

Chapter

3

Link Encryption

Everything should be made as simple as possible, but no simpler.

—Albert Einstein

IN THIS CHAPTER

Our first crypto technique is *link encryption*, the classic method of applying crypto to digital communications. Designed to hide secrets, link encryption can also protect data against forgery if used properly. Link encryption is a simple concept that can fit transparently into existing communications applications. The following topics are discussed:

- Security objectives achieved by link encryption
- In-line encryptor hardware that implements link encryption
- Point-to-point deployment example
- IP-routed deployment example
- Key recovery and escrowed encryption

Link encryption may not be the best choice for most applications, but it is the simplest, so we look at it first.

3.1 Security Objectives

Like any security measure, link encryption should only be used if it matches your security objectives. It is the traditional choice when you need complete separation between insiders and outsiders. Link encryption makes it easy for insiders to share data and almost impossible for outsiders to do so. Protection is transparent, except for the strong boundary between inside and outside. Here are specific security objectives you can achieve with link encryption.

- **Maintain confidentiality on an isolated set of computers.**
Your computers contain very sensitive data. You need to exchange data with other sensitive sites and keep the risk of leakage as low as possible.
- **Communications with outsiders is unwanted and to be blocked.**
You do not want to exchange data with unauthorized sites, and you want to prevent it from happening through accident, carelessness, or overt attempt.
- **Hide data traffic as much as possible.**
You really need to shield everything possible about the data you send, including details of message sources and destinations, and other communications control information. You assume that “insiders” will not leak the information you are trying to protect.
- **Safety and familiarity is more important than cost.**
You wish to use a well-established technique that is simple to understand. You are willing to pay a premium even though the technique may be more difficult to operate.

Here is a final security objective, listed separately because it is not unconditionally provided by “pure” link encryption:

- **Protect data transfers from forgery by outsiders.**
You need to protect data from tampering while in transit between authorized sites. You need assurance that the contents of an incoming message were in fact produced by a host at an authorized site.

Link encryption yields a highly reliable design from a security standpoint. If you have established a strong security perimeter within your organization, link encryption gives you a proven technique for maintaining it. If you strictly control the flow of physical documents in and out of the security perimeter, link encryption can provide complementary protection to electronic documents. You can build an environment with encryptors on every data link traversing the boundary. All readable electronic documents will reside inside the perimeter and the encryptors will protect any electronic data that leaves. This technique has been used for decades to protect the most sensitive military traffic, and to provide secure links between banking organizations.

Since link encryption is the oldest approach to data security, it has also developed a long history of failures from operational and technical flaws. In its simplest form, link encryption does not reliably protect against forged traffic. It blocks outside access so thoroughly that users will sometimes use dial-up links to secretly punch holes in its defense. And, like all crypto measures, it is vulnerable to risks from improper key management.

3.2 Product Example: In-line Encryptor

An in-line encryptor is the building block for link encryption. It is a hardware device with two data ports: one always handles plaintext and the other always handles ciphertext. When plaintext data arrives on the plaintext port, the encryptor transforms the data into ciphertext and transmits it out the ciphertext port. Data is likewise transformed from ciphertext to plaintext after arriving on the ciphertext port. A pair of encryptors will pass data between one another only if the crypto keys have been set up correctly between them.

A modern in-line encryptor can be a relatively small device with two network data link connections and a power supply connection (Figure 3-1). Some in-line encryptors are packaged as encrypting modems so that one data port is a connection for a communications line, like a telephone jack. Some devices will have a facility for manually installing or specifying a secret key, while others may permit keying via the data link connections.

In-line encryption can be applied to any data link technology. It is possible to construct encryptors to handle asynchronous or synchronous serial lines. The data being encrypted can be structured as packets and the encryption embedded in a network bridge device.

Be sure to check the encryptor's operating speed against your data link requirements. Crypto can be slow, so choose products and design systems accordingly. So-called "hardware"



Figure 3-1: IN-LINE ENCRYPTION DEVICE. This is a commercial in-line encryption device—the IRE Model HS Remote Encryptor. It is small enough for convenient desktop use. Photo courtesy of Information Resources Engineering, Inc.

implementations of some algorithms may in fact be slower than “software” implementations of newer algorithms.

3.2.1 Red/Black Separation

Rule 1 of cryptanalysis: check for plaintext.

—Robert Morris, NSA (retired)

Separate plaintext and ciphertext ports reflect a well-respected principle of good crypto design: *red/black separation*. The term is taken from a military tradition in which red data refers to plaintext and black data refers to ciphertext. The principle is that a well-designed crypto device will keep the two separated as much as possible (Figure 3-2). In high-end military systems this is taken to the logical extreme. Every part of a circuit is specifically identified according to whether it carries plaintext or ciphertext. Some devices have multiple power supplies to service the plaintext and ciphertext circuits separately.

There is a Cold War story about a clever bit of tunneling performed under the Berlin Wall. The target was a telephone conduit under the pavements of East Berlin. The eavesdroppers were pleased when they tapped in to the cable and detected encrypted Soviet message traffic. According to the story, the code itself was never cracked and never needed to be. The Soviet’s encryp-

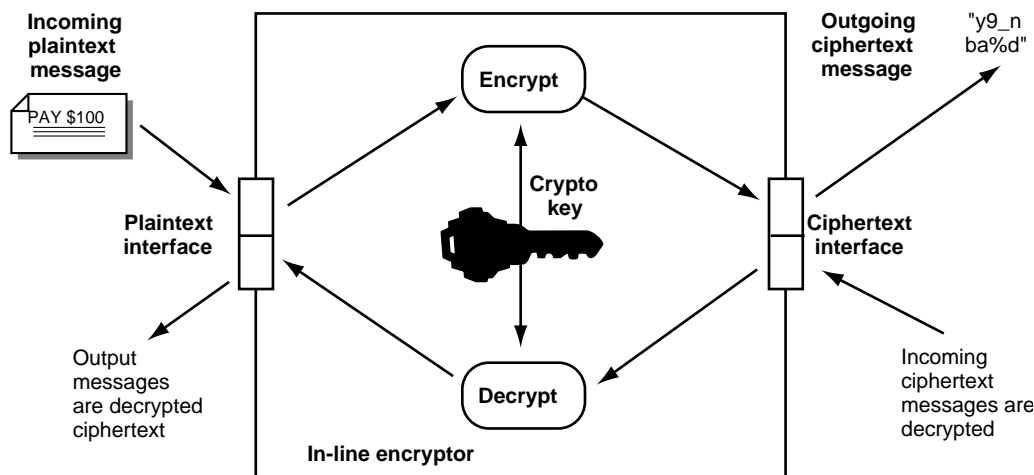


Figure 3-2: INSIDE AN IN-LINE ENCRYPTOR. There are two separate interfaces: one for plaintext and one for ciphertext. Data entering through the plaintext interface is always encrypted and sent out the ciphertext interface, and vice versa. The safest encryptors provide no way to “bypass” the encryption; crypto is applied to everything passing through it.

tion equipment had a flaw in its red/black separation. The plaintext of each encrypted message was also sent across the cable as unintended “random noise.”

