Chapter 11
Wireless Security

THE FOLLOWING CEH EXAM TOPICS ARE COVERED IN THIS CHAPTER:

- ✓ **Wireless access technology**
- ✓ **Network topologies**
- ✓ **Communication on protocols**
- ✓ **Mobile technologies**
- ✓ **Security policy implications**

There was a time when you needed actual physical access to a facility to get onto a company's network. This is no longer true. These days, you likely just need physical proximity. Wireless networks are ubiquitous, especially as devices that have no ability to take in a traditional wired connection become more predominant. The problem with wireless networks, though, is that they use radio waves as the transmission medium. The signal strength isn't nearly the same as an AM or FM signal you would pick up with a radio you would listen to. The principle is the same, though. A signal is sent from a transmitter through the air, and a receiver gets the signal, as long as the receiver is within range.

Because they are just radio signals, they can be intercepted if you are within range of the transmitter. You can also send your own signals. What we traditionally call Wireless Fidelity (Wi-Fi) is a set of specifications categorized as 802.11 that are managed by the Institute of Electrical and Electronics Engineers (IEEE). 802.11 is not the only set of wireless specifications, however. Another communications protocol that uses wireless signals for transmission is Bluetooth. Both of these are common and can easily be misconfigured.

Even without misconfigurations, there are enough issues in the protocols themselves to open up the possibility of compromise. These means of communications have become essential for business operations, but they are exposed. It's important to understand the protocols as well as how the protocols can be compromised. Using at least one of them, you may be able to get into systems and networks as long as you have physical proximity.

Wi-Fi

Wi-Fi encompasses the set of standards that fall under the IEEE's 802.11. Wi-Fi is actually a trademark of the Wi-Fi Alliance and is used for devices that have passed certification tests. Whether you call it Wi-Fi or 802.11, we're talking about the same thing—network connectivity over a wireless connection. This makes it seem straightforward, and in general, if you are using it, it is. However, Wi-Fi has been through several versions since its introduction, and substantial changes to how Wi-Fi operates have been made in some of those versions.

First, 802.11 is a set of specifications for the Physical and Data Link layers. The Physical layer specifies the modulation schemes used to transmit the data and also specifies the frequency spectrum used. The first version of 802.11 specified transmission in the 2.4 GHz range with data rates between 1 and 2 megabits per second. This frequency range is the same as that used by other technologies, since it falls into the industrial, scientific, and medical (ISM) band. Devices like microwaves and some cordless phones operate within the same frequency spectrum as Bluetooth devices and Wi-Fi, making it a little crowded and leading to the potential for conflict.

While an early version, 802.11a, supported transmission in the 5 GHz and 2.4 GHz ranges, it wasn't widely implemented, especially in the consumer space. It wasn't until 2009, when 802.11n was released, that 5 GHz became a viable frequency to use for Wi-Fi networks. Since then, 802.11ac and the proposed 802.11ax have both been specified to work in the 5 GHz range. This causes issues, of course, since older radios in Wi-Fi interfaces won't work on anything other than the frequency they have been tuned to. If a network decided to adopt 802.11ac in the 5 GHz range but they still had a number of 802.11g cards, those cards would not be able to connect to the new network.

802.11n made significant jumps in the data throughput capability of wireless interfaces. It was able to do this through the use of multiple input, multiple output (MIMO). This meant an interface could use multiple channels for input and output and bind them together. Instead of being limited to 54 Mbits per second, it was possible to get speeds up to 600 Mbits per second using 802.11n. If your access point supported MIMO but your card did not, you would still be limited to the lower bandwidths because your card was not capable of doing anything better than a single channel at 54 Mbits per second, at a maximum.

802.11 uses channels to prevent one set of signals from clobbering another set. In the United States, there are 11 channels that can be used for Wi-Fi communications in the 2.4 GHz band. A channel is a bounded range of frequencies, much like channels on a TV or a particular radio station. Each of those is allocated a range of frequencies on which to transmit, depending on the amount of bandwidth needed. Some parts of the world are allowed to use 14 channels. This is dependent on the regulatory body that manages the frequency spectrum for electromagnetic transmissions. If you are using a version of 802.11 that supports transmission in the 5 GHz range, there are other channels that are available, though that also varies from one country to another. There is less consistency regarding the use of the 5 GHz range around the world than there is with the 2.4 GHz range. Later versions of 802.11 operate or will operate in the 60 GHz range, and there are six channels for use in that range.

Another aspect that varies from one version of 802.11 to another is the distance the signal is expected to travel. Some of the 802.11 versions will carry only 11 feet or so. Others are expected to carry up to a couple of hundred feet. This varies quite a bit depending on the number of walls and the composition of the walls. A Wi-Fi signal can carry quite a bit farther outside because there is nothing to impede the signal. Inside is a very different story. In terms of capturing signal without actually being in the building, how far from the building you need to be will depend on how close the nearest access point is to an external wall and how many windows there are. Windows would generally allow the signal to pass out unimpeded.

There are two different ways to set up Wi-Fi networks from a topological perspective. One of these is far more common than the other, but it's still possible to see both. Beyond that, you need to be concerned with how the Wi-Fi network handles authentication and encryption. Both authentication and encryption mechanisms provide possible ways to attack the wireless network. This is especially true in cases where companies allow employees to attach their own devices to the corporate network.

Testing Tip

If you are using Wi-Fi testing, it is helpful to have a second Wi-Fi interface if you are using Wi-Fi as your primary means of accessing the network. If you are trying to test with the same interface you are otherwise connected to the network with, you may end up with a bumpy ride. You can easily get a secondary Wi-Fi interface that connects with USB. This way you can do your testing over one interface without disconnecting your network connections over your primary interface.

Wi-Fi Network Types

There are two types of wireless networks you are going to run into. We can talk about topology here because it's technically relevant, though it may be harder to conceptualize than with wired networks. Let's give it a shot, though. The first type of network, and one that is less common, especially in a business environment, is an ad hoc network. An ad hoc network is one that exists without any central routing or switching device. Figure 11.1 shows an ad hoc wireless network with different device types connected to it. You can think of it as a dynamic mesh network. Each device communicates directly with another device. The connections are stood up dynamically as two devices have a need to talk to one another. When a device comes onto the network, it shares its information with the other devices. They then know how to make a connection to that device if they need to.

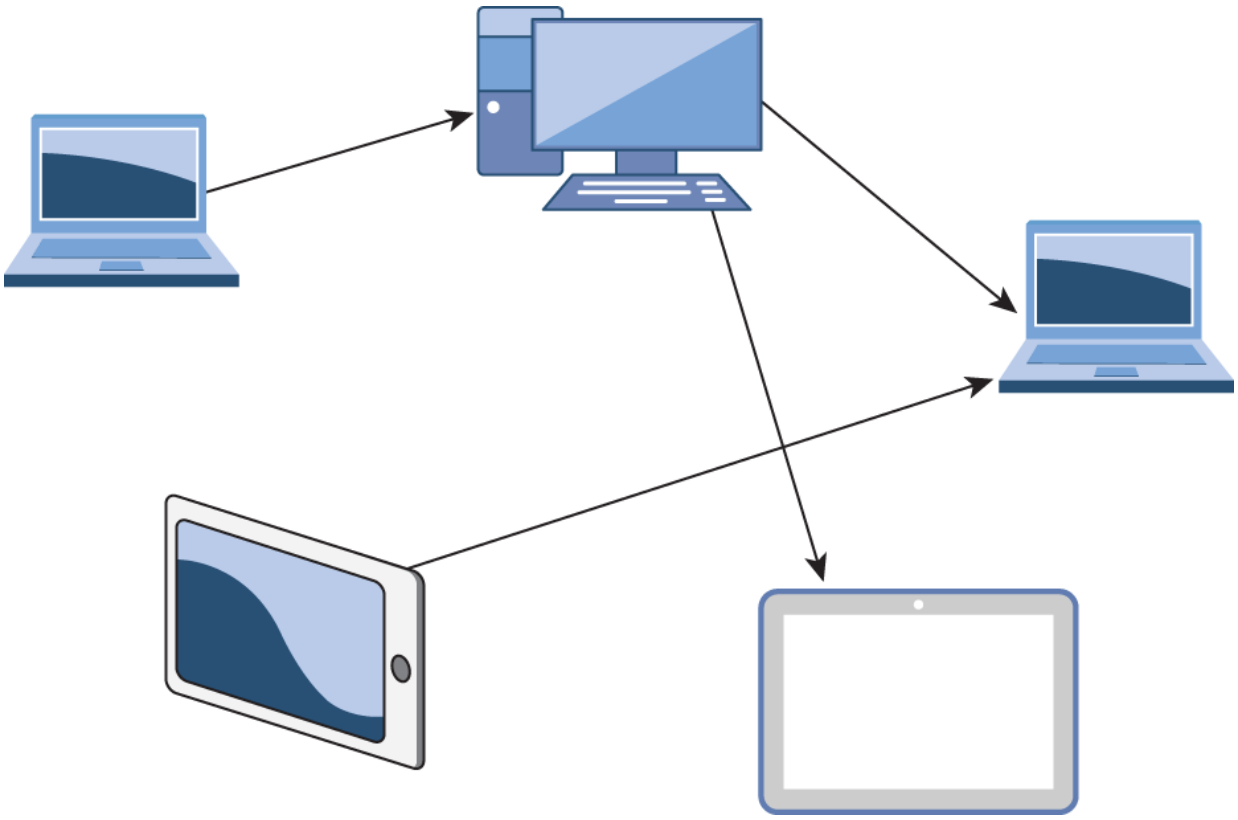


FIGURE 11.1 Wireless ad hoc network

This is not to say that anyone can connect to an ad hoc network. When an ad hoc network is started up by someone, they have to give the network a name. In addition to the network name, a password may be required to gain access. In an ad hoc network, all the devices have to be within range of each other because there is nothing else in place to forward messages to other devices at a distance. Ad hoc networks also don't provide encryption. Wi-Fi Direct, which replaces ad hoc networks, will allow encryption between stations.

While ad hoc networks are easy to set up and they can be fast, managing them is difficult, and they are certainly not suitable for an enterprise environment. Instead, there are infrastructure networks. An infrastructure network has a central device, which acts like a switch. Computers don't talk to one another directly. Instead, all messages go through an access point. Figure 11.2 shows an infrastructure-based network. The difference between a switch and an access point is that the switch controls the electrical signal. Since that's the case, the switch really does determine whether a system gets the traffic or not. An access point has no control over whether systems see the traffic or not because everything is in the air. The only thing you need to see it is a radio that can catch the signals.

