Realtime
publishers

The Shortcut Guide™ To

Securing Your Exchange Server and Unified Communications Infrastructure Using SSL

sponsored by

GeoTrust®

Don Jones

Copyright Statement

Chapter 2: Messaging, Unified Communications, Security, and SAN Certificates

More so than in the past, modern messaging environments are full of security risks—and the number of risks seems to grow almost daily. Messaging used to be limited to email, and email access was typically limited to the office network or, at most, to a few dial-up users. Today, messaging consists of numerous forms of communication including email, real-time communication, and more, and we're extending the reach of that messaging environment well outside the corporate network. Today's users expect to receive email over the Internet from home and when they're on the road. They want to access email using mobile devices. They want to use public computers in kiosks and hotel business centers. Securing that access is becoming increasingly challenging, and many of the old-school security techniques can impose significant overhead and cost that, in today's business climate, isn't always welcome.

This chapter will focus on defining these risk areas, with the goal of creating a fairly comprehensive inventory of the risks we need to deal with. We'll also look at some of the general practices currently used by most organizations to mitigate those risks, and explore some of the downsides. Interestingly, SAN certificates—the topic of the previous chapter—offer a solution to a major portion of those risks, and so we'll look at where SAN certificates fit, what they can do, and what they can't.

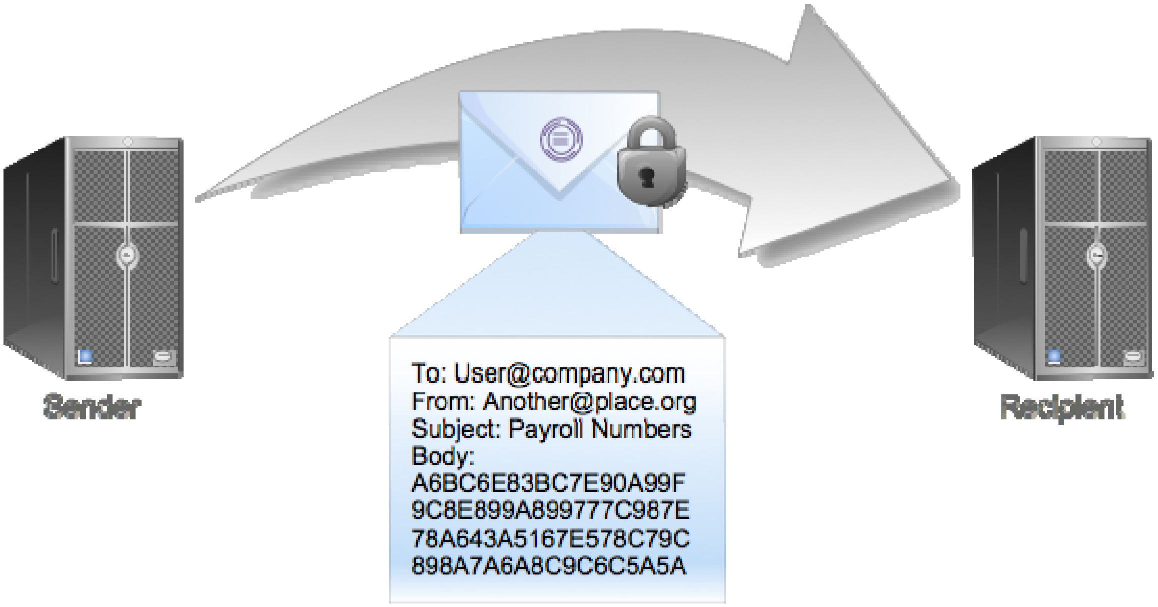
This chapter will focus specifically on Microsoft-centric messaging environments: Exchange Server, Office Communications Server (now called Lync), and so on. However, the principles discussed herein are universal: No matter what messaging system you use, you're going to find truths that you recognize and ideas to help make your messaging security better.

Security Concerns Around Email and Unified Communications

Although there are a lot of technical details around messaging security, they all boil down to two things: privacy and integrity. The privacy bit is probably the easiest one to state: Simply put, we don't want anyone reading our email, listening to conversations, or reading instant messages who isn't *supposed* to be. The actual implementation of that privacy, however, can be a bit more complex. For example, it's all well and good to create an encrypted connection between two servers, but that won't necessarily guarantee privacy. You also need to ensure that the server you're connecting to is *the one you intended.* After all, if you make an encrypted connection *to the wrong server,* you're sending your messaging to someone other than the intended recipient, and privacy is lost.

Integrity is another important aspect of messaging, at least for some messages. With integrity, we may or may not care about privacy, but we need to ensure that our message arrives at its destination exactly as we sent it. That is, regardless of whether we care who *sees* the message, we definitely want to ensure the message wasn't tampered with in-transit. Integrity is often combined with a desire for privacy, but the two are definitely distinct requirements.

Both privacy and integrity can be achieved through the use of digital certificates, although depending on the exact need, they might be different kinds of digital certificates. There are pros and cons, for example, related to the differing ways in which privacy can be achieved, and different kinds of certificates relate to each. Just to set the stage, let's consider an example. Figure 2.1 illustrates how an email can be encrypted using a digital certificate possessed by the sender. Anyone intercepting this traffic can see that there's an email being sent, can access the entire email, and can even see envelope information like who the message is for and who sent it. The body is encrypted, but a determined attacker could certainly make an attempt to decrypt it.



To: User@company.com
From: Another@place.org
Subject: Payroll Numbers
Body:
A6BC6E83BC7E90A99F
9C8E899A899777C987E
78A643A5167E578C79C
898A7A6A8C9C6C5A5A

Figure 2.1: Sending an encrypted email.

