generate logs that are sufficient to detect attacks or errors. Ideally, these logs are
centrally collected to make them easier to correlate and harder to tamper with.
• Shared responsibility The security obligations of the organization and of the supplier should be codified in a legally binding contract and audited periodically.
Again, the list is not exhaustive, but it should give you an idea of how the secure
design principles can be applied to a web services scenario. You should be prepared to do
likewise with a variety of other scenarios for the CISSP exam.
How are these web services actually delivered? The key is to focus on what service is
being delivered, and not on how it is implemented or where it is hosted (as long as it is
available). A service-oriented architecture (SOA) describes a system as a set of interconnected

but self-contained components that communicate with each other and with their clients through standardized protocols. These protocols, called application programming
interfaces (APIs), establish a “language” that enables a component to make a request
from another component and then interpret that second component’s response. The requests that are defined by these APIs correspond to discrete business functions (such
as estimated shipping costs to a postal code) that can be useful by themselves or can be
assembled into more complex business processes. An SOA has three key characteristics:
self-contained components, a standardized protocol (API) for requests/responses, and
components that implement business functions.
SOAs are commonly built using web services standards that rely on HTTP as a standard communication protocol. Examples of these are SOAP (which used to stand for the Simple Object Access Protocol) and the Representational State Transfer (REST)
architectures. Let’s look at these three (HTTP, SOAP and REST) in turn.

Hypertext Transfer Protocol

NOTE A cookie is just a small text file containing information that only one website can write or read.

Uniform Resource Identifiers A foundational component of HTTP is the use of the uniform resource identifier (URI), which uniquely identifies a resource on the Internet.
A typical URI looks like this:
http://www.goodsite.com:8080/us/en/resources/search
.php?term=cissp. Let’s look at its components in sequence:
1. Scheme This is another name for the protocol being used (e.g., HTTP or

HTTPS) and ends in a colon (:).
2. Authority There are three possible subcomponents here, but the second is the most prevalent:

• Username (optional) (and optional password, separated by a colon) followed by an at (@) symbol.
• Host in either hostname (e.g., www.goodsite.com) or IP address format.
• Port number (optional), preceded by a colon (e.g., :8080). Note that port 80 is assumed for HTTP schemes and port 443 for HTTPS schemes.

PART IV

HTTP is a TCP/IP-based communications protocol used for transferring resources (e.g., HTML files and images) between a server and a client. It also allows clients to send queries to the server. The two basic features of HTTP are that it is connectionless and stateless. Connectionless protocols do not set up a connection (obviously) and instead send their messages in a best-effort manner. They rely on some other protocol (in this case TCP) to ensure the message gets across. Stateless means that the server is amnesiac; it doesn't remember any previous conversations with any clients. Thus, whatever is needed for the server to "remember" has to be provided with each request. This is a role commonly played by session identifiers and cookies.

3. Path The path to the requested resource on the server. If the path is not specified by the client, it is assumed to be a single slash (/), which is the default document at the root of the website (e.g., the homepage). Subdirectories are indicated as they are in Linux/Unix by successive slashes (e.g., /us/en/resources/search.php).
4. Query (optional) An attribute-value pair preceded by a question mark (?) (e.g., ?term=cissp). Each additional pair is separated from the previous one by an ampersand (&).

Request Methods HTTP uses a request-response model in which the client requests one or more resources from the server, and the latter provides the requested resources (assuming, of course, they are available to the client). The protocol defines two request methods: GET and POST. The main difference for our discussion is that a GET request must include all parameters in the URI, while POST allows us to include additional information (e.g., parameters) in the body of the request, where it will not be

revealed in
the URI. So, in the previous example we can safely guess that the method used was GET
because we see the search term (cissp) in the URI.
Hypertext Transfer Protocol Secure HTTP Secure (HTTPS) is HTTP running over Transport Layer Security (TLS). Ensuring that all your web services require HTTPS
is probably the most important security control you can apply to them. Recall that
unencrypted requests can provide an awful lot of sensitive data, including credentials,
session IDs, and URIs. Ideally, you require TLS 1.3 on all your web servers and ensure
they do not allow unencrypted communications (by enforcing secure defaults).
An important consideration before you jump to HTTPS everywhere is whether you want to perform deep packet analysis on all your internal traffic. If you force use of
HTTPS, you will need to deploy TLS decryption proxies, which can be pricey and require
careful configuration on all your endpoints. The way these proxies work is by performing what is essentially a (benign) man-in-the-middle attack in which they terminate the
clients' secure sessions and establish the follow-on session to their intended server. This
allows the proxy to monitor all HTTPS traffic, which provides a measure of defense in
depth but may pose some challenges to the privacy by design principle. Many organizations deal with this challenge by whitelisting connections to certain types of servers (e.g.,
healthcare and financial services organizations), while intercepting all others.

SOAP
SOAP is a messaging protocol that uses XML over HTTP to enable clients to invoke processes on a remote host in a platform-agnostic way. SOAP was one of the first SOAs
to become widely adopted. SOAP consists of three main components:

- A message envelope that defines the messages that are allowed and how they are to be processed by the recipient
- A set of encoding rules used to define data types
- Conventions regarding what remote procedures can be called and how to interpret
their responses

Extensible Markup Language
The term XML keeps coming up for good reasons. Extensible Markup Language is a popular language to use if you want to mark up parts of a text document. If you've
ever looked at raw HTML documents, you probably noticed the use of tags such as <title>CISSP</title> to mark up the beginning and end of a page's title. These tags

enable both humans and machines to interpret text and process it (such as rendering it in a web browser) as the author intended. Similarly, XML enables the author of a text document to “explain” to a receiving computer what each part of the file means so that a receiving process knows what to do with it. Before XML, there was no standard way to do this, but nowadays there are a number of options, including JavaScript Object Notation (JSON) and YAML Ain’t Markup Language (YAML).

Representational State Transfer

Unlike SOAP, which is a messaging protocol, Representational State Transfer (REST) is an architectural pattern used to develop web services using a variety of languages. In REST, HTTP is used to provide an API that allows clients to make programmatic requests from servers. For example, a client of a RESTful service could insert a new user record using the HTTP POST method (which lets you send additional information in the body of the request) by sending the following URI:
https://www.goodsite.com/UserService/Add/1.
The server would know to read the body of the POST to get the new user’s details, create it, and then send a HTTP confirmation (or error). As you can see, REST essentially creates a programming language in which every statement is an HTTP URI.
Because every interaction with the system is spelled out in the URI, it is essential to use HTTPS as a secure default communications protocol. Of course, in keeping with the principle of zero trust, we want to authenticate clients and servers to each other, as well

PART IV

SOAP security is enabled by a set of protocol extensions called the Web Services Security (WS-Security or WSS) specification, which provides message confidentiality, integrity, and authentication. Note that, in keeping with HTTP’s stateless nature, the focus here is on message-level security. Confidentiality is provided through XML encryption, integrity through XML digital signatures, and single-message authentication through security tokens. These tokens can take on various forms (the specification is intentionally broad here), which include username tokens, X.509 digital certificates, SAML assertions, and Kerberos tickets (we’ll cover the last two in Chapter 17).
One of the key features of SOAP is that the message envelope allows the requester

to describe the actions that it expects from the various nodes that respond. This feature supports options such as routing tables that specify the sequence and manner in which a series of SOAP nodes will take action on a given message. This can make it possible to finely control access as well as efficiently recover from failures along the way. This richness of features, however, comes at a cost: SOAP is not as simple as its name implies. In fact, SOAP systems tend to be fairly complex and cumbersome, which is why many web service developers prefer more lightweight options like REST.

as put limits on what resources are available to each client. Another good security practice for RESTful services, which applies to any software system, is to validate all inputs before processing them. This mitigates a large number of possible injection attacks in which the adversary deliberately provides malformed inputs in order to trigger a system flaw.

Domain Name System

We covered the Domain Name System (DNS) in a fair amount of detail back in Chapter 11. Let’s return to it now in the context of its role in helping us to secure our networks. Early on in its history, DNS was most commonly targeted by attackers to hijack requests, redirecting the unwitting requesters to malicious hosts instead of the legitimate ones they were seeking. While this is still a concern that we’ll address in a bit, we also have to consider the much more common use of DNS to assist threat actors in conducting attacks, rather than being the target of attacks.

Since some of the most problematic adversarial uses of DNS depend on how this system works, let’s review the process by which DNS performs recursive queries. Recall from Chapter 11 that a recursive query means that the request can be passed on from one DNS server to another one until the DNS server with the correct information is identified. This is illustrated in Figure 13-6. First, the client queries its local DNS server, which may either be an authoritative source for it or have cached it after some other client’s request. Failing that, the server will typically start by consulting the root DNS server. The root server (there are actually a few of them for redundancy) will probably

say something like “No, but here is the address of the name server for all .com domains.”
The local server will then query that server, which will probably result in it responding
“No, but here is the address of the name server responsible for ironnet.com.” Finally, the
local server will query that other server, which will respond with an A record containing
the IP address of the www host.

Response:
“Ask com”

?

.com

et
onn

w.ir

Query for
www.ironnet.com

Is this host in
my cache?

Do

you

w
kno

ww

DNS root server

Do you know www.ironnet.com?

Do y

ou k

Client

Local DNS
server

now

.com DNS server

www

.iron

net.c

om?

Response:
A 10.0.0.7
ironnet.com
DNS server

Figure 13-6

A recursive DNS query

Response:
“Ask ironnet.com”

Preventing Common DNS Attacks
DNS is the Internet’s ubiquitous messenger; its queries and responses go everywhere,
and without them the Internet as we know it would not work. Because of its importance
to most other network systems, DNS traffic is seldom blocked by firewalls or routers.
Attackers quickly figured out that this ubiquity makes DNS a preferred tool to manipulate and use for their own nefarious purposes. Perhaps the cleverest application of DNS
for unintended purposes is its use to reach out and touch hosts in ways that are difficult
to block using pseudo-randomly generated domain names.
EXAM TIP You will not be tested on the material that covers the following DNS attacks, but note that these attacks are both important to know
and illustrative of the challenges we face in securing networks. If you are preparing for the exam only, feel free to move to the “Domain Name System Security Extensions” section.

