

# IVR Server

Following is an attempt to provide as much info as possible about IVR application. This is not a final doc and may need to be updated from time to time.

## Introduction

IVR consists of two parts:

1. Asterisk server
2. IVR application server

### Asterisk Server

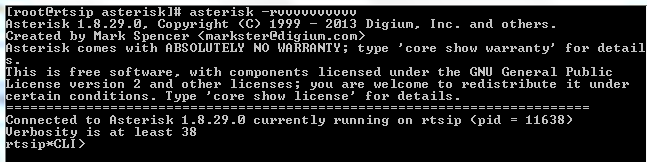
Asterisk server is a software PBX and provides the actual features related to calls & IVR handling. It’s normally run in Linux environment as it’s naturally compatible with it. Generally it’s installed as a Linux service. This service can be accessed as follows:

* To check the service status: **service asterisk status**
* To start the service: **service asterisk start**
* To stop the service: **service asterisk stop**

C:\Users\ahmedmo\Desktop\asteriskService.PNG

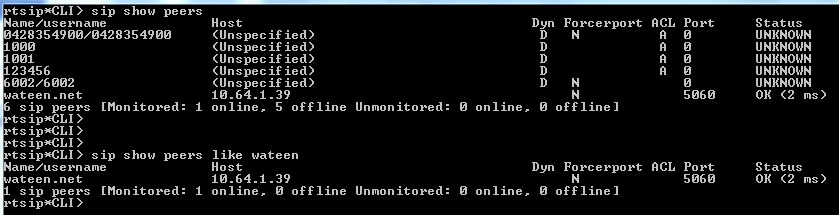
Asterisk provides a CLI environment which shows logs of the server activity as well as provides environment to Asterisk commands. To access Asterisk CLI:

* **asterisk –rvvvv** ( where ‘v’ is for log verbosity level)



To check the status of SIP trunk connectivity:

* **sip show peers** (displays all SIP peers configured in **sip.conf**)
* **sip show peers like <regex>** (same as above except for **like** part which matches against **Name/username** column values in the output)



SIP messages logging can be enabled for further verification as:

* **sip set debug on** (**sip set debug off** to disable it. Should be disabled except when needed)

### IVR Application Server

IVR application is a maven based standalone Java 1.6 project which utilizes AsteriskJava library for Asterisk related features. It’s packaged as **inov8-ivr-<version number><-SNAPSHOT>.jar**. The filename prefix **inov8-ivr** is permanent whereas **version number** is incremental based on semantic versioning. Following are the project dependencies:

aopalliance-1.0.jar

asterisk-java-1.0.0.CI-SNAPSHOT.jar

commons-dbcp-1.4.jar

commons-pool-1.6.jar

jcl-over-slf4j-1.7.7.jar

log4j-1.2.17.jar

mysql-connector-java-5.1.10.jar

slf4j-api-1.7.7.jar

slf4j-log4j12-1.7.7.jar

spring-aop-4.0.6.RELEASE.jar

spring-beans-4.0.6.RELEASE.jar

spring-context-4.0.6.RELEASE.jar

spring-core-4.0.6.RELEASE.jar

spring-expression-4.0.6.RELEASE.jar

spring-jdbc-4.0.6.RELEASE.jar

spring-tx-4.0.6.RELEASE.jar

spring-web-4.0.6.RELEASE.jar

task-core-0.0.1-SNAPSHOT.jar

#### Call Flow

Currently IVR is only used for PIN input & verification via server. The call flow is initiated via SOAP call (initPinRequest) and provided a request DTO containing data needed to initiate the call & play it during the call. The DTO also contains other transaction data which is not required by IVR but needed by microbank when IVR sends back the response. When a request lands on IVR, the required parameters are validated & an API call to Asterisk server is generated for call initiation. If the call fails, then retries are attempted after configured time interval. In any case, a final response is prepared and sent back to microbank server.