While Cold War attacks and security measures may be overkill for commercial applications, the lesson is clear: If you keep plaintext and ciphertext separate, you are less likely to mix up the two. If you transfer plaintext and ciphertext through the same network interface, you run a much greater risk of sending plaintext data in the wrong direction. This mistake is much harder to make if the networks are kept completely separate and plugged into separate ports on the encryptor.

3.2.2 Crypto Algorithm and Keying

Since the in-line encryptor processes a stream of data, we must either use a stream cipher or a block cipher with an appropriate streaming mode. In practice, a stream cipher is an unlikely choice. The only stream cipher with a significant commercial reputation is RC4, and in-line encryptors using it are rarely encountered. Most commercial devices were produced for the banking industry and implement the DES algorithm in some form. SKIPJACK-based encryptors have also appeared that use the Fortezza crypto module to perform the encryption operations; some of these systems incorporate a “key recovery” facility that is described in Section 3.5. Other acceptable block algorithms might be used in newer products. (Consult Section 2.3 to evaluate other choices.)

Block ciphers will require a mode in order to process the data stream. The CBC mode is widely used in this application, though CFB also would be a reasonable choice.

As noted earlier in Section 2.2.2, DES with a traditional, shorter 56-bit key is vulnerable to attack by a sufficiently motivated adversary. If you are hiding trade secrets from an aggressive foreign government, or commercial transactions that are particularly vulnerable to forgery, you may want a product with a longer key. Triple DES implementations would be a good choice.

An active link will carry a lot of traffic. Since keys become easier and more attractive to crack as more data is encrypted with them, the preferred device should have a way of rapidly switching keys. Many devices implement automatic key generation or exchange facilities. Devices used in the banking industry often follow American National Standards Institute (ANSI) standards for key exchange. Key handling protocols and capabilities are introduced in Section 4.5.

3.2.3 Encryptor Vulnerabilities

As with any security device, it is easier to choose the right one after considering the problems, from which they may suffer. This section reviews the following vulnerabilities:

- Replay attacks
- Rewrite attacks
- Covert signaling attacks

REPLAY ATTACKS

Many people assume that encrypted data is self-validating, since a message that decrypts sensibly can only be generated by a matching encryptor with the appropriate keys. The problem is that outsiders will have access to lots of encrypted data, and thus to data that “decrypts sensibly.” If an outsider captures some external data traffic and retransmits it, the encryptor might simply decrypt it a second time and deliver it like any other data it successfully decrypts. This can be a problem if an outsider can recognize and intercept an encrypted message of personal value.

Figure 3-3 shows Alice sending a payment order to Bob. Payment orders are only visible as ciphertext outside their respective sites. This payment order happens to benefit an outsider named Play-It-Again Sam. Somehow or other (and these things are much easier to arrange than we might like) Sam heard about the payment order and intercepted an encrypted copy of it. If Sam simply retransmits the encrypted message a few more times, Bob will receive additional payment orders to process. The encryption will not detect the forgery itself.

An interesting property of this attack is that it depends on the attacker knowing the contents of a particular message. Such attacks are called *known plaintext attacks*. It is always easier to attack crypto protections if the attacker has plaintext messages that match up with intercepted ciphertext.

This type of attack is thwarted by embedding extra information in the data so that duplicate transmissions are detected. For example, the plaintext message could include a message number,

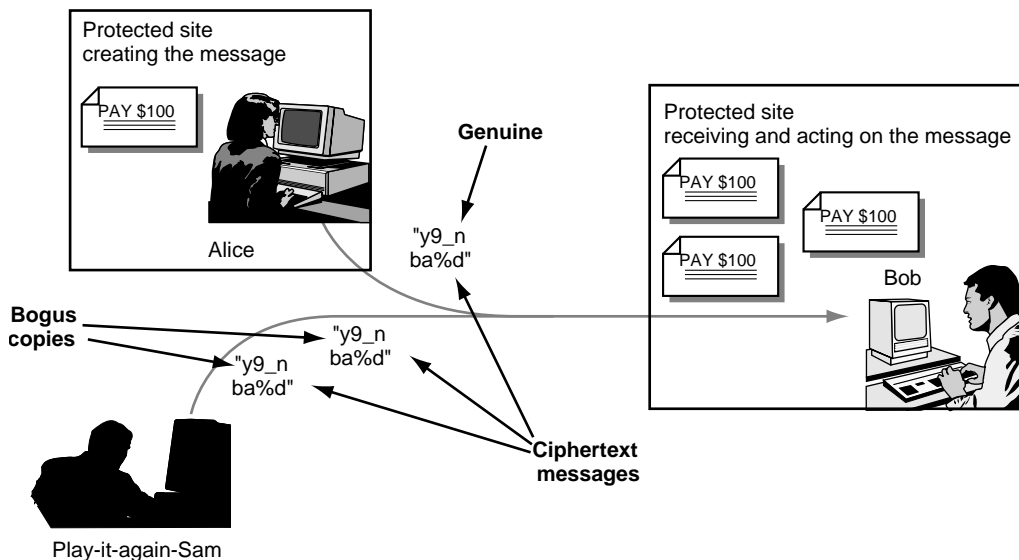


Figure 3-3: A SUCCESSFUL REPLAY ATTACK BY PLAY-IT-AGAIN SAM. Sam made a copy of a legitimate, encrypted message originally written by Alice. He gets money each time the message is sent, so he retransmits a few copies of it, without decrypting or otherwise changing it. Bob treats the copies as legitimate, since they were encrypted with the correct crypto key.