PART IV

Domain Generation Algorithms Once malware is implanted on target systems,
the adversaries still need to communicate with those hosts. Since inbound connection
attempts would easily be blocked at the firewall, most malware initiates outbound
connections to the attacker’s command and control (C2) infrastructure instead. The
problem for the attackers is that if they provide a hostname or IP address in the malware,
defenders will eventually find it, share it as an indicator of compromise (IOC),

and
reduce or negate the effectiveness of the C2 system.
To bypass signature detection by intrusion detection systems (IDSs) and intrusion
prevention systems (IPSs) that use these IOCs, malware authors developed algorithms
that can generate different domain names in a manner that appears random but produces
a predictable sequence of domain names for those who know the algorithm. Suppose I
am an attacker and want to hide my real C2 domains to keep them from being blocked
or removed. I develop a domain generation algorithm (DGA) that produces a new
(seemingly) random domain name each time it is run. Sprinkled somewhere in that
(very long) list of domains are the ones I actually want to use. The infected host then
attempts to resolve each domain to its corresponding IP address using DNS. Most of the
domains do not exist and others may be benign, so either way there is no malicious C2
communications that follow. However, since I know the sequence of domains generated
by the DGA and I know how quickly the malware will generate them, I can determine
approximately when a particular infected host will query a specific domain. I can then
register it the day before and rendezvous with the malware on that domain so I can
receive its report and/or issue commands. The defenders won’t know which domains are
my malicious ones and which are just noise meant to distract them.
Figure 13-7 shows three domains being generated by an infected host. The first two
that are queried do not exist, and thus result in an NXDOMAIN response from the
server, which means the domain was not found. The third domain resolves to a malicious
domain. When the authoritative (malicious) server for that domain receives the request,
it knows it comes from a compromised system and sends a response that, when decoded,
means “sleep for 7 hours.”

DNS server

Compromised system
7lybodzg6ka5w3qc1.com
NXDOMAIN
gln35e9s82.info
NXDOMAIN
x5bnb2gxs.org
10.0.0.6

Checking in…
Sleep for 7 hours
x5bnb2gxs.org
10.0.0.6

Figure 13-7

DGA in use by a compromised system

How can we detect and stop this kind of adversarial behavior? There are two general approaches. The first is to capture the malware and reverse engineer its DGA. We then play it forward (just like the attacker does) to determine which domains will be generated and when. Knowing this timeline, you can blacklist the domains and use the fact that a host attempted to reach them to infer that the querying system is compromised. Keep in mind that different compromised systems will be generating domain names at different times, so the task is onerous even for organizations that are mature enough to reverse engineer malware in the first place.

The second approach to detecting and stopping the use of DGAs is to analyze the domain names in each query to determine the probability of the query being legitimate.

You can see from Figure 13-7 that most domains generated by these algorithms look, well, random. They are not the sort of domain names that you would expect someone to pay money to register. If you find a domain that is highly suspicious, you can investigate the host to see if it is infected, or you could block or monitor the DNS query and response to see if there is anything suspicious in either. For example, in some cases, the response will come as an encoded or encrypted message in a TXT record. This approach is only practical if you have a fairly sophisticated artificial intelligence analysis system that can examine every DNS request and learn over time which ones are likely to be bad.

NOTE There are legitimate uses of DGAs. For example, some systems use them to test whether or not a system can reach the Internet and perhaps track who that system is. This is done by some developers for licensing, updating, or diagnostic purposes.

Attacker’s
C2 server

Response
“Ask com”
DNS root server

Query for
SUQ6MTIzNEBhY21l

. g00dsite.com

Is this host in
my cache?

Response:
“Ask g00dsite.com”
.com DNS server

Compromised system
Base64 encoded payload:
ID:1234@acme

Local DNS
server

Response:
TXT V2lwZSB0YXJnZXQh
g00dsite.com
Base64 encoded payload:
Wipe target!

Figure 13-8

Covert communication over a DNS tunnel

Distributed Denial of Service The third type of DNS attack targets someone else’s
infrastructure using your DNS servers. An attacker who owns (or can rent) a large army

PART IV

DNS Tunneling Malicious use of a DGA can be very hard to stop unless you have advanced capabilities at your disposal. Fortunately, however, this use is limited to
simple messaging between a compromised host and an external threat actor. But what if
we could use DNS to transfer more information? A lot more? It turns out that data can
be hidden in DNS queries using encoded host and other resource labels. DNS tunneling
is the practice of encoding messages in one or a series of DNS queries or responses for
exfiltrating or infiltrating data into an environment.
Figure 13-8 shows a very simple example of DNS tunneling that builds on our

discussion of recursive queries in Figure 13-6. In this case, the compromised system wants to check in with its C2 server, so it uses Base64 encoding to obfuscate its message, which contains its identifier. Let’s say that this is an infected host in the Acme Corporation, so its ID is 1234@acme. The recursive DNS query eventually is sent to the server that owns the malicious domain g00dsite.com. It decodes the hostname field, sees which of its bots this is from, and decides it is time to wipe the file system on the infected system. This command comes in the form of a TXT response that is also Base64 encoded.

A similar, but much less noticeable, use of DNS tunneling is to slowly exfiltrate data from the compromised system. Since DNS allows names of up to 63 characters between each dot, attackers can break down a longer file (e.g., a secret document) and exfiltrate it in a sequence of DNS queries to the same server or different servers.

Defending against DNS tunneling is similarly difficult to countering DGAs. Again, we could use network detection and response (NDR) solutions that use artificial intelligence to look for this type of behavior. However, because this type of attack (unlike DGAs) tends to rely on just a few domains, we could use domain reputation tools to determine whether any of our systems are making queries for suspicious or malicious domains.

of compromised systems (bots) can use them to overwhelm a target with name resolution responses to queries it didn’t send out in the first place. To see how this attack works, we must first consider that DNS is based on UDP, which means spoofing the source address of a query is trivial.

In a DNS reflection attack, the threat actor instructs each bot they control to send a query to one of many open DNS servers around the world, while spoofing the source addresses on those queries. Collectively, the responding servers then bombard the target with traffic. If you have a sufficient number of bots and servers doing this quickly enough, the results could take the target system offline. Even if the target is not a DNS server, it still has to process millions (or more) of UDP packets arriving each second, which can overwhelm the typical server. But what if we could amplify the effects?

A DNS amplification attack is characterized by small queries that result in very much
larger responses. A typical query is about 30 bytes and its response is around 45 bytes on
average. The following are three techniques that are used to turn this relatively equal ratio
of query to response size by a factor of up to 50 times:

• DNS ANY DNS has a (deprecated in 2019, but still used) diagnostic feature that allows a client to request all the information a server has on a given domain
name. By sending a query of type ANY, an attacker can cause the server to send all the records in that domain up to the maximum size of a DNS message, which is 512 bytes. Having a 30-byte query produce a 512-byte response is a 17× amplification.
• EDNS(0) There are several situations in which the 512-byte limit on DNS messages over UDP becomes problematic. In particular, it is not possible to implement DNS Security Extensions (DNSSEC) with this constraint. Therefore, the Internet Engineering Task Force (IETF) developed EDNS(0), the Extension Mechanisms for DNS, which allows for up to 4096-byte responses. Properly used by an attacker, this new maximum size represents a 136× amplification given a 30-byte query.
• DNSSEC One of the most practical ways to exploit the maximum size defined in EDNS(0) is, ironically, using DNSSEC. Going back to Figure 13-6, when the local DNS server requests the A record from the authoritative server for that
domain (the bottom left one), it also requests the DNSSEC associated with the zone. This is done to ensure the identity of the authoritative server (and hence the response) but results in a significantly larger response (because it includes a
digital signature). So, all an attacker needs to do is find open DNS servers that
have DNSSEC enabled and direct the bots at them.

Domain Name System Security Extensions
DNSSEC is a set of standards IETF developed to protect DNS from a variety of attacks.
Specifically, DNSSEC is focused on ensuring the integrity of DNS records, not their
confidentiality or availability. In plain-old DNS, a client makes a recursive query that,
eventually, is responded to by some server that claims to be authoritative and provides
an IP address. As we discussed in Chapter 11, however, this led to impersonation attacks

where unwitting clients were pointed to malicious hosts. In response to this threat, the
IETF came up with DNSSEC.
DNSSEC works by grouping records in a DNS zone according to their name and type (e.g., A, NS, MAIL) into Resource Record Sets (RRSets) that are then digitally

signed, with the
resulting signature going into a resource record signature (RRSig) record. The corresponding
public key is published in a DNSKey record. So, when we want to resolve a fully qualified
domain name (FQDN) using DNSSEC, we first retrieve the RRSet containing the name,
then we request the RRSig for that set, and finally we verify that the record has not been
tampered with. While this approach prevents impersonation and cache poisoning attacks, it
has, as we just saw, also opened the door to crippling amplification attacks.

DNS over HTTPS

DNS Filtering
Our final topic on securing DNS is perhaps the most obvious. Instead of allowing any
DNS request to go out of our organizational networks, what if we first filtered them to
block known malicious (or otherwise disallowed) domains from being resolved in the
first place? A DNS filter performs a similar role as a web proxy that blocks content that
is inappropriate, except that it works on DNS instead of HTTP traffic. There are many
commercial solutions that provide this functionality, but keep in mind they should be
deployed as part of a broader, defense-in-depth approach to securing DNS.