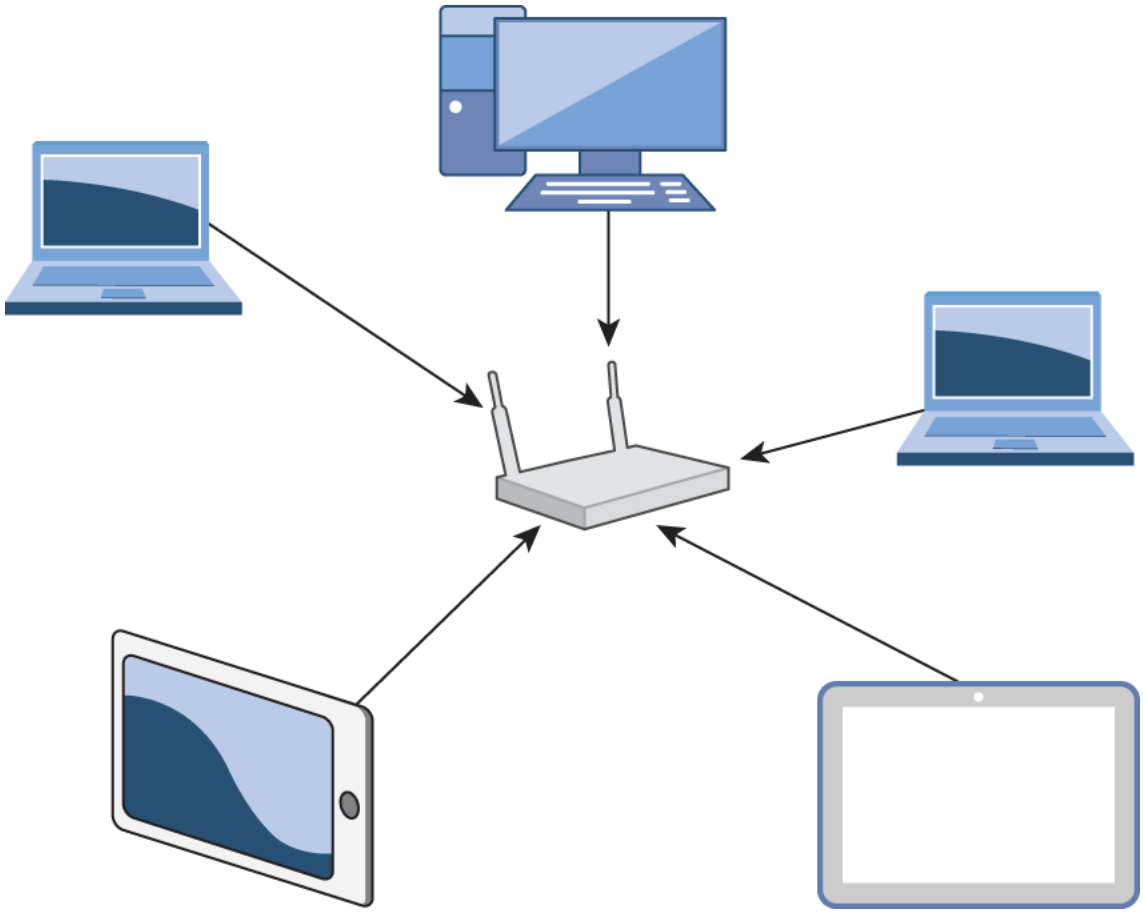


FIGURE 11.2 Wireless infrastructure network

To demonstrate that idea, take a look at Figure 11.3. This is a Wireshark capture with monitor mode enabled on the wireless interface. This means Wireshark gets everything down to the radio headers, which is the layer 1/2 set of headers for the frame. This particular system is not connected to any wireless network. You can think of that as having a wired interface being disconnected. As that's the case, Wireshark shouldn't be seeing anything, and if monitor mode was turned off, even with promiscuous mode on, there wouldn't be anything in the capture.

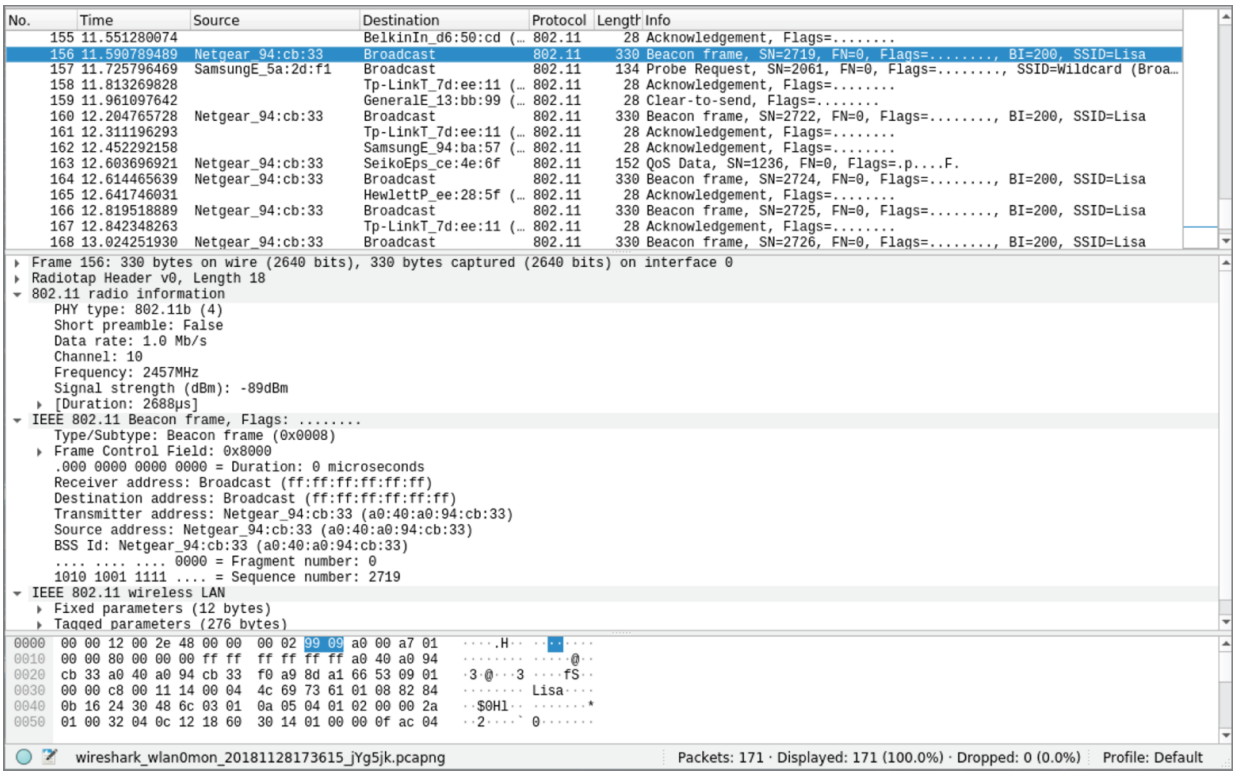


FIGURE 11.3 Wireshark capture of radio traffic

What you can see in the capture, though, is a lot of traffic. This is because wireless networks are very chatty. They have to be because there is no electrical connection that tells a system there is a network in place. Once monitor mode is on, though, you can see the messages between the access point and the clients. For example, access points send out beacon frames periodically. These are used to announce the presence of the network in the area. Additionally, clients will send out probe requests to identify networks in the area. When a probe request is sent out, the access point will send out a probe response. If you pull up a list of the wireless networks in the area, you may see your operating system, indicating that it is looking for networks in the area. This means it is sending out probe messages. To connect to a network, you need to first identify the network.

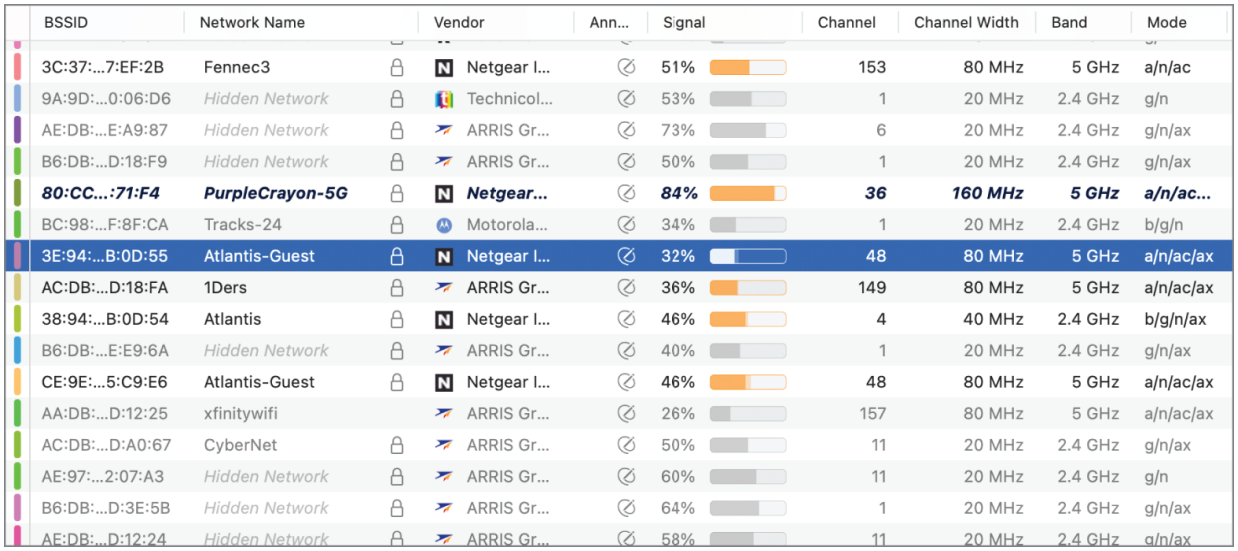
Wireshark Capture

Enable a monitor mode in Wireshark on your wireless network interface and capture traffic. Observe the radio transmissions that pass back and forth between your system and your access point. Disconnect from your wireless network, and then reconnect. Observe the radio traffic between as you reauthenticate to the network.

Wi-Fi Authentication

There are different forms of authentication that happen on Wi-Fi networks. The first we are going to go over is a simple machine-level authentication. This is where your client associates with the network. For that to happen, your system will send out probe requests to identify wireless networks in the area. Your client is looking for a service set identifier (SSID), which is essentially the name of your network. There may be multiple access points in a network that carry the same SSID, though, since you may have a large facility and the signal from a single access point isn't enough to carry through all of it. To distinguish from one access point to another when they are using the same SSID, there is a base service set identifier (BSSID). This looks like a MAC address and provides a way for a client to communicate with a specific access point, as needed. You can see an example of multiple

BSSIDs using the same SSID in Figure 11.4. The Atlantis-Guest network has two entries. Each has their own BSSID, while sharing the SSID of Atlantis-Guest.



BSSID	Network Name	Vendor	Ann...	Signal	Channel	Channel Width	Band	Mode
3C:37:...7:EF:2B	Fennec3	Netgear I...		51%	153	80 MHz	5 GHz	a/n/ac
9A:9D:...0:06:D6	*Hidden Network*	Technicol...		53%	1	20 MHz	2.4 GHz	g/n
AE:DB:...E:A9:87	*Hidden Network*	ARRIS Gr...		73%	6	20 MHz	2.4 GHz	g/n/ax
B6:DB:...D:18:F9	*Hidden Network*	ARRIS Gr...		50%	1	20 MHz	2.4 GHz	g/n/ax
80:CC...:71:F4	***PurpleCrayon-5G***	***Netgear...***		***84%***	***36***	***160 MHz***	***5 GHz***	***a/n/ac...***
BC:98:...F:8F:CA	Tracks-24	Motorola...		34%	1	20 MHz	2.4 GHz	b/g/n
3E:94:...B:0D:55	Atlantis-Guest	Netgear I...		32%	48	80 MHz	5 GHz	a/n/ac/ax
AC:DB:...D:18:FA	1Ders	ARRIS Gr...		36%	149	80 MHz	5 GHz	a/n/ac/ax
38:94:...B:0D:54	Atlantis	Netgear I...		46%	4	40 MHz	2.4 GHz	b/g/n/ax
B6:DB:...E:E9:6A	*Hidden Network*	ARRIS Gr...		40%	1	20 MHz	2.4 GHz	g/n/ax
CE:9E:...5:C9:E6	Atlantis-Guest	Netgear I...		46%	48	80 MHz	5 GHz	a/n/ac/ax
AA:DB:...D:12:25	xfinitywifi	ARRIS Gr...		26%	157	80 MHz	5 GHz	a/n/ac/ax
AC:DB:...D:A0:67	CyberNet	ARRIS Gr...		50%	11	20 MHz	2.4 GHz	g/n/ax
AE:97:...2:07:A3	*Hidden Network*	ARRIS Gr...		60%	11	20 MHz	2.4 GHz	g/n
B6:DB:...D:3E:5B	*Hidden Network*	ARRIS Gr...		64%	1	20 MHz	2.4 GHz	g/n/ax
AE:DB:...D:12:24	*Hidden Network*	ARRIS Gr...		58%	11	20 MHz	2.4 GHz	g/n/ax

FIGURE 11.4 Multiple BSSIDs for a single SSID

The process for getting your client connected to the access point requires that it authenticate with the access point. This may be a simple process, particularly if the access point has Open Authentication enabled. This is done to allow the client to start the conversation with the network. Actual authentication, particularly at the user level, happens later. The client sends an authentication request, and the access point will typically respond with an authentication response indicating “Successful.” This allows the client to move to the next stage.

Once the client has been authenticated, the next step is to associate the client with the network. This involves the client sending an association request to the access point. In the association request, the client, or *station* as it may be called, sends its capabilities. This would include the versions and speeds it supports. This allows for the access point and the station to negotiate the way they are going to communicate going forward. In Figure 11.5, you can see a diagram of the process that is required to get the station authenticated and associated with the access point.

FIGURE 11.5 Authentication and association steps

Of course, this is only part of the process required to get a station onto the network. There is probably also user-level authentication that's required. This authentication may make use of the IEEE's 802.1X standard. 802.1X allows for network-level authentication. It can also be used on wired networks to ensure that there aren't rogue devices in use. The newer versions of Wi-Fi encryption support enterprise authentication, which may use 802.1X to ensure that the user is authorized to get access to the network.

