Pervasive PSQL
Startup Process

A White Paper From

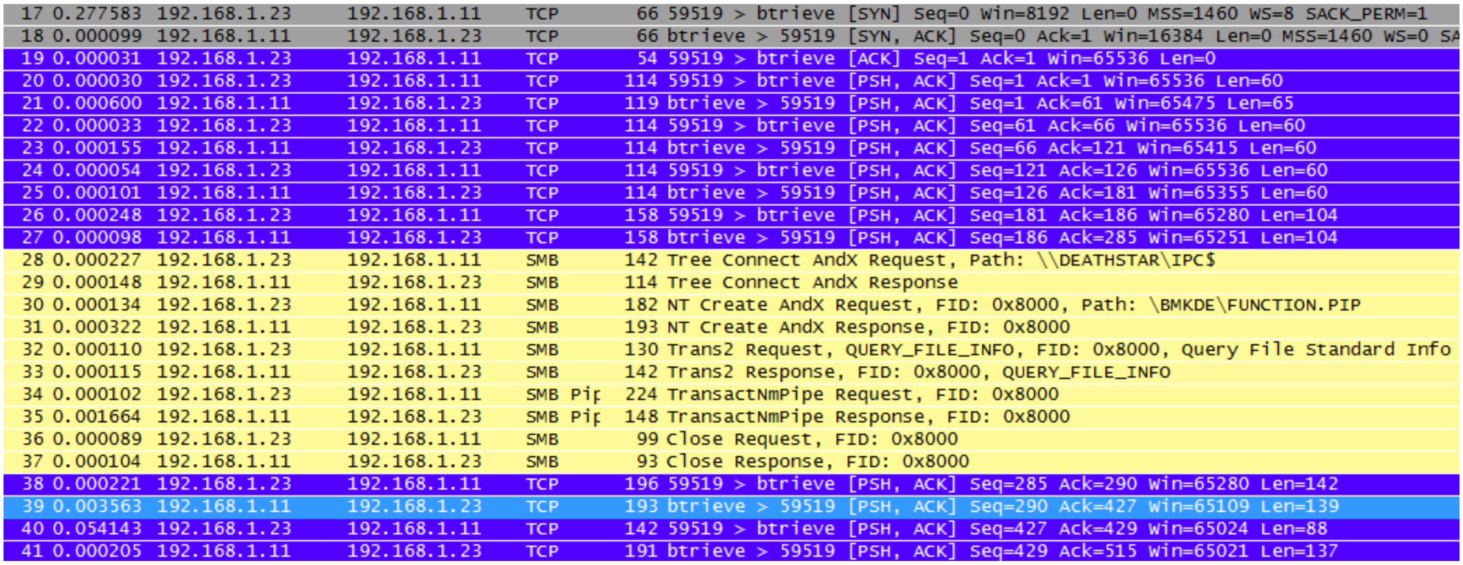
GOLDSTAR
SOFTWARE
www.GoldstarSoftware.com

For more information, see our web site at
http://www.goldstarsoftware.com

The Pervasive PSQL Startup Process

This document explains the basic startup process when a Pervasive-based application opens up a database file on a server for the first time. This information can be useful for troubleshooting your own Pervasive environment by seeing where your system deviates from the “standard” process.

Using Wireshark, we have captured the following conversation of an application opening up a data file on a Windows server.

