PSAM 3.3 Check Server Services

6.1 Services to check

PrivateServer runs a plethora of services, each one committed to its result. Some of them are strictly related and bounded to offer the secure voice connection. These are:

- **MySQL**
- **Asterisk**
- **Asterisk-Trunk**

Equally important is the **Tomcat** service, which provides the web console and thus is responsable for the management the of Asterisk/VoIP via the graphical appliance. This means that Tomcat itself is not mandatory to run the Secure Call service itself, but is very important considering the service as a whole and so including its configuration and management as well.

The **SSH** service is mandatory to perform a deep service analysis, so it is a required resource to be checked first,

6.2 What should we check?

The checks are related to the services performances on the server, meaning that we are not going to measure the bandwidth usage or the external reachabilty for each service. Instead we are going to check that the **PrivateServer** is correctly running the application beside each service exposed.

6.3 How to perform checkings

Besides the NOC operations running, which might warn you about any service problem, we will explain how to perform a direct check-up of the single services. This means to get sure that each service is running and is bound to the expected NIC as a first step. Once assured about this, it means also to get sure that each service is properly responding. "Properly" means within an expected time and furnishing correct informations along with the way you configured it.

6.3.1 Web Console/Tomcat

First of all you make sure that the **Web Console** (meaning the **Tomcat** service) is reachable and running. Use your preferred browser and point it to the Login page, as you did in the First Access Paragraph. Just remember to use the actual IP address and login/password pair for administrator, if you changed any of the above.

If you reach the login page, then the Tomcat and its related application are most probably running fine. If not, then jump to the "6.3.2 SSH" paragraph and from the command line restart the service:

Restart Tomcat

```
[root@ha1 ~]# service tomcat6 stop
Stopping tomcat6: [ OK ]
[root@ha1 ~]# service tomcat6 start
Starting tomcat6: [ OK ]
```

Wait 5/6 seconds to let the Tomcat set up itself completely and then try again to access the Web Console.

6.3.1.1 Preliminary service check

After you get in, please choose the "**Application**" menu entry from the **left side** menu (**main menu**), under the "Server Configuration" section.

Make sure that each application is active on the expected interface. If not, please change the settings accordingly with your needs as described in Applications Settings paragraph. After you're done, check if now the service you are testing is working fine.

If after such modification the service is still causing trouble or if you just want to perform deeper analysis, then go on reading.

6.3.2 SSH

Now you know for sure that the **SSH** service is running on the expected NIC, let's try to reach it. Open you SSH client and connect to the IP address you configured on the NIC you bonded the SSH service. We don't give you detail about ssh client setup as you are expected to know how to deal with it.

Login using the root user and its password (understand how we manage the root password?). Once you're in you get sure that the SSH service is running as expected.

6.3.3 MySQL

MySQL provides the data base for the entire appliance. Thus if it's not working properly, or worse not running at all, the expected outcomes are that no other service is working. We need first to make sure MySQL is up.

6.3.3.1 Check if MySQL process deamon is running at all

Stay in the SSH console and make sure MySQL is running. First check if the mysql.pid file is present. This file is created at the MySQL service boot, so its presence is an evidence the service has been started:

Check the pid file existance

```
[root@ha1 ~]# ls /var/run/mysqld/mysqld.pid
/var/run/mysqld/mysqld.pid
```

Next step is to control that the spoken file is not empty and that it really contains the process id of the MySQL deamon:

Find out MySQL exact PID

```
[root@ha1 ~]# cat /var/run/mysqld/mysqld.pid
1405
```

Now we got such number, we can use it to perform a process search to ensure the process is still in running state:

Check that MySQL is running

```
[root@ha1 ~]# ps auxw | grep `cat /var/run/mysqld/mysqld.pid`
mysql 1405 0.0 3.7 715880 38516 ? Sl Apr06 6:25 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql --
user=mysql --log-error=/var/log/mysqld.log --pid-file=/var/run/mysqld/mysqld.pid --socket=/var/lib/mysql/mysql.
sock
```

If anything of the above steps went wrong, then we try to start the service once again manually and check if any problems arise during the process:

Restart MySQL service

```
[root@ha1 ~]# service mysqld stop
Stopping mysqld: [ OK ]
[root@ha1 ~]# service mysqld start
Starting mysqld: [ OK ]
```

PSAM 3.2 Software Updates

PSAM 3.4 Check Network Status