CHAPTER

20

Copyright and Privacy Protection

The DeCSS case is almost certainly a harbinger of what I would consider to be the defining battle of censorship in cyberspace. In my opinion, this will not be fought over pornography, neo-Nazism, bomb design, blasphemy, or political dissent. Instead, the Armageddon of digital control, the real death match between the Party of the Past and the Party of the Future, will be fought over copyright.
—JOHN PERRY BARLOW

Be very glad that your PC is insecure—it means that after you buy it, you can break into it and install whatever software you want. What YOU want, not what Sony or Warner or AOL wants.
—JOHN GILMORE

20.1 Introduction

There are a number of reasons to consider technical mechanisms that support copyright and privacy in a single book chapter.

At the political level, there is the conflict alluded to by Barlow in the above quotation. The control of information has been near the center of government concerns since before William Tyndale (one of the founders of the Cambridge University Press) was burned at the stake for printing the Bible in English. The sensitivity continued through the eighteenth-century battles over press censorship, to the more recent doctrine that warfare is about controlling the information space of one's own nation and its competitors. In the last few generations, the great wealth accruing to the owners of literary, film, and music copyright has created another powerful interest in control.

At a system level, both copyright and censorship are access control issues, concerned with limiting access to some information to people in a particular group. In the former case, the group consists of people who have paid for the bits in question; in the latter, they meet some other criterion (such as being over 18, or nonresidents of Singapore, or whatever). Sometimes, they overlap, as with the common convention of restricting online pornography to users who can use a credit card with an "age check" service. (The assumption that all credit card holders are 18 or over may not hold up forever, giving rise to an interesting security failure of the "changing environment" type.) In general, users' real names matter: if identity is no longer sacrosanct, liability for sedition, copyright infringement, and defamation become shaky.

Privacy is also largely an access control issue. It's about being able to limit the number of people who can see who you're exchanging email with, what you're reading and what music you're listening to. In theory, there is no compelling reason why they should be in conflict, and in the pre-electronic world, they usually weren't Copyright was protected by the cost of small-scale duplication; it was simpler and cheaper to buy a book or a record than to make a single copy, and people who made large numbers of copies could usually be tracked down and prosecuted. The cost barrier to copying was eroded significantly by the photocopier and the cassette recorder, but they didn't change the basic economics. So books, records, and videos can be bought for cash and traded secondhand. But the move to a digital world is changing this. Although there are some systems, such as pay-TV, which depend on a physically tamper-resistant device, most copyright control is moving in the direction of registration. Once you have bought a software product, you're supposed to register as a user, and this business model is spreading to other media—which in turn is undermining privacy.

In this chapter, I'm going to use a technical view of privacy. Confidentiality means keeping information secret because of an obligation owed to a third party, while privacy refers to the ability to control the dissemination of information about oneself. In the privacy applications I've discussed up till now, these tend to overlap. For example, my medical privacy is implemented by imposing on my doctor a duty of confidentiality. But in this chapter I'm interested in the mechanisms I can use to protect my own privacy directly, starting from encrypted electronic mail and going up through online pseudonyms and networks of anonymous remailers to file systems whose owners can plausibly deny knowledge of their contents.

At this technical level, the tension between copyright and privacy becomes acute. Videos and music tracks that are not protected by physically tamper-resistant tokens can in principle be copied and shared; they can end up being traded informally, on a large scale, and without any payment to the copyright owner; and whatever the pressure brought on ISPs to curtail traffic in things like MP3 audio files, the existence of traceless communication systems might ultimately make enforcement efforts futile. On the other hand, a number of existing and proposed electronic distribution systems make encrypted content freely available: to decrypt it, the user must contact a server and buy a key—which usually means providing your name and address. This means that there's enormous amounts of "information exhaust," as one vendor puts it: a central license server knows exactly who bought access to what, and when. Marketers think this is magnificent; privacy advocates are appalled [260].

In addition, a number of the emerging technologies cut both ways. Data hiding techniques can be used to embed copyright marks invisibly in digital video; they can also be used for *steganography*, that is for hiding messages in other messages. The family snapshots that you email to your brother might actually contain a ripped-off track from your favorite band's latest CD. (They could also contain a message organizing demonstrators to picket an international trade conference, so the government interest is never far away.)

20.2 Copyright

The protection of copyright is now an obsession of the film, music, and book publishing industries (often referred to collectively—and perjoratively—by computer industry people as "Hollywood"). But this didn't start with the Internet. There were long and acrimonious disputes arose in many countries over whether blank audio or videocassettes should be subjected to a tax whose proceeds would be distributed to copyright owners. And the issue isn't confined to electronic media; in Britain, several million pounds a year are distributed to authors whose books are borrowed from public lending libraries [629]. Going back to the nineteenth century, there was alarm that the invention of photography would destroy the book publishing trade; and in the sixteenth, the invention of movable type printing was considered to be highly subversive by most of the powers that were, including princes, bishops, and craft guilds.

There is now a lot of work being done on *electronic copyright management systems* (ECMS), of which the most significant fielded example to date is pay-TV. We've already looked at the tamper-resistance aspects of pay-TV systems, and some of the protocol failures. I noted that such systems are highly challenging because the attackers can buy as many access tokens as they like for dismantling and study. But before we worry about high-tech systems, let's look at software protection, as most of the current copyright issues have been played out in the PC and games software markets over the last twenty years or so.

20.2.1 Software

Software for early computers was given away free by the hardware vendors or by users who'd written it. IBM even set up a scheme in the 1960s whereby its users could share programs they had written. (Most of these were useless, as they were too specialized, too poorly documented, or otherwise too hard to adapt.) So protecting software copyright was not an issue. Almost all organizations that owned computers were large and respectable; the software tended to require skilled maintenance; and so they often had full-time system engineers employed by the hardware vendor on-site. There are still sectors which operate on this business model. For example, one supplier of software for bank dealing rooms takes the view that anyone who pirates its code is welcome, as using it without skilled technical support would be a fast way for a bank to lose millions.

But when minicomputers arrived in the 1960s, software costs started to become significant. Hardware vendors began to charge extra for their operating system, and third-party system houses sprang up. To begin with, these mostly sold complete bespoke systems—hardware, software, and maintenance—so piracy was still not much of an issue. By the mid 1970s, some of them had turned bespoke systems into packages:

software originally written for one bakery would be parameterized and sold to many bakeries. The main type of copyright dispute in those days was when a programmer left your company to join a competitor, and their code suddenly acquired a number of your features. The question then was whether he'd taken code with him, or reimplemented it. The standard way to resolve such a problem is to look at *software birthmarks*, features of how a particular implementation was done, such as the order in which registers are pushed and popped. This continues to be an issue, and there are various code comparison tools available—many of them developed in universities to detect students cheating on programming assignments. (This thread of research leads to general-purpose plagiarism-detection tools, which can trawl through natural language, as well as code, and typically recognize a passage of text by indexing it according to the least-common words that appear in it [376]; on to systems used by humanities scholars to figure out whether Bacon wrote Shakespeare, and back to tools that try to identify the authors of viruses from their coding style [476].)

With time, people invented more and more things to do with software. So a firm that had bought a minicomputer for stock control (or contracted for time on a bureau service) might be tempted to run a statistical program too to prepare management reports. Meanwhile, the installed base of machines grew large enough for software sharing to happen more than just occasionally. In response, some system houses started to put in copyright enforcement mechanisms. A common one was to check the processor serial number; another was the *time bomb*. In 1981, when I worked for a company selling retail stock control systems, we caused a message to come up every few months saying something like "Fault no. WXYZ—please call technical support." WXYZ was an encrypted version of the licensed customer's serial number, and if the caller claimed to be from that customer we'd give them a password to reenable the system for the next few months. (If not, we'd send round a salesman.) This mechanism could have been defeated easily if the "customer" understood it, but in practice it worked fine—most of the time it was a low-level clerk who encountered the fault message and called our hotline.

Software piracy really started to become an issue when the arrival of microcomputers in the late 1970s and early 1980s created a mass market, and software houses started to produce products that didn't require technical support to install and run. Initial responses varied. In a famous open letter from Bill Gates in 1976, a year after Microsoft was founded, he complained that less than 10% of all microcomputer users had paid them for BASIC [319]. "Who cares if the people who worked on it get paid?" he asked. "Is this fair?" His letter concluded: "Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software."

Appeals to people's sense of fair play only got so far, and the industry next tackled the obvious difference between minis and micros—the latter had no processor serial numbers. Three general approaches were tried: to add uniqueness on to the machine, to create uniqueness in it, or to use whatever uniqueness happened to exist already by chance.

- The standard way to add hardware uniqueness was a *dongle*—a device, typically attached to the PC's parallel port, which could be interrogated by the software. The simplest just had a serial number; the most common executed a simple challenge-response protocol; while some top-end devices actually performed some critical part of the computation.

- A cheaper and very common strategy was for the software to install itself on the PC's hard disk in a way that was resistant to naive copying. For example, a sector of the hard disk would be marked as bad, and a critical part of the code or data written there. Now if the product were copied from the hard disk using the utilities provided by the operating system for the purpose, the data hidden in the bad sector wouldn't be copied and so the copy wouldn't work. A variant on the same theme was to require the presence of a master diskette which had been customized in some way, such as by formatting it oddly or even burning holes in it with a laser. In general, though, a distinction should be drawn between protecting the copy and protecting the master. It's often a requirement that people should be able to make copies for backup if they wish, but not to make copies of the copies (this is called *copy generation control*).
- A product I worked on stored the PC's configuration—which cards were present, how much memory, what type of printer—and if this changed too radically, it would ask the user to phone the helpline. It's actually quite surprising how many unique identifiers there are in the average PC; ethernet addresses and serial numbers of disk controllers are only the more obvious ones. Provided you have some means of dealing with upgrades, you can use component details to tie software to a given machine.

A generic attack that works against most of these defenses (or at least those that don't hide critical code somewhere uncopiable) is to go through the software with a debugger and remove all the calls made to the copy protection routines. Many hobbyists did this for sport, and competed to put unprotected versions of software products online as soon as possible after their launch. Even people with licensed copies of the software often got hold of unprotected versions as they were easier to back up and often more reliable generally.

The vendors also used psychological techniques.

- The installation routine for many business programs would embed the registered user's name and company on the screen, for example in the toolbar. This wouldn't stop a pirate distributing copies registered in a false name, but it could discourage legitimate users from giving casual copies to colleagues.
- Industry publicists retailed stories of organizations that had come unstuck when they failed to get a critical upgrade of software they hadn't paid for. One of the popular stories was of the U.S. army bases in Germany that didn't pay for the VAX VMS operating system, then got hacked after they didn't get a security patch.
- If early Microsoft software (Multiplan, Word, or Chart) thought you were running it under a debugger, trying to trace through it, it would put up the message, "The tree of evil bears bitter fruit. Now trashing program disk." It would then seek to track zero on the floppy and go "rrnt, rrnt, rrnt."

In the mid- to late-1980s, the market split. The games market moved in the direction of hardware protection, and ended up dominated by games console products with closed architectures, where the software is sold in proprietary cartridges. Business software vendors, however, generally stopped trying to protect mass-market products using predominantly technical means. There were several reasons.

- Unless you're prepared to spend money on seriously tamper-resistant dongle hardware that executes some of your critical code, the mechanisms will be defeated by people for whom it's an intellectual challenge, and unprotected code will be anonymously published. Code that isn't protected in the first place is less of a challenge.

- As processors got faster and code more complex, operating system interfaces became higher level, and software protection routines of the "bad disk sector" variety got harder to write. Now that it's possible to run a Windows NT system on top of Linux using vmware, application software can be completely shielded from machine specifics such as Ethernet addresses. The net effect is an increase in the cost and complexity of both protection and piracy.
- Protection is a nuisance. Multiple dongles get in the way of, or even interfere with, each other. Software protection techniques tend to make a product less robust and cause problems—as when your hard disk fails and you recover from backup to a new disk. Protection mechanisms can also cause software from different vendors to be unnecessarily incompatible, and in some cases unable to reside on the same machine.
- Technical support became more and more important as software products became more complex, and you only get it if you pay for the software.
- The arrival of computer viruses was great for the industry. It forced corporate customers to invest in software hygiene, which in turn meant that casual copying couldn't be condoned so easily. Within a few years, antivirus programs made life much harder for copy protection designers in any case, as nonstandard operating system usage tended to set off virus alarms.
- There was not much money to be made out of harrassing personal users as they often made only casual use of the product and would throw it away rather than pay.
- A certain level of piracy was good for business. People who got a pirate copy of a tool and liked it would often buy a regular copy, or persuade their employer to buy one.
- In Microsoft's case, customer reaction to its scare message was pretty negative.
- Many vendors preferred not to have to tackle issues such as whether the software was licensed to the user (in which case he could migrate it to a new machine) or to the machine (in which case he could sell the computer secondhand with the software installed). As both practices were common, mechanisms that made one or the other very much harder caused problems. The mechanisms that could easily deal with both (such as dongles) tended to be expensive.
- Finally, Borland shook up the industry with its launch of Turbo Pascal. Before then a typical language compiler cost about $500 and came with such poor documentation that you had to spend a further $50 on a book to tell you how to use it. Borland's product cost $49.95, was technically superior to the competition, and came with a manual that was just as good as a third party product. (So, like many other people, once I'd heard of it, pirated a copy from a friend, tried it and liked it, I went out and bought it.) 'Pile it high and sell it cheap' simply proved to be a more profitable business model—even for speciality products such as compilers.

