CHAPTER

23

The Bleeding Edge

What information consumes is rather obvious: it consumes the attention of its recipients. Hence a wealth of information creates a poverty of attention, and a need to allocate that attention efficiently among the overabundance of information sources that might consume it.

— Herb Simon

Voting machine software is a special case because the biggest danger to security comes from the people who are supposed to be responsible for it.

— Richard Stallman

23.1 Introduction

Our security group at Cambridge runs a blog, `www.lightbluetouchpaper.org`, where we discuss the latest hacks and cracks. We even found some vulnerabilities in the Wordpress blog software we use and reported them to the maintainers. But we weren't alone in finding flaws, and in October 2007, the blog itself was compromised by a Russian script kiddie who tried to put on some drug ads. The attack itself was only an inconvenience, as we spotted it quickly and recovered from backup, but it brought home how dependent we've all become on a vast range of applications that we just don't have time to evaluate. And the blog posts themselves show that many of the attacks, and much of the cutting-edge work in security research, hinge on specific applications. There will still be exploits against platforms like Windows and Symbian, but there are many more vulnerabilities out there in apps. As Microsoft cleans up its act, and as search engines make it easier to find machines running specific apps, that's where the action may well shift.

In the case of blog software, the Wordpress security engineering was not very impressive, but its competitors are even worse; and this one application alone exposes thousands of machines to compromise. There are many, many applications, and their developers usually don't care about security until they get hacked. The same learning process that Microsoft's gone through since 2000 will be repeated in one domain after another. But not all applications are the same; while some (like blog software) simply open up PCs to botnet recruitment, there are others from which money can be extracted directly, others that people rely on for privacy, and others that mediate power.

I've already discussed a number of more or less 'embedded' apps, from banking through alarms to prepayment meters. In this chapter I'm going to briefly describe four types of application that make up the bleeding edge of security research. They are where we find innovative attacks, novel protection problems, and thorny policy issues. They are: online games; web applications such as auction, social networking and search; privacy technologies such as anonymizing proxies; and, finally, electronic elections.

Games and Web 2.0 highlight the fact that the real 'killer application' of the Internet is other people. As more people come online in ever more contexts, we're creating complex socio-technical systems of a kind that never existed before. That means quite novel attacks, exploits, tussles and disputes. We've already seen several examples, such as click fraud and impression spam, as well as new variants on old scams.

Anonymity systems follow naturally: if you want to reap some of the benefits of web applications but not end up exposing your privacy to them, you may wish to play under a pseudonym. Mechanisms to do this have been discussed for decades and real systems have emerged; but as you'd expect from our discussion of inference control in Chapter 9, anonymity is much harder than it looks.

Finally, elections are a classic example of an application where anonymity is required, but coupled with accountability: you want voters in an election to be able to satisfy themselves that their vote was counted, yet not to be able to prove to anyone else who they voted for (so they can't be bribed or bullied). Elections are also the key interface between social computing and power.

23.2 Computer Games

Games were one of the first applications of all — pretty well as soon as the world's first proper computer, the EDSAC, was operational, research students were writing games for it. The early AI researchers started writing chess programs, believing that when this problem was solved, computers would be able to function more or less as people. And in my own spotty youth, the first

cryptanalysis program I wrote was to let me peek ahead into the rooms of a dungeon game.

There are limited opportunities for cheating at games of perfect information like chess, especially when playing against a computer. But there are many ways to cheat in other games, and they're becoming a big deal. Millions of people play online games, many of them bright and a lot of them poor; the large online worlds have a turnover larger than some small countries; and thousands of people make a living from online games, from the developers who create and maintain them to Chinese gold farmers. So the motives for cheating exist; and as games are software, and software has bugs, the means for cheating exist too. Yet if cheating becomes pervasive it spoils the fun. People don't enjoy an unfair fight — and in the long run even successful cheaters get bored (unless cheating and counter-cheating become the new game). Even the perception of cheating destroys players' enjoyment, so they go elsewhere and the game vendor loses a ton of money. So vendors make a serious effort to stop it. All told, online games provide a first-class social laboratory for the study of hacking, and game security has become the subject of serious study.

Computer games are also big business, as they have been for decades. They drove the home-computer boom of the 1970s that in turn spawned the PC industry; games consoles have been a huge market for microprocessors and memory chips; and gaming — whether on consoles or PCs — has largely driven the development of computer graphics [1367]. By 2001, game sales in the USA hit $9.4 billion, outperforming movie box-office sales of $8.35 billion. Comparing the two industries isn't straightforward, as movie stars have other sources of income too, and the industries are getting entangled with more and more movies being made with computer graphics effects. But in order-of-magnitude terms, computer games are probably of comparable economic importance to movies. Certainly a blockbuster online game grosses much more nowadays than a blockbuster movie; as games go online, you're selling subscriptions, not just one-off tickets [203].

'Security' in games has meant different things down through the years. The early arcade games of the 1970s concentrated on protecting the coin box against robbers. When Nintendo moved console games into the home, they subsidised the consoles from later sales of software cartridges and other add-ons, so a lot of effort was put into controlling which accessories could be used, as I discussed in section 22.6; their later competitors from Sega to Sony and Microsoft ended up fighting both legal and technical battles against reverse-engineers. Copy-protection of game software for PCs has also been a big deal, and there have been pre-release leaks of standalone games, just like prerelease leaks of movies. However the move to online computer games has trimmed the concerns. As a critical part of the game logic runs on a server, the client software can be given away, and the residual issue is whether players can get an unfair advantage. That's what I'll focus on now.

23.2.1 Types of Cheating

There are basically three types of cheating.

The first is where the original game has a known vulnerability that goes across into the online world and may be made worse. For example, a hand of contract bridge starts with players taking turns to bid how many tricks they think they can take. Each hand has four players, in two teams, and during the bidding no-one may communicate any information about the cards in their hand to their partner other than their public bids. In competitive play, a screen is placed diagonally across the table during bidding so that no-one can see their partner. Yet there are still occasional suspicions that some covert communication has taken place, for example in the players' tone of voice. In the real world, allegations of cheating are heard by a jury of experienced players, who take a view on whether the outcome was better than could have been expected in honest play. Even so, some decisions remain controversial for years: players may be exceptionally skilful, or lucky, and partners who've played together for years may communicate subconsciously without trying to.

Bridge is an example of two much more general problems, namely exploiting game rules, and cheating by collusion, both of which existed long before computers. Moving to online play creates both an opportunity and a problem. The opportunity is that if players are honest, all the bids can be mediated through the system: there's no tone of voice to give rise to a dispute. The problem is that if four people are playing online bridge together from their own homes, then there's nothing to stop a pair setting up a separate communications channel — a single text message of the cards in a hand is all it takes. Can technology help? Well, online bridge means online records, so you can mine the records of large numbers of games to determine whether a suspect pair do better over the long run than they should. It also makes it easier to run tournaments where each match is played simultaneously by many players using the same deal of cards — which makes a cheat easier to spot. Finally, it facilitates new ways of organising play: if you have an online game server available 24 by 7, people can turn up singly to play and start a game whenever four have arrived. So people play with many partners rather than just one; both the motive to cheat and the means are reduced, while the risks are increased.

Where's there's a single forum, such as a single dominant server, you can also construct global controls. For example, a problem in some games is *escaping*, where someone who's losing simply drops the connection. With a global service, you can remove the incentive for this by recording an escape as a loss (but only so long as your service is reliable — some game servers end a quarter of sessions in crashes, and this strategy wouldn't be popular there). Other exploits that are also found in real-world games include pretending to be less skilled than you are, losing a few games, but then winning once the other player gets confident and plays for more money; pool-room and poker

sharks have used such strategies for ages, and there are many team variants too. In some games, you can get an advantage by having multiple players, and get some to kill off others to accumulate points. The susceptibility of online games to this sort of rigging depends to a great extent on whether identity is durable. If people can easily create, or cheaply buy, new identities, then rigging games using multiple front identities becomes simpler. (This is known as a *Sybil attack* and is also a problem in peer-to-peer systems [396].) One way to deal with this is a reputation system — a topic to which I'll return when we discuss auctions. In others, rather than having many characters operated by one human player, you use the reverse trick of having one character operated by shifts of successive human operators; this is typically done in online mulitplayer games where the goal is to accumulate online time and points.

The second type of cheating is where known computer-security issues apply straight off to the world of gaming. Five percent of the badware measured by Symantec in the first half of 2007 was aimed at online games, with the two most common items being Trojans designed to steal account information from players of Gampass and Lineage [1239]. There's a great variety of other attacks, from straightforward phishing to denial-of-service attacks that push up the network latency of your opponent so you can beat him at blitz chess. A lot of the material in this book applies one way or another to gaming: cheaters hack servers, eavesdrop on communications, and tamper with client memory to make walls invisible (there's a survey and taxonomy at [1368]). Policy issues such as privacy turn out to matter here too: a lot of people in Second Life were annoyed when an enterprising company built a search engine that went through their homes and indexed their stuff. And just as in other applications, a lot of exploits arise because of the chance discovery of bugs — such as in one game where you could drive up to a wall in a jeep, hit the 'disembark' button, and appear instantly on the other side of the wall [203]. Many games have glitches of this kind, and knowledge of them spreads quickly.

The third type are the new cheating tactics that emerge because of the nature of computer games, and the online variety in particular. In tactical shooters, for example, success should depend on the player's tactics and shooting skill, not on the game mechanics. Yet there are always shortcomings in the game's physics model, often introduced by network latency and by the optimisations game designers use to deal with it. In effect, the developers try to deceive you into believing that their world is consistent with itself and with Newton's laws, when it isn't really. For example, you'd normally expect that in a shooting duel, you'd have an advantage if you have the lowest network latency, or if you move first. Yet the prediction algorithms used in many game clients can twist this or into an exclusive-or: a high-latency player has an advantage if he moves first. This is because clients cache information about nearby players, so if you leap round a corner, see your enemy and shoot, then the slower

your network connection is, the longer it will take before he can see you and respond. (There's a wide range of such tactics: see [203] for a discussion.)

There are many interesting borderline cases where an activity can be skill or cheating depending on the circumstances. Mike Bond coined the term 'neotactic' to refer to players subliminally exploiting network effects, such as the latency advantage for first movers. Are neotactics genius, or cheating? As in Bridge, the one can easily be mistaken for the other. But most people would say it's definitely cheating if you use mechanical assistance, such as a proxy server that slows down your packet stream just before you launch an attack.

