# Basic GTS Enterprise
# Device Communication Server

# SIMCom eLoc GL100
# Quectel @Track Air GL100
# Queclink @Track Air GL100
# Spark Nano @Track
# Enduro

projects@geotelematic.com
http://www.geotelematic.com

| Manual Revision HIstory | | | |
|---|---|---|---|
| **Rev** | **Date** | **Changed** | **Author** |
| 0.1.0 | 2010/12/31 | Initial Release | MDF |
| 0.1.1 | 2011/04/01 | Added device names Queclink GL100 name. Added "Command" tag information. | MDF |

# Device Communication Server – Quectel GL100

**Contents:**

# 1) Introduction

This manual describes how to configure and run the GTS Enterprise device communication server (DCS) for the **GL100** hardware GPS tracking/telematic devices. This is a small personal tracking device that is marketed and sold under the following names (herein called the "**GL100**"):

- SIMCom eLoc GL100
- Quectel @Track Air GL100
- Queclink @Track Air GL100
- Spark Nano @Track
- Enduro

The following features are supported for the **GL100** DCS:

- Receive incoming data packets via UDP/TCP
- Estimated GPS-based Odometer
- Simulated Geozone Arrival/Departure
- Support for sending commands to the device through the login web-interface.

## 2)  Configuring the Server

The following section refers to the runtime configuration file for the **GL100** device communication server, which can be found in the GTS Installation directory at "`dcservers/dcserver_eloc.xml`".

### 2.1) Changing the Server "Listen" Ports.

The ports on which the **GL100** DCS listens for incoming data packets is specified on the "`ListenPorts`" tag:

```
<ListenPorts
    tcpPort="37490"
    udpPort="37490"
    />
```

If required, the "listen" port can be changed to fit the requirements of your runtime environment.  The script programmed into the **GL100** device will also need to be configured to transmit data to the same port as the server used to listen for incoming data packets.

The "listen" ports must be open through the firewall in order for the remote device to send data to the **GL100** server.

If packet acknowledgment is required, any acknowledgments sent by the server back to the remote device must be sent from the same IP address to which the remote device sent it's data packet.  If your server responds to more than one IP address, then the **GL100**  server listener must be bound to the same IP address/interface used by the remote tracking devices.  This is set in the top-level "`dcservers.xml`" file, on the "`DCServerConfig`" tag, "`bindAddress`" attribute.

### 2.2) Setting the "Unique-ID" Prefix Characters.

The Unique-ID prefix characters can be set in the "`UniqueIDPrefix`" tag section:

```
<UniqueIDPrefix><![CDATA[
    gl_
    qgl_
    imei_
    *
    ]]></UniqueIDPrefix>
```

These prefix characters are used to 'prepend' to the IMEI number reported by the device to look up the owning Account/Device record for this device.  For instance, if the IMEI number is "`123456789012345`", then the system will search for the owning Device using the following Unique-ID keys, in the order specified:

```
gl_123456789012345
qgl_123456789012345
imei_123456789012345
123456789012345
```

Note that the '`*`' character by itself indicates that the system should look up the IMEI number without any prefixing characters.

To bind an **GL100** device to a specified Account/Device record, set the "Unique ID:" field on the Device Admin page to the appropriate prefixed unique-id value.  For example:

```
        Unique ID:  gl_123456789012345
```

### 2.3) Setting the GL100 Properties

Properties which effect the behavior of the **GL100** server are set in the "`Properties`" tag section.  The following properties may be set:

`<Property key="accuracyThreshold">12</Property>`
This is the maximum allowed GPS inaccuracy.  The GPS accuracy value provided in the GL100 event record increases as the GPS fix becomes more inaccurate.  Values below 10 are considered "accurate".  Accuracy values above this threshold will be considered an invalid GPS fix and the latitude/longitude will be set to 0.0/0.0.

`<Property key="minimumSpeedKPH">3.0</Property>`
This is the minimum acceptable speed value, below which the device will considered not moving, and the speed will be explicityly set to "`0.0`".

