Distributed Performance of IBM AIX 21 DCE RPC for AIX version 4.1

May, 10 1996

Debra Berding

IBM DCE Development Austin, Texas

Contents

Abstract

This paper reports on distributed performance of Remote Procedure Calls (RPCs) on IBM Distributed Computing Environment (DCE) for AIX version 4.1. This paper assumes the reader has a basic understanding of DCE RPC.

Trademarks

Performance Measurement Environment

Hardware Environment

The client was an IBM* RISC System/6000* Model 550 with 32MB RAM and 1.1 GB of hard disk. The server was an IBM RISC System/6000 Model 520H with 64MB RAM and 2.0 GB of hard disk. A 10 Mb Ethernet connection was used between the machines.

Software Environment

Both client and server machines had the following software installed:

- AIX* version 4.1
- AIX DCE version 2.1

Performance Measurement Methodology

Two sets of measurements, response times and throughput, were generated using the OSF RPC "perf" test case which measures RPC response time for different data transfer sizes (including null, which transfers no data) and security levels (none, call, packet privacy etc.). The "perf" test case uses an rpc array data type to transfer data from the client to the server.

The response time measurements also include a pipes test case. The pipes test case uses an rpc pipe mechanism to transfer data from the client to the server.

Note that the response time results are for single-threaded RPCs. The throughput results are for multi-threaded RPCs.

Generating Throughput Measurements

DCE is started and a dce_login as cell_admin is executed on both the client and server machines.

The server program, "server", is started with 25 threads and each of the following protocols:

- TCP (transmission control protocol)
- UDP (user datagram protocol)

Two instances of the client program, "client", are run with two threads each and with various data transfer sizes (null and 4KB) and security levels (none and packet privacy).

Generating Response Time Measurements

DCE is started and a dce_login as cell_admin is executed on both machines before beginning the benchmark run. The server is started with 5 threads. The client is run with one thread and various data transfer sizes (null, 4K, 8K, 100K and 2MB) and security levels (none, packet integrity and packet privacy).

When using the "UDP" or "TCP" protocol, the client and server may be on the same or different machine. When using the "TCP" protocol, and the client and server are on the same machine, local RPCs are optimized (by using Unix domain sockets).

The "TCP" and "UDP" results discussed in this paper are with the client and server on different machines. The "local" results discussed in this paper are with the client and server on the same machine.

RPC Throughput Results

Following is RPC throughput, measured in RPCs per second. When comparing throughput speeds, higher is better. The highest throughput is desirable. Note that the throughput numbers are generated using multi-threaded RPCs. See "Appendix A: RPC Throughput data" for the actual throughput numbers.