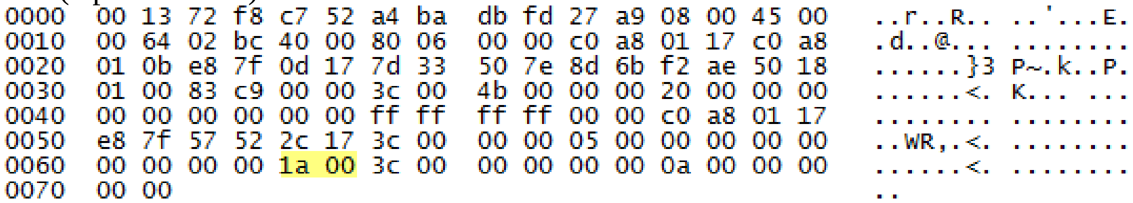


```
17 0.277583 192.168.1.23     192.168.1.11     TCP      66 59519 > btrieve [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=8 SACK_PERM=1
18 0.000099 192.168.1.11     192.168.1.23     TCP      66 btrieve > 59519 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460 WS=0 SA
19 0.000031 192.168.1.23     192.168.1.11     TCP      54 59519 > btrieve [ACK] Seq=1 Ack=1 Win=65536 Len=0
20 0.000030 192.168.1.23     192.168.1.11     TCP     114 59519 > btrieve [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=60
21 0.000600 192.168.1.11     192.168.1.23     TCP     119 btrieve > 59519 [PSH, ACK] Seq=1 Ack=61 Win=65475 Len=65
22 0.000033 192.168.1.23     192.168.1.11     TCP     114 59519 > btrieve [PSH, ACK] Seq=61 Ack=66 Win=65536 Len=60
23 0.000155 192.168.1.11     192.168.1.23     TCP     114 btrieve > 59519 [PSH, ACK] Seq=66 Ack=121 Win=65415 Len=60
24 0.000054 192.168.1.23     192.168.1.11     TCP     114 59519 > btrieve [PSH, ACK] Seq=121 Ack=126 Win=65536 Len=60
25 0.000101 192.168.1.11     192.168.1.23     TCP     114 btrieve > 59519 [PSH, ACK] Seq=126 Ack=181 Win=65355 Len=60
26 0.000248 192.168.1.23     192.168.1.11     TCP     158 59519 > btrieve [PSH, ACK] Seq=181 Ack=186 Win=65280 Len=104
27 0.000098 192.168.1.11     192.168.1.23     TCP     158 btrieve > 59519 [PSH, ACK] Seq=186 Ack=285 Win=65251 Len=104
28 0.000227 192.168.1.23     192.168.1.11     SMB     142 Tree Connect AndX Request, Path: \\DEATHSTAR\IPC$
29 0.000148 192.168.1.11     192.168.1.23     SMB     114 Tree Connect AndX Response
30 0.000134 192.168.1.23     192.168.1.11     SMB     182 NT Create AndX Request, FID: 0x8000, Path: \BMKDE\FUNCTION.PIP
31 0.000322 192.168.1.11     192.168.1.23     SMB     193 NT Create AndX Response, FID: 0x8000
32 0.000110 192.168.1.23     192.168.1.11     SMB     130 Trans2 Request, QUERY_FILE_INFO, FID: 0x8000, Query File Standard Info
33 0.000115 192.168.1.11     192.168.1.23     SMB     142 Trans2 Response, FID: 0x8000, QUERY_FILE_INFO
34 0.000102 192.168.1.23     192.168.1.11     SMB Pip 224 TransactNmPipe Request, FID: 0x8000
35 0.001664 192.168.1.11     192.168.1.23     SMB Pip 148 TransactNmPipe Response, FID: 0x8000
36 0.000089 192.168.1.23     192.168.1.11     SMB      99 Close Request, FID: 0x8000
37 0.000104 192.168.1.11     192.168.1.23     SMB      93 Close Response, FID: 0x8000
38 0.000221 192.168.1.23     192.168.1.11     TCP     196 59519 > btrieve [PSH, ACK] Seq=285 Ack=290 Win=65280 Len=142
39 0.003563 192.168.1.11     192.168.1.23     TCP     193 btrieve > 59519 [PSH, ACK] Seq=290 Ack=427 Win=65109 Len=139
40 0.054143 192.168.1.23     192.168.1.11     TCP     142 59519 > btrieve [PSH, ACK] Seq=427 Ack=429 Win=65024 Len=88
41 0.000205 192.168.1.11     192.168.1.23     TCP     191 btrieve > 59519 [PSH, ACK] Seq=429 Ack=515 Win=65021 Len=137
```

The first column here is the packet number, referred to below. The second column indicates the time elapsed since the previous packet, which tells us about resp[onse times and network latency. In this case, the workstation is at 192.168.1.23, and the server was at 192.168.1.11, so you can see the traffic flow.

