CHAPTER

19

Protecting E-Commerce Systems

If you try to buck the markets, then the markets will buck you.

—MARGARET THATCHER

19.1 Introduction

The protection of electronic commerce systems pulls together a lot of the topics discussed in previous chapters. Failures come from misconfigured access control, implementation blunders, theft of network services, inappropriate use of crypotology—you name it. In this chapter, I'll cover some protection issues specific to e-commerce, such as how online credit card payments are handled, and what goes wrong.

If you are a programmer building e-commerce systems for a dot-com startup, much of the material in this chapter should be fairly familiar to you. You are much more likely to get value from the chapters on access control, network security, and (particularly) on banking. The most likely attacks on your business don't involve the vulnerabilities in the Internet protocol suite or the payment infrastructure—and you can't do anything about those anyway.

The typical e-business startup appears to be most at risk from internal fraud. This is where most frauds in normal businesses come from, despite things like double-entry bookkeeping, which have evolved over centuries to control them. Many startups have none of these internal controls. They begin as a few people who all know each other and, if successful at raising capital, rapidly hire a lot of new staff who're focused on money and not very carefully screened. A survey found in October 2000, for example, that 37% of dot-com executives have shady pasts, compared with 10% found in due diligence checks on normal companies [257].

19.2 A Telegraphic History of E-Commerce

Many of the problems afflicting e-businesses stem from the popular notion that e-commerce is something completely new, invented in the mid-1990s. This is simply untrue.

Various kinds of visual signalling were deployed from classical times. Systems included heliographs (which used mirrors to flash sunlight at the receiver), semaphones (which used the positions of moving arms to signal letters and numbers), and flags. Land-based systems sent messages along chains of beacon towers, and naval systems between ships. To begin with, their use was military, but after the Napoleonic War, the French government opened its heliograph network to commercial use. Very soon, the first frauds were carried out. For two years, until they were discovered in 1836, two bankers bribed an operator to signal the movements of the stock market to them covertly by making errors in transmissions, which they could observe from a safe distance. Other techniques were devised to signal the results of horseraces. Various laws were passed to criminalize this kind of activity, but they were ineffective. The only solution for the bookies was to "call time" by a clock, rather than waiting for the result and hoping that they were the first to hear it.

From the 1760s to the 1840s, the electric telegraph was developed by a number of pioneers, of whom the most influential was Samuel Morse. He persuaded Congress in 1842 to fund an experimental line from Washington to Baltimore; this so impressed people that serious commercial investment started, and by the end of that decade, there were 12,000 miles of line being operated by 20 companies. This was remarkably like the Internet boom of the late 1990s [729].

Banks were the first big users of the telegraph, and they decided that they needed technical protection mechanisms to prevent transactions being altered by crooked operators en route. (I discussed the *test key* systems they developed for the purpose in the chapter on banking systems.) Telegrams were also used to create national markets. For the first time, commodity traders in New York could find out within minutes the prices that had been set in auctions in Chicago; likewise, fishing skippers arriving in Boston could find out the price of cod in Gloucester. A recent history of the period shows that most of the concepts and problems of electronic commerce were familiar to the Victorians [729]. How do you know who you're speaking to? How do you know if they're trustworthy? How do you know whether the goods will be delivered, and whether payments will arrive? The answers found in the nineteenth century involved intermediaries—principally banks that helped businesses manage risk using instruments such as references, guarantees, and letters of credit.

In the 1960s, banks in many countries computerized their bookkeeping, and introduced national interbank systems for handling direct payments to customer accounts, enabling banks to offer services such as payroll to corporate customers. In the early 1970s, this was extended to international payments, as described in the banking systems chapter. The next large expansion of electronic commerce took place in the late 1970s to mid-1980s with the spread of *electronic data interchange* (EDI). Companies ranging from General Motors to Marks and Spencer built systems that enabled them to link up their computers to their suppliers', so that goods could be ordered automatically. Travel agents built similar systems to order tickets in real time from airlines.

In 1985, the first retail electronic banking system was offered by the Bank of Scotland, whose customers could use Prestel, a proprietary email system operated by British Telecom, to make payments. When Steve Gold and Robert Schifreen hacked Prestel—as described in the chapter on passwords—it initially terrified the press and the bankers. They realized that the hackers could easily have captured and altered transactions. But once the dust settled and people thought through the detail, it became clear there was little real risk. The system allowed only payments between your own accounts and to accounts which you'd previously notified to the bank, such as your gas and electricity suppliers.

