数。

ERRON.H 　　　定义错误代码的助记符。

FCNTL.H 　　　定义在与open库子程序连接时的符号常量。

FLOAT.H 　　　包含有关浮点运算的一些参数和函数。

GRAPHICS.H 　 说明有关图形功能的各个函数，图形错误代码的常量定义，正对不同驱动程序的各种颜色值，及函数用到的一些特殊结构。

IO.H 　　　　 包含低级I/O子程序的结构和说明。

LIMIT.H 　　　包含各环境参数、编译时间限制、数的范围等信息。

MATH.H 　　　 说明数学运算函数，还定了 HUGE VAL 宏， 说明了matherr和matherr子程序用到的特殊结构。

MEM.H 　　　　说明一些内存操作函数(其中大多数也在STRING.H 中说明)。

PROCESS.H 　　说明进程管理的各个函数，spawn…和EXEC …函数的结构说明。

SETJMP.H 　　 定义longjmp和setjmp函数用到的jmp buf类型， 说明这两个函数。

SHARE.H 　　　定义文件共享函数的参数。

SIGNAL.H 　　 定义SIG[ZZ(Z] [ZZ)]IGN和SIG[ZZ(Z] [ZZ)]DFL常量，说明rajse和signal两个函数。

STDARG.H 　　 定义读函数参数表的宏。(如vprintf,vscarf函数)。

STDDEF.H 　　 定义一些公共数据类型和宏。

STDIO.H 　　　定义Kernighan和Ritchie在Unix System V 中定义的标准和扩展的类型和宏。还定义标准I/O 预定义流：stdin,stdout和stderr，说明 I/O流子程序。

STDLIB.H 　　 说明一些常用的子程序：转换子程序、搜索/ 排序子程序等。

STRING.H 　　 说明一些串操作和内存操作函数。

SYS\STAT.H 　 定义在打开和创建文件时用到的一些符号常量。

SYS\TYPES.H 　说明ftime函数和timeb结构。

SYS\TIME.H 　 定义时间的类型time[ZZ(Z] [ZZ)]t。

TIME.H 　　　 定义时间转换子程序asctime、localtime和gmtime的结构，ctime、 difftime、 gmtime、 localtime和stime用到的类型，并提供这些函数的原型。

VALUE.H 　　　定义一些重要常量， 包括依赖于机器硬件的和为与Unix System V相兼容而说明的一些常量，包括浮点和双精度值的范围。

## 3.需要哪些dll文件？

答：diCrPKI.dll，diCryptoSys.dll （宏定义的动态链接库）（dll是动态链接库），

Dll的制作过程如下：

(1.) 启动VC6.0

(2.)在FILE- >NEW- >PROJECTS中选择"WIN32 Dynamic-Link Library"，在Project Name中输入 "test1"，单击OK

(3.)在FILE- >NEW- >FILES中选择C++ SOURCE FILE，在FILE中输入test1.cpp，单击OK

(4.)在FILE- >NEW- >FILES中选择TEXT文件，在FILE中输入test1.h，单击OK

(5.) 在FILE- >NEW- >FILES中选择TEXT文件，在FILE中输入test1.def，单击OK

## 4.了解常见的网络拓扑结构？

答：网络拓扑结构，英文名称：network topology。 （男科瑞，t罢了b）

拓扑这个名词是从几何学中借用来的。拓扑学（TOPOLOGY）是一种研究与大小、距离无关的几何图形特性的方法。

**网络拓扑结构定义**：在计算机网络中指定设备和线路的安排或布局；在地理网络中指网络要素之间的连接关系。在选择拓扑结构时，主要考虑的因素有：安装的相对难易程度、重新配置的难易程度、维护的相对难易程度、通信介质发生故障时，受到影响的设备的情况。

