the Availability Digest

www.availabilitydigest.com
@availabilitydig

How Does Failover Affect Your SLA?

December 2014

Service Level Agreements (SLAs) usually include a limit on the amount of downtime that is tolerable for an application. Each application typically has its own SLA requirements. The SLA may exclude certain types of downtime, such as for periodic maintenance (planned downtime). It may also apportion the downtime calculation over the number of users impacted. For instance, if an hour of downtime affected only half the users, then the outage counts as a half hour of downtime.

Specifying Downtime

Often, rather than specify the allowable downtime, the SLA instead will stipulate the required uptime as a certain number of 9s. For instance, a requirement for an uptime of three 9s means that the application will be available 99.9% of the time. That is, the SLA will allow only 0.001% of downtime annually. This is equivalent to being down 8.76 hours per year.

The equivalence of uptime measured in 9s to downtime per year is as follows:

Two 9s	87.6 hours per year
Three 9s	8.76 hours per year
Four 9s	0.876 hours per year (52.6 minutes per year)
Five 9s	5.26 minutes per year
Six 9s	0.526 minutes per year (31.5 seconds per year)
Seven 9s	3.15 seconds per year
Eight 9s	0.315 seconds per year

An easy way to remember this table is that five 9s equals five minutes. Multiply by factors of ten or divide by factors of ten to obtain approximate values for the other entries.

Field experience shows that today's systems offer inherent availabilities in the range of three to four 9s. Windows and Linux servers generally deliver availabilities of 0.999 to 0.9995. Fault-tolerant systems like HP NonStop and Stratus ftServers provide availabilities of 0.9999. This is not to say that the hardware or operating systems of these platforms have such availabilities. Platform availabilities can range in the six to seven 9s. Instead, it may be other factors that reduce availability – application faults, operator errors, power and cooling outages, etc.

Improving Availability via a Backup System

Clearly, if an availability greater than the inherent system availability is required, there must be a means to continue operation in the presence of a system failure. This is often accomplished via a backup system to which application processing can fail over in the event of a production system failure. However, bringing the backup system online takes some time. How does that time affect the overall system availability? We explore this question below.

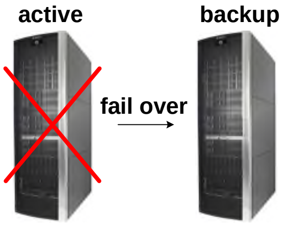
Your New SLA

Let us consider an illustrative example. You head your company's IT department. Meeting the availability requirements of the application SLAs is your responsibility.

You are currently running your critical applications on a NonStop server. You are confident that your applications will exhibit high availability – after all, NonStop servers are fault-tolerant. They will survive any single fault in the system. Your experience is that your NonStop server will fail about once every five years and will take about four hours to return to service. This represents an average downtime of 0.8 hours per year or four 9s of availability.

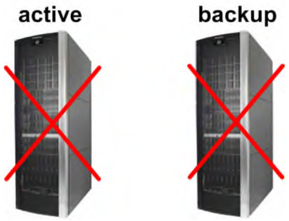
Your company is launching some new mission-critical applications, the SLAs for which call for an availability of six 9s. This represents 30 seconds of downtime per year or 2.5 minutes of downtime every five years (our assumed mean time between failures, or MTBF). Clearly, you need a second system to back up your active production system.

Active/Backup Systems



An active/backup system comprises two nodes – a production node and a backup node. The production node normally runs the application. The backup node may be performing other work, such as that used for development. Should the production node fail, application processing is moved to the backup node. This is called *failover.*

Will an active/backup system provide the availability required by the new SLA? Your applications are still up if the production node fails because the backup takes over. However, your applications are down if both nodes fail.

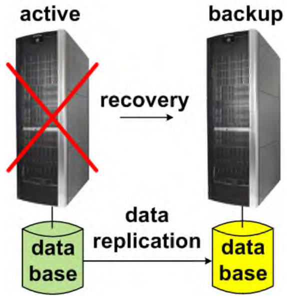


What is the probability of a dual-node failure? The probability that one node with four 9s availability will fail is[1] (1 – 0.9999) = 0.0001 = 10^{-4}. The probability that both nodes will fail is 10^{-4} x 10^{-4} = 10^{-8}. Thus, the redundant system has an availability of eight 9s, easily meeting your new SLA. Right?

