Understand Load Balancing Algorithm in CVP SIP Server Group

Contents

Introduction

This document describes how the load balancing algorithm works in Cisco Unified Customer Voice Portal (CVP) Session Initiation Protocol (SIP) server groups

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- CVP Server
- CVP Operations Console (OAMP)

Components Used

The information in this document is based on these software versions:

- CVP Server 9.0 and above
- CVP OAMP 9.0 and above

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

SIP Server Groups

SIP Server group is a dynamic routing feature that enables the originating endpoint to know the status of the destination address before attempting to send a SIP INVITE. Whether the destination is unreachable over the network, or is out of service at the application layer, the originating SIP user agent has knowledge of the status through a heartbeat mechanism.

The Server group features adds a heartbeat mechanism with endpoints for SIP. This feature allows faster failover on call control by eliminating delays due to failed endpoints.

Note: Server groups are not automatically created. Server groups are not created by the upgrade to Release 9.0(1). You must explicitly configure Server groups for their deployment, and turn the feature on after upgrading, in order to take advantage of the feature.

Note:Upgrade for customers who already use Local SRV. Customers who already have an srv.xml file configured with local SRV must run the import command mentioned below in order to put their configuration into the Unified CVP Operations Console Server database. Do this before saving and deploying any new server groups to avoid overwriting your previous configuration.

The Unified CVP SIP Subsystem builds on the local SRV configuration XML available with Release 9.0(1).

A Server group consists of one or more destination addresses (endpoints),which is identified by a Server group domain name. This domain name is also known as the SRV cluster domain name, or FQDN. The SRV mechanism is used, but the DNS server resolution of the record is not performed. Server groups remains the same as local SRV implementation (srv.xml), but the Server groups feature adds the extra heartbeat mechanism on top of it, as an option.

SIP Server Groups Load Balancing

For the load balancing algorithm among targets configured in the SIP Server group, the stack is following the selection algorithm specified in RFC 2782 :

To select a target to be contacted next, arrange all SRV RRs (that have not been ordered yet) in any order, except that all those with weight 0 are placed at the beginning of the list. Compute the sum of the weights of those RRs, and with each RR associate the running sum in the selected order. Then choose a uniform random number between 0 and the sum computed (inclusive), and select the RR whose running sum value is the first in the selected order which is greater than or equal to the random number selected. The target host specified in the selected SRV RR is the next one to be contacted by the client. Remove this SRV RR from the set of the unordered SRV RRs and apply the described algorithm to the unordered SRV RRs to select the next target host. Continue the ordering process until there are no unordered SRV RRs. This process is repeated for each Priority.

e.g

When we have 3 targets, A,B,C in the SIP server group having priority 1 and weight of 33 each,

Then algorithm works like this:

- Compute the sum of 3 weights that is 99
- create 0-33 ,33-66,66-99 three slots
- Take a random number from 0-99
- If it is 0< rn<=33 target number1, 33<rn>=66 target 2 and 66<rn>=99 target 3

This is how load is balanced, the load will be balanced between 3 targets.

Note: if target one is down, the load will NOT be balanced between target 2 and 3, but target 1's portion of load goes to target 2

Then algorithm works in this way:

- Compute the sum of 3 weights that is 99
- create 0-66,66-99 two slots
- Take a random number from 0-99
- If it is 0< rn<=66 target number 2, 66<rn>=99 target number 3

So that the target 2 will get more loads compared to target 3.