3

The Digital Wallet and the Copyright Box: The Coming Arms Race in Trusted Systems

Tell me, people of Orphalese, what have you in these houses? And what is it you guard with fastened doors? . . . But you, children of space, you restless in rest, you shall not be trapped nor tamed. Your house shall be not an anchor but a mast. . . . For that which is boundless in you abides in the mansion of the sky.

Kahlil Gibran, *The Prophet*

The term *trusted system* came originally from military terminology. It refers to computer systems that provide access to secret information for national and military purposes. In the last few years, its meaning has been broadened to include systems that protect and govern the use of digital objects and information for commercial purposes.

In chapter 2 I considered the development of personal document readers (PDRs), devices that many believe will become the delivery vehicle for digital published works in a multimedia blend. However, releasing valuable works to a digital medium creates a risk for publishers and authors, whose works could be copied and distributed without compensation.

The technological response to this risk is to use trusted systems, which protect digital works using a set of rules describing fees, terms, and conditions of use. These rules, written in a machine-interpretable digital-rights language, are designed to ensure against unsanctioned access and copying and to produce accurate accounting and reporting data for billing. Such trusted systems are the "copyright boxes" in the title of this chapter.

Creative works are not, of course, the only digital objects that need secure storage, accounting, and machine-governed rules of use. The idea of digital cash, or tokens, that can be used as money in cyberspace has caught the public imagination. Handheld trusted systems that permit the exchange of such tokens are the "digital wallets" of the title. Like physical

coins and bills, and unlike checks, digital cash can be exchanged anonymously and, like copyrighted works, according to specific rules. Just as unauthorized copying of a published work amounts to copyright infringement, unauthorized copying of digital tokens amounts to counterfeiting money. Other rules governing cash—for example, those forbidding the anonymous transfer of large sums of money into or out of countries—could cause users of digital cash to run afoul of import regulations and laws about money laundering.

Mondex, one of the companies that offers technology for commerce in digital cash, uses trusted systems based on *smartcards*, plastic cards the size of credit cards with built-in computer chips. These cards have been used in Europe for phonecards and other purposes for several years but, through the late 1990s, had a limited presence in the U.S. market. Mondex's digital wallet is used to read the amount of cash on a card and to transfer money between cards.

In a conversation about digital cash, John Reed of Citicorp quipped to me, "How do you know when Mondex has a bug?" Playing the straight man, I asked: "How do you know when Mondex has a bug?" "When M1 rises," he answered, naming the Federal index that measures the amount of cash in circulation. Such gallows humor about smartcards is not at all far-fetched. It illustrates an underlying fear about digital cash and trusted systems. In essence, the workings of digital technology are largely invisible. We may not realize that a system is broken or compromised until after the damage has been done.

In 1998 a phonecard-piracy scam came to light in Germany, where phonecards designed by Siemens for Deutsche Telekom pay phones are based on smartcards. Ordinarily, once a phonecard's balance reaches zero it is thrown away or given to collectors. A group of "pirates" from the Netherlands found a way to bypass the security of the EEPROM chip used on the cards without leaving physical evidence of tampering and to recharge the cards. They bought thousands of spent cards from collectors, recharged them, and resold them to tobacco shops and retail outlets across Germany. The losses were assessed at about $34 million. This was not the first attack on the cards. The European digital wallet arms race is producing successive generations of cards that are supposed to be more resistant to tampering.

All kinds of programs—not just digital cash and creative works—can benefit from secure storage and guarantees that they work properly and have not been tampered with. We want our computers to have trustworthy programs that are under our control. We want our computers to bring us information about the world but also to be discreet about revealing private information about us to others. As more of our everyday world comes under the control of software and is networked, the issue of computer trustworthiness will extend beyond our desktops and into other parts of our lives. As described in chapter 10, computers may eventually manage, not only our businesses but also our vehicles, homes, and even, through wearable computers, our bodies.

How secure are trusted systems? How secure do they need to be? How do we keep people from circumventing the safeguards of the box, or prevent computer viruses from inflicting malice and mischief on the accounting systems, the protected works, and the system user? How hard is it to break into a trusted system? If the contest between builders of trusted systems and hackers intent on breaking into them is essentially an arms race, is this a race that can be won?

The Coming Arms Race in Trusted Systems

> [This layer is a] complex layout that is interwoven with power and ground [wires] which are in turn connected to logic for the Encryption Key and Security Logic. As a result, any attempt to remove the layer or probe through it will result in the erasure of the security lock and/or the loss of encryption key bits.
>
> Manual for the Dallas Semiconductor DS5002FP, a security microprocessor

> [We] designed and demonstrated an effective practical attack that has already yielded all the secrets of some DS5002FP based systems used for pay-TV access control. . . . the attack requires only a normal personal computer . . . standard components for less than US $100, and a logic analyzer test clip for around US $200.
>
> Ross Anderson and Markus Kuhn, "Tamper Resistance—A Cautionary Note"

Companies developing trusted systems for protecting copyrighted works include International Business Machines, FileOpen, Folio, InterTrust, NetRights, SoftLock, Xerox, and Wave Systems. Initiatives to develop trusted systems are underway by Intel and Microsoft; companies developing

digital wallets include Cybercash, Digicash, and Mondex. Other companies are building systems for using digital cash in on-line shopping.