23.2.2 Aimbots and Other Unauthorized Software

That brings us on to one of the classic game cheats, namely bots. One of the persistent cheating strategies is to have code of your own to provide you with automation and support. People have written a huge variety of tools, from simple routines that repeatedly click a fire button (to hack the games where the rate at which you can physically fire is a factor) through proxies that intercept the incoming network packets, identify the bad guys, examine your outgoing shots, and optimise their aim. These *aimbots* come with different levels of sophistication, from code that does all the target acquisition and shooting, to human-controlled versions that merely improve your aim. They can hook into the packet stream as proxies, into the graphics card, or even into the client code. Another variant on the same theme is the *wall hack*, where a player modifies his software to see through walls — for example, by changing the graphics software to make them translucent rather than opaque.

Game companies who sell first-person shooters reckon that aimbots seriously spoil other players' fun, so they use encryption and authentication mechanisms to protect the packet stream. (These are usually proprietary and hackable but that will no doubt get fixed in time.) They also use guard software, such as Punkbuster, that uses anti-virus techniques to detect attempts to hook into game code or the drivers on which it relies. A recent innovation, found in Call of Duty 4, is to offer action replays of kills, seen from the viewpoint of the player who makes the kill: this enables the killed player to see whether he was killed 'fairly' or not. This may not only reduce cheating, but also the perception of cheating — which is almost as damaging to the game operator [204].

Inappropriate software can also be run on game servers. A common hack is to get an object such as a gun to run multiple copies of a script in parallel — a trick that could have been prevented by proper resource accounting at the server. However, servers face critical real-time demands and their designers try to keep them as simple as possible. Self-replicating objects are used to run service-denial attacks, or to create temporary buildings that escape the resource controls on permanent structures. And people program magic swords to do unexpected tricks.

It must be said, though, that the relatively uncontrolled game scripting languages which make this possible have also been turned to creative use. People have realised that game engines can be used to render whole movies in 3-d; the quality may not be as good as on Pixar's rendering farms, but it works, and it's essentially free. This has led to the growth of a whole new art form of *machinima* (machine cinema). As they say, it's a rare wind that blows nobody any good.

23.2.3 Virtual Worlds, Virtual Economies

Bots are also used in farming, where entrepreneurs do the boring legwork of accumulating game objects such as gold coins or magic swords for sale to impatient players. However, most of the farming is done by real people in low-wage countries from Romania to China. 'Gold farming' is now a significant export that's creating new economic opportunities for young people in remote villages that have few other employers [384]. The economy of a large online community, such as World of Warcraft — with 8 million subscribers, of whom half a million are online at any time — is larger that that of some countries.

This means in turn that macroeconomic effects, such as exchange rates and rents, start to matter. Second Life, for example, is essentially a 3-d chat room run by Linden Labs, which rents out space to third parties, who can then customise their property as they want. It has a local currency of Linden dollars, that can be bought for U.S. dollars using a credit card, either through Linden Labs or via third-party brokers. The currency enables in-game entrepreneurs to sell value-added items such as clothes, artwork, pornography and services. After the FBI cracked down on online casinos, there was a surge of interest in gambling in Second Life; so in April 2007 the Feds visited Linden Labs [1006]. Just before the visit, 26% of announcements were about gambling; after it, commercial rents fell. And the world of anti-money-laundering controls made its appearance when Linden Labs started discriminating between 'verified' and 'unverified' accounts (the former being those where the player had used a credit card to subscribe)[1].

Markets for game goods are also getting better organised. For several years, magic swords and gold coins were traded in grey markets on eBay, but starting with Sony's 'Station Exchange' in 2005, game operators began running proper auction sites where players can trade game goods for real money. In 2006, we had reports of the first serious fraud: crooks used stolen identities to set up hundreds of thousands of accounts on the South Korean game Lineage, with

[1]The Financial Action Task Force — an international body that bullies countries into asking people who open bank accounts to provide government-issue photo-ID and two utility bills — wants payment systems that don't participate in their 'identity circus' to impose limits on payment amounts and velocity. That's why accounts at PayPal or even African mobile-phone payment systems restrict what you can do if you're unverified.

allegedly some inside help, and cashed out by selling some $15 m in game goods [758]. This all helps to raise the stakes in gaming, and to make stability more important. It also brings us naturally to eBay and to other 'real-world' web applications.

23.3 Web Applications

While online computer games are partly implemented in servers and partly in client software, an increasing number of services are entirely online and accessed via a standard web browser. They range from auction services like eBay through search engines such as Google and Yahoo, online mail services like Hotmail and AOL, online word processors such as Google documents, and e-commerce sites selling all kinds of good things (and bad things). Some industries — travel, entertainment, insurance and bookselling — have moved much of their sales online. And the recent trend to social networking brings in all sorts of new angles.

There are many problems common to all manner of web sites. One is that web servers are often insufficiently careful about the input they accept from users, leading for example to the SQL insertion attacks I discussed in section 4.4.2. Another increasingly common vulnerability is *cross-site scripting* (XSS). Scripting languages such as javascript are supposed to observe a *same origin policy* in that scripts will only act on data from the same domain; you don't want a script from a Mafia-run porn site acting on your electronic banking data. But this policy has been repeatedly circumvented, exploiting everything from carelessly-written web services that let users upload html containing scripts that other users can then read and execute, to errors in the design of virtual machines — such as the Firefox javascript bug discussed in Chapter 18. Web services as diverse as Hotmail, Orkut, Myspace and even PayPal have been hacked using XSS exploits, and removing them completely from your site involves very careful processing of all user-supplied html code. However, even if your own site is clean, your customers may still be vulnerable. The latest tricks involve using web services to launder origin: for example, the attacker makes a mash-up of the target site plus some evil scripts of his own, and then gets the victim to view it through a proxy such as Google Translate [1239].

The problems are compounded when a single firm provides a wide range of services. Google, for example, offers everything from search through maps to mail and word-processing, and other large service companies also have broad offerings. Where many services live at the same domain, the same origin policy doesn't do much work, and there have indeed been a number of XSS-type vulnerabilities between applications at Google and elsewhere. There are also privacy consequences of service aggregation, which I'll come to later.

A further bundle of problems with web services is that their structure is usually at least partially open to inspection. A user is passed from one page to another as he goes through a process such as browsing, search, product selection and payment; the attacker can read the html and javascript source, observe how parameters are passed, and look for nuggets such as SQL queries that can be manipulated [78]. He can also look to see whether input choices are screened by javascript, and try bypassing this to see if interesting things can be done (this needn't mean buffer overflows, but even just such simple hacks as ordering stuff at a discount). He can also monkey with hidden fields to see if they're used to pass interesting data. A prudent developer will assume that clients are malicious — but most developers don't; the rush online by millions of businesses has totally outpaced the available security skills (and tools). As a result, many services are not only buggy but open to manipulation.

So much personal information is now stored on web-based applications that a successful attacker can make off with large amounts of exploitable data. In November 2007, Salesforce.com admitted that it had lost the contact lists of a number of its customers after an employee got phished; for example, its customer SunTrust had 40,000 of its own customers compromised of whom 500 complained of bad emails that seemed to come from SunTrust [743]. These emails tried to install malware on their machines. In an earlier incident, Monster.com's resume database was breached, compromising 1.3 million job seekers. (These incidents raise a question about the adequacy of current breach disclosure laws, many of which don't consider someone's email address and the name of one of their business contacts to be 'personal information' whose loss is notifiable — but clearly such losses should be notified.)

So much for general vulnerabilities. Let's now look at the specific problems of some common web services.

23.3.1 eBay

The leading auction site, together with its payment service company PayPal, are significant to the security engineer for quite a number of reasons. For starters, they're the phishermens' largest target by far, as well as being the platform for lots of old-fashioned fraud. Phishing attacks against PayPal directly are not much different from the attacks against banks that I discussed in Chapter 2 (except in that as PayPal isn't a bank, its customers don't have the protection of banking law, such as the U.S. Regulation E, and rely on PayPal's good will to make good their losses). Many other frauds are variants of old commercial scams. For example, hucksters email the underbidders in an auction, offering good similar to those that were on sale — but which turn out to be shoddy or nonexistent. And one of the biggest scams on eBay was run by a trader in Salt Lake City who sold laptops online. First he traded honestly, selling 750 laptops legitimately and accumulating a good reputation; then he

took money for 1000 more that he didn't deliver. That sort of trading strategy has been around as long as commerce has.

But the auction site adds another twist. It provides a reputation service whereby honest traders accumulate good references from their trading partners, and many small-time occasional sellers have acquired high trust ratings. So an increasingly common attack is to hijack one of their accounts — whether by password guessing, phishing or something more technical — and use this for fraud. Account takeovers have been reported to be growing rapidly from the start of 2007 [542].

The easy way to exploit a hijacked account is to sell nonexistent goods, take the money and run, but there are many variants. A trick that's growing in popularity in 2007 is the fake escrow site. The bad guy offers a car for sale; you win the auction; he then suggests that you use an escrow service to which he'll ship the car and you'll pay the money. Real escrow services do actually exist, such as `escrow.com`, but so do many dodgy services set up by fraud gangs [57]; if you wire them the money that's the last you'll see of it.

Escrow scams are an example of reputation theft, which brings us to Google.

23.3.2 Google

Google's security manager looks set to have one of the most interesting jobs in the business over the next five years, just as her counterparts at Microsoft have had over the last five. The sheer scale of Google's activities make it both a target and a conduit for all sorts of wickedness. Again, some of it's as old as sin, but other attacks are quite novel.

A good example is *Google hacking*, where people use a search engine to look for vulnerable machines. The online Google Hacking Database has hundreds of examples of search strings that turn up everything from unpatched servers to files containing passwords [810]. Suitable searches can also be used against human targets: these can be searches for anyone who's exposed themselves in some specific way, such as by leaving their social security number visible online, or searches for usable data on some specific person. If you're a possible target, it's a good idea to do the search first. For example, I was a target for a while of an animal-rights terror group[2], so I used the main search engines to find out where my home address could be found. Companies and governments regularly search for their own secrets too. Search has simply changed the world since Altavista burst on the world a little over a decade ago; inquiries that previously would have taken the resources of a major government can now be conducted from anyone's laptop or mobile phone in seconds. And although the benefits of this revolution greatly outweigh the costs, the costs aren't zero.

[2]I was an elected member of the governing body of Cambridge University, which was thinking of building a monkey house.

The two main innovations that enabled Google to pull away from the other search engines were the Pagerank algorithm, which ranks a page based on the number of other pages that link to it, and the idea of linking search to targeted advertising, rather than the banner ads used by earlier engines like Altavista. A number of small text ads are placed on the search result page, and advertisers bid against each other for specific search terms in a continuous auction. Advertisers are charged when a user clicks on one of their ads. Google also lets publishers put links on their web pages in which it serves ads relevant to the page content, and pays them a percentage of the click-through revenue. This has turned out to be hugely popular and profitable, and has revolutionised classified advertising.