Newer versions of Wi-Fi protocols support a different authentication protocol, which will be discussed later in the chapter.

Wi-Fi Encryption

An initial concern about wireless networks, especially when the technology started to be offered for commercial use, was privacy. Wired networks were considered to be private because in order to see any of the transmissions over a wired network, an attacker had to have physical access to the network. While this wasn't entirely true, there was still a requirement to provide similar privacy to wireless networks. Wired Equivalent Privacy (WEP) was born. WEP used either a 40-bit key or a 104-bit key. The 40-bit key was concatenated with a 24-bit initialization vector, adding randomization to the pre-shared key (PSK) value to create a 64-bit key for the Rivest Cipher 4 (RC4) encryption algorithm. When you configured an encryption key into an access point, you entered 10 hexadecimal digits. Each hexadecimal digit is 4 bits, so 10 hexadecimal digits is the 40 bits needed.

The key value is small because of U.S. government restrictions on exports of cryptography. Those restrictions were eventually lifted, allowing a larger key value.

Once encryption restrictions were lifted, vendors could use a 104-bit key provided by the network administrator. Along with the 24-bit initialization vector, that created a 128-bit key, doubling the key length but exponentially

increasing the strength of the key. That assumed, however, that the initialization vector was truly random. As it turns out, the initialization vector used by WEP was not truly random. This meant that with enough data, the key could be determined, leading to WEP encryption being cracked. Fortunately, there are other encryption mechanisms that can be used.

Wi-Fi Protected Access

Encryption is complex, and encryption schemes are under constant scrutiny. As noted, WEP was cracked. Improvements in technology, especially computing power, have made cracking encryption much easier, which generally limits the shelf life of new encryption mechanisms. Because of that, you may understand why it is time-consuming to develop new ways to encrypt data. WEP was considered problematic, but it couldn't be swapped out with a long-term solution easily. As a stopgap measure, the Wi-Fi Alliance released Wi-Fi Protected Access (WPA). WPA could be implemented without any hardware changes. The hardware that allowed WEP to function could also be used for WPA. WPA could be enabled with a firmware upgrade.

WPA introduced the Temporal Key Integrity Protocol (TKIP). Where WEP concatenated the initialization vector with the PSK, TKIP mixed the keys. This is part of what allowed WEP-enabled devices to work with WPA. The same key, derived in a different way, could be fed into the same RC4 algorithm to perform the encryption of messages. TKIP also doesn't use a session key, as is common in other cryptosystems. Instead, it uses a per-packet key. Each packet has its own key for encryption and decryption. This is the temporal part, since each key is time-bound. This was meant to address one of the issues with WEP, where a collection of frames could yield enough data to derive the key because of the weakness in the initialization vector algorithm.

As far as the integrity goes, messages in transit could be intercepted and altered. WEP used a cyclic redundancy check (CRC) to verify the integrity of the message. This was considered inadequate. Instead, WPA implemented a message integrity check (MIC), which is sometimes called a *message authentication code* (MAC). This is not to be confused with the media access control (MAC) address. This MAC is meant to verify the integrity and authenticity of the message that has been sent. The receiving party checks its own calculation of the MAC against the one included with the message. If the MACs match, the message has not been tampered with or otherwise corrupted in transit.

WPA supports two modes of authentication. The first one functions essentially the same as the key used in WEP and is called WPA-Personal. This is a PSK method. The PSK functions as a password and also provides the key that is used with the initialization vector to generate the encryption key used with the RC4 encryption algorithm. The second mode of authentication, called WPA-Enterprise, uses the Extensible Authentication Protocol (EAP). EAP can be used to support different backend password storage. As an example, you can use WPA-Enterprise with your existing enterprise user database and authentication scheme, such as Windows Active Directory. EAP is used to transmit authentication information along with success or failure responses.

Another means of authentication is Wi-Fi Protected Setup (WPS). This can require physical control of the access point, since it may involve pushing a button. There may also be a personal identification number (PIN) needed to complete WPS. It is used to handle key distribution so the users of the client devices don't know what the key is. The key is stored with the endpoint and is used to communicate with the access point when devices come online. There are issues with WPS, however. The PIN used with WPS can be recovered due to problems with the implementation of WPS.

Wi-Fi Protected Access 2

Wi-Fi Protected Access 2 (WPA2) is a standard that was ratified as IEEE 802.11i-2004, which means it's an amendment to the original 802.11. There have been several other amendments, including ones that provide the capability to handle MIMO streams as well as different channels and bandwidths. You may be familiar with them as 802.11b, 802.11g, 802.11n, and others. WPA2 was intended to be the long-term solution to the problems with WEP. Some of the problems with WEP continued with the use of WPA, since WPA continued to use the RC4 stream cipher. This is an older encryption cipher, and WPA is still used as an initialization vector. A poorly implemented initialization vector can lead to getting keys derived, just as with WEP. WPA2 worked to correct some of the issues with WPA, including replacing RC4 with AES-CCMP (Counter Mode CBC-MAC Protocol).

WPA2 improves the overall security of the encryption implementation for Wi-Fi. One of the biggest issues with encryption has always been protection of the key. WPA2 introduces a four-way handshake that didn't exist in earlier implementations. The four-way handshake is meant to improve the protection of the key. There are two

keys that are used through this process. The first is the pairwise master key (PMK), which is retained over time so it has to be protected and not transmitted. The key that gets derived is the pairwise transient key (PTK). There is also a group temporal key (GTK) that comes out of the handshake; it is used for broadcast or multicast traffic, meaning it's a key that is shared across multiple devices.

Figure 11.6 shows a diagram of what the four-way handshake process looks like. First, the access point generates a nonce, which is an ephemeral, random value. The access point sends its nonce to the station, along with a key replay counter, which is used to match messages that are sent so any replayed messages can get discarded. The station has what it needs at this point to generate the PTK. The station generates its own nonce and sends it to the access point, along with the key replay counter that matches the one sent by the access point. The station also sends a message integrity check (MIC, a message authentication code) to ensure there is no tampering with the message that is sent.

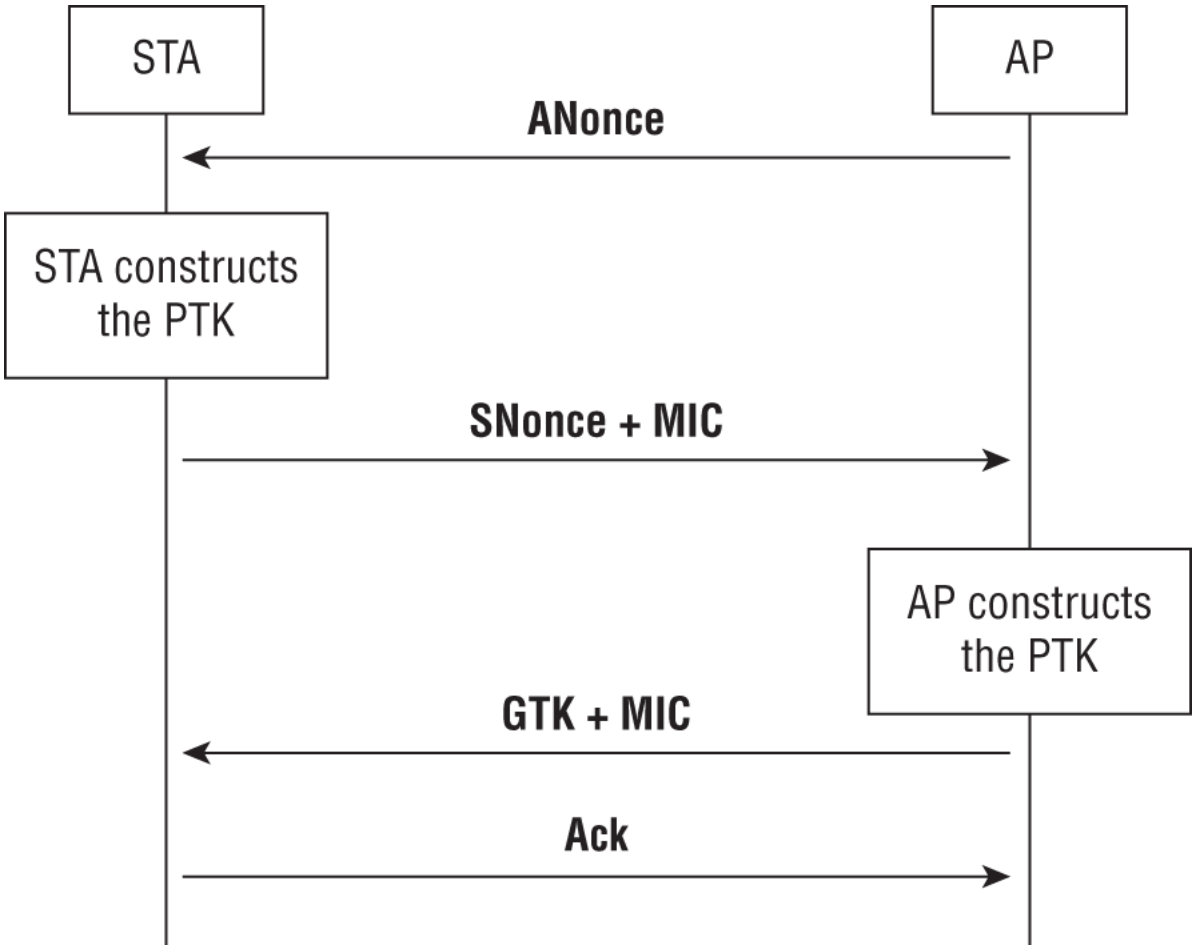


FIGURE 11.6 Four-way handshake for WPA2

Once the access point has the station's nonce, it can derive its own PTK. The access point sends the GTK on to the station, along with an MIC. The station sends along an acknowledgment, and the four-way handshake is complete. The four-way handshake completes authentication, since you have to have the right keys in order to be able to complete the handshake.

Just as with WPA, there is personal and enterprise authentication available, and the handshake process is the same, no matter which method of authentication is available. Personal requires a pre-shared key. Enterprise authentication uses 802.1X to handle user-level authentication, meaning someone connecting to the network has to provide a username and password. The authentication framework used is the EAP and the different variations of EAP. Figure 11.7 shows a list of different EAP variations and other authentication protocols that may be used. This is from a configuration page on a Linux system, setting up a wireless network connection.

A couple of common variations are the Lightweight Extensible Authentication Protocol (LEAP) and the Protected Extensible Authentication Protocol (PEAP). These frameworks, along with others, like EAP-TLS, are used to ensure that the authentication is done over a protected channel so any authentication parameters are not sent in the clear. Additionally, any key exchange is done across a protected channel. Often, this is over Transport Layer Security (TLS). This is the means by which web traffic is encrypted. It makes use of certificates and the asymmetric key encryption using public and private keys.