Contrast that with Figure 2.2, where the protocol connection between the sending and receiving server is encrypted, which is commonly done by using SSL and an SSL certificate. Here, we can't see "inside" the encrypted communications channel. An outsider intercepting this traffic wouldn't be able to readily distinguish it from any other kind of traffic—it could be a Web page, an email, a file, or anything. There's no information to start with, and the connection could in fact be carrying *multiple* pieces of data. It would be practically impossible to extract all of the packets related to a single email message, let alone attempt to decrypt it. Heck, the message itself might even be encrypted *within* the encrypted channel, making it even more difficult to crack.

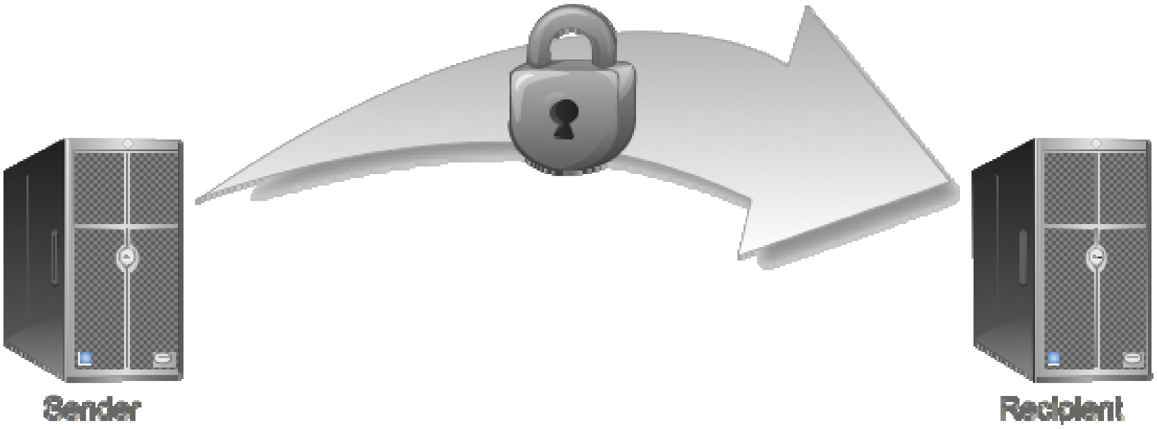


Figure 2.2: Encrypted connection containing…who knows what?

These are the kinds of subtleties we'll explore in this chapter. Neither approach is right or wrong but each one addresses certain security risks and each is appropriate for meeting specific organizational security requirements. Let's start by looking at the risks we're trying to protect against.

Security Risks: Exchange Server on Your Intranet

It's sometimes difficult to think of our internal networks as being anything other than totally safe. That is, if we could somehow live without a connection to the Internet, all would be well and all security risks would be gone. Unfortunately, that's just not true. Although we do tend to devote an inordinate amount of time and effort in protecting against outside threats, the fact is that most attacks continue to originate from *inside* our network, well within the firewalls and other measures we've put in place.

From an Exchange Server perspective, *privacy* is the primary concern. There are a number of places where email messages can be compromised within our network:

- The connection between the sending email client and the server—Network intercepts from within the intranet aren't common, but they're also not unheard of. There's a greater risk for companies who share a building with other companies because it's much easier for at attacker to gain access to the physical infrastructure.
- The storage on the Exchange Server itself—Although most companies provide effective physical security for their Exchange Server computers, it's not uncommon to see the odd messaging server sitting in a broom closet within smaller companies. If you can access those hard drives, you can access all the email they contain.
- The connection between Exchange Servers—This connection is a potential attack point because messages flow freely from server to server as they're routed around the organization. This occurs in all but the smallest organizations where a single Exchange Server computer fills all of the messaging roles.
- The connection between the Exchange Server computer and the intended recipient's client software—Again, this typically requires physical access to the network cabling, so it's rarely seen as a huge risk within most companies, but it's a risk you have to be aware of—even if you choose to acknowledge it and classify it as being too unlikely to worry about.

Most organizations have additional weak points, such as backup tapes used to store Exchange Server backups. These are obviously much easier to get hold of than a server's hard drives, and can be used to re-create an entire server, including all of the messages it stored. Message archival and eDiscovery servers hold additional copies of messages, and generate their own backup tapes to worry about.

It's easy to disregard these risks—for some organizations. For others, paying attention to these risks—no matter how unlikely they seem—is mandated by legislation and industry standards. For example, if your organization stores credit card numbers gathered from customers, then you're required to comply with the Payment Card Industry (PCI) Data Security Standard (DSS). You might think that only applies to, say, your customer database, but the minute any of that information makes it into an email, your Exchange Server computers have to comply with the standard as well. In the US, acronyms such as HIPAA, GLB, SOX, CFR, and others represent legislation that requires certain kinds of data to be kept private, and in most cases, that data is as likely to be in an email as it is to be in some carefully-guarded database.

Figure 2.3 is a visual inventory of the risks so far, with the specific risk areas noted as red triangles. Even the recipient's computer—which is a lot less likely to be physically secured and, in the case of a laptop, can be easily lost or stolen—is a risk point.

