Setting up an RTCP-XR collector for Polycom Productivity Suite

Polycom RTCP-XR Collector

Overview

Polycom phones are capable of sending call quality statistics if they have the Productivity Suite Licence. The phone uses the sip PUBLISH method to send statistics to a collector server. Below is a guide to setting up a collector using openSIPS.

Requirements

- Opensips Opensips
  - Perl Module opensips-perl
- Perl
  - Perl Opensips module
  - Perl DBI module
- Database (postgres)
- or Database (mysql)

Configurations

opensips.cfg

```
#
# $Id: opensips.cfg 5503 2009-03-22 16:22:32Z bogdan_iancu $
#
# OpenSIPS basic configuration script
#     by Anca Vamanu <anca@voice-system.ro>
#
# Please refer to the Core CookBook at:
#      http://www.opensips.org/index.php?n=Resources.DocsCookbooks
# for a explanation of possible statements, functions and parameters.
#


####### Global Parameters #########

debug=2
#log_stderror=yes
#log_facility=LOG_LOCAL0
log_facility=LOG_LOCAL7

fork=yes
children=4

/* uncomment the following lines to enable debugging */
#debug=6
#fork=no
#log_stderror=yes

/* uncomment the next line to disable TCP (default on) */
#disable_tcp=yes

/* uncomment the next line to enable the auto temporary blacklisting of
   not available destinations (default disabled) */
#disable_dns_blacklist=no

/* uncomment the next line to enable IPv6 lookup after IPv4 dns
   lookup failures (default disabled) */
#dns_try_ipv6=yes

/* uncomment the next line to disable the auto discovery of local aliases
   based on revers DNS on IPs (default on) */
#auto_aliases=no
```

```
/* uncomment the following lines to enable TLS support  (default off) */
#disable_tls = no
#listen = tls:your_IP:5061
#tls_verify_server = 1
#tls_verify_client = 1
#tls_require_client_certificate = 0
#tls_method = TLSv1
#tls_certificate = "/usr/local/etc/opensips/tls/user/user-cert.pem"
#tls_private_key = "/usr/local/etc/opensips/tls/user/user-privkey.pem"
#tls_ca_list = "/usr/local/etc/opensips/tls/user/user-calist.pem"


port=5060

####### Modules Section ########

#set module path
mpath="/usr/local/lib/opensips/modules/"

/* uncomment next line for MySQL DB support */
#loadmodule "db_mysql.so"
loadmodule "signaling.so"
loadmodule "sipmsgops.so"
loadmodule "sl.so"
loadmodule "tm.so"
#loadmodule "rr.so"
loadmodule "maxfwd.so"
#loadmodule "usrloc.so"
#loadmodule "registrar.so"
#loadmodule "textops.so"
#loadmodule "mi_fifo.so"
# This module is part of core (opensips 1.8.2)
# loadmodule "xlog.so"
loadmodule "perl.so"

#----------------- setting module-specific parameters ---------------

#----- mi_fifo params -----
#modparam("mi_fifo", "fifo_name", "/tmp/opensips_fifo")

# ----- rr params -----
# add value to ;lr param to cope with most of the UAs
# deprecated (opensips 1.8.2)
# modparam("rr", "enable_full_lr", 1)
# do not append from tag to the RR (no need for this script)
# modparam("rr", "append_fromtag", 0)


# ----- registrar params -----
# deprecated (opensips 1.8.2)
# modparam("registrar", "method_filtering", 1)



modparam("perl", "filename", "/usr/local/etc/opensips/messagedump.pl")
modparam("perl", "modpath", "/usr/local/lib/opensips/perl/")


####### Routing Logic ########


#main request routing logic

route{

        if (!mf_process_maxfwd_header("10")) {
                sl_send_reply("483","Too Many Hops");
                exit;
        }
```

```
        if (is_method("PUBLISH"))
        {
        perl_exec("messagedump");
                # After recording values in the DB, respond with 200 to the endpoint so the transaction does
not timeout
                t_reply("200","PUBLISH RECORDED");
                exit;
        }else {
                 sl_reply_error();
        }
}
```

messagedunmp.pl

```
use OpenSIPS qw ( log );
use OpenSIPS::Constants;
use DBI;

$my_database=voip;
$my_user=voip;
$my_password=password;
$my_host="127.0.0.1";
$my_platform="Pg";
# For MYSQL database use
# $my_platform="mysql";

sub messagedump {
    my $m = shift;
    my $name;
    my $number;
    my $localUA;
    my $remoteUA;
    my $callID;
    my $packetLoss;
    my $moslq;
    my $moscq;
    my $rtd;
    my $timeStart;
    my $timeStop;

#        open F,">>/tmp/opensips-perl-messagedump";
    my $fh = $m->getHeader("From");
    if($fh =~ /"(.*)"\s+<sip:(\d+)@.*$/)
    {
        $name=$1;
        $number=$2;
    }
#        print F "NAME:$name\nNUMBER:$number\n";
    my $mb = $m->getBody();
#        print F "$mb\n";
    if($mb =~ /.*CallID:(.+).*/)
    {
        $callID = $1;
        $callID =~ s/\@.*//g; #remove the @symbol and everthing after
    }
    if($mb =~ /.*LocalAddr:IP=(.+)\s+PORT.*/)
    {
        $localUA = $1;
    }
    if($mb =~ /.*RemoteAddr:IP=(.+)\s+PORT.*/)
    {
        $remoteUA = $1;
    }
    if($mb =~ /.*RTD=(.+)\s+ESD.*/)
    {
        $rtd = $1;
    }
    if($mb =~ /.*PacketLoss:NLR=(\d+\.\d+).*/)
    {
    $packetLoss=$1;
    }
    if($mb =~ /.*MOSLQ=(\d.\d)\s+MOSCQ=(\d.\d).*/)
    {
        $moslq = $1;
        $moscq = $2;
    }
    if($mb =~ /.*START=(.+)\s+STOP=(.+).*/)
    {
        $timeStart = $1;
        $timeStop = $2;
    }
```

database_structure_postgresql

```
CREATE TABLE qos (
    callid character varying(50),
    name character varying(50),
    number character varying(12),
    timestart timestamp without time zone,
    timestop timestamp without time zone,
    localua character varying(20),
    remoteua character varying(20),
    rtd integer,
    moslq character varying(10),
    moscq character varying(10),
    packetloss character varying(10)
);
```

database_structure_mysql

```
CREATE TABLE qos (
  callid varchar(50),
  name varchar(50),
  number varchar(12),
  timestart datetime,
  timestop datetime,
  localua varchar(20),
  remoteua varchar(20),
  rtd int(4),
  moslq varchar(10),
  moscq varchar(10),
  packetloss varchar(10)
);
```