This pattern, of high-profile hacks—which caused great consternation but which, on sober reflection, turned out to be not really a big deal—has continued ever since.

To resume this brief history, the late 1980s and early 1990s saw the rapid growth of call centers, which—despite all the hoopla about the Web—remain in 2000 by far the largest delivery channel for business-to-consumer electronic commerce. As for the Internet, it was not something that suddenly sprung into existence in 1995, as a Martian monitoring our TV channels might believe. The first time I used an online service to sell software I'd written was in 1984 or 1985; and I first helped the police investigate an online credit card fraud in 1987. In the latter case, the bad guy got a list of hot credit card numbers from his girlfriend, who worked in a supermarket; he used them to buy software from companies in California, which he downloaded to order for his customers. This worked because hot card lists at the time carried only those cards that were being used fraudulently in that country; it also guaranteed that the bank would not be able to debit an innocent customer. As it happens, the criminal quit before there was enough evidence to nail him. A rainstorm washed away the riverbank opposite his house and exposed a hide which the police had built to stake him out.

The use of credit cards to buy stuff electronically "suddenly" became mainstream in about 1994 or 1995, when the public started to go online in large numbers. Suddenly there was a clamor that the Internet was insecure, that credit card numbers could be harvested on a huge scale, and that encryption would be needed.

19.3 An Introduction to Credit Cards

For many years after their invention in the 1950s, credit cards were treated by most banks as a loss leader used to attract high-value customers. Eventually, in most countries, the number of merchants and cardholders reached critical mass, and the transaction volume took off. In Britain, it took almost 20 years before most banks found the business profitable; then all of a sudden it became extremely profitable. The credit card system is now extremely well entrenched as the payment mechanism used on the Net.

Because of the huge investment involved in rolling out a competitor to tens of thousands of banks, millions of merchants, and billions of customers worldwide, any new payment mechanism is likely to take some time to get established—with a possible exception, which I'll discuss shortly. When you use a credit card to pay for a purchase

in a store, the transaction flows from the merchant to her bank (the acquiring bank), which pays her after deducting a *merchant discount* of, typically, 4–5%. If the card was issued by a different bank, the transaction next flows to a switching center run by the brand (such as VISA), which takes a commission and passes it to the issuing bank for payment. Daily payments between the banks and the brands settle the net cash flows. The issuer also gets a slice of the merchant discount, but makes most of its money from extending credit to cardholders at rates usually much higher than the interbank rate.

19.3.1 Fraud

The risk of fraud from stolen cards was traditionally managed by a system of *hot card lists* and *merchant floor limits*. Each merchant gets a local hot card list—formerly on paper, now stored in her terminal—plus a limit set by their acquiring bank, above which they have to call for authorization. The call center, or online service, which she uses for this has access to a national hot card list; above a higher limit, they will contact the brand which has a complete list of all hot cards being used internationally; above a still higher limit, the transaction will be checked all the way back to the card issuer.

The introduction of *mail order and telephone order* (MOTO) transactions in the 1970s meant that the merchant did not have the customer present, and was not able to inspect the card. What was to stop a crook ordering goods using a credit card numbers he'd picked up from a discarded receipt?

Banks managed the risk by using the expiry date as a password, lowering the floor limits, increasing the merchant discount, and insisting on delivery to a cardholder address, which is supposed to be checked during authorization. But the main change was to shift liability so that the merchant bore the full risk of disputes. If you challenge an online credit card transaction (or in fact any transaction made under MOTO rules), the full amount is immediately debited back to the merchant, together with a significant handling fee. The same procedure applies whether the debit is a fraud, a dispute, or a return.

Of course, having the cardholder present doesn't guarantee that fraud will be rare. For many years, most fraud was done in person with stolen cards, and the stores that got badly hit tended to be those selling goods that can be easily fenced, such as jewelry and consumer electronics. Banks responded by lowering their floor limits. More recently, as technical protection mechanisms have improved, there has been an increase in scams involving cards that were never received by genuine customers. This *pre-issue fraud* can involve thefts from the mail of the many "pre-approved" cards that arrive in junk mail, or even applications made in the names of people who exist and are creditworthy, but are not aware of the application (*identity theft*). These attacks on the system are intrinsically hard to tackle using purely technical means.