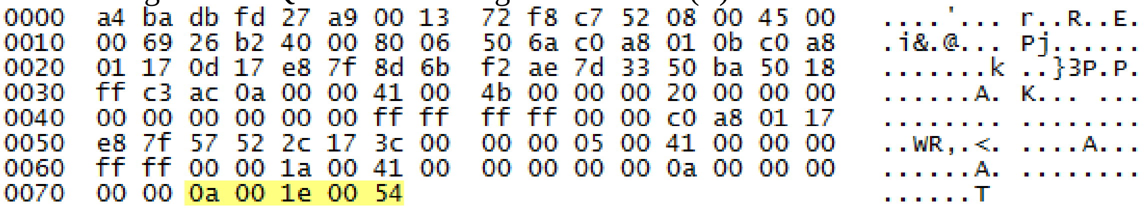
Now, let’s go through the process step by step.

1. The first need is for the workstation to attach to the database server via TCP. This requires a three-way TCP handshake (SYN, SYN/ACK, ACK) that is visible in packets 17-19. This standard TCP process is documented at length in the protocol specs.
2. Then, the database client queries the server to find its version using an old Btrieve 6.x Version call. We see this in packet 20 in the data block where the Version call (Opcode 0x1A) is visible.



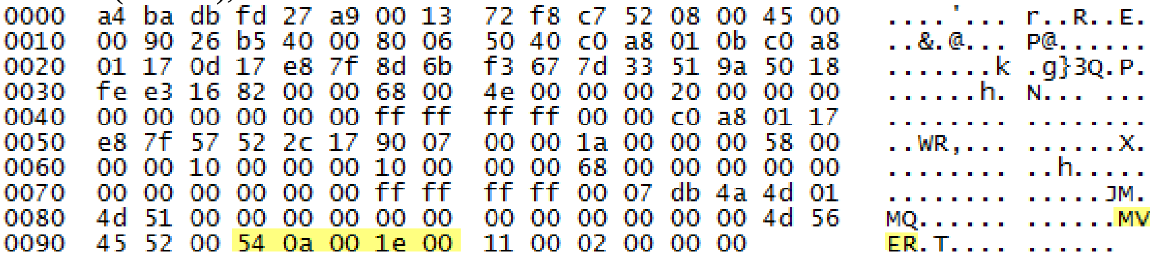
```
0000  00 13 72 f8 c7 52 a4 ba  db fd 27 a9 08 00 45 00   ..r..R.. ..'...E.
0010  00 64 02 bc 40 00 80 06  00 00 c0 a8 01 17 c0 a8   .d..@... ........
0020  01 0b e8 7f 0d 17 7d 33  50 7e 8d 6b f2 ae 50 18   ......}3 P~.k..P.
0030  01 00 83 c9 00 00 3c 00  4b 00 00 00 20 00 00 00   ......<. K... ...
0040  00 00 00 00 00 00 ff ff  ff ff 00 00 c0 a8 01 17   ........ ........
0050  e8 7f 57 52 2c 17 3c 00  00 00 05 00 00 00 00 00   ..WR,.<. ........
0060  00 00 00 00 1a 00 3c 00  00 00 00 00 0a 00 00 00   ......<. ........
0070  00 00                                              ..
```

3. The database replies with the version and engine type in Packet 21, showing us that this engine is PSQLv10.30 running on Windows (T).