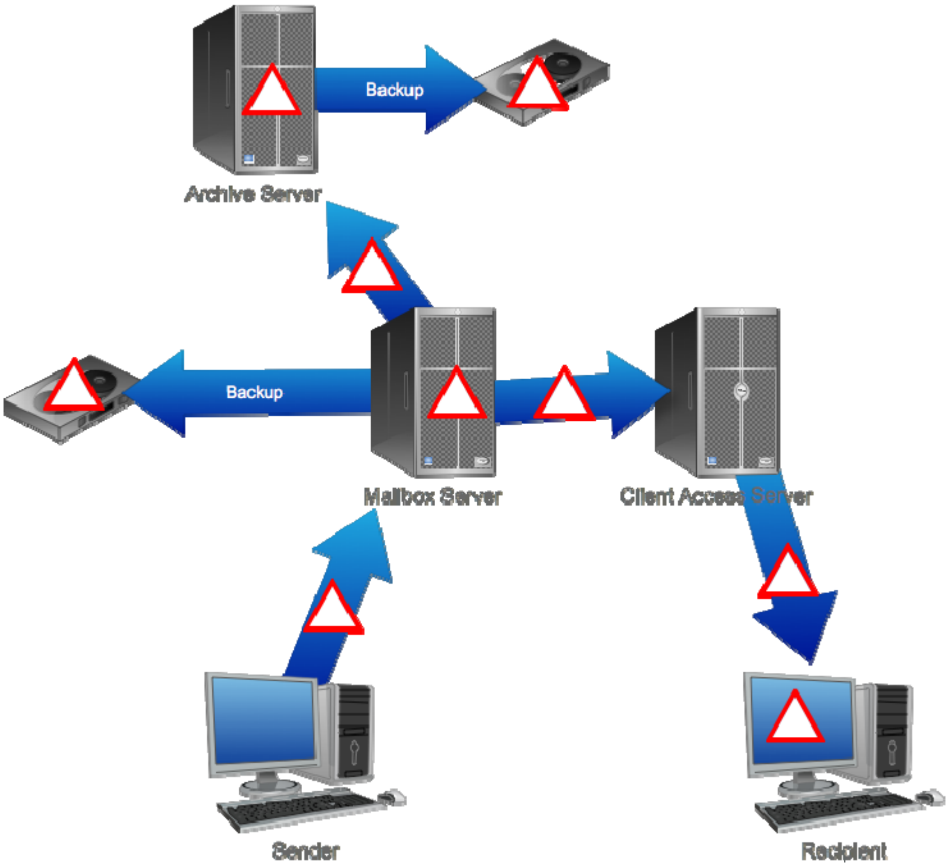


Figure 2.3: Exchange Server risks on the intranet.

Security Risks: Exchange Server on the Internet

Once you leave the “safety” of your intranet, of course, all security bets are off: The Internet is a wild, unregulated place that offers plenty of opportunity for information to be compromised. Figure 2.4 is a starting point for inventorying these risks.

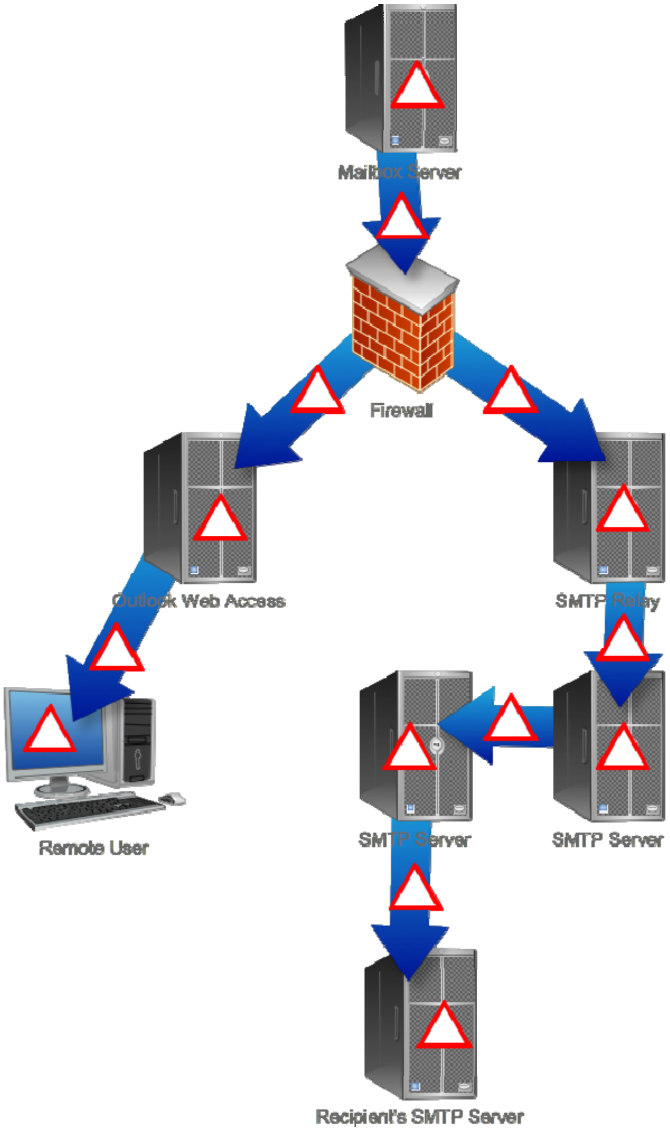


Figure 2.4: Exchange risks on the Internet.

Here's the short story: Email touches too many computers on the Internet, and you have no control over their security.

Even in what may feel like a "closed loop" scenario, there are ample risks. Consider a user employing a corporate laptop to access email via Outlook Web Access (OWA). The user's laptop will cache certain aspects of the messages, making it a privacy risk. The connection between the Web browser and server will traverse numerous routers and firewalls, any of which could intercept traffic and route copies of it someplace else. The Web server has to talk to the Exchange Server, which will commonly happen through a firewall (OWA servers are commonly deployed in DMZs; not pictured is the firewall that would normally protect the Web server from the Internet), and that connection—if someone can get access to the physical media—is a weak point.

Start sending messages to people in other companies, and the sky's the limit in terms of risk. You have *no idea* how many SMTP servers your messages might traverse. You have no knowledge of how they're secured, and you have no means of obtaining that information. Every single one of them will, by definition, be keeping a copy of every message they see for at least a short while—and if they're configured wrong or compromised, it may be a *long* while, or they may even copy someone on your messages without you knowing. This isn't a theoretical risk; it happens every day, all the time. If you haven't been "burned" by this yet, it's not because the problem isn't happening, it's probably because nobody's seen any *interesting* email from you. Yet.