19.3.2 Forgery

In the early 1980s, electronic terminals were introduced through which a sales clerk could swipe a card and get an authorization automatically. But the sales draft was still captured from the embossing, so crooks figured out how to re-encode the magnetic strip of a stolen card with the account number and expiry date of a valid card, which they often got by fishing out discarded receipts from the trash cans of expensive restaurants. A re-encoded card would authorize perfectly, but when the merchant submitted

the draft for payment, the account number didn't match the authorization code (a six-digit number typically generated by encrypting the account number, date, and amount). The merchants didn't get paid, and raised hell.

Banks responded in the mid-1980s by introducing *terminal draft capture*, where a sales draft is printed automatically using the data on the card strip. The crooks' response was a flood of forged cards, many produced by Triad gangs: between 1989 and 1992, magnetic strip counterfeiting grew from an occasional nuisance into half the total fraud losses [6]. VISA's response was *card verification values* (CVVs), three-digit MACs computed on the card strip contents (account number, version number, expiry date) and written at the end of the strip. They worked well at first; in the first quarter of 1994, VISA International's fraud losses dropped by 15.5%, while Mastercard's rose 67% [165]. Subsequently, Mastercard adopted similar checksums, too.

The crooks' response was *skimming*—operating businesses where genuine customer cards were swiped through an extra, unauthorized, terminal to grab a copy of the magnetic strip, which would then be re-encoded on a genuine card. The banks' response was intrusion detection systems, which in the first instance tried to identify criminal businesses by correlating the previous purchase histories of customers who complained.

In the late 1990s, credit card fraud rose sharply due to another simple innovation in criminal technology: the crooked businesses that skim card data absorb the cost of the customer's transaction rather than billing it. You have a meal at a Mafia-owned restaurant, offer a card, sign the voucher, and fail to notice when the charge doesn't appear on your bill. Perhaps a year later, there is suddenly a huge bill for jewelry, electronic goods, or even casino chips. By then you've completely forgotten about the meal, and the bank never had a record of it [318].

19.3.3 Automatic Fraud Detection

Consequently, a lot of work was done in the 1990s on beefing up intrusion detection. There are a number of generic systems that do anomaly detection, using techniques such as neural networks, but it's unclear how effective they are. When fraud is down one year, it's hailed as a success for the latest fraud-spotting system [61]; when the figures go up a few years later, the vendors let the matter pass quietly [714].

More convincing are projects undertaken by specific store chains that look for known patterns of misuse. For example, an electrical goods chain in the New York area observed that offender profiling (by age, sex, race, and so on) was ineffective, and used purchase profiling instead to cut fraud by 82% in a year. Its technique involved not just being suspicious of high-value purchases, but training staff to be careful when customers were careless about purchases and spent less than the usual amount of time discussing options and features. These factors can be monitored online, too, but one important aspect of the New York success is harder for a Web site: employee rewarding. Banks give a $50 reward per bad card captured, which many stores just keep, so their employees don't make an effort to spot cards or risk embarrassment by confronting a customer. In New York, some store staff were regularly earning a weekly bonus of $150 or more [525].

With the human out of the loop at the sales end, the only psychology from which a site designer can leverage is that of the villain. It has been suggested that an e-commerce site should have an unreasonably expensive "platinum" option, which few genuine customers will want to buy [721]. This performs two functions. First, it allows

you to do rudimentary purchase profiling. Second, it fits with the model of *Goldilocks pricing*, developed by online economists Shapiro and Varian, who point out that the real effect of airlines offering first-class fares is to boost sales of business class seats to travelers who can now convince their bosses (or themselves) that they are being "economical" [696]. Another idea is to have a carefully engineered response to suspect transactions: if you just say "bad card, try another one," then the fraudster probably will. You may even end up being "used" by the crooks as an online service that tells them which of their stolen cards are on the hot list, and this can upset your bank (even though the banks are to blame for the system design). A better approach is claim that you're out of stock, so the bad guy will go elsewhere [721].

19.3.4 Economics

There's a lot of misinformation about credit card fraud, with statistics quoted selectively to make points. In one beautiful example, VISA was reported to have claimed that card fraud was up *and* that card fraud was down, on the same day [380]

However, a consistent pattern of figures can be dug out of the trade publications. The actual cost of credit card fraud, before the recent rise, was about 0.15% of all international transactions processed by VISA and MasterCard [652], while national rates varied from 1% in America to 0.2% in the U.K. to under 0.1% in France and Spain. The prevailing business culture has a large effect on the rate. U.S. banks, for example, are much more willing to send out huge junk mailings of pre-approved cards to increase their customer base, and write off the inevitable pre-issue fraud as a cost of doing business. In other countries, banks are more risk-averse.

