


# mod\_com\_amd

## About

mod\_com\_amd is a FreeSWITCH™ commercial module used to analyze call progress and identify if it was answered by a human or a machine.



## Introduction

FreeSWITCH™ provides a licensed commercial Answering Machine Detection module. A key technology for autodialers is the ability to detect live human pickup and answering machine. Since there is no indication/hardware signal when a call is answered by an answering machine or voicemail system, autodialer systems have to analyze incoming audio in order to make a prediction. Currently, there is no algorithm that can achieve 100% accuracy.

## Getting Licenses

All FreeSWITCH™ purchases must be done through our billing system. Go to <http://freeswitch.com>, click on the top right menu called Account and then on Register.

### Order New Products

Visit the Order Form to browse the Products & Services we offer. Existing customers can also purchase optional extras and addons here.

[Go to Order Form »](#)

### Manage Your Account

Already registered with us? If so, click the button area from where you can manage your account.

[Secure Client Login »](#)

- Login
- Register**
- Forgot Password?

### Our Latest Tweets

- 8h ago - dont forget BugHunt today at 2PM Central on 888! [conference.freeswitch.org](http://conference.freeswitch.org)
- 9h ago - FreeSWITCH Bug Hunt on 888 @ 2PM CST! [ift.tt/1rxBRUs](http://ift.tt/1rxBRUs) #VoIP #SIP #WebRTC

FOLLOW US ON [twitter](#)

Language:

Copyright © 2014 FreeSWITCH. All Rights Reserved.

## Register Create an account with us . . .

First Name	<input type="text"/>	Address 1	<input type="text"/>
Last Name	<input type="text"/>	Address 2	<input type="text"/>
Company Name	<input type="text"/>	City	<input type="text"/>
Email Address	<input type="text"/>	State/Region	<input type="text" value="Choose One..."/>
Password	<input type="password"/>	Zip Code	<input type="text"/>
Confirm Password	<input type="password"/>	Country	<input type="text" value="United States"/>
Password Strength	<input type="text" value="Enter a Password"/>	Phone Number	<input type="text"/>

### Spam Bot Verification

Please enter the characters you see in the image below into the text box provided. This is required to prevent automated submissions.



[Register](#)

After registering, go to login page at <https://freeswitch.com/clientarea.php> and provide your credentials. Once logged in the system, you'll see a menu on top of the page called Products, click and then click on Order New Products. Direct link is <https://freeswitch.com/cart.php>.

My Products  
**Order New Products**  
View Available Addons

### Account Overview

Number of Products/Services: **3 (3)** - View »  
Number of Domains: **0 (0)** - View »  
Number of Support Tickets: **0** - View »  
Number of Referred Signups: **0** - View »  
Billing Information: **Use Default (Set Per Order)**

### Open Support Tickets **0**

Open New Ticket

Date	Department	Subject	Status	Last Updated
There are currently no open support tickets				

### Due Invoices **0**

<input type="checkbox"/>	Invoice #	Invoice Date	Due Date	Total	Balance	Status
There are currently no unpaid invoices						

On the cart page, scroll down your screen and you'll see AMD product description. Click on the Order Now button. Direct link is <https://freeswitch.com/cart.php?a=confproduct&i=1>

### Browse Products & Services

[FreeSWITCH Modules](#) | [Consulting Services](#) | [Product Addons](#) | [View Cart](#)

#### G.729 License Reset

Administration fee to reset your G.729 licenses once you've reached your max activations or moving to new hardware.

**\$20.00 USD One Time**

[Order Now](#)

#### G.729A

FreeSWITCH provides a G.729A licensed commercial module for \$10 per channel, that's one encoder and one decoder.

You can buy licenses separately and combine them later when they are for the same machine. If your machine stops working, contact us at [consulting@freeswitch.org](mailto:consulting@freeswitch.org) to get your licenses reset for the replacement machine.

1 single channel of G.729A License (x86\_64 & i386) All sales final.  
DOWNLOADS ARE AT <http://files.freeswitch.org/g729>

**NOTICE:** If you need to order channels for different servers ie. 30 for one and 20 for another please process them as separate orders. The licenses can NOT be split up after the license code is issued and once activated can NOT be moved to a new system.

