5
Network Topologies - Mapping It All Out

Which network topology should you use for a small business? What about for a large enterprise? Unfortunately, there is no *one-size-fits-all* approach to networks. There is too much disparity between the types of equipment each organization has. A large business may update their hardware on a regular basis, thereby keeping up with newer technology, whereas a smaller business may not update theirs for 5 to 10 years. With the speed at which technology develops, there will be vast differences in that time frame.

This chapter will describe common network topologies in use today and when they can be used. It highlights the advantages and disadvantages of each topology. It is important for all network engineers to understand these differences to ensure that they select the appropriate topology for their purposes, and be able to troubleshoot each.

The following topics will be covered in this chapter:

- Logical topologies versus physical topologies
- Bus topology
- Ring topology
- Star topology
- Mesh topology
- Hybrid topology

Logical versus physical topology

At this point, you might be wondering what a topology actually is, so I think it is probably best that we clear that up first and foremost. In relation to networks, a topology can be thought of as a map that details how the network fits together and how the data travels. Topologies can be classed as either physical or logical. A physical topology describes how the devices are connected together, whereas a logical topology describes how the data travels from device to device. This can be quite difficult for people to understand, so what I would like you to do is think about getting to work, home, or the local shopping center. My journey is like this:

1. Cycle to the local train station.
2. Train from the local train station to the destination station.
3. Tram from the destination train station to the tram stop close to work.
4. Walk from the tram stop to the office.

In essence, I have summarized my journey into a logical topology. I haven't described the journey in full, by telling you every turn that I take. That would be a physical topology. To confuse things further, quite often the physical and logical topology are the same. Do not worry too much if you are still a little confused; as we go through this chapter I will provide you with a great example that should reinforce this concept fully; however, the following two activities will also be of benefit to your understanding.

Activity 1: Choose a destination such as work or college, and create a logical topology of how you would go from home to that destination. This activity works better if you change modes of transport.

Activity 2: In this activity, you will create a physical topology for the preceding journey:

1. Go to Google Maps.
2. Search for your destination.
3. Choose **Directions** and plot a route from your home to your destination.
4. The output is your physical topology.

Having a good understanding of topologies will become extremely beneficial to you as you progress through your networking career, particularly when troubleshooting. In the following sections, we will discuss the common topologies in use, and highlight the advantages and disadvantages of each.

Bus topology

For clarity I'm going to draw the bus topology in a way that in some implementations could be classed as slightly inaccurate, but I will explain why afterward. A bus topology can generally be described as a backbone cable with devices connected directly to it, as shown in *Figure 5.1*:

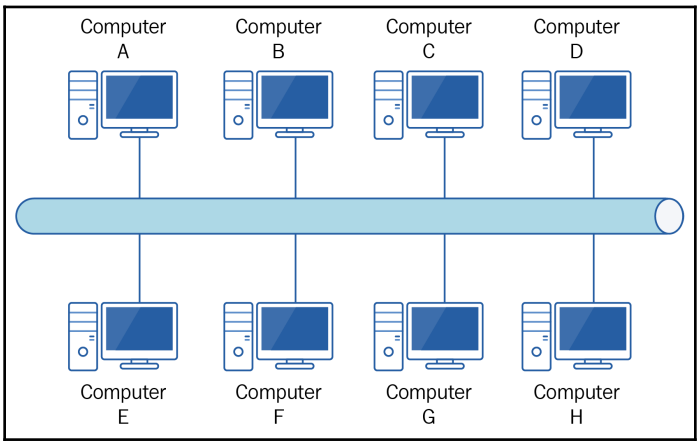

Figure 5.1: Bus topology

In the preceding diagram, I want to draw your attention to two things. Firstly, the backbone is quite clearly one single cable, and in some cases this would be correct, but in some others you might find it is made up of a number of shorter cables. Secondly, the various computers connecting to the backbone are shown as doing so through some form of intermediary cable. Again, in some instances this would be correct, and in others the devices would form part of the backbone and connect those shorter cables I mentioned in the first point.

In `Chapter 8`, *Media Types - Connecting Everything Together*, I will describe the two types of cable and their connectors that will dictate whether it's one long cable or a number of shorter cables. However, I will mention one characteristic of the cabling here. When the signal reaches the end of the cable, it will *bounce* back along the cable. To avoid this, each end of the cable will be fitted with a device called a terminator, which kills the signal and prevents this bounce.

At this stage, I would like to pause and talk about the access method used in a bus topology. This is deliberate as it will help you understand some of the advantages and disadvantages we will cover shortly. Recall that Wi-Fi uses an access method called **Carrier Sense Multiple Access/Collision Avoidance** (**CSMA/CA**); however, the Ethernet standard (IEEE 802.3) uses a similarly named access method called **carrier sense multiple access/collision detection** (**CSMA/CD**), which is used for a number of wired connections.