**常见网络拓扑结构：**

**(1)星型拓扑结构**

星型拓扑是由中央节点和通过点到通信链路接到中央节点的各个站点组成。

星型拓扑结构具有以下优点：

（1）控制简单。

（2）故障诊断和隔离容易。

（3）方便服务。

星型拓扑结构的缺点：

（1）电缆长度和安装工作量可观。

（2）中央节点的负担较重，形成瓶颈。

（3）各站点的分布处理能力较低。



**(2)总线拓扑结构**

总线拓扑结构采用一个信道作为传输媒体，所有站点都通过相应的硬件接口直接连到这一公共传输媒体上，该公共传输媒体即称为总线。

总线拓扑结构的优点：

（1）总线结构所需要的电缆数量少。

（2）总线结构简单，又是无源工作，有较高的可靠性。

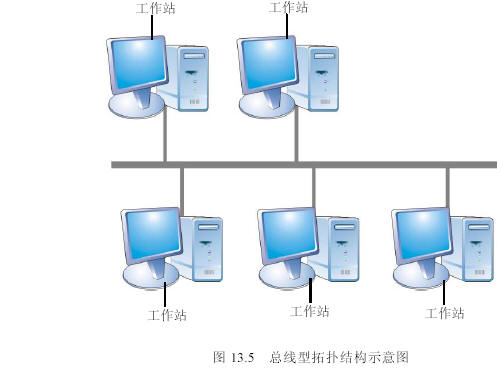
（3）易于扩充，增加或减少用户比较方便。

总线拓扑的缺点：

（1）总线的传输距离有限，通信范围受到限制。

（2）故障诊断和隔离较困难。

（3）分布式协议不能保证信息的及时传送，不具有实时功能



**(3)环型拓扑结构**

环型拓扑网络由站点和连接站的链路组成一个闭合环。

环型拓扑的优点：

（1）电缆长度短。

（2）增加或减少工作站时，仅需简单的连接操作。

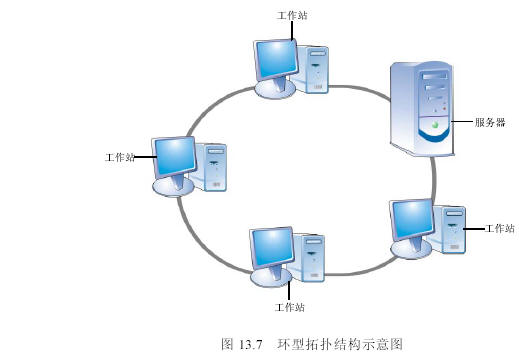
（3）可使用光纤。 （光纤是光导纤维的简写，是一种利用光在玻璃或塑料制成的纤维中的[全反射](http://baike.baidu.com/view/45238.htm" \t "_blank)原理而达成的光传导工具。）

