the Availability Digest

www.availabilitydigest.com
@availabilitydig

# Lloyds Banking Group Outage – a Correction

January 2017

In our February, 2014, and April, 2015, issue of the *Availability Digest*, we described an outage suffered by the Lloyds Banking Group. A subscriber to the *Digest* has pointed out that we were in error. We correct that error in this article.

Our earlier articles reported the following:

> "The Lloyds Banking Group Outage:
>
> On the afternoon of January 26, 2014, customers of the banks comprising Lloyds Banking Group could not use their debit cards; nor could they withdraw money from ATMs. The Lloyds Banking Group banks include Lloyds, TSB, and Halifax. Hundreds of thousands of customers, left at checkout counters or gas stations, were unable to pay for their purchases. The outage lasted from 3 PM to 7:30 PM. Once service was restored, there were additional delays as the backlog of transactions was cleared.
>
> According to sources, there was no maintenance or update activity going on at the time of the failure. Rather, the failure was caused by two of seven servers that process debit-card transactions. Conjecture is that one of the servers was a production server, and the other was its backup. It appears that the bank lost a server and then suffered a failover fault."

These reports were based on incomplete information available at the time as well as some conjecture. Our subscriber has reported that the following is an accurate description of the outage:

> "The Lloyds Banking Group Outage:
>
> Lloyds employed two HP NonStop 16200 systems in an active/active configuration to process its ATM and POS transactions. The outage was caused by a simultaneous failure of four out of eight CPU processors in one of the NS16200 NonStop production servers, which handled 50% of all ATM and POS traffic. The NS16200 server handling the other 50% of traffic was unaffected.
>
> The root cause of the failure was finally identified by HP as a rare firmware bug in HP's Logical Synchronization Module. The bug was so subtle that HP had to take the affected part back in order to reproduce the failure.
>
> Effectively, the failed server was still running at the point of failure but was ruled as "sick but not dead." This resulted in about 50% of transactions failing. However, the sick system had not really failed, thus complicating the failover process. Automatic failover to the other half of the active/active system did not take place. The transaction traffic had to be switched over manually to the other (unaffected) HP NonStop server.

> As this issue caused a major embarrassment and adverse publicity for Lloyds, they sought legal redress from HP; but a settlement was eventually reached out of court.
>
> Lloyds subsequently migrated to HP NonStop NB56000 servers in August that year and updated the disaster recovery process to handle the "sick, but not dead" server scenario.
>
> As this was an unusual type of failure, we are wondering whether there are many businesses out there that may not have ever thought through this type of failure event, let alone even planned or tested for it?"

We are sure that most users of NonStop systems never consider the simultaneous failure of multiple CPUs in a system. NonStop systems usually are configured so that they can carry the anticipated load in the presence of a single CPU failure. Multiple CPU failures represent a failure of the NonStop system that will require failover to an alternate system. Since most companies do not thoroughly test failover procedures, the failure of the production system easily can lead to a failover fault that will take down all services. This is probably what happened at Lloyds, "complicating the disaster recovery process" as our subscriber reported.

Do your failover testing procedures include the "sick but not dead" syndrome? How do you determine that a system is sick but not dead? One way would be to follow the lead of Lloyds. If a system appears to be operational, but it is rejecting transactions, then it is likely "sick but not dead." In this case, the system must be taken out of service and its transactions routed to the operational system.

Our thanks to our subscriber (who wishes to remain anonymous) for this correction and clarification. If you feel that an article is in error, please get in touch with the *Availability Digest* editor at editor@availabilitydigest.com and let us know.