Bluetooth feature noticed when running PeerHood

This was noticed when PeerHood daemon was run for approximately 8 hours for two times with bluetooth plugin only. D-link bluetooth adapter was used as bluetooth device. PeerHood uses a 15 seconds long inquiries (15*1.28 seconds at maximum) and it waited for 10 seconds between inquiries in first two tests. In third test the delay between inquiries was reduced to one second. In PeerHood tests D-Link device was used as bluetooth adapter. All tests (except the test run on N810 Internet tablet) were run on Ubuntu Linux (v. 8.04) using BlueZ library version 3.29 for Ubuntu.

PeerHood Tests

First and second test

In first run PeerHood daemon crashed (caused a segfault) after 7h 45min, no error messages was received because this version of PeerHood was built without debugging flags. For the second run PeerHood was built from scratch with debug flags to see what caused the error. In second run PeerHood daemon started to report after 8h 20min that it cannot open a new handle to device (message: `HCI device open failed: Too many open files`) and that the inquiry has failed (message: `BTPlugin::InquiryThread: HCI inquiry failed`). The failing of the inquiry resulted that PeerHood daemon started to report that it has gotten negative amount of responses (-1) without crashing. During the second test PeerHood daemon did the inquiry for 990 times before reporting a negative value, when total number of responses was 2209.

Third test

Third test was run with a reduced delay between inquiries and PeerHood did run for 16h 50min before started reporting negative amount of responses. During the third test PeerHood damon did run the inquiry loop for 2954 times before reporting a negative value and total number of responses was 8802.

Problems

Problem #1

The first problem (too many open files) was caused a bug in PeerHood `BTPlugin.cc` code inside `InquiryThread()`-function, this was noticed in second test. A new handle was opened to device every time the loop started and this caused the opening to failure at certain point. This was an unnecessary part in the code because the handle wasn't used in any way after it was opened (or closed). There is no need to open a new handle to device when `hci_inquiry()`-function is used because it will open a handle to device for every inquiry.

Solution to problem #1

This problem was solved by removing the part of the code from `BTPlugin::InquiryThread()` that did the opening of the handle. With further testing no more error messages or situations related to this were encountered. This was removed for third test and no error messages were encountered.

> Still in some cases this problem exists, it is very likely that this is caused by the `hci_inquiry` function and it's functionality. It opens a socket for every inquiry, in PeerHood case it doesn't seem to be able to close them (reporting too many open files). - julaakko

Problem #2

The second problem was that the BlueZ library (`hci_inquiry()` -function) did continuously report an error value after some time. Problem was that in PeerHood the error wasn't handled in any way or even printed in debug messages so another tests were required to be performed in order to get the error set by BlueZ-library (actually the error is set by a lower level function that BlueZ uses). There was no need to perform tests with PeerHood so a smaller test program was developed for this purpose, see Stress tests.

Solution to problem #2

Currently there is no solution for this – issue needs to be investigated further.

> No errors were detected when test program was used as a separate process for every device (second test). - julaakko

> No errors were detected when test program was used as a separate process and only one process was running at a time (third_test). - julaakko

> `Device or resource busy` and `Too many files open` errors do still exist after running PeerHood for several hours (~7 hours) with 15*1.28 second long inquiry and one second delay between inquiries. - julaakko

Possible solutions to problem:

- Received error(s) from library that communication with device isn't possible (resource or device busy) → increase delay between inquiries. If it doesn't solve the problem after some pre-defined time limit try to reset the adapter.
- Received error(s) from library that cannot open a new handle to device → try to reset device to get handles released.
- Make own inquiry-function that doesn't open new handle to device when inquiry is called. (Propose this in BlueZ -forum?)

Conclusions

The problem with `hci_inquiry()` reporting negative amount of responses doesn't seem to be depending of running time and variable usage and it doesn't seem to be related to the amount of responses from other devices, which then would possibly indicate an issue with hardware. One possibility is that library or device cannot keep up with this amount of inquiries. There was no sign of recovering from the fault condition, library continuously reported

negative amount of responses until the end (until was killed several hours from the first error report).