Bottom line: Every server that sees your message, and every connection that carries it, is a risk point. Securing them all individually is probably impossible. Let's just accept that risk and see what can be done to mitigate it.

Security Risks: Unified Communications

"Unified communications" means a lot of different things to different people, so let's focus our discussion on Microsoft's unified communications platform, which is now called Lync. This technology essentially builds off Microsoft's instant messaging platform, adding in support for voice communications, screen sharing, file sharing, and other features. Lync clients include computers, phones, Web-equipped devices, and more, providing a "communicate anywhere" experience.

So where are the security risks? Pretty much the same places they are with Exchange Server: every computer or connection involved in the communication is a risk point. Servers in this scenario tend to store little if any data, which helps decrease the number of potential risk points; however, users are apt to share *anything* across this kind of communication, specifically because it's deliberately easy to do so. Without the right security in place, your users could be broadcasting their desktop, documents, face, voice, and more across the public Internet in a way that's not terribly difficult to intercept, copy, and examine.

Common Legal and Industry Security Requirements

Most organizations have at least a vague, self-motivated desire for messaging privacy. Some organizations, however, have strong external requirements in the form of legislation and industry policies—such as the aforementioned PCI DSS, HIPAA, SOX, GLB, and so on. Although these requirements often drive security policies within affected organizations, in some cases, they can actually *complicate* security. Regardless, it's important to understand what's required so that we can find the simplest, most efficient way of meeting the requirement.

Privacy Requirements

Most laws and industry policies focus on privacy, typically the privacy of some protected group's information. With HIPAA, it's patients' personal data. With PCI DSS, it's cardholder data. With GLB, it's financial customers' data. Regardless, these all boil down to one simple idea: Don't allow protected data to be released to unauthorized individuals, under any circumstances, for any reason, at any time.

That pretty much means that *every single red triangle* in Figures 2.3 and 2.4 represent a potential problem. It's not enough to store data in a secured database if you're going to be broadcasting it in the clear all over your network. In fact, most laws and policies expressly require privacy for data *in transit* as well as data *at rest.* Many organizations know to protect storage-oriented systems like databases, file servers, and the like, but email is *extremely likely* to be used to transmit protected data.

Some organizations attempt to address the problem by simply not permitting email to be used to communicate protected information. In other words, your financial advisor might not be permitted to email you copies of your statements in PDF form, simply because the advisory firm doesn't feel they can secure email sufficiently to comply with the laws that govern them. In some cases, though, you simply have no choice: You need to secure the communications channel somehow. This is especially true for internal communications, such as employees needing to access email from outside the corporate network.

Discovery Requirements

Companies are increasingly being asked to meet strict eDiscovery requirements. Typically imposed by judicial systems, eDiscovery requires companies to, for example, be able to provide all internal communications related to a specific topic when the court issues a subpoena for that information. Companies commonly have a very limited amount of time to provide the information without being fined.

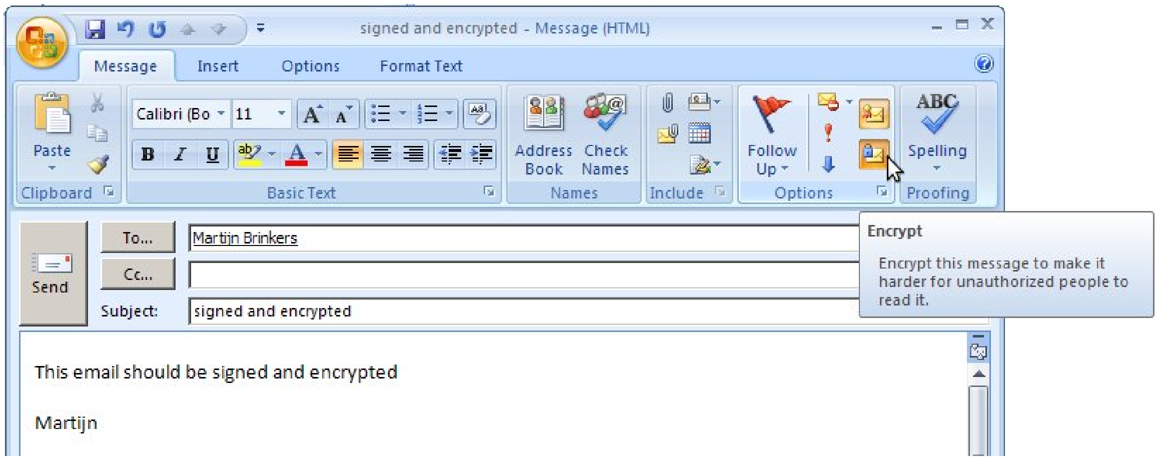
Discovery brings an interesting twist to security. Normally, you could achieve most messaging-related security goals by simply encrypting every individual message, every hard drive where messages are stored, and every communications channel under your control that's used to transmit messages. Not easy to implement, perhaps, but straightforward in theory. In order to implement rapid discovery, however, you pretty much *have* to have *unencrypted* copies of messages in some kind of searchable, indexed database. That makes security a bit harder to achieve, because in many cases, it eliminates per-message encryption as a possible security mechanism. Instead, you have to rely on encrypting message *stores* (for example, server hard drives) and the connections used to transmit messages. That makes securing the *connection* a lot more critical because the messages *carried by that connection* can't be individually encrypted.

General Directions for Improving Communications Security

The next two chapters in this guide will focus on specific mechanisms for implementing security, but in the next few sections of this chapter, I want to focus on general directions, including a good level of technical detail about how it all happens. Much of this will draw from the general cryptography discussion in the previous chapter, so refer back to that if you need a quick refresher on a particular term, acronym, or technique.

Encrypting and Signing Messages

Microsoft Outlook—perhaps the most commonly-used means of accessing Exchange Server—provides full support for encrypting and signing messages. As Figure 2.5 shows, both functions are readily accessible in the software's Ribbon or toolbar.



signed and encrypted - Message (HTML)

To... Martijn Brinkers

Cc...