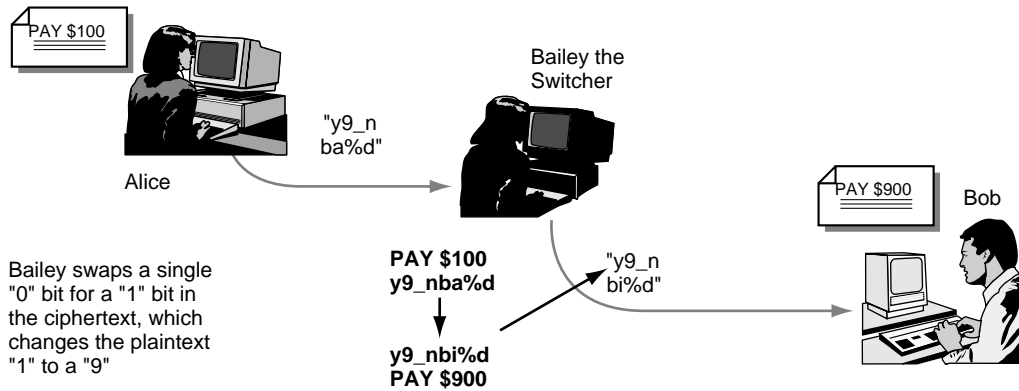


Figure 3-4: A SUCCESSFUL REWRITE ATTACK BY BAILEY THE SWITCHER. Bailey gets a percentage each time Bob processes a PAY message, so she decides to increase her income. She intercepts the ciphertext message and, without decrypting it, changes the dollar amount by swapping bits in the ciphertext. When Bob decrypts it, the ciphertext changes Bailey made will directly modify the plaintext.

and the recipient could discard any messages with a duplicate number. Connection protocols like the internet TCP protocol provide some degree of replay protection.

REWRITE ATTACKS

A rewrite attack is another known plaintext attack that forges messages. Certain crypto algorithms are vulnerable to it, depending on the technical aspects of how the plaintext and key are combined to produce the ciphertext. Simple mechanisms that use binary addition and rely primarily on hard-to-guess keys are especially vulnerable. For example, *one time pads* are vulnerable to this attack. This is important because people often make too much out of the theoretical security properties of one time pads. A true one time pad is *unconditionally secure*, a mathematical property indicating that brute force decryption is ineffective against it. In other words, attackers cannot extract the contents of a message if they do not already know it. However, a message is vulnerable to modification if the attackers know its contents and intercept it in transit.

This type of attack is illustrated in Figure 3-4. Again, Alice is sending a payment message to Bob. This time an outsider called Bailey the Switcher has found out about the message transmission and seeks to benefit by manipulating the message. If Bailey knows the exact contents of the message, she can combine it with the desired message to identify which bits in the encrypted message need to be changed. This works easily when there is a direct relationship between each bit in the plaintext and the corresponding bit in the ciphertext. If you change a bit in one, the same bit will change in the other. In Figure 3-4, Bailey changes a single bit in the letter “a” of the ciphertext, turning it into the letter “i.” This produces a corresponding 1-bit change in the plaintext, replacing the value \$100 with the value \$900.

As with replay, this attack is often thwarted by adding extra information in the data being sent. Here are some approaches that a good product might use to defend against these attacks:

1. **Block mode algorithm using CBC or CFB modes.** Avoid products using other modes, straight block ciphers, or Vernam techniques unless the product provides some other protection against rewrite attacks. Some alternatives are noted here. However, keep in mind that crude rewrite attacks are still possible even with CBC or CFB.
2. **Special packet format with random data.** If the underlying network is implemented using packets, the product can reformat the data into a special packet that defends against rewrite attacks. One approach would be to insert a random number into each packet, include it in the packet checksum, and encrypt the resulting packet. An attacker would not be able to predict the random number and thus could not reliably modify the packet.
3. **Special packet format containing a *crypto checksum* based on keyed data** (Section 5.3). This is a variant of the previous technique that is used in the IPSEC protocols described in Chapter 5. Another more expensive alternative is to use a *digital signature* to protect the data from modification (see Chapter 11).

COVERT SIGNALING ATTACKS

This last type of attack simply illustrates that link encryption's strong boundary does not unconditionally keep secrets inside. There will always be a way to leak information if there is a process inside that tries to do so. Generally this is not a problem in commercial settings since the need for confidentiality rarely justifies extreme security measures.

The technique is sometimes called an INFOSEC attack since it combines an attack on the site's crypto protection with an attack on its internal computers. The attack consists of inserting a subverted program into a host on the plaintext side of an encryptor. The program collects sensitive data and then transmits it to a program outside the security boundary by generating distinctive patterns in the stream of ciphertext messages. The actual techniques will depend on the system being attacked, but there will always be a way to send signals according to the formal dicta of information theory.

We only need to pay attention to this threat because it establishes a limit as to how much privacy protection crypto can give us. There have been no reported or even rumored instances of such an attack being used against a commercial target. For most applications this remains, at most, a theoretical risk. For most attackers it will be less costly and more reliable to try to locate and copy the physical plaintext than to use this technique.

3.2.4 Product Security Requirements

Here is an ordered list of basic security requirements for a high-security in-line encryptor. Remember that this is not the whole list. You will encounter more requirements when we look at how you will deploy the encryptors.

1. **Separate network connections for plaintext and ciphertext.** Keeping these connections physically separate is the most reliable approach. It is much too easy to mishandle plaintext and send it to the wrong recipient if both are sent through the same network connection. If you are spending the extra money and administrative overhead to do link encryption, be sure you buy equipment that is most likely to do it correctly.
2. **Crypto key length.** The device must provide a key size consistent with the sensitivity of the information you send across the data link. Review the questions in Section 2.3.4 if you need guidance on establishing your key size requirement.
3. **Crypto key changing.** The best devices will provide mechanisms to change the crypto key automatically and periodically or whenever a connection is established. This is preferred since it reduces the amount of ciphertext an attacker sees that is encrypted with a given key. Review the examples in Chapter 4 against the capabilities of candidate in-line encryptors.
4. **Crypto algorithm.** This, plus the key, determines how well your data is protected after it leaves your site. Your best choice is a respected algorithm like DES or one of the others noted in Section 2.3.2.
5. **Compliance with crypto hardware standards.** The U.S. government published Federal Information Processing Standard (FIPS) 140-1, which provides specifications for cryptographic modules used to protect digital communications.
6. **Replay and rewrite attack protection.** Is it likely that an attacker can identify a packet with predictable contents and cause you damage by modifying or replaying it? The answer depends on your traffic. Choose a product that defends against rewrite attacks if you're unwilling to predict what protocols your network might use.