Think again. You can achieve eight 9s if the backup node can take over instantly. But it can't!

Recovery Time

It takes a while for a backup node to take over processing. This is called *recovery time*. During the recovery, applications are down. Downtime becomes greater; availability becomes less. It is therefore imperative to reduce recovery time.



What must be done to bring up the backup node? Let's assume that the backup database has been kept up-to-date via data replication (tape backup can take hours to days to restore the database). The first step is to decide whether or not to fail over. What caused the production node to fail? Is it better to wait for it to get repaired, or will it be faster to bring the backup node online? This often requires a management decision and adds to the recovery time.

If the decision is made to fail over:

- Any work that the backup node is doing (such as development) has to be shut down.

[1] Note that the exponent of the failure probability is the number of 9s of availability.

- Applications have to be loaded.
- Networks have to be reconfigured.
- The database has to be mounted.
- The new production node has to be tested.

The above procedures describe a *cold standby*. Typical cold-standby recovery time is one to three hours.

Recovery time can be minimized by:

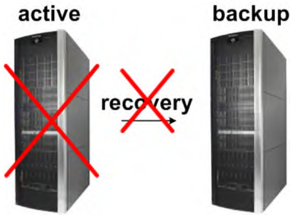
- preloading the applications of the backup node (this is called a *hot standby*).
- scripting backup procedures to minimize manual efforts (and errors).
- ensuring that the backup database is current and consistent (use a suitable data replication engine).
- practice, practice, practice.

By using these techniques, recovery time typically can be reduced to anywhere from ten minutes to one hour.

Failover Faults

However, one other factor that affects the availability of an active/backup system is *failover faults*. A failover fault occurs if the backup node cannot be brought online. In this case, we have a dual-node failure.

There are many causes of failover faults:



- Backup failure – the backup node has failed, and no one noticed.
- Configuration drift – changes made to the active node did not make it to the backup node.
- Script error – a failover script was wrong or out-of-date.
- Operator error – an error was made in a manual part of the failover procedure.

Failover faults can be minimized by periodic testing. However, many companies consider this to be a risky and expensive procedure. Applications are down during failover testing. What if things go wrong, and the active node cannot be brought back online? As a consequence, failover testing is typically done during off-hours. In addition, the availability of all senior staff must be ensured in case the failover fails.

Because of these factors, failover testing is often not thoroughly performed. Some companies never test failover – they depend upon faith and hope.[2] Without periodic testing, failover faults are all too likely to occur and represent a serious impediment to high availability.

An Availability Analysis

Let us analyze the impact of recovery time and failover faults on availability. We are going to use a little math, but we will keep it to a minimum. If you are mathematically challenged, ignore the math. You will still be able to understand the results.

In an active/backup system, there are three ways in which an application can be down:

- Both the active and backup nodes have failed.
- The applications are in the process of being recovered on the backup node.
- A failover fault has occurred.

[2] An excellent counterexample is Mayo Clinic. On a quarterly basis, Mayo fails over and runs on the alternate system until the next failover time (switch-and-stay, a good best practice). As a result, Mayo achieves a recovery time of fifteen minutes and has virtually no failover faults. See Tackling Switchover Times, *Availability Digest*; October 2006 – http://www.availabilitydigest.com/public_articles/0101/tackling_switchover_times.pdf

Let us calculate the expected amount of downtime for an application. Let:

f be the probability of failure of a node.
mtbf be the mean (average) time between failures for a node.
mtr be the mean (average) time to recover to the backup.
d be the probability of a failover fault.

Case 1: Dual-Node Failure

The probability that one node will fail is *f*. The probability that both nodes will fail is $f \times f = f^2$:

probability of a dual node failure = f^2

In our example, a node has an availability of four 9s. Therefore, the probability of failure of a node is (1 – 0.9999) = 0.0001 = 10^{-4}; and the probability of a dual-node failure is $10^{-4} \times 10^{-4} = 10^{-8}$:

This represents an average downtime of 0.3 seconds per year or 1.5 seconds every five years (our assumed *mtbf*). Note that this is an average. With a nodal downtime of four hours, the system will be down for four hours every 72,000 years. Dual-node failures are not very significant.