I tend to think of the last **A** in **CSMA/CA** as standing for **airwaves**. It's not a correct interpretation, but it helps me remember that CSMA/CA is used on Wi-Fi as the data travels the airwaves.

The first part of the CSMA/CD process is identical to CSMA/CA in that any device wishing to transmit on the network has to listen out for a gap in the traffic (carrier sense). If no device is talking the device will transmit its data; if a device is already transmitting, the device wishing to send data will wait a random amount of *back off* time, before repeating the process.

Let's revisit *Figure 5.1,* and update it. In the following example (*Figure 5.2*), **Computer B** is wanting to simply send some data to its neighbor **Computer C**. The arrow lines indicate the route of the data sent from **Computer B**. What do you notice about it? If you said that the data goes to all the devices and not just **Computer C**, you would be correct. There is no **Satellite Navigation** (**SatNav**) on the network that tells the data when it reaches the backbone cable to turn left, turn right, or go straight on. Herein lies one of the problems of a bus topology. It uses half-duplex transmissions and forms one big collision domain. If **Computer B** is transmitting, no one else can transmit, so it is not the most efficient method available.

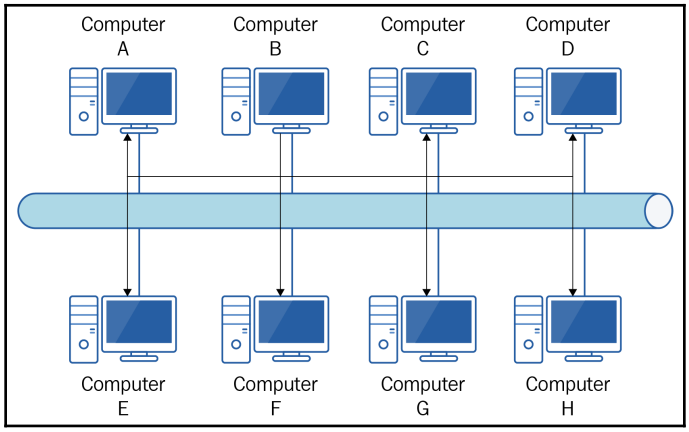

Figure 5.2: Data transmission on a bus topology

I'm going to return to the CSMA/CD process, and talk about what happens when two devices talk at the same time. In this scenario, both devices have listened to the network, and cannot hear anyone talking, so they both decide to send their data at the same time. You have probably had this yourself in normal conversations where no one is talking then all of a sudden two people break the silence at the same time. However, in CSMA/CD there is no polite *you first* option as such.

When two devices transmit at the same time, a collision occurs, and is detected. This detection is picked up through a change in amperage on the cable, and the transmitting devices send a jamming signal that tells all the other devices not to transmit. Both transmitting devices then *back off* for a random period of time, before repeating the transmission process. Once they have transmitted, normal service is resumed on the network.

Advantages

The relative simplicity of a bus topology means it is well suited for a small network, and can be extended with minimal effort. A bus topology also offers some resilience. As you see later, in some topologies a failure of one node or its connection can bring down a whole network. Look back at *Figure 5.2*: if that connection between **Computer B** and the backbone breaks, it only impacts communications to and from that device. A bus topology is also relatively cheap to implement, as it uses a minimal amount of cabling.

Disadvantages

I have mentioned already that a bus topology is great for a small network; however, as your network grows it becomes less and less efficient. Therefore, any company growth needs to be factored in when planning a network, particularly if you are leaning toward implementing a bus topology. The more devices you add, the more collisions will occur. I would concede, however, it is unlikely in this day and age that you would implement a bus topology network.

Although this type of network is resilient to a single device failure, the backbone cable serves as a single point of failure. If some break occurs on that one cable, then the whole network is lost. This is further exacerbated by the fact that a bus topology can be quite difficult to troubleshoot.

Ring topology

In a ring topology, each device is connected to two devices (*Figure 5.3*), and data is transferred by passing it on to the next device in the network. If the data is not for that device, it will forward it on to the next device and so on:

