What I Learned This Month: Java and DNS Issues, Take 2

Scott Chapman
American Electric Power

I had thought I had figured out the DNS issues with Java as Java caches DNS lookups forever by default. [1] As it turns out, that changed slightly in Java6—or at least the documentation changed slightly. To be honest, I'm not sure that I'm knowledgeable enough about Java to really correctly interpret the documentation. But we did recently run into a new and different problem with Java and DNS.

In particular, our primary DNS server went down one night due to a hardware failure. Of course we have backup servers and those DNS servers responded to DNS lookups just fine. So why did I get roped into an emergency performance call at 9 pm? A newly deployed application running in WebSphere Application Server v7 on z/OS was having major performance problems. In fact, it appeared to the application team that DB2 was somewhere between extremely slow to completely non-responsive.

As we were talking on the phone, somebody mentioned that the primary DNS server was down. At first we didn't think that that had any bearing on anything since we could manually do name lookups just fine. The backup servers were handling the requests a little slowly, 10 to 20 seconds, but 20 seconds is not multiple minutes and Java caches all those forever, right? And didn't I get called because the application is having trouble getting to DB2?

However, the application is using a type-4 JDBC connection instead of a type-2 connection. A type-2 connection is a direct cross-memory connection to DB2. A type-4 connection goes across TCP/IP and comes into DB2 via DDF. You might expect that you'd always want to use a type-2 connection from WAS on z/OS. But because DDF work is zIIP-eligible and because a type-4 connection can switch between active DB2 subsystems more easily, a type-4 connection **may** perform better and may allow for better availability. You need to test your applications to determine which is best for you. And I've heard some rumors that the rules may change in the future. So you may have to test again later. However, for this article, the important point here is that a type-4 connection goes across TCP/IP. But doesn't Java cache DNS lookups forever?

Somebody on the call mentioned that they had heard that another Java application server on another platform had been having similar performance problems since the DNS went down. They resolved the issue by moving one of the backup DNS servers to the top of the list of name servers. Personally, I had never even looked at that configuration in z/OS. It turns out that our DNS timeout value was set to 10 seconds for 2 attempts. That explained why nslookup was taking 20 seconds. But it didn't explain why we didn't seem to be getting responses back from DB2. Finally, lacking any better ideas, we moved the non-

[1] See http://www.cmg.org/measureit/issues/mit76/m_76_7.pdf

responsive primary DNS to the bottom of the list, and the application almost immediately started working normally!

So what happened here? I'm still not entirely sure. But I was able to replicate the problem the next day. You can override the system TCP/IP parms for a given batch job by including a SYSTCPPRM DD. So I did that for a Java batch job that did a single SQL statement to DB2 via a type-4 JDBC connection. This took about 2-5 seconds normally. Then I simulated a DNS failure by replacing the primary DNS address with a bogus address. My test job went from around 5 seconds to 80-100 seconds. I then changed the timeout value down to 1 second and the job took about 15 seconds. Clearly the impact is much higher than just the timeout value, but a higher timeout value did have a larger overall impact.

Unfortunately, I haven't figured out why this is. A couple of days after this happened we ran into bigger problems with this application and I haven't been able to get back to exploring the DNS impact on JDBC type-4 connections. Hopefully that problem will generate another article soon.

So what is the lesson this month? Simple: DNS failures may impact more things in more ways than you expect, at least in the Java world. We didn't note any significant problems with any of our non-Java applications. And keep an open mind about apparently non-related problems somehow being related to the problem you really care about.

As always, if you think I didn't learn my lesson correctly (or in this case can explain the behavior we saw with the JDBC connections) please email me at sachapman@aep.com and let me know!