Electronic Mail
Let’s now shift our attention to the third major service (along with web and DNS
services) that is required for virtually all major organizations: e-mail. Though it has lost
some ground to other business communication platforms such as Slack, Microsoft Teams,

PART IV

While DNSSEC ensures the integrity of DNS data, it does nothing to protect the
confidentiality or privacy of queries. Sure, you can be confident that the IP address you got
back was the right one, but what if anyone on the network can now see that you went to
a domain called embarrassingmedicalcondition.com? We know from our discussion of
TLS 1.3 earlier in this chapter that this URL will not go out in plaintext over HTTPS
(which, by the way, it will in TLS 1.2 and earlier), but it will still be visible before the
TLS handshake when the DNS query goes out. This is particularly problematic when
we are connected to public networks such as the Wi-Fi network at the local coffee shop.
DNS over HTTPS (DoH) is a (yet to be ratified) approach to protecting the privacy

and confidentiality of DNS queries by sending them over HTTPS/TCP/IP instead of unsecured UDP/IP. As of this writing, DoH is available on most platforms, though it is an optional feature that has to be configured. Keep in mind, however, that DoH provides confidentiality but (unlike DNSSEC) not integrity protections. Also, DoH was conceived as a privacy mechanism when using public networks. If you think back to the DNS-enabled attacks we discussed earlier in this chapter (especially DGA and DNS tunneling), DoH would actually make these much harder to detect unless you have a TLS decryption proxy in place. This is one of the reasons why the U.S. NSA recommended in 2021 that DoH not use external resolvers in enterprise networks.

Mail server

Mail server

E-mail client

E-mail client

SMTP

SMTP
Mail server
Receiver

Sender

Mail server

Figure 13-9

Mail server

SMTP works as a transfer agent for e-mail messages.

and Google Hangouts, e-mail remains a critical service in virtually all organizations. An e-mail message, however, is of no use unless it can actually be sent somewhere. This is where Simple Mail Transfer Protocol (SMTP) comes in. In e-mail clients, SMTP works as a message transfer agent, as shown in Figure 13-9, and moves the message from the

user’s computer to the mail server when the user clicks the Send button. SMTP also functions as a message transfer protocol between e-mail servers. Lastly, SMTP is a messageexchange addressing standard, and most people are used to seeing its familiar addressing scheme: something@somewhere.com.

Many times, a message needs to travel throughout the Internet and through different mail servers before it arrives at its destination mail server. SMTP is the protocol that carries this message, and it works on top of TCP because it is a reliable protocol and provides sequencing and acknowledgments to ensure the e-mail message arrived successfully at its destination.

The user’s e-mail client must be SMTP-compliant to be properly configured to use this protocol. The e-mail client provides an interface to the user so the user can create and modify messages as needed, and then the client passes the message off to the SMTP application layer protocol. So, to use the analogy of sending a letter via the post office, the e-mail client is the typewriter that a person uses to write the message, SMTP is the mail courier who picks up the mail and delivers it to the post office, and the post office is the mail server. The mail server has the responsibility of understanding where the message is heading and properly routing the message to that destination.

It is worth noting that basic SMTP doesn’t include any security controls. This is why the IETF published Extended SMTP (ESMTP), which, among other features, allows servers to negotiate a TLS session in which to exchange the messages. This implementation, referred to as SMTP Secure (SMTPS), can provide authentication, confidentiality, and integrity protections for mail transfers.

The mail server is often referred to as an SMTP server. The most common SMTP server software in the world is Exim, which is an open-source mail transfer agent (MTA).

SMTP works closely with two mail server protocols, POP and IMAP, which are explained in the following sections.

POP

Post Office Protocol (POP) is an Internet mail server protocol that supports incoming and outgoing messages. The current version is 3, so you’ll also see it referred to as POP3. A

mail server that uses POP, apart from storing and forwarding e-mail messages, works
with SMTP to move messages between mail servers. By default, POP servers listen on
TCP port 110.
A smaller organization may have only one POP server that holds all employee mailboxes, whereas larger organizations could have several POP servers, one for each
department within the organization. There are also Internet POP servers that enable
people all over the world to exchange messages. This system is useful because the messages
are held on the mail server until users are ready to download their messages, instead of
trying to push messages right to a person's computer, which may be down or offline.
The e-mail server can implement different authentication schemes to ensure an individual is authorized to access a particular mailbox, but this is usually handled through
usernames and passwords. Connections to these clients can be encrypted using TLS by
using the secure version of POP, known as POP3S, which typically listens on port 995.

IMAP
Internet Message Access Protocol (IMAP) is also an Internet protocol that enables users to
access mail on a mail server (the default TCP port is 143). IMAP provides all the functionalities of POP, but has more capabilities. If a user is using POP, when he accesses
his mail server to see if he has received any new messages, all messages are automatically

PART IV

E-mail spoofing is a technique used by malicious users to forge an e-mail to make
it appear to be from a legitimate source. Usually, such e-mails appear to be from
known and trusted e-mail addresses when they are actually generated from a malicious source. This technique is widely used by attackers these days for spamming
and phishing purposes. An attacker tries to acquire the target's sensitive information, such as username and password or bank account credentials. Sometimes, the
e-mail messages contain a link of a known website when it is actually a fake website
used to trick the user into revealing his information.
E-mail spoofing is done by modifying the fields of e-mail headers, such as the From, Return-Path, and Reply-To fields, so the e-mail appears to be from a trusted
source. This results in an e-mail looking as though it is from a known e-mail address.
Mostly the From field is spoofed, but some scams have modified the Reply-To

field
to the attacker’s e-mail address. E-mail spoofing is caused by the lack of security
features in SMTP. When SMTP technologies were developed, the concept of e-mail spoofing didn’t exist, so countermeasures for this type of threat were not embedded
into the protocol. A user could use an SMTP server to send e-mail to anyone from any e-mail address. We’ll circle back to these threats when we describe e-mail
security later in this section.

downloaded to his computer. Once the messages are downloaded from the POP server,
they are usually deleted from that server, depending upon the configuration. POP can
cause frustration for mobile users because the messages are automatically pushed down
to their computer or device and they may not have the necessary space to hold all the
messages. This is especially true for mobile devices that can be used to access e-mail servers. This is also inconvenient for people checking their mail on other people’s computers.
If Christina checks her e-mail on Jessica’s computer, all of Christina’s new mail could be
downloaded to Jessica’s computer.
If a user uses IMAP instead of POP, she can download all the messages or leave them
on the mail server within her remote message folder, referred to as a mailbox. The user
can also manipulate the messages within this mailbox on the mail server as if the messages
resided on her local computer. She can create or delete messages, search for specific
messages, and set and clear flags. This gives the user much more freedom and keeps
the messages in a central repository until the user specifically chooses to download all
messages from the mail server.
IMAP is a store-and-forward mail server protocol that is considered POP’s successor.
IMAP also gives administrators more capabilities when it comes to administering and
maintaining the users’ messages. Just like SMTP and POP, IMAP can run over TLS, in
which case the server listens for connections on TCP port 993.

E-mail Authorization
POP has the capability to integrate Simple Authentication and Security Layer (SASL),
a protocol-independent framework for performing authentication. This means that any protocol that knows how to interact with SASL can use its various

authentication
mechanisms without having to actually embed the authentication mechanisms within its code.
To use SASL, a protocol includes a command for identifying and authenticating a user to an authentication server and for optionally negotiating protection of subsequent
protocol interactions. If its use is negotiated, a security layer is inserted between the
protocol and the connection. The data security layer can provide data integrity, data
confidentiality, and other services. SASL's design is intended to allow new protocols to
reuse existing mechanisms without requiring redesign of the mechanisms and allows
existing protocols to make use of new mechanisms without redesign of protocols. The use of SASL is not unique just to POP; other protocols, such as IMAP, Internet
Relay Chat (IRC), Lightweight Directory Access Protocol (LDAP), and SMTP, can also
use SASL and its functionality.

Sender Policy Framework
A common way to deal with the problem of forged e-mail messages is by using Sender
Policy Framework (SPF), which is an e-mail validation system designed to prevent e-mail
spam by detecting e-mail spoofing by verifying the sender's IP address. SPF allows administrators to specify which hosts are allowed to send e-mail from a given domain by creating
a specific SPF record in DNS. Mail exchanges use DNS to check that mail from a given
domain is being sent by a host sanctioned by that domain's administrators.

DomainKeys Identified Mail
We can also leverage public key infrastructure (PKI) to validate the origin and integrity
of each message. The DomainKeys Identified Mail (DKIM) standard, codified in RFC 6376, allows e-mail servers to digitally sign messages to provide a measure of confidence
for the receiving server that the message is from the domain it claims to be from. These
digital signatures are normally invisible to the user and are just used by the servers sending and receiving the messages. When a DKIM-signed message is received, the server
requests the sending domain's certificate through DNS and verifies the signature. As long
as the private key is not compromised, the receiving server is assured that the message
came from the domain it claims and that it has not been altered in transit.

Domain-Based Message Authentication

SPF and DKIM were brought together to define the Domain-based Message Authentication, Reporting and Conformance (DMARC) system. DMARC, which today is estimated to protect 80 percent of mailboxes worldwide, defines how domains communicate to the rest of the world whether they are using SPF or DKIM (or both). It also codifies the mechanisms by which receiving servers provide feedback to the senders on the results of their validation of individual messages. Despite significant advances in securing e-mail, phishing e-mail remains one of the most common and effective attack vectors.

Multipurpose Internet Mail Extensions (MIME) is a technical specification indicating how multimedia data and e-mail binary attachments are to be transferred. The Internet has mail standards that dictate how mail is to be formatted, encapsulated, transmitted, and opened. If a message or document contains a binary attachment, MIME dictates how that portion of the message should be handled.

When an attachment contains an audio clip, graphic, or some other type of multimedia component, the e-mail client sends the file with a header that describes the file type. For example, the header might indicate that the MIME type is Image and that the subtype is jpeg. Although this information is in the header, many times, systems also use the file's extension to identify the MIME type. So, in the preceding example, the file's name might be stuff.jpeg. The user's system sees the extension .jpeg, or sees the data in the header field, and looks in its association list to see what program it needs to initialize to open this particular file. If the system has JPEG files associated with the Explorer application, then Explorer opens and presents the image to the user.

Sometimes systems either do not have an association for a specific file type or do not have the helper program necessary to review and use the contents of the file. When a file has an unassociated icon assigned to it, it might require the user to choose the Open With command and choose an application in the list to associate this file with that program.