France is interesting, as it seems, at first sight, to be an exceptional case, in which a particular technology has brought real benefits. French banks introduced chip cards for all domestic transactions in the late 1980s, and this reduced losses from 0.269% of turnover in 1987 to 0.04% in 1993 and 0.028% in 1995. However, there is now an increasing amount of cross-border fraud. French villains use foreign magnetic stripe cards— particularly from Britain [315, 652]—while French chip cards are used at merchants in non-chip countries [166]. But the biggest reduction in Europe was not in France but in Spain, where the policy was to reduce all merchant floor limits to zero and make all transactions online. This cut their losses from 0.21% of turnover in 1988 to 0.008% in 1991 [73].

The lessons appear to be that, first, card fraud is cyclical, as new defenses are introduced and the villains learn to defeat them; and second, that the most complicated and expensive technological solution doesn't necessarily work best in the field.

19.4 Online Credit Card Fraud: The Hype and the Reality

We turn now from traditional credit card fraud to the online variety. There was great anxiety in the mid-1990s that the use of credit cards on the Internet would lead to an avalanche of fraud, as "Evil Hackers" intercepted emails and Web forms, and har-

vested credit card numbers by the million. These fears drove banks and software vendors to devise two protocols to protect Web-based credit card transactions: SSL and SET, which I'll explain in the next section.

The hype surrounding this type of fraud has been grossly overdone. Intercepting email is indeed possible, but it's surprisingly difficult in practice—so much so that governments are bullying ISPs to install snooping devices on their networks to make court-authorized wiretaps easier [114]. But the cost of such devices is so high that the ISPs are resisting this pressure as forcefully as they can. I'll go into this further in Chapter 21. And although it is possible to redirect a popular Web page to your own site using tricks such as DNS cache poisoning, it's a lot simpler to tap the plain old telephone system—and no one worries much about the few credit card numbers that get harvested from hotel guests this way.

Credit card numbers are indeed available on the Net, but usually because someone hacked the computer of a merchant who disobeyed the standard bank prohibition against retaining customer credit card numbers after being paid. (As this book was going to press, VISA announced that, starting in 2001, all its merchants will have to obey 10 new security rules; for example, they must install a firewall, keep security patches up to date, encrypt stored and transmitted data, and regularly update antivirus software [752].) Likewise, fraudulent Web-based transactions do occur, but mainly because of poor implementation of the system whereby cardholder addresses are checked during authorization. The real problem facing dot-coms is disputes.

It is easy to repudiate a transaction. Basically, all the customer has to do is call the credit card company and say, "I didn't authorize that," and the merchant will be saddled with the bill. This was workable in the days when almost all credit card transactions took place locally, and most were for significant amounts. If a customer fraudulently repudiated a transaction, the merchant would pursue them through the courts and harrass them using local credit reference agencies. In addition, the banks' systems are often quite capable of verifying local cardholder addresses.

But the Internet differs from the old mail order/telephone order regime, in that many transactions are international, amounts often are small, and verifying overseas addresses via the credit card system is problematic. Often, all the call center operator can do is check that the merchant seems confident when reading an address in the right country. Thus, the opportunity for repudiating transactions—and getting away with it—is hugely increased. There are particularly high rates of repudiation of payment to porn sites. No doubt some of these disputes happen when a transaction made under the influence of a flush of hormones turns up on the family credit card bill, and the cardholder has to repudiate it to save his marriage; but many are the result of blatant fraud by operators.

At the time of writing, the press was reporting that the Federal Trade Commission was prosecuting the operators of scores of adult Web sites, including `playboy. com`, for billing thousands of users for supposedly free services. The scam was to offer a "free tour" of the site, demand a credit card number, supposedly to verify that the user was over 18, and then bill him anyway. Some sites billed other consumers who have never visited them at all [389]. (Of course, none of this should have surprised the student of more traditional telecomms fraud, as it's just cramming in a new disguise.) If even apparently large and "respectable" Web sites such as `playboy. com` indulge in such practices, it's much easier for consumers to get away with fraudulently repudiating transactions.

The critical importance of this for online businesses is that, if more than a small percentage of your transactions are challenged by customers, your margins will be eroded; and in extreme cases your bank may withdraw your card acquisition service. It has been reported that the collapse of sportswear merchant `boo.com` was because it had too many returns: its business model assumed a no-quibble exchange or refund policy. But too many of its shipments were the wrong size, or the wrong color, or just didn't appeal to the customers. In the end, the credit card penalties were the straw that broke the camel's back [721].

