CHAPTER

22

Copyright and DRM

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 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

22.1 Introduction

Copyright, and digital rights management (DRM), have been among the most contentious issues of the digital age. At the political level, there is the conflict alluded to by Barlow in the above quotation. The control of information has been near the centre 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 establishment of modern copyright law starting with the Statute of Anne in 1709, through the eighteenth century battles over press censorship, to the Enlightenment and the framing of the U.S. Constitution. The link between copyright and censorship is obscured by technology from time to time, but has a habit of reappearing. Copyright mechanisms exist to keep information out of the hands of people who haven't paid for it, while censors keep information out of the hands of people who satisfy some other criterion. If ISPs are ever compelled to

install filters that will prevent their customers from downloading copyrighted content, these filters could also be used to prevent the download of seditious content.

In the last few generations, the great wealth accruing to the owners of literary copyright, films and music has created another powerful interest in control. As the music and film industries in particular feared loss of sales to digital copying, they lobbied for sweetheart laws — the DMCA in America, and a series of IP Directives in Europe — that give special legal protection to mechanisms that enforce copyright. These laws are now being used and abused for all sorts of other purposes, from taking down phishing websites to stopping people from refilling printer cartridges.

The ostensible target of these laws, though, remains the DRM mechanisms that are used in products such as Windows Media Player and Apple's iTunes to control copying of music and videos that have been purchased online. I'll describe how DRM works. The basic mechanism is to make available an encrypted media file, and then to sell separately a 'license' which is the key to the media file encrypted using a key unique to the user, plus some statements in a 'rights management language' about what the user can do with the content. I'll also describe some interesting variants such as satellite TV encryption systems, copyright marking, traitor tracing, and Blu-Ray. And, of course, no discussion of copyright would be complete these days without some mention of file-sharing systems, and the mechanisms used by Hollywood to try to close them down.

Finally, there are some thorny policy issues tied up in all this. Economists pointed out that stronger DRM would help the platform industry more than the music industry, and their warnings have come true: Apple is making more money and the music majors are making less. The possible implications for video are interesting. And finally there are some serious privacy issues with rights management systems. Do you really want a license management server, whether in Redmond or Cupertino, to know every music track you've ever listened to, and every movie you've ever watched?

22.2 Copyright

The protection of copyright has for years been 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 in many countries about 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 the UK, several million pounds a

year are distributed to authors whose books are borrowed from public lending libraries [1050]. 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's a lot we can learn from historical examples such as book publishing, and pay-TV. But I'm going to start by looking 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. Also, the music industry forced the computer industry to introduce DRM, saying that without it they'd be ruined — and the computer industry for years retorted that the music industry should just change its business model, so it's interesting to use software as the baseline. Finally, the computer industry frequently argued in its tussles with the music majors that in an open platform such as the PC it's intrinsically hard to stop people copying bitstreams — so how did they themselves cope?

22.2.1 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 1960's whereby its users could share programs they had written. (Most of them were useless as they were too specialised, too poorly documented, or just 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 bank dealing room software takes the view that anyone who pirates their 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 1960's, software costs started to become significant. Hardware vendors started to charge extra for their operating system, and third party system houses sprang up. To begin with, they mostly sold you a complete bespoke system — hardware, software and maintenance — so piracy was still not much of an issue. By the mid-1970's, some of them had turned bespoke systems into packages: software originally written for one bakery would be parametrized and sold to many bakeries. The most common 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 important, 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 recognise a passage of text by indexing it according to the least common words which appear in it [589], on to systems used by humanities scholars to figure out whether Bacon wrote Shakespeare, and back to tools which try to identify the authors of viruses from their coding style [746].)

With time, people invented lots of useful 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 as well to prepare management reports. Meanwhile, the installed base of machines got large enough for software sharing to happen more than just occasionally. So 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*. When I worked in 1981 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 license serial number, and if the caller claimed to be from that customer we'd give them a password to re-enable 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 office.

Software piracy really started to become an issue when the arrival of microcomputers in the late 1970's and early 80's created a mass market, and software houses started to ship products that didn't require need support to install and run. Initial responses varied. There was a famous open letter from Bill Gates in 1976, a year after Microsoft was founded, in which he complained that less than 10% of all microcomputer users had paid them for BASIC [502]. '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 seized on the obvious difference between minis and micros — the latter had no processor serial numbers. There were three general approaches tried: to add uniqueness on to the machine, to create uniqueness in it, or to use whatever uniqueness happened to exist already by chance.

1. 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.
2. 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 in a strange way 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*).
3. A product I worked on stored the PC's configuration — what 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 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. You can stop this by having critical code somewhere really uncopiable, such as in a dongle, but the lesson from this arms race was that the kids with the debuggers would always break your scheme eventually.

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 will 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 favourite stories was of the U.S. army bases in Germany that didn't pay for the VAX VMS operating system and got hacked after they didn't get a security patch (described above in section 2.5.4).
- 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. The driver for this was that consumers are more sensitive about the sticker price of a product than about its total cost of ownership, so it makes sense to subsidise the cost of the console out of later sales of software for it (a strategy since adopted by printer makers who subsidise the printers from the ink cartridges). This strategy led to strict *accessory control* in which hardware protection was used to prevent competitors selling software or other add-ons unless they had paid the appropriate royalty.

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 which 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 got more complex, operating system interfaces became higher level, and software protection routines of the 'bad disk sector' variety became harder to write. And now that it's possible to run a Windows NT system on top of Linux using VMware or Xen, 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 or even interfere with each other. Software protection techniques tend to make a

product less robust and cause you problems — as when their hard disk fails and they 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 non-standard operating system usage tended to set off virus alarms.
- There was not much money to be made out of harassing 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 their scare message was pretty negative.
- Many vendors preferred not to have to worry about 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 second-hand with the software installed). As both practices were common, mechanisms that made one or the other very much harder caused problems. Mechanisms that could easily deal with both (such as dongles) tended to be expensive, either to implement, or in call-centre support costs.
- 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, borowed 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 anti-piracy trade organizations in most countries (in the USA, the Software Publishers' Association) that brought high-profile prosecutions of large companies that had been condoning widespread use of pirate PC software. This was followed up by harassing 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 a 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-honoured technique of using timebombs 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 [313]. 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 licence agreements are enforced using *license servers*, which are somewhat like dongles but are implemented on PCs which sit on a corporate network and limit the number of copies of an application that can run simultaneously. They can still be defeated by disassembling the application code, but as code becomes larger this gets harder, and combined with the threat of legal action they are often adequate.

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 Bill's more revealing sayings 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 [518].

The latest emphasis is on 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. And when I first tried Microsoft Antispyware Beta on a machine we had at home with Windows XP, it was denounced as a pirate copy — despite the fact that the PC had been bought legally in a local store. Microsoft did sort that out for me, but having flaky registration mechanisms clearly costs money, and building robust ones is not trivial if you're selling large volumes of many products through complex supply chains. (In fact, it's against the public interest for security patches to be limited to registered licensees; a security-economics analysis we did of the problem recommended that the practice be outlawed [62].)

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. For example, once they got their product registration sorted out, Microsoft found that a third of the copies of Office sold in Germany were counterfeit, and traced them to a small factory a few miles up the road from us in Cambridge. Almost all the factory's staff were unaware of the scam — they believed the company was a bona fide Microsoft supplier. They were even proud of it and their salesmen used it to try to get disk duplication business from other software vendors.

That is more or less what's done with the personal and small business sectors, but with medium sized and large businesses the main risk is that fewer legal copies will be purchased than there are machines which run them. The usual countermeasure is to combine legal pressure from software trade associations with site licences 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. Many large businesses prefer not to have machines registered online individually, as they want to keep their staff numbers and structures confidential from the vendors; many vendors respect this, but the downside is that an 'unprotected' binary originally issued to a large company is often the standard 'warez' that people swap. Many firms still hold 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 include the difficulty of tracing people who change addresses or trade PCs secondhand. It is just very expensive to maintain a high-quality database of millions of small customers.

For companies that do have deep pockets, such as Google, one option is to provide not just the software but also the processor to run it. It's worth noting that software-as-a-service may be the ultimate copyright protection or DRM

for software (or any other content that can live online): you can't buy it, freeze the version you're running, or use it offline. You may also get to control all your customers' data too, giving you impressive lockin. (I will discuss web services in the next chapter.)

With that exception, none of the mass-market protection technologies available at the beginning of the 21st 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 aren't too extortionate and the vendor isn't too unpopular. Small software companies are under less pressure, as their products tend to be more specialised and the risk of copying is lower, so they can often get away with making little or no effort to control copying. (And if they have only a few customers, it may even be economic for them to supply software as a service.)

There are also many alternative business models. One is to give away a limited version of the product, and sell online a password which 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 charge companies, as Netscape did. An even more radical model is to give your software away completely free, and make your money from selling services. The Linux industry makes its money from consultancy and support, while Web applications such as Google Documents make their money from advertising.

This experience has led many computer people to believe that 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.

22.2.2 Books

Carl Shapiro and Hal Varian present a useful historical lesson in the rise of book publishing [1159]. In 1800, there were only 80,000 frequent readers in England; most of the books up till then were serious philosophical or theological tomes. After the invention of the novel, a mass market appeared 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.

22.2.3 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 afterwards. (As a result, many of his works were lost to posterity.)

In recent years, there have been several flurries of industry concern. When the cassette recorder came along in the 1960's, 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 that 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 have the bandwidth to 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 into big business, and although there was some copying, there were huge sales of pre-recorded 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 said whether a track could be copied or not [648]. The idea was that copies made from a CD would be marked as uncopiable, so people could copy 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. (CDROMs also have a no-copy bit in the track header but this is almost universally ignored.)