**They are locked to the system you activate them on. Currently doesn't work on OpenVZ**

Starting from  
**\$20.00 USD One Time**

[Order Now](#)

#### AMD (BETA)

FreeSWITCH provides a licensed commercial Answering Machine Detection module for \$50 per channel.

A key technology for autodialers is the ability to detect live human pickup and answering machine. Since there is no indication/hardware signal when a call is answered by an answering machine or voicemail system, autodialer systems have to analyze incoming audio in order to make a prediction. Currently, there is no algorithm that can achieve 100% accuracy.

1 single channel of AMD License (x86\_64 & i386) All sales final.  
DOWNLOADS ARE AT <http://files.freeswitch.org/amd>

**NOTICE:** If you need to order channels for different servers ie. 30 for one and 20 for another please process them as separate orders. The licenses can NOT be split up after the license code is issued and once activated can NOT be moved to a new system.

**They are locked to the system you activate them on. Currently doesn't work on OpenVZ**

Starting from  
**\$100.00 USD One Time**

[Order Now](#)



On the next screen you need to type how many licenses you need. A license is needed for each created channel that you need to activate the detection. If you have 5 simultaneous calls and want to detect who's answering you'll need 5 ports/channels. Click on Add to Cart after selecting the number of licenses.

## Product Configuration

The product/service you have chosen has the following configuration options for you to choose from.

### Product/Service

#### FreeSWITCH Modules - AMD (BETA)

FreeSWITCH provides a licensed commercial Answering Machine Detection module for \$50 per channel. A key technology for autodialers is the ability to detect live human pickup and answering machine. Since there is no indication/hardware signal when a call is answered by an answering machine or voicemail system, autodialer systems have to analyze incoming audio in order to make a prediction. Currently, there is no algorithm that can achieve 100% accuracy. 1 single channel of AMD License (x86\_64 & i386) All sales final. DOWNLOADS ARE AT <http://files.freeswitch.org/amd> **NOTICE:** If you need to order channels for different servers ie. 30 for one and 20 for another please process them as separate orders. The licenses can NOT be split up after the license code is issued and once activated can NOT be moved to a new system. **They are locked to the system you activate them on. Currently doesn't work on OpenVZ**

### Billing Cycle

\$0.00 USD One Time

### Configurable Options

This product/service has some options which you can choose from below to customise your order.

Ports:  x Number of Ports \$50.00 USD

Add to Cart



If you need to activate licenses for different servers you need to buy them separately. For example, if you need 5 licenses to server A and more 10 to server B, add 5 licenses to the cart and then add 10 more again, you'll receive two activation codes, one for each server.

Now you can proceed to payment, and the end of the page there's a Complete Order button. The payment method is PayPal, if you don't have paypal account please proceed to [PayPal's registration page](#) and follow the instructions.

Home Products - Billing - Support - Open Ticket Hello, Italo! -

## Review & Checkout

Description	Price
FreeSWITCH Modules - AMD (BETA) * Ports: 2 x Number of Ports \$50.00 USD <a href="#">[Edit Configuration]</a> <a href="#">[Remove]</a>	\$100.00 USD
Subtotal:	\$100.00 USD
<b>Total Due Today:</b>	<b>\$100.00 USD</b>

### Your Details

**Promotional Code**

**Payment Method**

PayPal

**Notes/Additional Information**

You can enter any additional notes or information you want included with your order here...

After the payment, PayPal will notify FreeSWITCH™ billing system about the new payment and the license(s) will be sent to your email. This process takes a few minutes, if you do not receive the email in a few minutes please contact us at [consulting@freeswitch.org](mailto:consulting@freeswitch.org) or at #freeswitch at irc.freenode.net. For more information on how to contact us please visit [IRC](#) page.