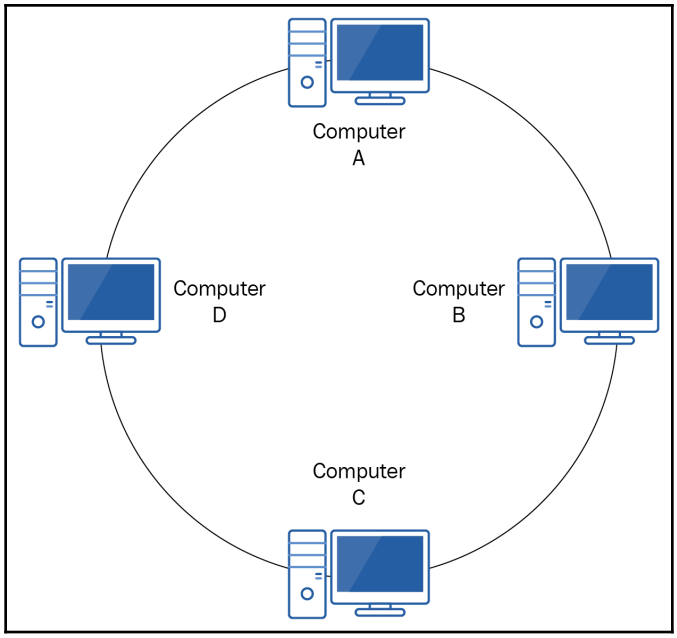

Figure 5.3: Ring topology

Early iterations of the ring topology were unidirectional, and quite often people would draw them as only going clockwise. Yes, they are unidirectional, but the devices do not understand the concept of clockwise and counterclockwise, so devices will transmit in one direction or the other depending on how they are configured. Looking back at *Figure 5.3*, let's imagine **Computer A** wanted to talk to **Computer D**. In a clockwise configuration, the data would pass through **Computer B** and **Computer C** en route to **Computer D**. In a counterclockwise configuration the data would transfer directly to **Computer D**.

In later iterations of this topology, traffic could be transmitted bidirectionally. This could either be *always on* or implemented to happen should a network fault occur. In *Figure 5.4*, we can see there is a fault between **Computer B** and **Computer C**:

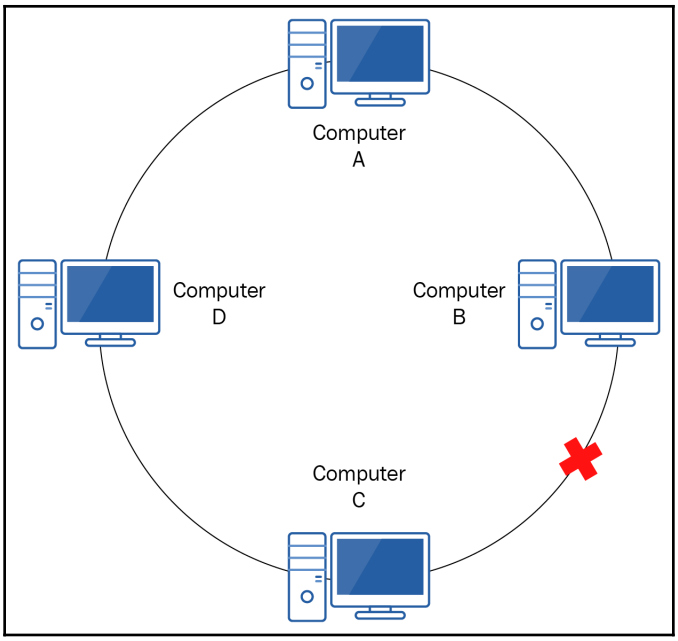

Figure 5.4: Ring topology with fault

However, if a bidirectional implementation was in place, **Computer A** could maintain communication with **Computer B** by sending data clockwise, and with **Computer C** and **Computer D** by sending data counterclockwise. The *reply* traffic would still be able to flow in the reverse direction. An example of a bidirectional ring topology is a **Fiber Distributed Data Interface** (**FDDI**), which will send data in both directions.

Advantages

A ring topology is relatively simple to troubleshoot. Looking back at *Figure 5.4*, if we know that **Computer A** can talk to **Computer B** but not to **Computer C**, then the issue must be somewhere between **Computer B** and **Computer C**, or on the devices themselves.

Additionally, as the devices in a ring topology are not fighting for access to the network media, no collisions take place, making it quite efficient. In addition to this, each device that receives the data will regenerate the signal before passing it on to the next device, thereby reducing signal attrition.

Finally, it is relatively simple to add a new device to a ring network. You disconnect the cable from one of the existing machines, plug that into the new device, and run a new cable between the new device and the device you had previously disconnected.

Disadvantages

I finished off the last section saying how simple it was to add a new device to a ring topology. While this is indeed the case, to do so will require any unidirectional networks to be brought down. If we are disconnecting a device, albeit temporarily, there is no way for the data to pass through. Likewise, if a device is faulty, it has the capacity to bring down the network, unless a bidirectional implementation is in place.

Another aspect to bear in mind is that, as each device receives the data, it has to perform a check of the data to see if it is for itself, before passing it on if it is not. On a small network, this is not too much of an issue, but as the network scales up, this will start causing considerable delay to the traffic.

