

**Introduction to Web
Application Load Testing**

Philip Johnson
Collaborative Software Development Laboratory
University of Hawaii

(1)

Concepts and terms

(2)

Testing, Profiling, Load Testing

Functional testing: <ul style="list-style-type: none">• Determine whether application implements desired capabilities correctly.	Non-functional testing: <ul style="list-style-type: none">• Determine whether functionally correct program provides adequate responsiveness.
Profiling: <ul style="list-style-type: none">• Examine a single software application's CPU and memory usage.• Result: Detect most significant performance problem in software application and remove it.	Load testing: <ul style="list-style-type: none">• Measure the applications ability to sustain a number of simultaneous users and/or transactions with adequate response times.• Result: changes to hardware, software infrastructure, or application design

(3)

Active vs. Passive Monitoring

Active Monitoring:

Generate traffic data from virtual users.

- Provides early warning of problems
- Measures against a baseline
- Enables pro-active problem solution before users impacted

Passive monitoring:

Collect traffic data from real users

- Provides actual level of users and their locations
- Non-intrusive
- Helps prioritize issues

You want both active and passive monitoring.

⁽⁴⁾

Flavors of Load Testing

Load testing:

- Test system against a specified number of users.
- Issue: can system handle users correctly with acceptable responsiveness?

Stress testing:

- Test system against specified number of users over long period of time.
- Issue: is system stable and reliable under sustained load?

Capacity testing:

- Test system against an increasing number of users.
- Issue: at what load does system begin failing to

⁽⁵⁾

Warning!

"Capacity" load testing is a denial-of-service attack!

When testing across network, you must:

- Think through impact on network and other users of performing test
- Notify and obtain permission from network administrators or ISPs
- Schedule test for off-hours

Failure to observe appropriate "netiquette" can lead to reprisals.

Do NOT:

- Capacity load test using katrina or ICS lab

⁽⁶⁾

Diagnosing web app performance problems is hard

My web application is slow. Should I:

- Add RAM?
- Move from Windows to Linux?
- Optimize my application code?
- Allocate more memory to the JVM?
- Move from Tomcat to Resin? To Apache?
- Move the database to its own server?
- Change the database indexing?
- Move from InstantDB to Oracle?
- Upgrade my Internet bandwidth?
- Use Enterprise Java Beans?

(7)

Load Testing Process

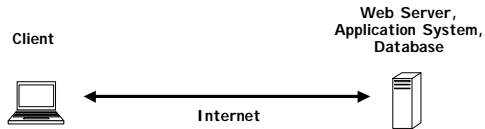
1. Determine load testing research question(s):
 - Should we upgrade to new server or software?
 - Where are bottlenecks in current system?
 - How many concurrent users without failure?
2. Determine the system data to collect
 - Response time, CPU usage, RAM usage, Page errors
3. Determine the testing tool to use
 - Basic: JMeter, WebApp Stress Tool
 - Enterprise-class: Mercury Interactive, etc.
4. Determine baseline data values
 - What is the hardware/software configuration of web app?
 - What are load test results under normal conditions?
5. Collect experimental data values
 - Collect load test data with new load, hardware, or software
6. Determine changes (if any) required to system.

(8)

Web Application Architectures

(9)

Simple client-server architecture

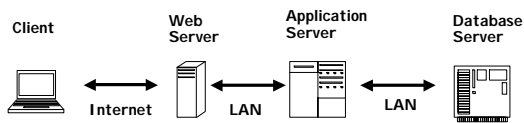


Six variables impact on performance:

- Internet bandwidth (T-1, 56K modem, etc.)
- Server hardware configuration (RAM, Disk speed, OS)
- Web server software (Tomcat vs Resin, etc.)
- Database server software (InstantDB, db40, etc.)
- Application design (caching, algorithms, etc.)
- Database design (indexes, tables, queries)

(10)

Simple N-tier architecture

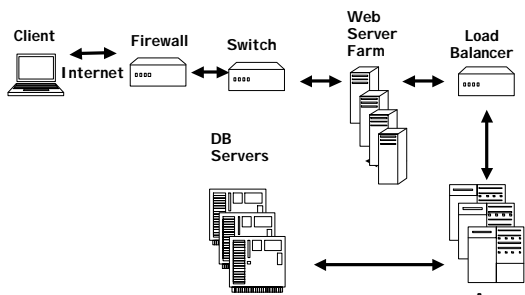


10 variables impact on performance:

- Internet bandwidth
- Web server hardware, software, app design
- App server hardware, software, app design
- DB server hardware, software, DB design

(11)

Industry-strength N-tier architecture



Don't try this at home, kids.

