Functional Specification for a Paging Server

<?xml version="1.0" encoding="utf-8"?>
<html>

Introduction

This document describes the functionality of a sipX Paging Server. It assumes the reader has some basic understanding of SIP and sipX.

The implementation of a Paging Server as described here could be based on the sipXtapi user agent SDK. It is believed that by using the sipXtapi implementation in the SIPfoundry sipx main branch as the basis, a robust and well performing paging server could be built with very reasonable effort. The sipXtapi code in the sipx project main line is well tested and hardened for server side use. The code in the sipXtapi branch offers additional functionality not required by a paging server and is more geared towards user agents that run on Windows.

Interested in helping out with this project? Please say so on the sipX mailing list.

Terminologies

This section defines some of the terms used in this document.

- Comm Server ? sipXecs component that is responsible for routing calls
- Config Server ? sipXecs component that is responsible for configuration and management services
  *Group paging feature code ? a set of digits, typically three digits beginning with *, such as *74, used to indicate a user?s intent to page a group of phones
- Paging Server ? the logical entity that provides group paging capability
- Pager ? the phone (user) which pages a group of phone users
- Pagee ? the phone (user) which is paged
- sipXtapi ? sipXecs libraries that provide a set of telephony APIs for developing telephony applications

Group Paging Functional Overview

sipXecs 3.6 supports one-to-one intercom/paging feature utilizing Polycom phone?s auto-answer feature using SIP alert-info header. However, this feature cannot page more than one phone. It is very desirable to be able to page multiple phone users at the same time. For example, one could page the entire Sales department or Engineering department. This ability is typically called group paging.

With group paging, an admin can set up paging groups, for example, Sales, Engineering, Support, etc. Each paging group contains a list of phone numbers to be paged. For example, one can create a paging group, assign it a group extension (e.g. 265), and attach it a list of phone extensions (e.g. 102 ? 108). To page every phone in the paging group, one would dial group paging feature code followed by the group extension. If the group paging feature code is set to *74, to page paging group with extension 265, one would dial *74265.

When a user (pager) pages a paging group, his call is routed to the paging server and a connection is established between the pager and the paging server. The paging server looks up from its database the list of phones in the paging group, and then establishes individual connections with each phone within the group. After all the connections have been completed, the paging server sends either a tone or an audio prompt to indicate to the pager that he can now start talking. The paging server then transmits the audio stream from the pager to all paged phones.

Paging Server Operation

Paging server should be implemented as a logical and independent entity, most likely a Linux process, which can run on any Linux computer. It can coexist with sipXecs or run on a separate computer.

The paging server shall depend on two configuration files, sipxpage-config and paging-group.xml. sipxpage-config contains the configuration parameters for the paging server. Paging-group.xml is a simple XML file that contains the definition of paging groups. The paging server uses paging-group.xml to construct its internal database used to look up the list of phones in a paging group. Both files will be created and delivered by Config Server. The contents and formats of both files will be defined later.

Paging server shall read sipxpage-config when it starts up and configures itself accordingly. It shall also checks if paging-group.xml is modified each time it receives a paging request. If the file has been modified since it was last checked, the paging server shall update its internal database with the latest paging-group.xml file. Since paging requests are not frequent events, so checking the modification status of paging-group.xml each time a paging request is received is not unreasonable when it comes to efficiency. Besides, it is more likely some modifications have been made since the elapsed time between each group paging is typically long.

Upon receiving an INVITE with Request-URI in the format of sip:<paging group extension>@<paging server IP address>, e.g. sip:265@10.1.1.23, the paging server shall perform the group paging service for the specified paging group (e.g. 265).

Phones to be supported

Phones from the following vendors shall be supported:

- Polycom (www.polycom.com)
- Snom (www.snom.com)
- Grandstream (www.grandstream.com)
- Aastra Telecom (www.aastratelecom.com)

Please note that each phone vendor uses its own mechanism to support the auto-answer feature, so the Paging Server needs to implement different auto-answer triggering mechanism for different phones. For example, Polycom phones use SIP Alert-Info header in the SIP INVITE to trigger auto-answer. Snom phones trigger auto-answer upon receiving a regular SIP INVITE request when they are configured to auto-answer incoming calls. Grandstream phones use SIP Call-Info header (Call-Info: answer-after=0) in the SIP INVITE request to trigger auto-answer. Aastra phones use SIP Alert-Info header (Alert-Info: info=alert-autoanswer) to trigger auto-answer.

For this reason, Paging-group.xml needs to contain phone type information in order for the Paging Server to generate an appropriate SIP INVITE request to trigger auto-answer.

Use Case

There is only one use case - page a paging group, which is described in this section.

Basic Scenario

Actors: pager, pagees

1. A pager dials a paging group extension.
2. The call request reaches the paging server.
3. A call leg is established between the pager and the paging server.
4. The paging server plays the audio prompt ?You are connected to the paging server. Please wait for the paging tone before start paging.?
5. The paging server looks up the paging group membership, and calls each group member simultaneously to establish a one-way audio path from the paging server to each group member.
6. After establishing the call legs with all group members, the paging server sends a soft tone, lasting roughly a second, to the pager to indicate that the paging can start.
7. The pager starts talking.
8. The paging server takes the audio RTP packets received from the pager, and forward them to all the group members.
9. The pager hangs up.
10. The paging server disconnects all the calls with the group members.