So when the user double-clicks that file, the associated program initializes and presents the file. If the system does not have the necessary program, the website might offer the necessary helper program, like Acrobat or an audio program that plays WAV files.

MIME is a specification that dictates how certain file types should be transmitted and handled. This specification has several types and subtypes, enables different computers

to exchange data in varying formats, and provides a standardized way of presenting the data. So if Sean views a funny picture that is in GIF format, he can be sure that when he sends it to Debbie, it will look exactly the same.

Secure MIME (S/MIME) is a standard for encrypting and digitally signing e-mail and for providing secure data transmissions. S/MIME extends the MIME standard by providing support for the encryption of e-mail and attachments. The encryption and hashing algorithms can be specified by the user of the mail application, instead of having it dictated to them. S/MIME follows the Public Key Cryptography Standards (PKCS). It provides confidentiality through encryption algorithms, integrity through hashing algorithms, authentication through the use of X.509 public key certificates, and nonrepudiation through cryptographically signed message digests.

Multilayer Protocols

Not all protocols fit neatly within the layers of the OSI model. This is particularly evident among devices and networks that were never intended to interoperate with the Internet.

For this same reason, they tend to lack robust security features aimed at protecting the availability, integrity, and confidentiality of the data they communicate. The problem is that as the Internet of old becomes the Internet of Things (IoT), these previously isolated devices and networks find themselves increasingly connected to a host of threats they were never meant to face.

As security professionals, we need to be aware of these nontraditional protocols and their implications for the security of the networks to which they are connected.

In particular, we should be vigilant when it comes to identifying nonobvious cyberphysical systems. In December 2015, attackers were able to cut power to over 80,000 homes in Ukraine apparently by compromising the utilities' supervisory control and data acquisition (SCADA) systems in what is considered the first known blackout caused by a cyberattack. A few years later, in 2017, attackers were able to exploit a previously unknown vulnerability and reprogram a Schneider Electric safety instrumented system (SIS) at an undisclosed target, causing the facility to shut down. At the heart

of most
SCADA systems used by power and water utilities is a multilayer protocol known as
DNP3.

Distributed Network Protocol 3
The Distributed Network Protocol 3 (DNP3) is a communications protocol designed for
use in SCADA systems, particularly those within the power sector. It is not a
generalpurpose protocol like IP, nor does it incorporate routing functionality. SCADA systems
typically have a very flat hierarchical architecture in which sensors and actuators are
connected to remote terminal units (RTUs). The RTUs aggregate data from one or more
of these devices and relay it to the SCADA master, which includes a human–machine
interface (HMI) component. Control instructions and configuration changes are sent
from the SCADA master to the RTUs and then on to the sensors and actuators.
At the time DNP3 was designed, there wasn't a need to route traffic among the
components (most of which were connected with point-to-point circuits), so networking
was not needed or supported in DNP3. Instead of using the OSI seven-layer model,

its developers opted for a simpler three-layer model called the Enhanced Performance
Architecture (EPA) that roughly corresponds to layers 2, 4, and 7 of the OSI model. There
was no encryption or authentication, since the developers did not think network attacks
were feasible on a system consisting of devices connected to each other and to nothing else.
Over time, SCADA systems were connected to other networks and then to the Internet
for a variety of very valid business reasons. Unfortunately, security wasn't considered
until much later. Encryption and authentication features were added as an afterthought,
though not all implementations have been thus updated. Network segmentation is not
always present either, even in some critical installations. Perhaps most concerning is the
shortage of effective IPSs and IDSs that understand the interconnections between DNP3
and IP networks and can identify DNP3-based attacks.

Controller Area Network Bus

Modbus
Like CAN bus, the Modbus system was developed to prioritize functionality over

security. A communications system created in the late 1970s by Modicon, now Schneider
Electric, Modbus enables communications among SCADA devices quickly and easily. Since its inception, Modbus has quickly become the de facto standard for communications between programmable logic controllers (PLCs). But as security was not built in,
Modbus offers little protection against attacks. An attacker residing on the network can
simply collect traffic using a tool like Wireshark, find a target device, and issue commands directly to the device.

Converged Protocols
Converged protocols are those that started off independent and distinct from one another
but over time converged to become one. How is this possible? Think about the phone
and data networks. Once upon a time, these were two different entities and each had its

PART IV

Another multilayer protocol that had almost no security features until very recently is
the one that runs most automobiles worldwide. The Controller Area Network (CAN) bus
is a protocol designed to allow microcontrollers and other embedded devices to communicate with each other on a shared bus. Over time, these devices have diversified so that
today they can control almost every aspect of a vehicle’s functions, including steering,
braking, and throttling. CAN bus was never meant to communicate with anything outside the vehicle except for a mechanic’s maintenance computer, so there never appeared
to be a need for security features.
As automobiles started getting connected via Wi-Fi and cellular data networks, their
designers didn’t fully consider the new attack vectors this would introduce to an otherwise
undefended system. That is, until Charlie Miller and Chris Valasek famously hacked a
Jeep in 2015 by connecting to it over a cellular data network and bridging the head unit
(which controls the sound system and GPS) to the CAN bus (which controls all the vehicle sensors and actuators) and causing it to run off a road. As automobiles become
more autonomous, security of the CAN bus becomes increasingly important.

own protocols and transmission media. For a while, in the 1990s, data networks sometimes rode over voice networks using data modems. This was less than ideal, which is
why we flipped it around and started using data networks as the carrier for

voice communications. Over time, the voice protocols converged onto the data protocols, which
paved the way for Voice over IP (VoIP).
IP convergence, which addresses a specific type of converged protocols, is the transition
of services from disparate transport media and protocols to IP. It is not hard to see that
IP has emerged as the dominant standard for networking, so it makes sense that any new
protocols would leverage this existing infrastructure rather than create a separate one.
Technically, the term converged implies that the two protocols became one. Oftentimes, however, the term is used to describe cases in which one protocol was originally
independent of another but over time started being encapsulated (or tunneled) within
that other one.

Encapsulation

We already saw (in Chapter 9) how encapsulation enables the transmission of data down
the seven layers of the OSI reference model. We came across encapsulation again earlier
in this chapter when we discussed techniques to tunnel (or encapsulate) one protocol's
traffic inside some other protocol. The next two sections describe two more examples. It
should be obvious that encapsulation can be helpful in architecting our networks, but it
can also have significant security implications.
When we covered DNS tunneling, we saw another, less helpful application of encapsulation. Threat actors develop their own protocols for controlling compromised
hosts and they can encapsulate those protocols within legitimate systems. It is important,
therefore, to not assume that just because we have a network link that should be transporting data of a certain protocol, it won't have something else embedded in it.
Whether encapsulation is malicious or benign, the point is that we need to be aware
of what traffic should be where and have the means to inspect it to ensure we are not
surprised.

Fiber Channel over Ethernet

Fibre Channel (FC) (also called Fiber Channel in the United States) was developed by the
American National Standards Institute (ANSI) in 1988 as a way to connect supercomputers using optical fibers. FC is now used to connect servers to data storage devices in
data centers and other high-performance environments. One of its best features is that
it can support speeds of up to 128 Gbps over distances of up to 500 meters. (Distances

of up to 50 km are possible at lower data rates.) While the speed and other features of
FC are pretty awesome for data centers and storage area network (SAN) applications, the
need to maintain both Ethernet and fiber-optic cabling adds costs and complexity to its
use in enterprise environments.
Fibre Channel over Ethernet (FCoE) is a protocol encapsulation that allows FC frames
to ride over Ethernet networks. Its use allows data centers to be almost exclusively
wired using Ethernet cabling. It is important to note, however, that FCoE rides on

top of Ethernet and is, therefore, a non-routable protocol. It is only intended for LAN
environments where devices are in close proximity to each other and efficiency is essential.

Internet Small Computer Systems Interface
A much different approach to encapsulation is exemplified by the Internet Small Computer Systems Interface (iSCSI), which encapsulates SCSI data in TCP segments. SCSI is
a set of technologies that allows peripherals to be connected to computers. The problem
with the original SCSI is that it has limited range, which means that connecting a remote
peripheral (e.g., camera or storage device) is not normally possible. The solution was to
let SCSI ride on TCP segments so that a peripheral device could be literally anywhere in
the world and still appear as local to a computer.

Network Segmentation

PART IV

Once upon a time, networks were flat (i.e., almost everyone within an organization was
in the same layer 2 broadcast domain) so that everyone could easily communicate with
everyone else inside the “trusted” perimeter. Network defenses were mostly (sometimes
solely) outward-facing. This led to the networks that were “crunchy on the outside but
soft and chewy on the inside.” Believe it or not, this was the design mantra for many
organizations for many years. Eventually, they realized that this design was a really bad
idea. For starters, they recognized that at least some attackers will get through their

perimeter defenses. Also, they learned that insider threats could be just as dangerous as external ones, and these insiders would have no problem moving through the soft and chewy interior network. Furthermore, they realized that most networks no longer have a neat concept of “inside” and “outside.” Instead, organizations increasingly rely on external systems such as those provided by cloud service providers.

Network segmentation is the practice of dividing networks into smaller subnetworks. An example is to divide the network by department, so that the finance department and marketing department are each in their own LAN. If they need to communicate directly, they have to go through a gateway (e.g., a router or firewall) that allows network administrators to block or detect suspicious traffic. This is a classic implementation of the zero trust security design principle.

The decision to segment a network begs a couple of questions. How many subnetworks should we have? Are more subnets better? There really is no one-size-fits-all answer, but generally, the smaller the subnetworks (and the more you have), the better. In fact, many organizations are implementing micro-segmentation, which is the practice of isolating individual assets (e.g., data servers) in their own protected network environment. Think of it as a subnet where the only devices are the protected asset and a security gateway.

So, how do we go about segmenting networks? We can do it either physically (using devices like switches and routers) or logically (using virtualization software). We’ll cover devices in detail in the next chapter, so let’s turn our attention to the most important technologies that enable segmentation and micro-segmentation.