Stress tests

For testing a simple program utilizing BlueZ `hci_inquiry()` -function was used. The purpose of this test program was to do an inquiry to all Bluetooth devices at certain intervals. D-Link DBT-122, 3Com 3CREB96 and Frontline Bluetooth devices were used for testing. All adapters were plugged into same machine and used in same environment with the same amount of surrounding Bluetooth devices. Details about used bluetooth devices:

	D-Link DBT-122	**3Com 3CREB96**	**Frontline - Bluetooth analyzer**
Bluetooth version	2.0	1.1	N/A
Chip manufacturer	Broadcom	CSR	CSR
USB version	2.0	1.1	1.1?
Used USB version	1.1	1.1	1.1

First test

First test was run for approximately 18 and half hours, from 22.7 16:08 to 23.7 10:41 . Test program did an inquiry for all devices one after another with a maximum inquiry duration of 3*1.28 seconds, meaning that one device would block for 3*1.28 seconds while waiting for response to inquiry. With three different devices this resulted in waiting time of 2*3*1.28 seconds at maximum for each device.

Results:

Device	**Inquiries**	**Responses**	**Avg. number of responses**	**Failed inquiries**	**Error messages**
D-Link	5746	22332	3.89	4	Device or resource busy
3-Com	5746	28565	4.97	4	Device or resource busy
Frontline	5589	28674	5.13	161	Device or resource busy

Analysis and conclusion

The problem might not be a hardware issue because library did not continuously report errors as it was an issue in PeerHood daemon tests. Instead the devices did “wake up” after reporting errors mentioned in the table. The D-Link bluetooth device was the least error prone together with 3-Com bluetooth device.

When these results are compared to results with PeerHood daemon tests there is no similarity in the actions of the same device. Although the conditions are almost similar: in PeerHood the delay between inquiries is slightly longer but `hci_inquiry()` is called in similar fashion. But there is a difference in that the PeerHood utilizes only one bluetooth device while this test did the inquiry for three different devices.

This would indicate an possible issue with the library or with the driver; the library or driver might hang or get into a error situation when only one device is used for a long time, with multiple devices it might be so that some variables are reformatted and this results in fewer error situations. Or the library isn't thread safe, in PeerHood it is possible that other threads access the library simultaneously which could then lead to error situations. In this test there was no simultaneous usage of the library, only one access to library per device at a time. It is also possible that the device cannot process the requests (inquiries) fast enough and it gets itself temporarily into some kind of fault condition, in PeerHood tests (2nd and 3rd) it didn't recover at any point. This issue needs to be investigated further.

Second test

Second test was run with a slightly modified version of the test program, a separate process was started for each bluetooth device. There was no delay between inquiries, maximum duration of 15*1.28 seconds. Test was run approximately for 23 hours, from 23.7 10:51 to 24.7 9:57. No errors were detected during the test.

Results:

Device	Inquiries	Responses	Avg. number of responses	Failed inquiries	Error messages
D-Link	6704	22970	3.43	0	-
3-Com	4319	20942	4.85	0	-
Frontline	4325	20597	4.76	0	-

Analysis and conclusion

Results indicate that when library is used in separate memory space it is less error prone - if it is a library issue. In PeerHood tests the inquiry was run inside one thread belonging to daemon process and then the library was (possibly) used by other threads inside the same memory space. This would indicate that errors will happen if the library is in shared use and other thread are using it at the same time. It might be that BlueZ library isn't thread safe and the errors could be related to it.