Designers of trusted systems for military and national security applications assume that the "security threat" will come from a determined, well-funded, malicious, and technically astute adversary. The U.S. Department of Defense *Orange Book* (DOD 1985) discusses the system requirements that must be met by defense contractors building trusted systems for the military. But what are the requirements for trusted systems that handle digital money or copyrighted works? The answer is "it depends."

The Economics of Pirateware

What are the risks that people will develop "pirateware"—hardware and software for circumventing trusted systems? Compared to digital cash and military applications, the threat to trusted systems for digital publishing is sometimes thought to be minimal. But is it? One way to assess the situation is to look at the economic motivations. What are the perceived risks, costs, and benefits (or value) for those who would infringe? What are they for those who would manufacture and sell pirateware? And what are the risks, costs, and value of digital publishing for rights owners? A basic dictum in the design of secure systems says that, to be effective, they must make the costs and risks of pirating much greater than the expected benefits.

As I discuss in chapter 4, the conventional wisdom about paper publishing is that the cost of making a photocopy of a substantial work is high, compared to its purchase price. Publishers believe that what is mainly needed to reduce losses from isolated acts of copying is a way to make it easier for basically honest people to stay honest. For example, if there were a simple and automated way to pay a modest fee to rights owners, such as by inserting a credit card into a copy machine, honest people would pay the royalty without further ado.

This line of thought suggests that the level of security required for protecting copyrighted digital works is similarly quite modest. But this is misleading. The risks and benefits of copying digital works are not really the same as they are for paper works. Without trusted systems, digital technology actually increases the publisher's risk by practically eliminating the infringer's costs of copying and distribution. A digital publisher has no

advantage over an infringer when it comes to manufacturing low-cost copies. With a few keystrokes, any computer user can copy a paragraph, an article, a book, or a lifetime of work and mail it electronically to thousands of people. In the absence of trusted systems, many publishers—fearing that digital distribution really means routine and potentially massive copyright infringement—withhold their valuable works from the Net. Because the losses from infringement of digital works are potentially so great, the benefits of such encroachments are also high. According to our basic dictum about security-system design, if the perceived value of the protected goods is high, then the expected cost of defeating the security system must be made even higher.

The Internet Edge for Trusted Systems

The dictum about designing for security does not assure us that we will actually have trusted systems. This is where resistance at the Internet edge for trusted systems comes into play. It is just as possible that we will not have trusted systems and that valuable digital goods (or substantial digital money) will not become available on the Internet. For any particular technological proposal, the pushback can be that the required security measures—which may include such things as special hardware, special software, or impractical changes to computers already in use—are simply too expensive.

The back-and-forth probing at the edge between the forces for creating trusted systems and the forces holding them back is fueled by the perceived economic value of digital publishing and the perceived expenses of adequate security. Each journey to the Internet edge is an attempt to find a way to serve some of the potential market.

The possible outcomes of the journey include prospects for digital publishing and digital money, prospects for piracy, and, potentially, for changes in the legal status of practices that undermine copyright. Technology consultant Matthew Miller (1997) illustrates this point with a perspective on the evolution of technology for protecting satellite-television transmissions with descrambler boxes. In the late 1980s, anyone who wanted to watch satellite television could set up a large dish antenna, hook up a descrambler box, and pay monthly fees. However, the technology of the descrambler box was so simple that it was widely duplicated and sold in the

underground and hobbyist markets. What were the risks and benefits to the pirate? Because television programming caters to a broad market, and because the same descrambler box would work for anybody with a satellite dish, the black market in descrambling boxes had broad appeal. Furthermore, there were legal ambiguities about whether signals broadcast in the air were in the public domain anyway. When the satellite broadcasters approached Congress asking for legislation to prohibit the manufacture and sale of the rogue descrambler boxes, they got little support. Legislators argued that broadcasters had done too little to protect their signals. Enforcement would be expensive, and legal relief could not adequately compensate for technological weakness. It made no sense to protect satellite transmission, even by a thin legal veil, until its trusted system technology attained a reasonable level of security.

In the years since this example played out, several different approaches to secure commercial trusted systems have been developed. No trusted system is perfectly secure, and some security arrangements are more costly than others. As it was in the satellite example, the legal status of systems for defeating copy protection is murky. Attitudes toward copyright and pirateware are still a matter of debate on the international scene. Even in countries with strict laws for protecting copyright, enforcement is uncertain.

One size does not fit all for trusted systems. The market for trusted systems will probably be stratified, with the least-expensive systems being used for the least-expensive works and the most-expensive trusted systems used for the most-expensive works. This prediction is based on several observations. The first is that technology for high security costs more than technology for low security. The second is that, with low-priced works, a certain amount of leakage may be offset by the broader market served by inexpensive or even free trusted systems. The security levels of such systems are not, however, appropriate for distributing digital objects when there is high incentive to steal them.

Technological Foundations of Trust

In the popular press, the use of encryption technology is often equated with high security in computer systems. This is a misleading association. To use the analogy of a door, consider the flap of a tent, the front door of the aver-

age house, and the combination door on a bank vault. Arguably tents, houses, and banks may all contain things of great value. In a nearsighted theoretical sense, it might seem desirable to put a security door on a tent to protect campers from wild animals. However, a solid metal door such as is used on a bank vault would destroy the portability of a tent without improving its safety, since a persistent adversary could easily come in through the canvas wall. The sobering truth about security is that there are many potential ways to defeat it, especially when it is not a primary design concern from the beginning.