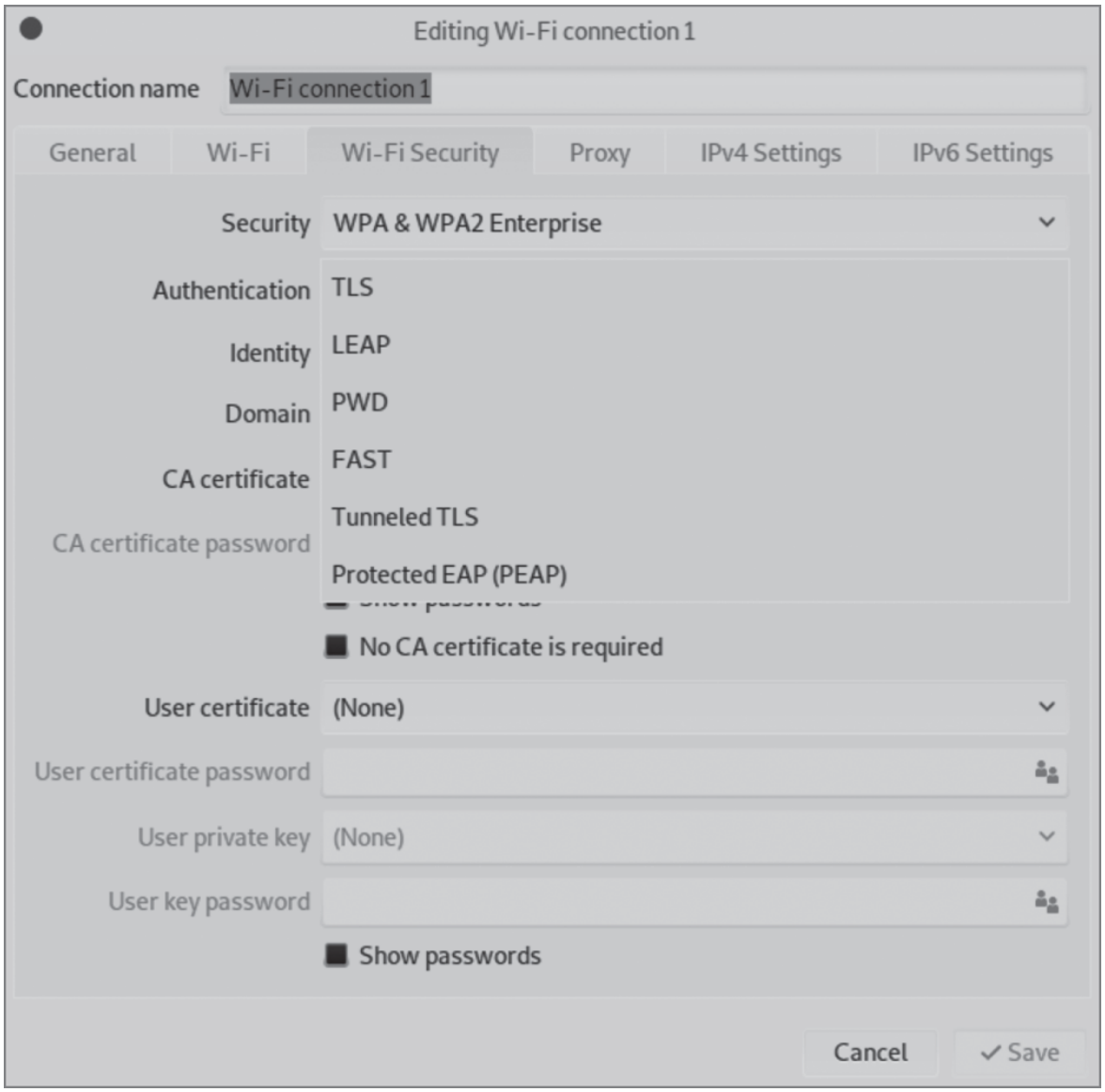


FIGURE 11.7 Wireless configuration under Linux

Wi-Fi Protected Access 3

In 2018, the Wi-Fi Alliance introduced Wi-Fi Protected Access 3 (WPA3) to help protect against some of the weaknesses in WPA2. The first thing is allowing longer encryption keys, up to AES-256 with SHA-384 for cryptographic hashing to perform message authentication. Additionally, WPA3 introduces Simultaneous Authentication of Equals (SAE) to dispose of the pre-shared key. Using a pre-shared key is a weakness because it's a piece of information that may be hard to fully protect. When pre-shared keys are used for authentication, anyone who can get the pre-shared key will be able to perform authentication. SAE is a password-based authentication strategy used to agree on keys that are then used to encrypt the session between the station and the access point.

The authentication and association process using SAE is different from the four-way process described earlier. This is the process that is followed using SAE.

1. The station (STA) sends a probe request to the access point (AP).
2. The AP sends a probe response to the STA.
3. STA sends an authentication commit to the AP. This is used to generate the pairwise master key (PMK) on the STA. This is an 802.11 authentication frame.
4. AP sends an authentication commit to the STA. This is used to generate the pairwise master key on the AP. This is also an 802.11 authentication frame.
5. STA sends an authentication commit to the AP. This is an 802.11 authentication frame. This contains a message with a key generated for the AP to validate.
6. AP sends an authentication commit to the STA. This is also an 802.11 authentication frame. It contains a message with a key generated for the STA to validate.
7. Regular association request.
8. Regular association response.
9. Four-way handshake using the PMK that was generated in the previous steps.

WPA3 also continues to mandate the use of Counter Mode Cipher Block Chaining Message Authentication Code Protocol, also known as CCM mode protocol (CCMP). This is a set of cryptographic protocols put together to address weaknesses from WEP. It uses a combination of encryption along with authentication and integrity to ensure that messages were not tampered with, and that those messages originate from the correct endpoint. This helps protect against man-in-the-middle attacks. CCMP uses a block cipher for encryption so messages are encrypted in fixed-length blocks. While meet-in-the-middle attacks, where plain text can be used to attack the encryption, are possible, CCMP helps protect against these sorts of attacks. A meet-in-the-middle attack is a way of weakening the key strength used to make it easier to brute-force the decryption process.

Bring Your Own Device

You probably have at least one mobile device, whether it's a smartphone, a tablet, or a laptop. It may also be a hybrid device like Microsoft's Surface. Because of the prevalence of these devices and their ease of use, people carry them around and have at least smartphones with them nearly all the time. Companies often want to take advantage of these computing resources, so they may have policies around how to allow people to use their own devices on enterprise networks. This policy is typically referred to as bring your own device (BYOD). A company may require devices to be approved or meet certain minimum requirements.

A company that allows personal devices on their network, generally through wireless networks because mobile devices generally don't have wired connections, may be open to any device connecting. This means they may not have a form of network access control (NAC) that can restrict which devices can talk to their wireless networks. They can do this by blocking MAC addresses to prevent anyone not on the allow list from even getting to the point of user-level authentication. As shown earlier, WPA2 with Enterprise authentication also provides protection by requiring that users authenticate.

Another thing to consider when it comes to accessing over Wi-Fi is network isolation. It's common for businesses to put Wi-Fi clients into an isolated network that behaves as though it's outside the enterprise. To access internal corporate resources, anyone on the Wi-Fi network, even after authentication, has to use a virtual private network connection to get access to business assets. They connect to a password-protected Wi-Fi network using WPA2-Enterprise authentication and encryption, and then they have to further authenticate to get a tunnel open to the inside of the network. This means even if you get access to the employee Wi-Fi network, typically separate from the guest Wi-Fi network that may also be available, you still won't have easy access to business assets.

Companies that have embraced BYOD, however, may have made it easier to gain access to their resources. Of course, BYOD isn't the only way employees can get access to corporate resources over Wi-Fi, but if BYOD is enabled, it likely means there aren't restrictions about devices that can connect, preventing someone from even being able to provide authentication credentials to get access.

A common approach for businesses who do allow BYOD may also implement a guest network. This is a separate, isolated network that untrusted users are put on. In order to get access to corporate resources, the clients may need to use a virtual private network to create a trusted, encrypted connection to the inside of the network.

Wi-Fi Attacks

There are different ways to attack Wi-Fi networks. First, as with any type of attack, you can start with reconnaissance. There are ways to do that using Wireshark and other tools. What you need when you are sniffing is to generate traffic. One way to do that is to force endpoints to authenticate themselves. You can use a deauthentication attack to do that. Another attack is something called the *evil twin*. This is a rogue access point, but one that mimics a legitimate access point. There is also a key reinstallation attack, sometimes called KRACK, which can be used to force endpoints to send messages in a way that attackers can decrypt them easily.

Testing Networks

You probably don't want to perform testing on your own network since it will disrupt traffic. Any device that relies on wireless connectivity for operation will stop functioning correctly when you perform some of these attacks. It would also be rude and illegal to test against your neighbors' access points. It's probably helpful to get another access point that you can test against.

Wireless Footprinting

Wireless footprinting is the process of identifying wireless networks in an area or understanding the boundaries of those wireless networks. Some people may refer to this process as war-driving, which is an older term. There are different tools you can use for this, including Kismet on Linux systems or WiFi Explorer on macOS. Each of these tools will provide you with a list of all the wireless networks that are advertising and can give you the signal strength readings as well. This will give you a sense of how far from the access point you are.

There are environmental factors that may impact the reach of a network. Building materials will have an impact on signal degradation. Plywood and drywall offer little resistance to a wireless signal. Concrete, on the other hand, has a significant impact, causing extensive signal loss. Wireless devices placed inside concrete buildings may not be accessible outside the building. Glass has a limited impact, so even if a building is made of concrete, the signal can still get out if there are enough windows in the building. Keeping access points away from windows will help to protect the signal from getting out. Bricks and masonry blocks also offer significant resistance to a wireless signal.

One limitation to wireless footprinting is the antenna technology in use. Most wireless antennas are omnidirectional, sending the signal out in all directions at once. This disperses the signal strength. The receiving antenna similarly is listening from all directions at once. One thing that can help to increase signal strength is a device that will focus the signal, enhancing what is received at the antenna. It has long been thought that using a Pringles can along with a wireless antenna will increase the signal strength. To the extent this is possible, it works because it turns the antenna into a directional antenna. This will allow more signal to hit the antenna, causing better reception. This does, though, mean you need to move the antenna around since it will pick up from the direction it is pointed in.

Wireless Footprinting

While you can get a list of the wireless networks available to you from the operating system wireless configuration settings, get a copy of a wireless assessment tool like Kismet, WiFi Explorer, or NetSpot. Identify the wireless networks available to you by signal strength.

Sniffing

As noted earlier, Wireshark can be used to capture network traffic that includes the radio headers. The radio headers are important when you are trying to gather information. They are not the only headers that carry information, though. You will want to see all the beacons from the access points and any authentication messages that may be sent between stations and access points. To see the entire stack, you need to set your wireless interface into monitor mode. One way to do this is by using the `airmon-ng` program from the `aircrack-ng` package. In the following code listing, you can see the use of `airmon-ng` to create a monitor interface from a wireless interface. In the end, you have a new interface that you can use to collect radio and other traffic.

Using airmon-ng

```
root@quiche:~# airmon-ng start wlan0

Found 3 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

  PID Name
  640 wpa_supplicant
  30835 NetworkManager
  30859 dhclient

  PHY      Interface     Driver          Chipset

  phy0     wlan0           rt2800usb     Ralink Technology, Corp. RT5372

            (mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan0mon)
            (mac80211 station mode vif disabled for [phy0]wlan0)
```

You'll notice some warnings at the start. This is because the version of Linux this is running under manages the network interfaces. `airmon-ng` needs to be able to make changes to the network interfaces, meaning it's better if the processes managing the interfaces aren't running. While this example is running under Linux, `aircrack-ng` does have a Windows package, and the same programs can be used there. Once the monitor interface has been created, you can use it as you would any other network interface. You can use `tcpdump` to capture all network traffic that passes across the monitor interface. In the following listing, you can see the use of `tcpdump` using the interface `wlan0mon`, which was created by `airmon-ng`.