Yet it's brought wave after wave of attacks. The big problem in 2006 was *click-fraud*; your firm's competitors click repeatedly on your ads, thereby burning up your ad budget. Click-fraud can also be done by publishers who want to maximise their commissions. Google added various algorithms to try to detect click fraud: repeated clicks from one IP address, for example, are discounted when it comes to billing. Attackers then figured out that the order in which ads appeared depends on the click-through rate, as Google optimises its own revenue by ranking popular ads higher. This led to a further service-denial attack, *impression spam*, in which your competitor repeatedly calls up the results pages in which your ads appear but doesn't click on them. This causes your click-through rate to plummet, so your ads get downgraded.

In 2007, one of the big problems was *Google arbitrage*. A publisher buys cheap ads to drive traffic to his site, where he writes about topics that attract more expensive ads. If customers who arrive at his site cheaply leave via a more expensive route, he makes a profit. Attitudes to this are mixed. Buying ads in order to sell ads has a long enough history; your local paper probably lives from classified advertising, and may also have posters all over town. Some advertisers think it's fraud: they pay for clicks from people fresh off searches, and instead get second-hand traffic from people escaping boring web pages that they didn't really want to go to. Google acts from time to time against the arbitrageurs, whose profits were dwindling by year end.

But this is just a small part of a larger problem, namely 'Made for Adsense' (MFA) sites. One pattern we've detected is the fake institution: the scamster copies an existing charitable or educational website with only minor changes to the content and uses the knock-off to host something with a high cost-per-click such as job ads. The idea is that where websites are highly ranked, copies of them will be too: and some of the bogus charities even set out to exchange links with real ones to further confuse the issue[3].

[3]That's how we stumbled across the network, when they offered an ad exchange with the Foundation for Information Policy Research, which I chair.

Another series of sites is run by a firm that buys up abandoned domain names, writes relevant editorial content, and fills them with ads. (The content is written for free by journalism students who look for places at which to 'intern' as part of their course). When I presented an analysis of such sites at Google, the reaction was mixed. There's a serious policy question here: although one might not think very much of the content, this is in some sense a new literary genre that's emerged on the back of the Adsense model, just as soap operas emerged once TV stations started being funded by adverts for fast-moving consumer goods. Googlers' view on balance was that such sites should be left to market forces. But there are clearly some close judgment calls between what's abuse and what's merely tiresome.

There are many interesting research problems here, such as how one goes about identifying and mapping bogus communities, where links have been manufactured to create a semblance of real social or economic activity in order to fool the Pagerank algorithm, and hidden communities, such as the network of sites based on abandoned domain names. The latter, at least, is similar to the problems faced by police and intelligence agencies searching for insurgent groups, while distinguishing bogus communities from genuine ones may also come to depend on increasingly sophisticated traffic analysis and social-network analysis of the sort discussed in sections 19.3.1, 21.5 and 24.3.2.

This brings us inevitably to the issue that most observers consider to be Google's Achilles heel: privacy. Privacy concerns operate at many levels.

- First, there's unauthorized access to data. When a firm with tens of thousands of employees holds personal data on hundreds of millions of people, there's a clear risk that information will leak — perhaps via a disgruntled insider, or perhaps via an exploit of some kind. These are basically the issues I discussed in Chapter 9, although on a much larger scale than ever before.
- Second, there's privacy law. For example, the European Commission is in dispute about how long clickstream data should be kept: Google's agreed to 'de-identify' clickstreams after 18 months, but this doesn't really satisfy the Commission. In section 9.3.1 I discussed how AOL released anonymised search histories for 'research', and some users were promptly identified from their search patterns; from the technical point of view, achieving privacy via anonymity requires frequent changes of pseudonym. (I'll return to privacy later when I deal with policy.)
- Third, there's lawful access to authorised data, as when the FBI (or a divorce lawyer) turns up in Mountain View with a subpoena.

Various people, for various reasons, will want to limit the possible damage resulting from one of more of these possible types of privacy exposure.

And the tensions look set to become steadily more serious as more information about us becomes available and searchable.

But before we go on to examine privacy technology, there's a third type of web service that can collect even more intimate and pervasive information: the social-networking site.

23.3.3 Social Networking Sites

Social networking sites such as MySpace and Facebook have taken off rapidly since about 2004, and are now used by large numbers of young people to organise their social lives. Other sites aim at organising professionals. The field is developing rapidly, but as I write in January 2008, the main users are still the young, and the typical site offers not just a place to store a home page but ways to link to your friends' pages too. The key idea is that the site mirrors the underlying social network: the added value is that it enhances your social network by helping you communicate with friends more conveniently and by making new friends. The access-control mechanisms typically let your friends see more of your stuff; there are messaging services such as forums, chat rooms and instant messenger, to support social interaction; and there are various structured methods of getting to know people. On some sites, you have to be introduced by mutual acquaintances; on others, you can search for people who meet given criteria such as age, location, musical tastes, hobbies, sex and sexual orientation. Society always had such mechanisms, of course, in the form of clubs, and some people see social networking merely as a kind of online cocktail party. However, the social-networking revolution enables rapid innovation of the mechanisms that people use to form friendships, look for partners and set up professional and business relationships.

The putative business model is, first, that the huge amount of information subscribers make available to the operators of these sites will enable even better targeted advertisement than on Google; and second, that friendship networks can create massive lockin, which is the source of value in the information industries. It's hard not to have a page on Facebook if all your friends do, and having a hundred million people keeping their address books and message archives on your servers rather than on their own PCs may seem like a license to print money[4].

So what problems should you anticipate when designing a social-networking site? Much government advice centres on the fearmongering aspects: young people reveal a lot about themselves on social sites, and can attract sex predators. It's certainly true that the Internet has had an effect on sex crimes, while

[4]Don't forget fashion though: in England everyone seems to have had a MySpace page in 2006, and most people have a Facebook page now in 2007 — but Brazilians use Orkut, and who can tell what will be cool in 2012?

no significant impact has been detected on any other category of offense. Its uptake across U.S. states and other countries was associated with a small rise in 'runaways', that is, under-18s leaving home without their parents' permission. Some runaways were no doubt kids escaping unsatisfactory homes or simply heading off to seek their fortunes; the key question is how many of them were abused. The figures show that Internet uptake was correlated with a drop in reported cases of rape, and that there was no increase in the murder rate [709]. One might worry about whether runaway teens turn to sex work, but prostitution also fell as the Internet spread. It might still be argued that a small increase in sexual abuse of runaway teens was masked by a larger fall in sex crimes overall, caused in turn by the greater availability of pornography; but the drop in sex crimes was significant only among male offenders aged 15–24 and there was no corresponding increase of older offenders. And young people I've talked to take the view that fending off unwanted advances from older men is just part of life anyway, whether offline or online.

The view you take of all this, if you're building such a system, may depend on whether your site is aimed at distance networking — as with photographers from different continents trading pictures and tips on Flickr — or at networks involving physical relationships. In the second case, the next question is whether you restrict membership to teens and above, as Facebook does, or let anyone join, as MySpace does. There's a reasonable discussion of the policy and technical issues from ENISA [615]. As for younger children, it's clearly prudent for parents to keep an eye on online activities; the junior members of our family get to use the computer in the kitchen, which also helps get across the message that the Internet is public space rather than private space. ENISA also recommends that schools should encourage rather than prohibit social network use, so that bullying can be reported and dealt with. Bullying has always been a low-grade problem for schools, erupting into very occasional tragedy with suicides or killings. Before the Internet, bullied children could escape for long periods of time at home and with friends; nowadays the taunting can follow them to their bedrooms. This can make it all the more miserable if their Internet use is secret and furtive. The cure is to bring things into the open.

These are, of course, broad issues that apply to Internet use generally, not just to social networking sites. And on the face of it, you might expect social networking sites to be less dangerous than random online chat rooms, as the service operator has an incentive to limit egregious abuse because of the associated reputation risk. But what are the interesting security engineering issues?

One emerging property of social networking systems is the sheer complexity of security policy. In Chapter 4, I discussed how access controls are simple close to the hardware, but get more complex as you move up the stack through the operating system and middleware to the application. Social networking applications attempt to encapsulate a significant subset of human behaviour

in groups; the result is ever-more complicated sets of rules, which are very difficult for users to manage.

For example, setting privacy policy on Facebook in October 2007 means wading through no less than six screens of access options — essentially a set of access control lists to different parts of your profile. And the controls aren't linear; photos, for example, have a policy of opt-in and opt-out. If I recognise you in a photo on someone's page, I can tag that photo with your name, but the tag won't be publicly visible until the photo owner agrees (the opt-in). If you're a Facebook member you'll be notified, and you can remove the tag if you want (the opt-out) [450]. However you might not notice the tag, and this could have consequences if the photo were embarrassing — say, a drunken party. For example, on New Year's day 2008, following the assassination of Benazir Bhutto in Pakistan, the UK press published a photo of her son and political heir Bilawal Bhutto, dressed up in a devil's costume with red horns for a Halloween party, which was found on the Facebook site of one of his student friends.

Many people just don't understand what the terms used in the access controls mean. For example, if you make photos available to the 'community', that means by default anyone with an email address within your institution — which has led to campus police having discussions with people who've uploaded photos of assorted kinds of rulebreaking activities [463]. Facebook also doesn't deal with multiple personae; for example, I'm Ross the computer scientist, Ross the technology-policy guy, Ross the musician, Ross the family man, and so on — and as I can't separate them, I've 'friended' only people I know from the first two communities on Facebook.

There are also some quite subtle policy issues: for example, you'd think it was alright to publish information that was already public? Wrong! There was a huge row when Facebook added a news feed feature that notified all your status changes to all your friends. Previously, if you'd broken up with your girlfriend, this would be publicly visible anyway (assuming you made partnership data public). Suddenly, such a change was automatically and instantly broadcast to your entire social circle — which upset a lot of people. It's not just that the site had automated some of the social aspects of gossip; it suddenly made social faux pas very visible [216]. This feature can now be turned off, but the extra complexity just makes it even harder for people to manage their privacy.

Another example is search. Most web services firms are tempted to use private data in public searches in order to make them more precise. Back in the early days (2004), it turned out that search on Friendster leaked private data: a chain of suitably-chosen searches could infer a user's surname and zip code even where this data had been set as private [902]. The moral was that public searches should never be allowed to return results based on private data. Another lesson that should have been more widely learned is that once the social network is known, inference control becomes much harder. Friendster

duly fixed its systems. Yet the bug was rediscovered in Facebook in 2007: Facebook had simply left it to users to decide whether they wanted to turn off search on private terms, and essentially none of them had done so [1197]. That's now fixed, but the issue arose yet again when Facebook made available stripped-down versions of user profiles to Google. Once more, it had been left to users to become aware of this risk and turn off the relevant feature; once more, almost nobody did so [694]. Facebook appears to have a strategy of dumping all the really hard security decisions on the users — so they can respond to criticism by blaming users for not turning off features X and Y. Searchability by default may be in their short-term financial interest, but the end result can too easily be unusable security plus unsafe defaults.

