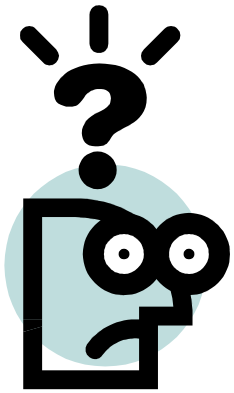

Are My Critical Systems *Really* Available "24x7?

A goal of "24 by 7" is a fine objective for critical information systems, but I believe many businesses are fooling themselves when they claim they are meeting this goal. Ironically, the problems with most high availability solutions aren't really due to flaws in the technology, but how they are *implemented*. In particular, I have noticed that "hot standby" systems are especially weak links.

In my experience, most high availability systems fail for these reasons:

(1) Critical components are missed
In order for a failover solution to work, every part must failover properly (or have some type of redundancy). If any piece is missed, you don't really have high availability. Unlike school, a score of 99% correct isn't good enough--it gets an "F" just like the system that is only 1% correct. When your online system is down and customers are mad, management doesn't really care that you got it *almost* right.

At one large insurance company, a critical server failed. As planned, the database failed over just fine, and was 100% usable. Unfortunately, a major part of the *application* didn't, because the designers neglected to include it in the process.

(2) Incomplete testing of the failover process
In the above case, a simple failover simulation would have immediately uncovered the deficiency. Incredibly, technical management assumed everything would work fine--*without testing*. In another case I witnessed, a failover didn't work because file systems had been modified on one node (the current node), but not on the failover node. This flaw was only uncovered when the failover didn't work.

(3) Overlooking likelihood of human error
By this I mean blunders made after the high availability system is implemented. I think this is the toughest problem to solve. The key focus in high availability solutions is usually reliability of the hardware. In reality, however, failures are often the result of *human error*.

Several years ago, I helped setup a VCS (Veritas Cluster Server) high availability solution. We tested everything many times. Failover worked flawlessly. A few months later, however, a real outage occurred, and the failover didn't work! Investigation showed that the system administrator (a very competent individual by the way) had recently deleted a directory that he thought wasn't needed. Of course, the directory contained files vital to VCS.

How could we have known that someone did something silly just after we stopped testing? The most obvious answer to this problem is to add strict configuration control, or some form of double-checking to the process. At one firm, a project manager, frustrated by multiple DBA mistakes, briefly implemented a policy of having one DBA look over the shoulder of another DBA while he typed in commands.

Wrap-up
High availability solutions that use "always on" multiple nodes avoid many of the problems intrinsic to hot-standby databases. Oracle's *RAC*, or multiple nodes regularly updated via replication are good examples. If your users are divided among multiple nodes, there is no question as to whether a given node is usable or not. Instead, the focus turns to capacity planning, and the logistics of promptly transferring users to an alternate node.

True 100% system uptime will likely remain a tough goal, but proper planning and testing can increase the uptime dramatically. To really achieve 24x7 however, we need to focus on the root cause of most system outages. The really difficulty problem is not how to increase hardware reliability, but how to increase *human* reliability. How do we design a system that provides this assurance: "People will never make a mistake" ?