Case 2: Recovery Time

In our example, the active node fails once every five years (*mtbf*). Applications will be down during the time it takes to recover to the backup node (*mtr*). Therefore, the probability that an application will be down during recovery to the backup node is *mtr* / *mtbf*:

probability of being down during recovery = *mtr* / *mtbf*

Let us consider a recovery time of thirty minutes. We will be down thirty minutes every five years while applications are recovering to the backup node.

Case 3: Failover Faults

In our example, the active node fails on the average of once every five years (*mtbf)*. The probability that the active node has failed is *f*. The probability that there will be a failover fault when the active node fails is *d*. Therefore, the probability of a failover fault is the probability that the active node will fail AND the probability that a failover fault will occur:

probability of a failover fault = *f* x *d*

In our example, the probability of the active node failing, *f*, is 10^{-4}. Let the probability of a failover fault following the failure of the active node be ten percent (0.1). One out of ten failovers will fail. This means that there will be a failover fault every fifty years on the average. (Note that this implies that effective failover testing has not been done.)

Thus, the probability of a failover fault is $10^{-4} \times 10^{-1} = 10^{-5}$. This is five 9s. Five 9s is a downtime of five minutes per year or fifteen minutes every five years.

Summary of the Analysis

To summarize the analysis, we have:

probability of a dual node failure = f^2
probability of being down during recovery = *mtr* / *mtbf*
probability of a failover fault = *f* x *d*

$$\text{probability of application downtime} = f^2 + (mtr \,/\, mtbf) + (f \times d)$$

To summarize our example:

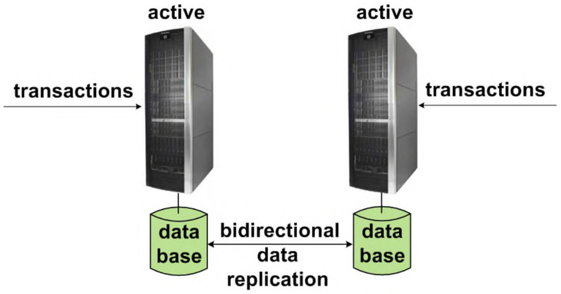
	per five years
downtime due to dual-node failures:	2.5 seconds
downtime due to recovery:	30 minutes
downtime due to failover faults:	15 minutes
total downtime	45 minutes

An average of 45 minutes of downtime every five years represents an availability of 0.99998.

We have reduced our average downtime from four hours every five years to 45 minutes every five years (good!). We have increased our availability from four 9s to almost five 9s (good!). We have missed our new SLA of six 9s (bad!).

So what can we do to meet our new SLA? The answer is an active/active system.

Active/Active Systems



An active/active system has two or more nodes. Every node is actively processing transactions. Every node has the same view of the application database. Whenever a node makes a change to its copy of the application database, that change is immediately replicated to the other nodes in the system.

If a node fails, all transactions are routed to the surviving node (or nodes). Recovery time, *mtr*, can be as little as several seconds - typically, new connections and sessions must be established with the surviving node(s). However, *mtr* can be effectively zero if clients maintain sessions with all nodes.

It is known that all nodes are operational; after all, they are actively processing transactions. Therefore, there will be no failover faults; and *d* is equal to zero.

Returning to our expression for application downtime:

$$\text{probability of application downtime} = f^2 + (mtr \,/\, mtbf) + (f \times d)$$

Let *mtr* be fifteen seconds (the application is down fifteen seconds every five years during recovery time). $d = 0$ (there are no failover faults). *f*, the nodal availability, is 10^{-4}. Then

$$f^2 = 10^{-8}$$
$$mtr \,/\, mtbf = 9 \times 10^{-8}$$
$$f^2 + mtr \,/\, mtbf = 10 \times 10^{-8} = 10^{-7}$$

This is an availability of seven 9s. Congratulations! We meet our availability of six 9s.

Another advantage of active/active systems is that there is no planned downtime. If nodal maintenance is required, simply shift all traffic to one node and upgrade the idle node. Then repeat this process for the other node.

Active/active systems provide true continuous availability. If a node fails, no one notices. If a data center blows up, no one notices.

Summary

The availability of an active/backup system is strongly affected by recovery times and failover faults. These factors are eliminated with active/active systems.

If you require application availability in excess of five 9s, consider an active/active architecture. Active/active systems minimize recovery times and eliminate failover faults. There are many examples of active/active systems that have been in service for decades without an outage.