Trust is based on two things: responsibility and integrity. When digital works are stored and used on trusted systems, the systems are responsible for accurately ensuring that they are used in accordance with the rules expressed in the terms and conditions. When digital cash is stored in digital wallets and spent on goods and services, trusted systems are responsible for accurately following fiducial rules for handling cash.

The integrity of trusted systems depends on three technological foundations: physical integrity, communications integrity, and behavioral integrity. Physical integrity refers to the capability of a trusted system to resist physical tampering. Communications integrity refers to its ability to detect any misinformation or lies it receives in its digital communications with other systems. Behavioral integrity refers to the persistent ability of trusted systems to enforce terms and conditions and to resist unauthorized modifications to their programming.

Physical Integrity

The possibility that a pirate will penetrate the hardware and thereby gain access to information stored in a repository is one kind of threat. Sensitive information in trusted systems includes not only protected works but also billing logs, encryption keys, digital cash tokens, and personal and financial information.

Different repositories can have different levels of physical integrity. A repository that can be compromised with a screwdriver would have a low level of physical integrity. A somewhat higher level of physical integrity would be a system with built-in sensors that enable it to detect a threat and to erase sensitive data. A still-higher level of physical integrity would cause

a system to self-destruct when it detects a threat, perhaps setting off alarms and telephoning for help.

Computer peripheral component interface (PCI) cards are devices roughly the size of a small paperback book that are used by plugging them into a computer. Using trusted systems allows PCI cards to hold certain central and sensitive data in financial services—for example, passwords, keys for authorizations, and possibly secret algorithms. To preclude unauthorized electronic probing of information stored on the PCI card, designers cover it in a material that has several layers of nichrome wire. To read the signals in the card's sensitive circuits, an attacker must first penetrate the cover material. Drilling a hole to gain access to the circuits is likely to break one of the wires, which would be sensed by the circuits and trigger a signal to erase sensitive data. This design feature is a first layer of defense against a physical attack.

The arms race in designing secure circuits is like a spy story or a master-level strategy game, with systems within systems and feints within feints. One line of attack takes advantage of the physical properties of computer memories, which are susceptible to low temperatures. If memory circuits are chilled to low enough temperatures, they are unable to change state; they cannot, for example, respond to a signal to erase their contents. Knowing this, attackers can first chill the card, then drill into it and, even, remove its components, confident that the system will be unable to erase its secret information. Before the card warms up, they can disconnect all the defense mechanisms, enabling them to read the data at leisure. A defense against this attack is to put thermal sensors on the card to signal an attack when the temperature drops. There are many other possible measures and countermeasures for designing trusted systems. The interplay between "attacks" and "countermeasures" makes the term *arms race* an appropriate metaphor for the design of trusted systems.

As trusted system defenses are elaborated, handling and shipping them can also become more difficult. Elaine Palmer of IBM tells a story about some secure PCI cards built by IBM for bank computers. A shipment of cards was bound for a bank in Moscow. The bank had already closed when the truck carrying the shipment arrived late in the afternoon. The truck had to be parked outside overnight in the cold Moscow winter. As the temperature fell, the cards' thermal sensors signaled "thermal attack." The operational information was erased before the cards could be installed.

Inexpensive smartcards, generally costing under ten dollars, are being used in an increasing number of systems, ranging from pay television to digital wallets. Because overcoming the tamper resistance of these applications results in substantial returns for the infringer, there have already been several cycles in the trusted system arms race for smartcards.

The typical smartcard has a single plastic-encapsulated chip containing an eight-bit microprocessor with memory and serial input and output. Key data stored in an erasable programmable read-only memory (EPROM), whose contents can be changed by using a twelve-volt signal. As smartcards lack batteries, they cannot use active defenses involving sensors, clocks, and preprogrammed responses to threats.

Anderson and Kuhn (1996) describe a wide range of attacks on the physical integrity of smartcards and other so-called tamper-proof equipment. The early smartcards used for pay-TV systems received their reprogramming signals along a programming voltage contact on the card. Subscribers who initially had their cards enabled for all channels could cover the contact with tape and then cancel their paid subscriptions, leaving the vendor unable to cancel their service.

Physical attacks can be divided into noninvasive and invasive approaches. Noninvasive approaches tend to exploit the responses of smartcards to unusual voltages and temperatures. In some processors, for example, repeatedly raising the supply voltage when a smartcard writes to a security causes the lock to release without erasing the memory it was protecting. Conversely, in another processor, a brief voltage drop sometimes releases the security lock without erasing secret data. "Glitch attacks" use transient signals to interfere with the operation of particular instructions, such as instructions for outputting data or checking passwords.

Current smartcards have almost no defense against direct access to the silicon circuits, that is, against invasive attacks. Some cards use capacitive sensors or optical sensors to detect the continued presence of a covering layer. These sensors tend to be easy to detect and avoid.