Another tension between corporate growth and security usability was a recent decision to allow third-party application developers access to profile data. When someone builds an app that allows people to export photos (say) from Flickr to Facebook, then how on earth are we to evaluate that? Even if the two systems are secure in isolation, there's no guarantee that this will compose — especially where systems have complex and ever-changing APIs, and complex hard-to-use privacy policies. Then, in late 2007, Facebook faced a revolt of its users after it introduced 'Beacon', an advertising system that told users' friends about what they'd just purchased on other websites, and made the feature opt-out rather than opt-in. Mark Zuckerberg, founder and chief executive, apologized to the site's users for handling the matter badly. (It remains to be seen whether future marketing ideas will be opt-in.)

There are both 'soft' and 'hard' issues bundled up here. At the soft end, people present different personae at different sites; for example, by placing different kinds of photos on Flickr and Facebook [1276]. At the nastier end, not all applications are written by large, respectable companies. Yet once you authorise a third-party application to access your profile, it can do most anything you can — including spamming your friends and selling their personal information to third parties. There's a sense in which making a 'friend' on Facebook is the digital equivalent of unprotected sex — you're exposed to everything they've got.

All the usual tussles between free speech and censorship pop up too. For example, in Flickr, you're not supposed to upload photos you wouldn't show your mum unless you tag them as 'restricted' (i.e. adult). You're not allowed to view such material if you're in Germany, or search for it in Singapore. Yet a colleague who uploaded art nudes had his account blacklisted as 'unsafe', even though he's quite happy to show his mum his work. And as far as data protection law is concerned, Facebook tends to reveal the data subject's race, sex life, health, religion, political beliefs and whether he belongs to a trade union — precisely those 'sensitive' types of data that get special protection under European privacy law [235].

There's a further bunch of problems at the interface between technical and social mechanisms. For example, you make confidential information available to your friends, one of whom gets his account compromised, and your data ends up public. When this was first reported (in 2003), pioneers expected that social pressures would make users more careful [961], but this hasn't happened. The underlying reasons may be familiar: in a world of strong network externalities, systems start off insecure in order to grow as quickly as possible. But while Windows and Symbian were insecure in order to appeal to complementers while building a dominant position in a two-sided market, social-network site operators bias their algorithms and their presentation to get people to enrol as many friends as possible. This undermines any possible social discipline.

Other socio-technical attacks include cross-site scripting vulnerabilities, of which there have been plenty [902]. Spam is rising fast, and a particularly ominous problem may be phishing. A recent experiment at Indiana University sent phish to a sample of students, asking them to check out an off-campus website that solicited entry of their university password. The success rate with the control group was 16% but a group targeted using data harvested from social networks were much more vulnerable — 72% of them were hooked by the phish [653]. Already there's a significant amount of phishing being reported on MySpace [796].

I'll finish up this section by making two more general points. The first is that, as the social-networking sites learn rapidly from experience and clean up their act, the largest medium-term problems may well be, first, the migration online of real-world social problems such as bullying; and second, that many teens put stuff online that they'll later regret, such as boasts of experiments with drink, drugs and sex that get dug up when they apply for jobs. In Seoul, a girl was asked to pick up some poo left by her dog, and refused; a bystander filmed this, she became known as 'dog poo girl', and she was hounded from university [1201]. Although that's an extreme case, the principle is not really new: people who posted immoderately on the old network news system sometimes found themselves haunted by 'the Ghost of Usenet Postings Past' [507]; and there tales going back centuries of social faux pas that ruined lives, families and fortunes. But while in olden times it would most likely be a lapse of manners at court that got you in bad trouble, now it can be a lapse of manners on the subway.

The world is steadily becoming more documented — more like the villages most of us lived in until the eighteenth century, where everyone's doings were known. Back then, the village gossips would serve up a mèlange of assorted factoids about anyone local — some true, and some false — which people would use when forming opinions. Nowadays, Google has taken over that role, and it's much less susceptible to social pressure to correct errors,

or forgive the occasional blunders of otherwise decent people. Also, much of the anonymity that people got from moving into towns during the industrial revolution is being lost. The effect of persistent social memory on social problems will be mixed. Bullying may be mitigated because of the record left behind, while the embarrassment problem may be resolved by some combination of a more relaxed attitude to youthful indiscretions, plus greater discretion on the part of youth. We'll have to wait and see which dominates, but early signs are that people are becoming more relaxed: the Pew Internet & American Life Project found that 60% of Americans are unconcerned in 2007 about the 'digital footprint' they leave behind, while in a survey in 2000, 84% were worried. So we're learning to cope' [1229]. (Discretion is part of coping, and that may involve the use of a pseudonym or nickname that isn't too easy for outsiders to link to your real person, but I'll discuss all that in the next section.)

Second, social network systems have the potential to do an awful lot of good. The Harvard sociologist Robert Putnam documented, in his influential book 'Bowling Alone', how social networks in America and elsewhere were damaged by the advent of television, the move to the suburbs and even the move by women into work (though TV did by far the most damage) [1052]. The baby-boom generation, who were the first to be raised with TV, are much less likely to join clubs, know our neighbours, meet frequently with friends or participate in team sports than our parents did, and the succeeding 'generation X' are less likely still. Now it seems that sociability is ticking upwards once more. What TV and mass consumer culture took away, the PC and the mobile phone may be giving back. Easier communication not only makes people communicate more but in different ways; the old communities based on geography are being supplemented by communities of shared interest. We academics were among the first to benefit; the communities of people interested in cryptography, or protocols, or number theory, have been held together as much by the Internet as by the conference circuit for many years. Now these benefits are spreading to everybody, and that's great.

Social-networking sites also provide a platform for rapid experimentation and innovation in new ways of making and maintaining friendships. And they may be brilliant for the geeky, the shy, the ugly, and people with borderline Asperger's. But to the extent that they try to encapsulate more and more of the complexity of real social life, their policies will become ever more complex. And just as we're limited in our ability to build large software systems by technical complexity, so social-networking systems may explore a new space in which policy complexity — security usability, in a new guise — may provide one of the ultimate limits to growth. It will be interesting to watch.

23.4 Privacy Technology

As business moves online, vast amounts of information start to get collected. In the old days, you walked into a record store and bought an album for cash; now you download a track from a server, which downloads a license to your PC. The central license server knows exactly who bought access to what, and when. Marketers think this is magnificent; privacy advocates are appalled [410]. The move to pervasive computing is also greatly increasing the amount of information held on us by others — for example, if people start using applications in their mobile phones to track their social networks and help them manage their time better [407]. There will no doubt be all sorts of 'must have' applications in the future that collect data about us, which means growing uncertainty about what will be available to whom.

Technology is already putting some social conventions under strain. In pre-technological societies, two people could walk a short distance away from everyone else and have a conversation that left no hard evidence of what was said. If Alice claimed that Bob had tried to recruit her for an insurrection, then Bob could always claim the converse — that it was Alice who'd proposed to overthrow the king and he who'd refused out of loyalty. In other words, many communications were *deniable*. Plausible deniability remains an important feature of some communications today, from everyday life up to the highest reaches of intelligence and diplomacy. It can sometimes be fixed by convention: for example, a litigant in England can write a letter marked 'without prejudice' to another proposing a settlement, and this letter cannot be used in evidence. But most circumstances lack such clear and convenient rules, and the electronic nature of communication often means that 'just stepping outside for a minute' isn't an option. What then?

Another issue is anonymity. Until the industrial revolution, most people lived in small villages, and it was a relief — in fact a revolution — to move into a town. You could change your religion, or vote for a land-reform candidate, without your landlord throwing you off your farm. Nowadays, the phrase 'electronic village' not only captures the way in which electronic communications have shrunk distance, but also the fear that they will shrink our freedom too.

Can technology do anything to help? Let's consider some 'users' — some people with specific privacy problems.

1. Andrew is a missionary in Texas whose website has attracted a number of converts in Saudi Arabia. That country executes citizens who change their religion. He suspects that some of the people who've

contacted him aren't real converts, but religious policemen hunting for apostates. He can't tell policemen apart from real converts. What sort of technology should he use to communicate privately with them?

2. Betty is your ten-year-old daughter, who's been warned by her teacher to remain anonymous online. What sort of training and tools should you give her to help her manage this?
3. Charles is an engineer at a Silicon Valley startup that's still in stealth mode, and he's running a blog — in contravention of his company's rules. How can he avoid getting caught and fired?
4. Dai is a human-rights worker in Vietnam, in contact with people trying to set up independent trade unions, microfinance cooperatives and the like. The police harass her frequently. How should she communicate with co-workers?
5. Elizabeth works as an analyst for an investment bank that's advising on a merger. She wants ways of investigating a takeover target without letting the target get wind of her interest — or even learn that anybody at all is interested.
6. Firoz is a gay man who lives in Teheran, where being gay is a capital offence. He'd like some way to download porn without getting hanged.
7. Graziano is a magistrate in Palermo setting up a hotline to let people tip off the authorities about Mafia activity. He knows that some of the cops who staff the office in future will be in the Mafia's pay — and that potential informants know this too. How does he limit the damage that future corrupt cops can do?

This helps to illustrate that privacy isn't just about encrypting phone calls or web traffic. For example, if Andrew tells his converts to download and use a particular cryptography program, then so will the police spies; and the national firewall will be set to detect anyone who sends or receives messages using that program. Andrew has to make his traffic look innocuous — so that the religious police can't spot converts even when they have full knowledge of what apostate traffic looks like.

And while suitable technical measures may solve part of Andrew's problem, they won't be much use with Betty's. The risk to her is largely that she will give out information carelessly that might come back to haunt her. Filtering software can help — if she's not allowed to give out her home address over the Internet, a filter can look for it, and beep if she gets careless — but most of the effort will have to go into training her.

There's also wide variation in the level at which the protection is provided. Betty's protection has to be provided mostly at the application layer, as the main problem is unintentional leaks via content; the same applies to Charles.

However, Charles might face more sophisticated analysis, perhaps at the hands of someone like Elizabeth: she might trawl through his postings looking for metadata from camera serial numbers in the images to names of workgroups or even printers embedded in documents, so that she can figure out who he's working with on his secret project.