Audio copying has become a headline concern again, thanks to 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 squeeze an audio CD track into a few megabytes, and universal broadband enables files of this size to be shared easily. Usage in universities is particularly heavy; by 1998, some 40% of the network traffic at MIT was MP3 traffic. Some students became underground disc jockeys and relayed audio streams around campus — without paying royalties to the copyright owners.

The initial response of the industry was to push for technical fixes. This led to the growth of the rights-management industry. It had its origins in work on

digital publishing and in the mechanisms used to protect pay-TV and DVDs, so let's look at those first.

22.2.4 Video and Pay-TV

The early history of videocassettes was just like 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 which 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 being able to rent videos greatly increased the number of VCRs 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 like 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 bring prosecutions.

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 which would allow a station operator to restrict reception of a channel in various ways. If he'd only bought the rights to screen a movie in Poland, then he'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 strict censorship laws. Most operators also wanted to be able to charge extra for specific events such as boxing matches.

22.2.4.1 Typical System Architecture

The evolution of early systems was determined largely by the hardware cost of deciphering video (for a history of set-top boxes, see [293]). The first generation of systems, available since the 1970's, 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 persistence.

The second generation of systems appeared in the late 1980's and employed a hybrid of analog and digital technologies: the broadcast was analogue, the subscriber control was digital. These included systems such as Videocrypt and Nagravision, and typically have three components:

- a subscription management service at the station enciphers the outgoing video, embeds various *entitlement management messages* (EMMs) and *entitlement control messages* (ECMs) in it, and issues access tokens such as smartcards to subscribers;
- a *set-top box* converts the cable or satellite signal into one the TV can deal with. This includes descrambling it;
- the subscriber smartcard personalises the device and controls what programmes the set-top box is allowed to descramble. It does this by interpreting the ECMs and 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 custom chip with a long product life, while security-critical functions that may need to be replaced in a hurry after a hack can be sold to the customer in a low-cost token that is easy to replace. If the set-top box itself had to replaced every time the system was hacked, the economics would be much less attractive.

The basic mechanism is that the set-top box decodes the ECMs from the input datastream and passes them to the card. The card deciphers 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 then uses the control words to descramble the video and audio streams. There's a detailed description in a patent filed in 1991 by NDS [314].

22.2.4.2 Video Scrambling Techniques

Because of the limitations on the chips available at low cost in the early 1990s, hybrid systems typically scramble video by applying a transposition cipher to picture elements. A typical scheme was the *cut-and-rotate* algorithm used in

Videocrypt. 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 (Figure 22.1):

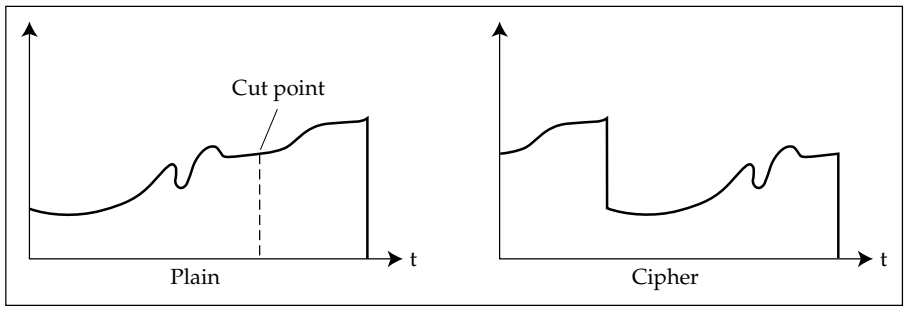


Figure 22.1: Cut-and-rotate scrambling

This involved analog-to-digital conversion of the video signal, storage in a buffer, and digital-to-analog conversion after rotation — a process which could just about be shoehorned into a low-cost custom VLSI chip by 1990. However, a systemic vulnerability of such systems is that video is highly redundant, so it may be 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 22.2 shows a frame of enciphered video, and Figure 22.3 the same frame after processing. By now, it's possible to do this on a PC [1222]. If this attack had been feasible earlier, it would have given a complete break of the system, as regardless of how well the smartcard managed the keys, the video signal could be retrieved without them. Hybrid systems are still used by some stations, particularly in less developed countries, together with frequent key changes to make life inconvenient for the pirates — whose problem is to somehow distribute the keys to their customers as they crack them.

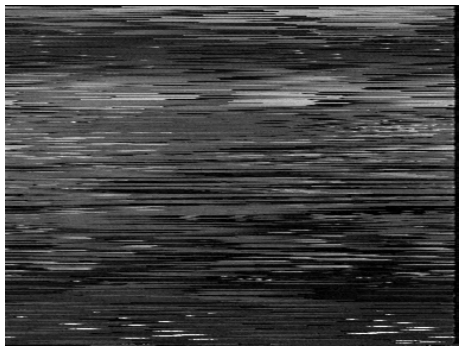


Figure 22.2: Scrambled video frame

Figure 22.3: Processed video frame

As for major developed-world operators, they moved to digital systems in the early 2000s. These digital systems work on the same principle — a set-top box with the crypto hardware and a smartcard to hold the personal keys that in turn decipher the content keys from ECMs. However the crypto now typically uses a block cipher to protect the entire digital video stream. I'll describe the current digital video broadcast systems in the next section.

The hybrid scrambling techniques lasted (just) long enough. However, they have some interesting lessons to teach, as they were subjected to quite determined attack in decade after 1995, so I'll spend a page or two going through what went wrong.

22.2.4.3 Attacks on Hybrid Scrambling Systems

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 white lists or black lists. 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 a bit over half a million a day. So 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 customer smartcard interprets the ECMs. If the current programme is one the subscriber is allowed to watch, then a keyed hash — essentially a message authentication code (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:

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

So if a subscriber stops paying his subscription, his card can be inactivated by sending an ECM ordering it to stop issuing control words; and it needs access to the ECM stream in order to compute the control words at all. So provided the cards can be made tamper-resistant, only compliant devices should have access to the master key K, and they should commit suicide on demand. So what could go wrong?

Well, the first attacks were on the protocol. 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, one person can record the stream of control words, by placing a PC between the smartcard and the set-top box, and post them to the Internet. Other people can video-record the scrambled program, and unscramble it later after downloading the control word stream [850]. Servers sprung up for this key=log attack exist, but were only a minor nuisance to the pay-TV industry; not many viewers were prepared to get a special adapter to connect their PC to their set-top box.

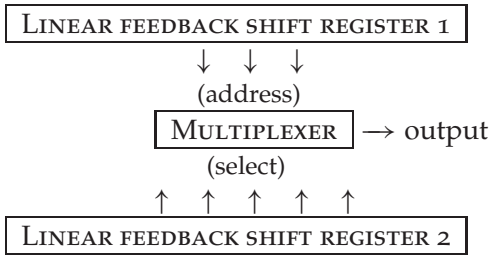


Figure 22.4: The multiplexer generator

Cryptanalysis also gave opportunities to the hackers. 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 (Figure 22.4).

The designers intended that breaking this cipher should involve guessing the key, and as this is 60 bits long a guess would take on average 2^{61} 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.) The effect of this is that as the high order four bits or so of each control word are easy to deduce from inter-line 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, and of interest to hobbyists rather than the mass market.

Other hobbyist attacks 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 [850]. Perhaps the most powerful of the 'amateur' attacks exploited a master key leakage: someone bought a second-hand PC, looked out of curiosity to see if 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. This enabled software to be written that would completely emulate a subscriber smartcard — in fact, it could be 'improved' in that it would not turn itself off when ordered to do so by an ECM.

Anyway, once this 'low-hanging fruit' had been picked, the commercial pirates turned to reverse engineering smartcards using microprobing techniques. In Chapter 16 I described the arms race between attackers and defenders. 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 took time. 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. By the middle of 1995, for example, the main UK satellite TV station (Sky-TV) was losing 5% of its revenue to pirate cards.

So 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?

The operators came up with all sorts of cunning tricks. One of their more effective ones 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.

One of the systems (Eurocrypt) had an efficient revocation scheme designed in from the start, and it's worth looking at briefly. Each of the subscriber smartcards contains a subscriber key k_i, and there is a binary tree of intermediate group keys *KGij* linking the subscriber keys to the currently active master key K_M. Each operational card knows all the group keys in the path between it and the master key, as in Figure 22.5.

In this scheme, if (say) key k_2 appears in pirate cards and has to be revoked, then the operator will send out a stream of packets that let all the other subscriber cards compute a new master key K_M. The first packet will be $\{K'_M\}_{KG12}$ which will let half the subscribers compute K'_M at once; then there will be a K'_M encrypted under an updated version of *KG*11: $\{K'_M\}_{KG'11}$; then this new group key *KG'*11 encrypted under *GK*22; and so on. The effect is that even with ten million customers the operator has to transmit less than fifty ECMs in order to do a complete key change. Of course, this isn't a complete solution: one also needs to think about how to deal with pirate cards that contain several subscriber keys, and hopefully how leaked keys can by identified without having to go to the trouble of breaking into pirate cards. But it's a useful tool in the countermeasures war.