Attackers can cut away the plastic with a sharp knife or hand lathe and then remove the chip. Or they can remove the plastic resin over a chip by applying nitric acid alternated with acetone washes. As nitric acid is used to clean chip surfaces during manufacture, it affects neither the silicon (or silicon oxide or nitride) nor any gold used on the chip. The information in the EPROM remains intact and is available for reading. This sort of attack

is sometimes used by so-called class I attackers—amateur pay-TV hackers, students, and others with limited technical resources. More sophisticated attacks can be carried out by pirates with access to focused-ion-beam workstations and infrared lasers.

At present, untested claims for tamper-resistant smartcards and other security processors should be taken with a grain of salt. For many systems the first "hostile reviews" appear only after they are put on the market. In the current escalating and open-ended technological arms race, state-of-the-art engineering practice for trusted systems will change rapidly over time. Prudent designs will employ appropriate techniques to ensure that the cost to the pirate will be much greater than the expected benefit. Systems that can be defeated by a simple attack to one component should be avoided.

Communications Integrity

Not all attacks on trusted systems require physical contact. Communication attacks are attacks made over the wire, when a nontrusted system tricks a trusted system into giving up or compromising digital goods, private information, or digital money. To carry out their functions, trusted systems must communicate with people and other systems. In digital-wallet applications, these communications include the transfer and validation of digital tokens representing money. In digital publishing, they include the transmission of digital works and billing data.

In general, a trusted system "views" the world through its communication channel. To use an analogy, imagine a trader locked in a room who has to obtain information and conduct all his or her business by telephone. Like a trusted system for buying and selling works or for transferring money, the trader can have certain passwords and keys. The world of this telephone trader (or trusted system) is rendered more complex by pirates, who may make fake telephone calls, masquerade as other parties, or listen in to calls.

Trusted systems, like our trader, need reliable ways to certify the bonafides of the party on the other end of a communication line, to verify that the messages exchanged are genuine and unchanged, and to keep the data communicated on the line secret from eavesdroppers. Meeting these goals requires communications integrity. In general, the foundations of communications integrity—secure and robust communications across insecure lines—lie in encryption technology.

When trusted systems connect with each other, they go through a registration process by which they identify themselves to each other and establish their bonafides. Once they are connected, they put each other through a series of tests—a *challenge-response protocol*—intended to weed out impostors and to protect the entrusted works. When registration succeeds, they establish a trusted session using encrypted communications.

Behavioral Integrity

How do we know that trusted systems will operate properly—even if they have not been physically compromised and can communicate securely and prove their identity? For the most part, the behavioral integrity of computers is determined by their programming. There are no fail-safe ways of writing general programs that guarantee correct behavior. There are, however, a number of ways of reducing the risks of error in sensitive operations: for example, by designing systems modularly—so that all critical operations take place in a small part of a program—or by defining system functionality in layers—allowing certain operations to be carried out only under restricted conditions. In addition, designers can build into other parts of the system checks and balances that log sensitive operations and even prevent them from being carried out. Finally, there are ways of writing programs that check other programs before they are run, essentially proving that certain properties are intact under all possible operating conditions.

All of these approaches have blindspots and shortcomings. A rogue programmer can violate modularity rules with hidden code, or violate security layers with trap doors. Checks and balances can be compromised by hiding transactions. Rogue compilers can insert statements into a program that a proof checker (or even a careful programmer) will never see. Furthermore, when the specifications for ensuring a program's correctness become as complex as the program itself, it becomes very difficult to have high confidence in the specifications themselves. In the end, our trust in a program is based on the capabilities, methods, and reputation of the organizations that write and certify programs.

In an ideal computer, the operations carried out by programs could be securely isolated. For example, a screen saver program could not interrupt an electronic-commerce application to steal data or fake an authorized transaction. Similarly, a word-processing program could not, under the

influence of a virus, modify a system file to provide a trapdoor for tampering with financial records. An applet used to drive an animation on a web page could not tunnel into files elsewhere in the computer, remove information, and alter their contents. However, no such protections are built into the operating systems presently in wide use today.

In the absence of trusted operating systems with such secure boundaries between programs, any program can in principle compromise any other program. In this situation, the only guarantee of behavioral integrity is the certification of all programs loaded onto the computer and the warranty that no program can be altered after it has been certified. However, the brute force needed to carry out this guarantee stalls when it confronts the inertia of the vast installed base of uncertified operating systems and popular applications.

Thus, in theory the foundations for trust in a trusted system are its physical integrity, communications integrity, and behavioral integrity. But the practical reality is that existing standard platforms are poorly suited for trusted systems. In the next section, we consider some of the design tensions and methods for building a generic trusted player under these circumstances.

Case Study: Anatomy and Operation of a Trusted Player

Trusted systems can be built into various applications. For example, we could build trusted systems into boomboxes for the metered playing of music. The personal document readers described in chapter 2 could be designed as trusted readers. Trusted printers that receive and spool digital works securely and put watermarks on pages can be built to carry information about the identity and authorized use of the work and about the printing event. Reception units for direct-broadcast satellites (DBS) are also trusted systems.