The industry then swung to legal solutions. The main initiative was to establish antipiracy trade organizations in most countries (in the United States, the Software Publishers' Association), which brought a number of high-profile prosecutions against large companies that had been condoning widespread use of pirate PC software. This was followed by harrassing medium and even small businesses with threatening letters demanding details of the company's policy on enforcing copyright—holding out a carrot of approved software audit schemes and a stick of possible raids by enforcement squads. All sorts of tricks were used to get pirates to incriminate themselves. A typical ruse was the *salted list*. For example, one trade directory product I worked on contained details of a number of bogus companies, with phone numbers directed to the publisher's help desk whose staff would ask for the caller's company and check it off against the list of paid subscribers.

Eventually, the industry discovered that the law not only provides tools for enforcement, but sets limits too. The time-honored technique of using time bombs has now been found to be illegal in a number of jurisdictions. In 1993, for example, a software company director in Scunthorpe, England, received a criminal conviction under Britain's Computer Misuse Act for "making an unauthorized modification" to a system after he used a time bomb to enforce payment of an disputed invoice [194]. Many jurisdictions now consider time bombs unacceptable unless the customer is adequately notified of their existence at the time of purchase.

The emphasis is now swinging somewhat back in the direction of technical mechanisms. Site license agreements are enforced using *license servers*, which are somewhat like dongles but are implemented on PCs that sit on a corporate network and limit the number of copies of an application that can run simultaneously. These servers can still be defeated by disassembling the application code, but as code becomes larger this gets harder; combined with the threat of legal action, they are often adequate. Other mechanisms include issuing such frequent updates to software that life becomes tiresome outside the official distribution chain; and (a cynic might say) making the operating system so unreliable that every few months it will crash completely, forcing all software to be reloaded from the distribution media.

The model to which the software industry is converging is thus one that combines technical and legal measures, understanding the limits of both, and accepting that a certain amount of copying will take place (with which you try to leverage fully-paid sales). One of the more revealing dicta of Billionaire Bill is:

Although about three million computers get sold every year in China, people don't pay for the software. Someday they will, though. And as long as they're going to steal it, we want them to steal ours. They'll get sort of addicted, and then we'll somehow figure out how to collect sometime in the next decade [332].

The latest developments have to do with online registration. If you design your product so that customers interact with your Web site—for example, to download the latest exchange rates, virus signatures or security patches—then you can keep a log of everyone who uses your software. But this can be dangerous. When Microsoft tried it with Registration Wizard in Windows 95, it caused a storm of protest. Also, a colleague found that he couldn't upgrade Windows 98 on a machine on his yacht since it was always offline. But the wind appears to be blowing in this direction.

It's also worth noting that different methods are used to counter different threats. Large-scale commercial counterfeiting may be detected by monitoring product serial numbers registered online; but such operations are found and closed down by using investigative agencies to trace their product back through the supply chains, and people are deterred from getting into the business in the first place using a combination of the seals and other secure packaging techniques discussed in Chapter 12.

That is more or less what's being done in the personal and small business sectors, but with medium and large businesses, the main risk is that fewer legal copies will be purchased than there are machines that run them. The usual countermeasure is to combine legal pressure from software trade associations with site licenses and rewards for whistleblowers. It's significant that companies such as Microsoft make the vast bulk of their sales from business rather than personal customers. This is perhaps the main reason that the industry holds back from using online registration to enforce copyright aggressively against personal users. The potential extra revenues are small given the possible costs of a public backlash. Other considerations are privacy laws (especially in Europe), and the difficulty of tracing people who change addresses or trade PCs secondhand.

To sum up: none of the low-cost protection technologies available at the beginning of the twenty-first century is foolproof, especially against a determined opponent. But by using the right combination of them, a large software vendor can usually get a tolerable result—especially if prices are not too extortionate and the vendor isn't too unpopular. Small software companies are under less pressure, as their products tend to be more specialized, and the risk of copying is lower, so they can often get away with making little or no effort to control copying.

There are also many alternative business models. One is to give away a limited version of the product, and sell online a password that unlocks its full functionality. Unix was popularized by giving it away free to universities, while companies had to pay. A variant on this theme is to give basic software away free to individuals but to charge companies, as Netscape did. An even more radical model is to give software away completely free, and make money from selling services ranging from consultancy and support to advertising on a Web site—as the Linux industry is now doing.

This experience has led many computer people to believe that ultimately the solution for "Hollywood's" problem lies in a change of business model. But before we dive into the world of protecting multimedia content, let's look briefly at a few historical precedents.

20.2.2 Books

Shapiro and Varian present a useful historical lesson in the rise of book publishing [696]. In 1800, there were only 80,000 frequent readers in England; until then, most books were serious philosophical or theological tomes. After the invention of the novel, a mass market emerged for books, and circulating libraries sprung up to service it. The educated classes were appalled, and the printers were frightened that the libraries would deprive them of sales.

But the libraries so whetted people's appetite for books that the number of readers grew to 5,000,000 by 1850. Sales of books soared as people bought books they'd first borrowed from a library. The library movement turned out to have been the printers' greatest ally, and helped create a whole new market for mass-market books.

20.2.3 Audio

Pirates have also been copying music and other audio much longer than software. Paganini was so worried that people would copy his violin concertos that he distributed the scores himself to the orchestra just before rehearsals and performances, and collected them again afterward. (As a result, many of his works were lost to posterity.)

In recent years, there have been one or two flurries of industry concern. When the cassette recorder came along in the 1960s, the record industry lobbied for (and in some countries got) a tax on audiocassettes, to be distributed to copyright holders. Technical measures were also tried. The Beatles' record "Sergeant Pepper" contained a 20 KHz spoiler tone, which should in theory have combined with the 21 KHz bias frequency of the tape to produce a 1 KHz whistle that would spoil the sound. In practice it didn't work, as many record players didn't pick up the spoiler tone. But in practice this didn't matter. Cassettes turned out not to be a huge problem because the degradation in quality is noticeable on home equipment; many people just used them to record music to listen to in their cars. Then, in the 1980s, the arrival of the Sony Walkman made cassettes big business; and although there was some copying, there were also huge sales of prerecorded cassettes, and the music industry cleaned up.

The introduction of *digital audio tape* (DAT) caused the next worry, because a perfect copy of the contents of a CD could be made. The eventual response was to introduce a *serial copy management system* (SCMS)—a single bit in the tape header that would indicate whether a track could be copied or not [410]. The idea was that copies made from a CD would be marked so that they could not be copied again; in this way, people could make copies of CDs they already owned, to listen to on the move, but couldn't make copies of the copies. This didn't work well, as the no-more-copies bit is ignored by many recorders and can be defeated by simple filtering. Again, this didn't matter as DAT didn't become widely used. (CD-ROMs also have a no-copy bit in the track header but this is almost universally ignored.)

Audio copying has recently become a headline concern again, thanks to the popularity of the MP3 format for compressing audio. Previously, digital audio was protected by its size—a CD full of uncompressed music can take 650 Mb. However, MP3 enables people to take an audio CD track of tens of megabytes and squeeze it into a few hundred kilobytes, making it practical to download over a dial-up modem line. Usage in universities is particularly heavy; in 1998, some 40% of the network traffic at MIT was MP3 traffic. Some students have become underground disc jockeys and relay audio streams around campus—without paying royalties to the copyright owners.

The initial response of the industry was to look for technical fixes. Alternative audio compression technologies were developed that did contain copyright protection mechanisms (for example, [483]), but failed to take off. Hollywood is still trying to pressure the computer industry into making platforms on which music copying is hard, but this is not happening.

- First, the PC is an open platform and it's intrinsically easy to copy bit streams inside it. There have been proposals to close the platform, such as by incorporating bus encryption (which I discussed in Section 14.5.2) into cache controller chips, or even the main Intel processor line. But the first step in this direction—a processor serial number in the Pentium III—met with huge public resistance. Attempts to keep DVD proprietary meant preventing Linux PCs from using DVDs, and have led to a fight I'll discuss later. So far all we've seen are hacks, such as encrypting the music stream all the way to the sound card driver software. The response to this is a modified sound card that grabs the deciphered data.

- Second, the success of new hardware depends on the availability of software, and vice versa. To launch a new platform for audio, it had better be backward-compatible with existing CDs and players. One trick that has been proposed is to encrypt only the least significant bits of the music track, and decode them in the sound card driver in the PC operating system. This way, people with existing CD players can get music, and people with authorized copy protected equipment can get higher quality music. However, the quality is most important with classical tracks, which are not economically important, and in any case the complete signal can be extracted using modified sound cards. Finally, excluding Linux users from next-generation audio will probably lead to the same kind of fight as over DVD.
- In any case, many CDs have been sold that contain easily reproducible, perfect-quality digital copies—in effect, billions of golden master disks that are completely outside the industry's control.

The next step Hollywood took was to sue, with the main targets being Web sites that allowed MP3s to be shared. Commercial MP3 sites are being bulldozed into setting up subscription channels and generally making their peace with the music industry. But they have been replaced in the firing line by systems such as Napster and Freenet, which enable users who wish to swap tracks to get in touch with each other directly. I'll come back to the potential of these systems when I discuss privacy mechanisms later in the chapter. Meanwhile, I can see no compelling reason why audio protection should develop all that differently from software protection: technical solutions that people found ways to defeat, followed by a legal onslaught that ran out of steam eventually, finally settling down to a mix of technical and legal controls that limit piracy even if it can't be eliminated entirely.

This is not to say that I expect a one-size-fits-all copyright control package to be developed. Just as Microsoft has different needs from a small specialist firm, and uses different methods, so also one can expect rather different controls on the current Top of the Pops than there will be on a specialist cult item such as a CD by the Bonzo Dog Doo-Dah Band.

- In the former case, the density of listeners is such that a track can spread widely by personal copying, but there is a very short shelf-life. Speed and fashion will be everything. Indeed, sales of fashion merchandise may become much more important than CD sales; as with the Netscape/Linux business model, it may make sense to give the "product" away free and make money on the "maintenance" (tours, T-shirts, fan club . . .).
- In the latter case, the appeal is timeless but to a scattered minority of enthusiasts, who copy tracks because they feel exploited by having to pay $17.95 for a CD that they already have on vinyl. Because of this, it might not even be possible to sue "violators" in many jurisdictions. So the trick may be slightly keener pricing and/or packaging that appeals to the collector.

I also expect that Hollywood will follow the software industry and adopt a somewhat more mature attitude to copying. After all, 70% of a market worth \$100 billion is better than 98% of a market worth \$50 billion. And just as a certain amount of copying helped market software, it can help music sales too: the Grateful Dead encouraged bootleg taping because they had learned it didn't harm their sales.

20.2.4 Video and Pay-TV

The early history of videocassettes is very similar to that of audio cassettes. At first Hollywood was terrified, and refused to release movies for home viewing. Again, there were technical measures taken to prevent copying—such as the Macrovision system that adds spurious synchronization pulses to confuse the recording circuitry of domestic VCRs—but again these turned out to be straightforward for technically savvy users to defeat. Then Hollywood became paranoid about video rental stores, just as book publishers had been about libraries: but Video rentals greatly increased the number of VCRs sold, and whetted people's desire to own their favorite movies. VCRs and videocassettes became mass-market products rather than rock stars' toys, and now sales of prerecorded cassettes make up most of the income of firms such as Disney. The business model has changed so that the cinema release is really just advertising for the sales of the video.

And now that many of the world's pre-teens demand that their parents build them a collection of Disney cassettes, just like their friends have, a videocassette pirate must make the packaging look original. This reduces the problem to an industrial counterfeiting one. As with mass-market software before the onset of online registration, or with perfumes and Swiss watches today, enforcement involves sending out field agents to buy cassettes, look for forgeries, trace the supply chain and prosecute the bad guys.

