the Availability Digest

www.availabilitydigest.com

Help! My Data Center is Down!

Part 6 – The Human Factor

March 2012

In many respects, a company's data center is part of its life blood. Significant investments are made to ensure that corporate data centers never fail. Unfortunately, they do.

Industry studies have shown that the human factor plays a role in about 70% of data-center failures. In some cases, it is a careless error on the part of an operator. In others, it is out-and-out malfeasance. Often, an otherwise controllable situation caused by some hardware or software fault is elevated to a full crisis by a human action.

In our previous articles on data-center failures, we focused on failures due to power, storage subsystems, network faults, and upgrades gone wrong. In this article, we look at some human contributions to data-center outages.[1] The stories are all true and are taken from the Never Again archives of the *Availability Digest*.

Carelessness

A good many outages are caused by careless operator errors that occur with a perfectly good operational system.

The Coffee Pot Fiasco

A particularly amusing incident (though not so at the time) was when a coffee pot took down a small data center. After running its active/active network successfully for several years on its existing equipment, the company decided to upgrade to the next version of the system that it was using for its two nodes. This was a major upgrade involving new hardware.

As best practices dictate, the system at each node was powered by a separate circuit protected by an uninterruptible power supply (UPS). When the new system was rolled in at one of the nodes, the operations staff found that all of the UPS power connectors were being used. So as not to delay the upgrade, the new system was temporarily connected to the facility's unprotected power. The plan was to correct this problem in short order by adding an additional connector to the UPS output.

However, the required power connector change was forgotten. As time went on, the load on the unprotected circuits gradually increased as the company grew. One fateful day, an employee performed a normal, everyday task. He or she plugged in the coffee pot to make fresh coffee. This was the straw that broke the camel's back. The coffee pot blew the circuit breaker, taking down everything that was on that

[1] Our thanks to The Connection for giving us permission to reprint this series of articles.

circuit. This included dropping primary power to the upgraded system, which had never been moved to the UPS circuit.

The system kept on running for a while on its internal UPS. Fast action on the part of the staff at the site restored the primary power in just 35 seconds – an admirable feat. Unfortunately, the system's internal UPS only lasted for 30 seconds. The node shut down and suffered a 30-minute outage until it was brought back online. Though this was a major fault, the fact that the applications were running active/active meant that the other system immediately assumed all of the load; and users were not affected.

The Case of the Flying Cable

The company's fault-tolerant system depended upon communications with the outside world. Therefore, its communication interface was totally redundant. It had dual communication processors connected to dual LANs driving redundant communication lines. Each set of equipment was powered by an independent external power source.

The communication subsystem hadn't failed for over a decade and seemed solid. Not quite! One day, a technician was pulling cables through the false flooring when a cable end came loose and went flying. It collided with the power strip supplying one side of the redundant network equipment. The on/off switches on the power strips had been disabled so that they would not be accidentally turned off. However, the recessed circuit breakers were active for safety reasons.

As luck would have it, the corner of the cable went into the recess of one of the circuit breakers and tripped it. But no problem because the other side of the communications system would carry on, right? Wrong. Long ago, with no one noticing, all of the communication equipment had been plugged into only one power strip; and that is the one that got hit. That ended a decade of perfect availability.

Ignoring a Software Bug Causes Train Wreck

An international long-haul railroad uses a fault-tolerant system to track its trains. Controllers monitor train movements and can control the trains by positioning switches and changing signal states.

The system had been in operation for over a decade with its normal set of problems, all of which had been corrected except for one known problem. There was one particular set of parallel tracks, one northbound and one southbound, that would lose the southbound train display if trains were on each set of tracks. The train controllers were well aware of this problem but had never reported it. Instead, they just remembered when a train had been lost. Since traffic was light, this hardly ever happened.

But one day it did happen while the controller's attention was diverted. When he returned to his console, he saw that the track was unoccupied; and he cleared the next train onto that track. Unfortunately, the track was not unoccupied – the ghost train was still there.

Also, unfortunately, at the same time the engineer of the trailing train was "otherwise occupied" and did not see the train in front of him. The resulting crash caused significant damage to the trains and the track. Fortunately no one was injured because of the low speed of the collision.

Console Command Takes Down Active/Active System

You have to work hard to take down an active/active system. However, one way to do this is for an operator to erroneously enter a series of commands that adversely affect all systems in the network.

Just such an incident happened to a two-node active/active system that had run for years without an outage. In fact, the system had undergone many rolling upgrades without a planned outage of any sort.

The upgrade started normally. The operations staff moved the users off of Node A in preparation for upgrading it. Once satisfied that all users were now properly being handled by Node B, the system manager brought up the maintenance console for Node A; and the command to stop the Node A system was entered.