环型拓扑的缺点：

（1）节点的故障会引起全网故障。

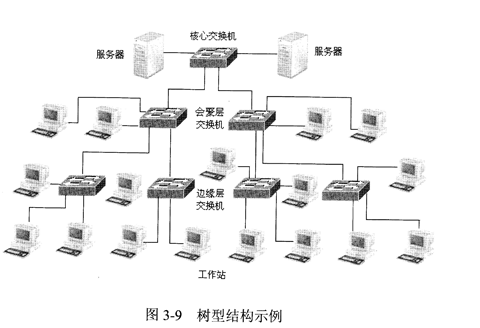
（2）故障检测困难。

（3）环型拓扑结构的媒体访问控制协议都采用令牌传达室递的方式，在负载很轻时，信道利用率相对来说就比较低。

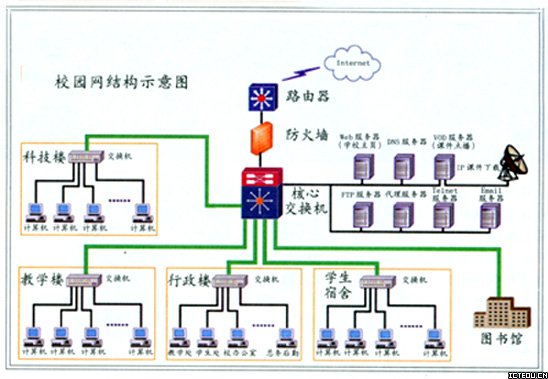


**其它网络拓扑结构：**

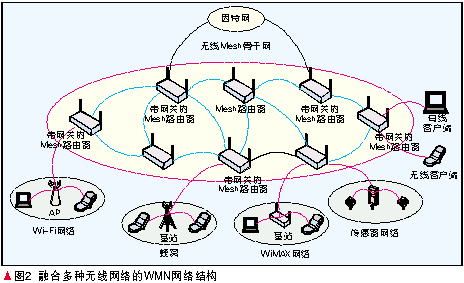
**树型拓扑结构**：树型结构实际上是星型结构的一种变形，它将原来用单独链路直接连接的节点通过多级处理主机进行分级连接。这种结构与星型结构相比降低了通信线路的成本，但增加了网络复杂性。网络中除最低层节点及其连线外，任一节点或连线的故障均影响其所在支路网络的正常工作。



**网状拓扑结构**：网状结构分为全连接网状和不完全连接网状两种形式。全连接网状中，每一个节点和网中其它节点均有链路连接。不完全连接网中，两节点之间不一定有直接链路连接，它们之间的通信，依靠其它节点转接。这种网络的优点是节点间路径多，碰撞和阻塞可大大减少，局部的故障不会影响整个网络的正常工作，可靠性高；网络扩充和主机入网比较灵活、简单。但这种网络关系复杂，建网不易，网络控制机制复杂。广域网中一般用不完全连接网状结构。



**蜂窝拓扑结构**：蜂窝拓扑结构是无线局域网中常用的结构。它以无线传输介质（微波、卫星、红外等）点到点和多点传输为特征，是一种无线网，适用于城市网、校园网、企业网。



# ****四．Linux命令学习****

## 1.怎么查询linux系统和aix系统的逻辑cpu核数，内存，硬盘，频率，网卡?

答：**linux**

#uname -a 查看内核/**操作系统**/CPU信息

# head -n 1 /etc/issue # 查看**操作系统版本(**CentOS release 5.4 (Final) CentOS版本5.4)

# cat /proc/cpuinfo # 查看**CPU信息（配置）cat /proc/meminfo静态的**

top能显示系统**内存**。我们常用的Linux下查看内容的专用工具是free命令。

下面是对内存查看free命令输出内容的解释：

total:总计物理内存的大小。

used:已使用多大。

free:可用有多少。

Shared:多个进程共享的内存总额。

Buffers/cached:磁盘缓存的大小。

df:**硬盘（Used使用的 Available空闲的 Use%使用过的%多少 ）df –lh**

ifconfig**网卡**

**频率：**内存频率dmidecode CPU频率sudo more /proc/cpuinfo （cpu MHz : 2500.000）

**查看cpu个数及型号**

        # cat /proc/cpuinfo | grep name | cut -f2 -d: | uniq –c

**.查看物理cpu个数**

        # cat /proc/cpuinfo | grep physical | uniq –c

**物理CPU个数;** cat /proc/cpuinfo | grep "physical id" | sort | uniq | wc -l

逻辑CPU个数：cat /proc/cpuinfo |grep"processor" | uniq –c

**查看网卡信息**

# dmesg | grep eth

*e1000: eth0: e1000\_probe: Intel(R) PRO/1000 Network Connection*

E1000：eth0的：e1000\_probe：英特尔（R）PRO/1000网络连接

*e1000: eth0: e1000\_watchdog\_task: NIC Link is Up 1000 Mbps Full Duplex, Flow Control: None*

E1000：eth0的：e1000\_watchdog\_task：NIC Link是1000 Mbps的全双工，流量控制：无

*eth0: no IPv6 routers present*

为eth0：没有IPv6路由器介绍

**aix系统**

uname 显示**操作系统信息**

prtconf就可以查看系统所有的信息 **cpu** 内存 硬盘

topas**内存**查看内存使用情况ps aux

**物理网卡**netstat -v 或lscfg -vp entX（netstat –in）

**频率：**

查看内存   
使用命令#  lsdev -Cc memory   
或者   
#bootinfo -r   
查看物理内存   
prtconf就可以查看系统所有的信息 cpu 内存 硬盘等..   
显示每一个CPU的频率   
#pmcycles -m   
查看机器型号，SN   
＃uname –Mu