One of the most ubiquitous kinds of trusted systems is the trusted player, a system for rendering or displaying a digital work. It is typically implemented as a combination of hardware and software on a personal computer. Figure 3.1 illustrates some of the software components of a generic trusted player. In this section, we consider the anatomy and operation of a generic trusted player and describe two transactions—purchasing a digital

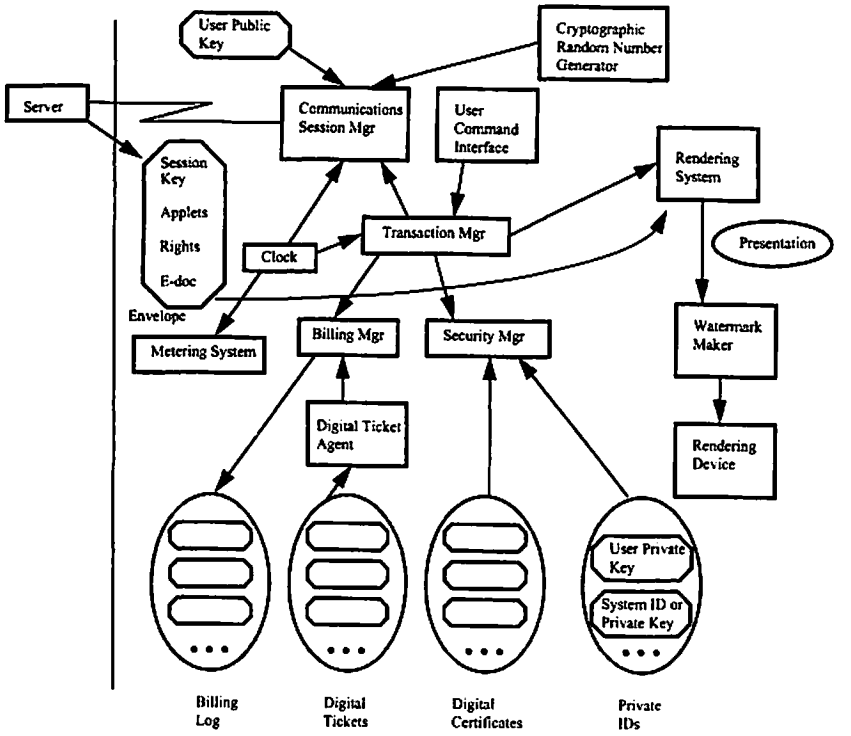

Figure 3.1
A Generic Trusted Player. This system is used to purchase copies of digital works, to store them securely for later use, and to render them on a display.

work and playing it—to illustrate its operation. We then consider some failure modes, possible threats, and means of defense.

No operation on a digital work can be done until the user logs on. The logging-on process is initiated by a command at the user's command interface, which activates the transaction manager and the security manager. First, the user must establish his or her bonafides, typically by supplying a password. (We defer until later a discussion of the security arrangements for invoking or validating the different modules involved in this transaction. However, it is worth noting that this process activates the user's private key for the session, access to which is a crucial security measure.)

If the user wishes to purchase a digital copy of a work, the player initiates a session with a trusted store that has a copy for sale. This process

involves a challenge-response protocol in which the two systems exchange data, test each other's certificates, evaluate each other's security classes, and establish cryptographic keys for the session.

Since the concept of a challenge-response protocol is well known in the art, our description of it will be brief. Each trusted system sends the other a digital certificate, which is digitally signed by a well-known repository; the certificate confirms the system's identity, public key, and security arrangements. Each system then constructs a *nonce*—a random sequence of digits—and sends it, encrypted in the other's public key, to the other party. Each trusted system decrypts the nonce, using its own private key, and sends it back to the other. The systems also synchronize clocks and check their "hot lists" of rogue systems. If the trusted player fails the nonce test, has too much deviance in its clock, or is on the hot list, the trusted store terminates the transaction.

Additional exchanges establish what works are available for sale and the terms and conditions for each transaction. After identifying a desired work, the user requests a copy from the user command interface. The request specifies the terms and conditions of the sale, including the price, any required certificates, and the security class. If the trusted player fails to satisfy any of the requirements for security, available funds, or valid certificates, the transaction is aborted. If the expiration date on the user's right to copy has expired, the transaction is also aborted. If all conditions are satisfied, the trusted systems begin a copy transaction, in which the store repository transmits an encrypted copy of the work. Typically, the work is block-encrypted, and the encryption key is itself encrypted in the user's public key and included with the work. The trusted player stores the work in encrypted form, and both systems make billing records of the transaction. If the transaction is not completed for any reason, both sides report the interruption in their billing records, and the trusted player deletes its encrypted partial copy.

Playing the work requires the user to exercise a play transaction, which is invoked by another command at the user command interface. In some systems, this merely amounts to pressing the "play" button; on others, the user may select from several different play options. Some works offer free copies—but charge for their play options. A play fee may be a flat fee, or it may be metered according to playing time.

When the user action initiates the play transaction, the transaction manager first checks that the requested right to play has not expired and summons the security manager, which checks that all the certificates required in the terms and conditions are available and valid. It then invokes the billing manager and the rendering system. In this example, we assume that the rendering system and the display are integral to the trusted player. If the rendering system is a separate unit, the trusted system begins another transaction backed by secure communication. If digital watermarking information is specified, the information about the user and purchase is encoded in hidden data (the digital watermark) when the work is delivered.

When the playing process is complete, the transaction manager informs the billing manager, which updates the billing log as appropriate.