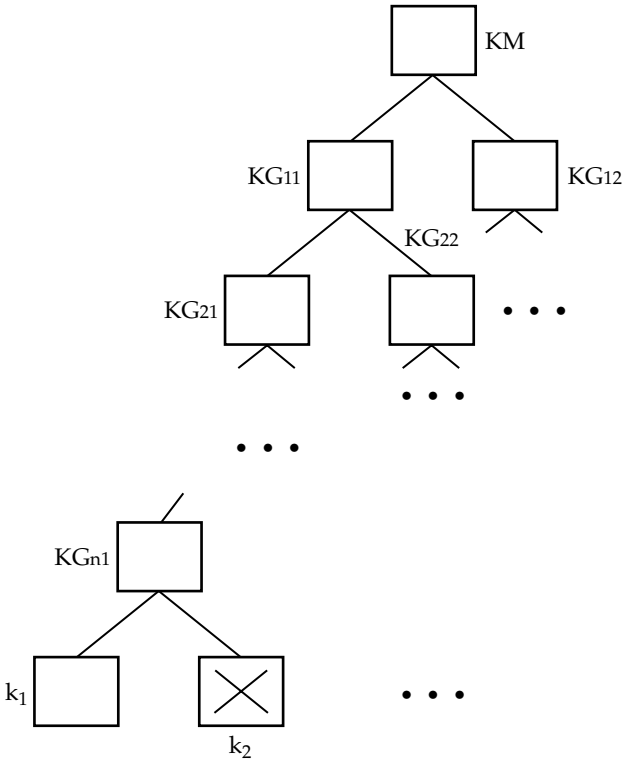


Figure 22.5: Binary revocation tree

Psychological measures were also used. For example, one cable TV station broadcast a special offer for a free T-shirt, and stopped legitimate viewers from seeing the 0800 number to call; this got them a list of the pirates' customers. Economic factors also matter here, as everywhere. 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 pirates race to market just as legitimate vendors do, and pirate cards have bugs just as Windows does. An understanding of economics helps you exploit them effectively: it's best to let a pirate build up a substantial user base before you pull the plug on him, as this gives him time to wipe out his competitors, and also as switching off his cards once he's established 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 become a persistent problem.

The pay-TV industry learned to plan in advance for security recovery, and to hide a number of features in their 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 much harder to forge by including proprietary encryption algorithms in the processor hardware. As the attacker couldn't simply read out the algorithm with a probing station but had to reverse engineer thousands of gates in the chip, they 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 who 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.

In the last big pay-TV piracy case in the 20th 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 [922]. So Sky TV's security consultants infiltrated a spy into his Dublin sales office, and she quietly photocopied enough documents to prove that the operation was really being run from the UK [645]. The British authorities didn't want to prosecute, so Sky brought a private prosecution and had him convicted. When he later escaped from jail, Sky's private detectives relentlessly hunted him down and eventually caught him in New Zealand, where he'd fled using a passport in a dead person's name [575].

So pay-TV history reinforces the lesson that one must make the engineering and legal aspects of copyright protection work together. Neither is likely to be adequate on its own.

22.2.4.4 DVB

Digital video broadcasting (DVB) largely operates using a set of standards that have evolved over the ten years since 1996 and that are controlled by the DVB Consortium, and industry group of over 250 members. The standards and many and complex, relating to IPTV and digital terrestrial TV as well as satellite TV, and to free-to-air services as well as pay-TV.

The protection mechanisms are still a work in progress, and some of them are covered by nondisclosure agreements, but here is a telegraphic summary. The conditional access mechanisms for pay-TV are similar to the hybrid system: the content encryption is digital, but the keys are generated by subscriber smartcards operating on EMMs and ECMs as before. The encryption uses the DVB Common Scrambling Algorithm, which is available only under NDA. The smartcards are not standardised (except at the interface level) so each broadcaster can use his favorite crypto tricks and suppliers; the piracy to date

seems to have involved smartcard cloning, but none of the major systems appear to have been broken since 2005 (the relentless viciousness shown by NDS in the Cary case may have had a deterrent effect).

Current standardization work focusses on Content Protection & Copy Management (CPCM), a set of mechanisms for protecting digital content after it's been descrambled and made available within the home. The aim is to enable set-top boxes and other consumer electronic devices to work together so that a customer who's bought a film from pay-TV can store it on a personal video recorder or PC and watch it later at his leisure. (At present, pay-TV operators that offer this do it by selling a more expensive set-top box that contains an extra tuner and a built-in PVR). The basic idea is that all the CPCM-compliant devices in a home will join an 'authorized domain' within which media can be shared and usage information will be logged. This work is still at the proof-of-concept stage. Established DRM vendors, such as Microsoft, already have mechanisms whereby protected content can be moved between compliant devices within their proprietary ecosystem, and there are strong economic incentives on DRM vendors to keep their systems incompatible, so as to maximise the lockin and thus the switching costs. There are also incentive issues with suppliers: how do you stop each individual equipment vendor from making his protection as weak as he can get away with, so that the resulting 'race to the bottom' undermines protection completely?

A good example of 'how not to do it' comes from the world of DVD.

22.2.5 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 built in at the last minute; arguments over this held up the launch of DVD and it was designed in a rush. (The story of how the DVD standards evolved is told in Jim Taylor's standard reference [1242], which also describes most of them.)

DVD has *region coding*: it divides the world into five regions, and disks are supposed to run only on players from some designated list of regions. The goal was to support the traditional practice of releasing a movie in the USA first, then in Europe and so on, in order to minimise the cost of producing physical film prints for use in movie theatres, and the financial loss if the film bombs. But a strong implementation of region coding was not in the vendors' interests; it became clear that users preferred to buy DVD players in which region coding could be turned off, so the vendors raced to ensure that everyone knew how to turn it off in their products. Region coding is less important now, as the Internet has pushed the studios towards near-simultaneous global

release of films; but the failure of region coding is yet another example of what happens when the people who have to implement some protection measure are not the people who suffer the costs of failure.

This left CSS, which was known to be vulnerable by the time that DVD was launched [1004]. I heard that the designers were 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. expert regulations; another story was that they only ever wanted to compel DVD player manufacturers to licence the CSS patent, so that makers could be compelled to implement other copy protection mechanisms as a condition of the license [193]. 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 the subject of much litigation, with numerous injunctions issued in the USA against web sites that have published it. (In fact the hated Digital Millennium Copyright Act, under which U.S. actions are often brought, was introduced in anticipation of the DVD to reinforce its technical protection.)

The flood of legal threats was counterproductive, as Hollywood just got everybody's back up. In 1999, a California court enjoined the posting of DeCSS, one of the CSS decryption programs, but not links pointing to implementations of it; a New York court added a prohibition of links in a case against the hacker magazine 2600 the next year. Such injunctions were seen as censorship, and software to decrypt CSS started appearing on websites outside the USA, on T-shirts, in songs, and in other forms of speech that more traditionally enjoy constitutional protection. This just got the software distributed ever more widely, and made Hollywood look foolish [772]. Their lawyers blundered on, persuading the government of Norway to prosecute a teenager, Jon Lech Johansen, who was one of the authors of DeCSS. He released the code in October 1999; was acquitted on appeal in 2003; and finally in 2004 the Norwegian government decided not to attempt a retrial.

Here's an abbreviated description of CSS[1]. It is based on a stream cipher similar to that in Figure 22.4 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 [1093]; 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}.

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 manufacturer keys

[1]Lawyers note: this was published in the first edition of this book in 2001.

(409 of them in 1999) *kmi*: $\{kd\}_{km1}$, $\{kd\}_{km2}$, $\{kd\}_{km3}$, . . . , $\{kd\}_{km409}$. 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.

The DVD consortium 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-produced consumer electronics just don't allow a few dollars more for a tamper-resistant processor. In effect, CSS contravened Kerckhoffs' principle, in that it depended for its protection on remaining secret, which was never realistic. I also don't think it was realistic for the consortium to think it could blacklist a large electronics firm, as this would not only have sparked off vicious litigation; it would also have meant that the millions of honest consumers who'd bought that company's products would then find they had to go out and buy a new DVD player. The outcry would have been too much; had this nuclear button ever been pressed, I expect governments could have rushed through laws demanding that all DVDs have all master keys. (I'll discuss later how the industry hopes to manage revocation better with the new Blu-Ray standard.)

Another set of problems came from the fact that the PC is an open platform. The DVD consortium required people producing DVD player software to obfuscate their code so that it would be hard to reverse engineer. Papers duly appeared on tricks for systematic software obfuscation [96]. 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 came from Linux, the open source PC operating system used by millions of people. The DVD consortium's philosophy and architecture were not consistent with making DVD drivers available to the Linux community. So as PCs with CD drives started being replaced in the shops with PCs fitted with DVD drives, the Linux user community either had to break CSS, or give up using Linux in favour of Windows. Under the circumstances, it was only a matter of time before someone figured out CSS and DeCSS appeared.

Anyway, DVD followed the usual pattern: Hollywood terrified, and refusing to release their best movies; technical measures taken to prevent copying, which got broken; then litigation. I wrote in 2001: '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'. This has come true; although some studios held out for a year or two, they all climbed on the DVD bandwagon within a few years, and Disney now makes most of its money from DVD sales. Peer-to-peer sharing is an issue, but because of the bandwidth and disk space constraints it's less so for films than with music.

So will we see the same pattern repeated with high-definition video? Let's look next at the new generation of DVD formats which are being introduced in the hope of grabbing the market for recordings made for high-definition TV, as well as for longer video recordings and larger mass storage of other data.