The intensity of attacks will also vary. Charles and Firoz might face only sporadic interest, while Dai is subjected to constant surveillance. She'll use anonymous communications not so much to protect herself, but to protect others who haven't yet come to the police's attention. There are huge differences in protection incentives: Andrew may go to a lot of trouble to make his website as harmless as possible to its visitors (for example, by hosting it on the same machine as many innocuous services), while the sites in which Firoz is interested don't care much about his safety. Andrew, Dai and Graziano all have to think hard about dishonest insiders. Different probability thresholds mark the difference between success and failure; plausible deniability of an association might be enough to get Charles off the hook, while mere suspicion would frustrate Elizabeth's plans. And there are different costs of failure: Elizabeth may lose some money if she's caught, while Firoz could lose his life.

We've come across anonymity mechanisms before, when we discussed how people who don't want their phone calls traced buy prepaid mobile phones, use them for a while, and throw them away. Even that's hard; and even Al-Qaida couldn't do it right. So what are the prospects for hard privacy online?

23.4.1 Anonymous Email – The Dining Cryptographers and Mixes

As we remarked in several contexts, the opponent often gets most of his information from traffic analysis. Even if the communications between Alice and Bob are encrypted and the ciphertext hidden in MP3 files, and even if on inspection neither Alice's laptop nor Bob's contains any suspicious material, the mere fact that Alice communicated with Bob may give the game away.

This is why criminals set much more store by anonymous communication (such as using prepaid mobile phones) than by encryption. There are many legitimate uses too, from the folks on our list above through anonymous helplines for abuse victims; corporate whisteblowers; protest groups who wish to dig an elephant trap for the government; anonymous student feedback on professors; anonymous refereeing of conference paper submissions, and anonymous HIV tests where you get the results online using a one-time password that came with a test kit you bought for cash. You may want to apply for a job without your current employer finding out, to exchange private email with people who don't use encryption, or fight a harmful and vengeful cult.

There are two basic mechanisms, both invented by David Chaum in the 1980's. The first is the *dining cryptographers problem*, inspired by the 'dining

philosophers' problem discussed in section 6.2.4. Several cryptographers are gathered around a table for dinner, and the waiter informs them that the meal has already been paid for by an anonymous benefactor, who could be one of the participants or the NSA. The cryptographers would like to know which. So pairs of principals share one time pads, after which each principal outputs a function of her 'I paid/I didn't pay' bit and everyone can later work out the total parity of all such bits. As long as not more than one of the cryptographers says 'I paid', even parity means that the NSA paid, while odd parity means that one of the diners paid, even if nobody can figure out who [286]. This mechanism can be considered the anonymity equivalent of the one-time pad; it gives 'unconditional anonymity', albeit at the cost of a laborious protocol and a lot of key material. Various extensions have been proposed, including one in which 'dining drug dealers' can auction a consignment of cocaine without the bidders' identities revealed to the other bidders or to the seller. Nobody except buyer and seller know who won the auction; and even the seller is not able to find out the identity of the highest bidder before committing to the sale [1219].

However, for practical anonymity applications, the pioneering innovation was another idea of Chaum's, the *mix* or *anonymous remailer* [284]. This accepts encrypted messages, strips off the encryption, and then remails them to the address that it finds inside. In its simplest form, if Alice wants to send anonymous email to Bob via Charlie and David, she sends them the message:

$$A \longrightarrow C : \{D, \{B, \{M\}_{KB}\}_{KD}\}_{KC}$$

Charlie now strips off the outer wrapper, finds David's address plus a ciphertext, and sends the ciphertext to David. David decrypts it and finds Bob's address plus a ciphertext, so he sends the ciphertext to Bob. Bob decrypts this and gets the message M.

Anonymous remailers came into use in the 1990s. To start off with, people used single remailers, but, as I mentioned in section 22.3.3, an early remailer was closed down following court action by the Scientologists, after it was used to post material critical of them. A lot of people still rely on services such as Hotmail and Hushmail that provide simple, low-cost anonymity, but if you might be subjected to legal compulsion (or sophisticated technical attack) it's wise not to have a single point of failure[5]. Chainable remailers were initially developed by the cypherpunks; they not only did nested encryption of outgoing traffic but also supported a *reply block* — a set of nested public keys and email addresses that lets the recipient reply to you. There are also *nymservers* that will store reply blocks and handle anonymous return mail automatically. The most common design at present is the Mixmaster remailer, which also

[5]'Wired' was surprised in November 2007 when it turned out that Hushmail responded to warrants [1177] — which itself is surprising.

protects against basic traffic analysis by padding messages and subjecting them to random delays [899].

A common application is anonymous posting to mailing lists with sensitive content — applications range from reporting security vulnerabilities through abuse support to anonymous political speech. Of course, an anonymous remailer could be an attractive honey trap for an intelligence agency to operate, and so it's common to send messages through a number of successive remailers and arrange things so that most of them would have to conspire to break the traffic. Even so, selective service-denial attacks are possible; if the NSA runs remailers X and Y, and you try a path through X and Z, they can cause that to not work; so you then try X and Y, which 'works', and you're happy (as are they). Remailer operators can also be subjected to all sorts of attacks, ranging from subpoenas and litigation to spam floods that aim get the operator blacklisted; David Mazières and Frans Kaashoek have a paper on their experiences running such a service [849]. The technology is still evolving, with the latest proposals incorporating not just more robust mechanisms for fixed-length packets and single-use reply blocks, but also directory and reputation services that will allow users to monitor selective service-denial attacks [344].

23.4.2 Anonymous Web Browsing – Tor

Anonymous connections aren't limited to email, but can include any communications service. As the web has come to dominate online applications, The Onion Router (Tor) has become the most widely-used anonymous communication system, with over 200,000 users. Tor began its life as an experimental US Navy Labs system, called Onion Routing because the messages are nested like the layers of an onion [1062]. The Navy opened it up to the world, presumably because you can usually only be anonymous in a crowd. If Tor had been restricted to the U.S. intelligence community, then any website getting Tor traffic could draw an obvious conclusion. U.S. Naval personnel in the Middle East use Tor to connect back to their servers in Maryland. They don't think of it as an anonymity system but as a personal safety system: they don't want anyone watching the house they're in to learn their affiliation, and they don't want anyone watching the servers in Maryland to learn where they are. In effect, they hide among local (Iraqi and Maryland) men looking for porn; and porn traffic also conceals human-rights workers in the third world. Tor may be a part of the solution adopted by several of our representative privacy users (Charles, Dai, Elizabeth, Firoz and maybe even Graziano), so I'll now discuss its design and its limitations[6].

[6]By way of declaration of interest, I hold a grant from the Tor Project that pays one of my postdocs to help develop their software. There are also commercial services, such as Anonymizer [79], that let you browse the web anonymously, but they're routinely blocked by repressive governments.

The Tor software consists of the Tor client, which forwards TCP traffic, a web proxy through which it talks to your browser, and optionally a 'Tor Button' that acts as an extension to the Firefox browser and lets you switch rapidly between normal and anonymous browsing. In the latter mode, the Tor button disables cookies, javascript and all other plugins. Volunteers with high-bandwidth connections enable the Tor client to also act as a server, of which there may be a few thousand active at any time. When you turn on a Tor client, it opens a circuit by finding three Tor servers through which it connects to the outside world. It negotiates an encrypted session with the first server, then through this it negotiates another encrypted session to the second server, through which it then sets up a third encrypted session to the exit node. Your web browser traffic flows in the clear from the exit node to your destination.

This brings us immediately to one widely-publicised Tor vulnerability — the *malicious exit node*. In September 2007, someone set up five Tor exit nodes, monitored the traffic that went through them, and published the interesting stuff [917]. This included logons and passwords for a number of webmail accounts used by embassies, including missions from Iran, India, Japan and Russia. (This gave an insight into password robustness policy: Uzbekistan came top with passwords like 's1e7u0l7c' while Tunisia just used 'Tunisia' and an Indian embassy '1234'.) Yet the Tor documentation and website make clear that exit traffic can be read, so clueful people would have either used a webmail service that supports TLS encryption, like Gmail, or else used email encryption software such as PGP (which I'll mention later).

The second problem with anonymous web browsing is the many side-channels by which modern content calls home. This is why the proxy distributed with the Tor client kills off cookies and javascript, but that's just the beginning. If Firoz downloads a porn movie, and his Windows Media Player then calls the porn server directly to get a license, the packet traffic from his IP address to a 'known Satanic' site may be a giveaway; but then, if he blocks the license request, he won't be able to watch the film. ActiveX controls, Flash and other browser add-ons can also open connections outside the proxy. For surveys of ways in which websites can defeat anonymising services, see [1091, 1194].

Third, while the Mixmaster and later remailers can make traffic analysis harder by dicing, padding and delaying traffic, this introduces latency that would not be acceptable in most web applications. Low-latency, high-bandwidth systems such as Tor are intrinsically more exposed to traffic analysis. A global adversary such as the NSA, that taps traffic at many points in the Internet, can certainly recover information about some Tor circuits by correlating their activity; in fact, they only need to tap a small number of key exchange points to get a good sample of the traffic [920] (so if the U.S. government figures in your threat model, it may be prudent to set up new Tor circuits frequently).

Finally, many applications get users to identify themselves explicitly, and others get them to leak information about who they are without realising it. In section 9.3.1 I discussed how supposedly anonymous search histories from AOL identified many users: a combination of local searches (that tell where you live) and special-interest searches (that reveal your hobbies) may be enough to identify you. So if you're using Tor to do anonymous search, and these is even the slightest possibility that your opponent might be able to serve a subpoena on the search engine, you had better set up new circuits, and thus new pseudonyms, frequently.

If your opponent is less capable, then traffic patterns may still give the game away. First, suppose you want to browse a forbidden website that has a known and stable structure; a modern commercial web page might contain some 30 objects ranging in site from a few hundred bytes to a few tens of kilobytes. This pattern is usually unique and is clearly visible even after TLS encryption. Even although Tor traffic (as seen by a wiretap close to the user) lies under three layes of Tor encryption, and even though cells are padded to 512 bytes, random web pages still leak a lot of information about their identity. So if Andrew wants his converts to view his website through Tor, and there's a real risk that they'll be killed if they're caught, he should think hard. Should be pad his webpages so that, encrypted, they will be the same size as a popular and innocuous site? Should be put short sermons on YouTube, of the same length as popular music tracks? Or should he use a different technology entirely?

An opponent who can occasionally get control of the forbidden website can play yet more tricks. Graziano, who's worried about Mafia takeover of the police's Mafia tip-off site, should consider the following attack. The Mafia technicians make a number of probes to all the Tor servers as the page is loaded, and from the effects on server load they can identify the path along which the download was made. They then go to the local ISP, which they bribe or coerce into handing over the traffic logs that show who established a connection with the entry node at the relevant time [919]. (So from Graziano's point of view, at least, the recent European directive compelling ISPs to retain traffic logs may not always help the police.)

