sipXconfig Remote Debugging

Remote debugging of sipXconfig

In many cases you can debug sipXconfig by running it directly from the debugger. However, it is also possible to connect to sipXconfig launched by the sipxpbx service.

Here we describe how to configure sipXconfig Java applications for remote debugging (JDWP). "Remote" in this context means that the Java debugger attaches to the process started by the sipx supervisor. The debugger can actually run on the same machine as sipX services.

While it's not strictly required, the first step should be building Java code with debug information:

```
cd {path/to/sipXconfig/build/dir}
../configure --enable-debug
# recompile all of sipx, or individial projects (e.g. make sipXcommons sipXconfig), it's up to you
make sipx
```

This only applies to sipxecs 4.4 or older systems

```
JAVAC_DEBUG=on JAVAC_OPTIMIZED=off {path/to/sipXconfig/source/dir}/configure <other configure params>
make all install
```

To launch the program in debug mode you need to set debug options, including the communication port for the debugger. sipXconfig startup script lets you pass the necessary information through environment variables.

```
export SIPXCONFIG_OPTS="-Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=4241"
```

Important thing to remember is that sipx-supervisor will run sipXconfig as root. And you need to modify root environment to pass this option properly. It's often easier to user alternative method: pass option through etc/sipxpbx/sipxconfigrc file that is sourced by sipxconfig.sh init script.

```
echo 'SIPXCONFIG_OPTS="-Xdebug -Xnoagent -Djava.compiler=NONE -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=4241"' \
> etc/sipxpbx/sipxconfigrc
```

In the example above we use port 4241. You can use other ports as long as the number you pass to the Java application is the same as the number you pass to the debugger.

Once the process is running you can attach your favorite debugger. If you use jdb, your command line should look like this:

```
$JAVA_HOME/bin/jdb -connect com.sun.jdi.SocketAttach:hostname=localhost,port=4241
```

Other debuggers and IDEs are also supported. Information on how to configure your favorite IDE can usually be found under *Remote Java Debugging* in the user manual.

You can configure Eclipse for remote debugging:

- open Run->Debug... window
- select Java Remote Application
- click

  ```
  New
  ```

- select sipXconfig project, enter host (localhost if you are using it on the same machine) and port number configured in SIPXCONFIG_OPTS
- choose a name for your configuration and save it
- start sipXconfig (by starting sipxpbx) and connect to it by selecting the newly created debug configuration from the Debug window - you'll be placed in DEBUG mode when the application hits one of your breakpoints

If you debug sipxconfig start-up set "suspend=y" option in SIPXCONFIG_OPTS.