22.2.6 HD-DVD and Blu-ray

As I was writing this chapter in 2007, a format war was raging between two proposed successors to DVDs, each backed by a large industrial consortium. HD-DVD and Blu-ray are in many respects similar; they both use shorter wavelength lasers to encode information more densely than an old-fashioned DVD does, so a standard disk will store 25 Gb and a double-layered one 50 Gb. Both of them use a content encryption system called AACS, while Blu-ray adds an interesting extra mechanism called SPDC. (As I was correcting the proofs in early 2008, it became clear that Blu-ray had won.)

22.2.6.1 AACS — Broadcast Encryption and Traitor Tracing

The Advanced Access Content System (AACS) has an open design, documented in [647]. Its design goals included the provision of robust, renewable protection on both embedded and general-purpose computers so as to limit output and recording of protected material to approved methods. The encryption is done using AES.

Key management is based on a *broadcast encryption scheme*. The basic idea, due to Li Gong and David Wheeler, is that you can give each user a number of different keys, chosen from a large pool, and arrange things so that any two of them will find some subset of keys that they have in common [540]. So a large number of people can talk to each other without needing a unique key for every other user (or public-key certificates, or a Kerberos server). Amos Fiat and Moni Naor applied this to pay-TV: the broadcaster gives each decoder a small set of keys from the pool, and sets up content keys so that each decoder can compute them using a subset of its keys [469]. When a decoder's compromised, its keys are no longer used, and many researchers have worked on optimising things (so that decoders don't need to hold too many keys, and quite a few of them can be revoked before key material has to be updated).

AACS gives each decoder 256 device keys, and information bundled with the disk tells the decoder which keys to use and how in order to create a *Processing Key*. The idea was that each processing key would be used only for a set of disks. The mechanism is a *Media Key Block* that tells the decoder how to

combine its existing media keys to get the Processing Key. The goal is to be able to remove single devices, as with the revocation tree in Figure 22.5, although the details are somewhat different; AACS basically uses a subset-difference tree that enables each decoder to arrive at the same result (for details, see the specification [647]). When a decoder is revoked, a new MKB can be distributed which won't work for that decoder. The processing key in turn protects a Volume Unique Key (VUK), that in turn protects a title key, that encrypts the content.

This creates a less catastrophic way to revoke a player whose keys have been incorporated in unlicensed ripping software. While with a traditional DVD the studios would have had to revoke all players by that manufacturer, AACS lets them revoke an individual machine; the broadcast encryption scheme lets them target one player out of billions, rather than one vendor out of several hundred.

Has it worked? Well, at present a steady stream of keys are being extracted from software DVD players, essentially by reading them from memory, and published. To begin with, these were VUKs, and that was bad enough: once a VUK is known, anyone can decrypt that disk's content. But recently processing keys have been published too. In theory, publishers should have used many different processing keys, but it now appears that many were using a single processing key for all disks. When it's compromised, they change it, but hackers don't seem to have too much difficulty digging out the processing keys from DVD player software — and each of these lets people decrypt all the content published up till the next key change. There has been the usual circus of lawyers issuing threatening letters, injunctions and takedown notices to websites where processing keys appear, while people who don't like Hollywood have delighted in publishing these keys in all sorts of innovative ways. The first processing key to be published appeared on T-shirts and registered in domain names; it appeared on Digg.com, and when the administrators removed it, Digg's users revolted and forced them to climb down. It's been déja vu all over again.

So broadcast encryption, although a neat idea, doesn't seem to have been enough on its own to stop people decrypting movies.

There are features in AACS that don't seem to have been used yet. For example, there's a *traitor tracing* scheme. The idea behind traitor tracing is that you add some unique marks to each copy, so if decrypted content fetches up on a peer-to-peer network, the studios know who to sue. Such methods have been used with the electronic distribution of films to movie theatres, where the goal is to identify theatres that let the pirates record new films on camcorders. Another application is in the Oscars, where Academy members are given 'screeners' — DVDs of candidate films. These used to leak regularly; but in 2004, after the studios started to individualize them, the actor Carmine Caridi was ordered to pay $300,000 after forensics identified him as the source

of leaked screeners that ended up providing masters for illegal copies in 21 countries [1250]. Since then, Academy members have been more careful.

With a small distribution, it's possible to mark copies at production, but this is not feasible when mass-producing disks. A method is therefore needed to add uniqueness during the decryption process. How AACS supports this is as follows. Each cell (time segment) is usually decrypted by a title key, but issuers can insert a *sequence key cell* — of which there will be up to eight, each encrypted with a different segment key, and there can be up to 32 sequence key cells in a disk. The key management mechanism is arranged so that different players will derive different sets of segment keys. So if decrypted content appears somewhere, the content owner can see which segment keys were used and from that work out which player was used. The actual marking of each segment might be something overt from the production process, such as the color of shirt an actor's wearing, or more likely a robust hidden mark (which I'll describe later). However, the traitor-tracing mechanisms in AACS do not seem to have been deployed so far.

22.2.6.2 Blu-ray and SPDC

So the content protection in HD-DVD was starting to look slightly shaky, which helped the competing format, Blu-ray, as it also supports a novel protection mechanism called *Self-Protecting Digital Content* (SPDC).

SPDC is a very neat idea that tries to tackle the underlying incentive problem — that while it's the studios that want to protect the content, it's the equipment makers that provide the protection. The innovation is that each player contains a virtual machine that can run content-protection code that's unique to each title. The studios write this code and can change it from one disk to the next, or even between versions of the same disk. This gives the studios a number of options for responding to a compromise, and they can do so directly rather than having to work through trade associations, standards bodies or the vendors themselves.

If a particular player is seen to be leaking content, they can blacklist it in future discs; if a particular type of player is leaking, they can arrange that it only outputs low-resolution video. When it comes to tracing they can do their own; content can be marked after decoding. This not only avoids relying on the AACS mechanisms to identify compromised players or player types, but is potentially much more flexible as marks can be embedded much more pervasively in audio and video (I'll discuss marking technology below). For these reasons, SPDC provides more resilient protection than the key-management and device-revocation mechanisms in AACS alone; it's described in an evaluation report [637]. As of the beginning of 2008, the studios have been struggling with teething problems with movies using SPDC, but have decided to favor it; it now looks like SPDC has helped Blu-ray win the format war.

There are two engineering challenges facing rights-management engineers. First, people increasingly expect to move content they've purchased from one device to another. If you buy and download the latest recording of Mahler's fifth, you expect to play it not just on your laptop and iPod, but also on your home audio system; and if you buy and download a baseball game, you'll want to see it on your wide-screen TV. Second, as you make content, and the keys that protect it, available on more and more platforms, so the risk of leakage rises — and the incentives can potentially drift ever more out of alignment.

In any case, as more and more digital content is distributed online using DRM, we need to look at that next.

22.3 General Platforms

In the mid-1990s, a number of researchers started work on ways to control the sale and distribution of digital goods over the Internet to customers with personal computers. The original applications included the distribution of newspapers and articles from scientific journals [221], although it was always understood that music and video would follow once networks had enough bandwidth.

The basic problem is that a PC, being a general-purpose computer, can in principle copy any file and send it to any other computer; unlike with analogue copying, copies are perfect, so millions of copies might be made from one original. The problem is compounded by the fact that, from the viewpoint of the content vendor, the PC owner is the 'enemy'. The music industry believed that unlimited copying would destroy their business; the computer industry told them that DRM was intrinsically impossible on a general-purpose computer, so they'd better get a new business model. The music and film industries, despite being a tenth of the computer industry's size, had much more clout in Congress (a Microsoft guy told me this was because the average Congressman was much keener to be photographed with Madonna than with Bill). So Hollywood got its way. The result is a number of products generally referred to as *digital rights management* or DRM.

Curiously, despite the acrimony of the argument between computer people and Hollywood in the 1990s about whether DRM was possible or desirable, it now seems that both of them may have been taking the wrong side in the argument! Stronger DRM has turned out to benefit the platform vendors, such as Apple, more than the content companies (and economists had predicted this). And although all DRM mechanisms seem to get broken sooner or later, the existence of a modest bar to copying does have effects on the business. But as download sites move to selling unprotected MP3s, it's not clear that it's strictly necessary. However, I'll come back to the policy issues later. First let's look at the most common DRM mechanism, Windows Media Player.

22.3.1 Windows Media Rights Management

At the time of writing, Windows Media Player (WMP) comes bundled with every copy of Windows sold on general-purpose PCs. This will change, as the European Court has found the bundling to be an anticompetitive practice and ordered Microsoft to make Windows available without it. However, people will still be able to download it for free, and given the large number of websites that use Windows Media file formats, it is bound to remain an important platform.

WMP replaced an earlier media player application when Windows 98 was released. It enables a user to play music, watch video and view photos, and has all sorts of features from MP3 player support to synchronisation of lyrics for Karaoke. However our main interest here is its ability to play files protected using *Windows Media Rights Management* (WMRM). This works as follows.

A store wanting to sell digital media encrypts each item using a content key and puts the encrypted files on a streaming media server that is typically linked to their web site. In order to access a media object, the customer must get hold of a license, which consists of the object identifier, the license key seed, and a set of instructions in a *rights management language* which state what she can do with it; how many times she may play it, whether she's allowed to burn it to a CD, and so on. The license is generated by a license server and encrypted using her public key. The license acquisition may involve her going to a registration or payment page, or it may happen silently in the background [1049].

In order to use this system, the customer must first personalize her media player, which is done automatically the first time she accesses protected content. The player generates a public/private keypair (WMP uses elliptic curve cryptography) and the public key is sent to the license server.