## Installing

### Updated Instructions for FreeSWITCH 1.8

Install the package

```
apt-get install freeswitch-mod-com-amd
```

Uncomment or create in autoload\_configs/modules.conf.xml

```
<load module="mod_com_amd" />
```

Validate license

```
freeswitch-license-validator
```

```
freeswitch product licencing tool
You will require one or more sales codes to activate licences

Enter a sales code, or a blank line to end: 0abd19753bf28fbb88a5d74
Enter a sales code, or a blank line to end:

Sales codes to be activated:
0abd19753bf28fbb88a5d74
OK (Y/N)? y

Success. The file licences.zip contains valid licences.
Unzip this to /etc/freeswitch/
```

Unzip the key

```
unzip licences.zip

Archive:  licences.zip
inflating: 0abd19753bf28fbb88a5d74.conf
```

Copy key to /etc/freeswitch (do not put the zip file in there)

```
cp 0abd19753bf28fbb88a5d74.conf /etc/freeswitch
```

Shutdown freeswitch

```
systemctl stop freeswitch
```

Start license server

```
freeswitch-license-server

Scanning /etc/freeswitch/0abd19753bf28fbb88a5d7.conf for licences
Purchase code 0abd19753bf28fbb88a5d74
10 channels of AMD
```

Start freeswitch again

```
systemctl start freeswitch
```

Check that amd is communicating with license server

```
fs_cli -x amd_info

Success checking AMD/0
AMD license available => 10
AMD license allocated => 0
```

Do some testing in dialplan. Download test wav file [Answering\\_Machine.wav](#).

```

<extension name="amd_test" continue="false">
  <condition field="destination_number" expression="^(amd|263)$"/>
  <condition field="{amd_available()}" expression="true">
  <!-- alternate way of checking channel availability -->
  <!--<condition field="{regex ${amd_info}|available => 0}" expression="false"--->
  <!-- <action application="set" data="media_bug_answer_req=true"/> -->
  <action application="set" data="amd_execute_on_machine=transfer machine_found XML default"/>
  <action application="set" data="amd_execute_on_person=transfer person_found XML default"/>
  <action application="set" data="amd_execute_on_unsure=transfer amd_unsure XML default"/>
  <!-- <action application="set" data="api_on_answer=uuid_displace ${uuid} start ${sounds_dir}
/Answering_Machine.wav 0 mr"/> -->
  <action application="answer"/>
  <action application="voice_start"/>
  <action application="playback" data="silence_stream://100,1000"/>
  <action application="playback" data="tone_stream://(2000,4000,440,480)"/>

  <!-- <action application="waitforresult" data="ivr/ivr-one_moment_please.wav"/> -->
  <!-- <action application="waitforresult"/> -->
  <!-- <action application="sleep" data="200"/> -->
  <!-- <action application="playback" data="tone_stream://%(200,100,500,400,300,50,25);loops=2"/> -->
  <action application="sleep" data="20000"/>
  <!-- <action application="log" data="CRIT AMD Result is => ${amd_status} => ${amd_result}"/> -->
  <action application="hangup"/>
  <anti-action application="playback" data="{sounds_dir}/All_Circuits_Busy.mp3"/>
  <anti-action application="hangup"/>
</condition>
</extension>
<extension name="Found Machine">
  <condition field="destination_number" expression="^(machine_found)$">
  <action application="playback" data="ivr/ivr-welcome_to_freeswitch.wav"/>
  <action application="log" data="CRIT AMD result is => ${amd_status} and AMD Status is =>
${amd_result}"/>
  <action application="voice_stop"/>
  <action application="hangup"/>
</condition>
</extension>
<extension name="Found Person">
  <condition field="destination_number" expression="^(person_found)$">
  <action application="playback" data="misc/if_you_are_this_person.wav"/>
  <action application="log" data="CRIT AMD result is => ${amd_status} and AMD Status is =>
${amd_result}"/>
  <action application="voice_stop"/>
  <action application="hangup"/>
</condition>
</extension>
<extension name="AMD Unsure">
  <condition field="destination_number" expression="^(amd_unsure)$">
  <action application="playback" data="misc/error.wav"/>
  <action application="log" data="CRIT AMD result is => ${amd_status} and AMD Status is =>
${amd_result}"/>
  <action application="voice_stop"/>
  <action application="hangup"/>
</condition>
</extension>

```

Checking for available license





This is the preferred method for checking channel availability

```
<condition field="${amd_available()}" expression="true">
```

or you can see if license are available by checking a regex against zero using the amd\_info api

```
<condition field="${regex ${amd_info}|available => 0}" expression="false">
```

if regex is false, then proceed with amd usage, else if true, do anti actions to notify customer with a custom sound file

```
<anti-action application="playback" data="{sounds_dir}/All_Circuits_Busy.mp3"/>
<anti-action application="hangup"/>
```