VLANs

One of the most commonly used technologies used to segment LANs is the virtual local area network (VLAN). A LAN can be defined as a set of devices on the same layer 2 (data link layer) broadcast domain. This typically means hosts that are physically connected to the same layer 2 switches. A VLAN is a set of devices that behave as though they were all directly connected to the same switch, when in fact they aren’t. This allows you to, for instance, ensure that all members of the finance team are on the same

(virtual) LAN,
even though they are scattered across multiple countries. The ability to segment networks
of users in this manner is critical for both functional and security reasons.
Virtually all modern enterprise-grade switches have the capability to use VLANs.
VLANs enable administrators to separate and group computers logically based on resource
requirements, security, or business needs instead of the standard physical location of the
systems. When repeaters, bridges, and routers are used, systems and resources are grouped
in a manner dictated by their physical location. Figure 13-10 shows how computers that
are physically located next to each other can be grouped logically into different VLANs.
Administrators can form these groups based on the users' and organization's needs instead
of the physical location of systems and resources.

Figure 13-10
VLANs enable
administrators to
manage logical
networks.

Floor 2

Floor 1

Floor 3

Floor 4

An administrator may want to segment the computers of all users in the marketing
department in the same VLAN network, for example, so all users receive the same
broadcast messages and can access the same types of resources. This arrangement could
get tricky if a few of the users are located in another building or on another floor, but
VLANs provide the administrator with this type of flexibility. VLANs also enable an
administrator to apply particular security policies to respective zones or segments.
This way, if tighter security is required for the payroll department, for example, the
administrator can develop a policy, add all payroll systems to a specific VLAN, and apply
the security policy only to the payroll VLAN.
A VLAN exists on top of the physical network, as shown in Figure 13-11. Each
Ethernet frame is prepended with a VLAN identifier (VID), which is a 12-bit field. This
means that we can define up to 4,095 VLANs in the same network. (The first and

last
VID values are reserved.) If workstation P1 wants to communicate with
workstation D1,
the message has to be routed—even though the workstations are physically next to
each
other—because they are on different logical networks.
NOTE The IEEE standard that defines how VLANs are to be constructed and
how tagging should take place to allow for interoperability is IEEE 802.1Q.

D2 Development
VLAN

D1

P1

D1

P2

P3

D2

D4

D5

P4

P4

P5

D3

D4

P5

D5

Physical netwrok

P1

Payroll
VLAN

P3

P2

D3

Virtual netwrok

PART IV

Figure 13-11
VLANs exist on
a higher level
than the physical
network and are
not bound to it.

Switch

Switch
Enterprise switch

While VLANs are used to segment traffic, attackers can still gain access to traffic that is supposed to be “walled off ” in another VLAN segment. VLAN hopping attacks allow attackers to gain access to traffic in various VLAN segments. An attacker can have a system act as though it is a switch. The system understands the tagging values being used in the network and the trunking protocols and can insert itself between other VLAN devices and gain access to the traffic going back and forth. This is called a switch spoofing attack. An attacker can also insert VLAN tags to manipulate the control of traffic at the data link layer in what is known as a double tagging attack. Proper configuration of all switches mitigates VLAN hopping attacks.

Virtual eXtensible Local Area Network
VLANs, however, have some significant limitations. For starters, remember that you’re limited to around 4,000 VLANs because the VID is 12 bits. While this sounds like a lot, it really isn’t if you happen to be a cloud-based service provider supporting hundreds of customers. Another challenge is that VLANs are layer 2 constructs separated by layer 3 routers. This means that all the hosts on a given VLAN must be on the same port of the same router. In other words, if the hosts are in different countries, it becomes really hard to join them to the same VLAN.
The Virtual eXtensible Local Area Network (VxLAN) is a network virtualization technology that encapsulates layer 2 frames onto UDP (layer 4) datagrams for

distribution
anywhere in the world. Whereas VLANs have VIDs, VxLANs have a virtual network identifier (VNI) that is 24 bits long, which gives us over 16 million segments. VxLANs
are mostly used in cloud environments where hosts and networks are virtualized. VxLANs are overlay networks on top of UDP/IP underlay networks. Each network switch or router that is part of a VxLAN has a virtual tunnel end point (VTEP) that
provides the interface between the underlay and overlay networks. When a VTEP receives a frame, it establishes a virtual tunnel on the overlay network connecting it to
the destination VTEP just long enough to deliver the frame. The VTEP encapsulates
this overlay frame in UDP datagrams that are then passed to the underlay network for
delivery.

Software-Defined Networks
Software-defined networking (SDN) is an approach to networking that relies on distributed software to provide unprecedented agility and efficiency. Using SDN, it becomes
much easier to dynamically route traffic to and from newly provisioned services and platforms. This means a new server can be quickly provisioned using a cloud service provider
in response to a spike in service requests and the underlying network can just as quickly
adapt to the new traffic patterns. It also means that a service or platform can be quickly
moved from one location to another and the SDN will just as quickly update traffic-flow
rules in response to this change. Unsurprisingly, the three biggest drivers to the adoption
of SDN are the growth in cloud computing, big data, and mobile computing.
How does SDN differ from traditional networking? Whereas traditional networking relies on network devices that coordinate with one another in a mostly decentralized

manner, SDN centralizes the configuration and control of devices. In a decentralized
environment, it takes time for routers to converge onto (or agree on) good routes. These
devices must normally be manually configured whenever any changes take place, which
is also a time-consuming task. In SDN, on the other hand, all changes are pushed out to
the devices either reactively (i.e., in response to requests from the devices) or proactively
(i.e., because the admins know a change is being made, such as the addition of 100
servers). Because it is centrally controlled, the SDN approach allows traffic to be routed

much more efficiently and securely. Perhaps the most important element of SDN is the
abstraction of control and forwarding planes.

Control and Forwarding Planes

NOTE Because traditional routing decisions are made by the controller in an SDN architecture, the network devices behave (and are referred to) as switches.

In a traditional network architecture, each networking device has its own control
plane and its own forwarding plane, both of which run on some sort of proprietary
operating system (e.g., Cisco IOS). The normal way of reconfiguring these traditional
devices is via a terminal connection of some sort. This means that an administrator must
remotely log into each device in order to change its configuration. Let’s suppose that
we want to support a distinct QoS for a new user. In order to do this, we’d modify the
configuration in each networking device that would be involved in providing services to
this user. Even assuming that we are able to do this without making any mistakes, we
still face the onerous task of manually changing these parameters whenever the terms of
the contract change, or when equipment is replaced or upgraded, or when the network
architecture changes. There are exceptions to these challenges, of course, but the point is
that making frequent, granular configuration changes is tough.

PART IV

The control plane is where the internetwork routing decisions are being made. Think of
this as the part of your router that runs the routing protocol, such as Open Shortest Path
First (OSPF). (The analogy is not perfect, but it is useful for now.) The control plane
is responsible for discovering the topology of neighboring networks and maintaining a
table of routes for outbound packets. Since most networks are pretty dynamic places in
which congestion along different routes is always changing, the control plane is a pretty
dynamic place as well. New routes are routinely being discovered, just as old routes are
dropped or at least flagged as slow or expensive. As you can see, the control plane is
mostly interested in effects that are more than one hop away.
The forwarding plane, by contrast, is where traffic forwarding decisions are

made.
Think of this as that part of your router that decides (very quickly) that a packet received
on network interface eth0 needs to be forwarded to network interface eth3. How does the
forwarding plane decide this? By using the products developed by the control plane. The
control plane is the strategic, methodical planner of traffic routing, while the forwarding
plane is the tactical, fast executioner of those plans. Unsurprisingly, the forwarding plane is
typically implemented in hardware such as an application-specific integrated chip (ASIC).

What About Automation?
One of the challenges of network administration is that most network devices (apart
from those that support SDN) do not have comprehensive mechanisms for
programmatically and remotely changing the configuration of the device. This is why
administrators have to manually log into each device and update the configuration.
Reading information is easier because these devices typically support SNMP, but
writing meaningful changes to the devices almost always requires manual
interaction or some third-party tool that comes with its own set of constraints.
Further complicating the issue of making dynamic changes, vendors typically
use their own proprietary operating system, which makes it harder to write a
script that makes the same changes to all devices in heterogeneous environments
that implement products from multiple vendors. This is the reason why many
organizations implement homogeneous network architectures in which all the
devices are manufactured by the same vendor. A big downside of this homogeneity
is that it leads to vendor lockdown because it is hard (and expensive) to change
vendors when that means you must change every single device on your network.
Furthermore, homogeneity is bad for security, because an exploit that leverages
a vulnerability in a network operating system will likely affect every device in a
homogeneous network.

In SDN, by contrast, the control plane is implemented in a central node that is
responsible for managing all the devices in the network. For redundancy and efficiency,
this node can actually be a federation of nodes that coordinate their activities with one
another. The network devices are then left to do what they do best: forward packets very
efficiently. So the forwarding plane lives in the network devices and the control plane
lives in a centralized SDN controller. This allows us to abstract the network devices
(heterogeneous or otherwise) from the applications that rely on them to communicate

in much the same way Windows abstracts the hardware details from the applications
running on a workstation.

Approaches to SDN
The concept of network abstraction is central to all implementations of SDN. The manner in which this abstraction is implemented, however, varies significantly among flavors
of SDN. There are at least three common approaches to SDN, each championed by a different community and delivered primarily through a specific technology:

• Open The SDN approach championed by the Open Networking Foundation (ONF) (https://opennetworking.org) is, by most accounts, the most common. It relies on open-source code and standards to develop the building blocks of an SDN solution. The controller communicates with the switches using OpenFlow, a standardized, open-source communications interface between controllers and network devices in an SDN architecture. OpenFlow allows the devices

implementing the forwarding plane to provide information (such as utilization data) to the controller, while allowing the controller to update the flow tables (akin to traditional routing tables) on the devices. Applications communicate with the controller using the RESTful or Java APIs.
• API Another approach to SDN, and one that is championed by Cisco, is built on the premise that OpenFlow is not sufficient to fully leverage the promise of SDN in the enterprise. In addition to OpenFlow, this approach leverages a rich API on proprietary switches that allows greater control over traffic in an
SDN. Among the perceived shortcomings that are corrected are the inability of OpenFlow to do deep packet inspection and manipulation and its reliance on a centralized control plane. This proprietary API approach to SDN is seen as enriching rather than replacing ONF’s SDN approach.
• Overlays Finally, one can imagine a virtualized network architecture as an overlay on a traditional one. In this approach, we virtualize all network nodes, including switches, routers, and servers, and treat them independently of the physical networks upon which this virtualized infrastructure exists. The SDN exists simply as a virtual overlay on top of a physical (underlay) network.