Subject: signed and encrypted

This email should be signed and encrypted

Martijn

Encrypt

Encrypt this message to make it harder for unauthorized people to read it.

Figure 2.5: Using Outlook to secure individual messages.

Signing, as discussed in the previous chapter, requires that the sender have a Class 1 digital certificate. The signature is generated using the sender's private key, and a copy of the sender's public key is included along with the signature. That copy is generally signed using the issuing Certificate Authority's (CA's) private key (something that the CA provides along with the Class 1 certificate). When received, the recipient must trust the issuing CA, meaning they have to have a copy of the CA's root certificate. That gives the recipient a copy of the CA's public key, which can be used to validate the message sender's public key. The sender's public key is then used to decrypt the signature, verifying the sender's identity and verifying the integrity of the message itself. No privacy is provided, here, but now the recipient has a copy of the sender's public key. Software like Outlook will typically store that public key in its address book, attaching it to the sender's contact card. Remember that the one piece of information that the sender and recipient have in common is a trust for the root CA that issued the sender's digital certificate.

It's a lot of cryptography, and we're not even getting any privacy out of the deal! Figure 2.6 illustrates what's happening.

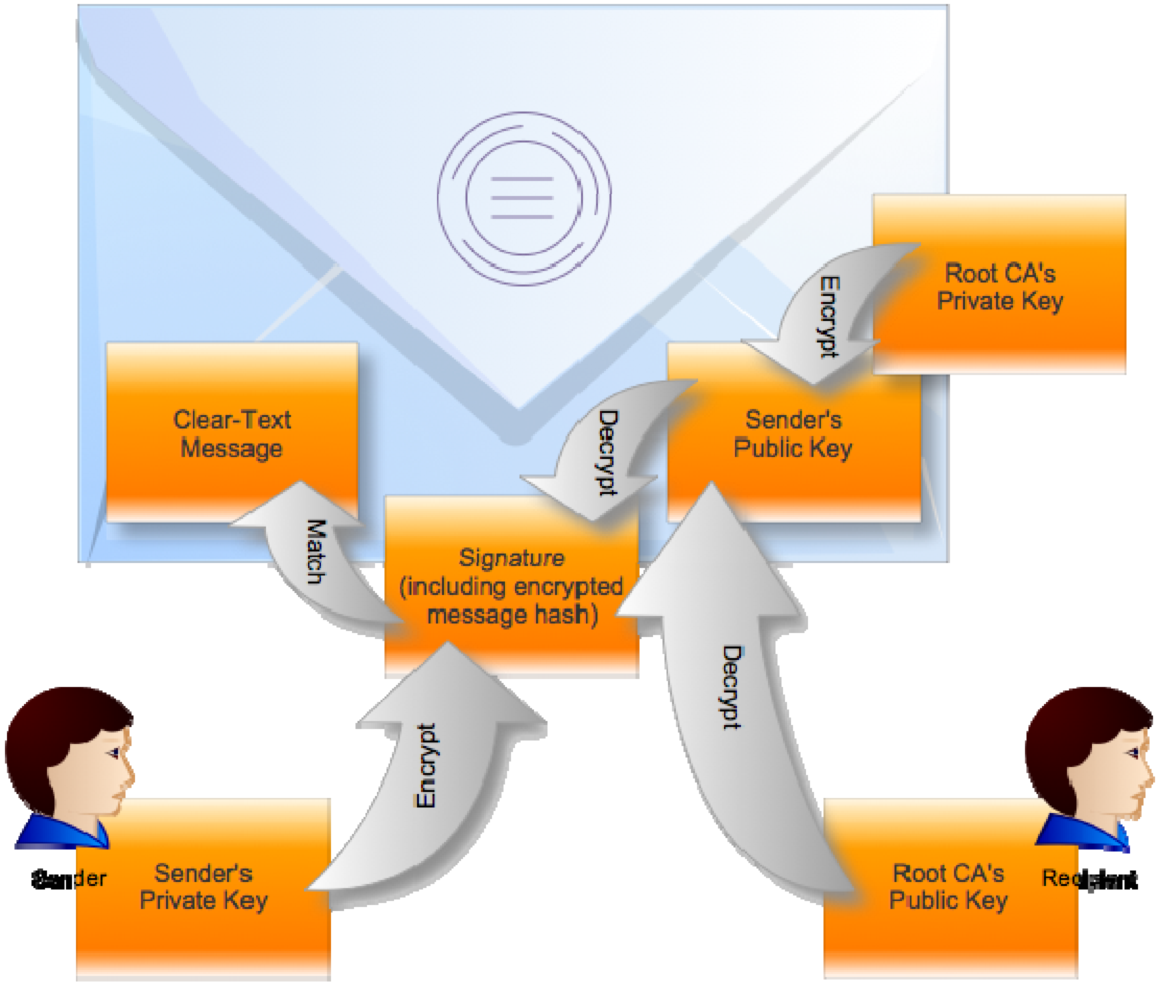


Figure 2.6: Signing a message.

Encryption requires that the sender have the recipient's public key—which is why folks will often ask someone to send them a digitally-signed message because that would contain the needed public key. The recipient's pubic key is used to encrypt the message, ensuring that only their private key—which ideally, only they possess—can decrypt the message. Figure 2.7 shows how this works.