There's no doubt that Tor is an extremely useful privacy tool, but it has to be used with care. It's more effective when browsing websites that try to respect users' privacy than when browsing sites that try to compromise them; and it's often used in combination with other tools. For example, human-rights workers in less developed countries commonly use it along with Skype and PGP.

23.4.3 Confidential and Anonymous Phone Calls

I discussed in Chapter 20 how criminals looking for anonymous communications often just buy prepaid mobiles, use them for a while, and throw them away. They are a useful tool for others too; among our representative privacy

users, Andrew might think of telling his converts to use them. But are not the only option, and they don't provide protection against wiretapping. If your opponent has the technology to do automatic keyword search or speaker recognition on phone traffic, as the NSA does, or the manpower to run a large number of wiretaps, as a typical third-world despot does, then you might want to consider voice over IP.

In theory, you can run VOIP communications through proxies like Tor [665]; but in practice, not many people do; and as anonymity usually means hiding in a crowd, that brings us to Skype. Skype is not only the largest VOIP operator, which gives safety in numbers; it's got a peer-to-peer architecture, so your calls go end-to-end; and the traffic's encrypted, with mechanisms that have undergone an independent evaluation [165].

So what can go wrong? Well, if Andrew were to use Skype to talk to his converts then he'd better not use the same username to talk to all of them; otherwise the religious police will learn this username from their bogus convert and search for everyone who calls it. Fortunately, you can get multiple, throwaway Skype usernames, and provided Andrew uses a different username for each contact Skype may be a suitable mechanism. The next problem, for some applications at least, is that Skype being owned by a large U.S. company is likely to respond to warrants[7] So if your threat model includes the U.S. Government, you'd better assume that the call content can be decrypted once the NSA identifies your traffic as of interest. You might be at risk if you're opposing a government, such as that of Uzbekistan, with which the USA has intelligence-sharing agreements; and you might also be at risk if Skype's parent company, eBay, has an office in the country whose police you're trying to hide your traffic from. So if Andrew's unsure about whether eBay would help out the Saudi government, he might use Skype largely as an anonymity mechanism, and use it to mask the transfer of files that are encrypted using a product such as PGP.

Human-rights workers such as Dai do in fact use Skype plus PGP plus Tor to protect their traffic, and the attacks to which they're subjected are the stuff of intelligence tradecraft. The police enter their homes covertly to implant rootkits that sniff passwords, and room bugs to listen to conversations. When you encrypt a phone call, you have to wonder whether the secret police are getting one side of it (or both) from a hidden microphone. Countering such attacks requires tradecraft in turn. Some of this is just like in spy movies: leaving telltales to detect covert entry, keeping your laptop with you at all times, and holding sensitive conversations in places that are hard to bug. Other aspects of it are different: as human-rights workers (like journalists but unlike

[7]Skype itself is actually a Luxembourg company, and its officers who respond to law enforcement are based there: so an FBI National Security Letter may not be effective, but a judicial warrant should be.

spies) are known to the host government, they need to avoid breaking the law and they need to nurture support structures, including overt support from overseas NGOs and governments. They also need — while under intermittent observation — to make covert contact with people who aren't themselves under suspicion.

23.4.4 Email Encryption

During the 'Crypto Wars' on the 1990s, cyber-activists fought their governments for the right to encrypt email, while governments pushed for laws restricting encryption. I'll discuss the politics in the next chapter. However one focus of that struggle, the encryption product *Pretty Good Privacy* (PGP), along with compatible free products such as GPG, have become fairly widely used among geeks. A typical use is by Computer Emergency Response Teams (CERTs) who encrypt information about attacks and vulnerabilities when they share it with each other. Many private individuals also have PGP encryption keys and some encrypt traffic to each other by default.

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

$$\{KS\}_{KB}, \{KS\}_{KC}, \{M, \{h(M)\}_{KA}^{-1}\}_{KS}$$

The management of keys is left to the user, the rationale being that a single centralized certification authority would become such an attractive target that it would likely be cracked or come under legal coercion. So the intended mode of operation is that each user collects the public keys of people she intends to correspond with and bundles them into a *public keyring* that she keeps on her system. The public keys can be authenticated by any convenient method such as by printing them on her business card; to make this easier, PGP supports a *key fingerprint* which is a one-way hash of the public key, presented as a hexadecimal string. Another mechanism is for users to sign each others' keys. This may simply be used as an integrity-protection mechanism on their public keyrings, but becomes more interesting if the signatures are exported. The set of publicly visible PGP signatures makes up the *web of trust*, which may be thought of as an early form of social network of people interested in cryptography. Yet another mechanism was to establish key servers; yet as anyone could upload any key, we ended up with keys for addresses such as `president@whitehouse.gov` not controlled by the people you might think. Colleagues and I also published a book of important public keys [67].

Many things were learned from the deployment and use of PGP during the 1990s. One of the most significant was usability. In a seminal paper, Alma Whitten and Doug Tygar did a cognitive walkthrough analysis of PGP 5.0 followed by a lab test, to assess whether motivated but cryptologically unsophisticated users could understand what was going on well enough to drive the program safely — to understand the need to generate a public/private keypair, figure out how to do so, encrypt messages and sign keys as needed, and make gross errors such as accidentally failing to encrypt, or trusting the wrong public keys. The analysis showed unsafe design decisions and defaults, such as downloading keys from the MIT server without making clear that this was happening. The actual test threw up much worse horrors. Only four of twelve subjects were able to correctly send encrypted email to the other subjects, and only three of them expressed any doubt about keys from the key server. Every subject made at least one significant error [1342]. The moral is that if you're going to get people without degrees in math or computer science to use encryption, you have to bury it transparently in another product (such as an online computer game) or you have to train them — and test them afterwards. So PGP and similar products can be an option for human-rights workers (and are used by them); but for lonely converts in a hostile country, encryption alone is questionable.

There may be other reasons why encrypting email is only part of the solution. In some countries, including Russia, Zimbabwe and the UK, the police have the power to require you to decrypt ciphertext they seize, or even hand over the key. This power is also available to the civil courts in many more countries, and to many tax authorities. Other situations in which coercion may be a problem include where soldiers or intelligence agents could be captured; where police power is abused, for example to seize a key on the grounds of a supposed criminal investigation but where in reality they've been bribed to obtain commercially confidential information; and even in private homes (kids can be abused by parents, as I noted in the chapter on medical privacy, and householders are sometimes tortured by robbers to get bank card PINs and to open safes [1326]).

Making encryption resistant to *rubber hose cryptanalysis*, as it's called, is hard, but it's possible at least to block access to old messages. For example, the U.S. Defense Messaging System supports the use of public encryption keys only once. Each user has a key server that will provide a fresh public encryption key on demand, signed by the user's signing key, and once the user receives and decrypts the message he destroys the decryption key. This forward secrecy property is also found in Skype; beating someone's passphrase out of them doesn't let you decipher old conversations. As for stored data, making that coercion-resistant brings us to the topic of steganography.

23.4.5 Steganography and Forensics Countermeasures

When your threat model includes coercion, simply destroying old keys may not always be enough, as the very existence of protected material can be sufficient to cause harm. In such circumstances, more complete plausible deniability can be provided by the use of *steganography*, which is about hiding data in other data. As an example, Fabien Petitcolas wrote a program called MP3stego, which will take some information you want to hide (such as an encrypted message) and hide it in audio: it takes an audio track and compresses it using the MP3 algorithm, and wherever it can make a random choice about the compression it uses this to hide the next bit of message. And the CIA is reported to have had a camera that hid data in the least significant bits of randomly-selected pixels [1020]. There are many steganography programs floating around on the net, but most of them are easy to break: they simply hide your message in the least-significant bits of an audio or video file, and that's easy to detect. Recall our discussion of steganography theory in section 22.4: the two participants, Alice and Bob, have to communicate via a warden, Willie, who wins the game if he can detect even the existence of a hidden message.

The classic use of steganography is hiding sensitive data (such as ciphertext, where that arouses suspicion) in innocuous communications, though increasingly nowadays people worry about protecting stored data. Most customs authorities have the power to require travellers to decrypt any material found on the hard disk of their laptop in order to check for subversive material, pornography and the like. There are many crude ways to hide the existence of files; at most borders it's still enough to have an Apple laptop, or a separate Linux partition on your hard disk which runs Linux, as the normal customs tools don't deal with these. But that problem will be fixed eventually, and against a capable opponent such low-level tricks are likely to be ineffective. Files can be hidden using steganography tools in larger multimedia files, but this is inefficient.

Adi Shamir, Roger Needham and I invented the *steganographic file system*, which has the property that a user may provide it with the name of an object, such as a file or directory, together with a password; and if these are correct for an object in the system, access to it will be provided. However, an attacker who does not have the matching object name and password, and lacks the computational power to guess it, can get no information about whether the named object even exists. This is an even stronger property that Bell-LaPadula; Low cannot even demonstrate the existence of High. In our initial design, the whole disk was encrypted, and fragments of the files are scattered through it at places that depend on the password, with some redundancy to recover from cases where High data is accidentally overwritten by a Low user [75, 856].

In recent years, file-encryption programs such as TrueCrypt have adopted this idea although in TrueCrypt's case the implementation is simpler: each encrypted volume has its free space overwritten with random data, and there may or may not be a hidden volume in there that can be revealed to a user with the right password.

Now TrueCrypt is one of the tools commonly used by human-rights workers; would it be sensible for Firoz to use it too? The answer is, as usual, 'it depends'. If the Iranian religious police normally only find TrueCrypt installed by human-rights workers there, he's likely to be treated as one of them if he's raided and it's found. In general, if the existence of a product is in itself incriminating, he might want to hide that too.

The fundamental problem facing forensic investigators, as I'll discuss in detail later in section 24.5, is the sheer volume of data found when searching a house nowadays: there can be terabytes of data scattered over laptops, mobile phones, cameras, iPods and memory sticks. If you don't want a needle to be found, build a larger haystack. So Firoz might have a lot of electronic junk scattered around his apartment, as cover for the memory stick that actually contains the forbidden pictures stashed in a hidden TrueCrypt container. He might even have some straight porn in the ordinary encrypted volume, so he's got something to give the police if they torture him. And there are many ad-hoc ways in which content can be made inaccessible to the casual searcher; he might damage the memory stick in some repairable way. If he had a forbidden movie in WMV format, he might delete its license from the WMP store — so the license store had to be restored from backup before the movie could be played. (A movie for which a license is no longer available is a much less suspicious kind of ciphertext than a TrueCrypt volume.)