Software-defined wide area networking (SD-WAN) is the use of software (instead of hardware) to control the connectivity, management, and services between distant sites. Think
of it as SDN applied to WANs instead of LANs. Similarly to SDN, SD-WAN separates the control plane from the forwarding plane. This means that network links, whether
they are leased lines or 5G wireless, are better utilized. Also, since the control plane is
centralized, security policies can be consistently applied throughout.
Another advantage of SD-WANs is that they are application-aware, meaning they know the difference between supporting video conferencing (low latency, loss tolerance),
supporting file transfers (latency tolerance, loss intolerant), or supporting any other sort

of traffic. This means SD-WANs use the right path for the traffic and are able to switch
things around as links become congested or degraded.

Chapter Review
Securing our networks is a lot more effective if we first understand the underlying
technologies and then apply secure design principles to their selection and integration.
This chapter built on the foundations of the previous two chapters to show common
approaches to building and operating secure networking architectures. We focused our
attention on network encryption and service security techniques but also covered how
to deal with dispersed networks and those with cloud service components. A key aspect
of our discussion was the application of the secure design principles at multiple points.
We'll continue this theme in the next chapter as we talk about securing the components
of our networks.

PART IV

Software-Defined Wide Area Network

Quick Review
• Link encryption encrypts all the data along a specific communication path.
• End-to-end encryption (E2EE) occurs at the session layer (or higher) and does not
encrypt routing information, enabling attackers to learn more about a captured
packet and where it is headed.
• Transport Layer Security (TLS) is an E2EE protocol that provides confidentiality
and data integrity for network communications.
• Secure Sockets Layer (SSL) is the predecessor of TLS and is deprecated and
considered insecure.
• A virtual private network (VPN) is a secure, private connection through an
untrusted network.
• The Point-to-Point Tunneling Protocol (PPTP) is an obsolete and insecure means
of providing VPNs.
• The Layer 2 Tunneling Protocol (L2TP) tunnels PPP traffic over various network
types (IP, ATM, X.25, etc.) but does not encrypt the user traffic.
• Internet Protocol Security (IPSec) is a suite of protocols that provides
authentication,
integrity, and confidentiality protections to data at the network layer.
• TLS can be used to provide VPN connectivity at layer 5 in the OSI model.
• A web service is client/server system in which clients and servers communicate
using HTTP over a network such as the Internet.

• A service-oriented architecture (SOA) describes a system as a set of interconnected
but self-contained components that communicate with each other and with their clients through standardized protocols.
• Application programming interfaces (APIs) establish a “language” that enables a system component to make a request from another component and then interpret that second component’s response.
• The Hypertext Transfer Protocol (HTTP) is a TCP/IP-based communications protocol used for transferring data between a server and a client in a connectionless
and stateless manner.
• A uniform resource identifier (URI) uniquely identifies a resource on the Internet.
• HTTP Secure (HTTPS) is HTTP running over TLS.
• The Simple Object Access Protocol (SOAP) is a messaging protocol that uses XML over HTTP to enable clients to invoke processes on a remote host in a platform-agnostic way.
• SOAP security is enabled by a set of protocol extensions called the Web Services
Security (WS-Security or WSS) specification, which provides message confidentiality,
integrity, and authentication.

• Representational State Transfer (REST) is an architectural pattern used to develop web services without using SOAP.
• A domain generation algorithm (DGA) produces seemingly random domain names in a way that is predictable by anyone who knows the algorithm.
• DNS tunneling is the practice of encoding messages in one or a series of DNS queries or responses for exfiltrating or infiltrating data into an environment.
• DNS reflection attacks involve sending a query to a server while spoofing the source address to be that of the intended target.
• A DNS amplification attack is characterized by small queries that result in very
much larger responses.
• Domain Name System Security Extensions (DNSSEC) is a set of IETF standards that ensures the integrity of DNS records but not their confidentiality or availability.
• DNS over HTTPS (DoH) is a (yet to be ratified) approach to protecting the privacy and confidentiality of DNS queries by sending them over HTTPS/TCP /IP instead of unsecured UDP/IP.
• E-mail spoofing is a technique used by malicious users to forge an e-mail to make
it appear to be from a legitimate source.
• Simple Authentication and Security Layer (SASL) is a protocol-independent framework for performing authentication that is typically used in POP3 e-mail systems.
• The Sender Policy Framework (SPF) is an e-mail validation system designed to prevent e-mail spam by detecting e-mail spoofing by verifying the sender’s

IP address.
• The DomainKeys Identified Mail (DKIM) standard allows e-mail servers to digitally sign messages to provide a measure of confidence for the receiving server that the message is from the domain it claims to be from.
• Domain-based Message Authentication, Reporting and Conformance (DMARC) systems incorporate both SPF and DKIM to protect e-mail.
• Secure MIME (S/MIME) is a standard for encrypting and digitally signing e-mail and for providing secure data transmissions.
• The Distributed Network Protocol 3 (DNP3) is a multilayer communications protocol designed for use in SCADA systems, particularly those within the power sector.
• The Controller Area Network (CAN) bus is a multilayer protocol designed to allow microcontrollers and other embedded devices to communicate with each other on a shared bus.
• Converged protocols are those that started off independent and distinct from one
another but over time converged to become one.

• Fibre Channel over Ethernet (FCoE) is a protocol encapsulation that allows Fibre
Channel (FC) frames to ride over Ethernet networks.
• The Internet Small Computer Systems Interface (iSCSI) protocol encapsulates SCSI data in TCP segments so that computer peripherals could be located at any physical distance from the computer they support.
• Network segmentation is the practice of dividing networks into smaller subnetworks.
• A virtual LAN (VLAN) is a set of devices that behave as though they were all directly connected to the same switch, when in fact they aren’t.
• Virtual eXtensible LAN (VxLAN) is a network virtualization technology that encapsulates layer 2 frames onto UDP (layer 4) datagrams for distribution anywhere in the world.
• Software-defined networking (SDN) is an approach to networking that relies on distributed software to separate the control and forwarding planes of a network.
• Software-defined wide area networking (SD-WAN) is the use of software (instead of hardware) to control the connectivity, management, and services between distant
sites in a manner that is similar to SDN but applied to WANs.

Questions

Please remember that these questions are formatted and asked in a certain way for a
reason. Keep in mind that the CISSP exam is asking questions at a conceptual level.
Questions may not always have the perfect answer, and the candidate is advised against
always looking for the perfect answer. Instead, the candidate should look for the best
answer in the list.

1. Which of the following provides secure end-to-end encryption?
A. Transport Layer Security (TLS)
B. Secure Sockets Layer (SSL)

C. Layer 2 Tunneling Protocol (L2TP)
D. Domain Name System Security Extensions (DNSSEC)

2. Which of the following can take place if an attacker is able to insert tagging
values into network- and switch-based protocols with the goal of manipulating traffic at the data link layer?
A. Open relay manipulation
B. VLAN hopping attack
C. Hypervisor denial-of-service attack
D. DNS tunneling

3. Which of the following provides an incorrect definition of the specific component
or protocol that makes up IPSec?
A. Authentication Header protocol provides data integrity, data origin

authentication, and protection from replay attacks.
B. Encapsulating Security Payload protocol provides confidentiality, data origin authentication, and data integrity.
C. Internet Security Association and Key Management Protocol provides a framework for security association creation and key exchange.
D. Internet Key Exchange provides authenticated keying material for use with encryption algorithms.
4. Alice wants to send a message to Bob, who is several network hops away from her.
What is the best approach to protecting the confidentiality of the message?
A. PPTP
B. S/MIME
C. Link encryption
D. SSH
5. Which technology would best provide confidentiality to a RESTful web service?
A. Web Services Security (WS-Security)
B. Transport Layer Security (TLS)
C. HTTP Secure (HTTPS)
D. Simple Object Access Protocol (SOAP)
6. Which of the following protections are provided by Domain Name System Security
Extensions (DNSSEC)?
A. Confidentiality and integrity
B. Integrity and availability
C. Integrity and authentication
D. Confidentiality and authentication
7. Which approach provides the best protection against e-mail spoofing?
A. Internet Message Access Protocol (IMAP)
B. Domain-based Message Authentication, Reporting and Conformance (DMARC)
C. Sender Policy Framework (SPF)
D. DomainKeys Identified Mail (DKIM)

8. Which of the following is a multilayer protocol developed for use in supervisory
control and data acquisition (SCADA) systems?
A. Controller Area Network (CAN) bus
B. Simple Authentication and Security Layer (SASL)
C. Control Plane Protocol (CPP)
D. Distributed Network Protocol 3 (DNP3)

9. All of the following statements are true of converged protocols except which one?
A. Distributed Network Protocol 3 (DNP3) is a converged protocol.
B. Fibre Channel over Ethernet (FCoE) is a converged protocol.
C. IP convergence addresses a specific type of converged protocols.
D. The term includes certain protocols that are encapsulated within each other.