This history suggests that technological fixes may not be as easy as many vendors claim, and that the main recources will be essentially procedural. American Express has announced that it will offer its customers credit card numbers that can be used once only; this will protect customers from some of the scams. In order not to run out of numbers, they will issue them one at a time to customers via their Web site (which will drive lots of traffic to them) [204]. Many other bankers are already coming to the conclusion that the way forward lies with better address verification, rather than with cryptography [62].

However, if you're working as a security engineer, then a lot of your clients will want to talk about technical matters, such as encrypting credit card numbers, so we'll look at the available crypto mechanisms anyway.

19.5 Cryptographic Protection Mechanisms

The existing cryptographic protection mechanisms used by the bank card industry—the PINs used at ATMs and some point-of-sale terminals, and the CVVs described in Section 19.3.2, which make card forgery more difficult—are largely ineffective online, so new mechanisms were developed. The most widely used is the Secure Sockets Layer protocol (SSL), an encryption system bundled with most Web browsers.

19.5.1 SSL

Recall that in public key encryption, a server can publish a public key *KS* and any Web browser can then send a message *M* containing a credit card number to it, encrypted using *KS:* $\{M\}_{KS}$. This is, in essence, what SSL does, although in practice it is more complicated. SSL was developed to support encryption and authentication in both directions, so that both `http` requests and responses could be protected against both eavesdropping and manipulation.

Here is a simplified description of the SSL version used to protect Web pages that solicit credit card numbers.

1. The client sends a *client hello* message to the server, which contains its name *C*, a transaction serial number *C#*, and a random nonce N_C.
2. The server sends a *server hello* message, which contains its name *S*, a transaction serial number *S#*, a random nonce N_S, and a certificate *CS* containing

its public key *KS*. The client now checks the certificate *CS* back to a root certificate issued by a company such as Verisign and stored in the browser.

3. The client sends a *key exchange* message containing a *pre-master-secret* key K_0, encrypted under the server public key *KS*. It also sends a *finished* message with a message authentication code (MAC) computed on all the messages to date. The key for this MAC is the *master-secret* K_1. This key is computed by hashing the pre-master-secret key with the nonces sent by the client and server: $K_1 = h(K_0, N_C, N_S)$. From this point onward, all the traffic is encrypted; we'll write this as $\{...\}_{KCS}$ in the client-server direction and $\{...\}_{KSC}$ from the server to the client. These keys are generated in turn by hashing the nonces with K_1.
4. The server also sends a *finished* message with a MAC computed on all the messages to date. It then finally starts sending the data.

$C \rightarrow S$: $C, C\#, N_C$

$S \rightarrow C$: $S, S\#, N_S, CS$

$C \rightarrow S$: $\{K_0\}_{KS}$

$C \rightarrow S$: $\{finished, MAC(K_1, everything\_to\_date)\}_{KCS}$

$S \rightarrow C$: $\{finished, MAC(K_1, everything\_to\_date)\}_{KSC}, \{data\}_{KSC}$

The SSL design goals included minimizing the load on the browser, and then minimizing the load on the server. Thus, the public key encryption operation is done by the client, and the decryption by the server; the standard encryption method (*ciphersuite*) uses RSA, for which encryption can be arranged to be much faster than decryption. (This was a wrong design decision, as browsers generally have a lot more compute cycles to spare than servers; the use of RSA has created a brisk aftermarket for crypto accelerator boards for Web servers.) Also, once a client and server have established a pre-master-secret, no more public key operations are needed, as further master-secrets can be obtained by hashing it with new nonces.

The full SSL protocol is more complex than this, and has gone through a number of versions. It supports a number of different ciphersuites; for example, export versions of browsers can be limited to 40-bit keys—a condition of export licensing that was imposed for many years by the U.S. government. Other ciphersuites support signed Diffie-Hellman key exchanges for transient keys, to provide forward and backward secrecy. SSL also has options for bidirectional authentication, so that if the client also has a certificate, this can be checked by the server. In addition, the working keys *KCS* and *KSC* can contain separate subkeys for encryption and authentication. For example, the most commonly used ciphersuite uses the stream cipher RC4 for the former and HMAC for the latter, and these need separate keys.