To the system manager's horror, the entire system suddenly shut down. As it turned out, he had not brought up the maintenance console for Node A. He had brought up the maintenance console for Node B. Oops! He had shut down the wrong node. Consequently, he had stopped the operational system; and all users were out of service.

The $38 Billion Keystroke

To protect oil revenues for the citizens of the state of Alaska, the Alaska Permanent Fund was set up to receive and invest proceeds from the sale of Alaskan oil. The fund has paid a yearly dividend to all Alaskan residents. In 2006, the fund balance had reached $38 billion.

On a fateful day in July, 2006, a computer technician working on a disk drive at the Department of Revenue mistakenly deleted the oil fund database. This was not a big problem because the data also existed on a redundant backup disk. However, under the pressure of the moment, the technician also managed to reformat the backup disk.

Not to despair. Like all good data centers, this data was backed up on magnetic tape. The only data that would be lost would be those transactions entered since the last update. The tapes were retrieved from storage. Only then did the magnitude of the disaster become apparent. The tapes were unreadable. The triple redundancy that was built into the system was not enough.

Over the next several days, employees and consultants tried vainly to salvage the data. The terrible truth finally had to be accepted. The last nine months of transaction history had been lost. This included 800,000 scanned images of paper applications. Fortunately, there was a fourth level of backup – the paper documents themselves, stored in over 300 cardboard boxes. Each of the 800,000 documents had to be rescanned, sent through quality control, written to the database, and linked to the appropriate person's account.

It took 70 people working nights and weekends almost two months to complete the recovery at a cost to the state of $200,000.

Data Center Taken Down by Noise

It's not a good idea to test a fire-suppression system by triggering it. But that's what happened to WestHost, a major web-hosting provider headquartered in Utah. On Saturday, February 20, 2010, the WestHost data center underwent a standard yearly test of its Inergen fire-suppression system. Unfortunately, a third-party test technician failed to follow the published pre-test check list and did not remove one of the actuators that activates the system. When the system was re-armed following the test, the actuator fired and triggered the release of the large blast of Inergen gas designed to put a fire out.

No one seemed to know at the time whether it was the pressure blast or the gas itself, but hundreds of servers and disk storage systems were severely damaged. Even worse, the backup disk drives were in the same facility as the servers, and many of the backup disks were destroyed. Some RAID drives were recoverable, and their servers were brought back into service. Though data recovery experts were able to restore data from some failed drives, other data was simply deemed nonrecoverable by the data-restoration experts.

It took six days for WestHost to get its data center back into operation. Subsequent studies by the manufacturers of the fire suppression system and of the Inergen gas showed that the disk damage was caused by the ear-splitting sound level from the warning alarms.

Malfeasance

Data centers have been taken down by the actions of disgruntled employees and by external hackers.

Blogging Site Put Out of Business by Disgruntled Employee

Started in 2003, JournalSpace was a popular and growing blog-hosting service. It was primarily a free service supported by advertising. Unfortunately, in December, 2008, thousands of blogs were wiped out when JournalSpace lost its entire database and was unable to recover.

Apparently, the database's demise was the malicious act of a disgruntled employee – even worse, the IT manager. JournalSpace claims that it had caught the IT manager stealing from the company. They summarily fired him; but he did a slash-and-burn on his way out, overwriting the entire database with garbage.

This should have been only a minor irritant because all that was needed to cure the problem was to restore the database from the backup copy. The problem? No backup copy! It was, of course, the IT manager's responsibility to ensure that a backup copy was periodically taken and preserved. However, though he dutifully backed up the HTML code for the site on a remote server, his backup strategy for the blog database was to use a RAID 2 mirrored disk. If one disk failed, the database was still available on the mirror.

In a panic, JournalSpace management sent the hard disks to a service known for recovering data from burnt, drowned, and crushed hard drives. Unfortunately, the answer was ultimately "no." The disks were unrecoverable. They had been overwritten with random data, obliterating the original data.

JournalSpace closed its doors.

Sony PlayStation Taken Down for Weeks by Hackers

This may be the biggest hacking story in history. From April 16th to April 18th, 2011, hackers gained access to Sony's online gaming servers and stole sensitive personal information for over 100 million accounts. Sony did not discover the breach until April 19th. It promptly closed down its online gaming services until it could restructure its security defenses.

Over 77 million PlayStation accounts were compromised. Sony announced that the enhancements it needed to beef up its security protection services would be time-consuming. It predicted that some services would be back online during the first week in May.

Then on May 1st, the magnitude of the disaster was discovered to be worse than originally thought. The investigating team found that the mid-April attacks had also similarly compromised the 25 million subscriber accounts for Sony Online Entertainment. The company shut down the SOE services the next day and ceased making predictions as to when services would be restored. Services were gradually restored through mid-May. Sony's gaming services had been down for several weeks!