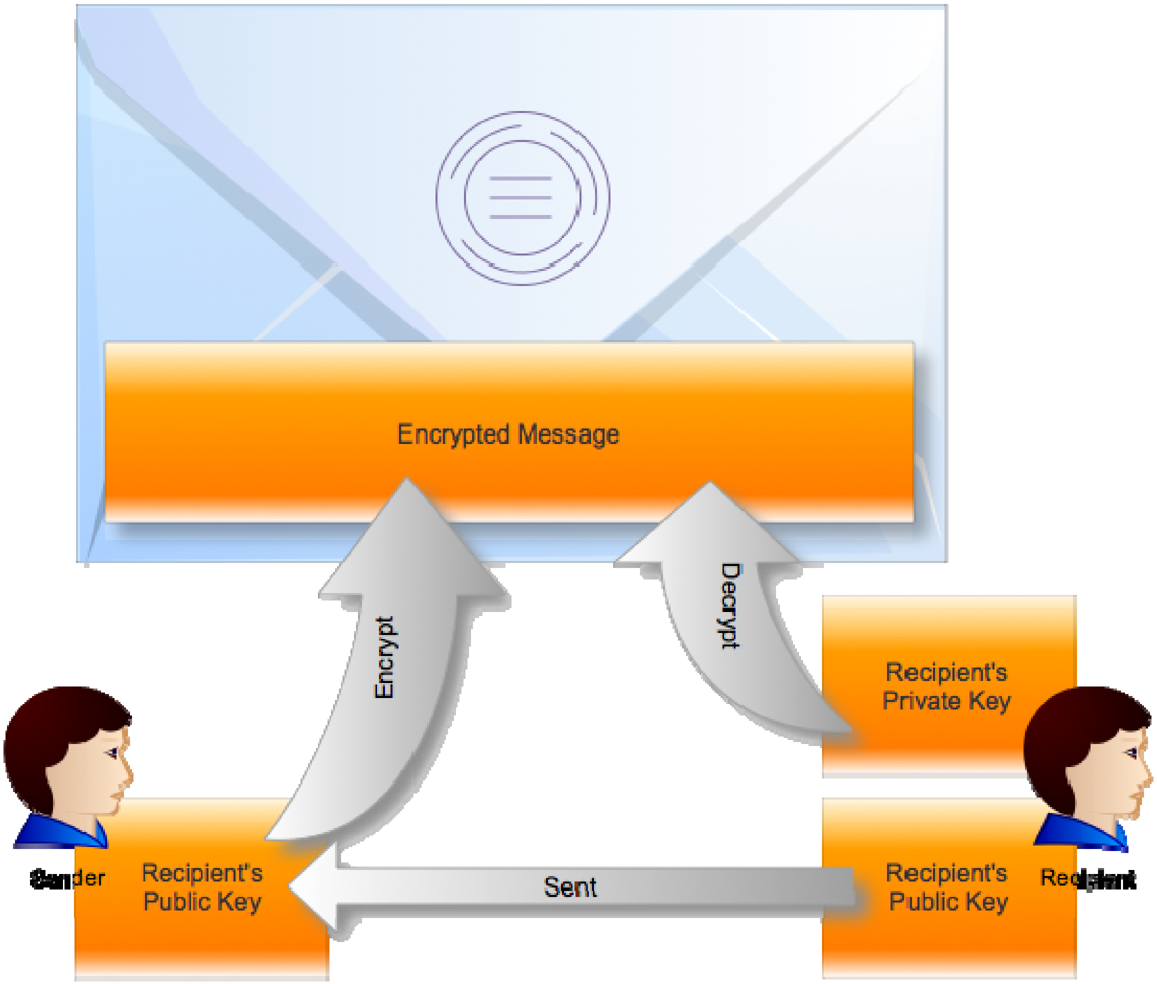


Figure 2.7: Encrypting a message.

Conceptually, this is a lot simpler than signing. A message can be both signed and encrypted by using the recipient's public key for the encryption, and the sender's private key for the signature.

Individually-encrypted messages aren't *completely* encrypted. Only the body of the message, and any attachments, are encrypted. The message *headers*—To, From, and other information—remains clear-text, and must stay that way in order for the message to be properly routed.

Class 1 certificates are easy to obtain commercially because purchasers typically only need to demonstrate control over their Inbox. That is, if they can log in and respond to a verification email, their identity is considered "verified" and a certificate tied to that email address is issued.

A problem with individually-encrypted messages is that, in most organizations, users have to remember to do it. Everyone other than the intended recipient has no access to the message, meaning administrators, auditors, attorneys, and so forth have no access—or, at best, have access only after performing a key-recovery process (assuming the recipient is a company employee). Many companies actually prohibit the use of encrypted messages, and do not issue certificates to their users. Doing so eliminates one potential form of messaging security, but also enables the company to set up their own security measures using other techniques, and ensure that they always have full access to messages sent by employees—a *requirement* for many organizations that are subject to the eDiscovery rules I discussed earlier.

Encrypting Storage

Given that many organizations want to have control over their encrypted data, storage encryption becomes an option. This can either be implemented as whole-drive encryption, a la Windows' built-in BitLocker feature, or per-file encryption that's typically managed by third-party software.

An advantage of this kind of encryption is that it protects against stolen storage media but doesn't really impact accessing the data by authorized personnel. A downside is that you simply can't encrypt every hard drive that your messages will touch. At most, applying whole-drive encryption to *every computer in your company* will protect the messages sent internally; once messages leave your network, though, you can assume that they'll be stored on non-encrypted storage. This is why some organizations simply don't want sensitive data sent via email: They don't want to enable per-message encryption because they would lose their ability to monitor their employees, respond to eDiscovery requests, and easily access data if an employee leaves. However, they can't ensure encryption for messages sent to the Internet. So for those companies, not sending information via email *at all* is the only sensible thing to do.

Encrypting Communications Channels

Protecting data in transit is often the biggest concern companies have. Given the relatively low incidence of server hard drive theft, and even of laptop theft or loss, protecting messages where they sit isn't often considered a major requirement. Protecting the data in transit, however, is a big one.

