CHAPTER

2

Usability and Psychology

Humans are incapable of securely storing high-quality cryptographic keys, and they have unacceptable speed and accuracy when performing cryptographic operations. (They are also large, expensive to maintain, difficult to manage, and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed. But they are sufficiently pervasive that we must design our protocols around their limitations.)

— Kaufmann, Perlman and Speciner [698]

Only amateurs attack machines; professionals target people.

— Bruce Schneier

2.1 Introduction

Many real attacks exploit psychology at least as much as technology. The fastest-growing online crime is *phishing*, in which victims are lured by an email to log on to a website that appears genuine but that's actually designed to steal their passwords. Online frauds like phishing are often easier to do, and harder to stop, than similar real-world frauds because most online protection mechanisms are not anything like as intuitively usable or as difficult to forge convincingly as their real-world equivalents; it is much easier for crooks to build a bogus bank website that passes casual inspection than it is for them to create a bogus bank in a shopping mall.

We've evolved social and psychological tools over millions of years to help us deal with deception in face-to-face contexts, but these are little use to us when we're presented with an email that asks us to do something. It seems to be harder to create useful asymmetry in usability, by which I mean that good use is

easier than bad use. We have some examples of asymmetry in physical objects: a potato peeler is easier to use for peeling potatoes than a knife is, but a lot harder to use for murder. However, much of the asymmetry on which we rely in our daily business doesn't just depend on formal exchanges — which can be automated easily — but on some combination of physical objects, judgment of people, and the supporting social protocols. (I'll discuss this further in the Introduction to Chapter 3.) So, as our relationships with employers, banks and government become more formalised via online communication, and we lose both physical and human context, the forgery of these communications becomes more of a risk.

Deception, of various kinds, is now the greatest threat to online security. It can be used to get passwords, or to compromise confidential information or manipulate financial transactions directly. The most common way for private investigators to steal personal information is *pretexting* — phoning someone who has the information under a false pretext, usually by pretending to be someone authorised to be told it. Such attacks are sometimes known collectively as *social engineering*. There are many other flavours. The quote from Bruce Schneier at the head of this chapter appeared in a report of a stock scam, where a bogus press release said that a company's CEO had resigned and its earnings would be restated. Several wire services passed this on, and the stock dropped 61% until the hoax was exposed [1128]. Hoaxes and frauds have always happened, but the Internet makes some of them easier, and lets others be repackaged in ways that may bypass our existing controls (be they personal intuitions, company procedures or even laws). We will be playing catch-up for some time.

Another driver for the surge in attacks based on social engineering is that people are getting better at technology. As designers learn how to forestall the easier techie attacks, psychological manipulation of system users or operators becomes ever more attractive. So the security engineer simply must understand basic psychology and 'security usability', and one of the biggest opportunities facing the research community is to learn more about what works and why.

2.2 Attacks Based on Psychology

Hacking systems through the people who operate them may be growing rapidly but is not new. Military and intelligence organisations have always targeted each other's staff; most of the intelligence successes of the old Soviet Union were of this kind [77]. Private investigation agencies have not been far behind. The classic attack of this type is pretexting.

2.2.1 Pretexting

Colleagues of mine did an experiment in England in 1996 to determine the threat posed by pretexting to medical privacy. We trained the staff at a health authority (a government-owned health insurer that purchased medical services for a district of maybe 250,000 people) to identify and report false-pretext calls. A typical private eye would pretend to be a doctor involved in the emergency care of a patient, and he could be detected because the phone number he gave wasn't that of the hospital at which he claimed to work. (The story is told in detail later in the chapter on Multilateral Security.) We detected about 30 false-pretext calls a week. Unfortunately, we were unable to persuade the UK government to make this training mandatory for health authority staff. Thirty attacks per week times 52 weeks in a year times 200 health authorities in England is a lot of privacy compromise! Many countries have laws against pretexting, including both the UK and the USA, yet there are people in both countries who earn their living from it [411]. A typical case is reported in [449], where a private eye collecting debts for General Motors was fined for conning civil servants into giving out 250 people's home addresses over the phone.

The year 2002 saw the publication of perhaps the most disturbing security book ever, Kevin Mitnick's *'Art of Deception'*. Mitnick, who got extensive press coverage when he was arrested and convicted after breaking into phone systems, related after his release from prison how almost all of his exploits had involved social engineering. His typical hack was to pretend to a phone company employee that he was a colleague, and solicit 'help' such as a password. Ways of getting past a company's switchboard and winning its people's trust have been taught for years in sales-training courses; Mitnick became an expert at using them to defeat company security procedures, and his book recounts a fascinating range of tricks [896].

Pretexting became world headline news in September 2006 when it emerged that Hewlett-Packard chairwoman Patricia Dunn had hired private investigators who had used pretexting to obtain the phone records of other board members of whom she was suspicious, and of journalists she considered hostile. She was forced to resign. The following month, the California Attorney General filed felony charges and arrest warrants against her and three private eyes. The charges were online crime, wire fraud, taking computer data and using personal identifying information without authorization. In March 2007, charges against her were dropped; a factor was that she was suffering from cancer. Her codefendants pleaded no contest to lesser counts of fraudulent wire communications, a misdemeanor, and got community service [93].

But fixing the problem is hard. Despite continuing publicity about pretexting, there was an audit of the IRS in 2007 by the Treasury Inspector General for Tax Administration, whose staff called 102 IRS employees at all levels, asked for their user ids, and told them to change their passwords to a known value.

62 did so. Now nearly 100,000 IRS employees have access to tax return data, so if you're a US taxpayer there might be 60,000 people who might be fooled into letting an intruder breach your financial privacy. What's worse, this happened despite similar audit tests in 2001 and 2004 [1131]. Now a number of government departments, including Homeland Security, are planning to launch phishing attacks on their own staff in order to gauge the effectiveness of security education. In the UK, the privacy authorities announced a crackdown and prosecuted a private detective agency that did blagging for top law firms [779].

Resisting attempts by outsiders to inveigle your staff into revealing secrets is known in military circles as *operational security*. Protecting really valuable secrets, such as unpublished financial data, not-yet-patented industrial research or military plans, depends on limiting the number of people with access, and also having strict doctrines about with whom they may be discussed and how. It's not enough for rules to exist; you have to train all the staff who have access to the confidential material, and explain to them the reasons behind the rules. In our medical privacy example, we educated staff about pretexting and trained them not to discuss medical records on the phone unless they had initiated the call, and made it to a number they had got from the phone book rather than from a caller. And once the staff have encountered, detected and defeated a few pretexting attempts, they talk about it and the message gets across loud and clear. Often the hardest people to educate are the most senior; a consultancy sent the finance directors of 500 publicly-quoted companies a USB memory stick as part of an anonymous invitation saying 'For Your Chance to Attend the Party of a Lifetime', and 46% of them put it into their computers [701].

Intelligence-agency rules are very much tougher. Most of the operational security effort goes into training staff in what not to do, instilling a culture of discretion that shades well over into anonymity. And since foreign intelligence agencies make many fewer approaches to spooks than private eyes make to medical-record clerks, a spymaster can't rely on a robust detection culture to spring up of its own accord. He has to have his own red team constantly testing his staff to ensure that they take the paranoia business seriously.

Some operational security measures are common sense, such as not tossing out sensitive stuff in the trash. Less obvious is the need to train the people you trust, even if they're old friends. A leak of embarrassing emails that appeared to come from the office of the UK Prime Minister and was initially blamed on 'hackers' turned out to have been fished out of the trash at his personal pollster's home by a private detective called 'Benji the Binman' who achieved instant celebrity status [828]. Governments have mostly adopted a set of procedures whereby sensitive information is 'classified' and can only be passed to people with an appropriate 'clearance', that is, background checks

and training. While this can become monstrously bureaucratic and wasteful, it does still give a useful baseline for thinking about operational security, and has led to the development of some protection technologies which I'll discuss later in the chapter on Multilevel Security. The disciplines used by banks to prevent a rogue from talking a manager into sending him money are similar in spirit but differ in detail; I discuss them in the chapter on Banking and Bookkeeping.

Pretexting is mostly used for attacks on companies, but it's starting to be used more against individuals. Here's the scam du jour in the USA, as I write this in 2007: the bad guy phones you pretending to be a court official, tells you you've been selected for jury duty, and demands your SSN and date of birth. If you tell him, he applies for a credit card in your name. If you tell him to get lost, he threatens you with arrest and imprisonment. Not everyone has the self-confidence and legal knowledge to resist this kind of sting.

2.2.2 Phishing

Phishing is in many ways a harder problem for a company to deal with than pretexting, since (as with the last scam I mentioned) the targets are not your staff but your customers. It is difficult enough to train the average customer — and you can't design simply for the average. If your security systems are unusable by people who don't speak English well, or who are dyslexic, or who have learning difficulties, you are asking for serious legal trouble, at least if you do business in civilised countries.

Phishing attacks against banks started in 2003, with half-a-dozen attempts reported [299]. The early attacks imitated bank websites, but were both crude and greedy; the attackers asked for all sorts of information such as ATM PINs, and were also written in poor English. Most customers smelt a rat. The attackers now use better psychology; they often reuse genuine bank emails, with just the URLs changed, or send an email saying something like 'Thank you for adding a new email address to your PayPal account' to provoke the customer to log on to complain that they hadn't. Of course, customers who use the provided link rather than typing in `www.paypal.com` or using an existing bookmark are likely to get their accounts emptied.

Losses are growing extremely rapidly (maybe $200 m in the USA in 2006, £35 m / $70 m in the UK) although they are hard to tie down exactly as some banks try to hold the customer liable and/or manipulate the accounting rules to avoid reporting frauds. The phishing business has plenty room for growth. Most UK losses in 2006 were sustained by one bank, while in the USA there are perhaps half-a-dozen principal victims. We are only just starting to see large-scale attacks on firms like eBay and Amazon, but I'm sure we will see many more; when compromising a password lets you change the target's email and

street addresses to your own, and then use their credit card to order a wide-screen TV, the temptation is clear.

If you are a bank or an online retail business, then a number of factors influence whether you get targeted. Some have to do with whether you're thought to be a wimp; banks that pursue fraudsters viciously and relentlessly in the courts, well past the point of economic rationality, seem able to deter attacks. The phishermen also prefer banks whose poor internal controls allow large amounts of money to be moved abroad quickly, which lack effective intrusion alarms, which take several days to check whether suspicious payments were authorised, and which don't try too hard to retrieve those that weren't. (I will discuss internal controls later — see the chapter on Banking and Bookkeeping.)

In the rest of this chapter, I'll first visit some relevant basic psychology and then apply it to the study of passwords — how you get users to choose good passwords and enter them accurately, and what you can do to stop users disclosing them to third parties. Finally there will be a brief section on CAPTCHAs, the tests websites use to check that a user is a human rather than a robot; these provide another angle on the differences between human minds and software.

2.3 Insights from Psychology Research

I expect the interaction between security and psychology to be a big research area over the next five years, just as security economics has been over the last five. This is not just because of the growing number of attacks that target users instead of (or as well as) technology. For example, terrorism is largely about manipulating perceptions of risk; and even outside the national-security context, many protection mechanisms are sold using scaremongering. (I'll return to the broader policy issues in Part III.)

Psychology is a huge subject, ranging from neuroscience through to clinical topics, and spilling over into cognate disciplines from philosophy through artificial intelligence to sociology. Although it has been studied for much longer than computer science, our understanding of the mind is much less complete: the brain is so much more complex. We still do not understand one central problem — the nature of consciousness. We know that 'the mind is what the brain does', yet the mechanisms that underlie our sense of self and of personal history remain quite obscure.

Nonetheless a huge amount is known about the functioning of the mind and the brain, and I expect we'll get many valuable insights once we get psychologists working together with security researchers on real problems. In what follows I can only offer a helicopter tour of some ideas that appear relevant to our trade.