Using p: tcpdump with a Monitor Interface

```
 root@quiche:~# tcpdump -i wlan0mon
 tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
 listening on wlan0mon, link-type IEEE802_11_RADIO (802.11 plus radiotap header), capture size
262144 bytes
 20:49:20.747433 1.0 Mb/s 2457 MHz 11b -19dBm signal antenna 1 Data IV:8cae Pad 20 KeyID 2
 20:49:21.190521 1.0 Mb/s 2457 MHz 11b -43dBm signal antenna 1 Clear-To-Send
RA:64:1c:b0:94:ba:57 (oui Unknown)
 20:49:21.387878 1.0 Mb/s 2457 MHz 11b -67dBm signal antenna 1 Probe Request (WifiMyqGdo) [1.0
2.0 5.5 11.0 Mbit]
 20:49:21.403289 1.0 Mb/s 2457 MHz 11b -67dBm signal antenna 1 Probe Request (WifiMyqGdo) [1.0
2.0 5.5 11.0 Mbit]
 20:49:21.537165 1.0 Mb/s 2457 MHz 11b -47dBm signal antenna 1 Probe Request (ZoeyWasHere) [1.0
2.0 5.5 11.0 Mbit]
 20:49:21.589809 1.0 Mb/s 2457 MHz 11b -19dBm signal antenna 1 Acknowledgment
RA:64:1c:b0:94:ba:57 (oui Unknown)
 20:49:21.689230 1.0 Mb/s 2457 MHz 11b -43dBm signal antenna 1 Clear-To-Send
RA:64:1c:b0:94:ba:57 (oui Unknown)
 20:49:23.247016 1.0 Mb/s 2457 MHz 11b -67dBm signal antenna 1 Acknowledgment
RA:ec:aa:a0:4d:31:a8 (oui Unknown)
 20:49:23.250814 1.0 Mb/s 2457 MHz 11b -65dBm signal antenna 1 Acknowledgment
RA:f6:aa:a0:4d:31:a8 (oui Unknown)
 20:49:23.594399 1.0 Mb/s 2457 MHz 11b -19dBm signal antenna 1 Acknowledgment
RA:64:1c:b0:94:ba:57 (oui Unknown)
 20:49:23.602831 1.0 Mb/s 2457 MHz 11b -21dBm signal antenna 1 Beacon (CasaChien) [1.0* 2.0*
5.5* 11.0* 6.0 9.0 12.0 18.0 Mbit] ESS CH: 11, PRIVACY
```

```
 20:49:24.258485 1.0 Mb/s 2457 MHz 11b -43dBm signal antenna 1 Clear-To-Send
RA:64:1c:b0:94:ba:57 (oui Unknown)
 20:49:24.635482 1.0 Mb/s 2457 MHz 11b -57dBm signal antenna 1 Probe Request (CasaChien) [1.0
2.0 5.5 11.0 6.0 9.0 12.0 18.0 63.5 Mbit]
 20:49:24.656324 1.0 Mb/s 2457 MHz 11b -51dBm signal antenna 1 Acknowledgment
RA:18:d6:c7:7d:ee:11 (oui Unknown)
 20:49:24.658753 1.0 Mb/s 2457 MHz 11b -51dBm signal antenna 1 Acknowledgment
RA:70:3a:cb:52:ab:fc (oui Unknown)
```

From the `tcpdump` output, you can see all of the radio traffic, including signal information. You can see the frequency being used for the communication, 2457 MHz in all the messages. You can also see the signal strength. This is measured in decibels in relation to a milliwatt, expressed as dBm. The decibel is a ratio of two values using a logarithmic scale. The dBm, however, isn't a ratio. It's an absolute value, as it uses the milliwatt as a reference point. It is a standard way of measuring signal strength in wireless networks. The very best signal you can get is –10 dBm. You can see values of –19 dBm to –67 dBm. These are still usable signals. The lowest you can get in a wireless network is –100. Anything beyond that isn't usable.

What you can see from the `tcpdump` output are probe requests for networks. These probe requests are coming from devices that were once connected to these networks. For example, there is a garage door opener that has wireless capabilities trying to find the network WifiMyqGdo. This is the network that the device is configured with when it comes from the manufacturer. This information may be useful if you try to do an evil twin attack. However, you may want to see other information from the packet capture. What you can do is just write out the packet capture. In the following listing, you can see writing out the packet capture from `tcpdump`. Because we are using the monitor interface, we'll still get all of the radio messages.

Writing a Capture from a Monitor Interface

```
 root@quiche:~# tcpdump -w radio.pcap -i wlan0mon
 tcpdump: listening on wlan0mon, link-type IEEE802_11_RADIO (802.11 plus radiotap header),
capture size 262144 bytes
 ^C125 packets captured
 126 packets received by filter
 0 packets dropped by kernel
```

Once you have a saved packet capture, you can open it in a program like Wireshark. Figure 11.8 shows the same probe request mentioned earlier in more detail. Here you can see not only the SSID, you can also see some of the capabilities being requested by the station. You'll also see the channel being requested and the vendor ID of the equipment. This is part of what makes it clear it's a garage door opener. The vendor ID for the sending device is Chamberlain Group, which is the company that owns LiftMaster garage door openers. You'll also see there is vendor-specific information in the probe for Broadcom. Broadcom is a common wireless chipset manufacturer. This suggests that there is an expectation that the wireless device knows about Broadcom capabilities and expects to communicate using some of them.

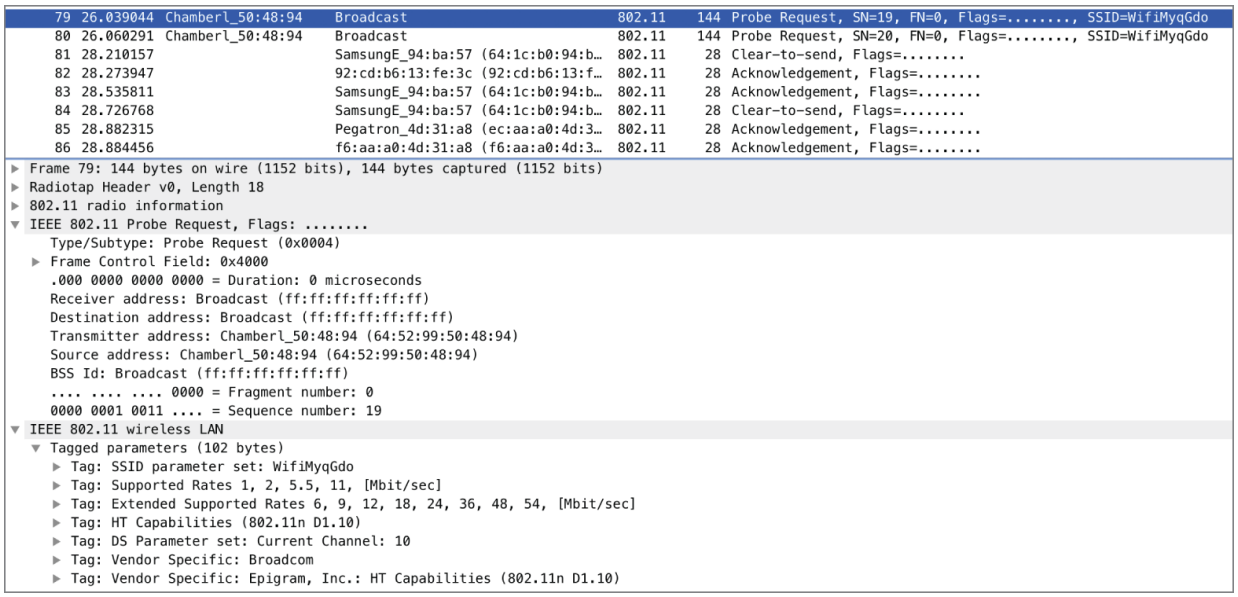


FIGURE 11.8 Probe request in Wireshark

Wireshark can also provide additional details about the wireless connection that you were not able to see in `tcpdump`. In Figure 11.9, you can see the radio header information. This includes the channel frequency and number as well as the signal strength. In theory, the signal strength can give you an indication of the proximity of the radio you are looking at. This does assume that all radios and antennas are roughly equivalent, meaning signal strength would be a function of distance. In practice, however, there are a lot of other factors, including any materials the signal is passing through. Signals don't pass very well through concrete or metal, for example.

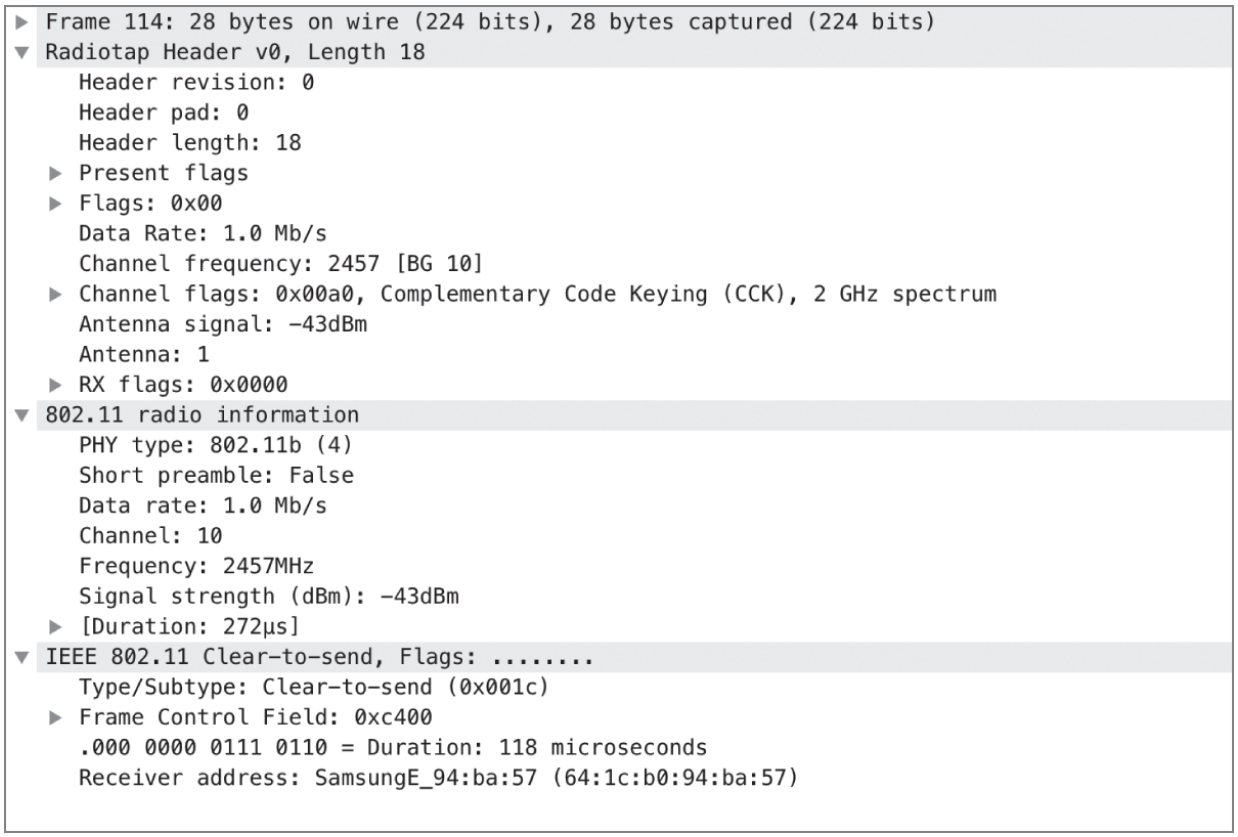


FIGURE 11.9 Radio headers in Wireshark

Radio headers and other wireless packets can provide us with some information. One of the challenges with this is that you are going to miss data unless you know the channel of the wireless network you want to sniff. If you are just trying to generally gather data, you may be able to frequency- or channel-hop if your card's driver supports that. When you channel-hop, though, you are moving from one channel to another, listening on each new channel. You are missing data on all the other channels. This is a limitation of wireless cards, just as with any other radio. You can tune into only a single frequency at a time. This means it's best to do some scanning ahead of time to identify which channel the network you really care about listening is on.

Deauthentication Attack

A deauthentication attack sends messages that force stations to reauthenticate against the access point. Essentially, it logs out any station, making the station reestablish the association. This is another place to use the `aircrack` suite of tools. In this case, we're going to use `aireplay-ng`. As discussed, the interface needs to be on the right channel to know what frequency to send messages on. Under Linux, this is done using the `iwconfig` program. Under Windows, this is done through the device properties of the wireless interface. In the following code, you can see setting the correct channel for the access point and then running a deauthentication attack. You will see 10 deauthentication messages being sent to the station.

Deauthentication Attack

```
root@quiche:~# sudo iwconfig wlan0mon channel 6
root@quiche:~# aireplay-ng -0 10 -a  C4:EA:1D:D3:78:39 -c 64:52:99:50:48:94 wlan0mon
18:22:59  Waiting for beacon frame (BSSID: C4:EA:1D:D3:78:39) on channel 6
18:23:00  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|62 ACKs]
18:23:00  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|60 ACKs]
18:23:01  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|34 ACKs]
18:23:01  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|32 ACKs]
18:23:02  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|61 ACKs]
18:23:02  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|57 ACKs]
18:23:03  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|59 ACKs]
```

```
 18:23:03  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 1|56 ACKs]
 18:23:04  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|55 ACKs]
 18:23:05  Sending 64 directed DeAuth (code 7). STMAC: [64:52:99:50:48:94] [ 0|43 ACKs]
```

To perform this attack, you need to have the BSSID of the access point. This is the MAC address for the device, so it can be used in the frame headers. You also need the MAC address of the station on which you want to run the deauthentication attack. You can also run deauthentications against all stations by omitting the `-c` parameter, which is the client MAC address. To get some additional details to run this attack, you can use `airodump-ng`, another program in the `aircrack-ng` suite of programs. In the following listing, you can see the output of `airodump-ng`. You'll see all of the BSSIDs that are within range of the system on which `airodump` is being run. This gives you the BSSID, the power level, the number of beacons, the channel number, and other details. At the bottom, you can see the stations and the SSIDs they are associated with.