Star topology

I always like to define a star topology as a network where all devices connect to a central point. I have seen numerous materials that refer to a *central hub*, and while this is correct terminology, I find some people get fixated on the word *hub*. That central point could be a hub device, it could be a switch, it could be a router, or it could be a server. Most likely, it will be a switch. You might be puzzled at my inclusion of a server in that list. While it is unusual, it can be done given the right hardware and software. I have also found that some materials draw a star topology with a device that looks like a server in the middle, and that becomes a fixation too.

In *Figure 5.5*, I have placed a switch as the central device, as this is the most common implementation:

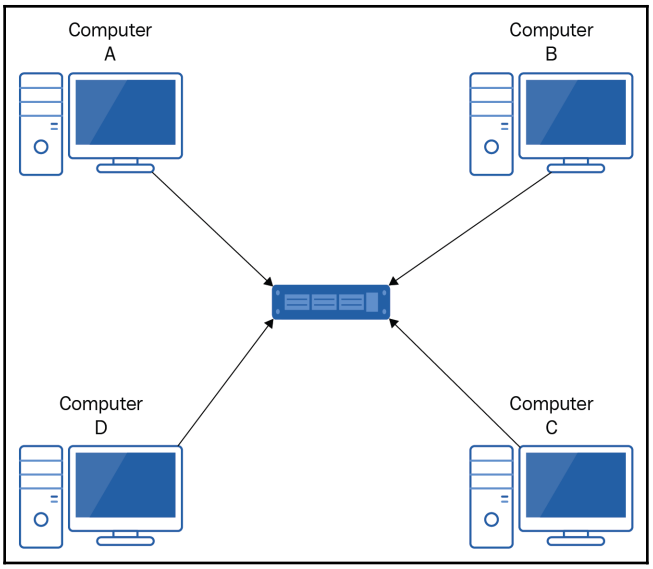

Figure 5.5: Star topology

Quite often you will see star topologies illustrated the way I have done in the preceding diagram, and there is nothing wrong with that. It makes it easier to interpret.

Look at *Figure 5.6*, and ask yourself whether it is a star topology:

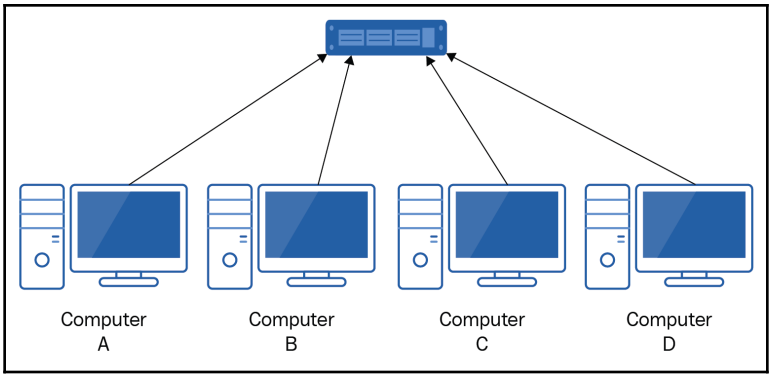

Figure 5.6: Another example for star topology

Remember, a star topology is one where devices connect to a central point. In the image, they all connect to that top switch. All communication must go through that one central device. Therefore, this is a star topology.

Advantages

There is a reason why the star topology is the most common topology currently in use. It is efficient and fairly resilient. The efficiency is dependent on what central device is in use, but remember that most modern topologies would use a switch as opposed to a hub to reduce collisions. The resilience comes from the fact that the failure of one device or connection generally does not bring down the whole network.

Looking at *Figure 5.7*, we can see that a fault on the cable between the central device and **Computer A** will still allow other devices to communicate:

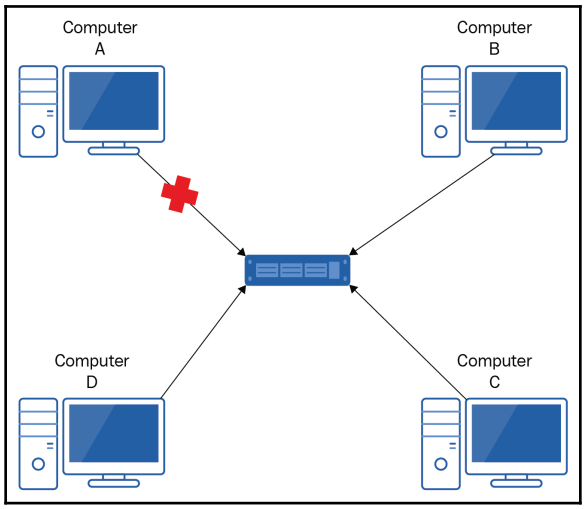

Figure 5.7: Fault on star topology

Other benefits of a star topology include the ability to add devices at will to the network without causing disruption. There is no requirement to take the network down just to add a new device. In addition, this topology scales well to large networks.

