XMPP Performance Testing using Open Source Tools (Tsung & JMeter)

Ramya Ramalinga Moorthy

by Neotys

AGENDA

- ❑ Application Background
- ❑ Importance of Realistic Workload Model Creation
- ❑ Performance Testing Strategy
- ❑ XMPP Performance Testing using Tsung – Lessons Learnt
- ❑ TSung / JMeter – Benefits & Challenges

Application Background

Messenger Application Background

- **System Under Test (SUT)** - The messenger application empowers millions of users to stay in touch with their connections.
- The messenger client application is available for Mobile devices (Android & IOS).
- The server infrastructure is hosted on AWS cloud.
- **Scope of the Engagement**
  - Performance & Scalability assessment for messenger application
  - Capacity sizing analysis to meet the projected load level for 2 years

Engagement Scope

- **Mobile Device Performance analysis**
  - Android device
  - IOS device
- **OpenFire XMPP Server Performance & Scalability Benchmarking**
- **Server Capacity Sizing**
  - XMPP Server (XMPP workload)
  - Billing Server (API workload)
  - Service Server (API workload)
  - DB Server (XMPP & API workload)

Workload Model Development

Introduction to XMPP

- E**X**tensible **M**essaging and **P**resence **P**rotocol (XMPP) – popularly used Instant messaging protcol
- XMPP is originally named as Jabber and it is the popular communication protocol for message oriented middleware based on XML
- Enables Client-Server communication by XML streams. Popular XMPP servers include ejabberd, OpenFire, MongooseIM, etc.
- User's contacts storage is called Roster
- XML Stream consists of XML blocks called Stanzas
  - Presence stanza
  - Message stanza

XMPP Workload Profile

- ❑ **Connect to Server**
- ❑ **Login to the Server**
- ❑ **Send Presence Message**
- ❑ **Perform Roster actons**
  - ➢ Get Roster
  - ➢ Add / Remove user
- ❑ **Perform Room actions**
  - ➢ Search rooms / Get Room Users
  - ➢ Join / Leave chat room
  - ➢ Send & Collect incoming messages
- ❑ **Perform Group actions**
  - ➢ Create/Delete groups
  - ➢ Add/Remove Room users
  - ➢ Send & Collect incoming messages
- ❑ **Perform Conference actions**
  - ➢ Join / leave Conference
- ❑ **Disconnect from Server**

- ❖ **Total Number of XMPP Messages : 25**
- ❖ **Average XMPP Messages per User profile : 10-12**
- ❖ **Number of User profiles created : #5**
  - **Type 1 - Idle User (Set Presence)**
  - **Type 2 - Simple user (Roster Actions)**
  - **Type 3 - Room User (Room actions)**
  - **Type 4 - Group User (Group actions)**
  - **Type 5 - Conference User (Conference actions)**