Notes on the parenthesis "()":

example for "false" when channels are available for use:

CORRECT USAGE DIALPLAN: without the parens, yields PASS:

```
<condition field="${regex ${amd_info}|available => 0}" expression="false">
```

Log:

```
Dialplan: sofia/internal/1003@192.168.150.3:65062 Regex (PASS) [amd_test] ${expand regex ${amd_info}|available => 0}(false) ==
/false/ break=on-false
```

INCORRECT USAGE: Dialplan with the parens, yields -ERR:

```
<condition field="${regex ${amd_info()}|available => 0}" expression="false">
```

Log:

```
Dialplan: sofia/internal/1003@192.168.150.3:65062 Regex (FAIL) [amd_test] ${regex ${amd_info()}|available => 0}(-ERR) == /false/
break=on-false
```

example for "true" when all channels are currently in use

CORRECT USAGE ESL: with the parens

```
freeswitch> expand regex ${amd_info()}|available => 0
true
```

INCORRECT USAGE ESL: without the parens

```
freeswitch> expand regex ${amd_info}|available => 0
false
```

## Older Instructions for Freeswitch 1.6

We have packaged our commercial modules into a single installer, All products currently offered (mod\_com\_amd, mod\_com\_g729, mod\_com\_g719, mod\_com\_g728) are all installed and take up very little space.

### Downloading and Running Installer

```
cd /tmp
wget http://files.freeswitch.org/amd/fs-latest-installer
chmod +x fs-latest-installer
./fs-latest-installer
```

This will guide you thru some installation steps. We try to detect the most common places your FreeSWITCH™ installation may be located, If we can't detect it then we will prompt for information about where your FreeSWITCH™ is installed. You can optionally you can provide the data as arguments to the fs-latest-installer binary:

### Installer arguments

```
./fs-latest-installer <bin_path> <mods_path> <config_path>
```

Once you accept the EULA, you'll see output similar to this:

### Installing

```
Thank You!  
Running Installer  
Stopping license server  
Installing new license server  
Installing new FreeSWITCH module  
Installing activation utility  
Creating license directory  
Running ldconfig...  
Cleaning up...  
  
Now you can activate your product license(s) by running /usr/local/freeswitch/bin/validator  
  
Thank you  
  
Done.
```

Once complete you'll need to activate your licenses using the validator outlined in the installation completion/thank you message. You'll use the same validator application for any of our commercial products that are available from our online store.

Once you start the validator you'll see output similar to this:

### Licencing tool - Installing new licence.

```
freeswitch product licencing tool  
  
You will require one or more sales codes to activate licences  
Enter a sales code, or a blank line to end: 8c87ca579c46258a5e2ee9f98ca3f93ca4752d26  
Enter a sales code, or a blank line to end:  
  
Sales codes to be activated:  
8c87ca579c46258a5e2ee9f98ca3f93ca4752d26  
  
OK (Y/N)? Y  
Success. The file licences.zip contains valid licences.  
Unzip this to /etc/freeswitch/
```

The resulting licences.zip will be in the current working directory, Unzip and copy the .conf files into /etc/freeswitch:

### Unzip Licenses

```
unzip licences.zip  
Archive:  licences.zip  
  inflating: 8c87ca579c46258a5e2ee9f98ca3f93ca4752d26.conf  
  
cp 8c87ca579c46258a5e2ee9f98ca3f93ca4752d26.conf /etc/freeswitch
```



#### Hardcode Path Here

**/etc/freeswitch** is a **hard coded path** the *freeswitch\_license\_server* will auto-scan up restart... You **MUST** put your <HEX>.conf file in /etc/freeswitch **regardless** of where you've installed FreeSWITCH.