`<Property key="minimumMovedMeters">0</Property>`
if the specified value is greater than '`0`', then subsequent received events will be omitted if they are within the specified number of meters to the previous event.  Useful for eliminating multiple events at the same location, when the device continues to periodically report a location even if the device is stopped.

`<Property key="estimateOdometer">true</Property>`
If "`true`", the DCS will calculate the current event odometer based on the distance traveled since the last valid GPS location.

`<Property key="simulateGeozones">true</Property>`
If "`true`", the DCS will check for geozone arrivals/departures and insert the appropriate arrive/depart events.

### 2.4) Setting the Available Server-To-Device Commands.

Commands which are to be sent to the **GL100** device based on user requests are defined in the "`Commands`" tag section:

```
    <Commands dispatchPort="31490">

        <!-- ACL -->
        <AclName>acl.dcs.eloc</AclName>

        <!-- Locate Now (Ping) via UDP -->
        <Command name="LocateNow" enabled="true">
            <Type>map,admin</Type>
            <Description>Locate Now</Description>
            <String protocol="udp">AT+GTRTO=gl100,1,%(YMDHMS)</String>
            <StatusCode></StatusCode>
        </Command>

        ...
```

```
        <!-- Locate Now (Ping) via SMS -->
        <Command name="LocateSMS" enabled="true">
            <Type>map,admin</Type>
            <Description>Locate SMS</Description>
            <String protocol="sms">AT+GTRTO=gl100,1,%(YMDHMS)</String>
            <StatusCode></StatusCode>
        </Command>
        ...

    </Commands>
```

The "`Commands`" tag attribute "`dispatchPort`" is used to define the local port that the web-interface will use to connect to the **GL100** device communication server to indicate that a given command should be sent.   The web-interface sends a command request to the **GL100** DCS, which then forwards the command to the **GL100** device based on the specified transport media (ie. UDP or SMS(email)).

The "`Commands`" sub-tag "`AclName`" is used to specify the ACL name used to allow/deny access to this device command feature for specific users.  This ACL name should also be specified in the "`private.xml`" or "`private_common.xml`" file.  To control specific commands, the ACL key defined in the "`private.xml`" or "`private_common.xml`" file, should be this ACL name, followed by a colon (":"), followed by the specific command name.

The "`Command`" sub-tags define the specific available commands.  The "`name`" attribute specifies the command name.  The "`enabled`" attribute specifies whether this command is enabled or disabled.  Disabled commands will not be displayed.

The "`Command`" tag, "`Type`" sub-tag, specifies where this command should be made available for user selection. Valid values may be spearated by commas, and may be "`map`" to specify that the command should be available on the Device Track Map page in a command pull-down selection menu, or "`admin`" to indicate that the command should be available on the "`Properties`" page which can be displayed from the Device Admin list page.

The "`Command`" tag, "`Description`" sub-tag, specifies the Description/Title that will be used to describe the command in the pull-down selection menu, or when displayed on the Device Admin Properties page.

The "`Command`" tag, "`String`" sub-tag, specifies the command which is to be sent to the **GL100** device.  The "`protocol`" attribute specified the transport media used to send the command to the device.  Acceptable values are wither "`udp`" or "`sms`" (sms commands will be sent to the SMS email gateway defined in the Device record). In the case of "`sms`" protocol/transport, the value of this tag is the command string which will be sent to the **GL100** device as a text message (assuming that an appropriate SMS gateway is in use).  The "`%(YMDHMS)`" specification in the comand string will be replaced with the current date/time value as required by the **GL100** device communication protocol documentation.

The "`Command`" tag, "`StatusCode`" sub-tag, specifies the status-code of an audit event that should be generated when this command is sent to the device at the request of a user.  If no status-code is specified, then no event will be generated.

## 3)  Running the Server

To begin listening for incoming events the server must be started.  This section describes the process for manually starting the **GL100** server, and how to set up the system to have is automatically start the **GL100** server on system reboot.


### 3.1) Manually Starting the Server

The command for manuall starting the **GL100** server is as follows:

```
>   cd $GTS_HOME
>   bin/runserver.pl -s eloc
```

To start the **GL100** server with debug logging (useful when testing or debugging), the option "-debug" may be added to the command line.