Although early versions of SSL had a number of bugs [784], the latest version (called TLS by Microsoft) appears to be sound (but it has to be implemented carefully [116]). It is being used for much more than electronic commerce—an example being medical privacy, where it's likely to replace proprietary networks and allow confidential patient data to be sent over the Internet [175]. SSL is also spreading from Web applications, and is now incorporated as an option in Win2K, where it can be used to set up secure sessions between machines in different domains. There are problems,

though, and some of the most glaring have to do with the nature and management of certificates. We'll talk about them in Section 19.5.3.

19.5.2 SET

Early experience with credit cards and the Internet showed that the real risk of compromise of credit card numbers did not come from eavesdropping on IP traffic—of which no case has ever been confirmed—but from hacking merchant Web servers and other end systems, which often retain credit card numbers, and are frequently attacked. So, in 1995–1996, there was an effort to develop a better payment protocol, which would use digital signatures rather than credit card numbers.

Eventually, a consortium which included Microsoft, Netscape, VISA, and MasterCard, came up with the *Secure Electronic Transaction* (SET) protocol. The ideas behind SET were:

- Customers will have public key certificates too, not just merchants. So customers can sign transactions, which include payment orders to their banks.
- The customer signs and enciphers two separate messages, one to the merchant, which contains a description of the goods and the price but not the credit card number; and another to the bank, which contains the price and the credit card number but not the description of the goods. The signatures are linked.
- The back-end transaction processing from the acquiring bank to the brand to the card-issuing bank uses the existing legacy systems.

SET was supposed to reassure customers that online transactions would be secure, and to reduce the cost of fraud (mainly by denying credit card numbers to merchants). The business model was that SET transactions would be treated as if the cardholder were present; that is, the bank would assume the fraud risk, and the merchant discount would be lower.

Because of the number of different legacy systems that had to be supported, and the range of features demanded by various industry players, SET is even more complex than SSL. Again, I'll give a simplified account of it.

First, the customer sends the merchant server her certificate *CC* for her public key *KC*, and a nonce, N_C. The server replies with certificated public keys for the merchant (*CS, KS*) and its bank (*CB, KB*), plus a transaction sequence number *S#*. Then the customer sends a message containing an order, encrypted under the merchant's public key, and a payment instruction, encrypted under the bank's public key. Hashes of both of these are signed with the customer's private signing key. Next is an authorization step, which can be performed online or deferred, as appropriate: the server sends the payment instruction to the acquiring bank, together with a summary of the order, which includes the amount payable but not the exact description of the goods. The bank checks all this and refers to the card-issuing bank, if necessary. If everything's in order, it sends the server an authorization response similar to the traditional one (with an amount and an authorization code), fortified with a signature.

$C \rightarrow S$: C, N_C, CC

$S \rightarrow C$: $S, S\#, CS, CB$

$C \rightarrow S$: $\{Order\}_{KC}$, $\{Payment\}_{KB}$, sig_{KC} $\{h(Order), h(Payment)\}$

$S \rightarrow B$: $\{Summary\}_{KB}$, $\{Payment\}_{KB}$

$B \rightarrow S$: sig_{KB} $\{Auth\_response\}$

SET appears to have met its specifications, but failed to succeed in the marketplace. The reasons are instructive.

- *First, the benefits turned out to be less than expected.* Many large merchants were breaking their cardholder agreements by retaining customer credit card numbers—principally for use as indexes in marketing databases—and were not prepared to stop using them. So a feature was added whereby merchants could get the credit card number from the acquiring bank. This was thought to negate much of the hoped-for security improvement. (In fact, it wasn't that bad as banks could have issued credit card numbers that were valid only for SET transactions, so stealing them wouldn't have mattered.)
- *Second, the costs were too high.* Building a public key infrastructure to issue all credit cardholders with public key certificates would have been enormously expensive. Performance was also an issue.
- *Third, there was nothing in it for the customers.* Customers trading on the Web under MOTO rules could reverse the transaction if they were unhappy—not just about the payment, but about the service, the product, or anything else. Using SET transferred them to cardholder-present rules, and in many countries removed this protection. Thus, customers were much worse off and would have been insane to use SET. Also, installing SET usually involved downloading megabytes of SET wallet and going through a laborious certification procedure.

In the end, SET cost too much and delivered too little; and as far as the customers were concerned, it was a disaster. It is being allowed to expire quietly. The main lesson is, perhaps, that when designing systems for e-business, you should deal with issues as they are in practice, rather than in theory, and think about how your design will affect the interests of principals other than your client.