**Linux cpu主频**

**processor : 0**处理器：0

**vendor\_id : GenuineIntel** vendor\_id：GenuineIntel

**cpu family : 6** CPU系列：6

**model : 23**型号：23

**model name : Intel(R) Core(TM)2 Quad CPU Q8300 @ 2.50GHz**型号名称：英特尔（R）酷睿（TM）2四核CPU Q8300@2.50GHz

**stepping : 10**步进：10

**cpu MHz : 2500.000** CPU兆赫：2500.000

**cache size : 2048 KB**缓存大小：2048 KB

**fpu : yes** FPU：

**fpu\_exception : yes** fpu\_exception：是

**cpuid level : 13** CPUID级别：13

**wp : yes** WP：是

**flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss syscall nx lm constant\_tsc up pni cx16 lahf\_lm**

标志：FPU VME DE PSE TSC MSR PAE MCE cx8 APIC SEP MTRR PGE MCACMOV拍拍pse36 CLFLUSH DTS ACPI MMX fxsr SSE SSE2 SS系统调用NX LMconstant\_tsc起来PNI CX16 lahf\_lm

**bogomips : 5000.00** bogomips：5000.00

**clflush size : 64** CLFLUSH大小：64

**cache\_alignment : 64** cache\_alignment：64

**address sizes : 40 bits physical, 48 bits virtual**地址大小：40位的物理，虚拟48位

**power management:** 电源管理：

## 2.linux和aix上怎么实际监控cpu利用率 进程，内存使用率？

答：vmstat,topes,iostat,sar等

sar 命令报告 CPU 的使用情况，I/O 以及其它系统行为。iostat 命令主要通过观察物理磁盘的活动时间以及他们的平均传输速度。

内存监控：

vmstat命令报告关于核心线程，虚拟内存，自陷（trap），磁盘以及CPU行为的统计。而且每种行为报告都被更细致地用百分比分别表示用户态、核态、空闲以及等待磁盘I/O等情况。

Top监控工具可以显示CPU占用率为前几位的进程，并提供CPU的实时活动情况。它是一个动态显示过程，即可以通过用户按键来不断刷新当前状态。如果在前台执行该命令，它将独占前台，直到用户终止该程序为止。

> /home/a.txt

## 3．nmon工具怎么安装，怎么使用，怎么看cpu，内存使用率？

nmon的**安装步骤如下**：

1）用root用户登录到系统中；

2）建目录：#mkdir /test；

3）把nmon用ftp上传到/test，或者通过其他介质拷贝到/test目录中；

4）执行授权命令：#chmod +x nmon。

**Linux**

1. 安装nmon

下载nmon\_x86\_11f.zip，上传到linux下，解压后改文件名为nmon。

打开nmon目录，有多个系统的nmon版本，选择一个nmon，修改权限。如：chmod 777 nmon\_x86\_fedora5。

即完成nmon工具的安装。

2. 使用nmon

Nmon工具使用非常的简单。

启动该程序。使用命令：./ nmon\_x86\_fedora5 –fT –s 5 –c 100。命令的含义是：-f输出文件，-T输出最耗资源的进程，-s收集数据的时间间隔（秒），-c收集数据的次数。