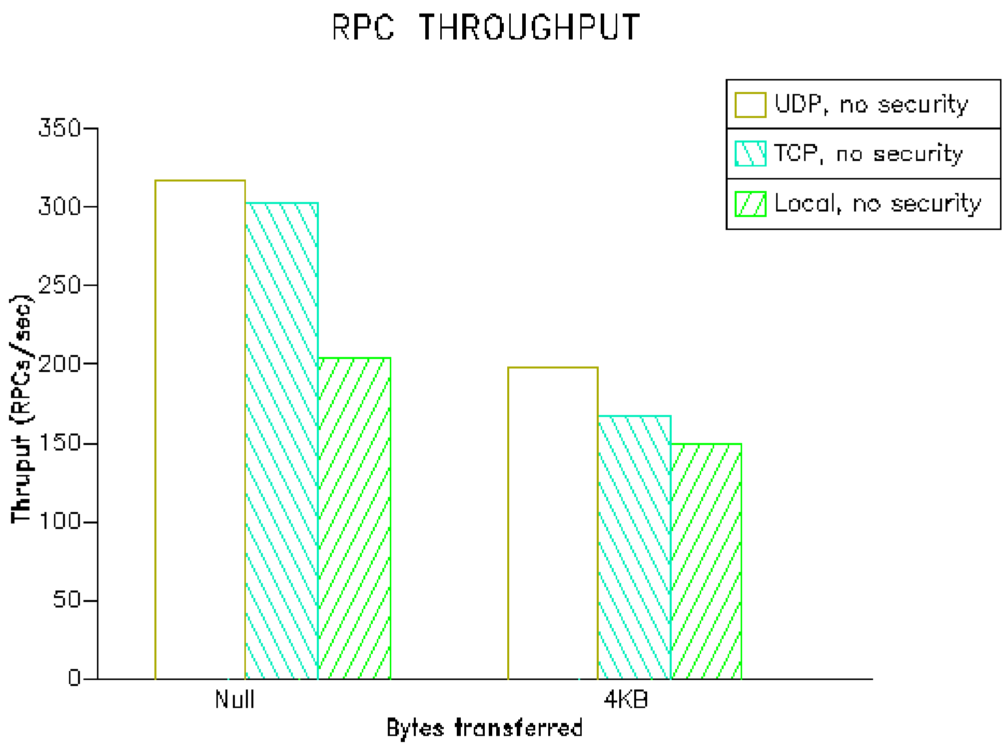

Figure 1: RPC Throughput - no security

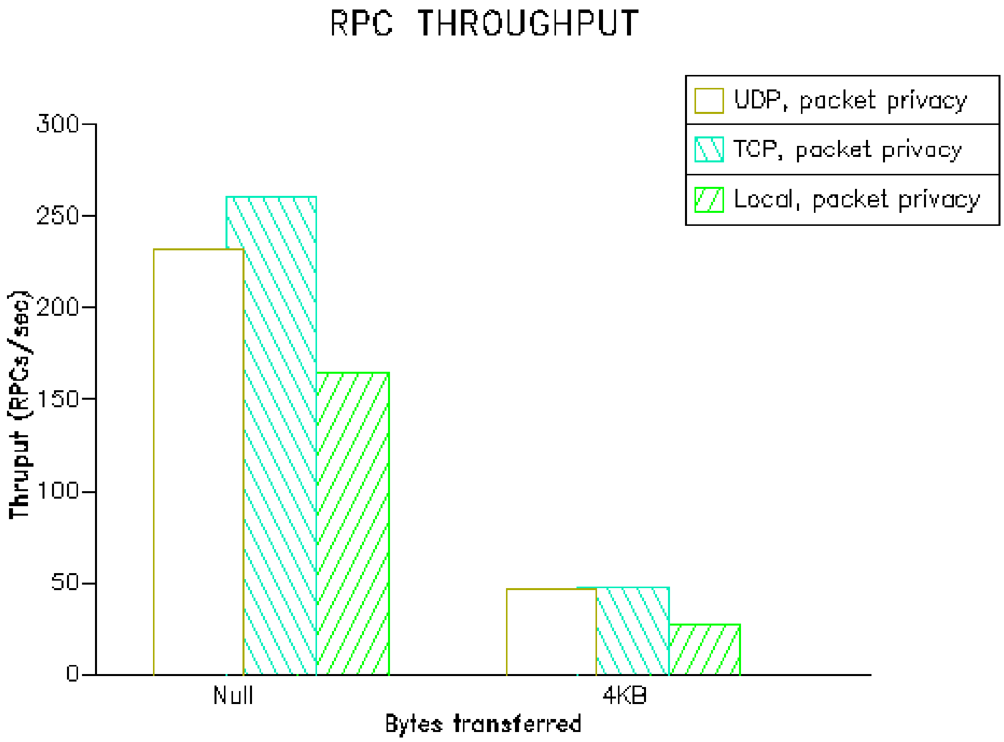

Figure 2: RPC Throughput - with encryption (packet privacy)

Observations:

- UDP has a similar throughput rate to TCP for the null data transfer size without security and for the 4KB data size with security.
- UDP has a slightly higher throughput rate than TCP for the 4KB data transfer without security and a slightly lower throughput rate than TCP for the null data transfer size with security.
- The performance cost of encryption is very apparent. See Figure 4 for additional performance information on different levels of security.
- For multi-threaded RPCs, local is not as efficient as UDP and TCP for smaller data transfer sizes. This is due to the fact that both the client and server are running on the same machine. However, see Figure 3 for response times for local single-threaded RPC performance.

RPC Response Time Results

Following are the RPC response times. When comparing response times, note that lower is better. The fastest response time is desirable. The response times are generated using single-threaded RPCs. See "Appendix B: RPC Response Times data" for the actual response time numbers.