To make the license process aware of your new licenses please run:

#### Reloading licence server

```
pkill -HUP freeswitch_licence_server
```

This will make the license server rescan the directory adding any additional licenses to your list of resources its capable of licensing. We just activated a license for mod\_com\_amd which when loading the module will look like this:

#### Loading mod\_com\_amd

```
2014-05-28 09:51:52.344344 [INFO] mod_enum.c:880 ENUM Reloaded
2014-05-28 09:51:52.344344 [INFO] switch_time.c:1369 Timezone reloaded 530 definitions
2014-05-28 09:51:52.344344 [INFO] mod_com_amd.c:749 AMD license count => 20
2014-05-28 09:51:52.344344 [CONSOLE] switch_loadable_module.c:1466 Successfully Loaded [mod_com_amd]
2014-05-28 09:51:52.344344 [NOTICE] switch_loadable_module.c:269 Adding Application 'waitforresult'
2014-05-28 09:51:52.344344 [NOTICE] switch_loadable_module.c:269 Adding Application 'voice_start'
2014-05-28 09:51:52.344344 [NOTICE] switch_loadable_module.c:269 Adding Application 'voice_stop'
2014-05-28 09:51:52.344344 [NOTICE] switch_loadable_module.c:315 Adding API Function 'amd_count'
2014-05-28 09:51:52.344344 [NOTICE] switch_loadable_module.c:315 Adding API Function 'amd_available'
2014-05-28 09:51:52.344344 [NOTICE] switch_loadable_module.c:315 Adding API Function 'amd_info'
freeswitch@internal> amd_info
AMD license available => 20
AMD license allocated => 0
freeswitch@internal> amd_available
true
freeswitch@internal> amd_count
20
```

As you can see its seeing the licenses that were just installed, you can use the various api calls registered to manage your licenses.

## Default Configuration

The default mod\_com\_amd configuration will be suitable in most cases, but if you need to tweak the configuration go to /usr/local/freeswitch/conf/autoload\_configs and open amd.conf.xml:

## amd.conf.xml

```
<configuration name="amd.conf" description="AMD Configuration">
  <!-- AMD -->
  <settings>
    <!-- silent_threshold: The level of volume to consider talking or not talking, same scale as used in
mod_conference -->
    <param name="silent_threshold" value="256"/>
    <!-- silent_initial: Time in ms for there to be silence after answer in order to result in "silent-
initial" with status of person -->
    <param name="silent_initial" value="4500"/>
    <!-- silent_after_intro: Time in ms after an initial non silent greeting in order to result in
silent-after-intro with status of person -->
    <param name="silent_after_intro" value="1000"/>
    <!-- silent_max_session: Time in ms of total silence before we allow detection to complete -->
    <param name="silent_max_session" value="200"/>
    <!-- noise_max_intro: Time in ms length of initial intro over which in order to result in max-intro
with status of person -->
    <param name="noise_max_intro" value="1250"/>
    <!-- noise_min_length: Time in ms minimum to be considered a word -->
    <param name="noise_min_length" value="120"/>
    <!-- noise_inter_silence: Time in ms of silence to be considered a word break -->
    <param name="noise_inter_silence" value="30"/>
    <!-- noise_max_count: If we have more than this many noise hits (words) result will be "max-count"
with status of machine -->
    <param name="noise_max_count" value="6"/>
    <!-- total_analysis_time: total time in ms that we will try to analyze a call -->
    <param name="total_analysis_time" value="5000"/>
    <!-- debug: set to 1 to get more debug information -->
    <param name="debug" value="1"/>
  </settings>
</configuration>
```

## Overriding Config Values via Dialplan

```
<action application="voice_start" data="silent_threshold=96,
    silent_initial=3500,
    silent_after_intro=1500,
    silent_max_session=200,
    noise_max_intro=2250,
    noise_min_length=175,
    noise_inter_silence=50,
    noise_max_count=10,
    total_analysis_time=5000,
    debug=1"/>
```

## Examples

Let's figure out how can we use the mod\_com\_amd.