The architecture is very similar to pay-TV conditional access, in that the bulk encryption task of protecting the music or video is separated from the personal task of key management, so the video doesn't have to be encrypted anew for each customer. And just as pay-TV smartcards can be replaced when keys are leaked or the key management mechanism compromised, so the key management functions of WMRM are performed in an 'individualized blackbox' (IBX) component of the software, which gets replaced as needed.

The IBX internals are not documented publicly by Microsoft, but according to a description by people who reverse-engineered WMP version 2 in 2001, the basic components are elliptic curve cryptography for public-key encryption; DES, RC4 and a proprietary cipher for symmetric crypto; and SHA-1 for hashing [1141]. The customer's private key is obscured by the blackbox and hidden in a file. Licenses the customer has previously acquired are kept in a license store; when a new object is encountered, the software looks up the object ID and key ID, and if a license isn't already stored for them, it sends a

session key encrypted under the license server's public key. The session key is used to encrypt the license that's sent back. Once the client software has used the session key to decrypt the license, it's added to the license store. Stored licenses have a further layer of encryption, presumably to stop people reading clear keys or manipulating permissions: the content key is encrypted using the customer's public key, and there's also a digital signature. By now, of course, the protocol may have been tweaked — as Microsoft has had to recover several times from hacks, and because WMP has now acquired many more features.

WMRM is used at its most basic to provide a streaming media service, to make it slightly harder for people to record music and video from news sites and Internet radio. (Vendors seem to have given up on audio, as it's very easy to record audio after decryption as it's passed to the sound card; but Microsoft is making a serious effort in Vista to stop people grabbing video in this way [570].) More sophisticated uses of WMRM include music subscription services, where you can download as many songs as you wish, but they will all become unplayable if you stop paying; and geographically-linked services, such as MLB.com which makes Major League baseball games available everywhere except in the team's home area — for which the rights have usually been sold for megabucks to local TV stations.

22.3.2 Other Online Rights-Management Systems

The Microsoft offering is fairly typical of rights-management systems. Apple's FairPlay, which is used in the iPod and in its media player QuickTime, also has tunes encrypted under master keys, and when a tune is bought the user is sent the master key encrypted under a random session key, plus the session key encrypted under his iTunes player's RSA public key. Session keys are backed up online on Apple's servers. As with Windows, a number of programs appeared that unlocked protected content, and Apply duly upgraded iTunes to stop these programs working in September 2006.

In the mid-2000s there were growing calls from consumer and industry groups for DRM systems to become more interoperable. Real Networks is the other major supplier of media player software for PCs, and its RealPlayer uses its own proprietary DRM. In a tussle reminiscent of the word-processing file-format wars of the 1980s, Real made their music tracks readable by iTunes in 2004; Apple responded by threatening litigation, and ended the compatibility in a 2006 upgrade. This should surprise no-one: recall the discussion in section 7.3.2 of how the value of a software company depends on the total lock-in of its customers. In 2007, Steve Jobs of Apple called on the music industry to sell music without DRM; this was surprising, given the incentives facing Apple. However, the gamble seems to be paying off, in that some music labels are now starting to make unprotected music available, but at higher prices and not on a large enough scale to threaten Apple's lock-in of its iTunes

customer base. Apple's revenues continue to soar, while the music majors see sales tumbling.

The Open Mobile Alliance (OMA) is a consortium that promotes standards for interoperable DRM on mobile devices. Its OMA DRM Version 1.0 is widely used to protect mobile phone ringtones; it mostly consists of a simple rights management language whose most important feature is 'forward lock': a way of marking content so that the phone knows it must not forward the object to any other device. Download protection is typically using standard TLS mechanisms. Version 2.0 is similar in its expressive power to WMRM or FairPlay, but is still not widely used. Network operators have objected to the idea that anyone else should have direct access to their customer base.

22.3.3 Peer-to-Peer Systems

Peer-to-peer file-sharing has become one of the main ways in which music is distributed online. After Napster's initial music-sharing service was closed down by music industry lawyers, systems such as Gnutella and Freenet borrowed ideas from the world of censorship-resistant systems to set up networks with no central node that could be closed down using legal attacks.

I was the designer of an early censorship-resistant system, the Eternity Service. I had been alarmed when an early anonymous remailer, `anon.penet.fi`, was closed down following legal action brought by the Scientologists [590]. 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 Lucifer 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 Eternity Service was 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 so that reconstruction could only be performed via remailer mechanisms [41]. A later version of this was Publius[2], which also provided a censor-resistant anonymous publishing mechanism [1309].

In 1999, Shawn Fanning, a 18-year-old drop-out, revolutionised the music business by creating the Napster service, which enabled people to share MP3

[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 1787–8 and which helped convince New York voters to ratify the United States constitution.

audio files with each other [931]. Rather than keeping the files centrally, which would invite legal action, Napster just provided an index so that someone wanting a given track can find out who else has got it and is prepared to share or trade. It attracted tens of millions of users, and then lawsuits from Hollywood that closed it down in 2001. The gap that it left behind was promptly filled by peer-to-peer networks such as Gnutella and Freenet [297] that were in turn inspired by the earlier censorship resistant systems. These were followed by commercial systems such as Kazaa and eMule, which were also targeted by music industry lawyers (and many of which acquired a reputation for being full of spyware).

The United States Copyright Office defines peer-to-peer networks as networks where computers are linked to one another directly rather than through a central server. The absence of a server that can be closed down by court order creates an interesting problem for music industry enforcers. They have tried persuasion, by playing up stories of people who'd carelessly set a P2P program to share all their hard disk, rather than just their music files. But the two tactics on which the music industry has relied are suing uploaders and technical attacks on the systems.

In section 21.5 I explained how social networks are often vulnerable to decapitation attack, as they rely for their connectivity on a small number of particularly well-connected nodes. Taking down these nodes will disconnect the network. Similarly, peer-to-peer networks have well-connected nodes, and in filesharing systems there are also nodes whose owners contribute more than their share of the uploaded content. The music industry has for some years been targeting key nodes for legal action, having filed over 20,000 lawsuits since 2003. In many cases people agree to cease and desist and pay a small penalty rather than fight a case; but in October 2007 a federal jury in Duluth, MN., convicted 30-year-old Jammie Thomas of copyright infringement for sharing material on Kazaa and ordered her to pay $9,250 for each of the 24 songs involved in the case.

On the technical-countermeasures side, it was long known that firms working for the music industry were uploading damaged music files to spam out systems (which will usually be legal), and it was suspected that they were also conducting denial-of-service attacks (which in many jurisdictions isn't). But there was no evidence. Then in September 2007, a company called Media Defender that worked for the music industry on 'file-sharing mitigation' suffered the embarrassment of having several thousand of its internal emails leaked, after an employee forwarded his email to Gmail and his password was compromised. The emails not only disclosed many details of the 'mitigation' tactics, but also revealed that the company worked closely with the New York Attorney General's office — potentially embarrassing in view of the industry's possibly illegal attacks on computers and invasions of privacy. It turned out that Media Defender's business model was to charge $4,000 per album per

month, and $2,000 per track per month, for 'protection' that involved attacks on twelve million users of fifteen P2P networks [1009].

Peer-to-peer systems have also allegedly been attacked by Comcast, which is said to have disrupted its customers' use of BitTorrent by sending forged reset packets to tear down connections. Comcast might prefer its customers to watch TV over its cable network, so they see its ads, rather than via filesharing; but the allegations raise some public policy issues if true: BitTorrent partners with content owners such as Fox and MGM, while Comcast is not a law-enforcement agency [155]. In any case, this harassment of file sharers is fuelling an arms race; the proportion of BitTorrent traffic that's encrypted rose from 4% to 40% during 2006–7 [1346].

22.3.4 Rights Management of Semiconductor IP

Another live problem is the protection of designs licensed for use in semiconductors. Companies like ARM make their living by designing processors and other components that they sell to firms making custom chips, whether by designing application-specific integrated circuits (ASICs) or by using Field-Programmable Gate Arrays (FPGAs).

The first problem is overrun production. A camera company licenses a circuit that they integrate into a bitstream that's loaded into an FPGA, that then becomes a key component in a new camera that they have made in a factory in China. They pay for 100,000 licenses, yet 200,000 cameras arrive on the market. There are two failure modes: the camera company could have ordered the extra production and lied to the IP owner, or the Chinese factory could be cheating the camera company. In fact, they could both be cheating, each having decided to make an extra 50,000 units. Now there are technical mechanisms that the camera company could use, such as personalising each camera with a serial number and so on after manufacture — but these could make it harder to cheat.

The second problem is knowing when a product contains a particular circuit. The camera company might have licensed a processor, or a filter, for one model, then built it into another cheaper model too without declaring it.

These risks cause a partial market failure, in that large IP vendors often prefer to license their best designs only to other large firms that they trust, so small startups can find it difficult to compete on equal terms [1348]. They also depress sales of FPGAs, whose manufacturers would dearly love a rights management system for design IP. The best that's been done so far are mechanisms to tackle the first problem by distributing encrypted bitstreams and updates for whole chips; the second problem is a lot harder, because the chip design tools would be squarely within the trust boundary. Customers would need to be able to evaluate designs, and debug designs, while maintaining some control on dissemination. At present, the best that can often be done is to

use side-channels for forensics. Owners of semiconductor IP can buy up samples of suspect goods, operate them, and observe the chips' precise power consumption, electromagnetic emissions, thermal signature and so on, which can often reveal the presence of a given functional component.

This brings us to the question of copyright marking, which has blossomed into a large and complex research area.