-

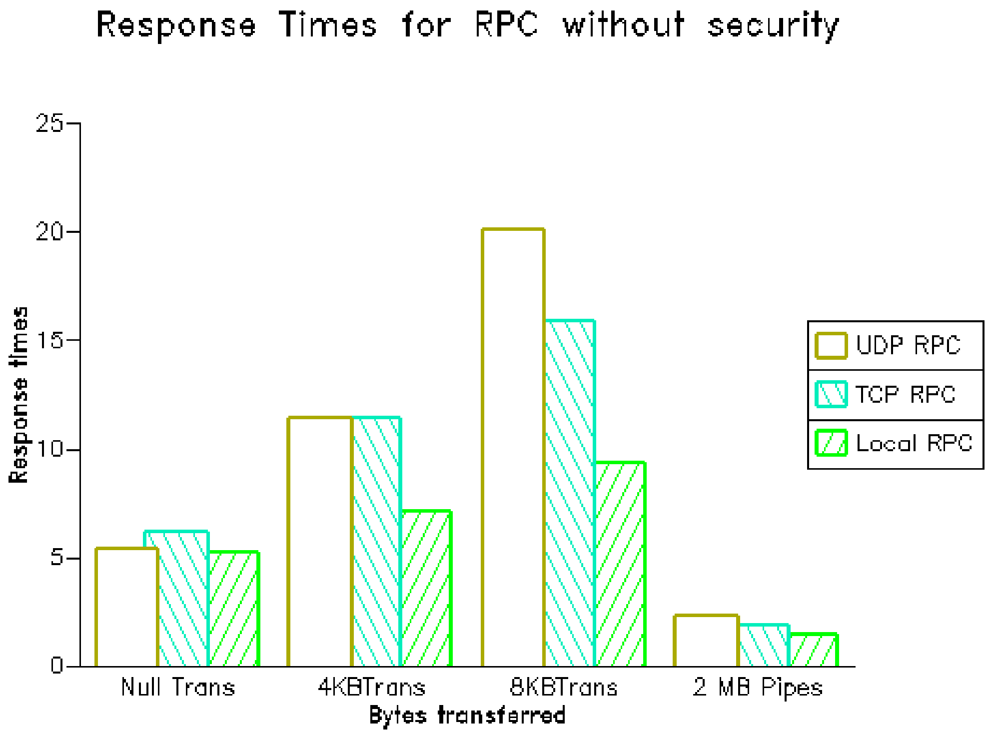

Figure 3: Comparison of UDP, TCP and local response times

Note that null, 4KB and 8KB transfers are measured in milliseconds, while pipes are measured in seconds.

Observations:

- The overhead for RPCs using TCP and UDP is similar for smaller data transfer sizes. However, TCP is more efficient at handling data for the larger data transfer sizes.
- Local RPCs using the TCP protocol automatically direct the data through the Unix stream transport layer resulting in improved performance. Local RPCs are 4-37% faster for smaller data transfer sizes and 40-55% faster for larger data transfer sizes than the "UDP" and "TCP" protocols for single- threaded RPCs.
- The pipes test case is comparable for UDP and TCP. Local RPC using pipes is approximately 25-40% faster than both UDP and TCP.
- Again, encryption is the most expensive security level from a performance perspective. See Figure 4 which shows the cost of different levels of security. There are 6 levels of security for RPCs:

```
none                    no security at all
connect                 checks for valid credentials (dce_login)
call                    checks for credentials for every RPC call
packet                  validates every packet within an RPC
packet integrity        packet and cyclic redundancy check (CRC)
packet privacy          packet integrity and encryption
```

Note that the first 4 security levels have a similar amount of overhead. Packet integrity is the point at which the overhead (approx 25-55%) becomes noticeable. Packet privacy level of security gives an overhead of approximately 50-300% depending on data transfer size.