2.3.1 What the Brain Does Worse Than the Computer

Cognitive psychology deals with how we think, remember, make decisions and even daydream. There are many well-known results; for example, it is easier to memorise things that are repeated frequently, and it is easier to store things in context. However, many of these insights are poorly understood by systems developers. For example, most people have heard of George Miller's result that human short-term memory can cope with about seven (plus or minus two) simultaneous choices [891] and, as a result, many designers limit menu choices to about five. But this is not the right conclusion to draw. People search for information first by recalling where to look, and then by scanning; once you have found the relevant menu, scanning ten items is only twice as hard as scanning five. The real limit on menu size is with spoken menus, where the average user has difficulty dealing with more than three or four choices [1039].

Our knowledge in this field has been significantly enhanced by the empirical know-how gained not just from lab experiments, but from the iterative improvement of fielded systems. As a result, the centre of gravity has been shifting from applied psychology to the human-computer interaction (HCI) research community. HCI researchers not only model and measure human performance, including perception, motor control, memory and problem-solving; they have also developed an understanding of how people's mental models of systems work, and of the techniques (such as task analysis and cognitive walkthrough) that we can use to explore how people learn to use systems and understand them.

Security researchers need to find ways of turning these ploughshares into swords (the bad guys are already working on it). There are some obvious low-hanging fruit; for example, the safety research community has done a lot of work on characterising the errors people make when operating equipment [1060]. It's said that 'to err is human' and error research confirms this: the predictable varieties of human error are rooted in the very nature of cognition. The schemata, or mental models, that enable us to recognise people, sounds and concepts so much better than computers do, also make us vulnerable when the wrong model gets activated.

Human errors made while operating equipment fall into broadly three categories, depending on where they occur in the 'stack': slips and lapses at the level of skill, mistakes at the level of rules, and mistakes at the cognitive level.

- Actions performed often become a matter of skill, but this comes with a downside: inattention can cause a practised action to be performed instead of an intended one. We are all familiar with such *capture errors*; an example is when you intend to go to the supermarket on the way

home from work but take the road home by mistake — as that's what you do most days. In computer systems, people are trained to click 'OK' to pop-up boxes as that's often the only way to get the work done; some attacks have used the fact that enough people will do this even when they know they shouldn't. (Thus Apple, unlike Microsoft, makes you enter your password when installing software — as this is something you do less often, you might just pause for thought.) Errors also commonly follow interruptions and perceptual confusion. One example is the *post-completion error*: once they've accomplished their immediate goal, people are easily distracted from tidying-up actions. More people leave cards behind in ATMs that give them the money first and the card back second.

- Actions that people take by following rules are open to errors when they follow the wrong rule. Various circumstances — such as information overload — can cause people to follow the strongest rule they know, or the most general rule, rather than the best one. Examples of phishermen getting people to follow the wrong rule include using `https` (because 'it's secure') and starting URLs with the impersonated bank's name, as `www.citibank.secureauthentication.com` — looking for the name being for many people a stronger rule than parsing its position.
- The third category of mistakes are those made by people for cognitive reasons — they simply don't understand the problem. For example, Microsoft's latest (IE7) anti-phishing toolbar is easily defeated by a picture-in-picture attack, which I'll describe later.

What makes security harder than safety is that we have a sentient attacker who will try to provoke exploitable errors.

What can the defender do? Well, we expect the attacker to use errors whose effect is predictable, such as capture errors. We also expect him to look for, or subtly create, exploitable dissonances between users' mental models of a system and its actual logic. Given a better understanding of this, we might try to engineer countermeasures — perhaps a form of cognitive walkthrough aimed at identifying attack points, just as a code walkthough can be used to search for software vulnerabilities.

2.3.2 Perceptual Bias and Behavioural Economics

Perhaps the most promising field of psychology for security folks to mine in the short term is that which studies the heuristics that people use, and the biases that influence them, when making decisions. This discipline, known as *behavioural economics* or *decision science*, sits at the boundary of psychology and economics. It examines the ways in which people's decision processes depart from the rational behaviour modeled by economists; Daniel Kahneman

won the Nobel Prize in economics in 2002 for launching this field (along with the late Amos Tversky). One of his insights was that the heuristics we use in everyday judgement and decision making lie somewhere between rational thought and the unmediated input from the senses [679].

Kahneman and Tversky did extensive experimental work on how people made decisions faced with uncertainty. They developed *prospect theory* which models risk aversion, among other things: in many circumstances, people dislike losing $100 they already have more than they value winning $100. That's why marketers talk in terms of 'discount' and 'saving' — by *framing* an action as a gain rather than as a loss makes people more likely to take it. We're also bad at calculating probabilities, and use all sorts of heuristics to help us make decisions: we base inferences on familiar or easily-imagined analogies (the *availability heuristic* whereby easily-remembered data have more weight in mental processing), and by comparison with recent experiences (the *anchoring effect* whereby we base a judgement on an initial guess or comparison and then adjust it if need be). We also worry too much about unlikely events.

The channels through which we experience things also matter (we're more likely to be sceptical about things we've heard than about things we've seen). Another factor is that we evolved in small social groups, and the behaviour appropriate here isn't the same as in markets; indeed, many frauds work by appealing to our atavistic instincts to trust people more in certain situations or over certain types of decision. Other traditional vices now studied by behavioural economists range from our tendency to procrastinate to our imperfect self-control.

This tradition is not just relevant to working out how likely people are to click on links in phishing emails, but to the much deeper problem of the public perception of risk. Many people perceive terrorism to be a much worse threat than food poisoning or road traffic accidents: this is irrational, but hardly surprising to a behavioural economist, as we overestimate the small risk of dying in a terrorist attack not just because it's small but because of the visual effect of the 9/11 TV coverage and the ease of remembering the event. (There are further factors, which I'll discuss in Chapter 24 when we discuss terrorism.)

The misperception of risk underlies many other public-policy problems. The psychologist Daniel Gilbert, in an article provocatively entitled 'If only gay sex caused global warming', discusses why we are much more afraid of terrorism than of climate change. First, we evolved to be much more wary of hostile intent than of nature; 100,000 years ago, a man with a club (or a hungry lion) was a much worse threat than a thunderstorm. Second, global warming doesn't violate anyone's moral sensibilities; third, it's a long-term threat rather than a clear and present danger; and fourth, we're sensitive to rapid changes in the environment rather than slow ones [526].

Bruce Schneier lists more biases: we are less afraid when we're in control, such as when driving a car, as opposed to being a passenger in a car or

airplane; we are more afraid of risks to which we've been sensitised, for example by gruesome news coverage; and we are more afraid of uncertainty, that is, when the magnitude of the risk is unknown (even when it's small). And a lot is known on the specific mistakes we're likely to make when working out probabilities and doing mental accounting [1129, 1133].

Most of us are not just more afraid of losing something we have, than of not making a gain of equivalent value, as prospect theory models. We're also risk-averse in that most people opt for a bird in the hand rather than two in the bush. This is thought to be an aspect of *satisficing* — as situations are often too hard to assess accurately, we have a tendency to plump for the alternative that's 'good enough' rather than face the cognitive strain of trying to work out the odds perfectly, especially when faced with a small transaction. Another aspect of this is that many people just plump for the standard configuration of a system, as they assume it will be good enough. This is one reason why secure defaults matter[1].

There is a vast amount of material here that can be exploited by the fraudster and the terrorist, as well as by politicians and other marketers. And as behavioural psychology gets better understood, the practice of marketing gets sharper too, and the fraudsters are never far behind. And the costs to business come not just from crime directly, but even more from the fear of crime. For example, many people don't use electronic banking because of a fear of fraud that is exaggerated (at least in the USA with its tough consumer-protection laws): so banks pay a fortune for the time of branch and call-center staff. So it's not enough for the security engineer to stop bad things happening; you also have to reassure people. The appearance of protection can matter just as much as the reality.

2.3.3 Different Aspects of Mental Processing

Many psychologists see the mind as composed of interacting rational and emotional components — 'heart' and 'head', or 'affective' and 'cognitive' systems. Studies of developmental biology have shown that, from an early age, we have different mental processing systems for social phenomena (such as recognising parents and siblings) and physical phenomena. Paul Bloom has written a provocative book arguing that the tension between them explains why many people are natural dualists — that is, they believe that mind and body are basically different [194]. Children try to explain what they see using their understanding of physics, but when this falls short, they explain phenomena in terms of deliberate action. This tendency to look for affective

[1] In fact, behavioral economics has fostered a streak of libertarian paternalism in the policy world that aims at setting good defaults in many spheres. An example is the attempt to reduce poverty in old age by making pension plans opt-out rather than opt-in.

explanations in the absence of material ones has survival value to the young, as it disposes them to get advice from parents or other adults about novel natural phenomena. According to Bloom, it has a significant side-effect: it predisposes humans to believe that body and soul are different, and thus lays the ground for religious belief. This argument may not overwhelm the faithful (who can retort that Bloom simply stumbled across a mechanism created by the Intelligent Designer to cause us to have faith in Him). But it may have relevance for the security engineer.

First, it goes some way to explaining the *fundamental attribution error* — people often err by trying to explain things by intentionality rather than by situation. Second, attempts to curb phishing by teaching users about the gory design details of the Internet — for example, by telling them to parse URLs in emails that seem to come from a bank — will be of limited value if users get bewildered. If the emotional is programmed take over whenever the rational runs out, then engaging in a war of technical measures and countermeasures with the phishermen is fundamentally unsound. Safe defaults would be better — such as 'Our bank will never, ever send you email. Any email that purports to come from us is fraudulent.'

It has spilled over recently into behavioural economics via the *affect heuristic*, explored by Paul Slovic and colleagues [1189]. The idea is that by asking an emotional question (such as 'How many dates did you have last month?') you can get people to answer subsequent questions using their hearts more than their minds, which can make people insensitive to probability. This work starts to give us a handle on issues from people's risky behaviour with porn websites to the use of celebrities in marketing (and indeed in malware). Cognitive overload also increases reliance on affect: so a bank that builds a busy website may be able to sell more life insurance, but it's also likely to make its customers more vulnerable to phishing. In the other direction, events that evoke a feeling of dread — from cancer to terrorism — scare people more than the naked probabilities justify.

Our tendency to explain things by intent rather than by situation is reinforced by a tendency to frame decisions in social contexts; for example, we're more likely to trust people against whom we can take vengeance. (I'll discuss evolutionary game theory, which underlies this, in the chapter on Economics.)

2.3.4 Differences Between People

Most information systems are designed by men, and yet over half their users may be women. Recently people have realised that software can create barriers to females, and this has led to research work on 'gender HCI' — on how software should be designed so that women as well as men can use it effectively. For example, it's known that women navigate differently from men in the real world, using peripheral vision more, and it duly turns

out that larger displays reduce gender bias. Other work has focused on female programmers, especially end-user programmers working with tools like spreadsheets. It turns out that women tinker less than males, but more effectively [139]. They appear to be more thoughtful, but lower self-esteem and higher risk-aversion leads them to use fewer features. Given that many of the world's spreadsheet users are women, this work has significant implications for product design.

No-one seems to have done any work on gender and security usability, yet reviews of work on gender psychology (such as [1012]) suggest many points of leverage. One formulation, by Simon Baron-Cohen, classifies human brains into type S (systematizers) and type E (empathizers) [120]. Type S people are better at geometry and some kinds of symbolic reasoning, while type Es are better at language and multiprocessing. Most men are type S, while most women are type E, a relationship that Baron-Cohen believes is due to fetal testosterone levels. Of course, innate abilities can be modulated by many developmental and social factors. Yet, even at a casual reading, this material makes me suspect that many security mechanisms are far from gender-neutral. Is it unlawful sex discrimination for a bank to expect its customers to detect phishing attacks by parsing URLs?