3.3 Deployment Example: Point-to-Point Encryption

This first deployment example does not necessarily involve Internet protocols. Point-to-point encryption can be applied to any protocol, standardized or custom, that passes between a pair of hosts over an untrustworthy data link. This deployment technique directly supports all the link encryption security objectives: strong isolation, traffic hiding, and simple design. It is also subject to every in-line encryptor vulnerability listed in the previous section.

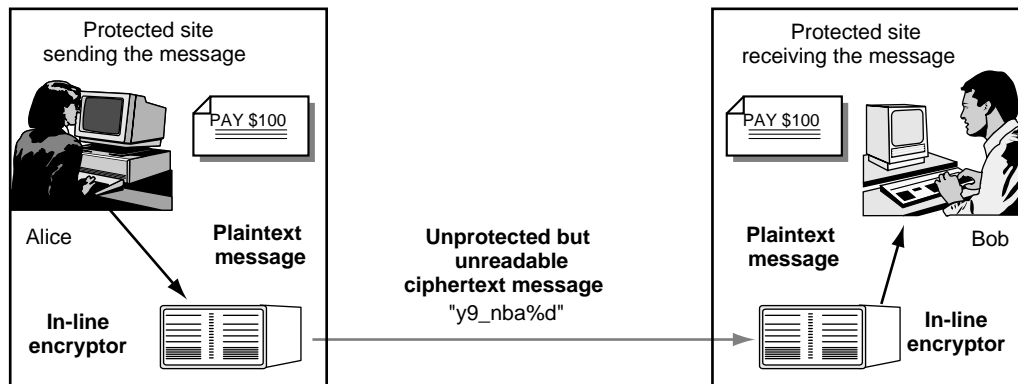


Figure 3-5: HARDWARE ARRANGEMENT FOR LINK ENCRYPTION. Bob and Alice protect their information by keeping it inside the protected site in which they work. Messages only leave a site by traveling through an in-line encryptor. The workstation, encryptor, and the external network connection being used all reside within the protected perimeter of the site. Only encrypted data leaves the site.

In its simplest and most secure form, each connection to another external host uses a separate data link and in-line encryptor. Each host, secure within its own physical site, passes its data through an in-line encryptor before reaching the modem or bridge that connects to another host somewhere in the outside world. Each pair of hosts arranges to use identical encryptors and the same cryptographic keys.

Figure 3-5 shows the hardware arrangement. Each host's data link is connected to the plaintext port of the in-line encryptor. The ciphertext port of the encryptor is then connected to the modem or bridge connecting to the outside data link. The host has no connections outside the site's physical boundary except through an encryptor. This ensures that outsiders can't access the hosts, since outsiders don't have the right crypto keys. Existing workstation software is unaffected by the link encryptors, as shown in Figure 3-6.

This technique has long been a favorite in high-security military applications. It disguises the data contents most completely. More importantly, it gives users the least opportunity to accidentally (or intentionally) leak data.

3.3.1 Point-to-Point Practical Limitations

Point-to-point encryption has a brute force simplicity that makes it an appealing solution from a security standpoint. However, it has certain properties that make it hard to use in every application. First, system users have no choice regarding encryption. You can either link to a host using encryption or you can't communicate at all. Second, it can be difficult to establish a link to a new host. These facts make point-to-point encryption a bad choice for some sites.

Encryption will never be free, nor will it soon be available to every possible destination. Not all data benefits from encryption. Information of a public nature is best sent in plaintext if

only because plaintext requires fewer computing cycles. This reduces the response time needed to retrieve the information. If a host only connects to other hosts via encryption, then there are hosts with which it will never be able to communicate. Any mandatory encryption arrangement is a double-edged sword. It protects from hostile outsiders, but also blocks access to beneficial outsiders.

Another problem is that point-to-point encryption scales very poorly. There is no prevailing industry standard for an in-line encryptor, so you will probably need to purchase more equipment each time you establish a secure link to another host. You must coordinate equipment, algorithms, and keys with every host with which you ever need to communicate, even if new equipment doesn't need to be purchased. This rapidly becomes extremely complex and expensive even if the number of hosts increases slowly.

3.3.2 Physical Protection and Control

Physical protection is the application of red/black separation to your host computers and the information they contain. All security begins with physical protection. To ensure the security provided by the crypto services, the devices using and providing the services must be physically protected. In addition, it is important to protect the integrity of the software on your hosts. Software on your host computer can send messages and perform computerized actions on your behalf. Subverted software can do so, too, and not necessarily with your knowledge or consent.

The first step is to control the physical integrity of your hosts. Typical organizations keep computers in offices and rarely give visitors or guests access to those computers. Provide additional physical security when it makes sense. For example, careful organizations place the payroll department in a separate office area with controlled access. The area is locked when the office is normally closed. This blocks physical access to those machines and reduces the risk of direct misuse.

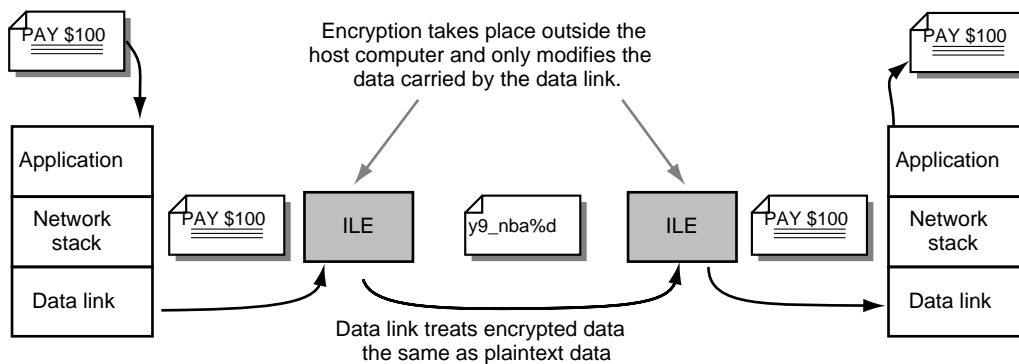


Figure 3-6: IN-LINE ENCRYPTORS DO NOT AFFECT EXISTING, INSTALLED SOFTWARE. Link encryption can operate completely outside the view of workstation software, retaining compatibility with existing applications. Changes can be restricted to hardware at the LAN level.

