mod_cidlookup

About

mod_cidlookup is a Caller*ID lookup API. This FreeSWITCH module allows one to:

- lookup a number to name mapping in a local database
- lookup a number to name mapping from a URL
- cache the results of the URL lookup in memcache (if the cache module is installed)
- if all else fails, lookup city and state information

Click here to expand ToC

Usage

```
<action application="cidlookup" data="$1"/>
```

```
<action application="set"
data="effective_caller_id_name=${cidlookup(${caller_id_number})}"/>
```

caller_id_number and caller_id_name are set by FreeSWITCH core. Users should set the channel variable effective_caller_id_name.

The API can be called via ESL or used directly from fs_cli.

```
cidlookup status|number [skipurl] [skipcitystate]
```

Requirements

mod_cidlookup only requires the modules that support your selected lookup methods. If any of the supporting modules are missing (mod_curl, local database, mod_memcache) that part of the functionality will be disabled. memcache support requires the mod_memcache module to be loaded.

Installing

To use mod_cidlookup:

Tell FreeSWITCH to compile in this module by editing modules.conf in /usr/src/freeswitch/trunk and uncomment the following line:

modules.conf

```
applications/mod_cidlookup
```

Now recompile FreeSWITCH:

linux command line

```
make ; make install
```

Now tell FreeSWITCH to load cidlookup and supporting modules by adding to or uncommenting their entries in modules.conf.xml in $FS_ROOT/conf/autoload_configs:

modules.conf.xml

```
<load module="mod_memcache"/>
<load module="mod_cidlookup"/>
```

Finally, edit the default config in the autoload_configs directory to hold your cidlookup configuration.

Now start FreeSWITCH and test.

CallerID / CNAM Database Resources

voip-info.org lists number-to-name database providers.

Some CNAM providers offer a way to perform the CNAM DIP via an e164 method or a SIP Subscribe method.

For information on how to perform the CNAM dip via e164, or ENUM see this page on Enum / e164 VoIP CNAM Dips.

There is no obvious way to perform the CNAM DIP via the SIP Subscribe method, although that does not mean that it isn't possible.

> **Caution**
> Sometimes the retrieval of information from certain online providers takes an appreciable time to occur. If you perform the CID lookup before you perform the <action application="answer"/> in your dialplan, the originating carrier might time out and drop the call or behave in unexpected ways.
>
> If you wish to perform the CID Lookup before answering the call in order to take action not only on numbers, but also on names, use <action application="pre_answer"/> before performing the CID lookup. This solves the problem of the carrier giving up on the call, amongst other problems.

Configuration

Parameters:

url - Uniform Resource Locator of the service that takes a properly formatted telephone number and returns as its only data the name. There are several online providers that do this or you can implement this yourself. The sample config uses the service from OpenCNAM.

whitepages-apikey - WhitePages.com offers a free reverse lookup API for low volume use. If you intend to run a high volume of queries, they offer a commercial service. Refer to developer.whitepages.com for an API key.

cache - Set to true or false. Use memcache to cache the number-to-name lookup

cache-expire - Number of seconds to keep the lookup in the cache.

odbc-dsn - database:user:password The ODBC database connection string

sql - The SQL query to execute. The only parameter supported is ${caller_id_number} which is translated to the number passed to cidlookup. The query should return a single value which is the name.

citystate-sql - The SQL query to execute when falling back to city, state query. The only parameter supplied is ${caller_id_number} which is the number passed to cidlookup. The query should return a single value which is the name of the city, state. This is ONLY used for NANPA numbers. They must be 11 digits and start with 1.

Caching

If memcache is enabled, the cache will be checked first before performing other lookups. However, only the results of successful URL lookups are cached. Lookups from the "City, State" fallback are not cached.