2.3.5 Social Psychology

This discipline attempts to explain how the thoughts, feelings, and behaviour of individuals are influenced by the actual, imagined, or implied presence of others. It has many aspects, from the identity that people derive from belonging to groups, through the self-esteem we get by comparing ourselves with others. It may be particularly useful in understanding persuasion; after all, deception is the twin brother of marketing. The growth of social-networking systems will lead to peer pressure being used as a tool for deception, just as it is currently used as a tool for marketing fashions.

Social psychology has been entangled with the security world longer than many other parts of psychology through its relevance to propaganda, interrogation and aggression. Three particularly famous experiments in the 20th century illuminated this. In 1951, Solomon Asch showed that people could be induced to deny the evidence of their own eyes in order to conform to a group. Subjects judged the lengths of lines after hearing wrong opinions from other group members, who were actually the experimenter's associates. Most subjects gave in and conformed, with only 29% resisting the bogus majority [90].

Stanley Milgram was inspired by the 1961 trial of Adolf Eichmann to investigate how many experimental subjects were prepared to administer severe electric shocks to an actor playing the role of a 'learner' at the behest of an experimenter playing the role of the 'teacher' — even when the 'learner'

appeared to be in severe pain and begged the subject to stop. This experiment was designed to measure what proportion of people will obey an authority rather than their conscience. Most will — consistently over 60% of people will do downright immoral things if they are told to [888].

The third of these was the Stanford Prisoner Experiment which showed that normal people can behave wickedly even in the absence of orders. In 1971, experimenter Philip Zimbardo set up a 'prison' at Stanford where 24 students were assigned at random to the roles of 12 warders and 12 inmates. The aim of the experiment was to discover whether prison abuses occurred because warders (and possibly prisoners) were self-selecting. However, the students playing the role of warders rapidly became sadistic authoritarians, and the experiment was halted after six days on ethical grounds [1377].

Abuse of authority, whether real or ostensible, is a major issue for people designing operational security measures. During the period 1995–2005, a hoaxer calling himself 'Officer Scott' ordered the managers of over 68 US stores and restaurants in 32 US states (including at least 17 McDonalds' stores) to detain some young employee on suspicion of theft and strip-search her or him. Various other degradations were ordered, including beatings and sexual assaults [1351]. A former prison guard was tried for impersonating a police officer but acquitted. At least 13 people who obeyed the caller and did searches were charged with crimes, and seven were convicted. MacDonald's got sued for not training its store managers properly, even years after the pattern of hoax calls was established; and in October 2007, a jury ordered McDonalds to pay $6.1 million dollars to Louise Ogborn, one of the victims, who had been strip-searched when an 18-year-old employee. It was an unusually nasty case, as the victim was then left by the store manager in the custody of her boyfriend, who forced her to perform oral sex on him. The boyfriend got five years, and the manager pleaded guilty to unlawfully detaining Ogborn. When it came to the matter of damages, McDonalds argued that Ogborn was responsible for whatever damages she suffered for not realizing it was a hoax, and that the store manager had failed to apply common sense. A Kentucky jury didn't buy this and ordered McDonalds to pay up. The store manager also sued, saying she too was the victim of McDonalds' negligence to warn her of the hoax, and got $1.1 million [740]. So as of 2007, US employers seem to have a legal duty to train their staff to resist pretexting.

But what about a firm's customers? There is a lot of scope for phishermen to simply order bank customers to reveal their security data. Bank staff routinely tell their customers to do this, even when making unsolicited calls. I've personally received an unsolicited call from my bank saying 'Hello, this is Lloyds TSB, can you tell me your mother's maiden name?' and caused the caller much annoyance by telling her to get lost. Most people don't, though. ATM card thieves already called their victims in the 1980s and, impersonating

bank or police officers, have ordered them to reveal PINs 'so that your card can be deactivated'. The current scam — as of December 2007 — is that callers who pretend to be from Visa say they are conducting a fraud investigation. After some rigmarole they say that some transactions to your card were fraudulent, so they'll be issuing a credit. But they need to satisfy themselves that you are still in possession of your card: so can you please read out the three security digits on the signature strip? A prudent system designer will expect a lot more of this, and will expect the courts to side with the customers eventually. If you train your customers to do something that causes them to come to harm, you can expect no other outcome.

Another interesting offshoot of social psychology is cognitive dissonance theory. People are uncomfortable when they hold conflicting views; they seek out information that confirms their existing views of the world and of themselves, and try to reject information that conflicts with their views or might undermine their self-esteem. One practical consequence is that people are remarkably able to persist in wrong courses of action in the face of mounting evidence that things have gone wrong [1241]. Admitting to yourself or to others that you were duped can be painful; hustlers know this and exploit it. A security professional should 'feel the hustle' — that is, be alert for a situation in which recently established social cues and expectations place you under pressure to 'just do' something about which you'd normally have reservations, so that you can step back and ask yourself whether you're being had. But training people to perceive this is hard enough, and getting the average person to break the social flow and say 'stop!' is really hard.

2.3.6 What the Brain Does Better Than the Computer

Psychology isn't all doom and gloom for our trade, though. There are tasks that the human brain performs much better than a computer. We are extremely good at recognising other humans visually, an ability shared by many primates. We are good at image recognition generally; a task such as 'pick out all scenes in this movie where a girl rides a horse next to water' is trivial for a human child yet a hard research problem in image processing. We're also better than machines at understanding speech, particularly in noisy environments, and at identifying speakers.

These abilities mean that it's possible to devise tests that are easy for humans to pass but hard for machines — the so-called 'CAPTCHA' tests that you often come across when trying to set up an online account or posting to a bulletin board. I will describe CAPTCHAs in more detail later in this chapter. They are a useful first step towards introducing some asymmetry into the interactions between people and machines, so as to make the bad guy's job harder than the legitimate user's.

2.4 Passwords

In this section, I will focus on the management of passwords as a simple, important and instructive context in which usability, applied psychology and security meet. Passwords are one of the biggest practical problems facing security engineers today. In fact, as the usability researcher Angela Sasse puts it, it's hard to think of a worse authentication mechanism than passwords, given what we know about human memory: people can't remember infrequently-used, frequently-changed, or many similar items; we can't forget on demand; recall is harder than recognition; and non-meaningful words are more difficult. The use of passwords imposes real costs on business: the UK phone company BT has a hundred people in its password-reset centre.

There are system and policy issues too: as people become principals in more and more electronic systems, the same passwords get used over and over again. Not only may attacks be carried out by outsiders guessing passwords, but by insiders in other systems. People are now asked to choose passwords for a large number of websites that they visit rarely. Does this impose an unreasonable burden?

Passwords are not, of course, the only way of authenticating users to systems. There are basically three options. The person may retain physical control of the device — as with a remote car door key. The second is that she presents something she knows, such as a password. The third is to use something like a fingerprint or iris pattern, which I'll discuss in the chapter on Biometrics. (These options are commonly summed up as 'something you have, something you know, or something you are' — or, as Simson Garfinkel engagingly puts it, 'something you had once, something you've forgotten, or something you once were'.) But for reasons of cost, most systems take the second option; and even where we use a physical token such as a one-time password generator, it is common to use another password as well (whether to lock it, or as an additional logon check) in case it gets stolen. Biometrics are also commonly used in conjunction with passwords, as you can't change your fingerprint once the Mafia gets to know it. So, like it or not, passwords are the (often shaky) foundation on which much of information security is built.

Some passwords have to be 'harder' than others, the principal reason being that sometimes we can limit the number of guesses an opponent can make and sometimes we cannot. With an ATM PIN, the bank can freeze the account after three wrong guesses, so a four-digit number will do. But there are many applications where it isn't feasible to put a hard limit on the number of guesses, such as where you encrypt a document with a password; someone who gets hold of the ciphertext can try passwords till the cows come home. In such applications, we have to try to get people to use longer passwords that are really hard to guess.

In addition to things that are 'obviously' passwords, such as your computer password and your bank card PIN, many other things (and combinations of things) are used for the same purpose. The most notorious are social security numbers, and your mother's maiden name, which many organisations use to recognize you. The ease with which such data can be guessed, or found out from more or less public sources, has given rise to a huge industry of so-called *'identity theft'* [458]. Criminals obtain credit cards, mobile phones and other assets in your name, loot them, and leave you to sort out the mess. In the USA, about half a million people are the 'victims' of this kind of fraud each year[2].

So passwords matter, and managing them is a serious real world problem that mixes issues of psychology with technical issues. There are basically three broad concerns, in ascending order of importance and difficulty:

1. Will the user enter the password correctly with a high enough probability?
2. Will the user remember the password, or will they have to either write it down or choose one that's easy for the attacker to guess?
3. Will the user break the system security by disclosing the password to a third party, whether accidentally, on purpose, or as a result of deception?

2.4.1 Difficulties with Reliable Password Entry

Our first human-factors issue is that if a password is too long or complex, users might have difficulty entering it correctly. If the operation they are trying to perform is urgent, this might have safety implications. If customers have difficulty entering software product activation codes, this can generate expensive calls to your support desk.

One application in which this is important is encrypted access codes. By quoting a reservation number, we get access to a hotel room, a rental car or an airline ticket. Activation codes for software and other products are often alphanumeric representations of encrypted data, which can be a 64-bit or 128-bit string with symmetric ciphers and hundreds of bits when public-key cryptography is used. As the numbers get longer, what happens to the error rate?

[2]I write 'identity theft' in quotes as it's a propaganda term for the old-fashioned offence of impersonation. In the old days, if someone went to a bank, pretended to be me, borrowed money from them and vanished, then that was the bank's problem, not mine. In the USA and the UK, banks have recently taken to claiming that it's my identity that's been stolen rather than their money, and that this somehow makes me liable. So I also parenthesise 'victims' — the banks are the real victims, except insofar as they commit secondary fraud against the customer. There's an excellent discussion of this by Adam Shostack and Paul Syverson in [1166].

An interesting study was done in South Africa in the context of prepaid electricity meters used to sell electricity in areas where the customers have no credit rating and often not even an address. With the most common make of meter, the customer hands some money to a sales agent, and in return gets one or more 20-digit numbers printed out on a receipt. He takes this receipt home and enters the numbers at a keypad in his meter. These numbers are encrypted commands, whether to dispense electricity, to change the tariff or whatever; the meter decrypts them and acts on them.

When this meter was introduced, its designers worried that since a third of the population was illiterate, and since people might get lost halfway through entering the number, the system might be unusable. But it turned out that illiteracy was not a problem: even people who could not read had no difficulty with numbers ('everybody can use a phone', as one of the engineers said). Entry errors were a greater problem, but were solved by printing the twenty digits in two rows, with three and two groups of four digits respectively [59].

A quite different application is the firing codes for U.S. nuclear weapons. These consist of only 12 decimal digits. If they are ever used, the operators may be under extreme stress, and possibly using improvised or obsolete communications channels. Experiments suggested that 12 digits was the maximum that could be conveyed reliably in such circumstances.

2.4.2 Difficulties with Remembering the Password

Our second psychological issue with passwords is that people often find them hard to remember [245, 1379]. Twelve to twenty digits may be fine when they are simply copied from a telegram or a meter ticket, but when customers are expected to memorize passwords, they either choose values which are easy for attackers to guess, or write them down, or both. In fact, the password problem has been neatly summed up as: ''Choose a password you can't remember, and don't write it down.''

The problems are not limited to computer access. For example, one chain of hotels in France introduced completely unattended service. You would turn up at the hotel, swipe your credit card in the reception machine, and get a receipt with a numerical access code which would unlock your room door. To keep costs down, the rooms did not have en-suite bathrooms, so guests had to use communal facilities. The usual failure mode was that a guest, having gone to the bathroom, would forget his access code. Unless he had taken the receipt with him, he'd end up having to sleep on the bathroom floor until the staff arrived the following morning.