Another physical security measure is to ensure that every external data link goes through an encryptor. A fully secure point-to-point network will not have any dial-up modems except those that go through an encryptor. In practice many organizations are finding this to be a difficult restriction with which to live, because modern networking has left the point-to-point approach far behind. We shall discuss that problem in the next section.

It is very important to protect your in-line encryptors from physical access by outsiders. Physical access gives them several opportunities to impair your security. Physical access might give them a way to extract crypto keys, allowing them to masquerade as a legitimate host when talking to your other hosts. An even easier trick is simply to disable the encryption by rewiring the network connections or by manipulating the encryptor somehow. A similar trick was played on an Army unit several years ago while it was deployed overseas for several months. The soldiers had trouble keeping their encrypted radios working and used local technicians to repair them. On returning home, engineers found that the local technicians had essentially disabled the radios' encryption.

A final physical security measure is to control the integrity of the software on your hosts. Virus checkers are the state of the art for most desktop systems, but they will not catch a customized attack on your own hosts. If you manage money or resources that are valuable enough to others, they could someday send you a program that generates messages or other transactions to their benefit without your knowledge or participation. This might not look at all like a virus. It might just look like a legitimate application program. The most effective defense is to compute checksums on all critical, unchanging files like executable binaries, and to validate the checksums regularly. In higher risk environments, cryptographic checksums should be used. This will catch any changes to executable code even if the change doesn't match the pattern of a known virus. Another important defense is to only accept software that clearly comes from a trustworthy source. But remember that no source is unconditionally trustworthy. In the past, major vendors have occasionally distributed viruses with their software.

3.3.3 Deployment Security Requirements

The following requirements for deploying point-to-point link encryptors are listed in priority order with the most important requirements given first.

1. **Appropriate link encryptors.** Select link encryptors after reviewing them against the product security requirements specified in Section 3.2.4.
2. **Physical isolation of hosts and encryptors.** Physical security measures must block outsiders from physical access to the hosts that handle encrypted traffic. The encryptors themselves must also be protected. Physical access gives outsiders several ways to bypass or subvert the crypto protections.
3. **Software integrity control.** Take steps to ensure that hosts on your network only execute software that is trustworthy and is known to do its job correctly. Regularly scan your

hosts for viruses or other potential infestations of subverted software. Virus checkers provide some protection but the best defense is to maintain and verify checksums on all critical files.

3.4 Deployment Example: IP-routed Configuration

Naturally, link encryption can also be applied to links carrying IP traffic. This yields a much more flexible and capable networking environment. In the IP tradition, individual site LANs connect to intrasite data links, and the intrasite data is passed through the in-line encryptors. Outsiders still see nothing but encrypted traffic.

As shown in Figure 3-7, almost all of the packet gets encrypted when link encryption is applied. The data link header is left in the clear so that the low-level network can process the packet. All remaining protocol headers are encrypted and will not be processed until after the packet is received and decrypted. If we follow the packet flow in Figure 3-8, we see that encryption is applied after the message passes through the router. On arrival at the destination site, the IP routing data is encrypted, but it is still decrypted before it arrives in the router.

An important difference between this deployment and the previous (point-to-point) one is that the encrypted traffic carries data from a much larger networking population. The larger population modifies the security properties of the solution somewhat. A benefit of the point-to-point case was that a user at one end point knew with high certainty that messages were created by the authorized user of the workstation at the other end point. The benefit of connecting additional hosts may be offset by the reduced certainty of the origination of individual messages. If the network's protections are properly in place we can at least say that messages originate from people with authorized physical access to the network.

This different architecture yields special security considerations at the site and the network level. At the site level we would handle the hosts and in-line encryptors differently than we did in the point-to-point case. Host security is less critical, but the encryptor security becomes more critical as each one carries more traffic than before. At the network level we must confront risks produced by a larger host population and the use of a specific and capable protocol suite. Site and network issues present special challenges to the link encryption security objectives set forth at the beginning of the chapter.

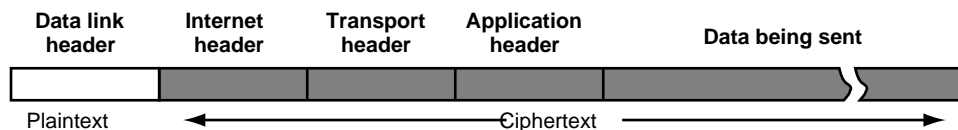


Figure 3-7: BASIC PACKET FORMAT USED BY IN-LINE ENCRYPTORS. Only the data link control information is sent in plaintext. All other header and user data is encrypted.