XMPP Test Scripts – XMPP Server Benchmarking

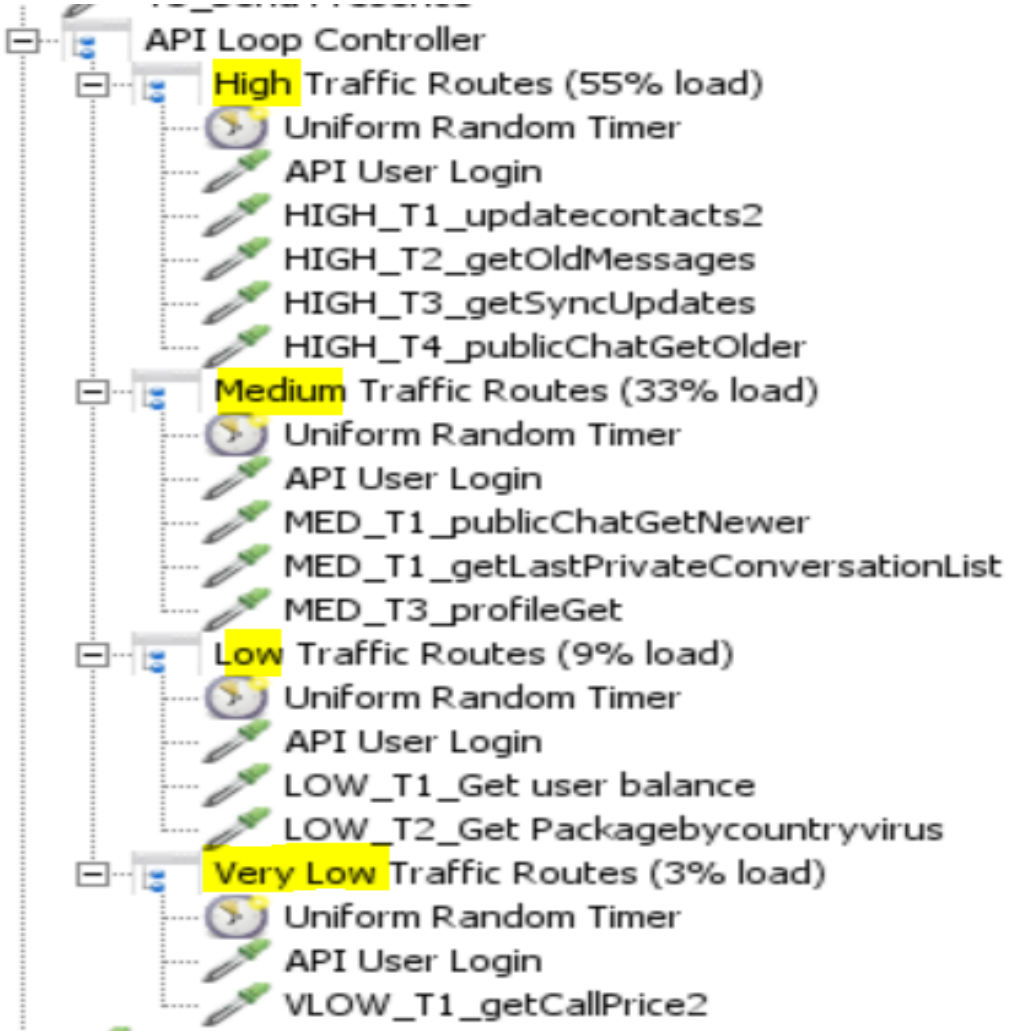
- **TSung Test Scripts : #5 User Profiles**
  - User Profile #1 (Idle User)
  - User Profile #2 (Simple User)
  - User Profile #3 (Room User)
  - User Profile #4 (Group User)
  - User Profile #5 (Conference User)

API Workload Profile

- ❑ User Balance
- ❑ Get last chat messages
- ❑ Update contacts
- ❑ Get Old messages
- ❑ Get old chat messages
- ❑ Get balance
- ❑ Get user profile
- ❑ Get Sync updates from devices
- ❑ Update User Profile
- ❑ Get Call rates
- ❑ Get Country details

- ❖ **Total Number of APIs : 20**
- ❖ **Average APIs used per User profile : 10-12**
- ❖ **Number of User profiles created : #4**
  - **Type 1 - High Frequently used APIs**
  - **Type 2 – Medium Frequently used APIs**
  - **Type 3 – Low Traffic APIs**
  - **Type 4 – Very low Traffic APIs**

API Test Plan – Service & Billing Server Benchmarking



- **JMeter Test Scripts : #4 User Profiles**
  - User Profile #1 : HIGH Traffic APIs
  - User Profile #2 : MEDIUM Traffic APIs
  - User Profile #3 : LOW Traffic APIs
  - User Profile #4 : VERY LOW Traffic APIs

XMPP & API – Realistic Workload used for Performance Assessment

User Load Distribution in Tsung				
Tsung	Tsung	Tsung	Tsung	Tsung
20%	15%	30%	25%	10%
1. Idle User Profile	2.Simple User Profile	3.Room User Profile	4.Group User Profile	4.Conference User Profile

User Load Distribution in JMeter	
JMeter	JMeter
60%	40%
Room User Profile + API User Profiles (1 to 4)	Group User Profile + API User Profiles (1 to 4)

Performance Test Strategy

Performance Test Strategy - XMPP Server Benchmarking

Phase 1 : Server Performance Benchmarking for XMPP Workload (#5 user profiles)

- Performance & Scalability Assessment for active user connections with an empty roster
- Performance & Scalability Assessment for active user connections with avg roster size of 50
- Performance & Scalability Assessment for active user connections with realistic roster size
  - 50% users – 35 contacts ; 25% users – 10 contacts ; 25% users – 100 contacts
  - Target of 200 logins/sec (1200 XMPP packets per second for a peak load steady state duration of 1 hour)
- Tested User Loads
  - 50K active user connections
  - 100K active user connections
  - 0.5 million active user connections
  - 1 million active user connections