或后台运行nmon，使用命令：nohup ./ nmon\_x86\_fedora5 –fT –s 5 –c 100。

3. 输出.nmon文件

在nmon执行完后，会在当前目录下输出.nmon文件。

输出文件的命名方式是服务器名\_日期时间.nmon。

4. 分析数据

输出.nmon的数据文件后，我们可以使用nmon工具的配套软件“nmon analyser v339.xls”进行数据的分析了。

打开“nmon analyser v339.xls”，按【analyse nmon data】按钮，选择要分析的.nmon文件，即输出分析的图标和数据xls表。

（若出现安全性提醒的框，在安全设置里将宏的安全性设为低即可。）

其中需要关注的几项系统性能数据为：

CPU\_ALL，CPU资源的使用情况；

DISK\_SUM，磁盘的总体使用情况；

DISK\_BUSY，磁盘忙的比例；

DISK\_READ，读磁盘的速率；

DISK\_WRITE，写磁盘的速率；

MEM，内存占用情况，包括系统缓存；

TOP，系统的top信息。

得到以上数据，可以作图平均响应时间，QPS，内存，磁盘读写等趋势图。总之，nmon工具在于分析进程的性能瓶颈，在于CPU，磁盘还是内存等。

**AIX**

**使用-**[**安装**](javascript:;)

首先下载 nmon 工具,下载时需注意要下载对应的系统版本。我们这里以 AIX 为例,下载的是nmon4aix\_11e.tar.gz 版本。下载后解压nmon4aix\_11e.tar.gz（里面包含README\_v11.txt、nmon\_aix51、nmon\_aix52ml5、nmon、nmon\_aix52ml2nmon\_aix53）,将文件夹名改为 nmon 并上传到 Linux 中(任何一个文件夹下均可以)。

为了能够正常的使用 nmon 工具,我们需要修改相关文件的权限。

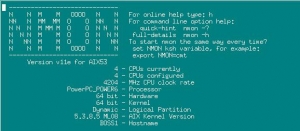
tbcs@BOSS1:/interface/interface/nmon$chmod 755 nmon nmon\_aix51nmon\_aix52ml2 nmon\_aix52ml5 nmon\_aix53

http://www.itpub.net/forum.php?mod=image&aid=765460&size=300x300&key=0b2c691ba59decad10f6b75743adb335&nocache=yes&type=fixnone

[1.jpg](http://newdx2.itpub.net/forum.php?mod=viewthread&tid=1573236)

现在可以使用nmon了。

Nmon工具使用比较简单,在nmon 目录下,输入 ./nmon

[[](http://newdx2.itpub.net/forum.php?mod=viewthread&tid=1573236)  
2.jpg](http://newdx2.itpub.net/forum.php?mod=viewthread&tid=1573236)

出现此画面，说明已安装成功。（这是AIX版本的画面，其他版本画面会稍有不同）

输入c可显示CPU的信息,“m”对应内存、“n”对应网络,“d”可以查看磁盘信息；“t”可以查看系统的进程信息；”

[[](http://newdx2.itpub.net/forum.php?mod=viewthread&tid=1573236)  
3.jpg](http://newdx2.itpub.net/forum.php?mod=viewthread&tid=1573236)

**三、数据采集**

以上实现的是对服务器的实时监控，在实际的性能测试中我们需要把一段时间之内的数据记录下来，可以使用如下命令

./nmon –f –t test –s 30 –c 10

上面命令的含义是：

-f ：按标准格式输出文件名称：<hostname>\_YYYYMMDD\_HHMM.nmon

-t输出最耗资源的进程  
-s：每隔n秒抽样一次，这里为30秒  
-c：取出多少个抽样数量，这里为10，即监控=10\*30/60=5分钟

test:监控记录的标题

该命令启动后，会在nmon所在目录下生成监控文件，并持续写入资源数据，直至360个监控点收集完成——即监控1小时，这些操作均自动完成，无需手工干预，测试人员可以继续完成其他操作。如果想停止该监控，需要通过“#ps –ef|grep nmon”查询进程号，然后杀掉该进程以停止监控。

nmon 输出文件的命名方式是服务器名\_日期时间.nmon

通过以下命令，将.nmon文件转化成.csv文件，在当前目录生成对应的.csv文件

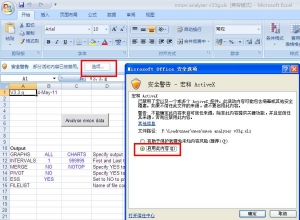
sort BOSS1\_110810\_1438.nmon>BOSS1\_110810\_1438.csv

  
 [4.jpg](http://newdx2.itpub.net/forum.php?mod=viewthread&tid=1573236)

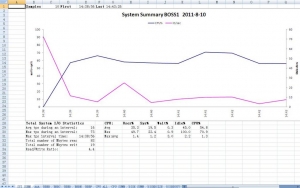
将BOSS1\_110810\_1438.csv文件下载到本地。通过nmon analyser工具（ nmon analyser v33g.xls）转化为excel文件。如何转化呢？步骤如下：