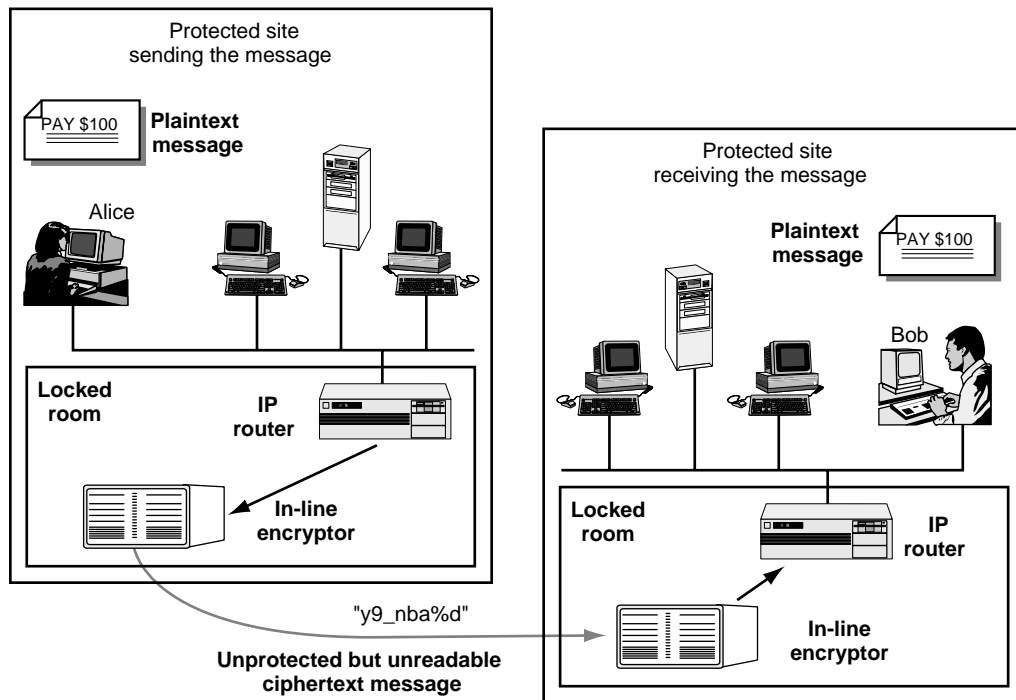


Figure 3-8: HARDWARE ARRANGEMENT FOR IP ROUTER ENCRYPTION. The site's perimeter protects the hosts from unauthorized access. The encryptors, routers, and external connections reside in a locked room to protect them from unnecessary access since they are critical to site security.

3.4.1 Site Protection

As with the point-to-point deployment, site security implements a form of red/black separation with the "red" part inside your protected perimeter and the "black" part along the external communications lines that carry your encrypted traffic. Unlike the point-to-point case, however, here we will treat hosts differently from the in-line encryptors. The hosts still need the degree of protection appropriate to keep outsiders from accessing them. The in-line encryptors, however, now need more protection.

Figure 3-8 illustrates the different levels of protection. The hosts are within the protected boundary of the site. The in-line encryptors are, too, except that they are further protected from unnecessary physical access.

The integrity of the protected network depends on the hosts themselves being protected from unauthorized access. They need to be within the site's physical, protected perimeter. Workstations should not be installed in areas frequented by outsiders or where unauthorized people could use them without appropriate supervision. In most cases it is sufficient to protect them the same as other important organizational papers and assets are protected. Naturally, if hosts on the

network are able to manipulate valuable assets, access control must be consistent with the risk of such assets being misused.

In this deployment the in-line encryptors and the routers connected to them are given extra protection. Ideally both devices are kept in a locked room and provide the only link between outside communications lines and the internal LAN wiring. This reflects the extra responsibility each in-line encryptor now carries. Instead of protecting communications between a pair of hosts, it now protects messages for a large number of hosts. Thus, compromise of the encryptor's keying material or of its physical integrity would risk a larger amount of organizational traffic. Furthermore, there is no benefit to giving workstation users access to the encryptor, its physical connections, or its keys. This prevents accidental or intentional acts that might disclose the keys or cause message traffic to bypass the encryption. The locked room becomes the point in your site at which red/black separation is implemented and enforced.

This arrangement reflects an important distinction that is not present in the point-to-point case. We protect the hosts for the information and control they represent, and not because they hold crypto keys. We protect the in-line encryptors with additional measures because they establish the red/black separation and individual workstation users are no longer responsible for it.

3.4.2 Networkwide Security

What we have constructed here with our in-line encryptors and internet routers is a private internet, with many of the strengths and weaknesses of the public Internet. The principal difference is that we have some control over the population of individuals that can physically access it. Beyond that, however, it is an internet and we face the associated problems in networkwide security.

The first problem this raises is that of accountability. The internet protocol mechanisms that associate messages with individuals are weak and unreliable. Even though a single individual might be associated with a specific host, the host's IP address is not always constant. Some installations assign them temporarily. Even when IP addresses are permanently assigned, a capable person can forge identities at practically every level of the protocol stack. This is made worse by the natural extensibility of the internet architecture. Any site can add a router and extend the network further. Thus we encounter another problem: It is hard to constrain the network's extent, and each new portion has the same capabilities as existing portions. The extreme case of the extensibility problem is that any host on the network is the potential site for a back door that lets the rest of the world into the protected network.

CHASING BACKDOOR CONNECTIONS

The internet protocols are designed to be persistent and inclusive. They will route data to any directly or indirectly connected device the protocols can reach. A router will forward a packet solely on the speculative belief built into its routing algorithm that, even if it doesn't recognize a

packet's destination address itself, another router somewhere will know how to route the packet. Unauthorized connections to other networks can exist, and if they do they can communicate with any destination inside your network.

As we noted earlier, any host on the network can route traffic between networks if it in fact connects two networks. And so, any site on the network can provide a connection to additional sites and networks. This makes the network very easy to extend but very hard to control. This can reduce the certainty you have regarding the security properties of your network.

Each time a new host is connected to the protected network, it is crucial to ensure that it complies with the network's security requirements. It must implement the appropriate set of physical security measures you need to protect it from unauthorized people. You may also need to verify that it does not contain software that could attack other hosts on the network or otherwise subvert the network's security. Perhaps most importantly, the new host must not provide connections to other hosts. Extension of the network should be controlled to ensure compliance with security requirements.

This approach has traditionally been used to construct classified networks used by the U.S. military. No computer equipment is ever allowed to handle classified information, whether networked or not, until it has been *accredited* to do so by a senior officer at the site with final authority for site security. Usually accreditation is granted after a review of the host computer's physical security and the computing system intended to be used for the classified work. There are special standards regarding the amount of physical protection required for information with different degrees of classification, ranging from strong locks and reinforced walls for secret information to complete vault-like enclosures for sensitive intelligence data.

When a network of computers must handle classified information, every component of the network must be equally protected: hosts, wires, hubs, routers, and so on. If data links must leave the protected area, then they generally use in-line encryption devices produced especially by the NSA to protect classified information. Other sites are connected only after accreditors of the affected sites agree that adequate security is provided by both sites and by the encrypted link. Connections to classified networks that link multiple sites are managed by a militarywide agency that establishes security and accreditation requirements.