22.4 Information Hiding

Hollywood's interest in finding new mechanisms for protecting copyright came together in the mid-1990's 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 which 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. For example, when you download an MP3 file from Apple's iTunes music store, it contains a fingerprint embedded in the audio that identifies you. The idea is that if you then upload your copy to a file-sharing system, the copyright owner can sue you. (This isn't universal: some people believe that fingerprinting depresses sales overall because of the legal hazards it creates for honest purchasers. Amazon, for example, does not mark MP3 downloads [577].)

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. The conceptual model, proposed by Gus Simmons [1169, 1176], 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, and 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 focussed 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 that 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 DRM systems have control tags bundled quite visibly with the content. In many cases this will be the appropriate technology. However, in what follows we'll concentrate on hidden copyright marks.

22.4.1 Watermarks and Copy Generation Management

The DVD consortium became concerned that digital video or audio could be decoded to analog format and then redistributed (the so-called 'analog hole'). They set out to invent a *copy generation management system* that would work even with analog signals. The idea was that a video or music track might be unmarked, or 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 programmes, but could not easily be abused for commercial piracy.

The proposed mechanisms depended on hiding one or more copyright marks in the content, and are reviewed in [193, 800]. 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 only record a marked track if given X, in which case only a $h(X)$ is written to the new disc. In this way, a 'copy once only' track in the original medium becomes a 'copy no more' track in the new medium. DVD-audio uses such a marking mechanism; SDMI also uses a *fragile watermark* that's damaged by unauthorised processing. (I'll discuss mark removal techniques in the next section.)

The main use of marking with video content will be if Blu-ray wins the standards war with HD-DVD. Then the SPDC processor will be able to embed unique fingerprints into decrypted content in real time. Each plaintext copy of a video derived from a decoder can be unique, and the studios can use forensic techniques to determine which decoder it came from. Unless the decoder itself

is compromised, the marking mechanism can give a much more resilient way of identifying subscribers who're leaking content. This raises a number of questions. First, how well does marking work? Second, how well does it scale? And third, what about the policy aspects?

Quality brings us back to our old friend, the ROC. It's not enough for a marking mechanism, whether used for copy management or for traitor tracing, to have a low missed alarm rate; it needs a low false alarm rate [892] too. If your legitimate DVD player were to detect a 'no-copy' mark in your wedding video by mistake, then you'd have to buy a pirate player to watch it. So what sort of marks are possible, and how robust are they against forgery, spoofing and other attacks?

22.4.2 General Information Hiding Techniques

Information hiding goes back even further than cryptology, having its roots in camouflage. Herodotus 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 [595]. Francis Bacon proposed a system which embedded a binary message in a book at one bit per letter by alternating between two different fonts [1016]. Until quite modern times, most writers considered hiding confidential information much more important than enciphering it [1345]. 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 low-probability-of-intercept radios.

When it comes to hiding data in other data, the modern terminology of the subject is as follows [1023]. 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 has been proposed.

- Many people have proposed hiding mark or secret message in the least significant bits of an audio or video signal. This 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 better technique is to hide the mark at a location determined by a secret key. This was first invented in classical China. The sender and receiver had copies of a paper mask with holes cut out of it at random 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 16th century by the Italian mathematician Cardan and is now known to cryptographers as the Cardan grille [676].
- A modern version of this hides a mark 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 [654]. However, if all the pixels are tweaked in this way, then again 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, a different 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 [160, 562]. You can think of this as like differential power analysis: the key tells you how to sort your input data into two piles, and if the key was right they're noticeably different.
- 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 [1251]. You have a number of secret sequences, each coding a particular symbol, and you add one of them to the content to mark it.
- 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 emphasises the encoding in the noisiest or

perceptually most significant parts of the image or music track, where it will be least obtrusive, and de-emphasises it in quiet passages of music or large expanses of color [208].

- 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 [221], or adding extra echoes to music below the threshold of perception [160]. So far, such techniques don't seem to have become as robust, or as widely used, 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 late 1990s: people invented marking schemes which other people broke, until eventually the technology became more mature and robust.

22.4.3 Attacks on Copyright Marking Schemes

Throughout this book, we've seen attacks on cryptographic systems that occasionally involved cryptanalysis but more often relied on mistaken assumptions, protecting the wrong things, protocol failures and implementation bugs. And in the history of technology as a whole, inventions tend to end up being used to solve problems somewhat different from the problems the inventor was originally thinking about. Copyright marking has been no different on either count.

- In the beginning, many people assumed that the main market would be embedding hidden copyright messages so that ownership of a work could be proved in court. This was mistaken. Intellectual property lawyers almost never have any difficulty in proving ownership of an exhibit; they don't rely on technical measures which might confuse a jury, but on documents such as contracts with bands and model release forms.
- 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 which serves to identify which customer re-sold protected content in violation of his licence, and a second mark with a unique key that can be disclosed in court when he's prosecuted.

- 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, you can 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. Even if this isn't possible — say the mark is applied in a tamper-resistant processor immediately after decryption and every device adds a different mark — then if the mark consists of small signals added at discrete points in the content, an opponent can just decrypt the content with several different devices and compare them to remove the marks.
- There have been various attempts to develop a marking equivalent of public key cryptography, so that (for example) anyone could insert a mark which 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 [334]. 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 [1015, 803]. Hence the drive to have mechanisms that enable you to put a different mark in each instance of the content, as the content is decrypted.
- Some neat steganalysis techniques were developed to break particular embedding schemes. For example, where the mark was added by either increasing or decreasing the luminosity of the image by a small fixed amount, the 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 [826].
- The first large vendor of marking systems — Digimarc — set up a service to track intellectual property on the web. They supplied tools to let picture owners embed invisible fingerprints, but their initial implementation 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 which crawled the web looking for marked pictures and reporting them to the copyright owner, but there were many ways to defeat this. 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 will together look just like the original when displayed on a web page but in which a copyright mark won't be detected (Figure 22.6) [1019].

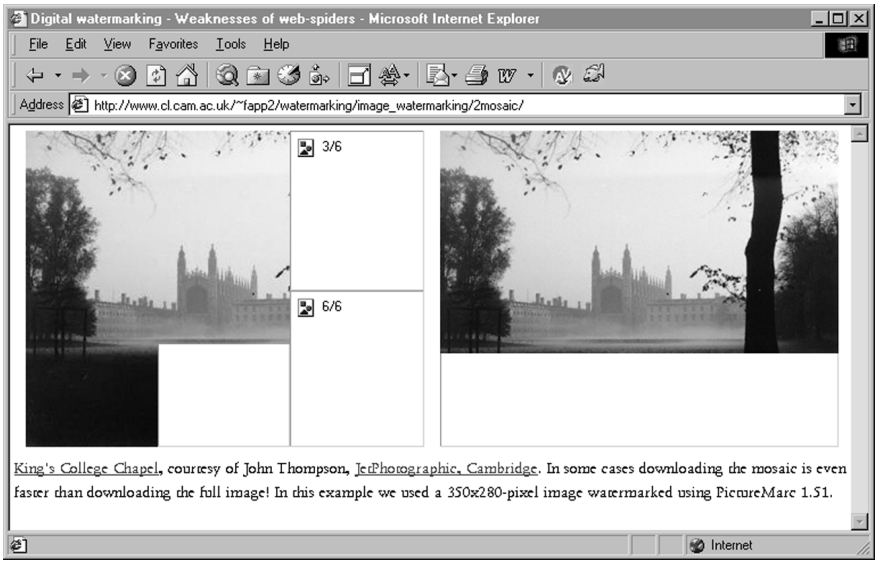


Figure 22.6: The Mosaic attack (courtesy Jet Photographic, `www.jetphotographic.com`)

Digimarc has now fixed the bugs and concentrate on monitoring broadcast streams; this enables advertisers, for example, to check whether the ads they've paid for have actually gone out. But they've found that a larger business is to move digital technology into the world of security printing: they put watermarks in ID documents to prevent them being copied, and licensed their marking technology to central banks as a counterfeit detection measure. For example, it's found in the new Euro banknotes, which it prevents from being scanned or copied using the latest equipment [1373]. Software packages such as Photoshop and Paintshop Pro now refuse to handle marked images.

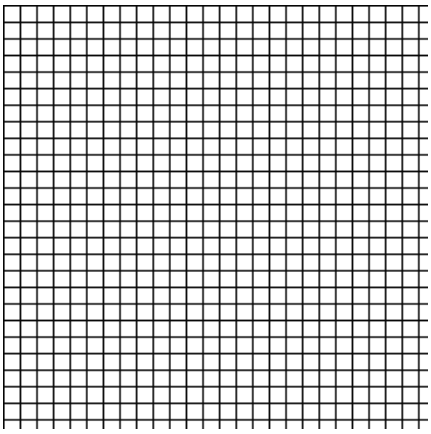
- However, the most general known attacks on copyright marking schemes involve suitably chosen distortions. Audio marks can be removed by randomly duplicating or deleting sound samples to introduce inaudible jitter; techniques used for click removal and resampling are also powerful mark removers. For images, a tool my students developed, called Stirmark, 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 22.7). This defeated almost all the marking schemes in existence when it was developed and is now a standard benchmark for copyright mark robustness [1019]. 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.