1） 下载后打开nmon analyser v33g.xls

2） 调整excel宏的安全级别，调整为最低或者如下操作

**** [**5.jpg**](http://newdx2.itpub.net/forum.php?mod=viewthread&tid=1573236)

3） 然后点击Analyser nmon [**data**](javascript:;) 按钮,选择下载下来的.csv文件，然后就会转化成.excel文件，生成图形化的文件，例如：



## 4.怎么清理linux环境的内存？

答：free –m

total 内存总数

used 已经使用的内存数

free 空闲的内存数

shared 多个进程共享的内存总额

buffers Buffer Cache和cached Page Cache 磁盘缓存的大小

-buffers/cache 的内存数:used - buffers - cached

+buffers/cache 的内存数:free + buffers + cached

可用的memory=free memory+buffers+cached

#sync

**cat /proc/sys/vm/drop\_caches**

要达到释放缓存的目的，我们首先需要了解下关键的配置文件/proc/sys/vm/drop\_caches。这个文件中记录了缓存释放的参数，默认值为0，也就是不释放缓存。他的值可以为0~3之间的任意数字，代表着不同的含义：

0 – 不释放  
1 – 释放页缓存  
2 – 释放dentries和inodes  
3 – 释放所有缓存

# 五．熟悉脚本并调试

## 1. 理解局部变量与全局变量？

答：**变量的作用：**变量存储在内存中不同的存储区域，将影响变量值的存在时间（即生存期）。 变量的生存期指程序在执行期间，变量存在的时间间隔，即从给变量分配内存至所分配内存被系统收回的那段时间。

**局部变量——内部变量**

定义：在函数模块内定义的，且只在本函数模块内有效。

说明：

（1）main中定义的变量只在main中有效

（2）不同函数中同名变量，占不同内存单元

（3）形参属于局部变量

（4）在复合语句中定义的有效变量

（5）局部变量可用存储类型：auto register static （默认为auto）

**全局变量---外部变量**

定义：在函数外定义，可为本文件所有函数共用

变量的作用域（有效范围）：从定义变量的位置开始到本源文件结束为止。

全局变量的作用域的特点：

（1）在全局变量与局部变量不同名时，其作用域是整个程序。

（2）在全局变量和局部变量同名时，在同名局部变量的作用域内，全局变量不起作用。

## 2.理解指针，字符串？

答：**指针**是一个用来指示一个内存地址的计算机语言的[变量](http://baike.baidu.com/view/296689.htm)或中央处理器(CPU)中[寄存器](http://baike.baidu.com/view/6159.htm)(Register)。指针一般出现在比较近[机器语言](http://baike.baidu.com/view/111847.htm)的语言，如汇编语言或C语言。[面向对象](http://baike.baidu.com/view/125370.htm)的语言如Java一般避免用指针。指针一般指向一个[函数](http://baike.baidu.com/view/15061.htm)或一个变量。在使用一个指针时，一个[程序](http://baike.baidu.com/view/17674.htm)既可以直接使用这个指针所储存的内存地址，又可以使用这个地址里储存的变量或函数的值。