19.5.3 PKI

Public key infrastructures (PKIs) are still an issue; there is frequent semantic confusion between "public (key infrastructure)" and "(public key) infrastructure." In the first, the infrastructure can be used by whatever new applications come along; I call this an *open PKI;* in the second, it can't; I call this a *closed PKI.*

Examples of open PKIs are:

- Merchants using SSL are supposed to have certificates for their public keys, and several companies such as Verisign will certify a public key as belonging to a particular company after doing appropriate due diligence.
- There are many proposals to base new online services, and particularly business-to-business services, on certified digital signatures.

- Many governments are thinking of issuing their citizens with public key certificates, probably in smartcards, as next-generation identity cards. Although most businesses are really only interested in whether they will be paid, governments offer a range of services (such as tax and welfare) that can be cheated by people who can masquerade as more than one person. So there is much government interest in promoting the use of PKI technology, and this has led to legislation, which I'll discuss in Chapter 21.

The classic examples of closed PKIs are to be found in the networks operated by military agencies and by banking service providers such as SWIFT, which use asymmetric cryptography but do not publish any keys. Now that Win2K includes SSL as an authentication mechanism, it can be used to set up secure wide area networking across a number of scattered sites in a company; and if the number of sites is at all large, this may involve the company operating its own PKI to manage the keys. So closed PKIs may become much more common; and even where a service using asymmetric cryptography is offered to the public, there may be no keys published. An example is the Mondex electronic purse, which uses RSA cryptography and further protects the keys in tamper-resistant smartcards.

At the time of writing, PKI was one of the most heavily promoted protection technologies. However, it has a number of intrinsic limitations, many of which have to do with the first interpretation—namely that the infrastructure is provided as a public service that anyone can use. I discussed many of the underlying problems in Chapter 6. Naming is difficult; and a certificate saying, "Ross Anderson has the right to administer the machine foo.com" means little in a world with dozens of people of that name.

One way to solve the naming problem is for each business to run its own closed PKI, which might be thought of at the system level as giving customers a unique account number which isn't shared with anyone else. This leads to the "one key or many" debate. Should I expect to have a single signing key to replace each of the metal keys, credit cards, swipe access cards, and other tokens that I currently carry around? Or should each of these be replaced by a different signing key? The second option is more convenient for business as sharing access tokens can lead to huge administrative costs and liability issues. It also protects the customer: I don't want to have to use a key with which I can remortgage my house to make calls from a payphone. It's just too easy to dupe me into signing a message by having the equipment display another, innocuous, one. (I don't know how to be confident even of a digital signature I make on my own PC, and I've worked in security for over fifteen years. Checking all the software in the critical path between the display and the signature software is way beyond my patience.) But the existing PKI machinery was largely developed to provide an electronic replacement for the telephone book, and tends to assume that everyone will have a unique name and a unique key. This in turn means an open PKI architecture.

This leads to political issues, such as, which CAs do we trust, and why? Various attempts have been made by governments to license certification authorities and to impose a condition that there be "back doors" for law enforcement access. Governments overwhelmingly favor the one-key-fits-all model of the world. It's also possible that open PKIs will be favored by network economics, which I discuss in Section 19.6: once a single PKI becomes dominant, the pressure on everyone to use it could lead to its being entrenched as a monopoly. (This is the reason for the high stock market valuation of VeriSign.)

There are numerous issues of implementation detail. For example, the dominant certificate format (X.509) does not have the kind of flexible and globally scalable 'hot card' system that the credit card industry has developed over the years. It rather assumes that anyone relying on a certificate can download a *certificate revocation list* from the issuing authority. This is tiresome and inefficient Better ways of managing certificate revocation have been proposed; the question is whether they'll get implemented. Also, X.509 is designed to certify names, when for most purposes people want to certify an authorization.

There are many other limitations of certificates:

- *Most users disable the security features on their browsers*, even if these weren't disabled by default when the software shipped. Recall that the third step of the SSL protocol was for the client browser to check the certificate against its stored root certificates. If the check fails, the browser may ask the client for permission to proceed; but the way most browsers are configured, it will just proceed anyway. This lets many e-commerce sites save themselves money by using expired certificates or even self-signed certificates; most users don't see the warnings (and wouldn't know how to respond if they did).
- *The main vendors' certificates bind a company name to a DNS name*, but are not authorities on either; and they go out of their way to deny all liability.
- *Competition in the certificate markets is blocked* by the need to get a new root certificate into Microsoft Internet Explorer (VeriSign stockholders will consider this to be not a bug but a feature).
- *Even when you do get a valid certificate, it may be for a company and/or DNS name different from that of the site you thought you were shopping at*, because the Web site hosting or the credit card acquisition was outsourced.
- *U.S. export regulations have meant that large numbers of sites use weak encryption.* A recent survey of SSL security showed that of 8,081 different secure Web servers, 32% weren't, for various reasons—too-short keys, weak ciphersuites, and expired certificates being the main causes [567].