Much more interesting technical protection mechanisms have been built into the last few generations of pay-TV equipment.

The advent of pay-TV, whether delivered by cable or satellite, created a need for *conditional access* mechanisms, to allow station operators to restrict reception of a channel in various ways. If they bought only the rights to screen a movie in Poland then they'd have to block German or Russian viewers within the satellite footprint from watching. Porn channel operators needed to prevent reception in countries like Britain and Ireland with savage censorship laws. Most operators wanted to be able to charge extra for specific events such as boxing matches.

20.2.4.1 Typical System Architecture

A number of systems were developed, and their evolution was determined largely by the hardware cost of deciphering video (for a history of set-top boxes, see [186]). The first-generation systems, available since the 1970s, were crude analog devices which used tricks such as inverting the video signal from time to time, interfering with the synchronization, and inserting spikes to confuse the TV's automatic gain control. They were easy enough to implement, but also easy to defeat; breaking them didn't involve cryptanalysis, just an oscilloscope and some patience.

The second generation of systems appeared in the late 1980s and employed a hybrid of analogue and digital technologies—the broadcast was analogue, the subscriber control was digital. These included systems such as Videocrypt, Eurocrypt, and Nagravision. A typical such system has three components:

- There is a subscription management service at the station, which enciphers the outgoing video, embeds various *entitlement control messages* (ECMs) in it, and issues access tokens such as smartcards to subscribers.
- There is a *set-top box* which converts the cable or satellite signal into one the TV can deal with. This includes descrambling it.
- Finally there is the subscriber smartcard, which personalizes the device and controls which programs the set-top box is allowed to descramble. It does this by interpreting the ECMs and by providing keys to the descrambling circuit in the set-top box.

This arrangement means that the complex, expensive processes such as bulk video scrambling can be done in a mass-produced standard device with a long product life, while security-critical functions—which may need to be replaced in a hurry after a hack—can be sold to the customer in a low-cost token that can easily be replaced. If the set-top box itself had to replaced every time the system was hacked, the economics would be much less attractive.

The set-top box decodes the ECMs from the input data stream, and passes them to the card; the card processes the ECMs to get both control messages (such as "smartcard number 123356: your subscriber hasn't paid, stop working until further notice"); and keys, known as *control words*, that are passed to the set-top box. The set-top box uses the control words to descramble the video and audio streams.

20.2.4.2 Video Scrambling Techniques

The most common video scrambling technique was *cut-and-rotate*. This scrambles one line of video at a time by cutting it at a point determined by a control byte and swapping the left and right halves (see Figure 20.1). This involved analogue-to-digital conversion of the video signal, storage in a buffer, and digital-to-analogue conversion after rotation, a process that could just about be shoehorned into a low-cost custom VLSI chip using the technology of the mid-1980s.

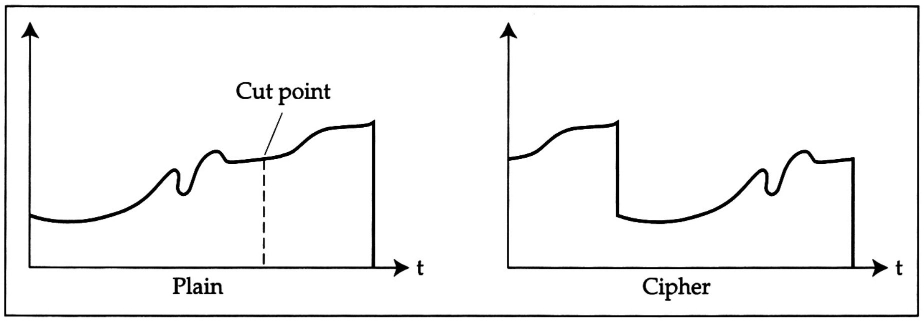


Figure 20.1 Cut-and-rotate scrambling.

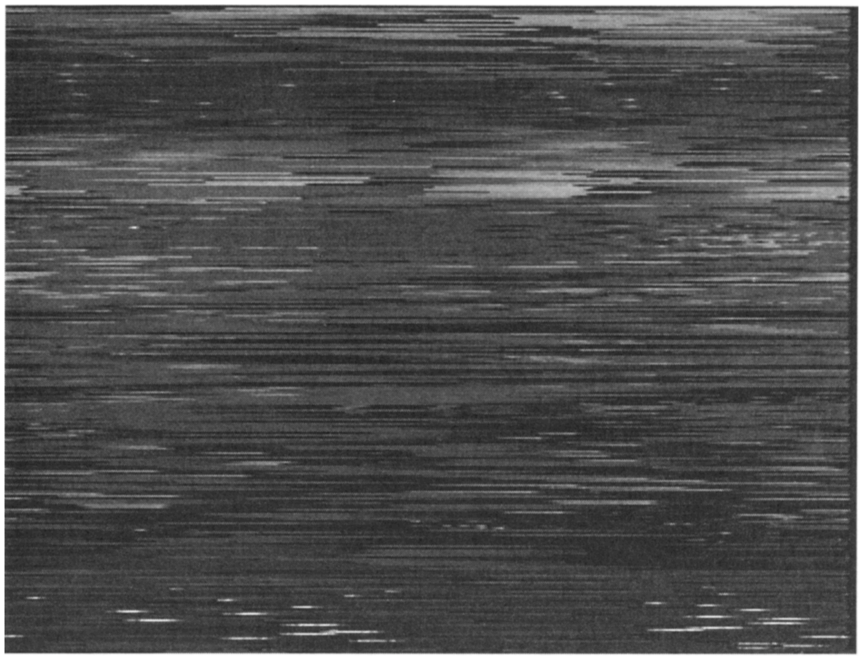


Figure 20.2 Scrambled video frame.

One systemic vulnerability of systems that encrypt only one line of video at a time was that successive lines of video were usually correlated, so it was often possible to reconstruct the image using signal processing techniques. This was first done by Markus Kuhn in 1995, and required the use of a supercomputer at the University of Erlangen to do in real time. Figure 20.2 shows a frame of enciphered video, and Figure 20.3 the same frame after processing. By the time of writing, it's possible to do this on a powerful PC (though still not quite in real time) [733]. If this attack had been feasible earlier, it would have caused a complete break of the system, because regardless of how well the smartcard managed the keys, the video signal could be retrieved without them. But the scrambling technique lasted (just) long enough; pay-TV operators are now moving their customers to fully digital systems in which attacks using properties of the analogue signal are irrelevant.

The generation of the control bytes is of independent interest. Every half second or so, the smartcard supplies the set-top box with a new control word, and this is loaded into a keystream generator which works as follows. There are two linear feedback shift registers (of lengths 31 and 29 in the Eurocrypt system) which generate long linear sequences. Some of the bits of register 1 are used as address lines to a multiplexer, which selects a bit from register 2; this bit becomes the next bit of the keystream sequence. Each successive byte of output becomes a control byte for the scrambler (see Figure 20.4).

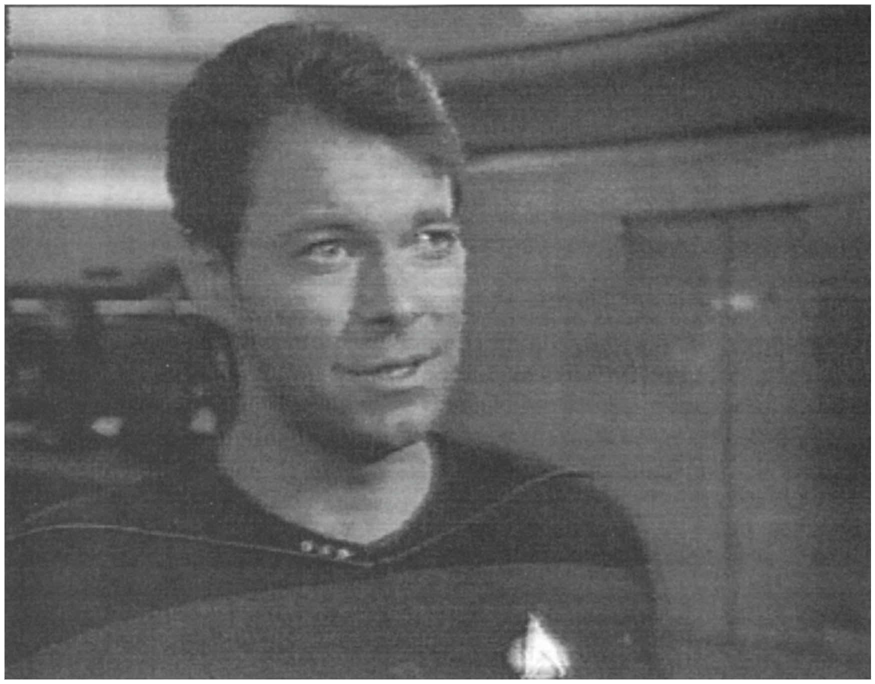


Figure 20.3 Processed video frame.

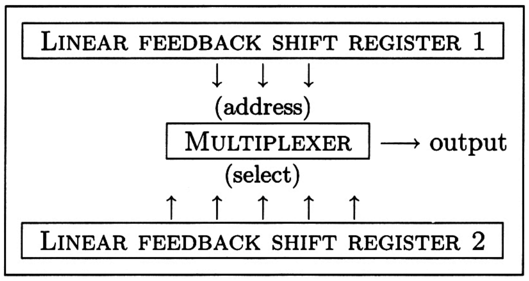

LINEAR FEEDBACK SHIFT REGISTER 1
(address)
MULTIPLEXER
output
(select)
LINEAR FEEDBACK SHIFT REGISTER 2

Figure 20.4 The multiplexer generator.

The designers intended that breaking this cipher should involve guessing the key; and as it is 60 bits long a guess would take on average 2^{59} trials, which is uneconomic—as it has to be done about twice a second. But it turns out that the cipher has a shortcut attack. The trick is to guess the contents of register 1, use this address information to place bits of the observed keystream in register 2, and if this causes a clash, reject the current guess for register 1. (I discovered this attack in 1985, and it's what got me interested in cryptography.) Now the high-order four bits or so of each control word are easy to deduce from interline correlations—it's the least-significant bits you really have to work hard for. So you can easily get about half the bits from a segment of keystream, and reconstruct the control word using cryptanalysis. But this computation is still comparable with the full signal processing attack. The stream cipher, like the scrambling technique, may be weak, but it survived (just) long enough. So the pirates had to attack the subscriber management mechanisms.

20.2.4.3 Subscriber Management Techniques

Given a population of set-top boxes that will unscramble broadcast video given a stream of control words, the next problem is to see to it that only paying customers can generate the control words. In general, this can be done with whitelists or blacklists. But the bandwidth available to last-generation pay-TV systems was low—typically, of the order of ten ECMs per second could be sent, or just over half a million a day. Thus, the blacklist approach was the main one. With a subscriber base of five million customers, sending an individual message to each customer would take over a week.

The basic protocol is that the smartcard interprets the ECMs; and if the current program is one the subscriber is allowed to watch, then a MAC is computed on a series of ECMs using a master key held in the card and supplied to the set-top box as the control word:

$$CW = MAC\ (K;\ ECM_1,\ ECM_2,\ ECM_3,\ ECM_4)$$

In this way, if a subscriber stops paying, their card can be inactivated by sending an ECM that orders it to stop issuing control words; and it needs access to the ECM stream in order to compute the control words at all.

20.2.4.4 What Went Wrong

The first attacks on this system were protocol attacks. Since the control word sent from the smartcard to the set-top box is the same for every set-top box currently unscrambling the program, it is possible for one person to place a PC between the smartcard and the set-top box, record the stream of control words, and post them to the Internet: other people can video-record the scrambled program, and unscramble it later after downloading the control word file [532]. Servers for this key log attack exist, but they are a minor nuisance to the pay-TV industry; not many viewers are prepared to get a special adapter to connect their PC to their set-top box. Others included *blockers*, which would prevent ECMs addressed to your card from being delivered to it; this way, you could cancel your subscription without the station operator being able to cancel your service. Others exploited a master key leakage: someone bought a second-hand PC, looked out of curiosity to see whether there were any interesting deleted files on the hard disk, and managed to undelete a complete subscriber management system for one pay-TV operator—including embedded master keys.

Once this "low-hanging fruit" had been picked, the commercial pirates turned to reverse-engineering customer smartcards using a series of attacks which I described in Chapter 14. But hardware-level fixes were limited to new card issues, and the operators didn't want to issue a new card more than once a year as it cost several dollars per subscriber, and the subscriptions were usually less than $20 a month. So other defensive techniques had to be found.