Disadvantages

Although the star topology is resilient to devices or their connections failing, I'd like you to look back at *Figure 5.5* and see if you can identify what could bring down the whole network. If you said the central device, you would be correct. That central device is a **single point of failure** (sometimes referred to as an **SPOF**). If that device goes down, no one is going to be talking. While there is usually some contingency in having spare (redundant) switches, the endpoint devices usually will only connect to one.

The other issue that is common with a star topology is that it can be quite expensive. The images that I have used so far have illustrated a relatively short cable between devices. In reality, in a star topology, your device will be connected to a wall port. That wall port will then be connected to a patch panel in a communications cabinet. Each wall port will have its only cable back to the cabinet. That all starts adding up to a lot of cable. There is no sharing of media here.

Token ring

At this point, I'm going to link back to logical and physical topologies, and give you the example that I promised at the start of this chapter. First let me explain how a token ring operates, and then I will dive into the topological areas.

In a token ring network (IEEE 802.5), a device can only talk when it is in possession of a *token*. That token is passed from device to device, until someone needs to talk, and they take possession of the token. Once they have finished with the token, they relinquish it for someone else to use.

Judging from its name, you are likely to assume that a token ring is a ring topology. Well, you are right and you are wrong. A token ring network has a physical star topology and a logical ring topology. Physically, the devices connect to a central device called a **media access unit** or **multiple access unit** (**MAU**), hence the physical star (dashed lines). But as far as the data is concerned, it goes from device to device (solid lines), and the MAU is ignored:

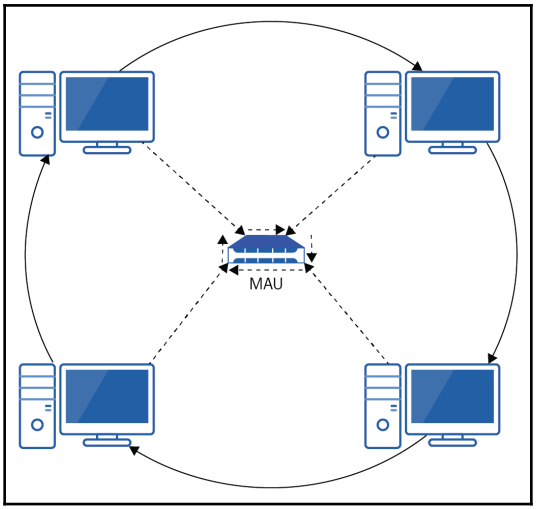

Figure 5.8: Token ring

Figure 5.8 shows a simplified token ring network. The red straight lines indicate the physical flow of the data. The green curved lines indicate the logical flow of the data. So you can see how this is a physical star and also a logical ring topology. I'd like to finish off this section by just mentioning something about the MAU. Recall that, in a ring network, if a device fails then it can impact on the whole network. Having an MAU in place overcomes that. If it detects that **Computer B** has failed, for example, it will skip that and pass the token to **Computer C**, thereby offering continuity of service.

Mesh topology

In `Chapter 4`, *Understanding Wireless Networking*, one of the topologies that we covered was a wireless mesh, and we can create a similar topology with a wired network. For ease, when I refer to mesh in this section I will be referring to a wired mesh unless I specify otherwise. A mesh network can take one of two forms, full mesh or partial mesh.

Exam tip: Unless specifically stated, the MTA does not differentiate between a wireless mesh and a wired mesh network. Do not over-think the questions by trying to differentiate between wired and wireless mesh networks yourself.

In a full mesh network, every device is connected to every other device. To be able to do this, devices will need to have a separate interface for each of the other devices. Now, while this is theoretically possible to do with the devices being a computer (if it had enough expansion slots on the motherboard), it is highly unlikely that this would happen. In the majority of mesh networks, the devices we are referring to would be either switches or routers, as they offer multiple available interfaces.