Few commercial organizations would require or benefit from that level of security management and control. Many do, however, need to isolate their sensitive enterprise network from the outside, so they share a continual dread with the military networks: the unexpected backdoor connection with inadequate security.

This problem was best illustrated in the 1985 movie *WarGames*, in which a high school kid inadvertently found a way to dial in to the computer systems that defended the country against a nuclear missile attack. He found this back door by programming his home computer's modem to dial every telephone number in a given area. Whenever a computer answered the phone, the number would be saved for later investigation. This technique is now known as *war dialing*. Many telephone companies have since implemented techniques to detect war dialing, but the technique still poses a threat.

The other side of the threat is the undesired back door. War dialing would not be so much of a threat if we were certain that dial-in access always went to security-conscious hosts. Unfortunately, the plunging cost of modems has spoiled this assumption. Many workstation vendors routinely include a modem with the computer systems they sell, along with convenient software to use it. Many people find it very tempting simply to connect an extra wire from the back of their computer to their office phone. This produces the worst kind of back door if the modem accepts dial-in connections, but it can also be dangerous when dialing out. We trace that problem back to the protocol stack and the IP layer.

As described in Section 1.3.2, the IP layer of a TCP/IP protocol stack typically does one of two things with packets: it transfers packets between the network and the host's application software or it forwards packets received on one network connection via another. This second process is often called *IP forwarding*. If a host computer contains two or more network connections (for example, a LAN interface and a modem) then it may be possible for the IP software to transfer packets between them. This risky behavior is not what most workstation users intend. Some vendors have recognized this risk and have produced TCP/IP packages that do not support IP forwarding between different interfaces. Others make forwarding a configuration option, allowing individual hosts to enable it if the user desires.

Dial-up IP connections combined with IP forwarding produce a difficult network management problem, and the solutions can be difficult to achieve. The military networks rely heavily on punitive sanctions. It is a federal crime to leak classified information and, by definition, every bit of data on a typical classified computer system is considered classified information. Thus, violations of security measures can lead to a vacation behind bars. Few commercial organizations can produce similar deterrents.

Commercial organizations rely primarily on proactive measures like education and physical protections, and use various detection techniques to locate violators. One large, multinational corporation established a rule that no packets on their corporate network may contain an external network address; any external addresses thus indicate that an external Internet connection has been made. The networking administrators detect "leaks" approximately once a month from all sources. Experts in IP routing also suggest that leaks can be controlled by tuning the routers to reject external packets traveling in the wrong direction with respect to an approved, external connection. See Chapter 8 on firewalls, regarding the management and use of controlled connections to untrusted hosts.

A technique used by many sites is to eliminate vigorously all desktop modems. Many organizations have already converted their internal phone system from traditional, modem-friendly analog lines to digital systems. Connecting a modem to a digital phone is at least ineffective and possibly damaging to the modem. Few individuals will be motivated enough to purchase converters, particularly when the connections are forbidden. Some sites also adopt the attackers' tools, using war dialers to seek dial-in modems within the organizations' incoming telephone lines.

3.4.3 Deployment Security Requirements

The following requirements for deploying IP-routed link encryptors are listed in priority order with the most important requirements given first.

1. **Appropriate link encryptors.** Select link encryptors after reviewing them against the product security requirements specified in Section 3.2.4.
2. **Physical protection of hosts.** Physical security measures must block outsiders from physical access to the hosts that handle sensitive data. Physical access may allow outsiders to type in their own transactions, or worse, install subverted code.
3. **Physical isolation of routers and encryptors.** Physical security measures must prevent physical access to network security devices by everyone except operators and administrators who need access to do their job. Routers as well as in-line encryptors must be treated as network security devices and protected accordingly.
4. **No access paths to untrusted hosts or sites.** This reflects the security objectives listed at the beginning of this chapter. If you need the strong isolation that link encryption can provide, then access to untrusted hosts is a risk factor you should avoid. If untrusted host access is a requirement, skip ahead to Chapter 8 on firewalls.
5. **Active measures against backdoor connections.** The general approach is to characterize potential techniques used by back doors and then try to detect any that might exist. Some sites use “war dialers” and scan the organization’s telephone banks for modem tones where there shouldn’t be any. Another approach is to control the IP addresses used on your network and to enforce address-based access rules on internal routers. Packets with addresses from outside the network can then be detected and blocked.

3.5 Key Recovery and Escrowed Encryption

Key recovery is a mechanism by which the keys that encrypt the data traveling between two users can be recovered by someone else, probably without the others’ awareness. The mechanism is generally intended to allow eavesdropping by a third party. *Escrowed encryption* is the system by which secret keys are stored for the purpose of key recovery. The secret keys are held in escrow until an authorized entity requests access to one. The entity then uses the escrowed key to recover the actual key used to encrypt a particular message.

This mechanism has no practical benefit in typical commercial applications, since data is encrypted only temporarily for the journey between sites. However, the U.S. government has been promoting this technology as a prerequisite for granting export licenses on cryptographic systems. Key recovery is seen as a way by which U.S. law enforcement and intelligence agencies will be able to eavesdrop on encrypted communications when so permitted by U.S. laws.