Litigation was tried, but it didn't work as well as the operators hoped. A lawsuit was lost against a pirate in Ireland, which for a while became a haven from which pirates sold cards by mail order all over Europe. The industry's lobbying muscle was deployed to bring in European law to override Dublin, but this took years and the losses were getting significant. By the middle of 1995, for example, the main U.K. satellite TV station (Sky-TV) was losing 5% of its revenue to pirate cards.

20.2.4.5 How It Was Fixed

All through the mid-1990s, pirates and the operators engaged in a war of countermeasures and counter-countermeasures. The operators would buy pirate cards, analyze them, and develop all sorts of tricks to cause them to fail. The problem faced by the operators was this: when all the secrets in your system are compromised, how can you still fight back against the pirates?

This might seem impossible to the conventional way of thinking about cryptology, but the operators managed it. One of their more effective techniques was an ECM whose packet contents were executed as code by the smartcard; in this way, the existing card base could be upgraded on the fly, and implementation differences between the genuine and pirate cards could be exploited. Any computation that would give a different answer on the two platforms—even if only as a result of an unintentional timing condition—could be fed into the MAC algorithm and used to make the pirate cards deliver invalid control words.

It's worth looking briefly at how to revoke the access rights of subscribers who stop paying. Each of the subscriber smartcards contains a subscriber key k_i, and a binary tree of intermediate group keys *KGij* links the subscriber keys to the currently active master key *KM* (Figure 20.5). Each operational card knows all the group keys in the path between it and the master key. In this scheme, if (say) key *k*2 appears in pirate cards and has to be revoked, the operator will send out a stream of packets that let all the other subscriber cards compute a new master key *KM*□ The first packet will be $\{KM\square\}_{KG12}$, which will let half the subscribers compute KM□at once; then there will be a *KM*□encrypted under an updated version of *KG*11: $\{KM\square\}_{KG\square\square 1;}$ then this new group key *KG*□1 encrypted under *KG* 22; and so on. The effect is that, even with ten million customers, the operator has to transmit fewer than 50 ECMs to do a complete key change. Of course, this isn't a complete solution: operators also need to think about how to deal with pirate cards that contain several subscriber keys, and how leaked keys can by identified without having to go to the trouble of reverse-engineering pirate cards. However, the binary revocation tree is a useful tool in the countermeasures war. (Using individual keys to protect group keys is not really new; Marks recounts how, during World War II, the Special Operations Executive sent its agents *iodoforms*, or open codes, enciphered under their personal keys [523]. When an iodoform was broadcast on the radio, it transmitted an order such as 'blow up a railway bridge' to many agents simultaneously.) Other applications with similar requirements include managing the shared "keys of the day" in naval task forces.

Psychological measures were also used. For example, one cable-TV station broadcast a special offer for a free T-shirt, but prevented legitimate viewers from seeing the 800-number to call; this got it a list of the pirates' customers. Economic factors also made a difference. Pay-TV pirates depend for their success on time-to-market as much as conventional software firms: a pirate who could produce a 99% correct forgery in three weeks would wipe out a competitor who produced a 99.9% forgery after three months. So pirate cards also have bugs, and exploiting them efficiently involves an understanding of pirate economics. It's best to let a pirate build up a substantial user base before you pull the plug on him, as this will destroy his credibility with more potential customers than an immediate response would. But if you leave him too long, he may acquire both the financial and technical resources to upgrade his customers to a high-quality forgery.

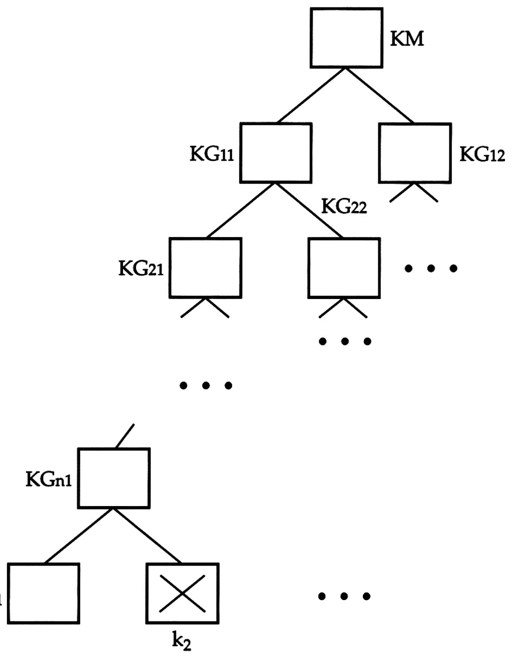


Figure 20.5 Binary revocation tree.

The main technical lesson learned by the pay-TV industry was to plan in advance for security recovery, and to hide a number of features in its products that weren't used initially but could be activated later. (As usual, the same lesson had been learned years previously by another industry—in this particular case the banknote printers.)

Eventually, the smartcards were made a bit more difficult to forge by including proprietary encryption algorithms in the processor hardware. When the attacker could no longer just read out the algorithm with a probing station, but had to reverse engineer part of the chip, it reduced to a few dozen the number of laboratories with the technical capability to do attacks. Many of these laboratories were drawn into the industry's orbit by consultancy deals or other kinds of sponsorship. Those that remained outside the tent, and appeared to pose a threat, were watched carefully. Vigorous legal enforcement provided the last link in the chain. The industry hunted down the main commercial pirates and put them out of business, whether by having them jailed or by drowning them in litigation.

For example, in the last big pay-TV piracy case in the twentieth century, British pirate Chris Cary was convicted of forging Sky-TV smartcards, whose design he had had reverse engineered by a company in Canada for $105,000. He then sold forgeries through a front company in Ireland, where counterfeit cards were not illegal at the time [568]. Sky TV's security consultants infiltrated a spy into Cary's Dublin sales office, and she quietly photocopied enough documents to prove that the operation was really being run from England [403]. The British police didn't want to prosecute, so Sky brought a private prosecution and had Cary convicted. When he later escaped from jail, Sky's private detectives relentlessly hunted him down and eventually caught him in New Zealand, where he had fled using a passport in a dead person's name [367].

The pay-TV story reinforces the business lesson that one must make the engineering and legal aspects of copyright protection work together. Neither is likely to be adequate on its own. An example of how not to do it comes from the world of DVD.

20.2.5 DVD

The consumer electronics industry introduced the *digital video disk* (DVD), later renamed the *digital versatile disk*, in 1996. As usual, Hollywood took fright and said that unless DVD had a decent copy protection mechanism, first-class movies wouldn't be released for it. So a mechanism called the *content scrambling system* (CSS) was introduced.

There is also a scheme whereby the world is divided into five regions, and disks are supposed to run only on players from some designated list of regions. This was to support the traditional business practice of releasing a movie in the United States first, then in Europe, and so on, in order to minimize the loss if it flops. This region code was the first to be broken and is now increasingly ignored by manufacturers. The globalization of markets for products such as DVDs is destroying the market for DVD players that will play only locally manufactured disks.

This left CSS, which was known to be vulnerable by the time that DVD was launched [601]. One industry story was that the designers had been told to come up with a copy protection scheme in two weeks, to use no more than 3,000 gates, and to limit the keylength to 40 bits so the equipment wouldn't fall foul of U.S. export regulations; another story was that DVD consortium only ever intended to compel player manufacturers to license the CSS patent from Matsushita, a condition of which would be implementation of other copy protection mechanisms [119]. No matter whose fault the design was, it's actually quite curious that their system held up for three years.

The detailed description of CSS is currently the subject of frantic litigation, with numerous injunctions issued in the United States against Web sites that have published the code. This is almost certainly futile, as there are plenty sites outside the United States where you can get it (such as [737]). However, because my publishers are located in the United States and I don't want them spending all my royalties on lawyers, here's a suitably abbreviated description.

CSS is based on a stream cipher which is similar to that in Figure 20.4 except that the multiplexer is replaced with a full adder: each successive keystream bit is obtained by adding together the next two outputs from the shift registers with carry. Combining the xor operations of the shift registers with the add-with-carry of the combiner can actually give a strong cipher, if there are (say) five shift registers with coprime lengths greater than 70 [656]. But in CSS, there are only two registers, with lengths 17 and 25, so there is a 2^{16} shortcut attack of exactly the same kind as the one discussed above.

Where the cipher is used to protect keys rather than data, there is a further mangling step; but this only increases the complexity to 2^{25}.

The DVD protocol is as follows. Each player has one or more keys specific to the manufacturer, and each DVD disk has a disk key, *kd*, encrypted under each of the current manufacturer keys, *kmi* (409 of them in 1999): $\{kd\}_{km1}$, $\{kd\}_{km2}$, $\{kd\}_{km3}$, . . . , $\{kd\}_{km409}$. There is also a hash of *kd*, computed by encrypting it with itself: $\{kd\}_{kd*}$ The actual content is protected under sector keys derived from *kd*. Of course, given that the cipher can be broken with 2^{25} effort, any disk key can be found from a single disk hash.

So CSS contravened Kerckhoffs' principle, in that it depended for its protection on the algorithm remaining secret. The DVD consortium appears not to have understood this, as it hoped to keep enough of the manufacturer keys secret by economic pressure. The idea was that if any manufacturer's master key got leaked, then it wouldn't be used on future disks, so his players wouldn't be able to play new releases. So manufacturers would implement decent tamper resistance—or so it was hoped. But the design of CSS doesn't support this. Given any key in the system, all the others can be found at once. Also, the economics of mass-producing consumer electronics doesn't allow the kind of processors required to give serious tamper protection.

Another set of problems came from the fact that the PC is an open platform. The DVD consortium's chosen method of dealing with this was that people producing DVD player software had to obfuscate their code so that it would be hard to reverse-engineer. Papers duly appeared on tricks for systematic software obfuscation [58]. These tricks may have pushed up the cost of reverse engineering from a few days of effort to a few weeks, but once the CSS design was out, that was it.

An even more serious problem with the openness of the PC came from Linux, the open source PC operating system used by millions of people. The DVD consortium's philosophy and architecture was not consistent with making DVD drivers available to the Linux community. So as PCs with CD drives started being replaced in the stores with PCs fitted with DVD drives, the Linux user community either had to break CSS or give up using Linux in favor of Windows. Under the circumstances, even if every DVD player had contained a pay-TV-grade smartcard processor, it was only a matter of time before someone read it out.[1]

One result of the break is a program (DeCSS) that will unprotect any DVD. The industry's reaction was to reach for their lawyers. Web sites in the United States which host DeCSS get hammered with injunctions, which simply cause the software to become ever more widely distributed and make the industry look foolish [491]. There are some quite unpleasant undercurrents, though. For example, copyright law traditionally allows *fair use*, which includes copying parts of a work for the purpose of scholarship, quotation, and even ridicule; the movie industry lawyers seek to squash this for digital

[1] This error may well be repeated with the *secure digital music initiative* (SDMI), a proposed replacement for MP3. SDMI will use encrypted audio streams that will be decrypted in the soundcard driver software in the PC operating system. There will also be a watermarking scheme. However, depriving Linux users—who probably include most of the world's computer science and engineering students—access to the latest audio unless they mount the despised Windows operating system is guaranteed to create many capable motivated opponents. The likely watermarking scheme—echo hiding—was already broken in [610] by Fabien Petitcolas.

media so that copyright holders have completely unfettered control over what happens to a digital work. This would be disastrous for universities, public libraries, and many other bodies, where the exploitation of fair use rights is strongly entrenched. So the battle referred to by Barlow in the quote at the head of this chapter has started.

A leading U.S. authority, Samuelson, takes the view that some copying is beneficial to publishers in a much wider range of industries than just software [665]. A European expert put it more strongly: copyright laws are tolerated only because they are not enforced against the large numbers of petty offenders [610]. It is worth noting that even if Hollywood gets all it wants in the U.S. courts, it's unlikely to get quite the same result in Europe, where copyright law specifically allows reverse engineering for the purpose of building compatible equipment, and where a video rental treaty may protect temporary copies [666]. I'll return to all this in Chapter 21 when we discuss e-policy.

Another point (made for example in [491]) is that small-scale copying of DVDs is uneconomic anyway, as home-burnable DVD disks cost more than prerecorded ones; that large-scale copying in the Far East already happens; and that the real reason for the litigation is that the publication of CSS enables anyone to build a DVD player without paying royalties to the DVD consortium.