Using airodump-ng to Acquire Information

```
 CH  6 ][ Elapsed: 6 mins ][ 2018-12-16 18:22

 BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

 18:D6:C7:7D:EE:11  -21      309      955    0  11  195  WPA2 CCMP   PSK  CasaChien
 C4:EA:1D:D3:78:39  -36      218       16    0   6  130  WPA2 CCMP   PSK  CenturyLink5191
 70:3A:CB:52:AB:FC  -40      211        0    0  11  130  WPA2 CCMP   PSK  CasaChien
 70:3A:CB:15:F5:A1  -41      139      121    0   1  130  WPA2 CCMP   PSK  CasaChien
 FA:8F:CA:6C:2D:6D  -42      160        0    0  11  130  OPN              Master Bedroom TV.m
 70:3A:CB:4A:41:3B  -44      211      156    0   6  130  WPA2 CCMP   PSK  CasaChien
 92:CD:B6:13:FE:3C  -55      196        0    0   6   65  OPN              HP-Setup>3c-M277
LaserJet
 70:8B:CD:CD:92:30  -65      164       17    0  11  195  WPA2 CCMP   PSK
Hide_Yo_Kids_Hide_Yo_WiFi
 EC:AA:A0:4D:31:A8  -67        0        0    0   9  195  WPA2 CCMP   PSK  HOME-C377-2.4
 A0:04:60:21:34:12  -66        0        6    0   9  -1   WPA              <length:  0>

 BSSID              STATION            PWR   Rate    Lost    Frames  Probe

 (not associated)   94:9F:3E:01:10:FB  -36    0 - 0      0       23
Sonos_lHe9qyhhveucJ2UnQWM67LjZ9g
 (not associated)   F0:18:98:0C:34:69  -38    0 - 1      0        2
 (not associated)   F0:6E:0B:69:60:1B  -40    0 - 1      0       12
 (not associated)   78:28:CA:09:8E:41  -46    0 - 0      0       45
Sonos_lHe9qyhhveucJ2UnQWM67LjZ9g
 (not associated)   94:9F:3E:00:FD:83  -46    0 - 0      0       41
Sonos_lHe9qyhhveucJ2UnQWM67LjZ9g
 (not associated)   94:9F:3E:0F:1D:81  -46    0 - 0      0       45
Sonos_lHe9qyhhveucJ2UnQWM67LjZ9g
 (not associated)   38:8B:59:20:E9:DF  -48    0 - 1      0       21
 (not associated)   90:CD:B6:13:7E:3C  -50    0 - 1      0       34   ZoeyWasHere
 (not associated)   64:52:99:50:48:94  -64    0 - 1      0       22   WifiMyqGdo
 (not associated)   C0:97:27:5A:2D:F1  -68    0 - 1      0        2
 (not associated)   34:7C:25:25:AB:92  -56    0 - 1      0        2
 18:D6:C7:7D:EE:11  38:F7:3D:04:92:79  -34    0e- 0e    22      517
 18:D6:C7:7D:EE:11  64:1C:B0:94:BA:57  -46    0 - 1e     0       17
 18:D6:C7:7D:EE:11  94:9F:3E:0F:1D:80  -46   54 -24      0      136
 18:D6:C7:7D:EE:11  F0:81:73:95:8C:47  -58    0e- 6e     0      113
 18:D6:C7:7D:EE:11  94:9F:3E:00:FD:82  -44   18 -24      0        7
 70:3A:CB:15:F5:A1  94:9F:3E:01:10:FA  -36   54 -24      0       38
 70:3A:CB:15:F5:A1  00:FC:8B:AB:0B:B8  -44    0e- 6e     0       17
 70:3A:CB:4A:41:3B  78:28:CA:09:8E:40  -46   48 -24      0       80
```

There are a couple of reasons you might want to run a deauthentication attack. One is to get a hidden Extended Service Set Identifier (ESSID). Since the ESSID isn't being broadcast, you can't retrieve it from an announcement from the access point. You can, though, get the ESSID from a station trying to associate with an access point. If you force the client into an unauthenticated state, that station will need to reauthenticate, which will allow you to capture the ESSID from the client when it reassociates.

Another reason is to capture handshakes during association. Under WEP, this traffic would allow for the retrieval of the encryption key. This means if you also captured all the encrypted traffic, you would be able to decrypt it. This was because of a weakness in the calculation of the initialization vector, a random number used to seed the key. The value ended up not being truly random, so it could be identified. This doesn't work with WPA2, though. The weakness from WEP doesn't exist in WPA2. This is not to say there aren't other ways to attack WPA2 networks.

Evil Twin

An evil twin is a rogue access point configured to look just like a legitimate access point, meaning it advertises a known SSID. You would think this was a fairly easy thing to do. You could just set up an access point and let stations associate to it. That may not help you an awful lot, though. You could get systems onto a network you control, hoping you could compromise them. You could also potentially capture unencrypted web traffic, which may include usernames and passwords. However, better would be to gather information from the stations, such as authentication information or, as needed, a PSK. This would give you a means to gain access to the enterprise network. Getting this information, though, requires more than just an access point.

Ideally, you would have an intelligent means of interacting with the stations and collecting information they may have. There are a few programs you can use that will make life a lot easier for you if you want to use an evil twin attack. One program is `wifiphisher`, which can be used to impersonate a wireless network while jamming a legitimate access point and then redirect traffic to a site managed by you. This means you don't have to worry about any encryption issues, if you want to pretend to be a commonly visited website. You don't have to try to decrypt the traffic because you can just capture it from your web pages.

Another tool you can use is `airgeddon`. This is a program that will work on Linux distributions and can be used for several types of wireless attacks. In the code listing that follows, you can see the initial menu, after it has tested your system for compatibility as well as the existence of all of the tools and libraries the program will need. Many of the attacks require the interface to be put into monitor mode, which `airgeddon` can do. Rather than putting the interface into monitor mode as part of an attack, it's a menu option. You'll also see there is a Handshake tools menu selection. This is where you could capture the handshakes done as part of association.

airgeddon Attack Menu

```
Interface wlan0 selected. Mode: Managed. Supported bands: 2.4Ghz

Select an option from menu:
---------
0.  Exit script
1.  Select another network interface
2.  Put interface in monitor mode
3.  Put interface in managed mode
---------
4.  DoS attacks menu
5.  Handshake tools menu
6.  Offline WPA/WPA2 decrypt menu
7.  Evil Twin attacks menu
8.  WPS attacks menu
9.  WEP attacks menu
---------
10. About & Credits
11. Options and language menu
```

We were talking about evil twin attacks, though. That's also a menu selection. Once you select Evil Twin Attacks Menu, you are presented with another set of menu options. Again, you can put the interface into monitor mode. You will also be able to choose different types of evil twin attacks. Just creating an evil twin access point is a good start, but if you use that attack, you will need to bring in other tools to capture data. It's easiest to let `airgeddon` just set up the complete attack for you. You can see that in addition to the evil twin, you can enable sniffing and `sslstrip`. The use of `sslstrip` will help to potentially decrypt web traffic. If this attack works, you will get web traffic in plain text. You can also use `bettercap` to perform spoofing attacks on top of just seeing wireless traffic.

This also allows you to use the Browser Exploitation Framework (BeEF) to exploit vulnerabilities in browsers. This could let you compromise a system through someone using their web browser.

Evil Twin Attack Menu

```
*************************** Evil Twin attacks menu
***************************
Interface wlan0 selected. Mode: Managed. Supported bands: 2.4Ghz
Selected BSSID: None
Selected channel: None
Selected ESSID: None

Select an option from menu:
---------
0.  Return to main menu
1.  Select another network interface
2.  Put interface in monitor mode
3.  Put interface in managed mode
4.  Explore for targets (monitor mode needed)
---------------- (without sniffing, just AP) -----------------
5.  Evil Twin attack just AP
---------------------- (with sniffing) ----------------------
6.  Evil Twin AP attack with sniffing
7.  Evil Twin AP attack with sniffing and sslstrip
8.  Evil Twin AP attack with sniffing and bettercap-sslstrip2/BeEF
------------- (without sniffing, captive portal) -------------
9.  Evil Twin AP attack with captive portal (monitor mode needed)
---------
```

One thing to note here is that `airgeddon` requires two wireless interfaces to perform an evil twin attack. It needs this second interface for DoS pursuit mode, which prevents the frequency hopping that can be a protection against wireless attacks. The second interface is also used to perform a denial-of-service attack against the legitimate access point. This keeps it from responding to the station instead of the rogue access point.

While `airgeddon` is a useful tool, there is nothing it does that is unique. Under the hood, it is using a collection of tools that you need to already have in place, like `hostapd`, a DHCP server, DNSSpoof, and a lightweight HTTP server. The advantage to `airgeddon` is that it pulls all of those tools together. This makes complex, multitool attacks considerably easier. All you need to do is run `airgeddon` rather than having to set up several tools and get all the parameters right with them running in the correct order, as needed.

Key Reinstallation

So far, you may have gotten the impression that wireless networks that are using WPA2 are invulnerable to attack, short of pretending to be the legitimate wireless network. When a station goes through a four-way handshake, one of the outcomes is a session key that gets used for encryption and decryption. It's called a *session key* because it's used for the duration of the session while the station is connected to the access point. As these are potentially vulnerable over time, they are often replaced if the connection is maintained over a long period of time. This is not the only problem with session keys. Since session keys are derived rather than known ahead of time, if an attacker can force the use of a known value during key derivation, the key can be known.

The four-way handshake is used to allow the station and the access point to derive a key based on a random value, or nonce, that is passed between the two ends. In the third message of the handshake, the client has what it needs to install the key used for the session. Since messages can get lost, especially as stations get farther from the access point, that third message can be retransmitted multiple times. We can take advantage of this in order to replay messages that have been captured previously. If the use of a known nonce could be forced, the encryption key would be known by the attacker. Since the key is known, messages can be decrypted or even injected into the communications channel.

If nonce value transmissions can be replayed, it means any key creation is vulnerable to attack. This not only means the session key between a station and an access point but also the group key. This exposes multicast and broadcast transmissions over a wireless network. Any modern wireless network, whether they are using WPA or WPA2, Enterprise or Personal, may be vulnerable to this sort of attack. As of this writing, there are not a lot of

tools available to automate the exploitation of this Key Reinstallation Attack (KRACK). There are, however, some fixes available for network elements and this is likely to no longer be an issue.

Using *mdk3/4*

Several tools exist for wireless attacks. One of these that can be used for multiple types of attacks is `mdk3`/`mdk4`. This is a tool that can be used to perform beacon flooding, authentication denial-of-service attacks, and deauthentication attacks, along with other tests. This is a tool that can be installed in Kali Linux and ParrotOS. Both versions are available in both operating systems. To get started, you need to enable monitor mode on the wireless interface. You can do this by using `airmon-ng` from the `Aircrack` toolkit. The following is the command used to turn on monitor mode on the interface `wlan0` on wireless channel 1:

```
kilroy@portnoy $ airmon-ng start wlan0 1
```

Once you have monitor mode enabled, you can start launching attacks. We can start an authentication attack on all available SSIDs on the channel you specified when you turned on monitor mode. The following is the command used to run the authentication attack using the interface `wlan0mon`:

```
kilroy@portnoy $ sudo mdk4 wlan0mon a
Waiting for targets…
Found new target AP 60:5F:8D:CD:77:A9
Found new target AP F8:BC:0E:05:BB:89
Found new target AP F8:BC:0E:05:D1:C9
Found new target AP 54:83:3A:AF:99:9D
Found new target AP 88:DA:1A:27:EB:C4
Found new target AP F8:BB:BF:7A:BF:03
Connecting Client F6:28:45:5A:18:AB to target AP 60:5F:8D:CD:77:A9 Status: No Response.
Packets sent:      1 - Speed:    1 packets/sec
```

This is a broad stroke attack, which isn't generally what you would be doing. Other attacks would be targeted at individual SSIDs. For these attacks, you'd need to provide some additional information. You could use a brute-force attack to determine whether an SSID was cloaked. It will also look for whether an SSID is available within range. The following is one of these attacks against an SSID named WubbleWiFi:

```
kilroy@portnoy $ sudo mdk4 wlan0mon p -e WubbleWiFi
AP responded on 0 of 1 probes (0 percent)
Packets sent:      1 - Speed:    1 packets/sec
AP responded on 0 of 127 probes (0 percent)
Packets sent:    128 - Speed:  127 packets/sec
```

Wireless Attacks

Use `mdk4` to perform testing on an access point in your area. You should try to perform a deauthentication attack. You can watch the attack to see how it works using Wireshark to monitor the radio traffic.