In *Figure 5.9*, I have created a very simple four-device mesh network. As you can see, each device is connected to the others, and therefore each has to have three interfaces available:

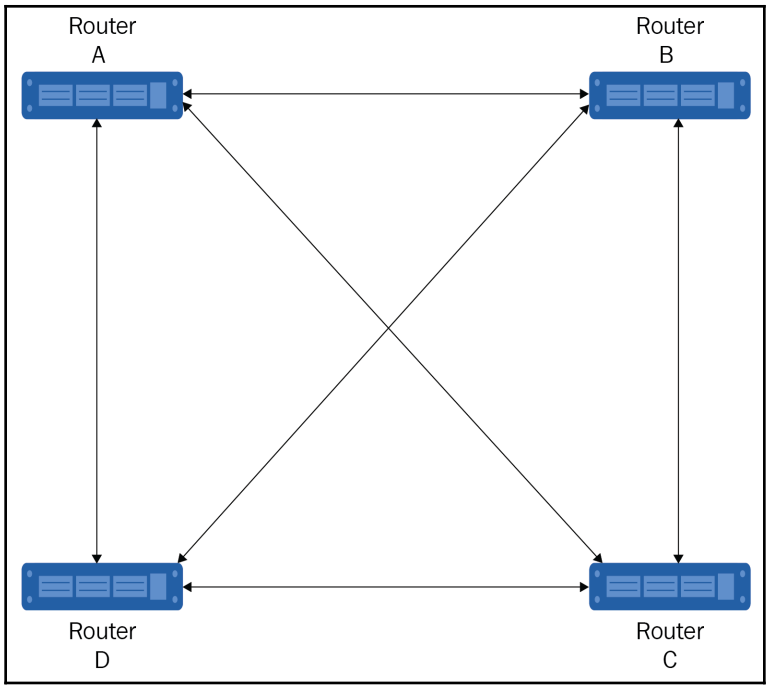

Figure 5.9: Full mesh topology

When planning a mesh network, it is important to be able to calculate how many interfaces you will need and how many cables. Look back at *Figure 5.8*: how many interfaces in total do you think we need, and how many cables? With this small network it is probably relatively easy to calculate just by counting from the image, but what if we have 100 devices? 300? 76,587? Don't worry, there is a pair of fairly simple formulas we can use:

n = number of nodes/devices
Number of interfaces = n(n-1)
Number of cables = (n(n-1))/2 or number of interfaces/2

So how many interfaces did we need for our four-device network?
Number of interfaces = n(n-1)
= 4(4-1)
= 4 x 3
= 12

Number of cables = n(n-1)/2
= 4(4-1)/2
= (4 x 3)/2
= 12/2
= 6

In contrast, a partial mesh network does not connect every device together, but will have some devices that are connected to all of the other devices. Those *fully connected* devices will usually be critical for the running of the infrastructure. *Figure 5.10* shows our four-device network as a partial mesh:

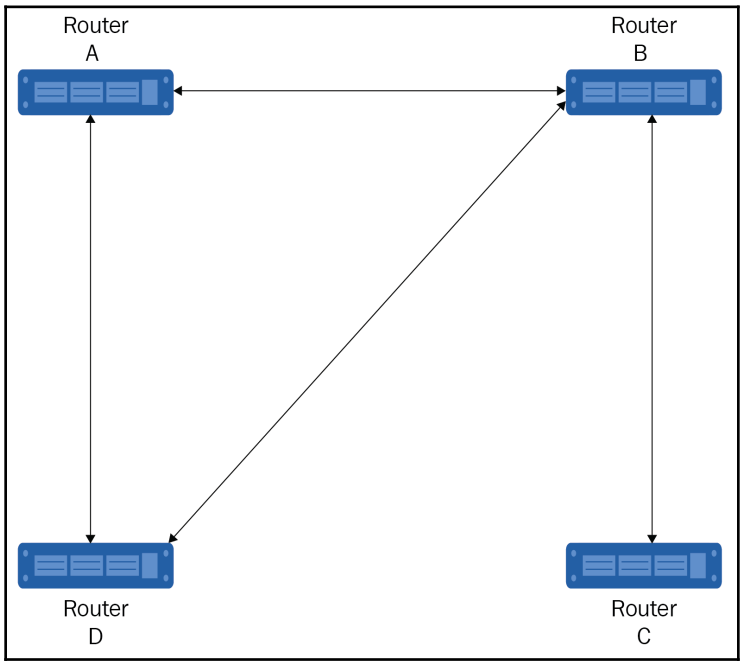

Figure 5.10: Partial mesh topology

As you can see only, **Router B** is connected to all of the other devices, and the remainder are a mixture of connecting to one or two devices.