Anyway, DVD is following the usual pattern: Hollywood terrified, and refusing to release its best movies; technical measures taken to prevent copying, which got broken; then litigation. A reasonable person might hope that once again the studios will see sense in the end, and make a lot of money from selling DVDs. There will be copying, of course, but it's not entirely trivial yet—even a DSL modem takes hours to send a 4Gb DVD movie to a friend, and PC disk space is also an issue. Eventually, as DVD drives replace CD drives in all PCs, we can expect to see rewriteable DVDs being widely used for backup; and it's completely predictable that whatever new mechanisms are fielded to prevent copying will be circumvented. But I also predict that in 10 years' time, the lineup of DVDs on my shelf will be pretty much the same as my videocassette lineup is today—about 50 prerecorded cassettes and maybe two dozen home-recorded ones, the former bought mostly for the family and the latter being mostly old TV programs that have a direct relevance to my work. I'm sure the industry can live with that.

Meanwhile, strenuous efforts are being made to improve DVD security by fitting the next generation of players with mechanisms based on copyright marking. This is an interesting technology, and worth a look.

20.3 Information Hiding

Hollywood's interest in finding new mechanisms for protecting copyright came together in the mid-1990s with the military's interest in unobtrusive communications and public concerns over government efforts to control cryptography, and started to drive rapid developments in the field of *information hiding*. This largely refers to techniques that enable data to be hidden in other data, such as when a secret message is hidden in an MP3 audio file, or a program's serial number is embedded in the order in which certain instructions are executed.

The Hollywood interest is in *copyright marks*, which can be hidden unobtrusively in digital audio, video, and artwork. These are generally either *watermarks*, which are hidden copyright messages, or *fingerprints*, which are hidden serial numbers.

The privacy interest is in *steganography*, whose purpose is to embed a message in some cover medium in such a way that its very existence remains undetectable. A common conceptual model, proposed by Simmons [700, 707], is as follows. Alice and Bob are in jail, and wish to hatch an escape plan; all their communications pass through the warden, Willie; and if Willie detects any encrypted messages, he will frustrate their plan by throwing them into solitary confinement. So they must find some way of hiding their secret messages in an innocuous-looking covertext. As in the related field of cryptography, we assume that the mechanism in use is known to the warden, so the security must depend solely on a secret key that Alice and Bob have somehow managed to share.

There is some similarity with electronic warfare. First, if steganography is seen as a low-probability-of-intercept communication, then copyright marking is like the related jam-resistant communication technique: it may use much the same methods but in order to resist focused attacks it is likely to have a much lower bit rate. We can think of Willie as the pirate who tries to mangle the audio or video signal in such a way as to cause the copyright mark detector to fail. Second, techniques such as direct sequence spread spectrum, which were originally developed for electronic warfare, are finding wide use in the information hiding community.

Of course, copyright marks don't have to be hidden to be effective. Some TV stations embed their logo in a visible but unobtrusive manner in the corner of the picture, and many ECMS systems have control tags bundled quite visibly with the content. In many cases, this is the appropriate technology. However, in what follows I'll concentrate on hidden copyright marks.

20.3.1 The DVD Marking Concept

A current objective of the DVD consortium is to find a copyright marking scheme that will enforce serial copy management. Videos might be unmarked, marked "never copy," or marked "copy once only"; compliant players would not record a video marked "never copy," and when recording one marked "copy once only" would change its mark to "never copy." Commercially sold videos would be marked "never copy," while TV broadcasts and similar material would be marked "copy once only." In this way, the DVD players available to consumers would allow unlimited copying of home videos and time-shifted viewing of TV programs, but could not easily be abused for commercial piracy. There is an overview of the proposed mechanisms in [119].

The basic idea is simple [504]. For each disk, choose a *ticket*, X, which can be a random number, plus copy control information, plus possibly some information unique to the physical medium, such as the wobble in the lead-in track. Use a one-way hash function h to compute $h(X)$ and then $h(h(X))$. Embed $h(h(X))$ in the video as a hidden copyright mark. See to it that compliant machines look for a watermark, and if they find one will refuse to play a track unless they are supplied with $h(X)$, which they check by hashing it and comparing it with the mark. Finally, arrange things so that a compliant device will record a marked track only if given X, in which case only $h(X)$ is written to the new disk. In this way, a "copy once only" track in the original medium becomes a "copy no more" track in the new medium.

Doing copy generation management using embedded marks, rather than with attached data, has the advantage that it can survive conversion from digital to analogue and back to digital. This leads to a number of problems. First, we need a method of embedding a mark in audio or video, which—even though it might take a lot of effort to embed—can be detected readily and is difficult for an attacker to remove. Second, the detection must be carried out by mass-market equipment, that is, using cheap processors or custom silicon with a limited gate count, and have a low false positive alarm rate [554]. For example, if your legitimate DVD player were to detect a mark in your wedding video by mistake, you'd have to buy a pirate player to watch it.

20.3.2 General Information-Hiding Techniques

Information hiding goes back even further than cryptology, having its roots in camouflage. Probably the first historical mention is in Herodotus who records tricks used during the wars between the Greeks and the Persians—including hiding a message in the belly of a hare carried by a hunter, tattooing it on the shaven head of a slave whose hair was then allowed to grow back, and writing it on the wooden base under the wax of a writing tablet [377]. Francis Bacon proposed a system that embedded a binary message in a book at one bit per letter by alternating between two different fonts [607]. Until quite modern times, most writers considered hiding confidential information much more important than enciphering it [805]. Military organizations still largely hold this view and have used all sorts of technologies, from the microdots used by spies in much of the twentieth century to the low-probability-of-intercept radios discussed in Chapter 16.

When it comes to hiding data in other data, the modern terminology of the subject is as follows [614]. The copyright mark, or in the case of steganography, the *embedded text*, is hidden in the *cover-text* producing the *marked text* or in the case of steganography the *stego-text*. In most cases, additional secret information is used during this process; this is the *marking key* or *stego-key*, and some function of it is typically needed to recover the mark or embedded text. Here, the word "text" can be replaced by "audio," "video," and so on, as appropriate.

A wide variety of embedding schemes have been proposed.

- In many ways the obvious technique is to hide the mark or secret message in the least-significant bits of the audio or video signal. Many public domain steganography tools do this. But it isn't usually a very good strategy, as the hidden data is easy to detect statistically (the least-significant bits are no longer correlated with the rest of the image), and it's trivial to remove or replace. It's also severely damaged by lossy compression techniques.
- A classic technique is to hide the mark or secret message at a location determined by the secret key. This was first invented in classical China. The sender and receiver had copies of a paper mask, which had holes cut out of it at ran-

dom locations. The sender would place his mask over a blank sheet of paper, write his message in the holes, then remove it and compose a cover message including the characters of the secret embedded message. This trick was reinvented in the sixteenth century by the Italian mathematician Cardan and is now known to cryptographers as the Cardan grille [428].

- A modern implementation of this hides a copyright or other message in a `.gif` format image as follows. A secret key is expanded to a keystream, which selects an appropriate number of pixels. The embedded message is the parity of the color codes for these pixels. In practice, even a quite large number of the pixels in an image can have their color changed to that of a similar one in the palette without any visible effects [413]. However, if all the pixels are tweaked in this way, then the hidden data is easy to remove by just tweaking them again. A better result is obtained if the cover image and embedding method are such that (say) only 10% of the pixels can safely be tweaked. Then, if the warden repeats the process, but with a different key, an independent 10% of the pixels will be tweaked and only 10% of the bits of the hidden data will be corrupted.
- In general, the introduction of noise or distortion—as happens with lossy compression—will introduce errors into the hidden data almost regardless of the embedding method unless some kind of error correcting code is added. A system proposed for banknote marking, Patchwork, uses a repetition code—the key selects two subsets of pixels, one of which is marked by increasing the luminosity and the other by decreasing it. This embeds a single bit; the note is either watermarked using that key, or it isn't [96, 357]. In the general case, one may want to embed more than one bit, and have the embedded data to survive very high levels of induced errors. So a common technique is to use direct sequence spread spectrum techniques borrowed from electronic warfare [748].
- Spread spectrum encoding is often done in a transform space to make its effects less perceptible and more robust against common forms of compression. These techniques are also commonly used in conjunction with perceptual filtering, which emphasizes the encoding in the noisiest or perceptually most significant parts of the image or music track, where it will be least obtrusive, and de-emphasizes it in quiet passages of music or large expanses of color [127].
- Some schemes use the characteristics of particular media, such as a scheme for marking print media by moving text lines up or down by a three-hundredth of an inch [135], or adding extra echoes to music below the threshold of perception [96]. So far, such techniques don't seem to have become as robust, as generic techniques based on keyed embedding using transform spaces, spread spectrum, and perceptual filtering.

Progress in copyright marking and steganography was very rapid in the last few years of the twentieth century. Its history has repeated that of cryptology, but on a much more compressed timescale: people invented marking schemes, which other people broke, and eventually the technology became more mature and robust.

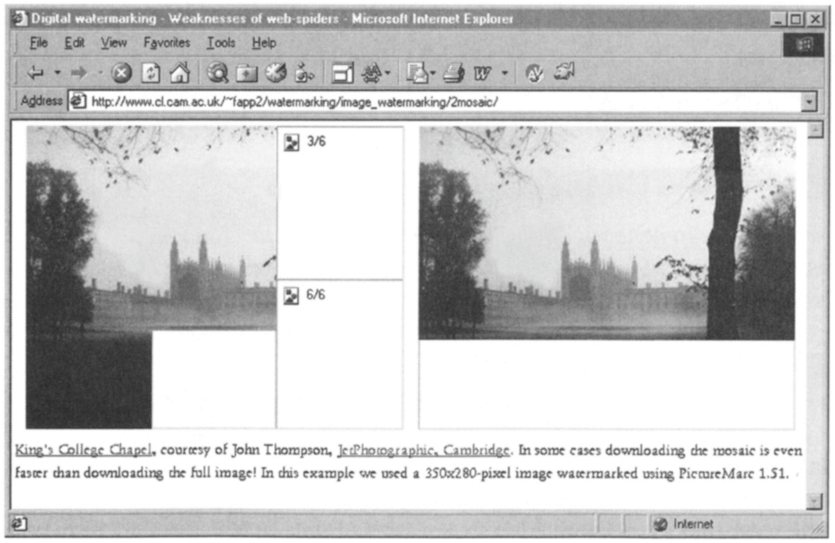
20.3.3 Attacks on Copyright-Marking Schemes

Throughout this book, I've described attacks on cryptographic systems that occasionally involved cryptanalysis, but more often relied on mistaken assumptions, protecting the wrong things, protocol failures, and the opportunistic exploitation of implementation bugs. Copyright marking has been no different.

- In the beginning, many people assumed that the main market would be watermarking—embedding hidden copyright messages so that ownership of a work could be proved in court. This has turned out to be mistaken. Intellectual property lawyers almost never have any difficulty in proving ownership of an exhibit; and they don't rely on technical measures that might confuse a jury, but on documents such as contracts with bands and model release forms. The legal use of copyright marks may rather be for fingerprints, namely hidden serial numbers.
- The first large vendor of marking systems—Digimarc—then set up a service to track intellectual property on the Web. This has clearly got some potential, as one the main costs faced by multimedia producers is tracking the copyright of large numbers of images and the royalties due to their owners. However, the Digimarc system could be easily defeated by guessing the master password or by modifying the marking software so that it would overwrite existing marks. They also had a "Marc spider," a bot that crawled the Web looking for marked pictures and reporting them to the copyright owner; but there were a number of ways to defeat this [610].
- Many marks are simply additive. This opens a whole series of possible vulnerabilities. For example, if all the frames in a video carry the same mark, it is possible to average them to get the mark and then subtract it out. An even simpler attack is to supply some known content to a marking system, and compare its input and output—just like the chosen plaintext attacks possible on some cipher systems. And if a picture, P, with a mark, m, is just $P + m$, then a competitor whose mark is $m\_$ might simply claim that the original was $P + m - m\_$, and so the published picture $P + m$ was really marked with $m\_$.
- As usual, many designers ignored Kerckhoffs' principle—that the security of a system should reside in the choice of key, not in the algorithm in use. But this principle applies with greater than usual force when marks are to be used in evidence, as this means disclosing them in court. In fact, as even the marking keys may need to be disclosed, it may be necessary to protect objects with multiple marks. For example, one can have a mark with a secret key that is system wide and that serves to identify which customer re-sold protected content in violation of his license, and a second mark with a unique key that can be disclosed in court when he's prosecuted.