Exchange Server tends to expose data in several ways:

- For purely-internal client-server communications, either Remote Procedure Calls (RPCs) or RPCs tunneled through HTTP—The latter communications can also be used over public networks, and can be secured by HTTPS.
- IMAP and POP3 protocols, used typically by non-Outlook, non-Exchange-aware clients to access email—Again, SSL can be used to secure these communications.
- OWA's HTTP traffic—This traffic can also be secured via SSL.
- Inter-server communications occur over a variety of protocols, and these can be secured via IP Security (IPSec), using either specific digital certificates or, in the right kind of Active Directory domain environment, by using automatically-generated or even shared encryption keys.

Figure 2.8 illustrates these different channels. The public communications—HTTP, IMAP, and POP—are the ones many companies worry about the most. These occur between client applications and Exchange Client Access servers; communications on the public Internet typically pass through a firewall of some kind. Securing these with SSL is straightforward, and is something we'll examine in the next chapter.

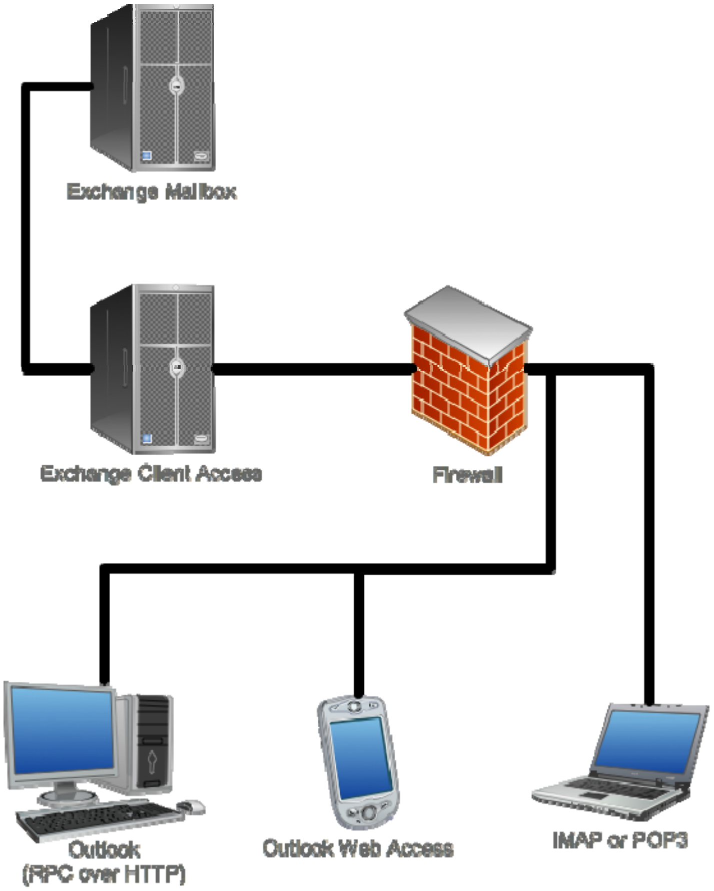


Figure 2.8: Exchange communications summary.

Authenticating Communications Channels

As mentioned earlier, it's rarely enough to simply *encrypt* communications channels. After all, if you aren't talking to the right server, you could be giving away your information whether it's encrypted or not. That's why *authentication* is an inherent part of the SSL/TLS protocol set. SSL/TLS doesn't require *mutual* authentication, which is generally a good thing: You don't care so much about what *machine* your users are on as you do the *users'* identity, so Exchange typically requires users to log in (often by providing a username and password). The server itself is authenticated by means of its SSL certificate. Here's how it works:

1. The client requests, or the server requires, a secured connection. In the case of the client, this usually starts as a protocol request, such as requesting an HTTPS Web page as opposed to an HTTP Web page. However, servers can be configured to only accept HTTPS; in those cases, the server will often accept an HTTP connection but will instruct the client to redirect to an HTTPS connection.
2. The server presents its SSL certificate to the client, along with a signature. This contains the server's public key, which is usually digitally-signed by the issuing CA's root private key. The client must trust the CA, meaning they need to have a copy of the CA's public key—just like in a digitally-signed email.
3. The client uses the CA's root public key to decrypt the signature, which validates the server's public key. The client uses the server's public key to decrypt the signature. This validates the server's identity—and the client checks to make sure the certificate's identity matches what the client was trying to access.
4. Typically, the client then calculates a new encryption key, encrypts *that* with the server's public key (meaning only the server can read it), and sends it to the server. This is often a *symmetric* key, meaning both the server and client will use this same temporary key to encrypt and decrypt traffic for this session.

This is a simplification; there are other steps involved where the two computers negotiate things like encryption key strength and algorithm based on the lowest-common-denominator supported by both. Figure 2.9, from the Computer Desktop Encyclopedia, illustrates some of these additional steps.

> **Note**
>
> Encryption is *not* mandatory with SSL, nor is authentication. A client could choose to ignore the authentication steps and simply accept the server's assertion of identity—by not, for example, checking the CA's signature on the server's certificate. The client and server can also agree to not use encryption once the channel is authenticated. However, in most real-world implementations, SSL is synonymous with "encryption with authentication of the server computer's identity."

