Log Capture Script Testing

This page explains how to perform testing of functions for log_capture* scripts.
One way to test is to create a build using your changes, flash it and then generate a bug report and examine the logs on server.

The second way is to perform function by function testing if you want a quick turnaround and can't wait for build. This writeup explains the second way.

Scrubbing in Dev Mode and Prod Mode

No scrubbing is done in dev mode.
if you want to test scrubbing, your ECU has to be in prod mode.

How to put the ECU in Prod Mode

For the purposes of testing, we can have lc_is_dev_mode return 1 which is considered equivalent to prod mode.

Outline of the Process

1. Copy log_capture_common.sh and log_capture_<ecu>.sh to a location that can be made writable within the ECU's file system. Details down below.
2. Make changes to the *copied* log_capture_<ecu>.sh and to log_capture_common.sh.
3. Run the log_capture_<ecu>.sh file using a certain set of parameters.
4. Examine the result.

Details of the Process

1. Copy log_capture_common.sh and log_capture_<ecu>.sh to a location that can be made within the ECU's file system.

ECU	Location to copy to	Command to make location Writable
Sync	/tmp	mount -u -oexec /tmp
ECG	/tmp	mount -u -oexec /tmp
TCU	/var/volatile/tmp	mount -o remount,rw,exec /var/volatile

2. Make changes to the copied log_capture_<ecu>.sh and to log_capture_common.sh.

   We change log_capture_<ecu>.sh so that it sources the local log_capture_common.sh otherwise it would source the installed one which would most likely be different from your common one and therefore not what you want.

File to Change	Line to Change	Changed line
log_capture_<ecu>.sh	`. ${config_dir}/scripts/log_capture_common.sh` Notice the space between . and $	`. ./log_capture_common.sh` Notice the space between two dots.
log_capture_common.sh	Locate the function lc_is_devmode(). Have it return 1. That's it.	

   We change log_capture_common.sh only if we want our scripts to run in prod mode.
   Note: you need not do this if you want to do perform dev mode testing.
3. Cd to /tmp and make log_capture_<ecu>.sh executable.
   Run log_capture_<ecu>.sh as shown in the table below.
   Here are some examples.

Function to Test	Command to Run	File to Examine
EcgLogPreResetFdpLogs	`./log_capture_ecg.sh ./result.log ./ EcgLogPreResetFdpLogs`	result.log
EcgLogSyslog	`./log_capture_ecg.sh ./result.log ./ EcgLogSyslog`	result.log
EcgLogFnvLog	`./log_capture_ecg.sh ./result.log ./ EcgLogFnvLog`	result.log
EcgLogSlogger	`./log_capture_ecg.sh ./result.log ./ EcgLogSlogger`	result.log

SyncLogConsolePreReset	./log_capture_sync.sh ./result.log ./ SyncLogConsolePreReset	result.log

Note 1 : In some cases there may not be any output in result.log. That's because some functions define their own output file. For example, navigation function

SyncLogNavTelenav generates a number of navigation*.log files in the current directory. So you would examine them instead of result.log.

You can find out what is it by looking at the source.

Note 2: Notice that a script is invoked from the current directory. If you simply run log_capture_<ecu>.sh instead of ./log_capture_<ecu>.sh it'll run from the installation location (/usr/bin?) which is not what you want.