Sample configuration file

Sample cidlookup configuration Expand source

```
<configuration name="cidlookup.conf" description="cidlookup Configuration">
  <settings>
    <param name="url"
value="https://api.opencnam.com/v2/phone/${caller_id_number}?format=pbx&account_sid=AC
COUNTSID&auth_token=AUTHTOKEN"/>
    <param name="cache" value="false"/>

    <param name="odbc-dsn" value="phone:phone:phone"/>
    <param name="sql" value="
SELECT name||' ('||type||')' AS name
  FROM phonebook p JOIN numbers n ON p.id = n.phonebook_id
  WHERE n.number='${caller_id_number}'
  LIMIT 1
    "/>
    <param name="citystate-sql" value="
    SELECT ratecenter||' '||state as name
      FROM npa_nxx_company_ocn
      WHERE npa = ${caller_id_number:1:3} AND nxx = ${caller_id_number:4:3}
      LIMIT 1
      "/>
  </settings>
</configuration>
```

Trivial sample configuration Expand source

```
 <configuration name="cidlookup.conf" description="cidlookup Configuration">
  <settings>
    <param name="url" value="https://api.opencnam.com/v2/phone/+${caller_id_number}"/>
    <param name="cache" value="false"/>
  </settings>
</configuration>
```

Sample SQL Schema - PostgresSQL (not sqlite)

Create sample SQL schema

```
create table phonebook(id serial PRIMARY KEY, name text '', notes text default 'h');
create table p_numbers(id serial PRIMARY KEY, phonebook_id integer, number text
default '', type text default 'h'); create unique index number on p_numbers (number);
alter table p_numbers add constraint numbers_phonebook_id_fkey FOREIGN KEY
(phonebook_id) REFERENCES phonebook(id) ON DELETE CASCADE;
```

Sample schema for PostgreSQL Expand source

```
phone=> \d phonebook
                          Table "fs.phonebook"
 Column |  Type   |                       Modifiers
--------+---------+-------------------------------------------------------
 id     | integer | not null default nextval('phonebook_id_seq'::regclass)
 name   | text    | not null
 notes  | text    | not null default ''::text
Indexes:
    "phonebook_pkey" PRIMARY KEY, btree (id)

phone=> \d p_numbers
                             Table "fs.p_numbers"
    Column    |  Type   |                        Modifiers
--------------+---------+-----------------------------------------------------------
 id           | integer | not null default nextval('p_numbers_id_seq'::regclass)
 phonebook_id | integer |
 number       | text    | not null default ''::text
 type         | text    | not null default 'h'::text
Indexes:
    "p_numbers_pkey" PRIMARY KEY, btree (id)
    "i_numbers" btree (number)
Foreign-key constraints:
    "numbers_phonebook_id_fkey" FOREIGN KEY (phonebook_id) REFERENCES phonebook(id) ON
DELETE CASCADE
```

Testing

To verify that mod_cidlookup is functioning correctly and using the options from your config file, type this in fs_cli:

fs_cli

```
cidlookup status

+OK
 url: https://api.opencnam.com/v2/phone/${caller_id_number}?format=pbx
 cache: false
 cache-expire: 86400
 odbc-dsn: phone
 sql:  SELECT name||' ('||type||')' AS name FROM phonebook p JOIN numbers n ON p.id =
n.phonebook_id WHERE n.number='${caller_id_number}' LIMIT 1
 ODBC Compiled: true
```

To test a sample lookup, invoke cidlookup with a properly formatted telephone number:

fs_cli

```
cidlookup 17035911635

HAYNES,FRANK
```

If you have DEBUG logging turned on, you'll see it trying each step and, if you're using the cache module, whether the data is stored in memcache or not.

This verifies that the module is running and it can load its configuration information.

Examples

The cidlookup application sets the channel variable **caller_id_name** if the lookup succeeds. The easiest way to use it is to put the following block towards the top of your dialplan/public.xml

Dialplan example Expand source

```
<extension name="cid_number_cleanup" continue="true">
      <condition field="caller_id_number" expression="^(?:\+)(\d+)$">
        <action application="set" data="effective_caller_id_number=$1" inline="true"/>
      </condition>
    </extension>

    <extension name="cid_name_cleanup" continue="true">
      <condition field="caller_id_name" expression="^(?:\+)(\d+)$">
        <action application="set" data="effective_caller_id_name=$1" inline="true"/>
      </condition>
    </extension>

    <extension name="cid_lookup-country_code_1" continue="true">
      <condition field="${module_exists(mod_cidlookup)}" expression="true"/>
      <condition field="caller_id_name" expression="^(?:\+)(\d+)$|^$"/>
      <condition field="caller_id_number"
expression="^(?:\+1|1)?([2-9]\d\d[2-9]\d{6})$">
        <action application="cidlookup" data="$1"/>
      </condition>
    </extension>
```