Bluetooth

It's not only network transmission that has gone wireless. Peripherals have also gone wireless. More than that, any two devices that would commonly communicate over a wire may use a wireless protocol called Bluetooth. Bluetooth is a wireless protocol allowing for short-range communication between one device and another. It operates in the set of frequencies referred to as the ISM band of 2.4 GHz. 802.11 uses a portion of the same frequency spectrum. While a Class A Bluetooth device has a range of about 100 meters due to a higher strength signal, most Bluetooth devices have a range of about 10 meters, requiring close proximity to interact with them.

Bluetooth offers a number of capabilities, described by profiles. Each Bluetooth device will implement a set of profiles, depending on the requirements for the device. For example, a Bluetooth headset would implement the Advanced Audio Distribution (A2DP) profile. This profile defines how multimedia audio may be streamed from one device to another. This is not the only profile a headset would implement, so each Bluetooth device may implement several profiles. When two devices are paired, they share their capabilities, meaning they tell each other what profiles they support.

The pairing process leads to a bond between two devices. Pairing requires verification, so each device should provide a means of indicating it expects to and wants to pair with the other device. Early devices used a simple four-digit number (PIN), which needed to be provided to indicate that the device requesting to pair knew the value. Devices that had limited input capabilities, like a headset or earpiece, would have a value hard-coded. You would be provided with the correct value as part of the documentation for the device. Later versions of Bluetooth expanded the capabilities available to verify pairing requests.

With Bluetooth v2.1, Simple Secure Pairing (SSP) was introduced. There were four mechanisms to complete a pairing request with this version, outside of the compatibility required to support older devices. The first was called Just Works because the pairing was supposed to just work without any interaction on the part of the user. If both ends have the means to take input, they can support numeric comparison. This is where each end presents a six-digit numeric value. If the two values match, the user can indicate that to be the case and the two endpoints can complete the pairing process.

In some cases, a peripheral may only be able to provide input but not output, such as, for example, a Bluetooth keyboard. When you pair that with your computer, your computer may prompt you to enter a value on the keyboard. If the values match, the pairing process completes. Again, this would be a six-digit value. Finally, you can do an out-of-band pairing. This would require another protocol to assist in the pairing process. For example, near-field communication (NFC) is sometimes used. You may have a device that expects you to bring another device like a smartphone or tablet close. Once the two devices are close, you can use an app on the smartphone or tablet to complete the pairing and setup process.

There are a few ways to attack Bluetooth. Keep in mind that no matter what you do, you need to be within close proximity of the device you want to attack. This can be easy in public places, though. Some of these attacks are bluejacking, bluesnarfing, and bluebugging.

Scanning

Before we go too far down the road of attacks, we should go over how you would determine where potential victims may be. You could just go through the process of adding a Bluetooth device, and your system will scan for devices. However, you can also make use of special-purpose software. This could be done using a program like `btscanner`. With `btscanner`, you can do both an inquiry scan and a brute-force scan. With an inquiry scan, your system puts itself into an inquiry scan state. In this state, your device listens for inquiries from other devices. There are 32 channels, and your Bluetooth receiver has to spend time listening on every one of them. In the following listing, you can see `btscanner` performing an inquiry scan.

Inquiry Scan Using btscanner

```
 |Time                  Address            Clk off  Class     Name
 |
 |2022/12/25 17:58:22  64:1C:B0:94:BA:58  0x2c42   0x0c043c  (unknown)
 ||2022/12/25 17:56:07  00:9E:C8:93:48:C9  0x38b3   0x0a0110  (unknown)
 ||
 |
 |
 |
 |
 |
 |
 |
 |
 |
 |
 |_____________________________________________________________________________
_____|
 | enter=select, Q=quit
|
 |starting inquiry scan
 |Found device 00:9E:C8:93:48:C9
 |Found device 64:1C:B0:94:BA:58
 |
```

The other type of scan is a brute-force scan. Whereas an inquiry scan has the receiver listening for inquiry, a brute-force scan actively sends messages to devices to determine what they are. To do this, it needs addresses to send messages to. This type of connection happens at layer 2, which means we are talking about MAC addresses. In the following listing, you can see the use of `btscanner` to perform a brute-force scan. To run this scan, you need to tell `btscanner` the starting address and the ending address you want to scan, which you can see being asked for. You would need to provide this in the form of a MAC address. Keep in mind that each octet has 256 values, so the more octets of the address you want to run through, the longer your scan will take.

btscanner Starting a Brute-Force Scan

```
Time                  Address            Clk off  Class     Name
2023/03/02 18:39:47  A8:51:AB:9C:66:ED  0x0000   0x000000  kilroy4KTV
```

```
Scanned device A8:51:AB:9C:66:FC (99%)
Scanned device A8:51:AB:9C:66:FD (99%)
Scanned device A8:51:AB:9C:66:FE (100%)
Scanned device A8:51:AB:9C:66:FF (100%)
```

This type of scanning will get you lists of devices. It won't tell you the profiles that the devices support. Just having the list of devices nearby may be enough to run some of the attacks commonly used against Bluetooth devices.

Bluejacking

Bluejacking is when an attacker sends data to a Bluetooth device without having to get through the pairing process, or perhaps the pairing happens without the receiver knowing about it. You could use a bluejacking attack to send an unsolicited message to a victim. This might be a picture or a text message. This could be a spoof attack, where you send a message that appears to be from someone else in order to get the recipient to do something. This attack uses the Object Exchange (OBEX) protocol to move the message or picture from one device to another.

Since this is just a one-way message, alone it isn't harmful. The harm from this sort of attack comes from getting the device owner to do something you want them to do. This may not be beneficial for them, though.

Bluesnarfing

Bluesnarfing is considerably more dangerous than bluejacking. Whereas bluejacking is sending data to a device, bluesnarfing is getting data from a device. Bluetooth devices have to be exposed to a certain degree to allow other devices to begin a pairing process. This opens up the possibility of another device taking advantage of that little window. It's been possible to gain access to a device over Bluetooth without having gone through the pairing process. If an attacker can gain access to someone's smartphone or another Bluetooth-enabled device with sensitive data on it, they may be able to extract that data.

As with other Bluetooth attacks, the attacker has to be within close proximity of the device they want to gain access to. There is a variation, however, called *long-distance bluesnarfing*, which can allow the same sort of attack from a greater distance.

A bluesnarf++ attack is similar to a bluesnarf attack but differs in the way the attacker gets access to the file system on the victim device. Bluesnarfing works over the OBEX protocol. If there is a file transfer protocol (FTP)

server running over OBEX, it is possible to gain access to the remote system without any authentication. This is because of the OBEX Push service.

Bluebugging

It's likely you're aware of the concern of listening devices around us. Amazon's Echo, Google's Home, and other similar devices are generally in a “listen all the time” mode. A bluebugging attack is something similar and was available long before we started willingly installing these other devices into our living and working spaces. A bluebugging attack uses Bluetooth to gain access to a phone in order to place a phone call. Once the phone call is placed, the attacker has a remote listening device. The initial attack has to be done in close proximity to the target, but the phone call will continue to provide a remote listening point no matter where the victim and attacker are in relationship to one another.

Bluebugging is, perhaps fortunately or unfortunately depending on which end you are on, easily identified and defeated. At least when the phone call is placed, it's obvious to the person who has the device. All they have to do is look at their device. They can then hang up. Of course, remediating the vulnerability that gave access to the attacker to begin with is a different story.

Bluebugging attacks, as well as the others, are old. This means they are less likely to be successful against modern devices, since vulnerabilities that can allow attackers easy access have generally been remediated in the intervening years since these attacks were identified. This is not to say they are irrelevant, however. There are still older devices in use that may allow these attacks to be successful.

Bluedump

A bluedump attack requires some knowledge on the part of the attacker but can allow a target system to trust an attacker system. The attacker has to know the BDADDR (hardware address) of a trusted device. The attacker spoofs that address in a connection attempt. The target device will respond with an authentication request. The attacker device responds with an HCI_Link_Key_Request_Negative_Reply message. This may cause the target device to delete the existing key used to authenticate the device being spoofed. The target device then goes into pairing mode, which would allow the attacker to pair with the target device, making the attacker's system a trusted device.

Bluesmack

This is a denial-of-service attack that can occur when the attacker uses the Logic Link Control and Adaptation Protocol (L2CAP) that is used to assess the connection and measure the time it takes for data to get from one device to another and back again (the round-trip time). An attacker can manipulate the size of the packets being sent. With the right size, the target device ends up being unusable.

Mobile Devices

Bluetooth attacks are one way to gain access to mobile devices, and Bluetooth is nice because you don't need to have physical access; you just need to be proximal to the victim. This allows you to work through the air rather than doing something physically. Given the ubiquitous nature of mobile devices, it would make sense that they would be high-value targets for attackers. While many attacks rely on specific vulnerabilities in applications and operating systems, there are some attacks that are common across mobile platforms. Because they work across multiple platforms, they are more classes of attacks than specific exploits.

There are currently two predominant platforms for mobile devices, though that doesn't mean there are only two platforms. It's also made more complicated because one of the platforms is Android. Android is software developed and managed by Google, but the hardware that runs Android is developed by numerous manufacturers. Google provides the source code for Android to vendors, and each vendor may implement it in different ways, altering the code and/or adding software and features. This means the Android platform is splintered and fragmented, not to mention the large number of Android versions that are in the wild. Android runs on smartphones and tablets. There is also a version of Android that runs on smart wearables like watches.

The other platform that is common is iOS, developed by Apple for the iPhone and iPad. iOS is a stripped-down version of macOS. The advantage to iOS from our perspective here is that Apple does a lot to drive users to the

latest software versions, including making it hard to run newer operating systems on older hardware. This means there are far fewer versions of iOS in the wild than there are of Android. There are two sides to this coin. There is a better chance that iPhone/iPad OS installs are closer to being up-to-date, with fewer known vulnerabilities than Android installations are subject to.

One of the most disturbing vectors for threats to mobile devices is through applications. Android comes with an application marketplace called the Google Play Store. Vendors like Samsung may also have their own application marketplaces as another option for applications to be installed on a mobile device. There are also third-party app stores that can be configured on devices. Allowing this is a setting on the Android platform. By default, it's generally disallowed, but users can bypass that protection if they wish.

Apple has its own marketplace. You can see the Apple App Store in Figure 11.10. iOS is not an open platform like Android. There is no way to configure additional marketplaces or install applications without breaking the operating system. This process, commonly called *jailbreaking*, has become nontrivial as Apple (and Google and Samsung) have found ways to fix vulnerabilities that allowed for the jailbreaking to happen. Users who may be inclined to jailbreak a phone may also be likely to install illegitimate software.

Mobile Device Attacks

Third-party applications are the best way to get access to mobile devices. Both Google and Apple vet applications that are submitted to their app marketplaces. This is not a perfect process, however. There have been applications that have made it through this vetting process in both marketplaces that are malicious in one form or another. Sometimes legitimate software even crosses lines, such as when Trend Micro software was pulled because of the data it was found to be collecting. The problem is even worse with application marketplaces that don't vet software—meaning running the software through tests and analysis. Getting software into a third-party marketplace is a way to insert spyware and other malware onto mobile devices.

FIGURE 11.10 Apple App Store

Many apps regularly collect data, either with the user's awareness or without. This data can be transmitted back to storage controlled by the app's owner. Most applications use frameworks provided by the operating system vendor (e.g., Apple or Google). These frameworks might commonly be used to send data over an HTTP connection, and they may even force encryption of that connection. There is no guarantee of that encryption, though, which means data can be leaked if the application collects data from the user and then sends that data to a remote storage location. This data leakage is potentially a serious problem, and it might also be used by an attacker to gather details that could be used for other attacks.

As mentioned, wireless networks aren't always protected. There may be rogue networks or unencrypted networks. This is another place where data leakage can happen. Using techniques shown earlier in this chapter, data can be collected over wireless networks. This may be especially useful in public places, where people are likely to be attached to wireless networks from their mobile devices. They may also be likely to be doing things like checking enterprise email or interacting over enterprise communications platforms like Microsoft Teams, Slack, or HipChat.