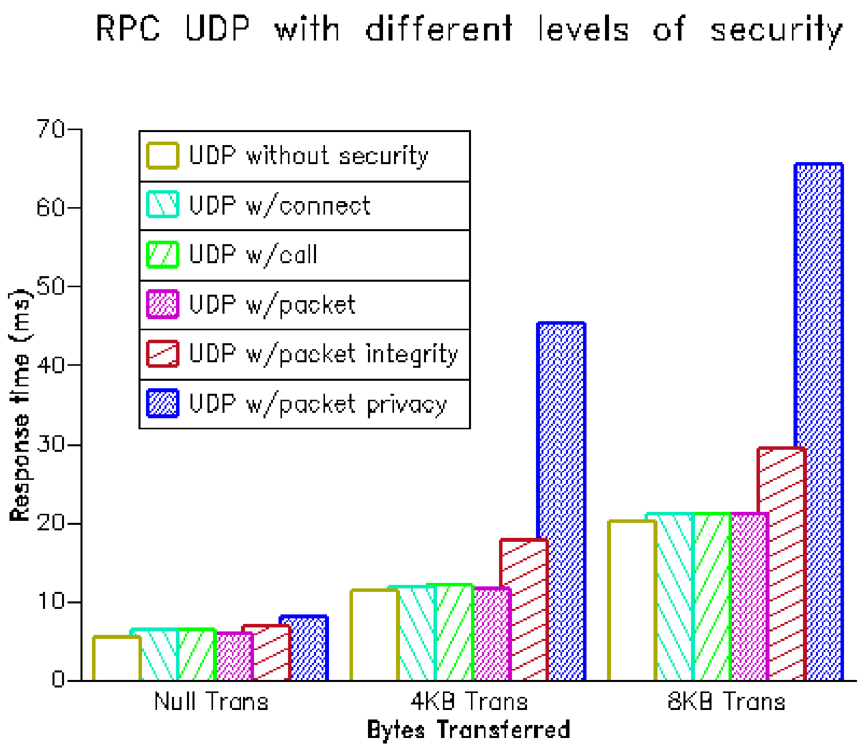

Figure 4: Comparison of UDP response times for different levels of security

---

Summary

- Use TCP for larger data transfer sizes (8KB+) and UDP for smaller data transfer sizes (less than 4KB) to optimize response times. To optimize throughput, use UDP without packet privacy and TCP with packet privacy.
- Local RPCs are faster in the single-threaded case.
- RPC with packet privacy is very expensive. It should be used only when necessary. Note that all the RPC security levels up to packet integrity have similar overhead. Packet integrity gives an acceptable level of overhead and protects against alteration of data by a third party.

Appendix A: RPC Throughput data

The RPC throughput results follow:

```
Throughput measured in RPCs/sec

RPC - UDP
 -NULL THROUGHPUT     317
 -4KB THROUGHPUT      198

 RPC - TCP
  -NULL THROUGHPUT    303
  -4KB THROUGHPUT     167

 RPC - Local
  -NULL THROUGHPUT    204
  -4KB THROUGHPUT     149

 RPC - UDP, SEC:
   PKT PRVCY
   NULL THROUGHPUT    232
  -4KB THROUGHPUT      47

 RPC - TCP, SEC:
   PKT PRVCY
  -NULL THROUGHPUT    261
  -4KB THROUGHPUT      48

 RPC - Local, SEC:
   PKT PRVCY
  -NULL THROUGHPUT    165
  -4KB THROUGHPUT      27
```

Appendix B: RPC Response Times data

The RPC response time results follow:

```
RPC ... UDP
NULL TRANSFER  (MS)             5.4
4 KB TRANSFER  (MS)            11.4
8 KB TRANSFER  (MS)            20.1
2 MB PIPES     (SEC)            2.3

RPC ... TCP
NULL TRANSFER  (MS)             6.2
4 KB TRANSFER  (MS)            11.4
8 KB TRANSFER  (MS)            15.9
2 MB PIPES     (SEC)            1.9

RPC ... UDP, SECURITY
CONNECT
NULL TRANSFER  (MS)             6.3
4 KB TRANSFER  (MS)            11.8
8 KB TRANSFER  (MS)            21.0
100 KB TRANSFER  (SEC)          0.1
```

```
RPC ... UDP, SECURITY
CALL
NULL TRANSFER  (MS)              6.3
4 KB TRANSFER  (MS)             12.0
8 KB TRANSFER  (MS)             21.0
100 KB TRANSFER  (SEC)           0.1

RPC ... UDP, SECURITY
PACKET
NULL TRANSFER  (MS)              5.9
4 KB TRANSFER  (MS)             11.6
8 KB TRANSFER  (MS)             21.1
100 KB TRANSFER  (SEC)           0.1

RPC ... UDP, SECURITY
PCKT INTGRTY (CHECKSUM)
NULL TRANSFER  (MS)              6.8
4 KB TRANSFER  (MS)             17.7
8 KB TRANSFER  (MS)             29.4
100 KB TRANSFER  (SEC)           0.2

RPC ... UDP, SECURITY
PCKT PRVCY (ENCRYPTION)
NULL TRANSFER  (MS)              8.1
4 KB TRANSFER  (MS)             45.4
8 KB TRANSFER  (MS)             65.6
100 KB TRANSFER  (SEC)           0.6

RPC ... Local
NULL TRANSFER  (MS)              5.2
4 KB TRANSFER  (MS)              7.1
8 KB TRANSFER  (MS)              9.4
2 MB PIPES  (SEC)                1.4
```

Appendix C: Acknowledgements

I would like to thank Dave Bachmann and Ray Pekowski for their input on the contents of this paper. I would also like to thank Tom Jones for his help in getting the charts in format suitable for the Web.

Appendix D: Biography

Debra A. Berding is a Staff Programmer. Currently, she is working in IBM Austin, TX, as a performance analyst on AIX and OS/2 DCE. She has an MS degree in computer science from Marist College.


```
Phone Mail:  (512) 838-1790
Internal  :  T/L   838-1790

FAX       :  (512) 838-5957
```

Internal : T/L 678-5957

email : dberding@vnet.ibm.com