Tool Used : Tsung 1.7.0
1 controller and #5 load gen client machines in AWS (Ubuntu OS instances)

Performance Test Strategy – Service & Billing Server Benchmarking

Phase 2 : Server Performance Benchmarking for API Workload (#4 user profiles)

- Performance & Scalability Assessment
  - 5K active user connections
  - 10K active user connections
  - 20K active user connections
    - 15000 Calls per second on Service Server & 1500 Calls per second on Billing Server (for the target data volumes on DB)

Tool Used : JMeter 5.1.1
1 controller and #8 load gen client machines in AWS
(Ubuntu OS instances)

Performance Test Strategy – All Servers

Phase 3 : Server Performance Benchmarking for XMPP + API Workload

- Performance Benchmarking for #5 XMPP User profiles and #4 API User profiles.
  - 10K active user connections
  - 50K active user connections
  - 100K active user connections
  - 0.5 million active user connections
  - 1 million active user connections

- Capacity Sizing for all Server components (XMPP server , Service server, Billing server and DB server) based on realistic load test results

Tool Used : Tsung 1.7.0 + JMeter 5.1.1 (with XMPP Plugin)

Tsung & OpenFire XMPP Server Details

Introduction to TSung

- Tsung is developed in Erlang that gives the high performance and scalbility characteristics – making it a best tool choice for high volume distributed load testing.
- It has the ability to simulate large number of simultaneous users distributed across regions from cloud
- Tsung supports installation on Linux, Solaris & Mac OS.
- 1 Million concurrent users load was created from **#5 instances of m4.xlarge** on EC2 for XMPP load testing.
- No recorder available for XMPP protocol. User session to be created as an XML script.
- XML script will not be parsed. Jabber plugin relies on Packet Acknowledgements.

What makes Tsung a best choice for XMPP Performance Testing

- Message Acknowledgement
  - Ack : Local
  - Ack : Global
- User Ramp up Settings
- Authentication support
  - SASL Plain
  - SASL Anonymous
- Read Parameter values from CSV file
- Roster Management
  - Addition of users
  - Deletion of users
  - Setting presence
- Think time settings
- Presence Status Options
  - Presence : broadcast
  - Presence : directed
- MUC Operation Support
  - Join / Exit room
  - Send message
- HTML Report Creatiion (tsung_stats.pl)
- Tsung Summary Report & Graphical Report
- Test Results comparison using Tsung Plotter

XMPP Performance Testing using Tsung/JMeter - Benefits & Challenges

XMPP Performance Testing Challenges

- Huge test data creation (3 million users with their realistic roster size + 1 million groups/rooms + 0.5 million conferences + chat history) using DB scripts & Tsung scripts.
- Creation of realistic workload model based on production log analysis & inputs from business stakeholders.
- Continuous configuration tuning activities on OpenFire XMPP server machine to accept 1 million users connections.
- Enhancements & debugging of Tsung scripts (on Mac / Ubuntu machines) to use the required tool features and meet the target load levels thru distributed load testing.
- Ensuring quick learning curve on XMPP and Tsung to meet the aggressive engagement timelines.

Tsung / JMeter – Benefits & Challenges for XMPP Testing

#	TSung	JMeter
1	Installation of Tsung along with pre-requisites (Erlang, Perl5 & Gnuplot) • For POC – Local Mac OS used • For Testing – AWS Ubuntu instances used with GUI installation	Straight forward installation
2	Excellent load generation capability leading to huge cost savings (though high end load generation machine required to create 1 million users)	Roughly 30-40% load generation capability compared to Tsung instance (leading to less cost savings)
3	Lacks good documentation/forum support resulting detailed self exploration	Excellent documentation/forum support hence very quick learning curve required
4	Knowledge in writing XML scripts is a pre-requisite leading to high script development time (though this approach gives protocol independence)	Not much knowledge on programming required due to rich GUI leading to quick script development time.
5	Very good Reporting capability (HTML & Graphical reports) using additional plugins	Good built-in reporting capability & with additional plugins for reporting

Thank you !!