the Availability Digest

www.availabilitydigest.com
@availabilitydig

Eliminate Those Single Points of Failure

October 2017

In order for a system to be continuously available, every component must be redundant. There can be no single point of failure. The failure of any component must be transparent to the users of the system.

This is often done by replicating servers and databases. For instance, in an active/active system, there are two or more servers actively processing transactions. A transaction can be sent to any server and be processed in exactly the same way it would be processed by another server. This implies that all servers have access to a common database. In some architectures, each server has its own database; and the servers are kept synchronized by bidirectional data replication. Whenever a server makes a change to its database, the change is immediately replicated to the databases of the other servers to keep them synchronized.

In other architectures, servers have access to multiple database arrays. When a server makes a database update, it updates all of the databases in the database array.

Oftentimes, even subcomponents of the servers and databases may be replicated to improve their reliability. For instance, it is common to find dual power supplies in a server or database so that the power supply does not become a single point of failure for one of these devices.

Not to be forgotten is the network interconnecting all of these devices. It must also provide two or more independent paths between each server and database so that there is no single point of failure in the network.

In previous Digest articles, I have referred to many systems that had a single point of failure that could take down the system. Here are some examples from these articles.

Anti-Virus – A Single Point of Failure?

What do active/active systems, clusters, fault-tolerant systems, and standby systems have in common? They all avoid a single point of failure.[1]

However, on April 21, 2010, McAfee proved this conjecture wrong. It sent out an antivirus update that immediately took down hundreds of thousands of computers worldwide. Worse still, the bad update required manual intervention on every individual computer to restore it to service, taking those data centers with thousands of Windows servers offline for hours and, in some cases, for days.

[1] Anti-Virus – A Single Point of Failure?, *Availability Digest*; May, 2010. http://www.availabilitydigest.com/public_articles/0505/mcafee.pdf

Windows Azure Downed by Single Point of Failure

The Windows Azure cloud went down when Microsoft made an update to RDFE, its Red Dog Front End. RDFE disperses user requests to the cloud's virtual machines.[2]

The problem arose when Microsoft made an update to RDFE. The change was tested on a small number of nodes within a single cluster and worked perfectly. The Azure developers then pushed the change out to all of the data centers worldwide, and that is when the problem exposed itself. Though existing applications continued to work, new applications could not be deployed in any of the data centers.

Due to the way that Azure is built, there can only be one RDFE in the entire cloud. Therefore, the RDFE is a single point of failure. The developers could not run one instance of an RDFE and then deploy it to other instances only after it had proven itself in production.

Boeing 787 Could Lose All Power

The Boeing 787 Dreamliner is a mid-size, wide-body replacement for the Boeing 767 and seats up to 335 passengers.[3] With the 787's heavy dependence on its electrical system, Boeing went to extraordinary measures to add redundancy so that the continuous availability of electrical power was assured. The plane uses six electrical generators. However, all the redundancy in the world is useless if the design includes a single point of failure. In the case of the 787, the single point of failure was in the software in the generator control unit.

In lab testing years after its first delivery of 787s, Boeing discovered a software error in this control unit. The error could result in a total loss of electrical power to the aircraft, even in flight. The condition occurred if electrical power were left on for about eight months without being turned off. More specifically, the problem became existent on the 248th day of continuous electrical power.

The single point of failure was caused by a software counter that would overflow after 248 days of continuous power. Fortunately, none of the operators of 787s left power on for that long.

Single Point of Failure Causes Verizon Outage

A stalled Verizon router interrupted DSL and fiber Internet service to much of New York and Massachusetts for an hour or more on October 5, 2010.[4] Though the router was part of a redundant network, it failed in such a way that it appeared to be operable to neighboring routers. Therefore, other routers continued to send traffic to it, which the stalled router simply dropped. Likewise, the backup router thought that the stalled router was still operational and did not take over. Ultimately, the offending router was rebooted; and service was restored.

Microsoft's Azure Cloud Goes Down – Again

This Azure outage was caused by two factors - a faulty upgrade and an improper deployment.

The upgrade was being made by Microsoft to its Azure Storage service. It had been extensively tested for several weeks. Unfortunately, a decision was made to roll out the configuration change to all regions

[2] Windows Azure Downed by Single Point of Failure, *Availability Digest*; November 2013.
http://www.availabilitydigest.com/public_articles/0811/azure.pdf
[3] Boeing 787 Could Lose All Power, *Availability Digest*, June 2015.
http://www.availabilitydigest.com/public_articles/1006/787_power_loss.pdf
[4] More Never Agains 4, *Availability Digest*; February 2010.
http://www.availabilitydigest.com/public_articles/0502/more_never_agains_4.pdf

simultaneously. Microsoft's standard protocol was not followed, which specified that changes should be rolled out in incremental batches. Thus, all regions received the update within a short period of time.

As soon as some of the regions came online with the Azure Storage performance upgrade, problems arose. The Storage blob front-ends went into an infinite loop and could take on no further traffic. This issue had gone undetected during the testing of the update. Other services built on top of blob storage began to experience outages.

By the time Microsoft staff had identified the problem, twelve of its nineteen regions were affected. The Azure cloud went down around most of the globe.

People Can Be a Single Point of Failure

We spend a lot of money on making our data centers redundant so that they will survive one or more failures. However, we seem to pay no attention to an equally important human component that needs redundancy – the fat finger. There were an abundance of failures that were caused by a staff member's inappropriate action:

- A State of Virginia technician pulled the wrong controller and crashed a redundant SAN that already had suffered a controller failure.
- A technician with DBS Bank made an unauthorized repair on a redundant SAN and took down both sides.
- A system operator mistakenly deleted the $38 billion Alaska Permanent Fund database and then deleted its backup.
- A maintenance contractor's mistake shut down the Oakland Air Traffic Control Center.
- Thirteen million German web sites went dark when an operator mistakenly uploaded an empty zone file.
- A test technician failed to disable a fire alarm actuator prior to testing the fire suppression system. The resulting siren noise damaged several disks, including the virtual backup disks.
- A system administrator closed all applications on one server in an active/active pair to upgrade it and then shut down the operating server.

All of these outages could have been prevented if a second pair of eyes were checking the actions of the staff member. The lesson is that any critical operation that could have a negative impact on operations should be checked by a second person prior to initiation. Let one person create the command or point to the board to be pulled, and have a second person confirm the action prior to executing it. This is especially important following a component failure that leaves the system with a single point of failure.

Summary

Single points of failure can spell disaster for a system. Every component in a system – including humans – need to be redundant if we are to achieve high availability. In that way, if a component fails, the system can continue to operate.

If some component is a single point of failure, should that component fail, the system will fail. We have relinquished any hope of achieving high availability.