Third test

Third test was run with same devices and only one device was running the inquiries at a time. Inquiry duration was 15*1.28 seconds at maximum, there was no delay between inquiries. This test was executed because this would show if the library hangs or gets into fault situation when only one process is using it at a time, which was the situation in PeerHood tests, although in

PeerHood tests there was the possibility that other threads inside the daemon process could use the same library. Test was run for 12 hours for each device. Results:

Device	Inquiries	Responses	Avg. number of responses	Failed inquiries	Error messages
D-Link	5673	22770	4.01	0	-
3-Com	2245	12077	5.34	0	-
Frontline	2247	12460	5.55	0	-

Analysis and conclusion

Results would indicate that the problem isn't an issue with the hardware because none of the adapters reported any errors. Most likely an issue with the library not being thread safe.

Fourth test

For fourth test a threaded version of the test program was used which did simultaneous inquiries with the devices to see if more errors occur when library is used by three different threads inside same memory space. Inquiry duration was 15*1.28 seconds for each device and every device waited one second between inquiries. The test was run for 18 hours with the same devices.

Results:

Device	Inquiries	Responses	Avg. number of responses	Failed inquiries	Error messages
D-Link	3705	10398	2.81	6	Device or resource busy
3-Com	3192	13585	4.26	0	-
Frontline	3197	13229	4.14	79	Device or resource busy

Analysis and conclusion

Results indicate that the Bluez library will sometimes fail if separate threads access the same library at the same time inside same memoryspace. It is highly unlikely that this was caused by hardware (previous tests showed that this isn't probably the issue), more likely a thread-safety issue in Bluez. Although this doesn't explain the continuous reporting of failures that was encountered in PeerHood tests. None of the adapters did continuously report errors, Fronline bluetooth adapter did report from 2 to 3 errors in a row but after that continued to work well. D-Link bluetooth adapter reported it's errors in two sequential inquiries and continued to work normally. Although the circumstances were almost similar to ones in PeerHood daemon (duration of inquiry and delay between inquiries was the same, simultaneous usage of the library - although in PeerHood only one adapter is used) same kind of behavior was not detected.

Test with N810

A test was run also on Nokia N810 Internet tablet and the same test program was used. The test was run approximately with 3*1.28 second inquiry duration (at maximum) for 19h 15min and no errors were detected. Program did 17510 inquiries during that time and got 87829 responses to those inquiries. Nokia N810 had 3.22 version of BlueZ library installed.

> In another test after running for 15 minutes PeerHood aborted with error message: thrown by new operator, see Problems in PeerHood.

Analysis and conclusion

Because N810 has only one Bluetooth adapter and the test was run only for it, the possible thread un-safety cannot be checked with this simple test, it would require that some other parts were added to program where library / device would be used in some other way at the same time when inquiries are done. With current program and conditions it was very likely that this test would give similar results to second stress test.

Bluetooth feature issue

Hardware-based problem?

1st stress test doesn't indicate this being a hardware issue - julaakko

2nd stress test supports the claim that this is not a hardware issue. More likely this would be a thread safety issue. - julaakko

Third stress test indicates that the issue isn't hardware related - at least when library isn't used by multiple threads inside same memory space. -julaakko

Library/driver dependant issue?

Third stress test indicates that the issue could be with the library and it not being thread safe. -julaakko

Fourth stress test indicates that some errors do occur when the library is used by multiple threads inside same memory space simultaneously. A very possible thread safety issue. -julaakko

Compare Windows and Linux (perhaps Mac) bluetooth stacks?

> Should we test this? - julaakko

PeerHood related

Other tests with PeerHood acted as if the library couldn't communicate with the device fast enough when the delay was very low between inquiries, reported error was `Device or resource busy`. When the delay was increased to 30 seconds PeerHood started to report that `Too many open files` and hours after that crashed (segfault). Inquiry delay was 15*1.28 seconds.

This would indicate that there is a problem with `hci_inquiry` function - in

some situations it operates too slow (or the device responds too slowly) and in some situations it doesn't close the handles it opens for inquiries (for every inquiry) but on the other hand it could be a result from somewhere else. It is needed to see if there are similar situations as there was in problem #1 that sockets/handles are opened within some thread loop without utilizing them or unnecessarily opening them every time although there could be a more efficient way. - julaakko