SIMPLEStone – A presence server performance benchmarking standard

Vishal K. Singh and Henning Schulzrinne

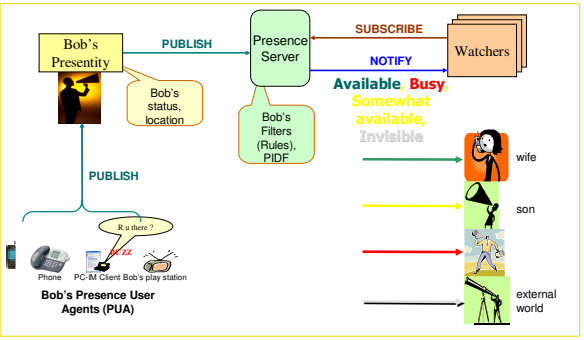
Columbia University in the City of New York

Abstract

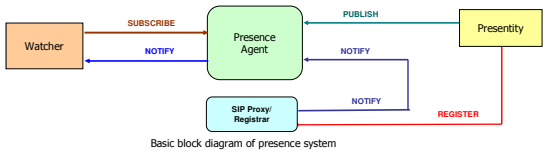
Presence is an important enabler for communication in IP based telephony systems. Presence based services depend on accurate and timely delivery of presence information. Hence, presence systems need to be appropriately dimensioned to meet the growing number of users, varying number of devices for every user as sources of presence, the rate at which they update presence information to the network and the rate at which network distributes the user's presence information to the watchers. SIMPLEStone proposes a simple set of metrics for evaluating and benchmarking the performance of SIMPLE based presence system. SIMPLEStone benchmarks the presence server by generating requests based on a work load specification. SIMPLEStone proposes to measure server capacity in terms of request handling capacity as an aggregate of all types of requests and the capacity of the server to handle individual request types.

What is Presence

Ability and willingness to communicate.
Rules about how and what part of presence info can be accessed
More detailed information includes location, preferred communication mode, current mood and activity



Presence Components

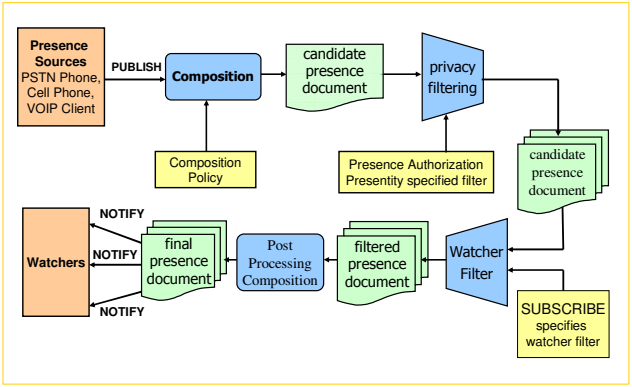


Basic block diagram of presence system

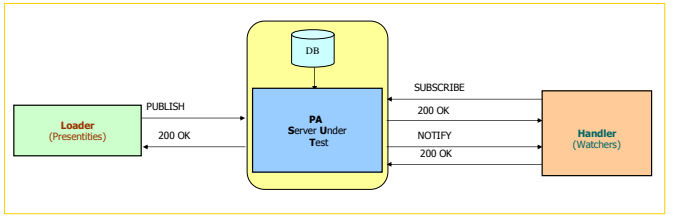
Presence Operations

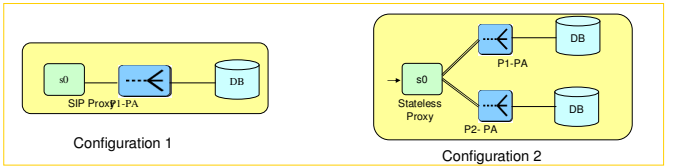
- Subscription
  - Subscribe to entities
  - Authentication of subscribers
  - Subscribers specify subscription rules
- Notification
  - Updating presence state to watchers
  - Delivering presence data
  - Send notifications to the watcher in a scalable manner in real time
- Publication
  - Send information to the server for distribution
  - Multiple sources for a single address
  - Updates communications means, and capabilities
  - Rate of change of data

Presence Server Data Processing



SIMPLEStone Architecture





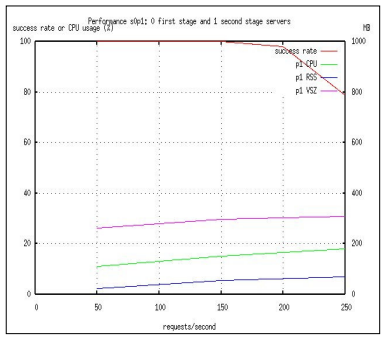
Presence Server Benchmarking

- To make informed, accurate decisions, presence-based services depend on the timely delivery of presence information to watchers
- Capacity planning and dimensioning
  - A service provider needs to know how many servers are good fora given user population
  - A server software vendor needs to specify the capacity of his server
  - Provisioning Network bandwidth
- Different servers and hardware platforms
  - A uniform evaluation and performance testing methodology
  - Benchmarking server software and hardware platform performance
- Repeatability of tests for acceptance testing after an upgrade or change in network topology
- Measure the performance of each components of presence server
  - SUBSCRIBE-NOTIFY Tests
  - SUBSCRIBE: PUBLISH-NOTIFY Tests

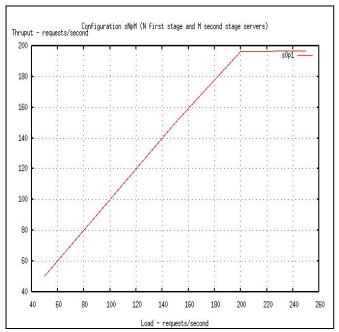
SIMPLEStone Workload Specification

1. Number of presentities and their SIP addresses which the loader uses to generate PUBLISH and handler subscribes to
2. Number of watchers and SIP addresses which the handler uses for sending SUBSCRIBE and server sends NOTIFY to
3. Request rate
   a) Rate of publication (loader sends PUBLISH). This is specified per presentity
   b) Subscribe rate (Total initial SUBSCRIBE count and number of subscriber's per presentity)
   c) Rate of renewing subscription (rate of SUBSCRIBE refresh)
4. Presence body specified in a file
5. Transport protocol type for the test (UDP,TCP,TLS)
6. Timeout interval for receipt of NOTIFY for each PUBLISH message
7. The names of the loader, handler and SUT host addresses and port numbers

Preliminary SIMPLEStone Results and Analysis



success rate vs cpu and memory



throughput vs. load

UDP	TCP or TLS
▪ Low overhead, no state maintenance, Higher throughput ▪ No file descriptor limit ▪ No congestion control ▪ NO TLS – Security. ▪ Fragmentation of UDP packet is disadvantageous because of possibility of loss of fragment, Hence handling larger data sizes (NOTIFY bodies) can be an issue ▪ Client failure detection using ICMP errors, number of retransmissions depends on effectiveness of client failure detection	▪ TCP state maintenance, higher overhead, lower throughput ▪ File descriptor limit ▪ Inbuilt congestion control ▪ Security using TLS ▪ Handles larger data sizes, all fragments will have guaranteed delivery, better for large NOTIFY bodies ▪ Easy failure detection during send call based on no TCP ACK. Effective failure detection to do retransmission control