

# MCR10197

## IMFT input driver: reattach a closed network connection Revision 1.0

Eric Swenson

October 22, 2026

### 1 Problem Description

An IMFT input driver attached to a network connection (`imft_tcp_io_`, over `tcp_io_ -listen`) stops and needs the operator whenever the remote output driver logs out. The console shows:

---

```
imft_driver_: System-B's output driver disconnected; local driver will wait for
             reconnection.
imft_driver_: End of information reached.
Attempting to synchronize with System-B.
iodd_: Unable to initialize driver.
```

Enter command or device/request\_type:

---

and the operator has to enter the device again. Until then the remote output driver cannot reconnect: nothing is listening on the port, so its connections are refused. An output driver logs out whenever it is stopped by its operator (`logout`) or its process ends, so in practice every restart of either system's output driver leaves the other system's input driver dead.

#### 1.1 Cause

The `imft_remote_logout_` handler in `imft_driver_`, when the driver runs automatically, goes to `RESYNCHRONIZE_DRIVER` to wait for the next `SYNC1` command on the *same* attachment. It drops the line and attaches it again only for dial service (`in_dial=/out_dial=`). That suits a hardwired line, which stays up after the remote driver logs out. A network connection does not: the remote end closes it after the `LOGOUT`, `tcp_io_` then returns `error_table$end_of_info` for every read, and `read_sync_command` treats any error as fatal and abandons initialization.

By contrast, when the connection is closed *without* a `LOGOUT` (for example, the remote process is killed), `imft_receive_object_` returns the error to the driver, which says "Fatal error during "receive" command; driver will re-initialize", detaches the line, and signals `re_init`; the driver recovers by itself. Only the orderly logout leaves it stuck.

## 2 Proposed Fix

In `read_sync_command` (internal to `synchronize_input_driver`), when the read fails with `error_table_$end_of_info` and the driver runs automatically, report

---

```
imft_driver_: Connection with System-B closed; waiting for it to reconnect.
```

---

detach the line (`detach_device`, which sends a hangup only for dial service) and go to `RESYNCHRONIZE_DRIVER`, which finds the line detached and attaches it again (`attach_and_init_line`, which also renews the input-wait event channel). For `tcp_io_ -listen` the attachment waits for the remote system to connect, so this cannot loop. This covers `SYNC1` and `SYNC2`, so a connection closed in the middle of synchronization is also recovered. `error_table_$end_of_info` is added to the declarations.

No new attach or driver argument is needed. The fix keys on the status code rather than on the I/O module: `end_of_info` on a read means the remote end closed the connection and nothing more can ever be read from this attachment. A hardwired line (`imft_tty_`, `imft_hasp_`) cannot be closed by the remote end and does not return it, so those drivers behave exactly as before. In non-automatic operation, nothing changes: the error is reported and initialization fails, as before.

The output driver needs no change: when the remote input driver's connection closes, it already re-initializes and reconnects by itself.

## 3 Documentation

No documentation changes.

## 4 Test

On two systems with an IMFT pair over TCP (A's output driver to B's input driver, both automatic):

1. On A, `logout` the output driver. B's input driver prints the "disconnected" message and then "Connection with A closed; waiting for it to reconnect.", and waits; it does not return to the command prompt.
2. Start A's output driver again. It synchronizes with B without operator action on B, and a request sent with `enter_imft_request` arrives.
3. Kill A's output driver process (no `LOGOUT`). B recovers as before ("Fatal error during "receive" command; driver will re-initialize").
4. Stop A's output driver in the middle of a transfer. B defers the request as before and then waits for the reconnection.
5. Repeat step 1 several times in a row; each time B waits again.
6. Non-automatic B input driver: step 1 gives "Unable to initialize driver" as before.

## 5 Files Changed

- `imft_driver.pl1` (`bound_imft.s.archive`, `>ldd>unb>source`); `bound_imft_` rebuilt and installed in `>unb`.

## 6 Version History

|            |     |                             |
|------------|-----|-----------------------------|
| 2026-10-22 | 1.0 | Initial version of the MCR. |
|------------|-----|-----------------------------|