Boundaries and Threats

We use threat analysis to determine in advance what can go wrong in a system, whether through malicious interference or equipment failure. The likely attacks on an actual implementation—and so the kind of analysis needed—vary according to the particulars of the implementation. In practice, what matters is that the examination of possible threats be systematic, thorough, and as realistic as possible. The analysis described in this section is intended to be educational and to provide a point of reference for our later discussion of security measures.

What's Worth Stealing?

To begin, it is worthwhile to account for the potential values (positive or negative) of some possible attacks. If a digital work is encrypted, there is little risk it will be copied when it is stored or transmitted; but if it can be stored in an unencrypted form without authorization, the risk is that the publisher or rights owner will lose revenue from unauthorized use or copying. The owner can limit this risk in various nontechnological ways—for example, by carrying insurance against leakage of works. System designers can also build into all trusted players measures that test the bonafides of various works, perhaps enabling them to watch for and report the sources of works identified as rogue copies. Such measures would increase the risks to those trying to defeat security measures (Samuelson 1996a).

Another potential risk is the change or misappropriation of the user's private encryption key. Changing the user's private key would be (at the very least) an inconvenience, as it would deprive the user of service until the situation is straightened out. In the meantime, the misappropriator could make fraudulent purchases, damage the user's credit, or violate the user's privacy. If a user's private keys are taken, the attacker can access all the works on the system and, furthermore, acquire additional works until the theft is detected or a credit limit is reached. An attacker who accesses or changes the billing log could delete or add billing data, resulting in inconvenience and perhaps lost revenues for rights owners. One means of reducing this risk is to arrange for transactions involving multiple systems to be reported to separate financial clearinghouses and then reconciled. In such cases, events not reported by one system would probably be reported by another.

Stolen digital certificates would have very little value to the attacker, because they need to be validated when used. Tampering with digital tickets is another matter. Digital tickets are limited-purpose digital tokens comparable to script or coupons for certain rights. They are prepaid and can be used once. Copying digital tickets is, therefore, very much like stealing or counterfeiting money.

To summarize: the security of trusted systems depends on protecting users' private keys and passwords, system private keys, digital tickets, and the billing log. In addition, attacks can do damage even when no data are stolen. The purpose of an attack might be, for example, to destroy a copy of a work, undermine the reputation of a competitor, create an inconvenience for a user, or obtain commercially useful information in violation of a user's privacy rights.

Attacks

We now examine the operation of some of the subsystems of trusted systems to identify potential points of attack. In general, our approach is to consider each system module involved in a transaction to ask what could happen if the module were compromised.

The first operation in our scenario occurs when the user logs in. One means of attack is capturing the user's password. Knowing the password

gives an attacker access to the system and the ability to masquerade as the user, making purchases or using services without authorization. A variation of this attack modifies the user command interface—so as to save the password somewhere in the clear. Another variation compromises the security manager (which hashes the password and compares it against a stored hash value). A third variation changes the stored hashed copy of the user's password to substitute a different password.

The second operation in our scenario involves the copy transaction for purchasing a digital work. One form of attack is to compromise the random-number generator used to create nonces. This could make determining the private system key mathematically easier and would enable the attacker to compromise the communications manager; the latter, in turn, could then invoke the security manager and save a copy of the work in the clear. Another attack would modify the transaction manager so that it aborts the transaction after the work has been received but before the receipt has been confirmed. (This attack is of no use if the protocol is designed to confirm receipt of the complete work before the decryption key is transmitted.) The security manager might then also be impaired and induced to release the system's private key.

The last attack in our scenario would compromise the play transaction for delivering the digital work by modifying the clock. Compromising this component would permit the system to exercise expired rights (for example, a free trial period). This change would prevent the transaction manager from invoking the billing manager or cause the billing manager or metering system to bill inaccurately. The security manager could also be altered to make it omit the checking of certificates or other terms and conditions or to release system keys or keys to the work. The rendering system could be compromised so as to release copies of individual "pages" or "screens" of the work. Or it could be modified to put false watermark data on the presentation or to leave out the watermark entirely.

It is evident from this summary of possible attacks that a trusted system built on a personal computer or workstation has a very long trust boundary. Essentially every module in the system is subject to attack, and an attack compromising any element of the system could cause loss of data, revenue, privacy, or damage to the reputation of a person or organization.

Countermeasures

The difficulty of designing practical trusted systems for digital publishing is inherent in the tension between the need for security and the widespread availability of computing platforms. If trusted systems are widespread but inadequately secure, publishers will not risk releasing their intellectual property on them. If they are demonstrably secure but expensive and rare, publishers will have no incentive to invest in trusted systems, because the market size will be insufficient to earn back the expenses of creating and distributing digital works. Faced with this seeming dilemma, we can come to two conclusions about appropriate courses of action.

The first suggests that designers need to create different classes of trusted systems; works of low value can circulate on low-security systems, and works of high value on systems of substantially greater security. This arrangement would make it crucial to know the difference; that is, to be able to reliably ascertain—by communicating with it—the security level of any trusted system. The second conclusion is that transmission of many works of intermediate value requires personal computers whose security has been augmented by the addition of secure hardware or substantial improvements in the installed base. This approach is more practical than expecting users to buy dedicated trusted systems for accessing secure documents.