Problems related to password memorability can be discussed under four main headings: naive password choice, user abilities and training, design errors, and operational failures.

2.4.3 Naive Password Choice

Since at least the mid-1980s, people have studied what sort of passwords are chosen by users who are left to their own devices. The results are depressing. People will use spouses' names, single letters, or even just hit carriage return giving an empty string as their password. So some systems started to require minimum password lengths, or even check user entered passwords against a dictionary of bad choices. However, password quality enforcement is harder than you might think. Fred Grampp and Robert Morris's classic paper on Unix security [550] reports that after software became available which forced passwords to be at least six characters long and have at least one nonletter, they made a file of the 20 most common female names, each followed by a single digit. Of these 200 passwords, at least one was in use on each of several dozen machines they examined.

A well-known study was conducted by Daniel Klein who gathered 25,000 Unix passwords in the form of encrypted password files and ran cracking software to guess them [720]. He found that 21–25% of passwords could be guessed depending on the amount of effort put in. Dictionary words accounted for 7.4%, common names for 4%, combinations of user and account name 2.7%, and so on down a list of less probable choices such as words from science fiction (0.4%) and sports terms (0.2%). Some of these were straighforward dictionary searches; others used patterns. For example, the algorithm for constructing combinations of user and account names would take an account *'klone'* belonging to the user 'Daniel V. Klein' and try passwords such as klone, klone1, klone 123, dvk, dvkdvk, leinad, neilk, DvkkvD, and so on.

Many firms require users to change passwords regularly, but this tends to backfire. According to one report, when users were compelled to change their passwords and prevented from using the previous few choices, they changed passwords rapidly to exhaust the history list and get back to their favorite password. A response, of forbidding password changes until after 15 days, meant that users couldn't change compromised passwords without help from an administrator [1008]. A large healthcare organisation in England is only now moving away from a monthly change policy; the predictable result was a large number of password resets at month end (to cope with which, sysadmins reset passwords to a well-known value). In my own experience, insisting on alphanumeric passwords and also forcing a password change once a month led people to choose passwords such as 'julia03' for March, 'julia04' for April, and so on.

So when our university's auditors write in their annual report each year that we should have a policy of monthly enforced password change, my response is to ask the chair of our Audit Committee when we'll get a new lot of auditors.

Even among the general population, there is some evidence that many people now choose slightly better passwords; passwords retrieved from phishing

sites typically contain numbers as well as letters, while the average password length has gone up from six to eight characters and the most common password is not 'password' but 'password1' [1130]. One possible explanation is that many people try to use the same password everywhere, and the deployment of password checking programs on some websites trains them to use longer passwords with numbers as well as letters [302].

2.4.4 User Abilities and Training

Sometimes you really can train users. In a corporate or military environment you can try to teach them to choose good passwords, or issue them with random passwords, and insist that passwords are treated the same way as the data they protect. So bank master passwords go in the vault overnight, while military 'Top Secret' passwords must be sealed in an envelope, in a safe, in a room that's locked when not occupied, in a building patrolled by guards. You can run background checks on everyone with access to any terminals where the passwords can be used. You can encrypt passwords along with data in transit between secure sites. You can send guards round at night to check that no-one's left a note of a password lying around. You can operate a clean desk policy so that a password can't be overlooked in a pile of papers on a desk. You can send your guards round the building every night to clean all desks every night.

Even if you're running an e-commerce website, you are not completely helpless: you can give your users negative feedback if they choose bad passwords. For example, you might require that passwords be at least eight characters long and contain at least one nonletter. But you will not want to drive your customers away. And even in the Army, you do not want to order your soldiers to do things they can't; then reality and regulations will drift apart, you won't really know what's going on, and discipline will be undermined. So what can you realistically expect from users when it comes to choosing and remembering passwords?

Colleagues and I studied the benefits that can be obtained by training users [1365]. While writing the first edition of this book, I could not find any account of experiments on this that would hold water by the standards of applied psychology (i.e., randomized controlled trials with big enough groups for the results to be statistically significant). The closest I found was a study of the recall rates, forgetting rates, and guessing rates of various types of password [245]; this didn't tell us the actual (as opposed to likely) effects of giving users various kinds of advice. We therefore selected three groups of about a hundred volunteers from our first year science students.

- The red (control) group was given the usual advice (password at least six characters long, including one nonletter).

- The green group was told to think of a passphrase and select letters from it to build a password. So 'It's 12 noon and I am hungry' would give `'I'S12&IAH'`.
- The yellow group was told to select eight characters (alpha or numeric) at random from a table we gave them, write them down, and destroy this note after a week or two once they'd memorized the password.

What we expected to find was that the red group's passwords would be easier to guess than the green group's which would in turn be easier than the yellow group's; and that the yellow group would have the most difficulty remembering their passwords (or would be forced to reset them more often), followed by green and then red. But that's not what we found.

About 30% of the control group chose passwords that could be guessed using cracking software (which I discuss later), versus about 10 percent for the other two groups. So passphrases and random passwords seemed to be about equally effective. When we looked at password reset rates, there was no significant difference between the three groups. When we asked the students whether they'd found their passwords hard to remember (or had written them down), the yellow group had significantly more problems than the other two; but there was no significant difference between red and green.

The conclusions we drew were as follows.

- For users who follow instructions, passwords based on mnemonic phrases offer the best of both worlds. They are as easy to remember as naively selected passwords, and as hard to guess as random passwords.
- The problem then becomes one of *user compliance*. A significant number of users (perhaps a third of them) just don't do what they're told.

So, while a policy of centrally-assigned, randomly selected passwords may work for the military, its value comes from the fact that the passwords are centrally assigned (thus compelling user compliance) rather than from the fact that they're random (as mnemonic phrases would do just as well).

But centrally-assigned passwords are often inappropriate. When you are offering a service to the public, your customers expect you to present broadly the same interfaces as your competitors. So you must let users choose their own website passwords, subject to some lightweight algorithm to reject passwords that are too short or otherwise 'clearly bad'. In the case of bank cards, users expect a bank-issued initial PIN plus the ability to change the PIN afterwards to one of their choosing (though again you may block a 'clearly bad' PIN such as 0000 or 1234). There can also be policy reasons not to issue passwords: for example, in Europe you can't issue passwords for devices that generate electronic signatures, as this could enable the system administrator to get at the signing key and forge messages, which would destroy the evidential value of the signature. By law, users must choose their own passwords.

So the best compromise will often be a password checking program that rejects 'clearly bad' user choices, plus a training program to get your compliant users to choose mnemonic passwords. Password checking can be done using a program like *crack* to filter user choices; other programs understand language statistics and reject passwords that are too likely to be chosen by others at random [353, 163]; another option is to mix the two ideas using a suitable coding scheme [1207].

2.4.4.1 Design Errors

Attempts to make passwords memorable are a frequent source of severe design errors — especially with the many systems built rapidly by unskilled people in the dotcom rush by businesses to get online.

An important example of how not to do it is to ask for 'your mother's maiden name'. A surprising number of banks, government departments and other organisations authenticate their customers in this way. But there are two rather obvious problems. First, your mother's maiden name is easy for a thief to find out, whether by asking around or using online genealogical databases. Second, asking for a maiden name makes assumptions which don't hold for all cultures, so you can end up accused of discrimination: Icelanders have no surnames, and women from many other countries don't change their names on marriage. Third, there is often no provision for changing 'your mother's maiden name', so if it ever becomes known to a thief your customer would have to close bank accounts (and presumably reopen them elsewhere). And even if changes are possible, and a cautious customer decides that from now on her mother's maiden name is going to be Yngstrom (or even '`yGt5r4ad`') rather than Smith, there are further problems. She might be worried about breaking her credit card agreement, and perhaps invalidating her insurance cover, by giving false data. So smart customers will avoid your business; famous ones, whose mothers' maiden names are in Who's Who, should certainly shun you. Finally, people are asked to give their mother's maiden name to a lot of organisations, any one of which might have a crooked employee. (You could always try to tell 'Yngstrom' to your bank, 'Jones' to the phone company, 'Geraghty' to the travel agent, and so on; but data are shared extensively between companies, so you could easily end up confusing their systems — not to mention yourself).

Some organisations use contextual security information. My bank asks its business customers the value of the last check from their account that was cleared. In theory, this could be helpful: even if someone compromises my password — such as by overhearing me doing a transaction on the telephone — the security of the system usually recovers more or less automatically. The details bear some attention though. When this system was first introduced, I wondered about the risk that a supplier, to whom I'd just written a check, had

a chance of impersonating me, and concluded that asking for the last three checks' values would be safer. But the problem I actually had was unexpected. Having given the checkbook to our accountant for the annual audit, I couldn't authenticate myself to get a balance over the phone. There is also a further liability shift: banks with such systems may expect customers to either keep all statements securely, or shred them. If someone who steals my physical post can also steal my money, I'd rather bank elsewhere.

Many e-commerce sites ask for a password explicitly rather than (or as well as) 'security questions' like a maiden name. But the sheer number of applications demanding a password nowadays exceeds the powers of human memory. Even though web browsers cache passwords, many customers will write passwords down (despite being told not to), or use the same password for many different purposes; relying on your browser cache makes life difficult when you're travelling and have to use an Internet café. The upshot is that the password you use to authenticate the customer of the electronic banking system you've just designed, may be known to a Mafia-operated porn site as well.

Twenty years ago, when I was working in the banking industry, we security folks opposed letting customers choose their own PINs for just this sort of reason. But the marketing folks were in favour, and they won the argument. Most banks allow the customer to choose their own PIN. It is believed that about a third of customers use a birthdate, in which case the odds against the thief are no longer over 3000 to 1 (getting four digits right in three guesses) but a bit over a hundred to one (and much shorter if he knows the victim). Even if this risk is thought acceptable, the PIN might still be set to the same value as the PIN used with a mobile phone that's shared with family members.

The risk you face as a consumer is not just a direct loss through 'identity theft' or fraud. Badly-designed password mechanisms that lead to password reuse can cause you to lose a genuine legal claim. For example, if a thief forges your cash machine card and loots your bank account, the bank will ask whether you have ever shared your PIN with any other person or company. If you admit to using the same PIN for your mobile phone, then the bank can say you were grossly negligent by allowing someone to see you using the phone, or maybe somebody at the phone company did it — so it's up to you to find them and sue them. Eventually, courts may find such contract terms unreasonable — especially as banks give different and conflicting advice. For example, the UK bankers' association has advised customers to change all their PINs to the same value, then more recently that this is acceptable but discouraged; their most recent leaflet also suggests using a keyboard pattern such as 'C' (3179) or 'U' (1793) [84].

Many attempts to find alternative solutions have hit the rocks. One bank sent its customers a letter warning them against writing down their PIN, and instead supplied a distinctive piece of cardboard on which they were supposed

to conceal their PIN in the following way. Suppose your PIN is 2256. Choose a four-letter word, say 'blue'. Write these four letters down in the second, second, fifth and sixth columns of the card respectively, as shown in Figure 2.1. Then fill up the empty boxes with random letters.

1	2	3	4	5	6	7	8	9	0
	b								
	l								
				u					
					e				

Figure 2.1: A bad mnemonic system for bank PINs

This is clearly a bad idea. Even if the random letters aren't written in a slightly different way, a quick check shows that a four by ten matrix of random letters may yield about two dozen words (unless there's an 's' on the bottom row, when you can get 40–50). So the odds that the thief can guess the PIN, given three attempts, have just shortened from 1 in 3000-odd to 1 in 8.

2.4.4.2 Operational Issues

It's not just the customer end where things go wrong. One important case in Britain in the late 1980's was R v Gold and Schifreen. The defendants saw a phone number for the development system for Prestel (an early public email service run by British Telecom) in a note stuck on a terminal at an exhibition. They dialed in later, and found that the welcome screen had an all-powerful maintenance password displayed on it. They tried this on the live system too, and it worked! They proceeded to hack into the Duke of Edinburgh's electronic mail account, and sent mail 'from' him to someone they didn't like, announcing the award of a knighthood. This heinous crime so shocked the establishment that when prosecutors failed to convict the defendants under the laws then in force, parliament passed Britain's first specific computer crime law.

