artdaq Utilities - Bug #21739

It should be possible to recover details about why a process didn't launch

01/22/2019 05:04 PM - John Freeman

Status:	Resolved	**Start date:**	01/22/2019
Priority:	Normal	**Due date:**	
Assignee:	John Freeman	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
Experiment:	-	**Co-Assignees:**	

Description

Right now, in direct process management, if a process (or set of processes) doesn't launch on a node, it can be difficult to determine the cause. To avoid a verbose spew to stdout that would overwhelm everything else (especially if a large number of processes are desired) unless we're at the highest debug level (currently 4) the output of the source of the setup script and the launch of the artdaq processes is suppressed. The downside of this is that if the source of the setup script returns nonzero or an artdaq process doesn't launch, the reason for this gets suppressed. Two real-world examples of this include the source of a setupARTDAQDEMO script returning nonzero for the following reason:

```
/home/jcfree/artdaq-demo_dec19/setupARTDAQDEMO: line 61: tonMg: command not found
```

and none of the processes launching on mu2edaq11:

```
boardreader: error while loading shared libraries: libsqlite3_ups.so.0: cannot open shared object file: No such file or directory
```

where it's important to note that the latter error does **not** make it into the MessageFacility logfile - in fact, no MessageFacility logfile gets created at all for the unlaunched process.

A (temporary) record of what happened if something goes wrong with a process launch should be saved, and in the event that something goes wrong, users should be pointed to it. If processes launch without a problem, the record should be deleted.

Associated revisions

Revision 5b4d5699 - 01/22/2019 10:37 PM - John Freeman

JCF: Add the process_launch_diagnostics_base function, to provide detail on what went wrong during a process launch

As described in Issue #21739, when using direct process management, in the event that not all processes successfully launched potentially useful info was getting suppressed. I've now added a function required by process management modules called "process_launch_diagnostics_base", which is called when not all processes launch and is meant to provide the user with further info on what went wrong. For direct process management, this means that the stdout and stderr output of the attempted process launch on every node where something went wrong gets printed; in the case of pmt (for the time being at least) this is just a no-op.

History

#1 - 01/23/2019 04:21 PM - John Freeman

- % Done changed from 0 to 100

- Status changed from New to Resolved

With commit 962ff8fba8d1e8553e69b72c963ec619f42d1fec on the develop branch, if we're running in direct process management mode, then when processes are launched the output on each host is saved in a file called <host>:/launch_attempt_<user>_partition<partition number>. If the processes don't all launch successfully, for each host where things weren't successful the user is pointed to the output file. Note that this file doesn't just contain MessageFacility output, it also contains stdout and stderr, and hence captures the examples given above. It's important to note that for MessageFacility console output we'll want to make the threshold strict (say, errors only, not warning or info) so there isn't a performance hit from output being directed to these files.