Advantages

The key advantage to using a mesh topology is fault tolerance. A full mesh topology offers a high level of redundancy. Should the direct connection between two devices fail, an alternate pathway will be used. This is usually an automatic or dynamic change after the connection is identified as being lost. In *Figure 5.11*, the connection between **Router B** and **Router C** has been lost. Which route could the data from **Router B** take to get to **Router C** now? Let's have a look:

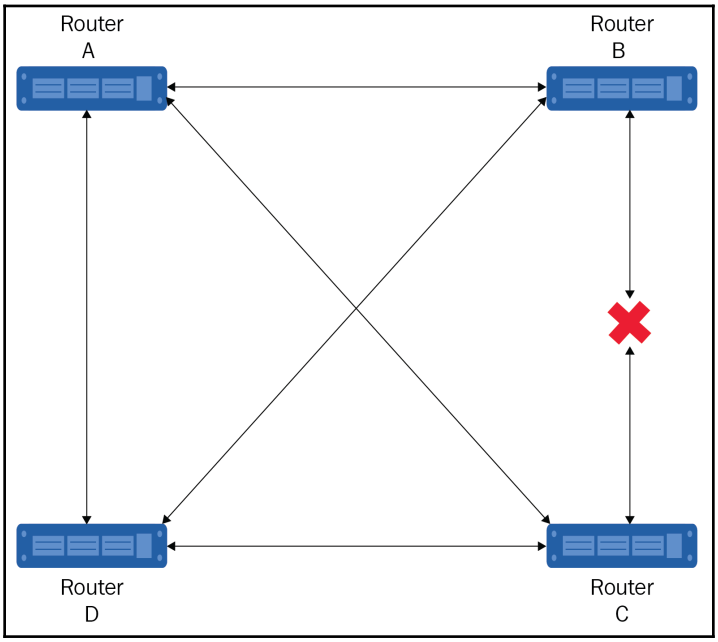

Figure 5.11: Mesh topology with failed connection

There are four possible alternative routes that could be taken in this scenario. The data could go in the following ways:

- **Router B | Router A | Router C**
- **Router B | Router D | Router C**
- **Router B | Router A | Router D | Router C**
- **Router B | Router D | Router A | Router C**

As you can see, on such a simple network, a full mesh topology gives quite a lot of redundancy.

Disadvantages

The obvious disadvantage to implementing this form of network is cost. A full mesh network is expensive to implement due to the number of cables and interfaces required, which is why a partial mesh could be seen as some form of compromise.

The other disadvantage is the skill set required to configure the mesh for redundancy. It is not quite as simple as plug the devices into each other, and it is ready to go. You need to ensure you configure the topology correctly, particularly the routing for failover. Failure to do so, could result in your *super fault tolerant* network crashing to a standstill.

Hybrid topology

One of the definitions of a hybrid is something consisting of mixed components, and that definition fits our purposes here. A hybrid topology is a network topology that connects two or more different network topologies together. Two such examples are a star-bus (*Figure 5.12*) and a star-ring (*Figure 5.13*):

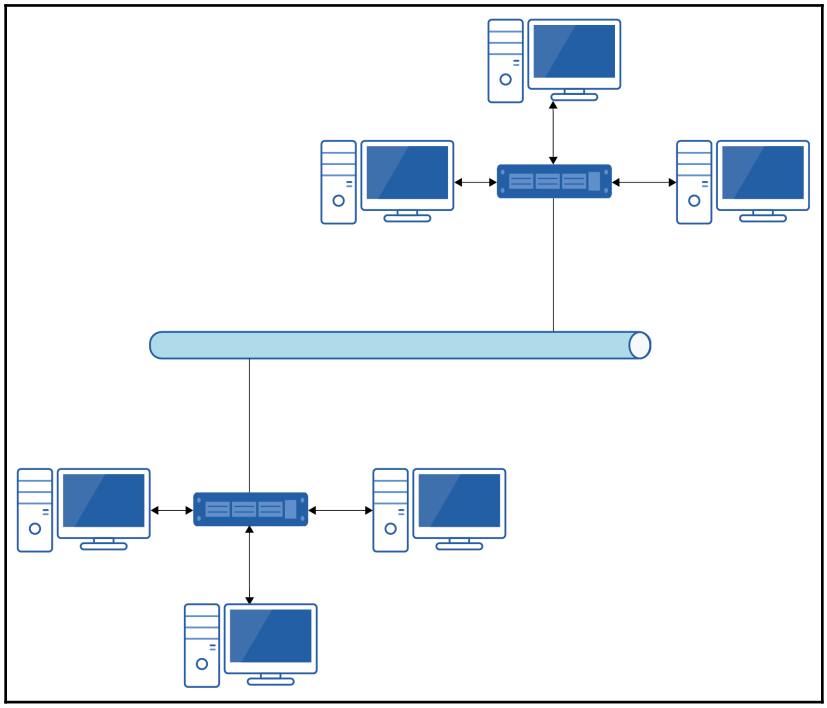

Figure 5.12: Star-bus hybrid topology

Looking at *Figure 5.13*, we can see the bus topology acting almost as a backbone for connecting the two star topologies. Any communications between the two star topology networks will have to abide by the *rules* of the bus topology, that is, follow the CSMA/CD process for access:

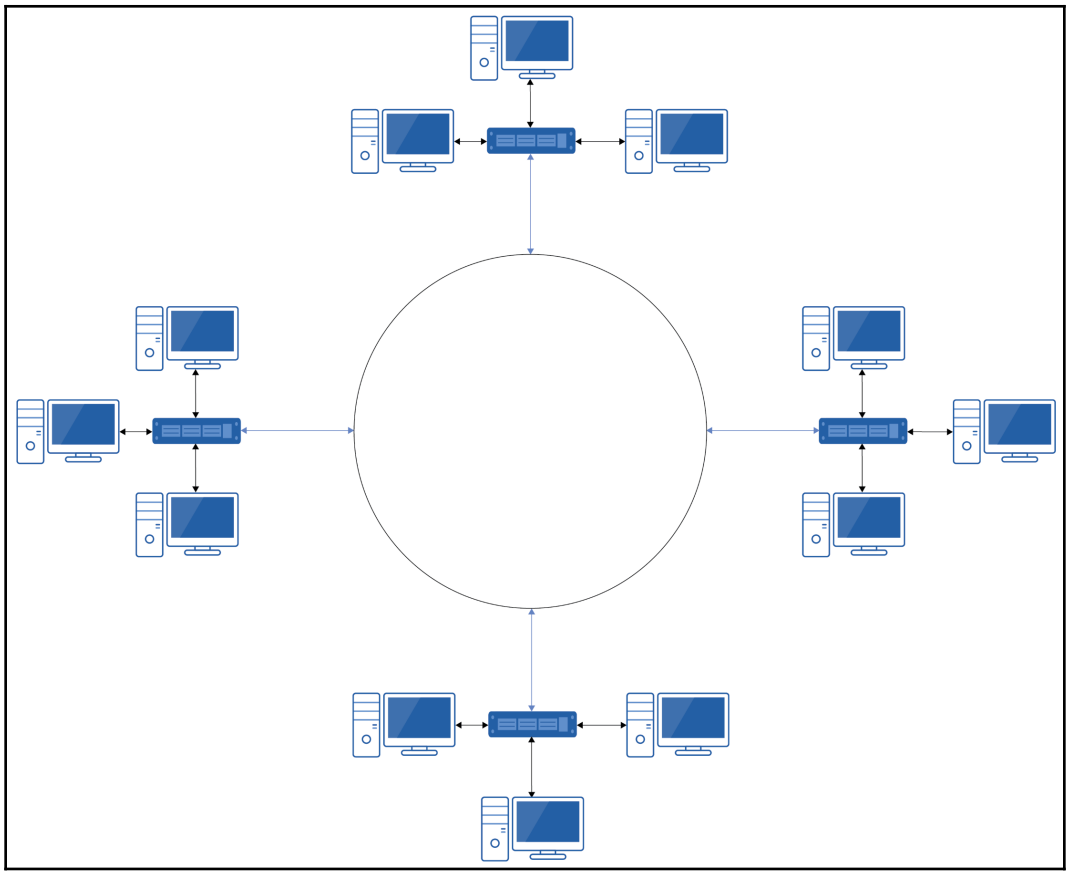

Figure 5.13: Star-ring hybrid topology

As you can see in *Figure 5.13*, the star-ring hybrid topology involves connecting a number of star topologies over a ring topology. Again, the *rules* of the intermediary topology need to be followed.

By using a hybrid topology, you are leveraging the benefits of the component topologies while minimizing the disadvantages. In the preceding two topologies, for example, by using stars as part of the topology, we are still allowing the devices in each star to communicate, even if the connection to the bus is lost.

Activity 3: Revisit the network plan that you created in `Chapter 2`, *Understanding Local Area Networks*. Now that you have learned about topologies in this chapter, which topology are you using?

Summary

We began this chapter by differentiating between logical and physical topologies before looking at the various topologies. The topologies covered included bus, ring, star, mesh, and hybrid, and we highlighted the benefits and disadvantages of each in terms of performance, resiliency, and cost.

You have learned how to understand the flow of the data using logical topologies and from that to understand some of the areas to troubleshoot on each topology in the event of a failure. On bus topologies, you have also learned about the importance of terminators to avoid signal bounce. In addition, you have learned the two calculations needed to identify the number of interfaces and cables in a full mesh topology, which will assist you greatly in planning such a deployment.

In the next chapter, you will be introduced to the first of the two intermediary network devices this book focuses on, namely, switches, with routers being covered in a later chapter. The next chapter discusses the purposes of switches, the benefits of using them over hubs, and how a switch makes a forwarding decision after receiving data.

Questions

1. In a token ring network, what is the central device known as?

(A) MAU
(B) Switch
(C) Router
(D) Server

2. Which access method is used on a bus topology?

(A) Token
(B) CSMA/CA
(C) Ticket
(D) CSMA/CD