A similar and very general error is failing to reset the default passwords supplied with certain system services. For example, one top-selling dial access system in the 1980's had a default software support user name of 999999 and a password of 9999. It also had a default supervisor name of 777777 with a password of 7777. Most sites didn't change these passwords, and many of them were hacked once the practice became widely known. Failure to change default passwords as supplied by the equipment vendor has affected a wide range of systems. To this day there are web applications running on databases that use well-known default master passwords — and websites listing the defaults for everything in sight.

2.4.5 Social-Engineering Attacks

The biggest practical threat to passwords nowadays is that the user will break system security by disclosing the password to a third party, whether accidentally or as a result of deception. This is the core of the 'phishing' problem.

Although the first phishing attacks happened in 2003, the word 'phishing' itself is older, having appeared in 1996 in the context of the theft of AOL passwords. Even by 1995, attempts to harvest these to send spam had become sufficiently common for AOL to have a 'report password solicitation' button on its web page; and the first reference to 'password fishing' is in 1990, in the context of people altering terminal firmware to collect Unix logon passwords [301][3].

Phishing brings together several threads of attack technology. The first is pretexting, which has long been a practical way of getting passwords and PINs. An old thieves' trick, having stolen a bank card, is to ring up the victim and say 'This is the security department of your bank. We see that your card has been used fraudulently to buy gold coins. I wonder if you can tell me the PIN, so I can get into the computer and cancel it?'

There are many variants. A harassed system administrator is called once or twice on trivial matters by someone who claims to be a very senior manager's personal assistant; once he has accepted her as an employee, she calls and demands a new password for her boss. (See Mitnick's book [896] for dozens more examples.) It even works by email. In a systematic experimental study, 336 computer science students at the University of Sydney were sent an email message asking them to supply their password on the pretext that it was required to 'validate' the password database after a suspected breakin. 138 of them returned a valid password. Some were suspicious: 30 returned a plausible looking but invalid password, while over 200 changed their passwords without official prompting. But very few of them reported the email to authority [556].

Within a tightly-run company, such risks can just about be controlled. We've a policy at our lab that initial passwords are always handed by the sysadmin to the user on paper. Sun Microsystems had a policy that the root password for each machine is a 16-character random alphanumeric string, kept in an envelope with the machine, and which may never be divulged over the phone or sent over the network. If a rule like this is rigidly enforced throughout an organization, it will make any pretext attack on a root password conspicuous. The people who can get at it must be only those who can physically access the machine anyway. (The problem is of course that you have to teach staff not

[3]The first recorded spam is much earlier: in 1865, a London dentist annoyed polite society by sending out telegrams advertising his practice [415]. Manners and other social mechanisms have long lagged behind technological progress!

just a rule, but the reasoning behind the rule. Otherwise you end up with the password stuck to the terminal, as in the Prestel case.)

Another approach, used at the NSA, is to have different colored internal and external telephones which are not connected to each other, and a policy that when the external phone in a room is off-hook, classified material can't even be discussed in the room — let alone on the phone. A somewhat less extreme approach (used at our laboratory) is to have different ring tones for internal and external phones. This works so long as you have alert system administrators.

Outside of controlled environments, things are harder. A huge problem is that many banks and other businesses train their customers to act in unsafe ways. It's not prudent to click on links in emails, so if you want to contact your bank you should type in the URL or use a bookmark — yet bank marketing departments continue to send out emails containing clickable links. Many email clients — including Apple's, Microsoft's, and Google's — make plaintext URLs clickable, and indeed their users may never see a URL that isn't. This makes it harder for banks to do the right thing.

A prudent customer ought to be cautious if a web service directs him somewhere else — yet bank systems can use all sorts of strange URLs for their services. It's not prudent to give out security information over the phone to unidentified callers — yet we all get phoned by bank staff who aggressively demand security information without having any well-thought-out means of identifying themselves. Yet I've had this experience now from two of the banks with which I've done business — once from the fraud department that had got suspicious about a transaction my wife had made. If even the fraud department doesn't understand that banks ought to be able to identify themselves, and that customers should not be trained to give out security information on the phone, what hope is there?

You might expect that a dotcom such as eBay would know better, yet its banking subsidiary PayPal sent its UK customers an email in late 2006 directing them to a competition at `www.paypalchristmas.co.uk`, a domain belonging to a small marketing company I'd never heard of; and despite the fact that they're the most heavily phished site on the web, and boast of the technical prowess of their anti-fraud team when speaking at conferences, the marketing folks seem to have retained the upper hand over the security folks. In November 2007 they sent an email to a colleague of mine which had a sidebar warning him to always type in the URL when logging in to his account — and a text body that asked him to click on a link! (My colleague closed his account in disgust.)

Citibank reassures its customers that it will never send emails to customers to verify personal information, and asks them to disregard and report emails that ask for such information, including PIN and account details. So what happened? You guessed it — it sent its Australian customers an email in October 2006 asking customers 'as part of a security

upgrade' to log on to the bank's website and authenticate themselves using a card number and an ATM PIN [739]. Meanwhile a marketing spam from the Bank of America directed UK customers to `mynewcard.com`. Not only is spam illegal in Britain, and the domain name inconsistent, and clickable links a bad idea; but BoA got the certificate wrong (it was for `mynewcard.bankofamerica.com`). The 'mynewcard' problem had been pointed out in 2003 and not fixed. Such bad practices are rife among major banks, who thereby train their customers to practice unsafe computing — by disregarding domain names, ignoring certificate warnings, and merrily clicking links [399]. As a result, even security experts have difficulty telling bank spam from phish [301].

But perhaps the worst example of all came from Halifax Share Dealing Services, part of a large well-known bank in the UK, which sent out a spam with a URL not registered to the bank. The Halifax's web page at the time sensibly advised its customers not to reply to emails, click on links or disclose details — and the spam itself had a similar warning at the end. The mother of a student of ours received this spam and contacted the bank's security department, which told her it was a phish. The student then contacted the ISP to report abuse, and found that the URL and the service were genuine — although provided to the Halifax by a third party [842]. When even a bank's security department can't tell spam from phish, how are their customers supposed to?

2.4.6 Trusted Path

The second thread in the background of phishing is *trusted path*, which refers to some means of being sure that you're logging into a genuine machine through a channel that isn't open to eavesdropping. Here the deception is more technical than psychological; rather than inveigling a bank customer into revealing her PIN to you by claiming to be a policeman, you steal her PIN directly by putting a false ATM in a shopping mall.

Such attacks go back to the dawn of time-shared computing. A public terminal would be left running an attack program that looks just like the usual logon screen — asking for a user name and password. When an unsuspecting user does this, it will save the password somewhere in the system, reply 'sorry, wrong password' and then vanish, invoking the genuine password program. The user will assume that he made a typing error first time and think no more of it. This is why Windows has a *secure attention sequence*, namely `ctrl-alt-del`, which is guaranteed to take you to a genuine password prompt.

If the whole terminal is bogus, then of course all bets are off. We once caught a student installing modified keyboards in our public terminal room to capture passwords. When the attacker is prepared to take this much trouble,

then all the `ctrl-alt-del` sequence achieves is to make his software design task simpler.

Crooked cash machines and point-of-sale terminals are now a big deal. In one early case in Connecticut in 1993 the bad guys even bought genuine cash machines (on credit), installed them in a shopping mall, and proceeded to collect PINs and card details from unsuspecting bank customers who tried to use them [33]. Within a year, crooks in London had copied the idea, and scaled it up to a whole bogus bank branch [635]. Since about 2003, there has been a spate of *skimmers* — devices fitted over the front of genuine cash machines which copy the card data as it's inserted and use a pinhole camera to record the customer PIN. Since about 2005, we have also seen skimmers that clip on to the wires linking point-of-sale terminals in stores to their PIN pads, and which contain mobile phones to send captured card and PIN data to the crooks by SMS. (I'll discuss such devices in much more detail later in the chapter on Banking and Bookkeeping.)

2.4.7 Phishing Countermeasures

What makes phishing hard to deal with is the combination of psychology and technology. On the one hand, users have been trained to act insecurely by their bankers and service providers, and there are many ways in which people can be conned into clicking on a web link. Indeed much of the marketing industry is devoted to getting people to click on links. In April 2007 there was the first reported case of attackers buying Google AdWords in an attempt to install keyloggers on PCs. This cost them maybe a couple of dollars per click but enabled them to target the PCs of users thinking of setting up a new business [1248].

On the other hand, so long as online service providers want to save money by using the open systems platform provided by web servers and browsers, the technology does not provide any really effective way for users to identify the website into which they are about to enter a password.

Anyway, a large number of phishing countermeasures have been tried or proposed.

2.4.7.1 Password Manglers

A number of people have designed browser plug-ins that take the user-entered password and transparently turn it into a strong, domain-specific password. A typical mechanism is to hash it using a secret key and the domain name of the web site into which it's being entered [1085]. Even if the user always uses the same password (even if he uses 'password' as his password), each web site he visits will be provided with a different and hard-to-guess password that is unique to him. Thus if he mistakenly enters his Citibank password into

a phishing site, the phisherman gets a different password and cannot use it to impersonate him.

This works fine in theory but can be tricky to implement in practice. Banks and shops that use multiple domain names are one headache; another comes from the different rules that websites use for password syntax (some insist on alphanumeric, others alpha; some are case sensitive and others not; and so on). There is also a cost to the user in terms of convenience: roaming becomes difficult. If only your home machine knows the secret key, then how do you log on to eBay from a cyber-café when you're on holiday?

2.4.7.2 Client Certs or Specialist Apps

One of the earliest electronic banking systems I used was one from Bank of America in the 1980s. This came as a bootable floppy disk; you put it in your PC, hit `ctrl-alt-del`, and your PC was suddenly transformed into a bank terminal. As the disk contained its own copy of the operating system, this terminal was fairly secure. There are still some online banking systems (particularly at the corporate end of the market) using such bespoke software. Of course, if a bank were to give enough customers a special banking application for them to be a worthwhile target, the phishermen will just tell them to 'please apply the attached upgrade'.

A lower-cost equivalent is the *client certificate*. The SSL protocol supports certificates for the client as well as the server. I'll discuss the technical details later, but for now a certificate is supposed to identify its holder to the other principals in a transaction and to enable the traffic between them to be securely encrypted. Server certificates identify web sites to your browser, causing the lock icon to appear when the name on the certificate corresponds to the name in the toolbar. Client certificates can be used to make the authentication mutual, and some UK stockbrokers started using them in about 2006. As of 2007, the mechanism is still not bulletproof, as certification systems are a pain to manage, and Javascript can be used to fool common browsers into performing cryptographic operations they shouldn't [1163]. Even once that's fixed, the risk is that malware could steal them, or that the phisherman will just tell the customer 'Your certificates have expired, so please send them back to us for secure destruction'.

2.4.7.3 Using the Browser's Password Database

Choosing random passwords and letting your browser cache remember them can be a pragmatic way of operating. It gets much of the benefit of a password mangler, as the browser will only enter the password into a web page with the right URL (IE) or the same hostname and field name (Firefox). It suffers from some of the same drawbacks (dealing with amazon.com versus amazon.co.uk,

and with roaming). As passwords are stored unencrypted, they are at some small risk of compromise from malware. Whether you use this strategy may depend on whether you reckon the greater risk comes from phishing or from keyloggers. (Firefox lets you encrypt the password database but this is not the default so many users won't invoke it.) I personally use this approach with many low-to-medium-grade web passwords.