**字符串**是字符串或串(String)是由零个或多个字符顺序所组成的数据结构。

Char是字符串

\*是指向的（是指针）一个空间或一个地址。

## 3.脚本调试方法设断点和分布调试怎么用？

答：通过断点可以使脚本在特定位置暂停执行。它可用于在执行期间的预定点处检查该脚本对应用程序的影响。**1** 将光标置于脚本中要停止执行的行上。

**2** 依次选择“插入” > “切换断点”，或者单击“调试”工具栏上的“断点”按钮。也可以按键盘上的 F9 键。将在脚本的左边距显示“断点”符号 ( )。

**3** 要禁用断点，请将光标置于包含断点符号的行上，然后单击“调试”工具栏上的“启用 / 禁用断点”按钮。“断点”符号中将会显示一个白点 ( )。禁用一个断点后，执行将在下一个断点处暂停。再次单击该按钮可以启用断点。

**分布：**要分步运行脚本，请执行下列操作：

**1** 依次选择“Vuser” > “分步运行”，或者单击“调试”工具栏上的“步骤”按钮。

VuGen 将执行脚本的第一行。

**2** 继续单击“步骤”按钮来执行该脚本，直到脚本运行完成为止。

F9设置断点，F10是单步调试。 F5是运行

## 4.怎么跟踪运行日志，输出脚本调试信息？

答：Extended log-----Parameter substitution

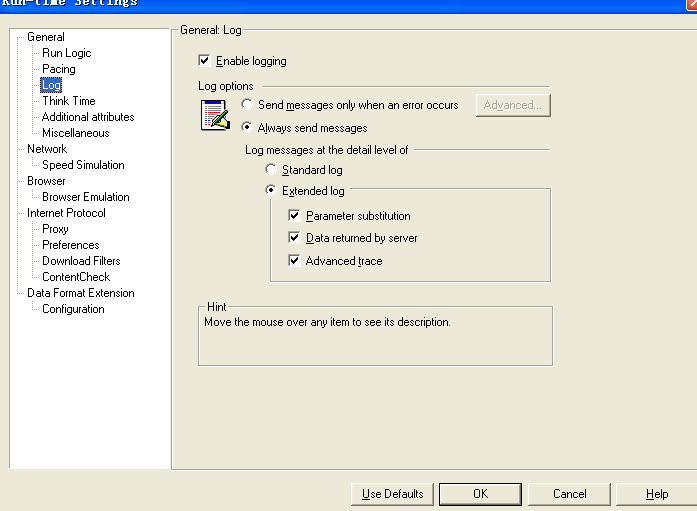
参数替换：选择此选项可以记录指定给脚本的所有参数及其相应的值

Extended log-----Data returned by server

选择此选项可以记录服务器返回的所有数据

xtended log-----Advanced trace 高级跟踪

选择此选项可以记录 Vuser 在会话期间发送的所有函数和消息。



【[**lr\_message**](http://www.51testing.com/?uid-6343-action-viewspace-itemid-13643)】  
 int lr\_message (const char \*format, exp1, exp2,...expn.);  
 中文解释：lr\_message函数将信息发送到**[日志](http://www.51testing.com/?uid-6343-action-viewspace-itemid-13643" \t "_self)**文件和输入窗口。在VuGen中运行时，输入文件为output.txt。

【lr\_log\_message】  
 int lr\_log\_message (const char \*format, exp1, exp2,...expn.);  
 中文解释：lr\_log\_message函数将消息发送到Vuser或代理日志文件（取决于应用程序），而不是发送到输出窗口。通过向日志文件  
    发送错误消息或[**其他**](http://www.51testing.com/?uid-6343-action-viewspace-itemid-13643)信息性消息，可以将该函数用于调试。

【lr\_error\_message】  
 int lr\_error\_message (const char \*format, exp1, exp2,...expn. );  
 中文解释：lr\_error\_message函数将错误消息发送到输出窗口和Vuser日志文件。要发送不是特定错误消息的特殊通知，请使用lr\_output\_message。

【lr\_output\_message】  
 int lr\_output\_message (const char \*format, exp1, exp2,...expn.);  
 中文解释：lr\_output\_message函数将带有脚本部分的行号的消息发送到输出窗口和日志文件。

【lr\_vuser\_status\_message】  
 int lr\_vuser\_status\_message (const char \*format);  
 中文解释：lr\_vuser\_status\_message函数向控制器或优化模块控制台的vuser窗口的“状态”区域发送字符串。它还将该字符串发送到vuser日志。从VuGen运行时，消息被发送到output.txt。

## 5.理解条件语句，常用的字符串处理函数？

答：if(表达式) 语句1;

如果表达式的值为非0，则执行语句1，否则跳过语句继续执行下面的语句。

如果语句1有多于一条语句要执行时, 必须使用{和} 把这些语句包括在其中, 此时条件语句形式为:

if(表达式)

{

语句体1;

}

if--else语句

除了可以指定在条件为真时执行某些语句外，还可以在条件为假时执行另外一段代码。在语句中利用else语句来达到这个目的。

if(表达式) 语句1;

else 语句2;