Though Sony is teaming with the FBI (the U.S. Federal Bureau of Investigation) and private investigators, the perpetrators have yet to be identified. However, there is evidence that points to a hacking group named "Anonymous" that wanted to get revenge for Sony's "unfair legal actions" against a well-known hacker who had managed to find and publish the secret keys to Sony's online games.

Twitter Taken Down by DDoS Attack

On Thursday, August 6, 2009, Twitter suddenly became unavailable to those trying to use it. During that day and much of the next, Twitter was down for a few hours, would seem to recover but would be

sluggish or subject to timeouts, and then would go down again. Continuous periods of outages and timeouts continued well into the next day.

It didn't take long for Twitter to conclude that it seemed to be a target of a distributed denial of service attack (DDoS), in which its servers were being swamped by spam messages. The spam messages were all queries against the blog of a single user who went by the user name Cyxymu. Clearly, someone was out to silence Cyxymu. But why?

It turned out that Cyxymu was a pro-Georgian blogger, a 34-year old economics lecturer from Tbilisi, Georgia, who had been criticizing Russia's conduct in its war the previous year over the disputed South Ossetia region. Cyxymu is the name of a town in the former Soviet Union.

In later posts, Cyxymu blamed Russia for the attack. He suggested that the timing of the attack was meant to silence him on the eve of the one-year anniversary of the Russian attack on Georgia.

Crisis Escalation

Many of the outages that we described in earlier parts of this series were hardware or software faults that should not have caused a serious incident. However, the fat fingers of humanity intervened to create serious crises. Here we briefly list some of them. The details can be found in our earlier parts.

Power Outages

- Google administration staff failed to complete failover documentation procedures that took down Google Apps for several hours.
- Maintenance staff failed to cut down a tree that triggered the great Northeast blackout.

Storage Outages

- A State of Virginia maintenance technician pulled the wrong controller board on a redundant SAN and took down 26 state agencies for a week.
- Déjà vu. A maintenance technician tried an unauthorized maintenance procedure of the redundant storage area network used by DBS, the largest bank in Southeast Asia, and took down all online banking services and its ATM and POS network for ten hours.
- American Eagle failed to test its tape recovery procedures as well as its backup data center and lost online sales for four days.
- A major Asia Pacific bank attempted to solve a database corruption problem by patching all three of its systems simultaneously, It succeeded in replicating the corruption to all three systems. It took months to untangle the mess.
- A major collections company failed to test its backup system and spent three weeks recovering from a RAID failure.

Networking Problems

- A Chinese purveyor of illicit game copies launched a DDoS attack on several competitors. The attack went awry and took down Chinese Internet services for hours.
- Over thirteen million German web sites became inaccessible for almost two hours when the German Internet authority uploaded new zone files that were empty. This meant that all web sites in those zones could not be reached, and email was rejected.
- A maintenance subcontractor's mistake shut down the Oakland Air Traffic Control center when the redundant system also failed. Controllers had to rely on cell phones to coordinate flights.

Upgrades Gone Wrong

- Every failed upgrade that we reported was caused by improper planning or testing by staff members.

And then there were all of the severed communication cables . . .

- Vandals cut communication cables in Silicon Valley and took down Internet, telephone, and wireless services for twelve hours.
- Communication services were interrupted for two days when a construction crew digging trenches for a new sewer system in downtown Las Vegas severed a conduit carrying several copper and fiber cables.
- Ditto for London users when contractors working on the Olympic site sent a large-thrust borer right through a deep BT tunnel, severing multiple fiber cables. Internet and other communication services were shut down for tens of thousands of customers.
- Ditto again when a construction saw severed 144 fiber cables in Wall Street, terminating 60 million connections.
- A contractor drilling test holes in Sydney's city center severed cables containing 10,000 communication lines. It required a week to return full service to all of the subscribers in the area.
- A fiber-optic cable was cut by workers laying a pipe for Australia's Queensland water grid, collapsing the communication network for over four hours. Communications were abruptly terminated for more than a million customers when rerouting failed.
- A 75-year old lady in the country of Georgia was digging for copper cable to sell on the black market when she dug up an optical fiber cable. The damage cut off Internet access to most of Azerbaijan and Georgia for half a day.

Summary

Data center outages are caused by many factors, but the human element is dominant among them. Not only can staff errors directly cause outages, but even worse, they can escalate a controllable problem into a major crisis. One would think that staff problems are the one area that we can effectively control. Evidently, that is not the case.

In our next and last part of this series, we will review what lessons we can learn from these failures. The past is bound to be repeated if we don't learn from our mistakes.