In short, as the world adapts to the digital age, people adopt ways of doing things, and these procedures in turn have weak points, which leads us back to tradecraft. What works will vary from one place and time to another, as it depends on what the local opponents actually do. But there are some principles. For example, anonymity loves company; it's much easier to hide in a crowd than in the middle of a desert. And in some applications, deniability may be enough: *Crowds* was a system in which users group together and do web page forwarding for each other, so that if one of them downloaded a subversive web page, the secret police have several hundred suspects to deal with [1067]. A similar scheme was devised by a well-known company CEO who, each morning, used to borrow at random one of the mobile phones of his managers, and have his switchboard forward his calls.

Forensics are subtly different: cops tend only to have tools for the most popular products. They can usually search for 'obvious' wickedness in Windows PCs but often can't search Macs at all; they have tools to extract address books from the three or four most popular mobile phones, but not obscure makes; they can wiretap the large ISPs but often not the mom-and-pop outfits.

They're also usually a bit behind the curve. They may know how to deal with Facebook now, but they probably didn't in 2004. Cool kids and gadget freaks may always be a few steps ahead.

23.4.6 Putting It All Together

Returning now to our list of typical privacy technology users, what can we say?

1. The missionary, Andrew, has one of the hardest secure communication tasks. He can't meet his converts to train them to use Tor and PGP properly, and religious factors might prevent them communicating covertly by joining an online computer game in which they played the roles of dragons, wizards and so on. Perhaps the simplest solution for him is Skype.
2. In the case of your daughter Betty, all the evidence is that parental concerns over the Internet are grossly over-inflated. Rather than trying to get her to surf the net using Tor (which she'd just consider to be creepy if her friends don't use it too), you'd be better to make that scams, phishing and other abuses into a regular but not obsessive topic of conversation round the dinner table. (Once she gets the confidence to join in the conversation, she may turn out to have better tales than you do.)
3. The corporate engineer, Charles, may find his main risk is that if he posts from a work machine, then even if he's using a throwaway webmail address, he might inadvertently include material in documents such as local printer or workgroup names or a camera serial number that the corporate security guy then finds on Google. The simplest solution is to use home equipment that isn't cross-contaminated with work material. Essentially this is a multilevel policy of the sort discussed in Chapter 8.
4. The human-rights activist Dai has one of the hardest jobs of all, but as she's being regularly shaken down by the police and is in contact with a network of other activists with whom she can share experiences, she at least has an opportunity to evolve good tradecraft over time.
5. The M&A analyst Elizabeth may well find that Tor does pretty well what she needs. Her main problem will be using it properly (even I once found that I'd misconfigured my system so that I thought I was browsing through Tor when I wasn't — and I'm supposed to be a security expert).
6. Firoz is in a pretty bad way, and quite frankly were I in his situation I'd emigrate. If that's not possible then he should not just use Tor, but get a Mac or Linux box so he's less exposed to porn-site malware. Some combination of cryptographic hiding, camouflage and deception may save

his life if he gets raided by the police; and perhaps he should join the Revolutionary Guard so the police won't dare raid him in the first place.

7. Graziano also has an extremely hard job. It's bad enough defending a covert network against one or two traitors at the client end (as Andrew must); defending against occasional treachery at the server side is even harder. Were I designing such a system I'd establish clear public policies and expectations on how informers' identity would be protected, so that any attempt by a future corrupt webmaster to subvert the procedures would be visible. I'd also test the system regularly by having undercover policemen call in as informers, and track their revelations, to spot bent cops who lose information. Where informers did identify themselves — deliberately or accidentally — I'd ensure that only one handler and his supervisor know this.

Wicked people use anonymity too, of course, and the many tales of how they fail underline the difficulty of finding true anonymity in the modern world. In a child-custody case in Taunton, England, the wife's lawyer emailed a bogus legal judgment to the father, pretending the email was from a fathers' rights charity. When the father read this out in court, the lawyer stood up and accused him of forgery. Outraged, the father tracked the email to a shop in London's Tottenham Court Road, where the staff remembered someone coming in to use their service and dug out still images from their CCTV camera which identified the lawyer, Bruce Hyman [397]. Mr Hyman was sent to prison for twelve months at Bristol Crown Court. He was an expert on evidence — and the first British barrister to be imprisoned in modern times for perverting the course of justice.

Richard Clayton wrote a thesis on anonymity and traceability in cyberspace, which Mr Hyman should perhaps have read [300]. There are many ways in which people who try to be anonymous, fail; and there are also many ways in which even people who made no particular effort to hide themselves end up not being traceable. It's hard to establish responsibility when abusive traffic comes from a phone line in a multi-occupied student house, or when someone accesses a dial-up account on a free ISP from a phone whose calling line ID has been blocked. ISPs also often keep quite inadequate logs and can't trace abusive traffic afterwards. So in practice, as opposed to theory, anonymity is already pretty widespread. This may gradually contract over time, because pressure over peer-to-peer traffic, spam and phishing may make ISPs manage things more tightly and respond better to complaints. The view of UK ISPs, for example, is that 'Anonymity should be explicitly supported by relevant tools, rather than being present as a blanket status quo, open to use and misuse' [307].

As privacy technology evolves, it may modify the shape of the trade-off between privacy and surveillance that is conditioned by the much larger-scale development of online technology in general. Privacy technology will be driven to some extent by the desire to evade copyright, by various political liberation agendas, and by criminal innovation. Tools invented to protect the privacy of the law-abiding, and of foreign lawbreakers whose subversion we support, will be used occasionally by criminals in our countries too. So far, there's little sign of it, but it's bound to happen eventually. For this reason a number of people have proposed *identity escrow* schemes in which net users have pseudonyms which normally provide identity protection but which can be removed by order of a court [255]. But such systems would put most of the privacy users we discussed in this section directly in harm's way. What's more, escrow mechanisms tend to be expensive and fragile, and cause unforeseen side-effects [4].

In the next chapter I'll describe the 'Crypto Wars' — the long struggle through the 1990s by governments to control cryptography, by demanding that keys be escrowed. Eventually they gave up on that; and the same arguments apply to anonymity systems. I believe we just have to accept that providing privacy to people we approve of means that occasionally some people we don't approve of will use it too. As Whit Diffie, the inventor of digital signatures and a leading crypto warrior, put it: 'If you campaign for liberty you're likely to find yourself drinking in bad company at the wrong end of the bar'.

23.5 Elections

One application of which all democracies by definition approve, and in which almost all mandate anonymity, is the election. However, the move to electronic voting has been highly controversial. In the USA, Congress voted almost four billion dollars to upgrade states' election systems following the Florida fiasco in 2000, and a lot of this money's been wasted on voting machines that turned out to be insecure. There have been similar scandals elsewhere, including the UK and the Netherlands.

Research into electronic election mechanisms goes back to the early 1980s, when David Chaum invented *digital cash* — a payment medium that is anonymous, untraceable and unlinkable [285, 287]. In section 5.7.3 I described the mechanism: a customer presents a banknote to a bank, blinded by a random factor so that the bank can't see the serial number; the bank signs the note; the customer then unblinds it; and she now has an electronic banknote with a serial number the bank doesn't know. There are a few more details you have to fix to get a viable system, such as arranging that the customer's anonymity fails if she spends the banknote twice [287]. Digital cash hasn't succeeded,

as it's not really compatible either with the anti-money-laundering regime or the banks' business models[8]. However the application on which a number of research teams are still working is the digital election. The voter can be given a ballot paper manufactured using the same general techniques as a digital coin; she can spend it with the candidate of his choice; and she can get caught if she cheats by double-spending.

There are a number of further requirements on electronic voting systems, of which perhaps the two most difficult to satisfy simultaneously are that the voter should be able to satisfy herself that her vote has been properly counted and that she should not be able to prove to anybody else how she voted. If she can, then the doors are opened to vote-buying and intimidation. Getting the anonymity and auditability right simultaneously depends on a good combination of physical security and computer-security mechanisms.

Digital elections remained something of an academic backwater until 2000, when the outcome of the U.S. Presidential election turned on a handful of disputed votes in Florida. At the time, I was attending the Applications Security conference in New Orleans, and we organised a debate; it rapidly became clear that, even though politicians thought that mechanical or paper voting systems should be replaced with electronic ones as quickly as possible, security experts didn't agree. A large majority of the attendees — including many NSA and defense contractor staff — voted (on an old-fashioned show of hands) they didn't trust electronic elections[9]. Nonetheless Congress went ahead and passed the Help America Vote Act in 2002, which provided $3.8 billion for states to update their voting equipment.

By the following year, this particular barrel of pork had degenerated into a national scandal. Many problems were reported in the 2002 elections [551]; then, the following summer, the leading voting-machine supplier Diebold left its voting system files on an open web site, a stunning security lapse. Avi Rubin and colleagues at Johns Hopkins trawled through them found that the equipment was far below even minimal standards of security expected in other contexts. Voters could cast unlimited votes, insiders could identify voters, and outsiders could also hack the system [731]. Almost on cue, Diebold CEO Walden O'Dell, who was active in the campaign to re-elect President Bush, and wrote 'I am committed to helping Ohio deliver its electoral votes to the president next year' [1320]. This led to uproar.

Electronic equipment had actually been used for some time to count ballots in a number of U.S. districts, but there are a number of different ways to do

[8] A variant may be used for pseudonymous credentials in Trusted Computing [220].

[9] One of the strongest arguments was a simple question: do you know how to clear the Internet Explorer cache? As the hotel didn't have an Internet connection, we all had to check our email at a café in Bourbon Street that had two PCs, one with Netscape and the other with IE. The attendees preferred Netscape as it was easy to stop the next user retrieving your password from the cache.

this. One option is optical scanning, where paper ballots are used but fed into a system that recognises votes, gets an operator to adjudicate difficult cases, and outputs the tally. This has the advantage that, if the count is challenged, officials (or a court) can send for the original ballots and count them by hand. Another alternative is the ballotmarking machine: the voter makes her choices on a touch screen, after which the machine prints out a voting form that she can inspect visually and drop into a ballot box. Many (but not all) of the problems arose from 'Direct-recording electronic' (DRE) voting systems, in which the voter's choice is entered directly into a terminal that tallies the votes and outputs the result at the end of the day. If the software in such a device is buggy, or open to manipulation, it can give the wrong result; and unless the result is wildly out of kilter with common sense, there's simply no way to tell. The only verification procedure available on many models was to press the 'count' button again to get it to print out the tally again. Even although voting machines are certified by the Federal Election Commission (FEC), the FEC rules don't require that a tally be independently auditable. This is wrong, according to the majority of experts, who now believe that all voting systems should have a voter-verifiable audit trail. This happens automatically with scanning systems; Rebecca Mercuri advocates that DRE equipment should display the voter's choice on a paper roll behind a window and get them to validate it prior to casting. (That was in 1992, and was reiterated in her thesis on electronic voting in 2000 [875, 876].)