(12)

Web app symptoms and typical underlying problems

(13)

Symptoms of web app problems

Long user response time
Long server response time
Memory leaks
High CPU usage
Too many open connections
Length queues for requests
Too many table scans of database
Database deadlocks
Erroneous data returned
HTTP errors
Pages not available

(14)

Typical database problems

Insufficient indexing
•Tune database indexing to improve query processing
Fragmented databases
•Place table records into adjacent pages.
Out-of-date statistics
•Degrade query optimizer performance
Faulty application design
•Excessive DB calls, excessive data requests

(15)

Typical Web Server problems

Poor server design

- Inefficient data or page caching
- Use of Active Server Pages

Memory problems

- Physical memory constraints

High CPU usage

- Usage > 70% indicates problems

(16)

Typical app server problems

Poor database tuning

- Application server sending too many DB requests

Poor cache management

- Produces high CPU usage, disk access

Poor session management

- Produces high CPU usage, disk access
- Time-outs

Poor security design

- Excessive use of https protocol.

(17)

Network problems

Requires specialized testing to uncover (must stress network without causing application/db stress).

Potential sources of network problems:

- Firewall throughput
- Internet access throughput
- Load balancers, gateways, routers

(18)

Load testing tools

(19)

Example Load Testing Tools

Enterprise Solutions:

- Astra LoadTest, Mercury Interactive
 - \$10K for 50 v-users, \$30K for 250 v-users
- RadView WebLoad
 - \$10K for 100 v-users, \$38K for 1K v-users

Personal Solutions:

- Microsoft Web Application Stress Tool
 - Free from webtool.rte.microsoft.com
 - Windows-specific
- JMeter
 - Open source
 - Cross-platform

We will look at JMeter

(20)

A load testing scenario

1. Determine load testing research question(s):
 - What is the baseline performance of EmailFinder?
2. Determine the system data to collect
 - User response time
 - Server CPU and memory usage
 - Web app memory usage
 - Any failures
3. Determine the testing tool to use
 - JMeter
4. Determine baseline data values
 - Server hardware: PIII/700, 384m RAM, FastEthernet LAN

Let's look at the JMeter configuration required to provide more baseline data.

(21)

Downloading, Installation, Invocation

Go to <http://jakarta.apache.org/jmeter/>
Download and expand the JMeter application zip file.
Change directory to jakarta-jmeter/bin
Invoke jmeter.bat (or jmeter on unix).
Don't forget to RTFM:
• Online documentation
• www.mail-archive.com/jmeter-user@jakarta.apache.org
• Documentation is not spectacular.

(22)

Initial screen



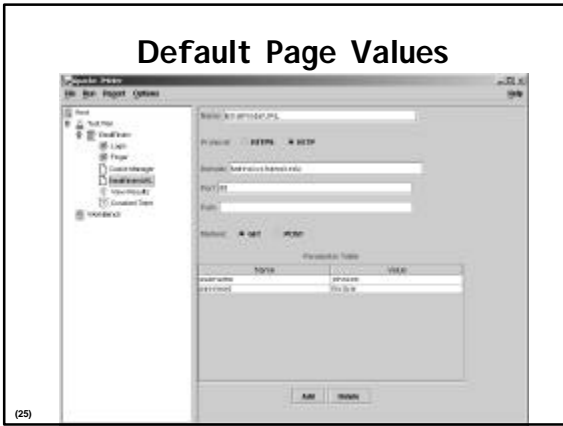
(23)

Top-level Thread Group

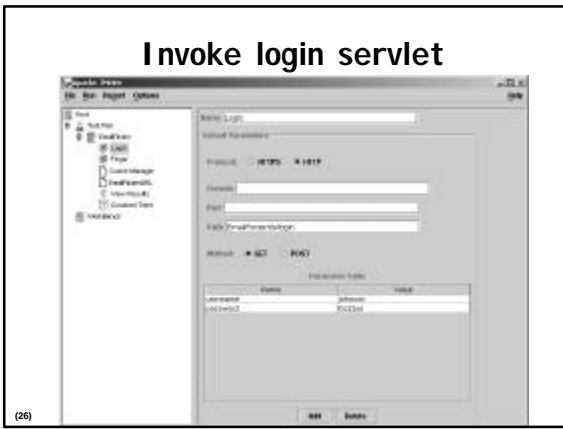


(24)