Easier Dialplan

The documentation isn't quite clear that what we're really trying to set here is effective_caller_id_name (vs caller_id_name). For an inbound route it would look something like this:

Expand source

```
<extension name="Demo Inbound" >
   <condition field="context" expression="public"/>
   <condition field="destination_number" expression="12024561000">
       <action application="set" data="call_direction=inbound"/>
       <action application="answer"/>
       <action application="sleep" data="1000"/>
       <action application="set"
data="caller_id_name=${cidlookup(${caller_id_number})}"/>
       <action application="set" data="effective_caller_id_name=${caller_id_name}"/>
       <action application="ivr" data="Your_IVR"/>
   </condition>
</extension>
```

> This will cause a lookup if the call comes from "Anonymous" too, but perform an SQL lookup of "blank" (it strips alpha characters it seems) which will return unrelated results. This example should be not be used in production systems.

A failed lookup will result in a value of "-ERR" -- check for that value prior to setting effective_caller_id_name.

Fallback to "City, State"

If the name lookup fails or if you wish only to provide the general location of the number based on telco records, an additional database can be utilized. The table needs to be loaded into the same database as your cidlookup directory database. One source is the CSV file available at: npa nxx file

One way to load this data into PostgreSQL is with the following DDL

PostgreSQL DDL Expand source

```
CREATE TABLE npa_nxx_company_ocn (
    npa smallint NOT NULL,
    nxx smallint NOT NULL,
    company_type text,
    ocn text,
    company_name text,
    lata integer,
    ratecenter text,
    state text
);
CREATE UNIQUE INDEX npanxx_idx ON npa_nxx_company_ocn USING btree (npa, nxx);
```

You can then load the data using psql with

psql Expand source

```
phone=> \copy npa_nxx_company_ocn from 'npa-nxx-companytype-ocn.csv' with csv
phone=> select count(*) from npa_nxx_company_ocn ;
 count
--------
 163900
(1 row)
```

The default config will work with the above data structure.

Setting Outbound Name

Some phones, including Polycom and Snom, support setting the callee's name once the call is complete. This allows the caller to see the name of the person being called as well as the number. Using cidlookup the name could come out of your corporate directory and if that doesn't work will be queried against a CNAM lookup service. Assuming you've already normalized your number to E.164 (without the leading +), just add the following to your dialplan prior to the bridge:

Callee Dialplan XML

```
<action application="export" data="callee_id_name=${cidlookup($1)}" />
```

Using OpenCNAM

OpenCNAM (https://www.opencnam.com/) is a large CNAM provider that works well with FreeSWITCH. OpenCNAM provides two tiers of CNAM lookups:

- A Hobbyist tier, which allows you to lookup a maximum of 60 cached CNAM queries per hour, completely free
- A Professional tier, which allows unlimited real-time CNAM queries, for a fee ($0.004 per successful lookup at the time of this writing)

To get started with OpenCNAM's Hobbyist tier, you don't need to do anything. OpenCNAM's Hobbyist tier doesn't require any registration or accounts and is enabled by default in the *cidlookup.conf.xml* configuration file.