- There have been various attempts to develop a marking equivalent of public key cryptography, so that (for example) anyone could insert a mark that only one principal could detect, or anyone could detect a mark that only one principal could have inserted. The former seems just about feasible if the mark can be inserted as the cover audio or video is being manufactured [210]. The latter is the case of particular interest to Hollywood. However, it seems a lot harder than it looks, as there is a very general attack. Given a device that will detect the mark, an attacker can remove a mark by applying small changes to the image until the decoder cannot find it anymore [606, 505].
- Some neat steganalysis techniques were developed to break particular embedding schemes. For example, when the mark was added by either increasing or decreasing the luminosity of the image by a small fixed amount, this caused the peaks in the luminosity graph to become twin peaks, which meant that the mark could be filtered out over much of many images [519].
- Another family of attacks exploit the properties of particular media. For example, the typical Web browser, when presented with a series of graphics images, will display them one after another without any gaps; so a marked image can often be chopped up into smaller images, which together will look just like the original when displayed on a Web page but in which a copyright mark won't be detected (see Figure 20.6) [610].
- The most general known attacks on copyright marking schemes involve suitably chosen distortions. Audio marks can be removed by randomly duplicating or deleting sound samples to introduce inaudible jitter; techniques used for click removal and resampling are also powerful mark removers. For images, there is a tool we developed called Stirmark, which introduces the same kind of errors into an image as printing it on a high-quality printer and then scanning it again with a high quality scanner. It applies a minor geometric distortion: the image is slightly stretched, sheared, shifted, and/or rotated by an unnoticeable random amount (see Figure 20.7). This defeated almost all the marking schemes in existence when it was developed, and is now a standard benchmark for copyright mark robustness [610]. In general, it's not clear how to design marking schemes that will resist a *chosen distortion attack*, in which the attacker who understands the marking scheme mangles the content in such a way as to cause maximum damage to the mark while doing minimal damage to the marked content.

For a fuller account of attacks on copyright marking schemes, see [610, 611]. The technology's improving slowly but the limiting factor appears to be the difficulty of designing marking schemes that remain robust once the mark detection algorithm is known. If any copy control scheme based on marking is implemented in PC software or low-cost tamper-resistant processors, it's only a matter of time before the algorithm gets out; then expect to see people writing quite effective unmarking software.


Digital watermarking - Weaknesses of web-spiders - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address
3/6
6/6
Internet

http://www.cl.cam.ac.uk/~fapp2/watermarking/image_watermarking/2mosaic/

King's College Chapel, courtesy of John Thompson, JetPhotographic, Cambridge. In some cases downloading the mosaic is even faster than downloading the full image! In this example we used a 350x280-pixel image watermarked using PictureMarc 1.51.

Figure 20.6 The Mosaic attack (courtesy Jet Photographic, `http://www.jetphotographic.com`).

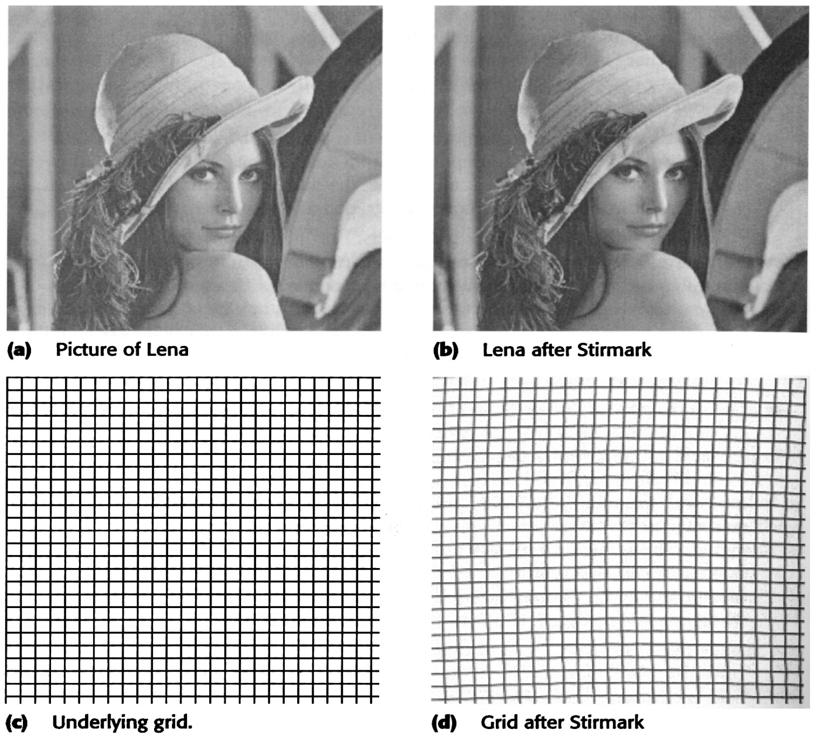


Figure 20.7 The effect of Stirmark.

20.3.4 Applications of Copyright-Marking Schemes

The applications of marking techniques are much broader than just DVDs and still pictures distributed on the Net. Radio adverts in the United States are commonly marked with a serial number, to enable auditing agencies to check automatically whether stations are playing them as often as they claim. Color copiers sold in the United States have their serial number hidden in the bit patterns of copies, as a means of detecting currency forgers [797]. Apparently there will be digital watermarks in the new Euro notes, which will shortly replace many European currencies; and there has been a call for proposals from the U.S. Bureau of Engraving and Printing, which wants to do something similar.

While most copyright marks have to be robust to withstand distortion attacks, some applications have been found for marks that are deliberately made as fragile as possible. One proposal highlights any changes made in an image *after* it was applied, and might be useful in assuring the integrity of images to be used in evidence [489]. Another proposed use of fragile watermarking is to hold the tickets for the DVD copy protection scheme [119].

Then there's a class of proposed applications that have to do with convenience or safety, rather than preventing malicious behavior. It has been proposed that music broadcast over the radio should be marked with the CD's number, so that someone who likes it could order the CD automatically by pressing a button. And in medicine, digital versions of X rays often get separated from the patient's details, as the various proprietary file formats get mangled through numerous protocol conversions; this safety problem could be solved by embedding patient details directly in the image.

Finally, perhaps a quarter to a third of information-hiding research doesn't aim at Hollywood's requirements, or those of the Bureau of Engraving and Printing, but at hiding information for privacy purposes.

20.4 Privacy Mechanisms

The technology of privacy includes two types of mechanism: those with which people discharge obligations of confidentiality to third parties, as discussed in Chapter 8, and those which individuals can use to protect their own privacy in the face of surveillance or other intrusion by third parties. The former are more important in the general scheme of things: without the obligations of confidentiality owed to us by doctors, lawyers, bankers, and other service providers, society would be very different. However, the citizen's ability to keep certain things private remains an important backstop, and privacy mechanisms are of much wider importance. To understand why, we have to examine what these mechanisms achieve.

In pre-technological societies, the available protection included not just cryptography in the form of hand ciphers and steganography in the form of prearranged signals, but the fact that two people could walk a short distance away from everyone else and have a conversation that left no hard evidence of what was said. If Alice claimed that

Bob had tried to recruit her for an insurrection, then Bob could always claim the converse—that it was Alice who'd proposed to overthrow the king and he who'd refused out of loyalty. In other words, some communications were *deniable*. Plausible deniability remains an important feature of some communications today, from everyday life up to the highest reaches of intelligence and diplomacy. In some circumstances, it can be implemented by convention: for example, in some countries, a participant in litigation can write a letter marked "without prejudice" to another and propose a settlement; this letter cannot be used in evidence. However, there are many circumstances without such clear and convenient rules, and where the electronic nature of communication means that "just stepping outside for a minute" isn't an available option. What then?

Another issue is anonymity. Until the industrial revolution, most people lived in small villages where everyone knew everyone else's business. For many people, it was a relief to move into a town, where this wasn't the case. Nowadays, the phrase "electronic village" not only captures the way in which electronic communications have shrunk distance, but also for many people the fear that as everything goes online, as data get collected about every transaction, accumulated into marketing profiles and sold, so we will return to something resembling the status quo of the seventeenth century. Everything about us will be known. Of course, if you live in a country such as Germany with fierce data protection laws, then you may be safe (as long as all your business remains in Germany). But as soon as you shop at an online store in the United States, that protection is gone. Is there some way to conduct online business anonymously?

20.4.1 Content Hiding: PGP

One of the best-known and widely used privacy tools is encryption of electronic mail. The market-leading product, *Pretty Good Privacy* (PGP), has done much to raise public awareness of the issues—especially since the U.S. government harassed its author, Phil Zimmermann, and threatened to prosecute him for allegedly breaking U.S. export controls by making encryption software available on the Net.

PGP has a number of features but in its most basic form, each user generates a private/public keypair. To protect a message, you sign a hash of it using your private key, encrypt both the message and the signature with a randomly chosen session key, then encrypt the session key using the public key of each of the intended recipients. Thus, if Alice wants to send an encrypted email to Bob and Charlie, she forms the message:

$$\{KS\}_{KB}, \{KS\}_{KC}, \{M, \mathrm{sig}_{KA}\{h(M)\}\}_{KS}$$

The management of keys is deliberately left to the user, the rationale being that a single centralized certification authority would become such an attractive target that it would likely be cracked or come under legal coercion. The intended mode of operation is that each user collects the public keys of people she intends to correspond with, and bundles them into her *public keyring* which she keeps on her system. The public keys can be authenticated by any convenient method such as by printing them on her business card. To make this easier, PGP supports a *key fingerprint* which is a one-way hash of the public key, presented as a hexadecimal string.

Another mechanism to help users manage trust is that they can sign each others' keys. This may simply be used as an integrity protection mechanism on their public keyrings, but becomes more interesting if the signatures are exported. The set of publicly visible PGP signatures makes up the *web of trust:* the idea is that if Alice wants a public key for Bob, with whom she has not corresponded before, then she might be lucky enough to find a key for Bob signed by Charlie, and a key for Charlie signed by David, whom she already trusts. The resulting *certificate chain:*

$\mathrm{sig}_{KC}\{KB\}$, $\mathrm{sig}_{KD}\{KC\}$, $\mathrm{sig}_{KA}\{KD\}$

can be taken to be equivalent to the trust relationship she seeks, namely $\mathrm{sig}_{KA}\{KB\}$ provided she is prepared to take the risk of either Charlie or David having been dishonest or incompetent.

Other mechanisms for distributing PGP keys have been developed. One of the most widely used is a series of *key servers* which contain large collections of PGP keys. Some caution is needed, as anyone can put up a key there with any attached email address; keys for addresses such as `president@whitehouse.gov` aren't controlled by the people you might normally associate with them. There was also a book of important public keys published [42]; this also contains information about some bugs in early versions of PGP.

Of course, encrypting email is only part of the solution. In some countries, including Russia, Zimbabwe, and Britain, the police have the power to require you to decrypt ciphertext they seize, or even hand over the key. This power is also available to the civil courts in many countries under subpoena, and to many tax authorities. Other situations in which coercion may be a problem include where soldiers or intelligence agents could be captured; where police power is abused, for example to seize a key on the grounds of a supposed criminal investigation but where in reality they've been bribed to obtain commercially confidential information; and where private individuals may be tortured by robbers into revealing information such as the secret codes for their bank cards and the location of safes [793].

In such circumstances, there is a serious problem with systems where private keys are long-lived. If the taxman seizes your private key while investigating you, then leaves this on a server shared with other government agencies, these agencies now have the power to decrypt any of your old incoming messages which they happen to have stored—and perhaps also to forge your digital signature.

So the latest versions of PGP have separate keypairs for encryption and signature:

- Your public signature verification key is the long-term key which you get people to sign, print on your business card, include in your signature file, and so on.
- You generate a set of time-limited encryption/decryption keypairs and sign the public encryption keys using your long-term signature key.
- You delete your private decryption keys after they expire.

The U.S. Defense Messaging System uses a similar mechanism, but it supports the use of short-lived public encryption keys. Each user has a key server that will provide a fresh encryption key on demand, signed by the user's signing key; once the message is received and decrypted, the decryption key is destroyed.

However, there are limits to what can be done with cryptography alone, and many conventional IT security mechanisms can even endanger privacy [296]. Encryption use may mark your messages for traffic analysis; authentication can identify users unambiguously, removing wriggle room in censorship, defamation, and copyright infringement cases; and in many jurisdictions, naive encryption can be countered by what's called *rubber hose cryptanalysis*—the police simply beat the key out of you. (Countries such as Britain are slightly more civilized; there's now a law there that lets a police officer demand your key and send you to jail if you refuse. I'll discuss this in Chapter 21.)

20.4.2 Content Deniability—Steganography

When the threat model includes coercion, simply destroying old keys may not be enough, as the very existence of protected material can be sufficient to cause suspicion. In such circumstances, more complete plausible deniability can be provided by the use of steganography. If the secret message is well hidden in an innocuous cover object such as an MP3 audio track, then with luck the opponent will never suspect that anything clandestine is taking place.