6.2 Alternative Scenario: Unknown paging group

When the paging server receives a paging group extension that is not defined in the paging-group.xml, it shall play the audio prompt ?You have entered an unknown paging extension. Please look up the correct extension and try again.?, and then hangs up the call.

6.3 Alternative Scenario: No call leg is established with pagees

When the paging server cannot establish a call leg with any paging group member, it shall play the audio prompt ?Failed to establish the paging call. Please try again later.?, and then hangs up the call with the pager.

6.4 Alternative Scenario: Fail to establish call legs with some of the pagees

If the paging server fails to establish call legs with some of the pagees within the specified paging timeout, the paging server shall stop attempting to establish call legs with these pagees, and proceed to send the soft tone to the pager to start paging so that paging can proceed with at least some of the group members.

6.5 Alternative Scenario: abnormal call disconnection with the pager

If the call leg between the pager and the paging server is abnormally disconnected, as opposed to the normal termination via SIP BYE request, the paging server shall terminate immediately all the call legs established with the group members.

6.6 Alternative Scenario: all call legs with the pagees disconnected

If all the call legs between the paging server and the pagees get disconnected while the paging is in progress, the paging server shall play the audio prompt ?Paging errors occurred. Please try again later.?, and then hangs up the call with the pager.

6.7 Alternative Scenario: some call legs with the pagees disconnected

If some of the call legs between the paging server and the pagees get disconnected for some reason while the paging is in progress, the paging shall continue with the remaining pagees.

6.8 Alternative Scenario: the paging group dialed has a paging in progress

If the paging group dialed by the pager has a paging in progress, the paging server shall play the audio prompt ?An active paging session is in progress. Please try again later.?, and then hangs up the call.

Note: It is possible that a pagee belongs to multiple paging groups, so the pagee can be paged by multiple paging sessions at the same time. This can create error scenarios that are covered by the scenarios described so far, so there is no need to have a separate scenario for this situation. However, test cases shall include this scenario for testing purpose.

Non-Functional Requirements

- Sipxpage-config shall contain at least the following configurable parameters:
    - Port #: the TCP port number that the Paging Server will listen for SIP requests. The default shall be 5060.
    - Paging timeout: the timeout value (in seconds) within which connections with all paging group members shall be established. The timer starts when the connection between the pager and the Paging Server is established. The default value shall be 30 seconds.
- The codec used for paging shall be G.711. Since RTP packets need to be replicated without performing any audio processing, all audio streams involved in the paging need to use the same codec. G.711 is the lowest common denominator because virtually all VoIP endpoints support G.711, and hence the choice.
- Changes made to paging-group.xml shall be effective in the next paging operation, without having to restart the Paging Server. The possible changes include, for example, adding / deleting additional paging groups, modifying paging group memberships, etc.
- The Paging Server shall interoperate with sipXecs.
- The Paging Server shall be developed with C++ programming language.
- The Paging Server shall run on Linux operating system (both FC5 and Red Hat RHEL4).

8 Non Requirements

- It is not required to page a non-local phone, which is only reachable via either a PSTN gateway or a SIP border element, such as a session boarder controller.

9 Scalability and Performance

The scalability has two dimensions: scale within a server and scale with multiple servers. The performance, measured by the number of simultaneous pagees being paged, shall scale up and down roughly linearly with the power of the server. Performance shall also scale up linearly with multiple servers.

For the purpose of establishing a performance baseline, the following server or equivalent shall be used: Intel Pentium processor with Hyperthreading, running at 2.8 GHz, with 1GB of memory. With the above configuration, the paging server shall be able to page 50 pagees simultaneously at a minimum. It is anticipated that the performance will be much higher than the minimum requirement specified above; however, it is a good initial requirement. 50 pagees can be a combination of multiple paging sessions, e.g. one paging session with 50 pagees or five paging sessions, each with 10 pagees.

10 Reliability

The paging server shall achieve 99.99% availability.

11 Testability

This paging server shall be testable by itself. The paging server shall be able to demonstrate that it meets all functional and non-functional requirements.

12 Maintainability

A design document shall be delivered so that other developers can easily understand the code and be able to enhance, maintain, and support it. The design document shall describe the major components of the paging server, their interfaces, the static structural views of the all major components, and the dynamic views of the components (e.g. collaboration amongst the components, state diagrams, etc.), preferably in UML (Unified Modeling Language).

The code shall follow Pingtel?s coding standard to ensure good coding practice and ease of maintenance.
The code shall be buildable with Pingtel and SIPfoundry?s build system, using autotools.
It is highly desirable that a set of unit tests be delivered with the Paging Server written in CPPUnit to unit test the major components of the system by itself.

All the software test fixtures developed for the Paging Server project shall be delivered so that the test fixtures can be used for future regression tests

13 System Integration with sipXecs

The paging server can run standalone. However, it is more usable when it is integrated with sipXecs.
The integration involves Config Server. Config Server shall present GUIs to enable an admin to configure group paging feature code, paging groups, and the paging server, and then delivers sipxpage-config and paging-group.xml files to the paging server. In addition, Config Server shall configure the appropriate Comm Server service configuration files so that group paging calls can get routed to the paging server, and that SIP request URI is transformed into a format expected by the paging server. The SIP Request URI received by Comm Server is in the format of sip:<paging group feature code><paging group extension>@example.com, and this shall be transformed into sip:<paging group extension>@<paging server?s IP address>.

Config Server shall also configure the phones defined in the paging groups to enable their auto-answer feature.

</html>