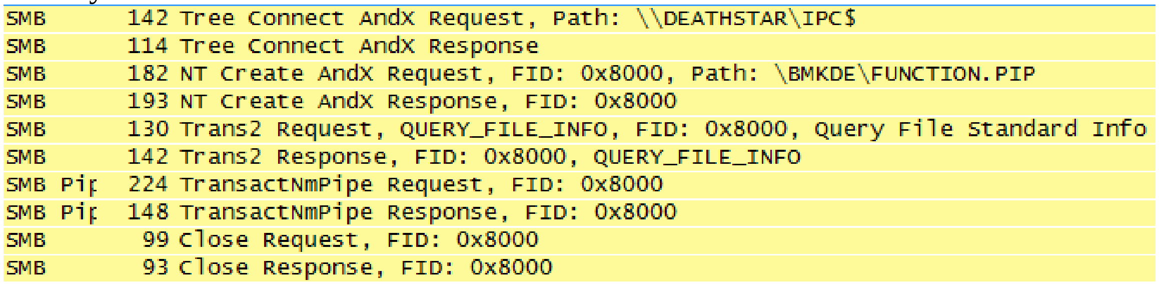
```
0000  a4 ba db fd 27 a9 00 13  72 f8 c7 52 08 00 45 00   ....'... r..R..E.
0010  00 69 26 b2 40 00 80 06  50 6a c0 a8 01 0b c0 a8   .i&.@... Pj......
0020  01 17 0d 17 e8 7f 8d 6b  f2 ae 7d 33 50 ba 50 18   .......k ..}3P.P.
0030  ff c3 ac 0a 00 00 41 00  4b 00 00 00 20 00 00 00   ......A. K... ...
0040  00 00 00 00 00 00 ff ff  ff ff 00 00 c0 a8 01 17   ........ ........
0050  e8 7f 57 52 2c 17 3c 00  00 00 05 00 41 00 00 00   ..WR,.<. ....A...
0060  ff ff 00 00 1a 00 41 00  00 00 00 00 0a 00 00 00   ......A. ........
0070  00 00 0a 00 1e 00 54                               ......T
```

4. Packets 22 and 23 are Btrieve Reset commands, designed to kill off the session that was just opened.
5. Packets 24 and 25 are a negotiation where the Client asks the server if PARC (Pervasive Auto-ReConnect) is enabled, and they negotiate on a reconnect time if needed.
6. Packets 26 and 27 show a newer 7.90 database call to request the Microkernel Version (MVER), which reconfirms v10.30.



```
0000  a4 ba db fd 27 a9 00 13  72 f8 c7 52 08 00 45 00   ....'... r..R..E.
0010  00 90 26 b5 40 00 80 06  50 40 c0 a8 01 0b c0 a8   ..&.@... P@......
0020  01 17 0d 17 e8 7f 8d 6b  f3 67 7d 33 51 9a 50 18   .......k .g}3Q.P.
0030  fe e3 16 82 00 00 68 00  4e 00 00 00 20 00 00 00   ......h. N... ...
0040  00 00 00 00 00 00 ff ff  ff ff 00 00 c0 a8 01 17   ........ ........
0050  e8 7f 57 52 2c 17 90 07  00 00 1a 00 00 00 58 00   ..WR,... ......X.
0060  00 00 10 00 00 00 10 00  00 00 68 00 00 00 00 00   ........ ..h.....
0070  00 00 00 00 00 00 ff ff  ff ff 00 07 db 4a 4d 01   ........ .....JM.
0080  4d 51 00 00 00 00 00 00  00 00 00 00 00 00 4d 56   MQ...... ......MV
0090  45 52 00 54 0a 00 1e 00  11 00 02 00 00 00         ER.T.... ......
```