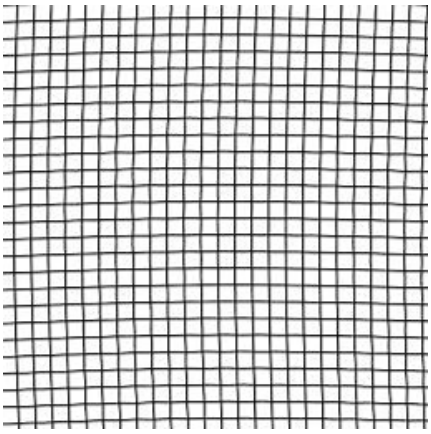
For a fuller account of attacks on copyright marking schemes, see [1019, 1020]. The technology's improving slowly but the limiting factor appears to be the difficulty of designing marking schemes that remain robust once the mark detection algorithm is known. If any copy control scheme based on marking 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.

(a) Picture of Lena

(b) Lena after Stirmark



(c) Underlying grid



(d) Grid after Stirmark

Figure 22.7: *The effect of Stirmark*

The security-by-obscurity issue has led to at least one political row. The Digimarc software used in commercial graphics packages to prevent them loading images of currency is only available under NDA. Thus proposed legislation to compel its use will place open-source graphics packages such as Gimp in danger.

22.4.4 Applications of Copyright Marking Schemes

The applications of marking techniques are much broader than just DVDs and banknotes. Color copiers sold in the U.S. have hidden their serial number in the bit patterns of copies, for example, as an extra means of detecting currency forgers [1331]. Another technique is to embed fragile marks in still pictures as they are taken, so that alterations are readily detected by forensic labs [769].

A further class of proposed applications have to do with convenience or safety rather than preventing malicious behaviour. It's 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 might be solved by embedding patient details directly in the image.

Finally, a fair proportion 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. I'll discuss such applications in the next chapter.

22.5 Policy

The IP policy debate got heated in the 1990s are early 2000s, as a series of laws from copyright term extension to America's Digital Millennium Copyright Act (DMCA) shifted power to the owners of 'intellectual property' — copyrights, patents and trademarks — in ways that many people in the computer industry and elsewhere felt to be threatening. Stricter copyright enforcement impinges on free-speech rights, for example: the law used to provide 'fair use' or 'fair dealing' exemptions that enable you to parody or criticise a work by someone else, but these exemptions weren't respected in all countries during the rush to update copyright laws. In addition, the DMCA gives copyright owners the power ('Notice and Take Down') to compel ISPs to take down websites with infringing material. Although there is also a provision ('Notice and Put Back') for the subscriber to file a counter notice and have his stuff put back within 14 days unless the copyright owner files suit, in practice many ISPs will just terminate a customer's service rather than get involved in litigation.

There are continuing concerns about the effect of DMCA on libraries, especially as more and more material becomes electronic: the legal controls that allowed, for example, library lending are being replaced by technical controls that don't [730]. For example, when I applied for planning permission to extend my kitchen, I had to file four copies of a local plan; and the map software at our university library only lets you print three copies. This is of course quite deliberate. Legal controls are supplemented by access controls, and the legal privilege given to those access controls by the DMCA creates a new bundle of de-facto rights, criticised by many legal scholars as 'paracopyright' [364]. In effect, copyright regulations are no longer made by lawmakers in Washington or Brussels, but on the hoof by programmers working for Microsoft or Apple. The result, according to copyright law critics such as Larry Lessig and Pamela Samuelson, has been to greatly decrease the rights of copyright users.

At the same time, copyright law has suddenly become relevant to millions of people. Whereas in the past it was only a concern of specialists such as publishers, it now touches the lives of everyone who downloads music, time-shifts movies using a TiVo, or maintains a personal web page. As the law has failed to keep up with technology, the gap between what it permits and what people actually do has become wider. In the UK, for example, it's technically illegal to rip a CD on to an iPod, yet as this is one of the main reasons that people buy CDs nowadays, the British Phonographic Industry (the trade body) graciously says it won't sue anybody. The law-norm gap can only become wider as we move increasingly to a remix culture, and as the many minor infringements that used to take place in private, or undocumented public, spaces (such as singing a song in a pub) go online (as when a phone video clip of the song gets on someone's social-network page). John Tehranian calculates that a typical law professor commits over 80 copyright infringements a day, carrying statutory penalties of over $10 m [1243].

On the other side of the coin, there are the privacy concerns. In the old days, people would buy a book or a record for cash; the move to downloads means that DRM license servers run by firms such as Microsoft and Apple have a record of what people watch and listen to, and this can be subpoena'ed. (It's equally true that the move to online bookselling has created similar records at Amazon.) These records are also used for marketing. A survey for the Privacy Commissioner of Canada found many examples of intrusive behavior, including e-book software profiling individuals, DoubleClick advertising in a library service, systems tracking individuals via IP addresses, and contradictions between vendors' started privacy policies and observed behaviour — including undisclosed communications to third parties [468]. Not one of the organisations whose products and services were tested complied with requests from the testers to disclose personal information held about them. It looks inevitable that such DRM systems break European privacy law too, as it's based on the same principles. A particularly

extreme case arose in 2005 when Microsoft decided that Sony's XCP system was 'spyware' and and 'rootkit', and would thus be removed by Windows Defender and the Malcicious Software Removal Tool [884]. So next time you hear a large company bellyaching about how its users break the law by copying its music or its software, bear in mind that it may well be a lawbreaker too.

Where will it all lead? Until recently, the copyright law debate was a straight fight bewteen corporate lobbyists on the one hand, and activists such as academics and the free software community on the other; the latter had the arguments, but the former always got their way with legislatures. This has now changed, and in two interesting ways. First, the IP lobby started to run out of steam, and second, Hollywood started to realise that stronger DRM was likely to shift power away from content owners towards the platform vendors.

22.5.1 The IP Lobby

First, the IP lobby. This has its modern origins in an effort by the drug company Pfizer to extend patent protection on its drugs from the USA to less developed countries like Brazil and India in the 1970s. The story is told in a history by Peter Drahos and John Braithwaite [398]; in summary, Pfizer and the other drug companies allied themselves with the music and film industry (who wanted to cut bootlegging and copying), the luxury-goods industry (who wanted to reduce the number of cheap knock-offs), and a number of other U.S. players (including, it should be said, the Business Software Alliance), and persuaded the U.S. government to start applying pressure to less developed countries to bring their patent, copyright and trade-mark laws in line with America's. From the mid-1980s onwards this was largely a matter of bilateral action, but in 1994 the a treaty on Trade-Relates Aspects of Intellectual Property Rights (TRIPS) was signed, followed by two treaties of the World Intellectual Property Organisation (WIPO) in 1996. Essentially the USA and the EU got together and bullied holdouts like India and Brazil.

The implementation of these treaties stirred up a lot of opposition in developed countries as people began to realise how they might be affected. In the USA, the Digital Millennium Copyright Act of 1998 made it an offence to circumvent a copyright-protection mechanism, as required by WIPO, while in the European Union the Copyright Directive of 2001 had a similar effect. This was seen as enabling vendors to create closed platforms and control competition; it was also seen as a threat by the free and open source software movement, and by security researchers — especially after the Russian researcher Dmitri Sklyarov was arrested at a U.S. conference at the request of Adobe after his employer had sold tools circumventing password protection on pdf documents.

There were many other high-profile incidents; for example, I was on the program committee of the 2001 Information Hiding Workshop when an

attempt was made by the Recording Industry Association of America (RIAA) to force the program chair to pull a paper by Ed Felten and his students describing vulnerabilities in a copyright marking scheme being touted for a digital music standard [335]. This led to a lawsuit in which Ed sued RIAA, which became a landmark academic-freedom case [424]. The irony is that the promoters of this challenge had issued a public challenge to academics and others to break their scheme. The next case was Bunnie Huang's book 'Hacking the Xbox': this described in detail how, as an MIT student, he'd overcome the protection mechanisms in the first version of Microsoft's games console [629]. The book he wrote caused his publisher to take fright, but he found another one and the publicity can't have done his sales any harm. In any case, the encroachment on liberties threatened by rights-management mechanisms and anti-hacking laws led to the growth of digital rights NGOs in a number of countries (others had them already as a result of the 'Crypto Wars'; I'll discuss all this in more detail in Part III).

The turning point appears to have come in 2003–4, as the IP lobby was trying to steer a further measure through Brussels, the IP Enforcement Directive. This would have further ratcheted up the penalties on infringers and removed the prospects for public-interest defences based on free speech or fair use. This time opponents of the measure managed to assemble a sufficiently strong coalition of interests opposed to stronger IP enforcement that the measure was substantially amended. By then there were the beginnings of decent economic arguments, such as the first result mentioned in section 7.5.5 that downloading doesn't actually harm music sales; and meanwhile the IP lobby had seriously overreached.

For example, the IP folks tried to compel every country in Europe to make patent infringement a crime, rather than just a civil matter. This was designed by the leading drugs companies to undermine firms who manufacture and sell generic versions of drugs once they have come off patent. At present, drug companies try to prolong their patents by 'evergreening' — filing subsidiary, later patents, with often dubious derivative claims — which the generic drug-makers deal with by offering their distributors indemnities against having to pay damages. Making infringement a criminal matter would have upset these arrangements. This caused the generic drugmakers to oppose the directive vigorously, along with supermarkets, car parts dealers and consumer groups. Even the software industry started to get nervous: we pointed out to Microsoft that thousands of companies believe that Microsoft is infringing their patents, but don't have the money to go the distance in a civil court. If patent infringement became a crime, surely they would take their grievances to the police? Would Bill risk arrest on some future trip to Europe?