The server will start, and logging information will be sent to the file "$GTS_HOME/logs/eloc.log".

For debug purposes, it is sometimes useful to have the logging output sent directly to the console, instead of the log file.  In this case the option "-i" can also be added to the command-line.   When logging to the console, hit control-C to stop the server.

To view the running server, you can use the "bin/psjava" command:

```
>   $GTS_HOME/bin/psjava

  PID   Parent  L User      Java class/jar
 ------ ------  - --------  -------------------------------------------------
  54639(     1) 1 opengts   org.apache.catalina.startup.Bootstrap
  68936(     1) 1 opengts   /usr/local/GTS_2.2.4-B23/build/lib/eloc.jar
```

To stop the running **GL100** server, enter the following command:

```
>   cd $GTS_HOME
>   bin/runserver.pl -s eloc -kill
```

This will stop the running **GL100** server.


### 3.2) Automatically Starting the Server on System Reboot

The auto-start script for Fedora is located at "$GTS_HOME/bin/onboot/fedora/opengts", and should have been installed into the system directory "/etc/init.d" when the GTS was installed.

This startup script uses the configuration specified in the file "$GTS_HOME/bin/serverList" to determine which device communication servers to start up when the system is rebooted.  The line in this file that refers to the **GL100** DCS should appear similar to the following:

```
    execServer  "Quectel GL100"  "eloc"  "${option}"   ""
```

The first quoted string contains the DCS description.  The second quoted string contains the ID of the device communication server and must match a library jar file with the same name at "$GTS_HOME/build/lib/eloc.jar".  The third quoted string must contain the exact value "${option}", which is used within the startup script to pass command-line arguments to the DCS startup code.  The forth quoted string is used to pass other optional default or constant arguments to the DCS startup code.

### 3.3) Monitoring the Log Files

When started, the **GL100** DCS will create the following output log files:

`$GTS_HOME/logs/eloc.pid`
This file contains the process-id (PID) of the **GL100** DCS execution process.  This PID is used by the "`-kill`"
option to terminate the running **GL100** DCS.

`$GTS_HOME/logs/eloc.log`
This log file is where all other logging information is placed regarding the receipt and parsing of data from the
remote **GL100** tracking devices.  As this file grows, it will be "rotated" into other log files timestamped as follows:
    `eloc.log.`*`YYYYMMDDHHMMSS`*`.log`
Where "*`YYYYMMDDHHMMSS`*" represents the Year/Month/Day/Hour/Minutes/Seconds time that the file was trimmed
and rotated.

The "`eloc.og`" file will reflect any current connection attempts from remote **GL100** tracking devices.  As devices
send their data to the server, the receipt of the incoming data packets, along with parsing results, will be displayed
in this log file.

# 4) Adding Devices to an Account

When data is received from a remote **GL100** tracking device, the **GL100** server looks up the IMEI number in the Device table to determine which Account/Device owns this device. This section describes how to create a Device record and associate an **GL100** tracking device with the Device record.

## 4.1) Creating a New Device Record

Using the web-interface, log in to the appropriate Account which should own the **GL100** tracking device, then traverse to the "Device Admin" page (or "Vehicle Admin", etc, if so named). Create a new Device as indicated in the GTS Enterprise Tutorial documentation, then "Edit" the newly created Device record.

On the Edit page, there will be a field described as follows:

Unique ID: [                    ]

In this field enter the value "`gl_`*<IMEI_Number>*", replacing "*<IMEI_Number>*" with the device IMEI number. For instance, if the IMEI number is "`123456789012345`", then enter the value "`gl_123456789012345`" in this "Unique ID:" field.

After making changes to the Device record, click the "Change" button.

## 4.2) The "Server ID" Field

The "Server ID" field displayed as a column title on the Device list page, and as a read-only field on the Device Edit page, is assigned a value when the **GL100** device sends its first data packet to the server. Until then, this value will remain blank.