同样，当语句1或语句2是多于一个语句时，需要用{}把语句括起来。

例如：

if(x>=0) y=x;

else y=-x;

这种结构是从上到下逐个对条件进行判断,一旦发现条件满足就执行与它有关的语句, 并跳过其它剩余阶梯;若没有一个条件满足,则执行最后一个else 语句n。最后这个else常起着缺省条件的作用。同样,如果每一个条件中有多于一条语句要执行时,必须使用{和}把这些语句包括在其中。条件语句可以嵌套,这种情况经常碰到,但条件嵌套语句容易出错,其原因主要是不知道哪个if对应哪个else。

if--else if--else结构。

if(表达式1)

语句1;

else if(表达式2)

语句2;

else if(表达式3)

语句3;

在Loadrunner中提供了C对字符串出了的相关功能函数，如下：

|  |  |
| --- | --- |
| **lr\_save\_datetime** | 把当前日期和时间保存到一个参数中 |
| **lr\_save\_int** | 把一个整数保存为参数 |
| **lr\_save\_searched\_string** | 保存一个字符数组相对于字符串出现的部分。 |
| **lr\_save\_string** | 把一个字符串保存到参数中 |
| **lr\_save\_var** | 把字符串的一部分内容保存为参数 |
| **lr\_eval\_string** | 返回参数的实际内容或返回一个包含参数的字符串的实际内容 |

## 6. 理解什么时候使用退出本次循环什么时候退出当前用户?

答：**本次循环：**1.在场景设定时间到了，退出本次循环。2.当脚本报错无法进行下一次操作时，退出本次循环（当实际和预期不一致的时候退出本次循环）。3.当脚本运行完毕后，退出本次循环。

**当前用户：**1.当脚本进行迭代时，登录放在Action里，运行下一次迭代时，退出当前用户。

## 7. diCryptoSys和diCrPKI常用函数的左右，每个参数的意义?

## 8. 理解发送一个请求到解析返回报文的过程？

## 9.脚本中重要函数lr\_xml\_get\_values，memset，Aes\_Off，Aes\_On，AddHMac，EncryptPwd？

答：lr\_xml\_get\_values 函数在 XML输入字符串 XML 中查询匹配查询条件的值。Aes\_Off解密，Aes\_On加密，memset功能名

result = lr\_xml\_get\_values("XML={response}",

"ValueParam=error",

"Query=/error/@string",

"NotFound=Continue",LAST);

memset(Data2,0,10240);

Aes\_On(Data2,lr\_eval\_string("{BODY}"),ClientKey,ClientIV);

AddHMac(Data2,strlen(Data2),ClientMac,20);

lr\_save\_string(Data2,"POSTBODY");

memset(Response\_2,0,10240);

Aes\_Off(Response\_2,Response,ServerKey,ServerIV); //解密

lr\_save\_string(Response\_2,"response");

lr\_convert\_string\_encoding(Response\_2,LR\_ENC\_UTF8 , LR\_ENC\_SYSTEM\_LOCALE, "response2");

lr\_message("登录返回结果【%s】\n",lr\_eval\_string("{response2}"));

# 六、基本问题排查

## 1.排除问题的常用过程？

## 2. 项目中遇到问题举例？

# 七、方案与报告文档

## 1. 参考《方案模板》，总结出要点？

**2. 参考《测试用例模板》，总结出要点？**

答; 用例编号、用例名称、性能测试需求、测试目案例描述的、环境需求、前提条件、序号、场景类型、测试步骤、数据说明、监控指标及预期结果（TPS：，HPS：，ART：，交易成功率：，CPU%：，内存使用率：）、实际结果

## 3. 参考《报告模板》，总结出要点？

答：测试目的、测试环境、测试工具、业务场景描述、场景执行结果和资源占用情况（单业务是找最佳用户、混合业务测试、疲劳业务测试）

# 八、学习知识汇总

## 1. 参考《RYT性能项目汇总》，总结出要点？

## 2. 据一个月的学习，从项目入手需要准备的资料，环境，开始描述到最终形成报告的过程，和过程需要掌握的知识？