Some other common attacks against mobile devices are the same as those you would think about for traditional desktop systems. A mobile device may have a different processor, and the operating systems do provide some protections for applications so they don't interact with other applications on the device. This is a technique called

sandboxing. Phishing attacks are just as possible on mobile devices. Malware is possible on mobile devices, just as it is on desktop systems. Malware can include spyware, collecting information from the mobile device.

Bad programming is always an issue. No matter how much testing is performed, between the software vendor and the platform vendor, bugs happen. In some cases, bad encryption mechanisms may be used. This may happen when programmers think there are better ways to encrypt than by standard means. It's commonly felt that openly available encryption is best because it gets to be analyzed and tested by a lot of people who are the best in their fields.

Another possibility from bad programming practices is improperly handling session data. Session data is used to provide information about the state of the application communications. It lets the client and server know where they are in the application flow. If session information isn't handled properly, sessions can be impersonated. They may also be replayed. This may be especially possible in combination with weak encryption.

As more users bring their own devices into the workplace, it can be harder to protect the enterprise because not every avenue of attack can be closed down on a personal device. The company needs to protect its assets, but the smartphone isn't a company asset. There is an interesting intersection here, which attackers can take advantage of.

You don't have to go really fancy with your mobile device attacks, though. You can go old-school. Smishing attacks, short for Short Message Service (SMS) phishing, use text messages to attack users. Figure 11.11 shows an example of a real-life smishing message. Attackers are returning to these old-school attacks. You can see some of the hallmarks of this being an attack message. Sometimes, you will see hallmarks of using the mixed case hTtP because it's thought to be a way of evading detection. This message doesn't have that indicator. Old intrusion detection systems didn't take mixed case indicators into account and would miss malicious messages that weren't exactly as you'd expect (either HTTP or http, for instance). This is a scare tactic, though, which is an indicator it is a malicious message. Especially with so much news about vulnerabilities, this could be the type of message that would lure in someone who wasn't very sophisticated technically.

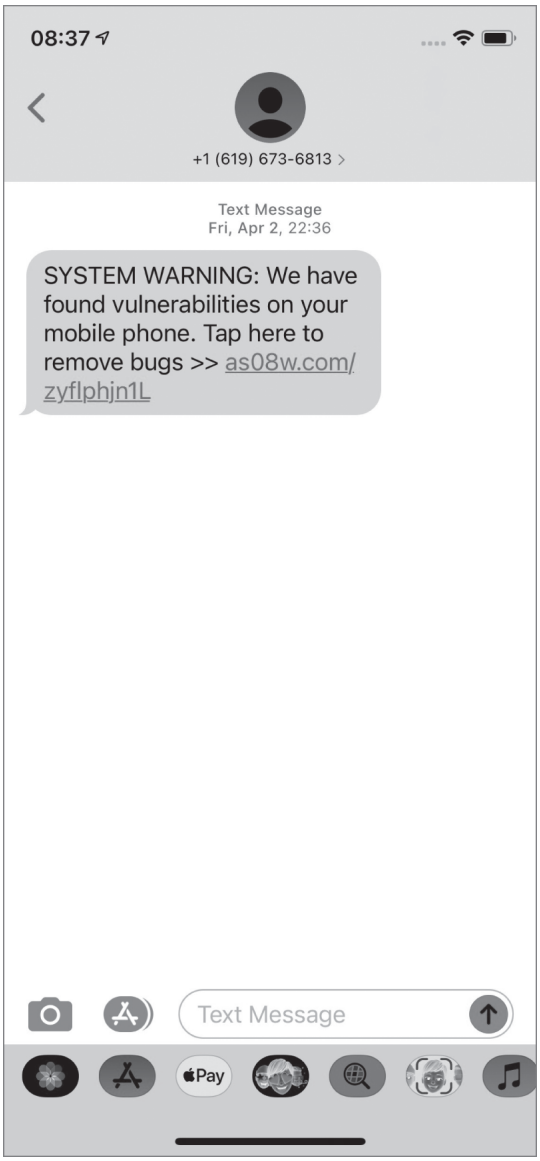


FIGURE 11.11 Smishing message

Source: Courtesy of Rick Read

The domain name here is highly suspicious, though sometimes you will see URLs that are even more suspicious. Sometimes, the URL will include a raw IP address. A smishing message received a few months ago included a URL with a raw IP address. The URL used 82.118.21.203 instead of a domain name or a fully qualified domain name. In case you are questioning, though, you can do a simple lookup of this IP address, or even domain names in case you are concerned about them. In this case, the IP address maps to Russia. While this in and of itself isn't suspicious, it wouldn't be a source of a website T-Mobile was using to get customers to visit using a text message, which was the company the text message was purporting to come from. The following was the hostname lookup for this IP address, at the time of the receipt of the text message. In the intervening months, during edits to the book, the hostname has changed. You can see the new name after the original name.

```
kilroy@milo $ host 82.118.21.203
203.21.118.82.in-addr.arpa domain name pointer kamal2.bakkali59.pserver.ru.

kilroy@milo $  host 82.118.21.203
203.21.118.82.in-addr.arpa domain name pointer nebrosika4.isplevel.pro.
```

Summary

Wireless networks are common today, especially as mobile devices become more ubiquitous. There are two common types of wireless networks. The first is an ad hoc network where the stations organize themselves into the network. A station is an endpoint in a wireless network. If there is an access point that the stations connect to, it's an infrastructure network. When stations connect to an access point, they have to authenticate. There may be open authentication, which means anyone can join the network without needing to know a password. Modern wireless networks use WPA for encryption and WPA networks use a four-way handshake to authenticate the station and also derive the encryption key.

WEP was the first attempt at encrypting wireless communications. Unfortunately, the initialization vector, a random value meant to seed the encryption, wasn't actually random and could be guessed, leading to key leakage. The second pass was meant as a stopgap, and that was WPA. Eventually, the final replacement for WEP came out, called WPA2. WPA and WPA2 support both personal and Enterprise authentication. Personal authentication uses a pre-shared key. Enterprise authentication can support username and password authentication. WPA3 was introduced in 2018, which included the use of stronger encryption keys as well as the use of Simultaneous Authentication of Equals to better protect the transmissions between the station and access point.

Wireless networks can be attacked. In networks that use WEP, the encryption key could be derived if enough frames were collected from the wireless network. Common sniffing applications like Wireshark can be used to sniff wireless networks. The wireless interface needs to be put into monitor mode to collect the radio headers. Wireshark may be capable of doing this on its own, depending on the network interface. You can also use the `airmon` program from the `aircrack-ng` suite. You can also use `aircrack-ng` to try to crack the password used for the network.

One way to gather enough data is to use a deauthentication attack. This attack sends messages to clients telling them they have been deauthorized on the network. The attacking system spoofs the messages to appear as though they are coming from the access point. Simultaneously, the attacker may attempt to jam the access point so it doesn't get and, more important, respond to the station. The attacker may continuously send these disassociation messages to force the victim station to keep sending messages trying to authenticate against the network.

Evil twin attacks are where the attacker establishes a network that appears to be a legitimate network with the same SSID. Stations would attempt to authenticate against the rogue access point. The rogue access point could then collect any authentication credentials, and once the station had passed those along, the evil twin would allow the connection, gathering traffic from the station once the station was connected.

Another attack is the key reinstallation attack. This is an attack where traffic is reused to force a station to use a key that's already known. This means the attacker knows what the key is, meaning traffic can be decrypted.

Of course, 802.11 is not the only wireless communications protocol. Bluetooth is often used. There are several Bluetooth attacks, though some of them may be outdated and won't work on modern devices. But there are plenty of legacy devices that use older Bluetooth implementations where these attacks can work. Bluejacking is using Bluetooth to send a message to a target system. Bluesnarfing is collecting data from a target system using Bluetooth. Bluebugging is using Bluetooth to connect to a target system and then initiating a call out from the target system so conversations can be bugged.

Mobile devices commonly use wireless protocols like 802.11 and Bluetooth. There are ways to inject malicious applications onto mobile devices like tablets and smartphones. This may be done using a third-party app store that users may be convinced to add as an option to their device. Mobile devices are also vulnerable to the same sorts of attacks as desktops, such as phishing, malware, and programming errors that can open the door for data leakage.

Review Questions

You can find the answers in the appendix.

1. What are the two types of wireless networks?

    A. Star and ring

    B. Bus and hybrid

C. Infrastructure and hybrid

D. Infrastructure and ad hoc

2. How many stages are used in the WPA handshake?

A. Two

B. Four

C. Three

D. One

3. What mode has to be enabled on a network interface to allow all headers in wireless traffic to be captured?

A. Promiscuous

B. Monitor

C. Radio

D. Wireless LAN

4. What wireless attack would you use to take a known piece of information to be able to decrypt wireless traffic?

A. Sniffing

B. Deauthentication

C. Key reinstallation

D. Evil twin

5. What is the purpose of performing a Bluetooth scan?

A. Identifying open ports

B. Identifying available profiles

C. Identifying endpoints

D. Identifying vendors

6. What is the purpose of a deauthentication attack?

A. Disabling stations

B. Forcing stations to reauthenticate

C. Reducing the number of steps in the handshake

D. Downgrading encryption

7. What is the policy that allows people to use their own smartphones on the enterprise network?

A. Bring your own device

B. Use your own device

C. Bring your own smart device

D. Use your own smart device

8. What part of the encryption process was weak in WEP?

A. Keying

B. Diffie-Hellman

C. Initialization vector

D. Seeding vector

9. What is the WPA four-way handshake used for?

A. Passing keys

B. Deriving keys

C. Encrypting messages

D. Initialization seeding

10. What is the SSID used for?

A. Encrypting messages

B. Providing an IP address

C. Identifying a network

D. Seeding a key

11. What kind of access point is being used in an evil twin attack?

A. Infrastructure

B. Ad hoc

C. WPA

D. Rogue

12. How does an evil twin attack work?

A. Phishing users for credentials

B. Spoofing an SSID

C. Changing an SSID

D. Injecting four-way handshakes

13. What is a method to successfully get malware onto a mobile device without having to get the user to do something they wouldn't normally do?

A. Using the Apple Store or Google Play Store

B. Using external storage on an Android

C. Using a third-party app store

D. Jailbreaking

14. What would you use a bluebugging attack for?

A. Identifying Bluetooth devices nearby

B. Listening to a physical space

C. Enabling a phone's camera

D. Gathering data from a target system

15. What would a signal range for a Class A Bluetooth device commonly be?

A. 300 ft.

B. 3,000 ft.

C. 75 ft.

D. 500 ft.

16. What tool could you use to enable sniffing on your wireless network to acquire all headers?

A. Ettercap

B. `tcpdump`

C. `aircrack-ng`

D. `airmon-ng`

17. Why is bluesnarfing potentially more dangerous than bluejacking from the standpoint of the victim?
    A. Bluejacking sends while bluesnarfing receives.
    B. Bluejacking receives while bluesnarfing sends.
    C. Bluejacking installs keyloggers.
    D. Bluesnarfing installs keyloggers.
18. What tool would allow you to run an evil twin attack?
    A. Wireshark
    B. Ettercap
    C. `wifiphisher`
    D. `aircrack-ng`
19. What types of authentication are allowed in a WPA-encrypted network?
    A. Handshake and personal
    B. Personal and enterprise
    C. Enterprise and handshake
    D. 802.11 and personal
20. What wouldn't you see when you capture wireless traffic that includes radio headers?
    A. Capabilities
    B. Probe requests
    C. SSIDs
    D. Network type
21. What does WPA3 use to start the authentication and association process between stations and access points?
    A. Four-way handshake
    B. Mutual authentication of peers
    C. Simultaneous authentication of equals
    D. Separate authentication with encryption
22. Which of these Bluetooth attacks can result in a denial of service?
    A. Bluejacking
    B. Bluesnarfing
    C. Bluedumping
    D. Bluesmack
23. What technique would you likely be using if you had a Pringles can along with a laptop?
    A. Bluesnarfing
    B. Wireless footprinting
    C. Evil twin
    D. Key reinstallation
24. What piece of information would you need to have to perform a Bluedump attack?
    A. BDADDR
    B. BDIP

C. IPADDR

D. DUMPADDR

25. What type of building material would you select to keep the wireless signal mostly in the building, rather than leaking out?

A. Plywood

B. Glass

C. Sheetrock

D. Concrete

OceanofPDF.com