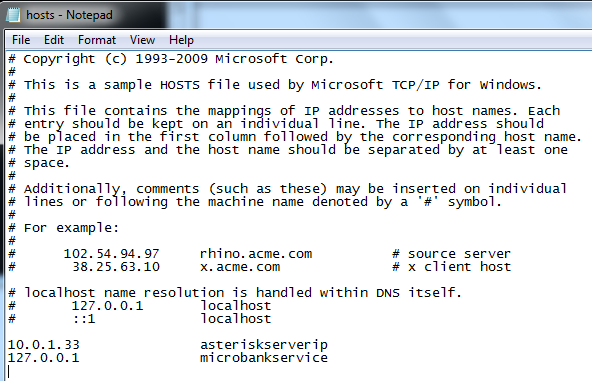
#### Structure

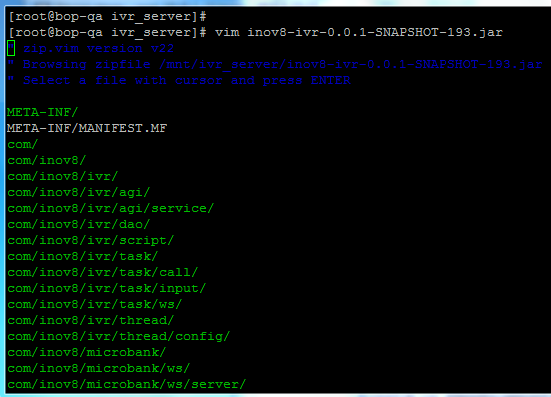
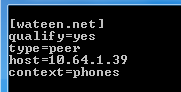
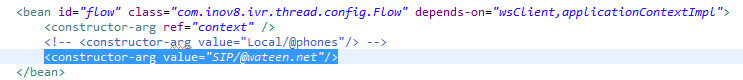
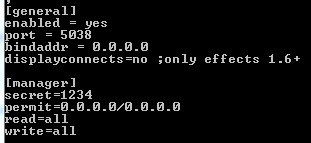
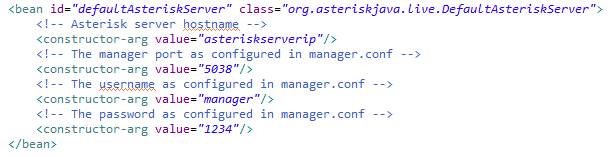
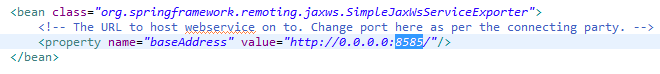
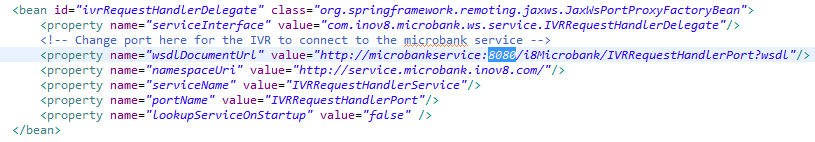
It’s a maven based project having base package ‘com.inov8’. In this package, there is a ‘Main’ class that is the starting point of the application. There are other sub packages as well according to the feature / functionality e.g. ‘util’ package contains utility classes, ‘microbank’ package contains web service related interfaces & classes & ‘ivr’ contains IVR specific classes. Configuration files are in the maven ‘resources’ directory. Spring bean definitions are in ‘agi\_context.xml & app\_context.xml’.

#### IVR Configuration

Before running the application, some app & environment configuration are needed & verified.

1. JDK 1.6 or later should be installed & available.
2. **hosts** file should be updated for **Linux** or **Windows**. In **Linux**, this file is located at **/etc/hosts** whereas for **Windows** at **%systemroot%\ System32\drivers\etc\hosts**. Admin/root level **rights** are needed. For **Windows**, open notepad using context menu option **Run as Administrator**. **Hosts** file contains IP-hostname mappings as **<IP><space/tab><hostname>** per line. Following needs to be in the **hosts** file as per **IVR** configuration files:
   1. No need of the steps ‘**b’** or ‘**c**’ if **IP** is given in configuration instead of hostnames.
   2. By default hostname for **Asterisk** server in **IVR** configuration file is set as **asteriskserverip**.
   3. To connect to **microbank** web service, the hostname by default is set as **microbankservice**.