The latest round in the U.S. voting saga comes from California, Florida and Ohio. The Californian Secretary of State Debra Bowen authorized and paid for a large team of computer scientists, led by University of California professors David Wagner and Matt Bishop, to do a top-to-bottom evaluation of the state's voting systems, including source code reviews and read-team attacks, in order to decide what equipment could be used in the 2008 elections. The reports, published in May 2007, make depressing reading [215]. All of the voting systems examined contained serious design flaws that led directly to specific vulnerabilities that attackers could exploit to affect election outcomes. All of the previously approved voting machines — by Diebold, Hart and Sequoia — had their certification withdrawn, and were informed they would need to undertake substantial remediation before recertification. A late-submitted system from ES&S was also decertified. California could still take such radical action, as perhaps three-quarters of the 9 million people who voted in 2004 did so using a paper or optical-scan ballot. As this book went to press in December 2007, Ms Bowen had just said that electronic voting systems were still not good enough to be trusted with state elections. 'When the government finds a car is unsafe, it orders a recall', she said. 'Here we're talking about systems used to cast and tally votes, the most basic tool of democracy'. [1343].

A similar inspection of Florida equipment was carried out by scientists at Florida State University; they reported a bundle of new vulnerabilities in the Diebold equipment in July 2007 [514]. Ohio followed suit; their evaluation

of election equipment and standards came to similar conclusions. All the evaluated equipment had serious security failings: data that should have been encrypted wasn't; encryption done badly (for example, the key stored in the clear next to the ciphertext); buffer overflows; useless (and misapplied) physical security; SQL injection; audit logs that could be tampered with; and undocumented back doors [855]. Interestingly, the Florida and Ohio teams found plenty of new vulnerabilities that the California team missed, and all were working quickly; this raises interesting questions about the total number of security flaws in these systems.

Our experience in the UK is broadly similar, although the detail is different. Tony Blair's government progressively expanded the use of postal and other absentee forms of ballot, which was criticised by opposition parties as it made vote-buying and intimidation easier. Party workers (of which Blair's Labour party had more) could pressure voters into opting for a postal ballot, then collect their ballot forms, fill them out, and submit them. Plans to extend voting from the post to email and text were criticised for making this existing low-grade abuse easier and potentially open to automation. Finally, in the May 2007 local government elections, electronic voting pilots were held in eleven areas around the UK. Two of my postdocs acted as scrutineers in the Bedford election, and observed the same kind of shambles that had been reported at various U.S. elections. The counting was slower than with paper; the system (optical-scan software bought from Spain) had a high error rate, resulting in many more ballots than expected being sent to human adjudicators for decision. (This was because the printers had changed the ink halfway through the print run, and half the ballot papers were the wrong shade of black.) Even worse, the software sometimes sent the same ballot paper to multiple adjudicators, and it wasn't clear which of their decisions were counted. In the end, so that everyone could go home, the returning officer accepted a letter of assurance (written on the spot by the vendor) saying that no vote would have been miscounted as a result. Yet the exercise left the representatives from the various parties with serious misgivings. The Open Rights Group, which organised the volunteers, reported that it could not express confidence in the results for the areas observed [987].

There was an interesting twist in the Netherlands. DRE voting machines had been introduced progressively during the 1990s, and cyber-rights activists were worried about the possibility of tampering and fraud along the lines observed in the USA. They discovered that the machines from the leading vendor, Nedap, were vulnerable to a Tempest attack: using simple equipment, an observer sitting outside the polling station could see what party a voter had selected [541]. From the security engineer's perspective this was great stuff, as it led to the declassification by the German intelligence folks of a lot of Cold War tempest material, as I discussed in section 17.4.2 (the Nedap machines are also used in Germany). The activists also got a result: on October 1 2007 the District Court in Amsterdam decertified all the Nedap machines.

As for other countries, the picture is mixed. The OpenNet Initiative (of which I've been a member since 2006) monitors election abuses in the third world. We have found that in some less-developed country elections, the state has systematically censored opposition parties' websites and run denial-of-service attacks; in others (typically the most backward), elections are rigged by more traditional methods such as kidnapping and murdering opposition candidates. The evidence of electronic cheating is less clear-cut but is often suspected. Take for example Russia. I wrote in the first edition in 2001: 'I sincerely hope that the election of Vladimir Putin as the president of Russia had nothing to do with the fact that the national electoral reporting system is run by FAPSI, a Russian signals intelligence agency formed in 1991 as the successor to the KGB's 8th and 16th directorates. Its head, General Starovoitov, was reported to be an old KGB type; his agency reported directly to President Yeltsin, who chose Putin as his successor' [509, 678]. Yet by the time Putin's party was re-elected in 2007, the cheating had become so blatant — with gross media bias and state employees ordered to vote for the ruling party — that the international community would not accept the result as free and fair.

Wherever you go, electronic abuses at election time, and abuses of electronic election equipment, are just one of many tools used by the powerful to hang on to power. It's worth remembering that in Florida in 2000, more voters were disenfranchised as a result of registration abuses than there were ballots disputed because of hanging chads. And just as the government can bias an election by making it harder to vote if you haven't got a car, he could conceivably make it harder to vote if you haven't got a computer. It's not unknown for the ballot to be made so somplex as to disenfranchise the less educated. And large-scale abuses can defeat even technical ballot privacy; for example, in a number of less-developed countries, districts that elected the 'wrong' candidate have been punished. (And although we shake our heads in sorrow when happens in Zimbabwe, we just shrug when a British government channels extra money to schools and hospitals in marginal constituences.) In fact, it has struck me that if an incumbent wants to change not 1% of the votes, but 10% — say to turn a 40–60 defeat into a 60–40 victory — then bribing or bullying voters may provide a more robust result than tinkering with the vote-tallying computer. Voters who've been bribed or bullied are less likely to riot than voters who've been cheated. The bullied voters in Russia didn't riot; the cheated voters in Kenya did.

So high-technology cheating shouldn't get all, or even most, of an election monitor's attention. But it needs to get some. And it behoves citizens to be more sceptical than usual about the latest wizzo technology when it's being sold to us by politicians who hope to get reelected using it. Finally, even where politicians have comfortable majorities and aren't consciously trying to cheat, they are often vulnerable to computer salesmen, as they're scared of being accused of technophobia. It takes geeks to have the confidence to say stop!

23.6 Summary

Some of the most challenging security engineering problems at the beginning of the twenty-first century have to do with the new online applications that are sweeping the world, from online games through search and auctions to social networking. This chapter was really just a helicopter tour of new services and the new cheating strategies they've spawned.

Much of what goes wrong with online services, as with anonymity services and digital elections, is just the same as we've seen elsewhere — the usual sad litany of bugs and blunders, of protection requirements ignored in the rush to market or just not understood by the developers of the early systems. Elections in particular provide a sobering case history of proprietary systems developed for an application that was known to be sensitive, and by developers who made all sorts of security claims; yet once their products were exposed to fresh air and sunlight, they turned out to be terrible.

What's also starting to become clear is that as more and more of human life moves online, so the criticality and the complexity of online applications grow at the same time. Many of the familiar problems come back again and again, in ever less tractable forms. Enforcing privacy is difficult enough in a large hospital, but just about doable. How do you enforce privacy in something as big as Google, or as complex as Facebook? And how do you do security architecture when ever more functionality is provided to ever more people by ever more code written by constantly growing armies of inexperienced programmers? Traditional software engineering tools helped developers get ever further up the complexity mountain before they fell off. How do you see to it that you don't fall off, or if you do, you don't fall too hard?

Research Problems

This leads me to suggest that one of the critical research problems between now and the third edition of this book (if there is one) will be how protection mechanisms scale.

The hard mechanism-design problem may be how one goes about evolving 'security' (or any other emergent property) in a socio-technical system with billions of users. In the simple, million-user government applications of yesteryear, a central authority could impose some elementary rules — a 'Secret' document had to be locked in a filing cabinet when you went to the toilet, and a 'Secret' computer system needed an Orange book evaluation. But such rules were never natural, and people always had to break them to get their

work done. Trying to scale access-control rules to social networking sites like Facebook is probably already beyond the complexity limit, and the revolution has only just started.

In a truly complex socio-technical system you can expect that the rules will evolve in a process whereby the technology and the behaviour continually adapt to each other. But at present the incentives faced by the system developers are also wrong; site operators want search while users want privacy. Governments will want to get in on the act, but they're an order of magnitude too slow and have perverse incentives of their own. So what sorts of mechanisms can be evolved for rule negotiation? Will it simply be survival of the fittest, spiced with the drives of fashion, as one social-networking site replaces another? Or is there some legal framework that might help?

Further Reading

The standard reference on game security at present is by Greg Hoglund and Gary McGraw's book [617]. For the history of computer games, and cheating, read Jeff Yan and Brian Randell in [1367]; Jeff's thesis discusses online bridge [1364]. There's an annual conference, NetGames, which usually has a number of papers on cheating, and the Terra Nova blog on virtual worlds has regular articles on cheating.

The best general work I know of on security in web services is Mike Andrews and James Whitaker's 'How to Break Web Software' [78]. There are many books on specific services, such as John Battelle's book on Google [125] and Adam Cohen's of eBay [310]; and if you need to explain the problem to management, read (or give them) the article by Jeremy Epstein, Scott Matsumoto and Gary McGraw about how web-service developers get software security wrong [436]. As for social networking, I don't think the definitive book has come out yet.

As for privacy technology, the best introduction to anonymous remailers is probably [849]. I don't know of a good treatment of privacy and anonymity technology in real-world contexts; the above vignettes of Andrew and others are my own modest attempt to fill that gap. To go into this subject in depth, look through the anonymity bibliography maintained by Roger Dingledine, Nick Matthewson and George Danezis [82], and the survey of anonymity systems by George Danezis and Claudia Diaz [347]. For traffic analysis, you might start with the survey by Richard Clayton and George Danezis [346].

There's now quite a literature on electronic voting. The issues are largely the same as with voting by mail or by phone, but not quite. An early survey of the requirements, and of the things that can go wrong, was written by Mike

Shamos [1149], who is also a prominent defender of electronic voting [1150]; while Roy Saltzman (for many years the authority at NIST) discusses things that have gone wrong in the USA, and various NIST recommendations, in [1103]. The leading critics of current electronic voting arrangements include David Dill's Verified Voting Foundation, and Rebecca Mercuri, whose 2000 thesis on 'Electronic Vote Tabulation — Checks & Balances' [876] might perhaps have been heeded more, along with an early report on the feasibility of Internet voting from the State of California [253]. Certainly, the recent evaluation reports from California [215], Florida [514], Ohio [855] and Britain [987] lend strong confirmation to the sceptical viewpoint.