When viewing a list of created Device records with assigned **GL100** devices, records which still have blank "Server ID" fields indicate that no incoming data packet has been received for this particular Device.

# 5)  Testing a New Configured Device

This section describes the process for monitoring newly configured **GL100** devices that have been assigned to an Account/Device record.

## 5.1) Monitoring for Incoming Connections

The Account report "Last Known Device Location" can be used to display the last know location of a given device, which can also be used to determine whether any events have been received from a specific **GL100** device.

The "Server ID" field on the Device record will also indicate if a data packet has arrived from a specific **GL100** device and successfully assigned to the Device record.

If no indication on the Device reports, or "Server ID" field is evident, then the log file itself can be consulted for indications of incoming data packets from the **GL100** device.  The information in the log file can indicate whether an IMEI number may not have been properly assigned, so the **GL100** DCS is unable to determine which Account/Device the incoming data packet belongs to.

## 5.2) Viewing the Unassigned Device Report

In the case where an **GL100** device is put into service without having been assigned to an Account/Device record, or where the IMEI number was incorrectly entered in to the Device record, the **GL100** DCS may not know to which Account/Device the incoming data packet belongs.

When the **GL100** DCS cannot determine the ownership of an incoming data packet, it will place the IMEI and currently GPS location into the "UnassignedDevices" table.  The "Unassigned Devices" report can be selected from the System Administrator login panel ("System Admin" tab, "SysAdmin Reports" menu option, "Unassigned Devices" report).  This report will show the "Server ID"  **GL100**), "Unique ID" (IMEI number), and the last time data was received from this device.  This information can be used to determine whether an IMEI number was ever assigned to an Account/Device record, or if an IMEI number was incorrection assigned to an Account/Device (ie. transposed digits, etc).

## Appendix)

## A) Troubleshooting GL100 Device Connection Issues

The following are fequently-asked-questions regarding commonly occurring connection issues.

**Q**: I've configured an **GL100** device to send data to the server, but have not received any data.
**A**: Monitor the "`eloc.og`" file for possible incoming connections from the device.  If there is no indication that the server is receiving any communication from the remote device, the most common reasons to check include:
- Make sure device has a valid/active SIM card.
- Make sure the device has been programmed with the proper APN ("Access Point Name") configuration as specified by your wireless service provider.
- Make sure the device has been programmed with the proper host and port of your server.
- Make sure the server firewall allows incoming UDP/TCP connections on the specified port.   If the server itself provides its own firewall, then check the firewall settings.  On Linux, this is usually controlled by "`iptables`".  The command to display the current iptables configuration is "`iptables-save`" (must be run as "`root`").  See "http://www.faqs.org/docs/iptables/iptables-save.html" for more information.

**Q**: I see data arriving for my device in the "`eloc.log`" file, but it is always the same event that is being sent over and over.
**A**: If this occurs for all configured/connected **GL100** devices, the problem is likely that returned UDP acknowledgments are not being returned properly to the device (if the device is configured to require acknowledgments).  The most likely reason for this is that your computer responds to more than one IP address, and the returned UDP packets are being sent from a different IP address than the one that the device is configured to send data to.  This can be fixed by setting the "`bindAddress`" attribute in the "`dcservers.xml`" file in the GTS installation directory (then restart the **GL100** DCS).  In some cases, the SIM card wireless service provider does not allow returned UDP packets to be sent from the server back to a device.  In this case, it may be necessary to program the **GL100** devices to not require a return acknowledgment.
**A**: If this occurs for only one device (ie. other devices are reporting as expected), this this is likely due to the GPS receiver's inability to obtain a new GPS fix, and the previous GPS fix is being resent.  This usually means that the device is simply in an area where the GPS satellites cannot be seen (ie. Indoors, etc).  On rare occasions, this can mean that the GPS antenna has become unplugged, or has been damaged.

**Q**: The received events have a valid latitude/longitude, but do not have an odometer value.
**A**: The **GL100** DCS property "`estimateOdometer`" allows enabling a calculated odometer value, based on the distance traveled between successive GPS points.  To enable a calculated estimated odometer value, make sure this property is set to "`true`".