1. The **microbank** & **Asterisk** servers should be **reachable** & same for **IVR** server. Issue ping command for general connectivity. The **ports** should also be reachable. For **Asterisk**, default port is **5038** & for **IVR** it is **4573** but these can be changed through configuration. Same goes for **microbank**.
2. **IVR** configuration application is as follows:
   1. If the configuration files are within the JAR then in **Linux** use **vim** command and in **Windows** use any compression tool like **WinRar**, **7-Zip** etc to edit the files & update the jar with the modified files.  
        
      
   2. In **project.properties**, update the **IP** in the value of **agi.url** key with the **reachable IP** of the machine it’s on. For example, if **IVR** is on machine ‘A’ having IP 192.168.1.2 using which other machines / servers can access it then set the IP in the above mentioned property value to this IP value.
   3. In **app\_context.xml**, using text search:
      1. Find **id=”flow”**. This will lead to an XML tag named **bean** with the same ID as given in the search. Within this XML tag, set the value of its 2nd child tag, **constructor-arg**, to **SIP/@<context name of sip trunk peer as in sip.conf>**. By default its set to **SIP/@wateen.net** as per current local environment.  
           
         sip.conf  
           
           
           
         bean with id=”flow”  
         
      2. Find **id="defaultAsteriskServer"**. This will lead to an XML tag named **bean** with the same ID as given in the search. Within this XML tag, set the values of all of its child tags named **constructor-arg** as per requirement. 1st value is for **Asterisk** server **IP** or **hostname** (if mapping is given in **hosts** file). 2nd value is for the **port**, 3rd for the **username** (in below snapshot as **[manager]**) & 4th for the **password** (in below snapshot as **secret**) configured in **manager.conf**.  
           
         manager.conf  
           
           
         bean with id=”defaultAsteriskServer”  
         
      3. Find **class="org.springframework.remoting.jaxws.SimpleJaxWsServiceExporter"**. This will lead to an XML tag named **bean** with the same ID as given in the search. Within this XML tag, update the **port** number of the value of its child tag, **property**, having **name="baseAddress"** to the port number the **microbank** server will connect to the **IVR** web service.  
           
         
      4. Find **id="ivrRequestHandlerDelegate"**. This will lead to an XML tag named **bean** with the same ID as given in the search. Within this XML tag, update the **port** number of the value of its child tag, **property**, having **name="wsdlDocumentUrl"** to the port number the **IVR** will connect to the **microbank** web service.  
           
         

#### IVR App Execution

IVR server can be run as follows:

* First check if **IVR** server is already running through the following commands:
  + Linux: **‘ps –ef|grep inov8.ivr’**
  + Windows: Use task manager to find the correct Java process.
* The IVR JAR can be executable containing configuration files within itself. It can be placed with the dependency JARs in the same directory and run in the background with the following commands:
  + Linux: **'java -Dinov8.ivr -jar inov8-ivr-<version#>.jar &'** (tested on Fedora 20)
  + Windows: **'start /B java –Dinov8.ivr -jar inov8-ivr-<version#>.jar'** (tested on Win 7)
* The IVR server can be started & maintained through a Linux bash script, **ivr.sh**. The script when 1st time copied to Linux server needs to be made executable by issuing command:   
  “**chmod +x ivr.sh”**  
    
  The script takes 1 argument which may be **status**, **stop**, **start**, or **restart**. In this case, the configuration & JAR files should be at specified locations.  
  + Configuration: **/opt/inov8/ivr/resources**
  + Dependency JARs: **/opt/inov8/ivr/lib**
  + **IVR** JAR: **/opt/inov8/ivr**

#### IVR Logs

IVR app generates logs based on the **log4j.properties** configuration file. Currently 2 types of log files are generated.

* **root.log** contains general logging info
* **ws.log** contains all request / response related to web services.

The log files location is set in the above properties file against the key **logPath**. The path suffix excluding file name remains the same i.e. **logs/inov8/ivr**. For example /var/logs/inov8/ivr or /mnt/ivr\_server/logs/inov8/ivr or it could be any other path but it should always end with **logs/inov8/ivr**.

By default log paths are as follows:

* If self-executable JAR is running then the logs are located within the same directory as the **IVR** JAR at the path **logs/inov8/ivr**
* If IVR server is being run through script then the log path defaults to **/var/logs/inov8/ivr**