### XML Dialplan Example

The first way you can use mod\_com\_amd is from the xml dialplan. When the module is loaded, three application are created:

- voice\_start
- voice\_stop
- waitforresult

Let's build a dialplan that when the call gets answered by a machine one message gets delivered to the voicemail:

## AMD Dialplan XML Example

```
<extension name="amd_example" continue="false">
  <condition field="destination_number" expression="^5551212$">
    <action application="set" data="media_bug_answer_req=true"/>
    <action application="set" data="amd_execute_on_machine=transfer machine_detected XML default"/>
    <action application="voice_start"/>
    <action application="playback" data="/usr/local/freeswitch/sounds/en/us/callie/ivr/8000/ivr-welcome_to_freeswitch.wav"/>
    <action application="playback" data="/usr/local/freeswitch/sounds/en/us/callie/ivr/8000/ivr-welcome_to_freeswitch.wav"/>
    <action application="playback" data="/usr/local/freeswitch/sounds/en/us/callie/ivr/8000/ivr-welcome_to_freeswitch.wav"/>
    <action application="playback" data="/usr/local/freeswitch/sounds/en/us/callie/ivr/8000/ivr-welcome_to_freeswitch.wav"/>
    <action application="info"/>
    <action application="hangup"/>
  </condition>
</extension>

<extension name="Found machine">
  <condition field="destination_number" expression="^(machine_detected)$">
    <action application="wait_for_silence" data="300 30 5 25000"/>
    <action application="sleep" data="8000"/>
    <action application="playback" data="/usr/local/freeswitch/sounds/en/us/callie/ivr/8000/ivr-welcome_to_freeswitch.wav"/>
    <action application="info"/>
    <action application="hangup"/>
  </condition>
</extension>
```

Pay attention to the variable `amd_execute_on_machine` and the application `voice_start`.

When the call gets answered FreeSWITCH™ will route it to extension "amd\_example". This extension will set the variable `amd_execute_on_machine`, which will transfer the call to destination "machine\_detected" at context "default" if the call gets answered by a machine, otherwise the playback actions will get executed (dialplan lines 6, 7, 8 and 9).

If a machine answers this call, it'll be transferred to our next extension, `machine_detected`, which can be useful when you want to leave message in the customer voicemail, in this case you can use the application `wait_for_silence` to wait the end of voicemail initial greeting and then leave your message.

`mod_com_amd` sets a variable called "amd\_status" and his value can be "human" or "machine". You can see the value with the "info" application (dialplan line 20).

In this first example, you can originate a call to 5551212 using:

### Originating call

```
bgapi originate {ignore_early_media=true}sofia/profile/number 5551212
```

This will make a call to `sofia/profile/number` and when answered FreeSWITCH™ will look for extension that matches 5551212 in your dialplan.

## Lua Example

If you're building your routing logic with Lua, you can use the same application and variables used earlier in the xml dialplan. Example:

## dialer.lua

```
local dst_number = argv[1]
-- Connecting to the freeswitch API.
api = freeswitch.API()
use_amd = api:executeString("amd_available")

subscriber_session = freeswitch.Session("{ignore_early_media=true}sofia/gateway/mygw/" .. dst_number)

while (subscriber_session:ready() and not subscriber_session:answered()) do
  -- Waiting for answer.
  freeswitch.msleep(500)
end

if subscriber_session:ready() and subscriber_session:answered() then
  freeswitch.consoleLog("INFO", string.format("Number answered call %s.", dst_number))
  freeswitch.consoleLog("INFO", string.format("AMD Enable on %s.", dst_number))
  if use_amd == "true" then
    subscriber_session:execute("voice_start")
    -- Giving some time to AMD to work on the call.
    subscriber_session:sleep(3000)
    subscriber_session:execute("voice_stop")
    amd_detect = subscriber_session:getVariable("amd_status")
    if amd_detect == "machine" then
      freeswitch.consoleLog("INFO", "amd_status: machine")
      subscriber_session:execute("wait_for_silence", "300 30 5 25000")
      subscriber_session:execute("playback", "/usr/local/freeswitch/sounds/en/us/callie/ivr/8000/ivr-welcome_to_freeswitch.wav")
      subscriber_session:hangup()
      return
    end
    -- Do your actions if human answered. Ex. Transfer to operator/user 100.
    subscriber_session:execute("bridge", "user/100")
  end
else
  freeswitch.consoleLog("INFO", string.format("Cannot call %s", dst_number))
  return
end
```