10. Suppose you work at a large cloud service provider that has thousands of customers
around the world. What technology would best support segmentation of your customers' environments?
A. Virtual local area network (VLAN)
B. Virtual eXtensible Local Area Network (VxLAN)
C. Software-defined wide area networking (SD-WAN)
D. Layer 2 Tunneling Protocol (L2TP)

Answers
1. A. TLS and SSL are the only two answers that provide end-to-end encryption, but SSL is insecure, so it's not a good answer.
2. B. VLAN hopping attacks allow attackers to gain access to traffic in various VLAN segments. An attacker can have a system act as though it is a switch. The system understands the tagging values being used in the network and the trunking protocols and can insert itself between other VLAN devices and gain access to the traffic going back and forth. Attackers can also insert tagging values
to manipulate the control of traffic at this data link layer.
3. D. Authentication Header protocol provides data integrity, data origin authentication, and protection from replay attacks. Encapsulating Security Payload protocol provides confidentiality, data origin authentication, and data integrity. Internet Security Association and Key Management Protocol provides a framework for security association creation and key exchange. Internet Key Exchange provides authenticated keying material for use with ISAKMP.
4. B. Secure Multipurpose Internet Mail Extensions (S/MIME) is a standard for encrypting and digitally signing e-mail and for providing secure data transmissions
using public key infrastructure (PKI).

5. C. Either TLS or HTTPS would be a correct answer, but since web services in general and RESTful ones in particular require HTTP, HTTPS is the best choice.

Keep in mind that you are likely to come across similar questions where multiple answers are correct but only one is best. SOAP is an alternative way to deliver web services and uses WS-Security for confidentiality.
6. C. Domain Name System Security Extensions (DNSSEC) is a set of IETF standards that ensures the integrity and authenticity of DNS records but not their
confidentiality or availability.
7. B. Domain-based Message Authentication, Reporting and Conformance (DMARC) systems incorporate both SPF and DKIM to protect e-mail. IMAP does not have any built-in protections against e-mail spoofing.
8. D. DNP3 is a multilayer communications protocol designed for use in SCADA systems, particularly those within the power sector.
9. A. DNP3 is a multilayer communications protocol that was designed for use in SCADA systems and has not converged with other protocols. All other statements are descriptive of converged protocols.
PART IV

10. B. Since there are thousands of customers to support, VxLAN is the best choice
because it can support over 16 million subnetworks. Traditional VLANs are capped at just over 4,000 subnetworks, which would not be able to provide more than a few segments to each customer.

This page intentionally left blank

14

CHAPTER

Network Components
This chapter presents the following:
- Transmission media
- Network devices
- Endpoint security
- Content distribution networks

The hacker didn’t succeed through sophistication. Rather he poked at obvious places, trying to enter through unlocked doors. Persistence, not wizardry, let him through.
—Clifford Stoll,
The Cuckoo’s Egg
In the previous chapter, we covered how to defend our networks. Let’s now talk about
securing the components of those networks. We need to pay attention to everything from
the cables, to the network devices, to the endpoints, because our adversaries will poke at
all of it, looking for ways to get in. We (defenders) have to get it right all the time; they
(attackers) only need to find that one chink in our armor to compromise our systems. In
this chapter, we focus on physical devices. In the next chapter, we’ll drill into the software
systems that run on them.

Transmission Media

We've already talked a fair bit about the protocols that allow us to move data from point A to point B, but we haven't really covered what actually carries this information. A transmission medium is a physical thing through which data is moved. If we are speaking with each other, our vocal chords create vibrations in the air that we expel from our lungs, in which case the air is the transmission medium. Broadly speaking, we use three different types of transmission media:

- Electrical wires Encode information as changes in the voltage level of an electric current. Typically, we use cables, which are two or more wires encased within a sheath.
- Optical fibers Transmit data that is encoded in the wavelength (color), phase, or polarization of the light. The light is generated by either an LED or a laser diode. As with electrical wires, we usually bundle multiple fibers into cables for longer distances.

643

644

- Free space The medium we use for wireless communications, covered in Chapter 12. Any electromagnetic signal can travel through free space even outside our atmosphere. We tend to use mostly radio signals in free space, but every now and then you may encounter a system that uses light, such as infrared laser beams.

Types of Transmission

Physical data transmission can happen in different ways (analog or digital); can use different synchronization schemes (synchronous or asynchronous); can use either one sole channel over a transmission medium (baseband) or several different channels over a transmission medium (broadband); and can take place as electrical voltage, radio waves, or optical signals. These transmission types and their characteristics are described in the following sections.

Analog vs. Digital

A signal is just some way of moving information in a physical format from one point to another point. You can signal a message to another person through nodding your head, waving your hand, or giving a wink. Somehow you are transmitting data to that person through your signaling method. In the world of technology, we have specific

carrier signals that are in place to move data from one system to another system. The carrier signal is like a horse, which takes a rider (data) from one place to another place. Data can be transmitted through analog or digital signaling formats. If you are moving data through an analog transmission technology (e.g., radio), then the data is represented by the characteristics of the waves that are carrying it. For example, a radio station uses a transmitter to put its data (music) onto a radio wave that travels all the way to your antenna. The information is stripped off by the receiver in your radio and presented to you in its original format—a song. The data is encoded onto the carrier signal and is represented by various amplitude and frequency values, as shown in Figure 14-1.

Digital signal

Analog signal

Amplitude

Frequency

Figure 14-1 Analog signals are measured in amplitude and frequency, whereas digital signals represent binary digits as electrical pulses.

Asynchronous vs. Synchronous

Analog and digital transmission technologies deal with the characteristics of the physical carrier on which data is moved from one system to another. Asynchronous and synchronous transmission types are similar to the cadence rules we use for conversation synchronization. Asynchronous and synchronous network technologies provide synchronization rules to govern how systems communicate to each other. If you have ever spoken over a satellite phone, you have probably experienced problems with communication synchronization. Commonly, when two people are new to using satellite phones, they do not allow for the necessary delay that satellite communication requires, so they “speak over” one another. Once they figure out the delay in the connection, they resynchronize their timing so that only one person’s data (voice) is transmitting at one time, enabling each

Data being represented in wave values (analog) is different from data being

represented
in discrete voltage values (digital). As an analogy, compare an analog clock and a digital
clock. An analog clock has hands that continuously rotate on the face of the clock. To
figure out what time it is, you have to interpret the position of the hands and map their
positions to specific values. So you have to know that if the small hand is on the number
1 and the large hand is on the number 6, this actually means 1:30. The individual and
specific location of the hands corresponds to a value. A digital clock does not take this
much work. You just look at it and it gives you a time value in the format of hour:minutes.
There is no mapping work involved with a digital clock because it provides you with data
in clear-cut formats.
An analog clock can represent different values as the hands move forward—1:35 and
1 second, 1:35 and 2 seconds, 1:35 and 3 seconds. Each movement of the hands represents
a specific value just like the individual data points on a wave in an analog transmission. A
digital clock provides discrete values without having to map anything. The same is true
with digital transmissions: the values are almost always binary, meaning they are either a
1 or a 0—no need for mapping to find the actual value.
Computers have always worked in a binary manner (1 or 0). When our
telecommunication infrastructure was purely analog, each system that needed to
communicate over a telecommunication line had to have a modem (modulator/
demodulator), which would modulate the digital data into an analog signal. The sending
system's modem would modulate the data on to the signal, and the receiving system's
modem would demodulate the data off the signal.
Digital signals are more reliable than analog signals over a long distance and provide a
clear-cut and efficient signaling method because the voltage is either on (1) or not on (0),
compared to interpreting the waves of an analog signal. Extracting digital signals from
a noisy carrier is relatively easy. It is difficult to extract analog signals from background
noise because the amplitudes and frequencies of the waves slowly lose form. This is
because an analog signal could have an infinite number of values or states, whereas a
digital signal exists in discrete states. A digital signal is a square wave, which does not have
all of the possible values of the different amplitudes and frequencies of an analog signal.
Digital systems can implement compression mechanisms to increase data

throughput,
provide signal integrity through repeaters that “clean up” the transmissions, and multiplex
different types of data (voice, data, video) onto the same transmission channel.

person to properly understand the full conversation. Proper pauses frame your words in
a way to make them understandable.
Synchronization through communication also happens when we write messages to each other. Properly placed commas, periods, and semicolons provide breaks in text so
that the person reading the message can better understand the information. If you see
“stickwithmekidandyouwillweardiamonds” without the proper punctuation, it is more
difficult for you to understand. This is why we have grammar rules. If someone writes
a letter to you that starts from the bottom and right side of a piece of paper, and that
person does not inform you of this unconventional format, you will not be able to read
the message properly, at least initially.
Technological communication protocols also have their own grammar and synchronization rules when it comes to the transmission of data. If two systems are
communicating over a network protocol that employs asynchronous timing, they use start and stop bits. The sending system sends a “start” bit, then sends its character, and
then sends a “stop” bit. This happens for the whole message. The receiving system knows
when a character is starting and stopping; thus, it knows how to interpret each character
of the message. This is akin to our previous example of using punctuation marks in
written communications to convey pauses. If the systems are communicating over a network protocol that uses synchronous timing, then they don’t add start and stop bits.
The whole message is sent without artificial breaks, but with a common timing signal
that allows the receiver to know how to interpret the information without these bits. This
is similar to our satellite phone example in which we use a timing signal (i.e., we count
off seconds in our head) to ensure we don’t talk over the other person’s speech.
If two systems are going to communicate using a synchronous transmission technology,
they do not use start and stop bits, but the synchronization of the transfer of data takes
place through a timing sequence, which is initiated by a clock pulse.
It is the data link protocol that has the synchronization rules embedded into it. So

when a message goes down a system's network stack, if a data link protocol, such as
High-level Data Link Control (HDLC), is being used, then a clocking sequence is in
place. (The receiving system must also be using this protocol so that it can interpret the
data.) If the message is going down a network stack and a protocol such as Asynchronous
Transfer Mode (ATM) is at the data link layer, then the message is framed with start and
stop indicators.
Data link protocols that employ synchronous timing mechanisms are commonly used in environments that have systems that transfer large amounts of data in a predictable
manner (i.e., data center environment). Environments that contain systems that send
data in a nonpredictable manner (i.e., Internet connections) commonly have systems
with protocols that use asynchronous timing mechanisms.
So, synchronous communication protocols transfer data as a stream of bits instead
of framing it in start and stop bits. The synchronization can happen between two systems using a clocking mechanism, or a signal can be encoded into the data stream to
let the receiver synchronize with the sender of the message. This synchronization needs
to take place before the first message is sent. The sending system can transmit a digital
clock pulse to the receiving system, which translates into, "We will start here and work
in this type of synchronization scheme." Many modern bulk communication systems,

