a laptop and you can stand to be without the system for a few days if it dies, then you don't really need to worry about developing a fault tolerant solution. On the other hand, if the data is the key to running your business and you can't survive without it, then a fault tolerant plan is required.

Fault tolerant systems help protect against single points of failure. There are different variants of fault tolerant plans that can be implemented, from full-time, always-on solutions to offline backups that might take a day or two to restore. I'll explore some of those concepts after discussing contingency planning.

Contingency Plans

Business continuity concepts should all be part of a company's *contingency plan*, which is simply a plan for action in the event of a problem. Having a plan in place will help drive the appropriate actions when a problem happens. Without a plan, chaos can ensue.

Contingency plans will vary by company, but in general, here are some steps to take to create one:

Step 1: Perform a business impact analysis. The first thing to understand is what impact, if any, failures of components or loss of data would have on the business. If there is no impact, then there's really no need for a contingency plan, but at least you're making an informed decision. This should be done for all critical systems, including computers and infrastructure.

Step 2: Identify preventive systems. What systems does the company have in place to prevent issues from happening? Start with environmental things such as smoke detectors and sprinkler systems as well as security systems to control access. Then move on to more specific computer-related items such as power backups and security policies.

Step 3: Develop a recovery plan. In the event of a problem, what is the plan? Specifically understand who is responsible for specific actions and what is the chain of authority. Recovery plans need to account for all types of problems, such as data loss from hackers, equipment failure, and natural disasters.

Step 4: Test the recovery plan. Just because the plan is created doesn't mean it will work properly! Test it in live simulations including the people involved. The testing phase can help identify potential issues and inspire revisions to the plan.

Step 5: Set up a maintenance and review schedule. Plans need to adapt over time based on changing circumstances. Maybe a company added five more servers and 50 more clients; the plans need to change. Review as often as needed and make appropriate changes. Also test the preventive equipment identified in step 2.

Step 6: Implement training. Training is critical to ensure that the first responders know what to do and the people that follow after them understand their roles as well. Delivering periodic training should be incorporated into the maintenance and review schedule.

No plan is entirely foolproof, but having a plan will help things run smoothly if and when an event happens.

Replication and Redundancy

Replication is what it sounds like—it's a full working copy of whatever data, computer, or network that's being considered. If you're talking about replicating data, it could mean that the same data is written to two hard drives, databases, or servers at the same time. On a larger scale, some organizations have entire networks replicated (called *hot sites*) in case of failure. If not having any fault tolerance plan is at one end of the risk aversion spectrum, replication is at the other. It leaves no room for any failures to impact data or systems. The downside to replication is that it's the most expensive of the fault tolerant options.

One step down from replication is redundancy. With *redundancy*, there are devices in place to help keep things running normally for a short period of time, until the original problem can be fixed. Redundancy can be implemented on data, networks, and power sources.

Data Redundancy

If having one hard drive is good, then having two hard drives is better. Perhaps you need additional storage space, so you add another drive. There's nothing wrong with doing that, but what happens if one (or both) of your hard drives fails? Having that extra drive didn't help. In fact, it added another potential point of failure for your computer.

There are ways that you can add additional hard drives to a computer and get benefits beyond increased storage capacity. You can make your disk reads/writes a little faster, and you can also create fault tolerance by having extra protection against disk failures. You do this by implementing *RAID*.

RAID stands for *redundant array of independent disks*, which is multiple physical hard disks working together as a team for increased performance, increased reliability, or both. There are more than 10 different implementations of RAID. The three most popular (and important) versions are as follows:

RAID 0 Also known as *disk striping*, RAID 0 is where at least two drives are combined to create one logical volume. Equal amounts of space are used on each drive, and data is written across the volume like a stripe. *RAID 0* is not RAID in every sense because it doesn't provide the fault tolerance implied by the *redundant* component of the name. Data is written across multiple drives, so you essentially have two drives reading or writing the same data set at once. This makes for faster data access. However, if any one of the drives fails, all content is lost. To protect data, an additional form of redundancy or fault tolerance should be used in concert with RAID 0.

RAID 1 Also known as *disk mirroring*, *RAID 1* is a method of producing fault tolerance by writing the same data simultaneously to two separate drives. If one drive fails, the other contains all of the data and will become the primary drive. However, disk mirroring doesn't help access speed, and the cost is double that of a single drive.