Most of the trusted system solutions currently on the market augment the system security of personal computers with software but not hardware. Such systems can defend against casual attacks by uninformed users but not against determined attacks by knowledgeable users with specialized software tools. Nor are they proof against attacks by software viruses unknown to the system. Because this approach fails to provide secure memory, designers have had to limit the functionality of trusted systems in various ways; for example by requiring software-based trusted systems to authorize payments up-front while the system is on-line. As such systems are generally considered inadequately secure to support metered fees, they rely on network-accessed servers to keep track of usage, inventory authorization certificates, and hold prepaid tickets. Finally, these systems tend to be used only for works of relatively low value, because their measures for protecting encryption keys are subject to software attack.

A primary goal of augmenting the hardware of a personal computer is to provide secure storage of valuable data: that is, keys, billing logs, pass-

words, and digital tickets. The basic idea is to limit access to sensitive data to hardware and software correctly carrying out a particular protocol in a particular context.

Even when a personal computer is augmented with such secure hardware, there is still a long trust boundary to defend. An attack on virtually any module in the trusted system can lead to loss of data, privacy, or funds. One response to this risk is to locate all the modules within a secure co-processor, such as a PCI card, that contains memory, a clock, and disk storage. At the current state of the art in personal computers, this approach would require the security system and the user's own computer to have roughly the same speeds and storage capabilities, making the security co-processor too expensive for most applications.

An important, and cheaper, alternative is to store and execute most trusted system modules on the user's computer but to check them for tampering each time they are executed. This operation is roughly the same as that employed by virus-checking software, except that more powerful methods could be used to ensure the trustworthiness of certified software. General virus-checking programs, which scan files for known viruses, know the identity (instruction patterns) of viruses but not of the programs they are defending. Trusted systems could work the other way around. A better approach, known in the art but not widely used, is to digitally sign and hash all software modules of the trusted system. Thus, when a module is written and installed for use in trusted systems, it would be checked by a certifying body and warranted to faithfully carry out its function. A digital hash function (such as MD5 or a related algorithm) would be used to compute a hash value for the binary code of the program included in the signature. The hash value would be signed by the certifying agency and also by the trusted system itself at the time of installation. So, whereas virus-checking programs can protect arbitrary software (but only against known viruses), the signature-and-hash approach would protect only a particular set of trusted software but would defend it against even previously unknown viruses.

A trusted system with this kind of security system would first check the hash value of any module before executing it. One way to do this would be to arrange for the overall execution of the trusted system to be controlled by a small kernel running on a co-processor on a security card. This kernel would launch security modules by copying them from the disk, checking

their hash values, and then starting their execution. In a more powerful approach, the co-processor could exercise considerable control over the execution of the host computer as needed—for example, by running tamper-checking diagnostics on host-system hardware and generally overseeing the execution of all trusted system software.

This operation could be arranged in many variations for defense against different levels of attack. For example, some approaches would suspend operation of the modules in mid-execution to check again whether there has been any tampering during run-time. As always, the goal of such measures would be to raise the bar high enough to discourage determined attackers. Given enough control over the computer and the loading of programs, a trusted system built in this manner would be essentially immune to a software attack, although it would still be subject to sophisticated and more expensive attacks involving combined hardware and software devices.

Before leaving our discussion of augmenting hardware, we should mention two other likely components of a security card for trusted systems: encrypting chips and a clock. The use of specialized encryption chips would allow use of longer encryption keys. Putting clocks on a card remedies a blatant weak spot of most personal computers. The system clock, which on most computers is easy to reset to another date and time, is a feeble barrier to attack. Adding a tamperproof clock to the external card would be a relatively inexpensive defense.

A Cautionary Tale

A fail-safe way to distribute and use digital goods without the bother of special hardware of any kind would be very attractive to publishers and users. In the late 1990s, several companies announced that they had developed such systems, claiming that they would provide protection for owners of intellectual property on the Net.

One such system promoted by Company X was described in a nationally prominent newspaper in 1998. The company's real name is not of interest here, because its technology is very similar to those of other companies, and because the arms race in trusted systems has barely begun. This particular company was founded by people with backgrounds in intelligence work, and their system received favorable comments from several

academic computer scientists. However, as suggested earlier, the harder test for any trusted system is the "hostile review," in which determined specialists try to breach the system's security. Very few systems offered for copyright protection are tested in this way prior to commercial release. Under these circumstances, security failure is quite likely to coincide with financial losses.

In the Company X product, a consumer who, for example, wants to view a movie can pay $2 to make a one-time-use digital file or $20 for the right to unlimited viewing. The consumer receives a digital license agreement, selects the one-time-use option, and pays for the movie with his or her credit card. The consumer's computer stores an encrypted copy of the agreement and then receives an encrypted version of the movie. When the user wants to watch the movie, a special computer program (the "player") matches the encrypted movie with the licenses stored in the computer. If a valid license is found, a key unlocks the movie and allows the user to view it. Once the movie has been played, all that is left is a scrambled file—unless the license is updated and another fee is paid.

Safe as such a system may sound, it is vulnerable to several possible attacks.

Copy Attack

After receiving the movie and the license, but before watching it, the user can foil the system by copying the movie and the license to backup storage. To view it a second time, the user first deletes the license and the reencrypted movie from the computer, then retrieves the unaltered license and movie from backup storage, thus restoring the system to the state it was in before it was played—which permits the user to watch the movie a second time. If the user's system has a tamperproof clock, this attack can be thwarted to some extent by using time stamps that limit the use of the movie or certificates to a given time frame. As most computers do not have tamperproof clocks, the user can first reset the system clock to the time at which he or she purchased the movie.