Many banks try to disable this feature by setting `autocomplete="off"` in their web pages. This stops Firefox and Internet Explorer storing the password. Banks seem to think this improves security, but I doubt it. There may be a small benefit in that a virus can't steal the password from the browser database, but the phishing defence provided by the browser is disabled — which probably exposes the customer to much greater risk [913].

2.4.7.4 Soft Keyboards

This was a favorite of banks in Latin America for a while. Rather than using the keyboard, they would flash up a keyboard on the screen on which the customer had to type out their password using mouse clicks. The bankers thought the bad guys would not be able to capture this, as the keyboard could appear differently to different customers and in different sessions.

However the phishing suppliers managed to write software to defeat it. At present, they simply capture the screen for 40 pixels around each mouse click and send these images back to the phisherman for him to inspect and decipher. As computers get faster, more complex image processing becomes possible.

2.4.7.5 Customer Education

Banks have put some effort into trying to train their customers to look for certain features in websites. This has partly been due diligence — seeing to it that customers who don't understand or can't follow instructions can be held liable — and partly a bona fide attempt at risk reduction. However, the general pattern is that as soon as customers are trained to follow some particular rule, the phisherman exploit this, as the reasons for the rule are not adequately explained.

At the beginning, the advice was 'Check the English', so the bad guys either got someone who could write English, or simply started using the banks' own emails but with the URLs changed. Then it was 'Look for the lock symbol', so the phishing sites started to use SSL (or just forging it by putting graphics of lock symbols on their web pages). Some banks started putting the last four digits of the customer account number into emails; the phishermen responded by putting in the first four (which are constant for a given bank and card product). Next the advice was that it was OK to click on images, but not on

URLs; the phishermen promptly put in links that appeared to be images but actually pointed at executables. The advice then was to check where a link would really go by hovering your mouse over it; the bad guys then either inserted a non-printing character into the URL to stop Internet Explorer from displaying the rest, or used an unmanageably long URL (as many banks also did).

As I remarked earlier, this sort of arms race is most likely to benefit the attackers. The countermeasures become so complex and counterintuitive that they confuse more and more users — exactly what the phishermen need. The safety and usability communities have known for years that 'blame and train' is not the way to deal with unusable systems–the only remedy is to make the systems properly usable in the first place [972].

2.4.7.6 Microsoft Passport

Microsoft Passport was on the face of it a neat idea — a system for using Microsoft's logon facilities to authenticate the users of any merchant website. Anyone with an account on a Microsoft service, such as Hotmail, could log on automatically to a participating website using a proprietary protocol adapted from Kerberos to send tickets back and forth in cookies.

One downside was that putting all your eggs in one basket gives people an incentive to try to kick the basket over. There were many juicy security flaws. At one time, if you logged in to Passport using your own ID and password, and then as soon as you'd entered that you backtracked and changed the ID to somebody else's, then when the system had checked your password against the file entry for the first ID, it would authenticate you as the owner of the second. This is a classic example of a *race condition* or time-of-check-to-time-of-use (TOCTTOU) vulnerability, and a spectacular one it was too: anyone in the world could masquerade as anyone else to any system that relied on Passport for authentication. Other flaws included cookie-stealing attacks, password reset attacks and logout failures. On a number of occasions, Microsoft had to change the logic of Passport rapidly when such flaws came to light. (At least, being centralised, it could be fixed quickly.)

Another downside came from the business model. Participating sites had to use Microsoft web servers rather than free products such as Apache, and it was feared that Microsoft's access to a mass of data about who was doing what business with which website would enable it to extend its dominant position in browser software into a dominant position in the market for consumer profiling data. Extending a monopoly from one market to another is against European law. There was an industry outcry that led to the establishment of the Liberty Alliance, a consortium of Microsoft's competitors, which developed open protocols for the same purpose. (These are now used

in some application areas, such as the car industry, but have not caught on for general consumer use.)

2.4.7.7 Phishing Alert Toolbars

Some companies have produced browser toolbars that use a number of heuristics to parse URLs and look for wicked ones. Microsoft offers such a toolbar in Internet Explorer version 7. The idea is that if the user visits a known phishing site, the browser toolbar turns red; if she visits a suspect site, it turns yellow; a normal site leaves it white; while a site with an 'extended validation' certificate — a new, expensive type of certificate that's only sold to websites after their credentials have been checked slightly more diligently than used to be the case — then it will turn green.

The initial offering has already been broken, according to a paper jointly authored by researchers from Stanford and from Microsoft itself [650]. Attackers can present users with a 'picture-in-picture' website which simply displays a picture of a browser with a nice green toolbar in the frame of the normal browser. (No doubt the banks will say 'maximise the browser before entering your password' but this won't work for the reasons discussed above.) The new scheme can also be attacked using similar URLs: for example, `www.bankofthewest.com` can be impersonated as `www.bankofthevvest.com`. Even if the interface problem can be fixed, there are problems with using heuristics to spot dodgy sites. The testing cannot be static; if it were, the phishermen would just tinker with their URLs until they passed the current tests. Thus the toolbar has to call a server at least some of the time, and check in real time whether a URL is good or bad. The privacy aspects bear thinking about, and it's not entirely clear that the competition-policy issues with Passport have been solved either.

2.4.7.8 Two-Factor Authentication

Various firms sell security tokens that produce a one-time password. This can be in response to a challenge sent by the machine to which you want to log on, or more simply a function of time; you can get a keyfob device that displays a new eight-digit password every few seconds. I'll describe the technology in more detail in the next chapter. These devices were invented in the early 1980s and are widely used to log on to corporate systems. They are often referred to as *two-factor authentication*, as the system typically asks for a memorised password as well; thus your logon consists of 'something you have' and also 'something you know'. Password calculators are now used by some exclusive London private banks, such as the Queen's bankers, Coutts, to authenticate their online customers, and we're now seeing them at a handful of big money-centre banks too.

There is some pressure[4] for banks to move to two-factor authentication and issue all their customers with password calculators. But small banks are chronically short of development resources, and even big banks' security staff resist the move on grounds of cost; everyone also knows that the phishermen will simply switch to real-time man-in-the-middle attacks. I'll discuss these in detail in the next chapter, but the basic idea is that the phisherman pretends to the customer to be the bank and pretends to the bank to be the customer at the same time, simply taking over the session once it's established. As of early 2007, only one or two such phishing attacks have been detected, but the attack technology could be upgraded easily enough.

The favoured two-factor technology in Europe is the chip authentication program (CAP) device which I'll also describe in the next chapter. This can be used either to calculate a logon password, or (once man-in-the-middle attacks become widespread) to compute a message authentication code on the actual transaction contents. This means that to pay money to someone you'll probably have to type in their account number and the amount twice — once into the bank's website and once into your CAP calculator. This will clearly be a nuisance: tedious, fiddly and error-prone.

2.4.7.9 Trusted Computing

The 'Trusted Computing' initiative, which has led to TPM security chips in PC motherboards, may make it possible to tie down a transaction to a particular PC motherboard. The TPM chip can support functions equivalent to those of the CAP device. Having hardware bank transaction support integrated into the PC will be less fiddly than retyping data at the CAP as well as the PC; on the other hand, roaming will be a problem, as it is with password manglers or with relying on the browser cache.

Vista was supposed to ship with a mechanism (remote attestation) that would have made it easy for bank software developers to identify customer PCs with high confidence and to stop the bad guys from easily tinkering with the PC software. However, as I'll describe later in the chapter on access control, Microsoft appears to have been unable to make this work yet, so bank programmers will have to roll their own. As Vista has just been released into consumer markets in 2007, it may be 2011 before most customers could have this option available, and it remains to be seen how the banks would cope with Apple or Linux users. It might be fair to say that this technology has not so far lived up to the initial hype.

[4]In the USA, from the Federal Financial Institutions Examination Council — which, as of September 2007, 98% of banks were still resisting [1003].

2.4.7.10 Fortified Password Protocols

In 1992, Steve Bellovin and Michael Merritt looked at the problem of how a guessable password could be used in an authentication protocol between two machines [158]. They came up with a series of protocols for encrypted key exchange, whereby a key exchange is combined with a shared password in such a way that a man-in-the-middle could not guess the password. Various other researchers came up with other protocols to do the same job.

Some people believe that these protocols could make a significant dent in the phishing industry in a few years' time, once the patents run out and the technology gets incorporated as a standard feature into browsers.

2.4.7.11 Two-Channel Authentication

Perhaps the most hopeful technical innovation is two-channel authentication. This involves sending an access code to the user via a separate channel, such as their mobile phone. The Bank of America has recently introduced a version of this called SafePass in which a user who tried to log on is sent a six-digit code to their mobile phone; they use this as an additional password [868]. The problem with this is the same as with the two-factor authentication already tried in Europe: the bad guys will just use a real-time man-in-the-middle attack.

However, two-channel comes into its own when you authenticate transaction data as well. If your customer tries to do a suspicious transaction, you can send him the details and ask for confirmation: 'If you really wanted to send $7500 to Russia, please enter 4716 now in your browser.' Implemented like this, it has the potential to give the level of authentication aimed at by the CAP designers but with a much more usable interface. Banks have held back from using two-channel in this way because of worries that usability problems might drive up their call-centre costs; however the first banks to implement it report that it hasn't pushed up support call volumes, and a number of sites have been implementing it through 2007, with South African banks being in the forefront. We have already seen the first serious fraud — some Johannesburg crooks social-engineered the phone company to send them a new SIM for the phone number of the CFO of Ubuntu, a charity that looks after orphaned and vulnerable children, and emptied its bank account [1017]. The bank and the phone company are arguing about liability, although the phone company says it's fixing its procedures.

Even once the phone-company end of things gets sorted, there are still limits. Two-channel authentication relies for its security on the independence of the channels: although the phishermen may be able to infect both PCs and mobile phones with viruses, so long as both processes are statistically independent,

only a very small number of people will have both platforms compromised at the same time. However, if everyone starts using an iPhone, or doing VoIP telephony over wireless access points, then the assumption of independence breaks down.

Nonetheless, if I were working for a bank and looking for a front-end authentication solution today, two-channel would be the first thing I would look at. I'd be cautious about high-value clients, because of possible attacks on the phone company, but for normal electronic banking it seems to give the most bang for the buck.

2.4.8 The Future of Phishing

It's always dangerous to predict the future, but it's maybe worth the risk of wondering where phishing might go over the next seven years. What might I be writing about in the next edition of this book?

I'd expect to see the phishing trade grow substantially, with attacks on many non-banks. In November 2007, there was a phishing attack on Salesforce.com in which the phisherman got a password from a staff member, following which customers started receiving bogus invoices [614]. If it gets hard to phish the banks, the next obvious step is to phish their suppliers (such as Salesforce). In a world of increasing specialisation and outsourcing, how can you track dependencies and identify vulnerabilities?

Second, research has shown that the bad guys can greatly improve their yields if they match the context of their phish to the targets [658]; so phish will get smarter and harder to tell from real emails, just as spam has. Authority can be impersonated: 80% of West Point cadets bit a phish sent from a bogus colonel, and a phisherman who uses a social network can do almost as well: while emails from a random university address got 16% of students to visit an off-campus website and enter their university password to access it, this shot up to 72% if the email appeared to come from one of the target's friends — with the friendship data collected by spidering open-access social-networking websites [653]. Future phishermen won't ask you for your mother's maiden name: they'll forge emails from your mother.

On the technical side, more man-in-the-middle attacks seem likely, as do more compromises of endpoints such as PCs and mobile phones. If a banking application running on Vista can only do business on the genuine motherboard, then the attacker will look for ways to run his software on that motherboard. If 'trusted computing' features in later releases of Vista can stop malware actually pressing keys and overwriting the screen while a banking application is running, this might bring real benefits (but I'm not holding my breath).