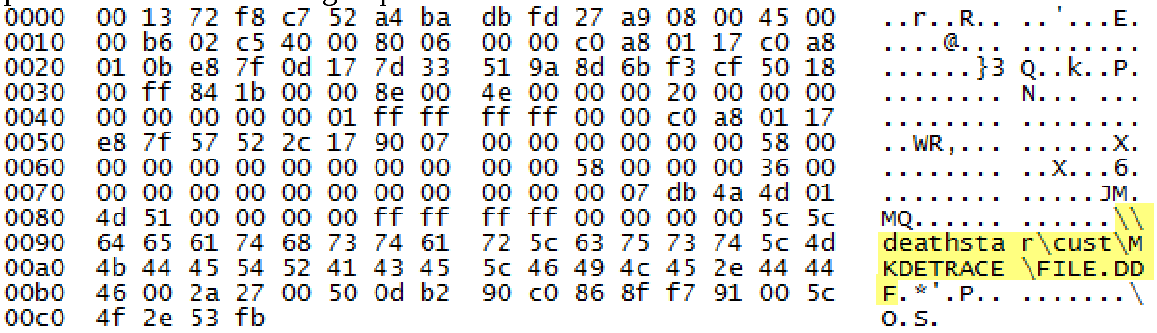
7. Now that the version is known, we have to authenticate to the database server's OS. During this process, a Named Pipe call is initiated to the server to trade security information with the server.



```
SMB       142 Tree Connect AndX Request, Path: \\DEATHSTAR\IPC$
SMB       114 Tree Connect AndX Response
SMB       182 NT Create AndX Request, FID: 0x8000, Path: \BMKDE\FUNCTION.PIP
SMB       193 NT Create AndX Response, FID: 0x8000
SMB       130 Trans2 Request, QUERY_FILE_INFO, FID: 0x8000, Query File Standard Info
SMB       142 Trans2 Response, FID: 0x8000, QUERY_FILE_INFO
SMB Pip   224 TransactNmPipe Request, FID: 0x8000
SMB Pip   148 TransactNmPipe Response, FID: 0x8000
SMB        99 Close Request, FID: 0x8000
SMB        93 Close Response, FID: 0x8000
```

This provides the user account information and returns a security ID that will be used to validate that the user has appropriate rights to access the files on the server. (Note that this is NOT done on the Workgroup Engine, which utilizes no operating system security.)

8. The next packet, 38, includes the FileOpen request, which passes the full UNC pathname to the file being requested.



```
0000   00 13 72 f8 c7 52 a4 ba  db fd 27 a9 08 00 45 00    ..r..R.. ..'...E.
0010   00 b6 02 c5 40 00 80 06  00 00 c0 a8 01 17 c0 a8    ....@... ........
0020   01 0b e8 7f 0d 17 7d 33  51 9a 8d 6b f3 cf 50 18    ......}3 Q..k..P.
0030   00 ff 84 1b 00 00 8e 00  4e 00 00 00 20 00 00 00    ........ N... ...
0040   00 00 00 00 00 01 ff ff  ff ff 00 00 c0 a8 01 17    ........ ........
0050   e8 7f 57 52 2c 17 90 07  00 00 00 00 00 00 58 00    ..WR,... ......X.
0060   00 00 00 00 00 00 00 00  00 00 58 00 00 00 36 00    ........ ..X...6.
0070   00 00 00 00 00 00 00 00  00 00 00 07 db 4a 4d 01    ........ .....JM.
0080   4d 51 00 00 00 00 ff ff  ff ff 00 00 00 00 5c 5c    MQ...... ......\\
0090   64 65 61 74 68 73 74 61  72 5c 63 75 73 74 5c 4d    deathsta r\cust\M
00a0   4b 44 45 54 52 41 43 45  5c 46 49 4c 45 2e 44 44    KDETRACE \FILE.DD
00b0   46 00 2a 27 00 50 0d b2  90 c0 86 8f f7 91 00 5c    F.*'.P.. .......\
00c0   4f 2e 53 fb                                         O.S.
```

9. Packet 39 is the reply showing the file is now open (3.5ms later).
10. The last two packets (40 and 41) are the Statistics call that the client generates to verify the existence and length of each key for key length validation. Notice that the response time here is only 0.2ms, indicating that out network latency is very low on this environment.