With hindsight, this was bound to be the eventual fate of the IP movement. A rich, powerful lobby isn't stopped by fine words, or by outrage from university professors and free-software activists (such as the words of John

Perry Barlow at the head of this chapter). It's stopped when it comes up against another rich, powerful lobby pushing in the opposite direction. That point now appears to have been passed; for example, in June 2007 WIPO abandoned the attempt by the IP lobby to conclude a new treaty on broadcasting.

The IP movement have have passed its high water mark, but it is not quite dead yet; following the election of Nicholas Sarkozy as President of France, they persuaded him to back a law that would make ISPs responsible for copyright enforcement — which would push up their costs and make them less competitive [115]. It will be interesting to see whether French ISPs — or the European Commission — manage to push back on this one; even though the tide is no longer flowing in the IP lobby's direction, there are bound to be a lot of skirmishes like this before the law finally stabilises.

There are some other problems with copyright that people will have to worry about eventually. Some copyright activists have assumed that once copyright expires — or assuming that lots of material can be made available under a Creative Commons license — then everything will be hunky-dory. I doubt it. Curating old bits costs money, just as curating old manuscripts does; indeed the film industry has recently discovered that archiving digital productions actually costs more than they used to pay in the old days, when they just locked away the master copies in an old salt mine. There's just an awful lot of bits generated during digital production, and copying them to new disks every few years isn't cheap. In the long term, once bitstrings belong to nobody, who will pay for their upkeep? Some lawyers would like to extend copyright term indefinitely, but that violates the social contract on which copyright is based and it also doesn't solve the problem properly: many publishers have failed to look after their own back catalogue properly and had to retrieve copies from national deposit collections. When NGO colleagues and I considered this problem the best solution we could come up with was a digital preservation law plus a taxpayer-funded digital deposit library [418].

22.5.2 Who Benefits?

As I discussed in section 7.5.5, a further important development came in 2005. In January of that year, Google's chief economist Hal Varian addressed a DRM conference in Berlin and asked who would benefit from stronger DRM. He pointed out that, in classical economic theory, a technical link between two industries would usually benefit the more concentrated industry (for example, car makers and car parts). Now the platform industry is concentrated (Apple, Microsoft) while the music industry is less so (four majors and many independents): so why should the music industry expect to be the winners from better DRM? Economic theory says that platform vendors should win more.

The music industry scoffed, and yet by the end of that year they were hurting — lobbying governments and the European Commission to 'do something'

about Apple, such as forcing it to open its FairPlay DRM scheme. It's now becoming clear that music downloading — with or without DRM — is changing the structure and dynamics of the music industry. Bands used to rely on the majors to promote them, but now they can do that themselves by giving away their albums on their websites; they always made most of their money from performances, and now they make more than ever.

In the first edition of this book, I predicted that big bands might follow the Linux business model: 'it may make sense to give the ''product'' away free and make money on the ''maintenance'' (tours, T-shirts, fan club . . .)', while for more specialised acts 'the trick may be slightly keener pricing and/or packaging that appeals to the collector'. So far, that was called right. I also wrote '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'. On that one, the current state of play is that CD sales are declining steadily, and while download sales are starting to rise, they're not rising fast enough yet to compensate. We shall have to wait and see.

22.6 Accessory Control

There was concern in the late 1990s and early 2000s that the prohibitions against reverse engineering introduced at Hollywood's request via the DMCA would have evil effects on competition. Indeed, one of the most important and rapidly-growing uses of cryptographic mechanisms and of rights-management technology generally since the DMCA was passed is in accessory control. The familiar example is the printer cartridge.

The practice started in 1996 with the Xerox N24 (see [1221] for the history of cartridge chips). In a typical system, if the printer senses a third-party cartridge, or a refilled cartridge, it may silently downgrade from 1200 dpi to 300 dpi, or even refuse to work at all. In 2003, expiry dates and ink usage controls were added: cartridges for the HP BusinessJet 2200C expire after being in the printer for 30 months, or 4.5 years after manufacture [827]; and modern cartridges now limit the amount of ink dispensed electronically rather than waiting for it to run out physically. The latest development is region coding: you can't use U.S. ink cartidges in a recently UK-purchased HP printer.

All this has caused grumbling; the European Parliament approved a 'Directive on waste electrical and electronic equipment' designed to force member states to outlaw the circumvention of EU recycling rules by companies who design products with chips to ensure that they cannot be recycled [230, 332]. But by the time this was translated into actual regulation, the authorities had

relented. The printer companies lobbied hard and the regulations eventually said that so long as the vendors accept empty cartridges back for disposal, they will have discharged their obligations to the environment [1323], while the UK government has said that as cartridges are consumables, the law won't apply [1097].

In the USA, the matter was decided in court. The printer maker Lexmark sued SCC, which (as I recounted in the chapter on tamper resistance at 16.5.1) had reverse-engineered their print-cartridge crypto, alleging violation of the Digital Millennium Copyright Act. Although they won at first instance, they lost on appeal in 2004 [790]. In a similar case, Chamberlain (who make garage door openers) sued Skylink (who made compatible openers) and also lost, losing the appeal too in 2004. This appears to settle U.S. law in favour of a free market for cryptologists, which was always the position before the DMCA came along [1110]. A firm wanting to control its aftermarket using crypto chips is free to hire the smartest cryptographers it can find to build authentication chips that are really hard to hack, and its competitors are free to hire the smartest cryptanalysts they can find to try to reverse-engineer them. Other fields in which crypto is used for accessory control include games console software and add-ons, and mobile phone batteries[3].

Other industries are eyeing up the technology. For example, the carmaker Volkswagen makes a range of models at different prices, and this is reflected in the price of spares; you can pay $12 for an air filter for a Skoda, $40 for the same air filter for a Volkswagen, and over $100 for the same air filter for an Audi. Will we in future see cars authenticating their major spare parts, to prevent arbitrage? (If the motor industry's smart it will be marketed as a 'safety measure' to stop unsafe foreign parts — this is the line Motorola took when it introduced authentication chips into phone batteries.)

Is accessory control objectionable? The economist Hal Varian analyses it as follows [1286]:

> The answer depends on how competitive the markets are. Take the inkjet printer market. If cartridges have a high profit margin but the market for printers is competitive, competition will push down the price of printers to compensate for the high-priced cartridges. Restricting after-purchase use makes the monopoly in cartridges stronger (since it inhibits refills), but that just makes sellers compete more intensely to sell printers, leading to lower prices in that

[3]There are limits to the freedom to reverse engineer. In the UK, a man was convicted in October 2007 of selling modchips that allowed people to run programs on games consoles without their being signed by the vendor. This relies on a 2003 law implementing the EU Copyright Directive that made Member States prohibit the sale of devices designed to circumvent copy protection. He is appealing [879]. However, a court in Helsinki made a possibly conflicting ruling [1216] — so it's not clear that we have a consistent law in Europe.

market. This is just the old story of ''give away the razor and sell the blades.''

However, in many other industries it might be anticompetitive; it just depends on how concentrated the industry is, and in winner-take-all platform markets it could be particularly objectionable [53]. So it's a good job that early fears of a legal prohibition against reverse engineering for compatibility have proved largely unfounded. Of course, the right to reverse engineer is not the same as the right to succeed at reverse engineering, and so there may be cases in the future — as shrinking feature sizes and growing complexity make reverse engineering harder — where firms locked out of a market will have to use antitrust law to get access.

22.7 Summary

The technical protection of digital content against unauthorised copying is a difficult problem both technically and politically. It's difficult technically because general-purpose computers can copy bitstrings at no cost, and it's difficult politically because Hollywood's attempts to impose rights-management technology on a reluctant computer industry have done a lot of collateral damage. Some of problems people agonised about — including the effects on reverse engineering for compatibility — have failed to materialise, but there is still real complexity. At the technical level, compatibility between DRM systems is a big issue, and it's not at all clear how DRM will work in the home of the future where people will expect to play music and videos that they've bought on multiple devices. At the political level, it appears that the music industry has brought down grief on itself by insisting on stronger DRM, as this simply shifted power in the supply chain from itself to the platform vendors (and specifically Apple). It remains to be seen whether the same will happen to Hollywood: will the move to high-definition video result in Microsoft stealing their lunch? At least there's one strategy for them to fight back — by using the SPDC mechanisms to assume part of the role of platform owner.

At any rate, the development of copyright and rights-management systems over the last ten years has been a fascinating if bumpy ride, and its evolution over the next ten will surely be too.

Research Problems

There are many interesting research problems in copyright management. Some of them we've already touched on, such as how to build cheaper tamper-proof hardware tokens, and better ways of embedding copyright marks in digital

pictures and sound. But a huge amount of research was done on these topics from the mid-90s until about 2005; there are many competing proposals, and lawyers are fighting through patent thickets. Novel ideas would surely be of interest to many stakeholders, but the low-hanging fruit may have been plucked by now.

Further Reading

Software copy protection techniques are discussed at length in [561]; there's a brief history of technical protection mechanisms for audio and video in [487]; and a racy account of the coevolution of attack and defense in pay-TV systems in [850]. 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), while there's a lawyer's view at [565].

There is an overview of information hiding techniques, including steganography and information hiding, in a special issue of the Proceedings of the IEEE [822]; for attacks on marking schemes in particular, see [1019, 1020]. For more detail there's a recent book on the subject [697]. Kahn is, as usual, good historical background reading [676]. A useful guided tour of U.S. copyright law is by Gordon [546]. Ongoing research work can be found in the proceedings of the workshops on information hiding [42, 97, 1022]. And finally, no chapter on copyright would be complete without a reference to Larry Lessig's books on the subject [784, 785] and Pam Samuelson's writings [1107, 1108, 1109, 1110].