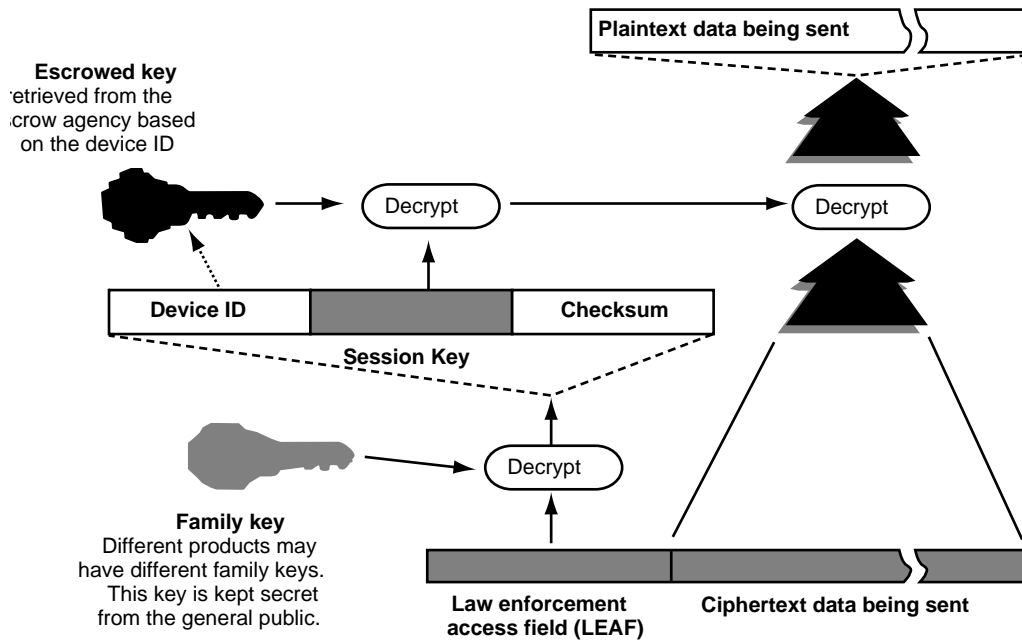


Figure 3-9: KEY RECOVERY WITH THE ESCROWED ENCRYPTION STANDARD. To recover the contents of a message, we fetch the family key for the product and use it to decrypt the LEAF carried with the ciphertext data. We extract the device ID and contact the escrow agents that hold the keys for that device. If the paperwork is in order, the escrow agents give us the escrowed key for that device. We use that key to decrypt the session key stored in the LEAF. The session key will decrypt the ciphertext data.

Without such a facility these agencies would not be able to decipher intercepted messages if the crypto measures are correctly applied.

Several approaches to key recovery have been proposed and implemented. The Escrowed Encryption Standard (EES), established by the U.S. government in 1994, is used in the CLIPPER and CAPSTONE chips. CLIPPER is designed for use in encrypted telephones for the commercial market, and CAPSTONE is designed for messaging applications like e-mail. Both use the SKIPJACK block cipher algorithm. The chips are designed to support the typical range of crypto functions yet also provide a secure but reliable “back door” for decrypting any data they encrypt. The EES defines how this back door is supposed to work.

The key recovery process is illustrated in Figure 3-9. As an example, let us assume that Alice is using CAPSTONE encryption for her e-mail and the FBI needs to wiretap Alice’s communications. They begin by collecting ciphertext that Alice sends to other people. Using the family key built into all devices they decrypt the law enforcement access field (LEAF) from an intercepted message. The LEAF contains a device identifier that identifies the specific CAPSTONE chip that Alice is using to encrypt her messages.

Next, the FBI approaches the escrow agencies that hold Alice's keys. These agencies are organizations that keep copies of the keys belonging to CLIPPER and CAPSTONE chips, indexed by device identifier. The lists of keys are produced when the chips are manufactured and are kept under strict security. The agencies are not to release keys except to authorized officials with appropriate legal documents that authorize the wiretapping. The FBI provides the appropriate documents along with the device identifier for Alice's CAPSTONE chip.

Once the FBI has received the escrowed keys for Alice's device, they can decrypt the session key stored in the LEAF. The session key is the particular key used to encrypt the data in the message that carried the LEAF. Different messages may be encrypted with different secret keys, so each message must carry its own LEAF. Regardless of what key is used for a particular message, a copy of it will always be stored in the LEAF and transmitted with the message. This restriction is enforced by the CLIPPER and CAPSTONE chips. They will not decrypt a ciphertext message correctly unless the LEAF is present. The FBI can then read all of Alice's messages as long as they have the family key and the escrowed key for Alice's particular chip.

Although key recovery serves no practical benefit, the mechanism's presence might be tolerated by users if it brings along no serious drawbacks. Here are some of the problems associated with the key escrow and recovery facilities implemented in the CLIPPER and CAPSTONE chips:

- **Increased product costs**

It is far more expensive to manufacture escrowed crypto devices because of increased security requirements. Conventional crypto devices do not have secret information built into them. Security during manufacturing is solely concerned with guaranteeing correctness of implementation. The escrowed keys in escrowed crypto devices must be generated, installed, and delivered to escrow agencies under extremely tight security. Proposed government standards for manufacturing escrowed devices suggest security measures comparable to those that protect the government's most sensitive intelligence information. Such measures are always very expensive. These costs will be passed on to the users of crypto devices, dramatically increasing product costs.

- **Increased product complexity**

Developers have a fairly broad latitude when incorporating conventional crypto mechanisms into a system. When using escrowed encryption, however, they are restricted to the interface provided by the CLIPPER or CAPSTONE chips. This produces unexpected design constraints, increasing development costs.

Once the chips are integrated into the design, the developers must still take care to handle the LEAF correctly when formatting and transmitting messages. Errors in LEAF handling will render a message unreadable, reducing the product's reliability.

- **Financial support of escrow agencies**

Escrow agencies must comply with stringent and costly operating procedures. They must protect the keys against the most sophisticated attacks, since theft of the escrowed keys

would be an incredible prize. The agencies must also follow strict rules for accepting and releasing escrowed keys in order to fulfill their obligations and maintain their credibility as escrow agents. These activities will be expensive and, in an environment of shrinking government budgets, costs will eventually be passed on to crypto users.

Technological purists might also point out that carrying the session key in every message is an unnecessary risk no matter how thoroughly it is encrypted. In practice, the risk to users of someone cracking session keys in this manner is probably far smaller than other security risks over which users have more control, like the handling of plaintext within their workstations.

Despite the costs, some systems will need to incorporate escrowed encryption and key recovery. In such cases it is important to recognize up front that the system will be more complex and costs will be higher. Many applications might achieve better security by using weaker crypto that doesn't require escrow and key recovery. Resources can then be applied to operating convenience, which sometimes yields the biggest payback in improved security.

3.6 For Further Information

- **Voydock and Kent, “Security Mechanisms in High-Level Network Protocols”**
The classic paper on attacking network messages that are “protected” by encryption.
- **Department of Commerce, Federal Information Processing Standards, “FIPS-140-1: Security Requirements for Cryptographic Modules”**
The standard for cryptographic hardware devices.
- **Department of Commerce, Federal Information Processing Standards, “FIPS-185: Escrowed Encryption Standard”**
The official standard for escrowed encryption as used by CLIPPER, CAPSTONE, and Fortezza. The requirements for escrowed encryption continue to evolve, but this is still the only published standard.