Default Page Values



Invoke login servlet



Specify session cookie

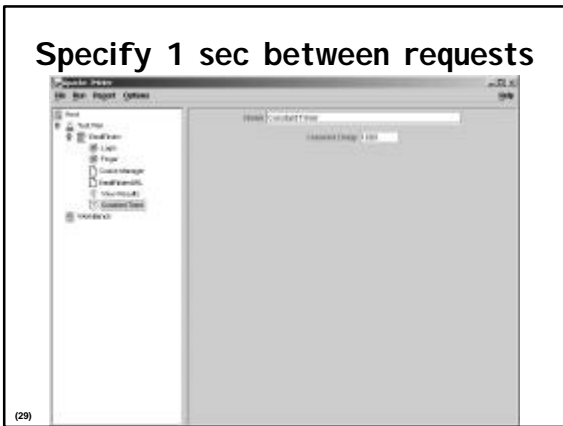


Invoke finger servlet



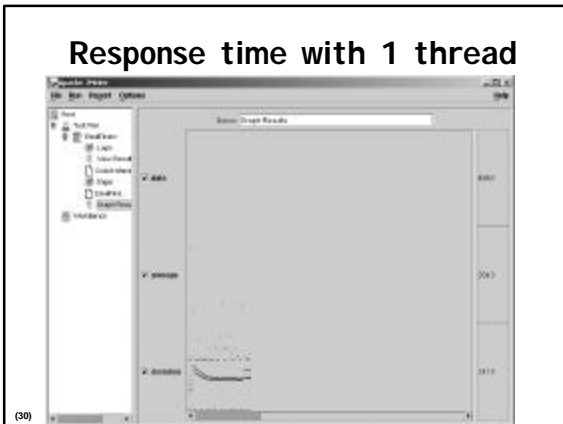
(28)

Specify 1 sec between requests



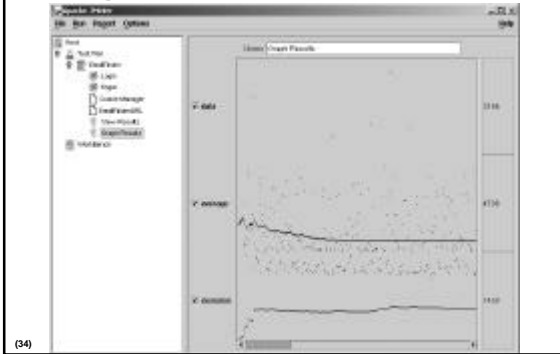
(29)

Response time with 1 thread



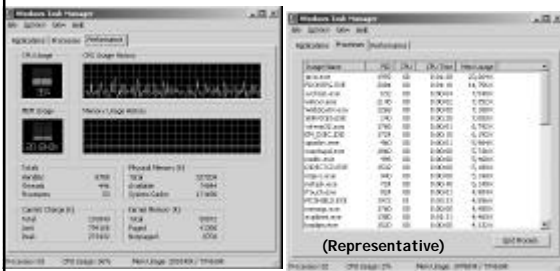
(30)

Response time with 10 threads



(34)

Server stats, 10 threads



(Representative)

(35)

Conclusions

With 10 threads at rate of 1 page/second,

- Response time averages 5 seconds.
- Server CPU utilization at around 30%
- Web server process requires 25MB.

Load test methodology problems:

- Web server used for other web apps
- Did not vary search term, session ID.
- Stress testing (12-24 hr) not performed
- Capacity testing not performed.
- No checking for correct page response as load increased.

Next steps:

- Check server under stress testing conditions.
- Fix methodology problems.
- Repeat with alternative hardware, software.

(36)

JMeter Tips

Use 1.6.1 for now. (I had problems w/1.7b)

Make sure you load your .jmx file inside of "Test Plan", otherwise it isn't executed.

Possible additions to jmeter.bat's java cmd:

- Xms64m -Xmx128m (increase memory)
- Dsun.java2d.noddraw (eliminate GUI flicker)

When entering a field value in GUI, make sure to exit field so that value is seen!

(37)

JMeter pros and cons

Pros:

- Open source
- Active development
- Free
- Cross-platform
- Extensible architecture support "plug-ins"
- Good project to help work on.
- Best free alternative I could find.

Cons:

- Documentation marginal
- How to automate 100's of separate v-users with separate session IDs?
- Unclear how to detect failure (i.e. 404s)--- maybe assertion?

(38)

Summary

(39)

Typical order of fixes

1. Improve current application design.
 - Algorithms, caching, DB calls, memory use
- 2a. Upgrade hardware
 - RAM, CPU, network bandwidth
- 2b. Upgrade software infrastructure
 - OS, web server, database
3. Upgrade system architecture
 - Client-server to basic n-tier
 - Basic n-tier to enterprise n-tier
 - Requires software and hardware changes.

(40)

Keep it in perspective

Functional correctness is more important than non-functional correctness.

Not every WebApp requires scalability.

High volume web sites require professional load testing.

(41)