Call this lua script using:

```
luarun dialer.lua 55552222
```

## Testing Dialplan

XML for self testing the config setting... be sure to download the [Answering\\_Machine.wav](#) file and put in your sounds directory. With this setup, you should be able to tweak settings and get a feel for their values beyond the default.

```

<extension name="amd_test" continue="false">
  <condition field="destination_number" expression="^(amd_test|263)$">
    <action application="set" data="media_bug_answer_req=true"/>
    <action application="set" data="amd_execute_on_machine=transfer machine_found XML default"/>
    <action application="set" data="amd_execute_on_person=transfer person_found XML default"/>
    <action application="set" data="amd_execute_on_unsure=transfer amd_unsure XML default"/>
    <action application="voice_start"/>

    <action application="set" data="api_on_answer=uuid_displace ${uuid} start ${sounds_dir}
/Answering_Machine.wav 0 mr"/>
    <action application="answer"/>

    <action application="waitforresult" data="ivr/ivr-one_moment_please.wav"/>

    <action application="sleep" data="200"/>
    <action application="playback" data="tone_stream://%(200,100,500,400,300,50,25);loops=2"/>
    <action application="sleep" data="200"/>
    <action application="log" data="CRIT AMD Result is => ${amd_status} => ${amd_result}"/>
    <action application="hangup"/>
  </condition>
</extension>

<extension name="Found Machine">
  <condition field="destination_number" expression="^(machine_found)$">
    <action application="playback" data="ivr/ivr-welcome_to_freeswitch.wav"/>
    <action application="log" data="CRIT AMD result Machine Found is => ${amd_status} => ${amd_result}"/>
    <action application="voice_stop"/>
    <action application="hangup"/>
  </condition>
</extension>

<extension name="Found Person">
  <condition field="destination_number" expression="^(person_found)$">
    <action application="playback" data="misc/if_you_are_this_person.wav"/>
    <action application="log" data="CRIT AMD result Machine Found is => ${amd_status} => ${amd_result}"/>
    <action application="voice_stop"/>
    <action application="hangup"/>
  </condition>
</extension>

<extension name="AMD Unsure">
  <condition field="destination_number" expression="^(amd_unsure)$">
    <action application="playback" data="misc/error.wav"/>
    <action application="log" data="CRIT AMD result Machine Found is => ${amd_status} => ${amd_result}"/>
    <action application="voice_stop"/>
    <action application="hangup"/>
  </condition>
</extension>

```



uuid\_displace have options [m]ux and [r]ead, so you can hear the ensuing found extension audio, and also induce the answering machine file onto the read channel so amd can operate on it.

more technical info about mod\_com\_amd

mod\_amd - Answering Machine Detection for FreeSWITCH

mod\_amd supplies three different dialplan apps and several channel variables.

It also fires events during the detection process. They are listed here for reference.

Apps:

voice\_start - starts the answering machine detection on a channel

voice\_stop - stops the answering machine detection on a channel

waitforresult [<file to play while waiting>] - waits for AMD to return a result, and optionally playback a file

Channel variables, results:

amd\_status - Contains whatever was detected:

'person', 'machine', 'unsure'

Basically, anything other than 'machine' should be assumed to be a human

amd\_result - Contains more information about what was detected:

'too-long' - detection process took longer than total\_analysis\_time (amd.conf.xml)

'max-count' - max noisy frames detected, probably a machine speaking

'max-intro' - max noisy frames detected during intro, probably a machine speaking

'silent-initial' - nothing heard on line, assume human

'silent-after-intro' - short burst of noisy frames followed by silence - probably human

Channel variables, control:

execute\_on\_machine\_app - application to execute if machine detected

execute\_on\_machine\_arg - argument to application to execute if machine detected

Events:

CUSTOM::AMD::EVENT - Action: Start Talking

CUSTOM::AMD::EVENT - Action: Stop Talking

SWITCH\_MEDIA\_BUG\_ADD - Fired at start of detection process

SWITCH\_MEDIA\_BUG\_REMOVE - Fired at stop of detection process

Look for variable\_amd\_status header

USAGE

mod\_amd can be used three ways:

via event socket, where an event-based script watches for AMD events and acts accordingly

via the supplied JavaScript file

via the dialplan by setting "execute\_on\_machine\_app" and "execute\_on\_machine\_arg"

See default.xml for simple examples