Starting from the top end of the market, I would not be surprised to see exclusive private banks issuing their customers with dedicated payment devices — 'Keep your account $50,000 in credit and get a Free Gold Blackberry!' Such

a device could do wireless payments securely and perhaps even double as a credit card. (I expect it would fail when the marketing department also decided it should handle ordinary email, and the crooks figured out ways of pretexting the rich accountholders into doing things they didn't really mean to.)

At the middle of the market, I'd expect to see phishing become less distinguishable from more conventional confidence tricks. I mentioned earlier that the marketing industry nowadays was largely about getting people to click on links. Now Google has built a twelve-figure business out of this, so if you're a crook, why not just advertise there for victims? It's already started. And indeed, research by Ben Edelman has found that while 2.73% of companies ranked top in a web search were bad, 4.44% of companies who had bought ads from the search engine were bad [416]. (Edelman's conclusion — 'Don't click on ads' — could be bad news in the medium term for the search industry.)

On the regulatory side of things, I expect more attempts to interfere in the identity market, as governments such as America's and Britain's look for ways to issue citizens with identity cards, and as international bodies try to muscle in. The International Telecommunications Union tried this in 2006 [131]; it won't be the last. We will see more pressures to use two-factor authentication, and to use biometrics. Those parts of the security-industrial complex have been well fed since 9/11 and will lobby hard for corporate welfare.

However, I don't believe it will be effective to rely entirely on front-end controls, whether passwords or fancy widgets. Tricksters will still be able to con people (especially the old and the less educated), and systems will continue to get more and more complex, limited only by the security, usability and other failures inflicted by feature interaction. I believe that the back-end controls will be at least as important. The very first home banking system — introduced by the Bank of Scotland in 1984 — allowed payments only to accounts that you had previously 'nominated' in writing. The idea was that you'd write to the bank to nominate your landlord, your gas company, your phone company and so on, and then you could pay your bills by email. You set a monthly limit on how much could be paid to each of them. These early systems suffered almost no fraud; there was no easy way for a bad man to extract cash. But the recipient controls were dismantled during the dotcom boom and then phishing took off.

Some banks are now starting to reintroduce controls — for example, by imposing a delay and requiring extra authentication the first time a customer makes a payment to someone they haven't paid before. Were I designing an online banking system now, I would invest most of the security budget in the back end. The phishermen target banks that are slow at recovering stolen funds [55]. If your asset-recovery team is really on the ball, checks up quickly on attempts to send money to known cash-extraction channels, claws it back vigorously, and is ruthless about using the law against miscreants, then the

phishermen will go after your competitors instead. (I'll discuss what makes controls effective later, in the chapter on Banking and Bookkeeping, especially section 10.3.2.)

2.5 System Issues

Although the fastest-growing public concern surrounding passwords is phishing, and the biggest research topic is psychology, there are a number of other circumstances in which attackers try to steal or guess passwords, or compromise systems in other ways. There are also technical issues to do with password entry and storage that I'll also cover briefly here for the sake of completeness.

I already noted that the biggest system issue was whether it is possible to restrict the number of password guesses. Security engineers sometimes refer to password systems as 'online' if guessing is limited (as with ATM PINs) and 'offline' if it is not (this originally meant systems where a user could fetch the password file and take it away to try to guess the passwords of other users, including more privileged users). The terms are no longer really accurate. Some offline systems restrict password guesses, such as the smartcards used in more and more countries for ATMs and retail transactions; these check the PIN in the smartcard chip and rely on its tamper-resistance to limit guessing. Many online systems cannot restrict guesses; for example, if you log on using Kerberos, an opponent who taps the line can observe your key encrypted with your password flowing from the server to your client, and then data encrypted with that key flowing on the line; so she can take her time to try out all possible passwords.

Password guessability is not the only system-level design question, though; there are others (and they interact). In this section I'll describe a number of issues concerning threat models and technical protection, which you might care to consider next time you design a password system.

Just as we can only talk about the soundness of a security protocol in the context of a specific threat model, so we can only judge whether a given password scheme is sound by considering the type of attacks we are trying to defend against. Broadly speaking, these are:

Targeted attack on one account: an intruder tries to guess a particular user's password. He might try to guess the PIN for Bill Gates's bank account, or a rival's logon password at the office, in order to do mischief directly. When this involves sending emails, it is known as *spear phishing*.

Attempt to penetrate any account on a system: the intruder tries to get a logon as any user of the system. This is the classic case of the phisherman trying to get a password for any user of a target bank's online service.

Attempt to penetrate any account on any system: the intruder merely wants an account at any system in a given domain but doesn't care which one. Examples are bad guys trying to guess passwords on an online service so they can send spam from the compromised account, or use its web space to host a phishing site for a few hours. The modus operandi is often to try one or two common passwords (such as 'password1') on large numbers of randomly-selected accounts. Other possible attackers might be teens looking for somewhere to hide pornography, or a private eye tasked to get access to a company's intranet who is looking for a beachhead in the form of a logon to some random machine in their domain.

Service denial attack: the attacker may wish to prevent the legitimate user from using the system. This might be targeted on a particular account or system-wide.

This taxonomy helps us ask relevant questions when evaluating a password system.

2.5.1 Can You Deny Service?

Banks often have a rule that a terminal and user account are frozen after three bad password attempts; after that, an administrator has to reactivate them. This could be rather dangerous in a military system, as an enemy who got access to the network could use a flood of false logon attempts to mount a service denial attack; if he had a list of all the user names on a machine he might well take it out of service completely. Many commercial websites nowadays don't limit guessing because of the possibility of such an attack.

When deciding whether this might be a problem, you have to consider not just the case in which someone attacks one of your customers, but also the case in which someone attacks your whole system. Can a flood of false logon attempts bring down your service? Could it be used to blackmail you? Or can you turn off account blocking quickly in the event that such an attack materialises? And if you do turn it off, what sort of attacks might follow?

2.5.2 Protecting Oneself or Others?

Next, to what extent does the system need to protect users from each other? In some systems — such as mobile phone systems and cash machine systems — no-one should be able to use the service at someone else's expense. It is assumed that the attackers are already legitimate users of the system. So systems are (or at least should be) carefully designed so that knowledge of one user's password will not allow another identifiable user's account to be compromised.

Where a user who chooses a password that is easy to guess harms only himself, a wide variation in password strength can more easily be tolerated. (Bear in mind that the passwords people choose are very often easy for their spouses or partners to guess [245]: so some thought needs to be given to issues such as what happens when a cheated partner seeks vengeance.)

But many systems do not provide strong separation between users. Operating systems such as Unix and Windows may have been designed to protect one user against accidental interference by another, but they are not hardened to protect against capable malicious actions by other users. They have many well-publicized vulnerabilities, with more being published constantly on the web. A competent opponent who can get a single account on a shared computer system that is not professionally managed can usually become the system administrator fairly quickly, and from there he can do whatever he likes.

2.5.3 Attacks on Password Entry

Password entry is often poorly protected.

2.5.3.1 Interface Design

Sometimes the problem is thoughtless interface design. Some common makes of cash machine had a vertical keyboard at head height, making it simple for a pickpocket to watch a customer enter her PIN before lifting her purse from her shopping bag. The keyboards were at a reasonable height for the men who designed them, but women — and men in many countries — are a few inches shorter and were highly exposed. One of these machines 'protected client privacy' by forcing the customer to gaze at the screen through a narrow slot. Your balance was private, but your PIN was not! Many pay-telephones have a similar problem, and *shoulder surfing* of calling card details (as it's known in the industry) has been endemic at some locations such as major US train stations and airports.

I usually cover my dialling hand with my body or my other hand when entering a card number or PIN in a public place — but you shouldn't design systems on the assumption that all your customers will do this. Many people are uncomfortable shielding a PIN from others as it's a visible signal of distrust; the discomfort can be particularly acute if someone's in a supermarket queue and a friend is standing nearby. In the UK, for example, the banks say that 20% of users never shield their PIN when entering it, as if to blame any customer whose PIN is compromised by an overhead CCTV camera [84]; yet in court cases where I've acted as an expert witness, only a few percent of customers shield their PIN well enough to protect it from an overhead camera. (And just wait till the bad guys start using infrared imaging.)

2.5.3.2 Eavesdropping

Taking care with password entry may stop the bad guys looking over your shoulder as you use your calling card at an airport telephone. But it won't stop other eavesdropping attacks. The latest modus operandi is for bad people to offer free WiFi access in public places, and harvest the passwords that users enter into websites. It is trivial to grab passwords entered into the many websites that don't use encryption, and with a bit more work you can get passwords entered into most of them that do, by using a middleperson attack.

Such attacks have been around for ages. In the old days, a hotel manager might abuse his switchboard facilities to log the keystrokes you enter at the phone in your room. That way, he might get a credit card number you used — and if this wasn't the card number you used to pay your hotel bill, he could plunder your account with much less risk. And in the corporate world, many networked computer systems still send passwords in clear over local area networks; anyone who can program a machine on the network, or attach his own sniffer equipment, can harvest them. (I'll describe in the next chapter how Windows uses the Kerberos authentication protocol to stop this, and `ssh` is also widely used — but there are still many unprotected systems.)

2.5.3.3 Technical Defeats of Password Retry Counters

Many kids find out that a bicycle combination lock can usually be broken in a few minutes by solving each ring in order of looseness. The same idea worked against a number of computer systems. The PDP-10 TENEX operating system checked passwords one character at a time, and stopped as soon as one of them was wrong. This opened up a *timing attack*: the attacker would repeatedly place a guessed password in memory at a suitable location, have it verified as part of a file access request, and wait to see how long it took to be rejected [774]. An error in the first character would be reported almost at once, an error in the second character would take a little longer to report, and in the third character a little longer still, and so on. So you could guess the characters once after another, and instead of a password of N characters drawn from an alphabet of A characters taking $A^N/2$ guesses on average, it took $AN/2$. (Bear in mind that in thirty years' time, all that might remain of the system you're building today is the memory of its more newsworthy security failures.)

These same mistakes are being made all over again in the world of embedded systems. With one remote car locking device: as soon as a wrong byte was transmitted from the key fob, the red telltale light on the receiver came on. With some smartcards, it has been possible to determine the customer PIN by trying each possible input value and looking at the card's power consumption, then issuing a reset if the input was wrong. The reason was that a wrong PIN caused a PIN retry counter to be decremented, and writing to the EEPROM memory

which held this counter caused a current surge of several milliamps — which could be detected in time to reset the card before the write was complete [753]. These implementation details matter.

2.5.4 Attacks on Password Storage

Passwords have often been vulnerable where they are stored. There was a horrendous bug in one operating system update in the 1980s: a user who entered a wrong password, and was told ''sorry, wrong password'' merely had to hit carriage return to get into the system anyway. This was spotted quickly, and a patch was shipped, but almost a hundred U.S. government systems in Germany were using unlicensed copies of the software and didn't get the patch, with the result that hackers were able to get in and steal information, which they are rumored to have sold to the KGB.

Another horrible programming error struck a U.K. bank, which issued all its customers the same PIN by mistake. It happened because the standard equipment in use at the time for PIN generation required the bank programmer to first create and store an encrypted PIN, and then use another command to print out a clear version on a PIN mailer. A bug meant that all customers got the same encrypted PIN. As the procedures for handling PINs were carefully controlled, no one in the bank got access to anyone's PIN other than his or her own, so the mistake wasn't spotted until after thousands of customer cards had been shipped.

Auditing provides another hazard. In systems that log failed password attempts, the log usually contains a large number of passwords, as users get the 'username, password' sequence out of phase. If the logs are not well protected then attacks become easy. Someone who sees an audit record of a failed login with a non-existent user name of `e5gv,8yp` can be fairly sure that this string is a password for one of the valid user names on the system.

2.5.4.1 One-Way Encryption

Password storage has also been a problem for some systems. Keeping a plaintext file of passwords can be dangerous. In MIT's 'Compatible Time Sharing System', `ctss` (a predecessor of Multics), it once happened that one person was editing the message of the day, while another was editing the password file. Because of a software bug, the two editor temporary files got swapped, with the result that everyone who logged on was greeted with a copy of the password file!