Stored data is particularly difficult. Most customs authorities have the power to require travellers to decrypt any material found on the hard disk of their laptop in order to check for subversive material, pornography, and the like. There are many crude ways to hide the existence of files, such as having a separate partition on your hard disk that runs Linux, which the customs men probably won't understand—but against a capable opponent such defenses are ineffective, and over time even the customs man will acquire suitable tools. Files can be hidden using steganography tools in larger multimedia files, but this can be inefficient.

This led to the design of the *steganographic file system*, which has the property that a user may provide it with the name of an object, such as a file or directory, together with a password; and if these are correct for an object in the system, access to it will be provided. However, an attacker who does not have the matching object name and password, and lacks the computational power to guess it, can get no information about whether the named object even exists. This is an even stronger property than Bell-LaPadula; Low cannot even demonstrate the existence of High. The user can give the customs man the Low password, and deny that a High password exists; the customs man should never be able to prove that the user lied.

The whole disk is encrypted, and fragments of the files are scattered through it at places that depend on the password, with some redundancy to recover from cases where High data is accidentally overwritten by a Low user [49]. There is an early implementation described in [536]. Of course, a really robust implementation would have to take account of many of the multilevel security issues discussed in Chapter 7, from covert channels to limiting the damage that can be done by malicious code; there are also some peculiarly difficult threats to steganographic systems, such as what happens when successive snapshots of the system are taken by a Low user who then tries to deduce whether any High writes have occurred meanwhile. This problem is still not fully solved, and better implementations would be useful.

20.4.3 Association Hiding—Remailers and the Dining Cryptographers

However, there are limitations to what even steganography can do. As I remarked in several contexts, the opponent often gets most of his information from traffic analysis. Even if the communications between Alice and Bob are encrypted, and the ciphertext is hidden in MP3 files, and even if on inspection neither Alice's laptop nor Bob's contains any suspicious material—whether because it's hidden on a stego file system or because it was simply memorized and deleted—the mere fact that Alice communicated with Bob may give the game away. This is why criminals set much more store by anonymous communication (such as using prepaid mobile phones) than by encryption. Of course, there are legitimate uses too, such as anonymous helplines for abuse victims, whistleblowers, police informants, and protest groups that want to dig a perfectly legal elephant trap for the government of the day. There's anonymous student feedback for university professors, anonymous refereeing of conference paper submissions, and anonymous HIV tests where you get the results online using a one-time password that came with a test kit you bought for cash. You may want to apply for a job without your current employer finding out, to exchange private email with people who don't use encryption, or fight a harmful cult. There's also the simple matter of preserving privacy in a world where ever more businesses collect and trade personal information. So how can anonymity be assured online?

There are two basic mechanisms, both invented by David Chaum in the 1980s. The first is the *mix* or *anonymous remailer* [177]. This is a device which accepts encrypted messages, strips off the encryption, and then remails them to the address that it finds inside. In its simplest form, if Alice wants to send anonymous email to Bob via Charlie and David, she composes the message:

A □ C: $\{D, \{B, \{M\}_{KB}\}KD\}KC$

Charlie now strips off the outer wrapper and finds David's address, plus a ciphertext. He sends the ciphertext to David, who decrypts it and finds Bob's address, plus a ciphertext. He sends the ciphertext to Bob, who decrypts it and gets the message *M*. Of course, an anonymous remailer could be an attractive honey trap for a law enforcement agency or intelligence agency to operate, and so it's common to send messages through a number of successive remailers and arrange things so that most of them would have to conspire to break the traffic.

There are many refinements on this basic technique. In order to prevent an opponent tracking messages through one remailer after another, it's common for message sizes to be fixed; for remailers to batch up messages or to forward them after random delays; and for message replay to be detected. Some allow replies to unknown destinations, and others don't; anonymous replies may be handled by a pseudonym service [531].

Anonymous connections aren't limited to email, but can include any kind of communications service: an experimental U.S. Navy system, called *Onion Routing*—because the messages are nested like the layers of an onion—can be used as a communications primitive on which services such as mail and Web access can be layered [637]. There's also a design for anonymous networks of ISDN digital telephones, which might conceivably be built on top of third-generation mobile services [312, 419]. Indeed, the existence of anonymous communication channels greatly simplifies the design of more complex services with anonymity requirements, such as elections and digital cash [708]; and in the real world they can be usefully implemented by non-cryptographic means such as broadcast by access tokens or other low-cost portable devices [732].

While anonymous communications based on remailers provide protection that depends on all sorts of aspects of the implementation—such as whether replay or other chosen-traffic attacks are possible—there is another stronger mechanism that is not so dependent, and thus may be considered the anonymity equivalent of "unconditional security." This was also introduced by Chaum as the *dining cryptographers' problem*, inspired by the "dining philosophers' problem" in distributed systems, discussed in 6.1.4.

Several cryptographers are gathered around a table for dinner, and the waiter informs them that the meal has already been paid for by an anonymous benefactor, who could be one of the participants or the NSA. The cryptographers would like to know who. So pairs of principals share one-time pads, after which each principal outputs a function of her "I paid/I didn't pay" bit, and everyone can later work out the total parity of all such bits. As long as not more than one of the cryptographers says "I paid," even parity means that the NSA paid, while odd parity means that one of the diners paid, even if nobody can figure out who [179]. Various extensions have been proposed, including one in which "dining drug dealers" can auction a consignment of cocaine without revealing the bidders' identities to the other bidders or to the seller. Nobody except buyer and seller know who won the auction; and even the seller is not able to find out the identity of the highest bidder before committing to the sale [732].

Doing anonymity properly is hard. As mentioned above, the anonymous remailer itself might be owned by the enemy. One option is to buy a service from a company whose main business is providing anonymity, of which the most prominent is Zero Knowledge Systems—such a firm has a lot to lose if they are exposed as dishonest or incompetent at their chosen trade. Another is to use remailers operated by cypherpunks or by a research team at a major university. Even then, there are still potential problems. There are all sorts of attacks involving chosen traffic insertion, which may allow powerful opponents (those who can monitor traffic at a large number of places in the Internet) to track relationships between correspondents [359]. Even more service denial attacks are possible on the remailer itself. People who want the service closed down can send large amounts of junk mail to or through the system; they can try to get it into a mail loop with high-volume mailing lists; they may even turn up with a subpoena. The best account of such attacks comes from David Mazières and Frans Kaashoek's experience of running the MIT server [531].

Another possibility is for the mail or Web forwarding functions to be undertaken by the users rather than by a centralized service. *Crowds* is a system in which users group together and do Web page forwarding for each other. In this way, if one of them downloads a subversive Web page, then the secret police have several hundred suspects to deal with [641]. A similar scheme was devised by a well-known CEO who, each morning, helps himself at random to one of the mobile phones of his managers, and has his switchboard forward his calls.

For many purposes, elaborate technical protection mechanisms are unnecessary. There are several online services which enable people to browse the Web anonymously, such as Anonymizer [52]. Users can set up a session with these services and enter the URLs of Web pages they want fetched; the anonymizing service will do this, while filtering those parts of the `http` protocol (such as cookies) that could reveal the client's identity. Some of the services offer encrypted sessions, in some cases at a premium price. In fact, any Web cache will provide some level of anonymity, because pages are fetched on the user's behalf. However, that does mean that the cache will have some very interesting logs!

Implementing high-quality anonymity is hard, not just for all these reasons but also for those discussed in Section 19.7: merchant sites are forever dreaming up new cache-busting tricks to ensure that customers see their ads and make their identities available, and many of these can break anonymity in one way or another. For a survey of anonymizing services, see [524]; for a discussion of defeating them using Web redirects, Java applets, and so on, see [716, 654]. It is particularly hard to flush the internal state of some browsers, such as Internet Explorer, so care must be taken if the opposition might gain control of your PC later.

However, the most common anonymity services are the Internet cafe and the throwaway Web-based email address. Many places offer Net access for a fixed cash sum per hour, and there are services that offer free email accounts which can be accessed from a browser and are supported by advertising, without any authentication of the users' claims to identity. Some of these services offer SSL-encrypted access for added privacy. Combining the two is a very attractive proposition for the acutely privacy conscious, because, provided you pay cash, there may be no durable record at all linking your electronic persona to your physical person. Of course such services occasionally get abused. There was public alarm in Britain after a neo-nazi downloaded bomb-making information in a London cybercafe, after which he bombed black and Asian districts and a gay bar, killing three people and injuring more than 70 others; but it's unclear that he actually got anything from the anonymity [191]. Of course, cybercafes and throwaway email accounts are useful for all sorts of legitimate purposes.

20.4.4 Association Deniability—Digital Cash

Even if you use a throwaway email account, you may want to shop on the Net, and this usually means giving a credit card number. As I mentioned in the chapter on electronic commerce, the merchants will routinely build a marketing profile of you, indexed by your credit card number (even though this breaches the banks' standard conditions of business). You may be lucky enough to get a credit card in a false name. (This can even be legitimate—in the United States if you work for an organization in the intelligence community, and in Britain if you're entitled to police protection for some reason.) But this still won't stop the transactions that you make being linked together into a profile for the marketers.

This raises the question of whether there is an electronic equivalent of cash—that is, a payment medium that is anonymous, untraceable, and unlinkable. There have been various attempts to do this. Some vendors of electronic purses claim that their products are anonymous, as the purse itself has only a serial number, and the link between it and the customer's name is known only to the issuing bank. (Some of them have got into trouble with advertising and trading standards authorities as their claims weren't all that well founded.) The most interesting protection concept in this space is *digital cash*, another invention by Chaum [178, 180].

In Chapter 6, I explained the underlying technical idea—the blind signature. The customer constructs a banknote according to an agreed format, and presents it to the bank for signature after multiplying it by a suitable random blinding factor. Things are arranged so that after the signature is done, the blinding factor can be removed, leaving a digital coin, or, more accurately, a digital cashier's check whose serial number is not known to the bank. Extra features are needed to ensure that the bank can detect whether a coin has been spent twice [180]; a modern digital cash system is described in [134].

Digital cash has been tried, but so far has not succeeded in the marketplace; the first company to launch it as a product, Digicash Inc., ended up in bankruptcy. The search for applications continues. There have been pilot projects involving road tolls; another possibility is the management of pseudonyms for private online e-commerce [134]; still another is that medical insurance schemes could adopt anonymous health credit cards to protect patient privacy [117].

The basic idea behind all these systems is that a customer's relationship with a merchant can be revealed only by the customer (for example, by showing a receipt). These systems require as part of their infrastructure an anonymous communication system (otherwise, the merchant can just read off the customer's name directly, such as from an email header). This makes them of limited appeal to e-business, and expensive. There are also intrinsic limitations. If, for example, an online transaction involves the shipment of physical goods, there will be a delivery address. If the product is intangible, such as software or audio, the copyright owner may want some means of pursuing you if you distribute copies widely. So the ultimate use of digital cash technology may be in a closed application such as road tolling. Related technologies may be used to protect voters in online elections—a subject to which I'll return in the next chapter.

20.4.5 Other Applications and Issues

The control of meta-information, and applications of anonymity and deniability, surface in a number of other applications.

20.4.5.1 The Right to Remain Ignorant

One of the most difficult things to assure in automated systems, whether with the mechanisms described here or those described in Chapter 8, is the right to not know something. The classic example is that in many countries you have the right not to know the outcome of a DNA test that a relative has for an inheritable disease. Your relative does have a right to know, and he may tell others—in theory, he might tell everyone else in the world. This is not just a problem technically, but also for the data protection laws of a number of countries [741].

20.4.5.2 Location Security

In the chapter on telecoms security, I mentioned the location security mechanism in GSM—the temporary mobile subscriber identity or TMSI. This turned out to be relatively easy for the police to defeat; and in some countries the phone companies' logs of mobile users' location history were made available to the police anyway (there was a political furor in Switzerland when people realized this was happening there). Many countries, including the United States, have now passed laws or regulations demanding that this information be available on production of a warrant (or even on demand) to the police; there are further requirements such as tracking mobiles which make emergency calls. Third generation mobile services will provide location information accurate to 250 m, and in Europe at least it looks like there will be a legal requirement for phone companies to keep this for a year in case the police want it. What's more, many businesses plan to offer location-based services, with marketing pitches such as "50c off a Big Mac at the McDonalds you're about to drive past". There are even proposals for authentication schemes in which mobile terminals would only be allowed access to a system if they were in a particular area, such as within the confines of a military base [237].

Thus there appears little prospect that real location security services will be offered in public networks. There's no technical reason for this—in principle, one could use digital coins to pay for network access, and a more elaborate design is presented in [456]—but given business and regulatory pressure it's unlikely to happen. The most one can expect is that users may get medium-grade privacy from each other.