Fake-Player Attack

A more sophisticated attack can be used by a skillful programmer to decompile the player module and make a new version of it without the security-

enforcing features. This modified player can be posted on bulletin boards around the Net (until authorities find out and object) or sent around more surreptitiously to people on underground Net mailing lists. This attack creates the risk of a catastrophic system failure for the publisher that could affects all the works using this technology.

Virus Attack

This attack is like the fake-player attack, except that the modifications to the player program are caused by a computer virus turned loose in the network. In this case, the people who watch the movie for free are arguably not guilty of willful infringement. They can say they did not realize that their free use of the movies was caused by an undetected virus.

Liberator Attack

This attack is, again, like the fake-player attack, except that the player is modified to make an unencrypted copy of the movie, which is then circulated on standard video-player applications. This attack can be thwarted to some degree by watermarking the movie so that the identity of the original purchaser can be determined from any copy found in circulation. A hacker defense against such tracing is to use a stolen credit card to buy the movie in the first place. In either case, the publisher is unlikely to recover damages.

At the time of this writing, Company X's product is not in widespread use. Conventional wisdom in the security community is that systems like this will be broken when they are widely used—resulting in a very public "hostile review" and failure.

Reflections

Trusted systems are at a nexus of several forces at the Internet edge. The market opportunity for digital publishing is a powerful force for change. But a countervailing pushback for the status quo is the inertia created by the huge installed base of computers and software designed for neither security nor commerce.

Legal and political forces are moving into the fray in several industries. In chapter 4 I consider the evolution of laws related to Internet commerce,

especially with regard to the interplay between copyright and contract law. The evolving political issue of U.S. export policy for cryptographic technology will also influence the development of trusted systems. Because of their original application to military and intelligence communications, cryptographic systems are controlled and classified under import and export regulations as "munitions." Another concern about cryptographic methods is that they could be routinely used with impunity in socially harmful ways—for example, by enabling organized crime to enjoy secure records, secure communications, and invisible money-laundering.

Meanwhile, foreign suppliers of cryptographic technology have begun to use longer key lengths (thereby achieving higher levels of communication security) than their American competitors. The overall effect of this situation on the security of trusted systems is limited, because they are currently constrained more by the lack of certified applications and operating systems than by regulations about the length of encryption keys.

At its core, the drive toward digital commerce and publishing is a shift that lets local businesses increase their global reach by taking a shortcut through cyberspace. However, cyberspace is not simply fast and ubiquitous; it is also largely invisible and intangible. By relying on cyberspace as it exists today, we are moving from local commerce in tangible goods with neighbors that we more or less know and trust to a trade in invisible goods with people we don't know who use computer systems that we need to trust. It is easy to see why this journey to the Internet edge provokes so much pushback and uncertainty.

Political life abounds with issues of boundaries and trust. American currency bears the phrase "In God we trust." Yet President Theodore Roosevelt was widely applauded for his advice to "Walk softly but carry a big stick." When do we give our trust to neighbors, governments, foreigners, computers, even ourselves?

Trust is not simply given; it is built and earned. If we cannot trust computer systems and the invisible and intangible processes that drive them, our only alternative is to increase the visibility and tangibility of those processes. The current chaos at the Internet edge reflects confusion, because we have no assurances of system integrity. To be trustworthy, systems for

Internet commerce and publishing need to visibly demonstrate their integrity and the accountability of the entities they represent.

Computer systems have the potential to provide wonderful visibility and accountability. Just as bank systems provide an audit trail in monthly statements, so trusted systems could provide accounting summaries. Just as political institutions back up the occasional bank failure and respond to claims of errors, so new institutions could stand behind computer systems for commerce. Creating institutions that can certify trusted systems is part of the overall challenge of enabling the information marketplace to grow. Viewing trusted systems in this light makes it clear that there is important social and legal work to be done at the Internet edge.

4

The Bit and the Pendulum: Balancing the Interests of Stakeholders in Digital Publishing

Information doesn't want to be free.
It wants to be paid for.

Member of the audience, Computers, Privacy, and Freedom Conference, March 1997

The drive toward digital publishing reflects our need to be heard. It speaks powerfully to the dream that everyone ought to have instant access to the best ideas, the most creative works, and the most useful information. On a global network publishers can distribute digital works nearly instantaneously at low production costs, giving consumers the convenience of twenty-four-hour automated shopping.

Technology does not, however, exist in a vacuum. Even if all the technological obstacles to trusted systems described in chapter 3 were removed, serious social and legal issues related to digital publishing would remain. At present, then, the potential for digital publishing remains just that—a potential. The market remains nascent because the medium has failed, so far, to balance the interests of important stakeholders. In this chapter, therefore, we consider the dream of digital publishing and the co-evolution of technological, business, and legal innovations needed to balance those interests.

The Pendulum Swings

Computers and the digital medium itself are sometimes seen as the major barriers to digital publishing. When personal computers and desktop publishing first appeared in the early 1980s, many publishers saw digital publishing as too risky. At the time, numerous factors, such as the lack of an installed base of computers and the high costs of production, reinforced