If you need to query more than 60 requests per hour, or prefer to have more accurate Caller ID information (with real-time CNAM queries), you can create an OpenCNAM account on their website, deposit funds into your account, then update your *cidlookup.conf.xml* configuration file appropriately. Once you've created an OpenCNAM account, you'll notice that on your dashboard page you have two account tokens at the top of your page: an *Account SID* and *Auth Token*. To make FreeSWITCH use OpenCNAM's Professional tier and thereby perform only real-time CNAM queries, modify your *cidlookup.conf.xml* file's URL attribute thus:

OpenCNAM Professional config Expand source

```
<configuration name="cidlookup.conf" description="cidlookup Configuration">
 <settings>
  <param name="url"
value="https://api.opencnam.com/v2/phone/${caller_id_number}?format=pbx&account_sid=YO
UR_ACCOUNT_SID&auth_token=YOUR_AUTH_TOKEN"/>
  <param name="cache" value="false"/>
 </settings>
</configuration>
```

Using PHP

If you run also a webserver and have PHP on it you can query web directories for the caller id. The following example uses http://tel.search.ch to look up Swiss phone numbers. They offer an API with XML output from which the name will be parsed. To use their API you have to get the key. In this case you can get the key from here: http://admin.tel.search.ch/api/getkey

Prerequisite is the php5-curl library that must be installed on your server. Adjust this as required.

PHP web server config Expand source

```
<?php

/*****************************************************
CONFIG
*****************************************************/

$apiKey = '******';



/*****************************************************
BELOW BE DRAGONS
*****************************************************/

$cid = $_GET['cid'];
$check=strlen($cid);

// Do a few checks for the input... e.g. only lookup swiss numbers
if($check == 10) {
        // 10 digits as required
} elseif($check == 11) {
        $sub1 = substr($cid, 0 ,2);
        $sub2 = substr($cid, 2);
        // check if swiss country code is added
        if($sub1 == '41') {
                $cid = '0' . $sub2;
        }
} elseif ($check == 9) {
        // check if leading 0 is missing
        $cid = '0' . $cid;
}

// Run query
$q = 'http://tel.search.ch/api/?key=' . $apiKey . '&pos=1&maxnum=1&was=' . $cid;


$ch=curl_init();
curl_setopt($ch,        CURLOPT_URL,            $q);
curl_setopt($ch,        CURLOPT_HEADER,         0);
curl_setopt ($ch,       CURLOPT_RETURNTRANSFER, 1);

$result = curl_exec($ch);
curl_close($ch);

// As http://tel.search.ch provides an xml result, you have to filter out the number
$result = explode('<tel:name>', $result);
```

```
$result = explode('</tel:name>', $result[1]);
$result = $result[0];

if($result == '') {
        // If there is no result, then assign the cid as result. If you do more
checks, then you may want to comment the following line out
         $result = $cid;
} else {
         $result = $result;
}

$result = utf8_encode($result);
```

```
echo $result;

?>
```

Save that script somewhere on your server and enhance the freeswitch/conf/autoload_configs/cidlookup.conf.xml with this

cidlookup.conf.xml

```
<param name="url" value="http://<server
address>/cid/index.php?cid=${caller_id_number}"/>
```

The above example assumes that you named the PHP script index.php and that it resides in the /cid/ folder from the server.

Background CID Lookups

A CID lookup can introduce a delay of several seconds, depending on which service you use. If you answer with an IVR, the CID lookup can happen in the background, while the IVR is playing the greeting.

To do this, first create the following Lua script:

cid_bg.lua Expand source

```
--[[
  Lookup the CID name, and set the effective_caller_id_name variable.
  This script will run in the background, so we need to set the variable via the UUID.
]]
api = freeswitch.API();
uuid = argv[1];
if not uuid or uuid == "" then return end;
number = api:executeString("uuid_getvar " .. uuid .. " caller_id_number");
name = api:executeString("cidlookup " .. number);
api:executeString("uuid_setvar " .. uuid .. " effective_caller_id_name " .. name);
```

Next, call the script from the "public" dialplan as follows:

Dialplan context public Expand source

```
<extension name="incoming main">
    <condition field="destination_number" expression="^(12025551212|18035551212)$"
require-nested="false">
      <condition field="${cond(${caller_id_name} == ${caller_id_number} ?true:false)"
expression="^true$">
        <action application="set" data="api_result=${luarun cid_bg.lua ${uuid}}"/>
      </condition>
      <action application="answer"/>
      <action application="sleep" data="1000"/>
      <action application="ivr" data="main_ivr"/>
      <action application="transfer" data="operator_vmail XML default"/>
    </condition>
  </extension>
```