RAID 5 Combines the benefits of RAID 0 and RAID 1, creating a redundant striped volume set. Unlike RAID 1, however, *RAID 5* does not employ mirroring for redundancy.

Several years ago, one of my former students related a story to me about a server crash at his company. A server had mysteriously died over the weekend, and the technicians were greeted with the problem first thing Monday morning. Not to worry, they thought, because they made regular backups.

After several attempts to restore the backup tape, a second, more serious problem was readily apparent. The backup didn't work. They couldn't read the data from the tape, and it was the only backup tape they had. It wasn't going to be a very good Monday. Ultimately, they ended up losing extensive data from the server because their backup didn't work.

How do you prevent tragedies like this from happening? Test your backups. After you make a backup, ensure that you can read from it. If you've just backed up a small amount of data, restore it to an alternate location and make sure that you can read it. If you are backing up entire computers, a good idea is to run a test restore on a separate computer. No matter what your method, test your backup, especially when it's the first one you've made after setting up backups or you have made backup configuration changes. It isn't necessary to test each single backup fully after that, but it is a good idea to spot-check backups on occasion.

Here are two more ideas that will help if you back up locally:

1. If you use discs, rotate backup discs. Alternate discs every other backup period, or use a separate disc for each day of the week. This lessens the risk of having a bad disc bring you down.
2. Store your backups off-site. If your backup is sitting on top of the server and you have a fire that destroys the building, then your backup won't do you any good. There are data archiving firms that will, for a small fee, come and pick up your backup tapes and store them in their secure location.

Be vigilant about backing up your data, and in the event of a failure, you'll be back up and running in short order.

Disaster Recovery

If an event such as data loss or equipment failure does happen, restoring access will be the most important task to complete. The *disaster recovery* plan should be a key component to the contingency plan, which will identify the priorities to take in the event a disaster strikes.

First, if a disaster does strike, ensure that it's under control before beginning to recover from it. For example, while it might sound silly to say it, make sure that the fire is out before you begin restoring access to systems. That might be kind of obvious, but for some situations, such as hackers attacking a network, it can be a little subtler to understand when everything is under control.

When recovering from a disaster, follow the priorities as outlined in the contingency plan. Usually, you work from the biggest stuff to the smallest. For example, restoring access to servers would take priority over fixing someone's email client. And, of course, hardware needs to be in good working order before data restoration can begin.

Exploring Computer Support Concepts

Computer support isn't typically thought of as one of the sexy jobs in IT, but it's definitely one of the most critical and visible jobs. Users don't care how elegant your network design is, how bulletproof your security model is, or how user-friendly your website is if their computer won't boot up. And when it won't boot up, their best friend is the computer support person.

Sometimes you will fix computers in person, while at other times you'll deal with challenging situations over the phone. In either case, you need to be prepared to diagnose the problem quickly and implement a solution. The user will be counting on you—and you get to be the hero! Okay, so maybe the user won't treat you like a hero and there probably won't be a parade in your honor, but you will definitely know that they appreciate your help.

Computer support is synonymous with troubleshooting. To troubleshoot well, you need to understand the theory and process of troubleshooting. This includes the basic concepts and resources that apply in most situations. In addition, there are common scenarios that you will encounter—knowing how to solve the common issues will help you think about how to solve the difficult ones as well.

Understanding Troubleshooting Theory

When troubleshooting, you should assess every problem systematically and try to isolate the root cause. Yes, there is a lot of art to troubleshooting, and experience plays a part too. But regardless of how "artful" or experienced you are, haphazard troubleshooting is doomed to fail. Conversely, even technicians with limited experience can be effective troubleshooters if they stick to the principles. The major key is to start with the issue and whittle away at it until you can get down to the point where you can pinpoint the problem—this often means eliminating, or verifying, the obvious.

Although everyone approaches troubleshooting from a different perspective, a few things should remain constant. First, always back up your data before making any changes to a system. Hardware components can be replaced, but data often can't be. For that reason, always be vigilant about making data backups.

Second, establish priorities—one user being unable to print to the printer of their choice isn't as important as a floor full of accountants unable to run payroll. Prioritize every job and escalate it (or de-escalate it) as needed.

Third, but perhaps most important, document everything—not just that there was a problem but also the solution you found, the actions you tried, and the outcomes of each.