Location privacy mechanisms may, however, be fielded in embedded systems, such as road tolls in Germany, where data protection law prohibits the retention of vehicle details once the vehicle moves out of sight of the toll gantry, unless the toll has not been paid [164]. Of course, it will always be open to individuals to devise their own protection measures, as with the businessman mentioned above who randomly borrows a different mobile phone each day. However, in the absence of such extreme measures, location privacy seems set to be one of the more difficult things to achieve in the years ahead.

20.4.5.3 Peer-to-Peer and Censorship-Resistant Systems

If there were an anonymous channel that couldn't be jammed, then you could use it to send out copyrighted, blasphemous, or libellous material without getting caught. This is one of the central tensions between anonymity, copyright, censorship, and civil liberties.

An early anonymous remailer, `anon.penet.fi`, was closed down following legal action brought by the Scientologists. It had been used to post a message that upset them. This contained an affidavit by a former minister of their church, the gist of which was reported to be an allegation that once members had been fully initiated they were told that the rest of the human race was suffering from false consciousness; that in reality, Jesus was the bad guy and the Devil was the good guy. Well, history has many examples of religions that denounced their competitors as both deluded and wicked; the Scientologists' innovation was to claim that the details were their copyright. They were successful in bringing lawsuits in a number of jurisdictions.

The reaction of the Internet community has included a number of designs for distributed file stores, some of which deliberately use anonymity mechanisms to make this kind of censorship much more difficult. An early proposal was the Eternity Service, designed to provide long-term file storage by distributing file fragments across the Net, encrypted so that the people hosting them would not be able to tell which fragments they had, and reconstruction could only be performed through remailer mechanisms [27]. A modern version of this is Publius[2], which also provides a censor-resistant anonymous publishing mechanism [785]. Another successor system is Freenet which seeks to provide communications as well as file storage services [189].

But probably the most heavily used decentralized file-sharing service is Napster [570]. This enables Net users who are online to share MP3 audio files with each other. Rather than keeping the files centrally, which would invite legal action, Napster simply provides an indexing service so that someone wanting a given track can find out who else has got it and is prepared to share or trade. In addition to litigation from Hollywood, Napster has attracted perhaps 10–20 million users and a number of imitators (such as gnutella and mojonation). Given the huge volume of MP3 traffic it has generated, there will be a temptation to use this as cover traffic, and to share other data through the system. This other traffic might be encoded steganographically in MP3 files or simply wrapped so that it appears to be in MP3 format.

These two streams of development—censorship resistance and file sharing—have recently been coalescing into the new discipline of peer-to-peer networking. This burst on the scene in the middle of 2000, and is starting to encompass other issues, such as ad-hoc networks of mobile devices.

Early computer networks were many-to-one (or, as we would now call them, client-server): many terminals connected to one mainframe. The early ARPANET went to the other extreme, with each connected machine being an equal peer of the others. As the ARPANET grew into the Internet, more hierarchy and organization got added, first with services such as DNS and telnet, and later with the development of large commercial Web sites. By now, Tim Berners-Lee's vision of the Web as a person-to-person communications mechanism has been turned into a client-server model in which people's PCs act as more-or-less dumb terminals for large Web servers.

Peer-to-peer networking is often seen as a return to basics. The participating machines become equals once more, and can all communicate with each other. The driving forces are not just copyright and censorship, though. The rapid growth of the net has left DNS behind; there are now simply not enough IP addresses. Most dial-up Web users are now allocated a temporary IP address by their ISP, so applications that enable users to talk to each other, such as ICQ, have had to invent their own naming systems that function despite the intermittent connectivity of the principals.

Another reason for interest in peer-to-peer networking is the arrival of ad-hoc mobile network technologies such as Bluetooth. Within a few years, you will probably have a personal network of dozens of devices: your organizer, your mobile phone, your heart monitor, your PC at home, your burglar alarm—and some of these will make transient connections to things such as train ticket dispensers and the laser printer in a client's office. Centralised administration of such networks will be impossible (thank goodness), and the current Internet infrastructure (such as DNS) will probably not be able to cope. The way forward appears to be for principals to use many infrastructures,

[2] For non-U.S. readers: the revolutionaries Alexander Hamilton, John Jay, and James Madison used the pen name Publius when they wrote the Federalist Papers, a collection of 85 articles published in New York State newspapers in 1787–1788 and which helped convince New York voters to ratify the United States Constitution. The reference is to the U.S. right to anonymous political speech.

rather than just one, and to tailor the infrastructure to the application. Some infrastructures may be hierarchical (as with ICQ) while others may be decentralized for extreme survivability (as with censorship-resistant systems) and yet others may be transient (as with ad-hoc networks).

We have recently started to realize that many of the techniques developed for systems like Eternity and Publius, plus those developed for secure ad-hoc networking such as [731] and [732], can bring quite a lot to this party. The field is young and vigorous; a collection of papers should appear shortly as ‘Peer-to-peer: Harnessing the disruptive potential of collaborative networking’, edited by Andy Oram and published by O’Reilly. (I got an advance copy of this too late to include it in this book’s bibliography.)

20.4.5.4 Subversive Group Computing

An interesting engineering problem is the extent to which it’s possible to integrate the technologies discussed here to provide systems that are highly resistant to all kinds of survillance and coercion. This problem has been called *subversive group computing* and may be thought of as the set of technologies necessary for a subversive group—say, for example, a Tibetan group wishing to throw off Chinese rule. The threat model here involves not just pervasive surveillance and determined service denial attacks, but also the regular subversion of group members.

One can imagine a “covert superhighway” that would enable group members to communicate with each other using anonymity mechanisms; distributed file stores for propaganda that would otherwise be suppressed; steganographic file systems to help group members appear innocuous if caught; and—as a backstop—a cell mechanism to limit the damage that could be done by a group member who is “turned.” Such a hypothetical system might be thought of as a generalization of the mechanisms for enabling a group of servers to withstand and recover from an integrity failure of one of their number, such as AT&T’s Rampart and IBM’s Proactive Security, which we discussed above in 6.2.2.

There’s an obvious direct interest in such techniques not just for national liberation groups and counterintelligence agencies, but also from the point of view of public policy generally, as they will set the technical limits of both privacy and surveillance. And, if recent history is any guide, they are likely to be at least as much driven by the desire to evade, or enforce, copyright as by any particular political liberation agenda. It’s likely that these technologies will also find some wider criminal use, but as Whitfield Diffie puts it, “If you campaign for liberty, you’re likely to find yourself drinking in bad company at the wrong end of the bar.”

20.4.5.5 Abuse

Finally we turn to the more common kinds of abuse, such as spam, mail bombing, and harassment generally. In meatspace, such harassment is controlled by social pressures and at the extreme by physical confrontation: you can shut your house door in the face of an unwanted salesman. The same holds for cash transactions. If you grab a banana from a market stall and run off without paying, the stallholder can run after you.

One of the critical ways cyberspace is different is that these physical aspects of security and control are absent, and especially so if users can shelter behind unbreakable anonymity. For this reason, a number of people have proposed *identity escrow* schemes under which Net users have pseudonyms that normally provide identity protection but that can be removed by order of a court [155].

A contending view is that, given the ease with which people can get throwaway email addresses, and log on through cybercafes or via prepaid mobile phones, anonymity will continue to exist and will occasionally be a factor in abuse; and given our experience of real-world abuse so far, other mechanisms are likely to be at least as effective as tracing the perpetrators.

Unsolicited commercial email—*spam*—is our main teacher when it comes to abuse. It's routine for spammers to hide the source address of their product to prevent counter-attacks by irritated Net users. Throwaway email addresses are one technique they use; another is simple mail forgery. A number of strategies have been adopted to manage it. Many of the most effective systems use a system of *bait addresses*, which are publicized on newsgroups and mailing lists so that spammers will add them to their lists; addresses from which spam is sent to the bait addresses are then blacklisted. A particularly effective variant on this theme adds a delay of a few hours to all messages received from unknown hosts, in order to see if they turn up at a bait address meanwhile [411].

Another technique is *rate control:* an ISP may limit the number of email messages that a subscriber can send. One supplier of anonymity services, Zero Knowledge Systems, has decided against identity escrow and in favor of rate control, plus the cancellation of pseudonyms from which abusive or illegal material is sent [821]. (An interesting footnote here is that De Montfort University in Britain has blocked Zero Knowledge at its firewall on the grounds that it can no longer enforce its policy of blocking pornographic materials. A debate over academic freedom is brewing.) This doesn't provide a huge disincentive, as the nym can't be traced back to its purchaser, and he can simply buy a new one. We'll just have to see how this pans out in practice.

Spam interacts with protection mechanisms in various ways. A notable one is that spammers sometimes use remailer services to conceal themselves, so it's routine for anonymity services to implement rate control (this also makes it more difficult to attack the system by sending large volumes of traffic to a single pseudonym and trying to see where the traffic goes). Another is the *reverse mail bomb:* by forging spam (or other offensive messages) that appear to come from your victim, you cause him to be deluged with angry messages from recipients.

Finally, many of the real difficulties ISPs have in tracking down the perpetrators of abuse are not due to people hiding behind remailers, but have to do with real-world problems. Examples are the difficulty of establishing responsibility when abusive traffic comes from a phone line in a multi-occupied student house, or when someone accesses a dial-up account on a free ISP from a phone whose calling line ID has been blocked. ISPs also often keep quite inadequate logs, and hence can't trace abusive traffic later. So, in practice, as opposed to theory, anonymity is already pretty widespread [190].

Abuse—whether technical, social, or even criminal—is not going to go away, and means of reducing its impact and holding the culprits responsible will continue to be an issue. For example, the view of U.K. ISPs is that, "Anonymity should be explicitly supported by relevant tools, rather than being present as a blanket status quo, open to use and misuse" [190]. The detailed design of these tools is likely to remain contentious for some time.

20.5 Summary

Some of the most difficult security engineering problems at the beginning of the twenty-first century have to do with copyright and privacy. In the absence of affordable tamper-proof devices, will the enforcement of intellectual property rights necessarily mean detailed monitoring of who reads which book, who listens to which music, and who runs which software? If the tools are made available to enable people to prevent monitoring, will this imply large-scale piracy of copyrighted matter?

This may at first sight seem to be the modern equivalent of the old philosophers' question of what happens when an irresistible cannon ball meets an immovable post—both can't exist in the universe at the same time. But, as always, it's not that simple. The problems facing Hollywood, and the problems of defending one's private space against intrusion, are more subtle and will involve the judicious combination of a range of tools. These tools have the feature that they manage meta-information: the source of a message, its destination, whether it's been paid for, whether copying is allowed, whether it should no longer be readable after a certain date, and so on. By a choice of suitable mechanisms, some quite subtle combinations of properties can be engineered.

That's not to say that there will be no conflict between copyright and privacy—merely that governments that rush to infringe personal liberties at the behest of the film and music industries fail to understand the problems, and deserve to have their legislation overturned by the judiciary. Also, the headline nature of large-scale copyright piracy should not deflect managers and policymakers from other serious real-world abuses, such as spam and harassment generally.

This seems an appropriate note to finish the technical part of this book. Part 3 will deal with issues of policy, assurance, economics, and management.

Research Problems

There are many interesting research problems in copyright management. Some of them I've already touched on, such as whether much cheaper tamper-proof hardware tokens can be built. Others are the subject of intense activity, such as better ways of embedding copyright marks in digital pictures and sound. There are also business modelling issues, which seem to have got little attention. For example, could we reuse the work done on modelling epidemic thresholds for computer viruses to find out how much it's economic to spend on combatting various kinds of content piracy?

Privacy is also an active field of research and innovation. Perhaps the most difficult problems have to do with preventing the abuse of anonymous services, particularly by people whose goal is to close down an anonymous service by abusing it deliberately.

Further Reading

Software copy protection techniques are discussed at length in [356]; there's a brief history of technical protection mechanisms for audio and video in [307]; and a racy account of the co-evolution of attack and defense in pay-TV systems in [532]. More information about pay-TV, and the available information on DVD, can be found at various Web sites (which may move because of legal harassment); a lawyer's view can be found at [361].

There is an overview of information hiding techniques, including steganography and copyright marking, in a special issue of the Proceedings of the IEEE [515]; for attacks on marking schemes in particular, see [610, 611]. For more detail, there's a recent book on the subject [443]. Kahn is, as usual, good historical background reading [428]. The best introduction to anonymous remailers is probably [531]. Finally, ongoing research work can be found in the proceedings of the workshops on information hiding [28, 59, 613].