Asynchronous	Synchronous
Simpler, less costly implementation	More complex, costly implementation
No timing component	Timing component for data transmission synchronization
Parity bits used for error control	Robust error checking, commonly through cyclic redundancy checking (CRC)
Used for irregular transmission patterns	

Used for high-speed, high-volume transmissions

Each byte requires three bits of instruction
(start, stop, parity)

Minimal protocol overhead compared to
asynchronous communication

Table 14-1

Main Differences Between Asynchronous and Synchronous Transmissions

such as high-bandwidth satellite links, use Global Positioning System (GPS) clock
signals to synchronize their communications without the need to include a separate
channel for timing.
Table 14-1 provides an overview of the differences between asynchronous and synchronous transmissions.
As you read, analog transmission means that data is being moved as waves, and digital
transmission means that data is being moved as discrete electric pulses. Synchronous
transmission means that two devices control their conversations with a clocking mechanism, and asynchronous means that systems use start and stop bits for communication
synchronization. Now let’s look at how many individual communication sessions can
take place at one time.
A baseband technology uses the entire communication channel for its transmission,
whereas a broadband technology divides the communication channel into individual and independent subchannels so that different types of data can be transmitted simultaneously. Baseband permits only one signal to be transmitted at a time, whereas
broadband carries several signals over different subchannels. For example, a coaxial cable
TV (CATV) system is a broadband technology that delivers multiple television channels
over the same cable. This system can also provide home users with Internet access, but
this data is transmitted at a different frequency range than the TV channels.
As an analogy, baseband technology only provides a one-lane highway for data to get
from one point to another point. A broadband technology provides a data highway made
up of many different lanes, so that not only can more data be moved from one point to
another point, but different types of data can travel over the individual lanes.
Any transmission technology that “chops up” one communication channel into multiple channels is considered broadband. The communication channel is usually a specific range of frequencies, and the broadband technology provides delineation

between these frequencies and provides techniques on how to modulate the data onto
the individual subchannels. To continue with our analogy, we could have one large
highway that could fit eight individual lanes—but unless we have something that defines

PART IV

Broadband vs. Baseband

How Do These Technologies Work Together?

If you are new to networking, it can be hard to understand how the OSI model, analog and digital, synchronous and asynchronous, and baseband and broadband technologies interrelate and differentiate. You can think of the OSI model as a structure to build different languages. If you and Luigi are going to speak to each other in English, you have to follow the rules of this language to be able to understand each other. If you are going to speak French, you still have to follow the rules of that language (OSI model), but the individual letters that make up the words are in a different order. The OSI model is a generic structure that can be used to define many different "languages" for devices to be able to talk to each other. Once you and Luigi agree that you are going to communicate using English, you can speak your message to Luigi, and thus your words move over continuous airwaves (analog). Or you can choose to send your message to Luigi through Morse code, which uses individual discrete values (digital). You can send Luigi all of your words with no pauses or punctuation (synchronous) or insert pauses and punctuation (asynchronous). If you are the only one speaking to Luigi at a time, this would be analogous to baseband. If ten people are speaking to Luigi at one time, this would be broadband.

these lanes and have rules for how these lanes are used, this is a baseband connection. If we take the same highway and lay down painted white lines, post traffic signs, add on and off ramps, and establish rules that drivers have to follow, now we are talking about broadband.

A digital subscriber line (DSL) uses one single phone line and constructs a set of highfrequency channels for Internet data transmissions. A cable modem uses the available frequency spectrum that is provided by a cable TV carrier to move Internet traffic to and from a household. Mobile broadband devices implement individual channels over a

cellular connection, and Wi-Fi broadband technology moves data to and from an access point over a specified frequency set. The point is that there are different ways of cutting up one channel into subchannels for higher data transfer and that they provide the capability to move different types of traffic at the same time.

Cabling

The different types of transmission techniques we just covered eventually end up being used to send signals over either a cable or free space. We already covered wireless communications in Chapter 12, so let's talk about cabling now. Electrical signals travel as currents through cables and can be negatively affected by many factors within the environment, such as motors, fluorescent lighting, magnetic forces, and other electrical devices. These items can corrupt the data as it travels through the cable, which is why cable standards are used to indicate cable type, shielding, transmission rates, and maximum distance a particular type of cable can be used.

Figure 14-2
Coaxial cable

Twisted-Pair Cable

Twisted-pair cabling has insulated copper wires surrounded by an outer protective jacket. If the cable has an outer foil shielding, it is referred to as shielded twisted pair (STP), which adds protection from radio frequency interference (RFI) and EMI. Twisted-pair cabling, which does not have this extra outer shielding, is called unshielded twisted pair (UTP).

The twisted-pair cable contains copper wires that twist around each other, as shown in Figure 14-3. This twisting of the wires protects the integrity and strength of the signals they carry. Each wire forms a balanced circuit, because the voltage in each pair uses the

same amplitude, just with opposite phases. The tighter the twisting of the wires, the more

Figure 14-3
Twisted-pair
cabling uses
copper wires.

Outer
jacket

Insulated
wires

Copper wire
conductor

PART IV

Coaxial cable has a copper core that is surrounded by a shielding layer and grounding
wire, as shown in Figure 14-2. This is all encased within a protective outer jacket. Compared to twisted-pair cable, coaxial cable is more resistant to electromagnetic interference
(EMI), provides a higher bandwidth, and supports the use of longer cable lengths. So,
why is twisted-pair cable more popular? Twisted-pair cable is cheaper and easier to work
with, and the move to switched environments that provide hierarchical wiring schemes
has overcome the cable-length issue of twisted-pair cable.
Coaxial cabling is used as a transmission line for radio frequency signals. If you
have cable TV, you have coaxial cabling entering your house and the back of your TV.
The various TV channels are carried over different radio frequencies. Modems allow us
to use some of the “empty” TV frequencies for Internet connectivity.

UTP Category

Characteristics

Usage

Category 1

Voice-grade telephone cable for up
to 1 Mbps transmission rate

No longer in use for data or phones.

Category 2

Data transmission up to 4 Mbps

Historically used in mainframe and
minicomputer terminal connections,
but no longer in common use.

Category 3

10 Mbps for Ethernet

Used in older 10Base-T network
installations and legacy phone lines.

Category 4

16 Mbps

Normally used in Token Ring
networks.

Category 5

100 Mbps; two twisted pairs

Sometimes used in legacy 100BaseTX; deprecated in 2001 for data but
still used for telephone and video.

Category 5e

1 Gbps; four twisted pairs, providing
reduced crosstalk

Widely used in modern networks.

Category 6

1 Gbps, but can support 10 Gbps up
to 55 meters

Used in newer network installations
requiring high-speed transmission.
Standard for Gigabit Ethernet.

Table 14-2

UTP Cable Ratings

resistant the cable is to interference and attenuation. UTP has several
categories of cabling,
each of which has its own unique characteristics.
The twisting of the wires, the type of insulation used, the quality of the

conductive
material, and the shielding of the wire determine the rate at which data can be transmitted. The UTP ratings indicate which of these components were used when the cables were manufactured. Some types are more suitable and effective for specific uses
and environments. Table 14-2 lists the cable ratings.
Copper cable has been around for many years. It is inexpensive and easy to use. A
majority of the telephone systems today use copper cabling with the rating of voice
grade. Twisted-pair wiring is the preferred network cabling, but it also has its drawbacks.
Copper actually resists the flow of electrons, which causes a signal to degrade after it
has traveled a certain distance. This is why cable lengths are recommended for copper
cables; if these recommendations are not followed, a network could experience signal
loss and data corruption. Copper also radiates energy, which means information can be
monitored and captured by intruders. UTP is the least secure networking cable compared
to coaxial and fiber. If an organization requires higher speed, higher security, and cables
to have longer runs than what is allowed in copper cabling, fiber-optic cable may be a
better choice.

Fiber-Optic Cable
Twisted-pair cable and coaxial cable use copper wires as their data transmission media,
but fiber-optic cable uses a type of glass that carries light waves, onto which we modulate
the data being transmitted. The glass core is surrounded by a protective cladding, which
in turn is encased within an outer jacket.

Fiber Components
Fiber-optic cables are made up of a light source, an optical fiber cable, and a light
detector.
Light Sources

Convert electrical signal into light signal.

- Light-emitting diodes (LEDs)
- Diode lasers

Optical Fiber Cable

Data travels as light.

• Single mode Small glass core, used for high-speed data transmission over long distances. They are less susceptible to attenuation than multimode fibers.
• Multimode Large glass core, able to carry more data than single mode fibers, though they are best for shorter distances because of their higher attenuation levels.
Light Detector

Converts light signal back into electrical signal.

NOTE The price of fiber and the cost of installation have been steadily decreasing, while the demand for more bandwidth only increases. More organizations and service providers are installing fiber directly to the end user.

Cabling Problems
Cables are extremely important within networks, and when they experience problems,
the whole network could experience problems. This section addresses some of the more
common cabling issues many networks experience.

PART IV

Because it uses glass, fiber-optic cabling has higher transmission speeds that allow
signals to travel over longer distances. Fiber-optic cabling is not as affected by attenuation
and EMI when compared to cabling that uses copper. It does not radiate signals, as does
UTP cabling, and is difficult to eavesdrop on; therefore, fiber-optic cabling is much more
secure than UTP, STP, or coaxial.
Using fiber-optic cable sounds like the way to go, so you might wonder why you would
even bother with UTP, STP, or coaxial. Unfortunately, fiber-optic cable is expensive and
difficult to work with. It is usually used in backbone networks and environments that
require high data transfer rates. Most networks use UTP and connect to a backbone that
uses fiber.

Original digital signal

Transmission

Destination

Background
noise