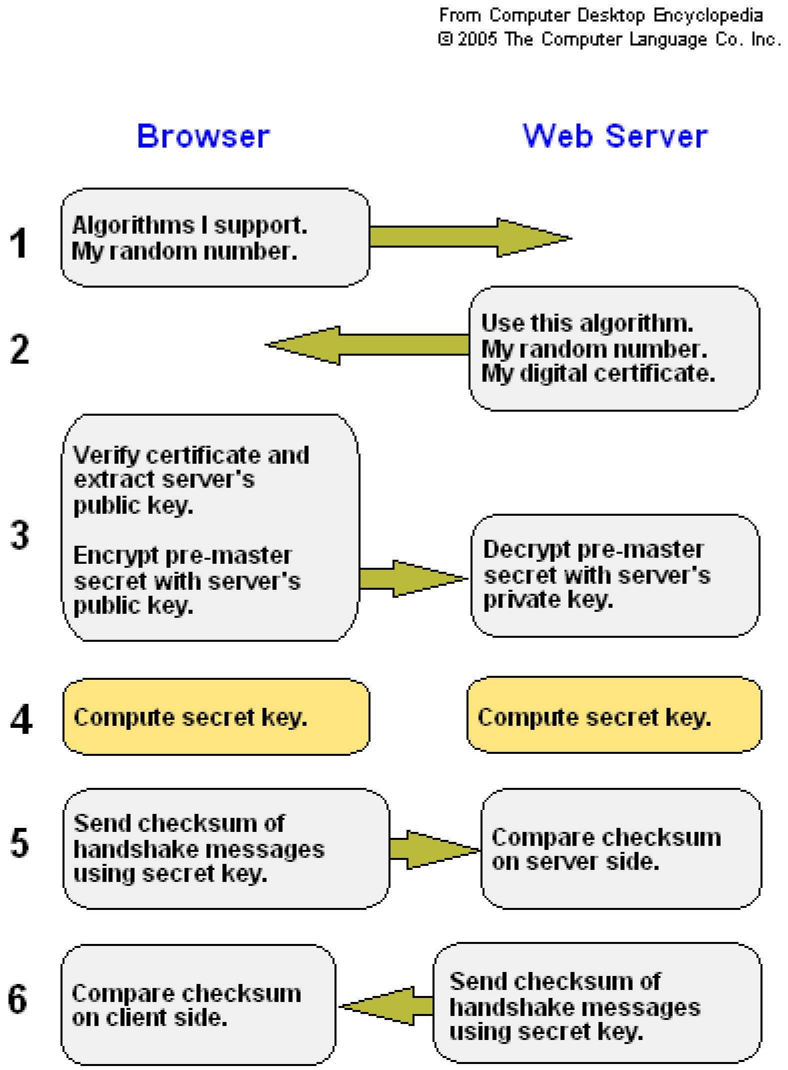

From Computer Desktop Encyclopedia
© 2005 The Computer Language Co. Inc.
Browser
Web Server
1
Algorithms I support.
My random number.
2
Use this algorithm.
My random number.
My digital certificate.
3
Verify certificate and extract server's public key.
Encrypt pre-master secret with server's public key.
Decrypt pre-master secret with server's private key.
4
Compute secret key.
Compute secret key.
5
Send checksum of handshake messages using secret key.
Compare checksum on server side.
6
Compare checksum on client side.
Send checksum of handshake messages using secret key.

Figure 2.9: Negotiating an SSL connection.

Provisioning Mobile Devices

A last general approach to messaging security pertains to mobile devices. Because they're relatively new (compared with laptops), mobile devices often come with more modern security features, many of which can be activated server-side.

For example, most modern smartphones are capable of accepting a server certificate for authentication and encryption. That authentication process can also include instructions on how the mobile device must behave (a feature explicitly supported in Exchange's ActiveSync technology, but also supported by companies who emulate ActiveSync, including Google's corporate Gmail product). The mobile device might be instructed to lock itself after a minute of inactivity, and to require a password or PIN in order to unlock. This behavior can help make the device more secure: If it's lost or stolen (something that's easier with a phone than with a full-sized laptop), the thief would be less able to access any data stored on the device, and less able to use the user's stored credentials to access the messaging server.

User Education: Cues and Prompts for Better Security

Very few of your security precautions will be effective without some end-user education. For example, let's suppose that you implement full SSL security on all of Exchange Server's communications channels. Great—server authentication and encryption are in place. But, particularly with OWA connections, that's useless unless your users know what to look for, and what *not* to do.

For example, Figure 2.10 shows some of the visual cues—many of which are small and unobvious—that users must look for in order to ensure they're getting an encrypted, authenticated connection. Without this cue, users could be connecting to an incorrect server, which could be impersonating a corporate OWA site in an attempt to garner user email addresses, user names, and passwords.

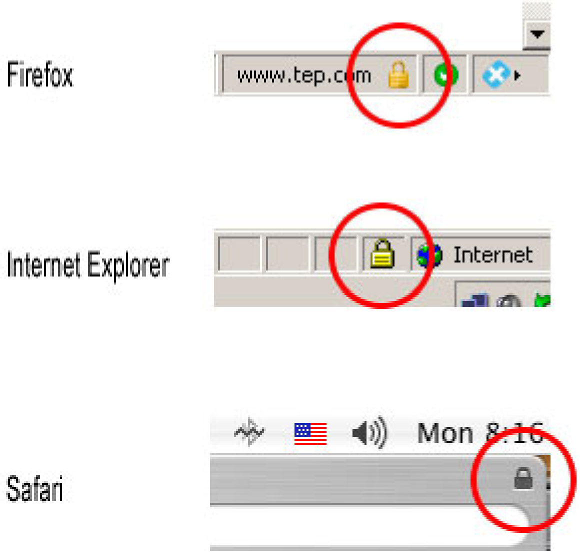


Figure 2.10: Browser SSL cues.

Helping users understand what security measures are in place, and what they can (and should) do to *verify* those measures, is as important as getting the right SSL certificates in place, the right encryption enabled, and other security steps.

Coming Next

In the next chapter, we'll look specifically at security practices for Exchange that can help address the security issues and challenges we've identified in this chapter. We'll use the basic techniques outlined in this chapter, but address them specifically to the Exchange environment.

Download Additional Books from Realtime Nexus!

Realtime Nexus—The Digital Library provides world-class expert resources that IT professionals depend on to learn about the newest technologies. If you found this book to be informative, we encourage you to download more of our industry-leading technology books and video guides at Realtime Nexus. Please visit http://nexus.realtimepublishers.com.