As a result of such incidents, passwords are often protected by encrypting them using a one-way algorithm, an innovation due to Roger Needham and Mike Guy. The password, when entered, is passed through a one-way function and the user is logged on only if it matches a previously stored value. However,

it's often implemented wrong. The right way to do it is to generate a random salt, hash the password with the salt, and store both the salt and the hash in the file. The popular blog software Wordpress, as of October 2007, simply stores a hash of the password — so if the attacker can download the password file for a Wordpress blog, he can look for weak passwords by comparing the file against a precomputed file of hashes of words in the dictionary. What's even worse is that Wordpress then uses a hash of this hash as the cookie that it sets on your browser once you've logged on. As a result, someone who can look at the password file can also get in by computing cookies from password hashes, so he can attack even an adminstrator account with a strong password. In this case, the one-way algorithm went the wrong way. They should have chosen a random cookie, and stored a hash of that too.

2.5.4.2 Password Cracking

However, some systems that do use an encrypted password file make it widely readable (Unix used to be the prime example — the password file was by default readable by all users). So a user who can fetch this file can then try to break passwords offline using a dictionary; he encrypts the values in his dictionary and compares them with those in the file (an activity called a *dictionary attack*, or more colloquially, *password cracking*). The upshot was that he could impersonate other users, perhaps including a privileged user. Windows NT was slightly better, but the password file could still be accessed by users who knew what they were doing.

Most modern operating systems have fixed this problem, but the attack is still implemented in commercially available password recovery tools. If you've encrypted an Office document with a password you've forgotten, there are programs that will try 350,000 passwords a second [1132]. Such tools can just as easily be used by a bad man who has got a copy of your data, and in older systems of your password file. So password cracking is still worth some attention. Well-designed password protection routines slow down the guessing by using a complicated function to derive the crypto key from the password and from a locally-stored salt that changes with each file; the latest WinZip, for example, allows less than 1000 guesses a second. You can also complicate a guessing attack by using an odd form of password; most password guessers try common words first, then passwords consisting of a root followed by an appendage, such as 'Kevin06'. Users who avoid such patterns can slow down the attacker.

2.5.5 Absolute Limits

Regardless of how well passwords are managed, there can be absolute limits imposed by the design of the platform. For example, Unix systems used to

limit the length of the password to eight characters (you could often enter more than this, but the ninth and subsequent characters were ignored). The effort required to try all possible passwords — the *total exhaust time*, in cryptanalytic jargon — is 96^8 or about 2^{52}, and the average effort for a search is half of this. A well-financed government agency (or a well-organised hacker group) can now break any encrypted password in a standard Unix password file.

This motivates more technical defenses against password cracking, including 'shadow passwords', that is, encrypted passwords hidden in a private file (most modern Unices), using an obscure mechanism to do the encryption (Novell), or using a secret key with the encryption (MVS). The strength of these mechanisms may vary.

For the above reasons, military system administrators often prefer to issue random passwords. This also lets the probability of password guessing attacks be estimated and managed. For example, if L is the maximum password lifetime, R is login attempt rate, S is the size of the password space, then the probability that a password can be guessed in its lifetime is $P = LR/S$, according to the US Department of Defense password management guideline [377].

There are various problems with this doctrine, of which the worst may be that the attacker's goal is often not to guess some particular user's password but to get access to any account. If a large defense network has a million possible passwords and a million users, and the alarm goes off after three bad password attempts on any account, then the attack is to try one password for every single account. Thus the quantity of real interest is the probability that the password space can be exhausted in the lifetime of the system at the maximum feasible password guess rate.

To take a concrete example, UK government systems tend to issue passwords randomly selected with a fixed template of consonants, vowels and numbers designed to make them easier to remember, such as CVCNCVCN (eg `fuR5xEb8`). If passwords are not case sensitive, the guess probability is only $21^4.5^2.10^2$, or about 2^{29}. So if an attacker could guess 100 passwords a second — perhaps distributed across 10,000 accounts on hundreds of machines on a network, so as not to raise the alarm — then he'd need about 5 million seconds, or two months, to get in. With a million-machine botnet, he could obviously try to speed this up. So if you're responsible for such a system, you might find it prudent to do rate control: prevent more than one password guess every few seconds per user account, or (if you can) by source IP address. You might also keep a count of all the failed logon attempts and analyse them: is there a constant series of guesses that could indicate an attempted intrusion? (And what would you do if you noticed one?) With a commercial website, 100 passwords per second may translate to one compromised user account per second. That may not be a big deal for a web service with 100 million accounts — but it may still be worth trying to identify the source of any industrial-scale password-guessing attacks. If they're from a

small number of IP addresses, you can block them, but this won't work so well if the attacker has a botnet. But if an automated-guessing attack does emerge, then another way of dealing with it is the CAPTCHA, which I'll describe next.

2.6 CAPTCHAs

Recently people have tried to design protection mechanisms that use the brain's strengths rather than its weaknesses. One early attempt was *Passfaces*: this is an authentication system that presents users with nine faces, only one of which is of a person they know; they have to pick the right face several times in a row to log on [356]. The rationale is that people are very good at recognising other people's faces, but very bad at describing them: so you could build a system where it was all but impossible for people to give away their passwords, whether by accident or on purpose. Other proposals of this general type have people selecting a series of points on an image — again, easy to remember but hard to disclose. Both types of system make shoulder surfing harder, as well as deliberate disclosure offline.

The most successful innovation in this field, however, is the CAPTCHA — which stands for 'Completely Automated Public Turing Test to Tell Computers and Humans Apart.' You will probably have seen these: they are the little visual puzzles that you often have to solve to post to a blog, or register for a free email account. The idea is that a program generates some random text, and produces a distorted version of it that the user must decipher. Humans are good at reading distorted text, while programs are less good. CAPTCHAs first came into use in a big way in 2003 to stop spammers using scripts to open thousands of accounts on free email services, and their judicious use can make it a lot harder for attackers to try a few simple passwords with each of a large number of existing accounts.

The CAPTCHA was devised by Luis von Ahn and colleagues [1304]. It is inspired by the test famously posed by Alan Turing as to whether a computer was intelligent, where you put a computer in one room and a human in another, and invite a human to try to tell them apart. The innovation is that the test is designed so that a computer can tell the difference between human and machine, using a known 'hard problem' in AI such as the recognition of distorted text against a noisy background. The idea is that breaking the CAPTCHA is equivalent to solving the AI problem.

As with all new security technologies, the CAPTCHA is undergoing a period of rapid coevolution of attack and defence. Many of the image recognition problems posed by early systems turned out not to be too hard at all. There are also possible protocol-level attacks; von Ahn mentioned in 2001 that in theory a

spammer could use a porn site to solve them, by getting people to solve them as the price of access to free porn [1303]. This has since become a folk legend, and finally, in October 2007, it actually started to happen: spammers created a game in which you undress a woman by solving one CAPTCHA after another [134]. Also in that month, we saw the first commercial CAPTCHA-breaking tools arrive on the market [571].

Finally, the technology can be integrated with authentication and authorisation controls in potentially useful new ways. An interesting example comes from the banks in Germany, who are introducing an anti-phishing measure whereby if you authorise a payment online the bank sends you the payee, the amount and your date of birth, integrated into a CAPTCHA that also contains a challenge, such as 'if you want to authorize this payment please enter the thirteenth password from your list'. This lets them use a static list of one-time passwords to authenticate actual amounts and beneficiaries, by ensuring that a real-time man-in-the-middle attack would require a human in the loop. It may be a better technology than the CAP calculator; it will certainly be less fiddly than entering transaction details twice. Time will tell if it works.

2.7 Summary

Usability is one of the most important and yet hardest design problems in many secure systems. It was long neglected as having less techie glamour then operating systems or cryptographic algorithms; yet most real attacks nowadays target the user. Phishing is the most rapidly growing threat to online banking systems, and is starting to be a problem for other sites too. Other forms of deception are also likely to increase; as technical protection improves, the bad guys will target the users.

Much of the early work on security usability focused on passwords. Critical questions to ask when designing a password system include not just whether people might re-use passwords, but also whether they need to be protected from each other, whether they can be trained and disciplined, and whether accounts can be frozen after a fixed number of bad guesses. You also have to consider whether attackers will target a particular account, or be happy with breaking any account on a machine or a network; and technical protection issues such as whether passwords can be snooped by malicious software, false terminals or network eavesdropping.

However, there is no 'magic bullet' in sight. As minor improvements in protection are devised and fielded, so the phishermen adapt their tactics. At present, the practical advice is that you should not be a soft touch — harden your system enough for the phishermen to hit your competitors instead. This involves not just security usability issues but also your internal controls, which

we will discuss in later chapters. You should assume that some user accounts will be compromised, and work out how to spot this and limit the damage when it does happen.

Research Problems

There is a lot of work being done on phishing, but (as we discussed here) none of it is no far a really convincing solution to the problem. We could do with some fresh thinking. Are there any neat ways to combine things like passwords, CAPTCHAs, images and games so as to provide sufficiently dependable two-way authentication between humans and computers? In general, are there any ways of making middleperson attacks sufficiently harder that it doesn't matter if the Mafia owns your ISP?

We also need more fundamental thinking about the relationship between psychology and security. Between the first edition of this book in 2001 and the second in 2007, the whole field of security economics sprang into life; now there are two regular conferences and numerous other relevant events. So far, security usability is in a fairly embryonic state. Will it also grow big and prosperous? If so, which parts of existing psychology research will be the interesting areas to mine?

Further Reading

When I wrote the first edition of this book, there was only a small handful of notable papers on passwords, including classic papers by Morris and Thompson [906], Grampp and Morris [550], and Klein [720], and some DoD guidelines [377]. Since then there has arisen a large research literature on phishing, with a compendium of papers published as [659]. Perhaps the greatest gains will come when security engineers start paying attention to standard HCI texts such as [1039], and researchers start reading widely in the psychology literature.

A text I've found helpful is James Reason's *'Human Error'*, which essentially tells us what the safety-critical systems community has learned from many years studying the cognate problems in their field [1060]. Recently, we've seen the first book on security usability — a collection of the early research papers [333]. There is also an annual workshop, the Symposium On Usable Privacy and Security (SOUPS) [1240].

I'm loth to provide much of a guide to the psychology literature, as I don't know it as well as I ought to, and we've only just started on the project of building 'security psychology' as a discipline. It will take some years for us to find which psychological theories and experimental results provide us with

useful insights. But here are some pointers. Tom Gilovich, Dale Griffin and Danny Kahneman put together a volume of papers summarising the state of play in the heuristics and biases tradition in 2002 [529]; while a more gentle introduction might be a book chapter by Richard Samuels, Steven Stich and Luc Faucher discussing the tensions between that tradition and the evolutionary psychologists [1106]. It may also be of interest that a number of psychologists and primatologists (such as Nicholas Humphrey, Richard Byrne and Andy Whiten) have argued that we evolved intelligence because people who were better at deception, or at detecting deception in others, had more surviving offspring — the so-called 'Machiavellian Brain' hypothesis [250]. This might lead us to wonder whether security engineering is the culmination of millions of years of evolution! (Other psychologists, such as Simon Baron-Cohen, would deflate any such hubris by arguing that nurturing the young was at least as important.) Further fascinating analogies with evolutionary biology have been collected by Raphael Sagarin and Terence Taylor in their book *'Natural Security'*.

Finally, if you're interested in the dark side, *'The Manipulation of Human Behavior'* by Albert Biderman and Herb Zimmer reports experiments on interrogation carried out after the Korean War with US Government funding [162]. It's also known as the Torturer's Bible, and describes the relative effectiveness of sensory deprivation, drugs, hypnosis, social pressure and so on when interrogating and brainwashing prisoners.