There is also a serious problem with consumer protection law. Introducing a presumption that digital signatures are valid undermines the signer's rights; in paper systems the risk of fraud is borne by the party who relies on the signature. In the absence of such a presumption, it makes no difference to the cardholder's liability whether the sites at which he shopped had valid certificates with appropriate names; and in any case, there's no convenient way for him to record his transactions to show that he exercised due diligence. I go into some of these issues at greater length in Chapter 21.

In short, while public key infrastructures can be useful in some applications, they are unlikely to be the universal solution to security problems as their advocates seem to believe. They don't tackle most of the really important issues at all.

19.5.4 EDI and Business-to-Business Systems

The early examples of electronic commerce given in Section 19.2, such as the telegraph and EDI, were largely business-to-business systems. These systems provide

many of the examples of working PKIs. We looked at an example in detail in Section 9.3.1—the SWIFT network used since the mid-1970s to send secure payment messages between banks in different countries, and upgraded in the 1990s to use public key techniques. Essentially, the same architecture has been adopted for a number of other systems, including the CREST system used to register ownership of all U.K.-listed equities and to support share dealing between banks and brokers.

A more modern example of a business-to-business system is Bolero, an EU system that handles electronic bills of lading [262, 458, 492]. These are the legal documents that confer ownership of shipping cargoes. Their average value is about $25,000, but, depending on the price of oil, an oil tanker's cargo can be worth $100,000,000. Cargoes are often traded many times while the ship is at sea, and many of the traders are rather dubious shell companies. In addition, the enforcement of sanctions against rogue states means that their national intelligence agencies are often involved in using such shell companies to buy things like oil when they shouldn't. So this is a high-value, high-threat environment. Quite some care has to be taken to ensure that bills of lading cannot be duplicated, and that their ownership history can be established.

Bolero uses two main protection mechanisms. First, there is tamper resistance. IBM 4753 or 4758 cryptoprocessor cards are used to hold the bills; and there is a protocol involving digital signatures that is used to transfer them. Second, a central registry is used to ensure that the name of the unique holder can be determined at all times. There is also a registration authority (which vets the credentials of organizations) and a certification authority (which signs the public keys of individuals authorized by registered organizations).

Another example is healthcare EDI systems, which typically send test results, such as radiology and cytology from large hospitals and laboratories, to family doctors and local clinics. Here the requirement isn't non-duplicability, but authenticity plus confidentiality. Early systems provided this using closed, proprietary messaging networks; more modern systems have encryption and authentication mechanisms that work at message gateways. The dozens of messages sent each day from a hospital laboratory to a general medical practice are batched up, signed with the hospital's signing key, encrypted with the practice's public key, and shipped by email. In an ideal world, messages would be signed individually by the consultant writing the opinion, and addressed to the physician treating the patient; in practice, this is hard, as both labs and physicians use proprietary systems that can communicate only using EDIFACT gateways. These are specialized systems that do format conversion; they are too expensive for everyone to have them running on their PC. This design causes a number of problems. For example, if a record of the digital signatures of lab reports is to be kept in case of malpractice litigation, then the whole batch must be kept, which conflicts with privacy rules about the destruction of records pertaining to patients who have died or moved away.

This kind of problem isn't limited to healthcare. Many systems have message processing functions that cause subsets of transaction data to go to different destinations in an organization. Protecting the integrity of structured data can be much harder than it looks.

So, for a number of reasons, implementing business-to-business secure communications isn't at all straightforward. As innovative business models proliferate, there are bound to be expensive design errors. I'll discuss how to reduce the probability of these in Part 3.

19.5.5 E-Purses and Micropayments

In the first half of the 1990s, a large number of *electronic purse* systems were developed. The idea was to replace debit cards with something that would work offline without being more vulnerable to forgery. The typical implementation involved both customers and merchants having chip cards, each with a value counter, plus a card-to-card payment protocol whereby two cards would authenticate each other—one would be debited and the other would be credited. I described the design of a typical protocol in Section 2.7.1.

The promoters of these schemes had huge hopes. Europay Austria, for example, was confident that its product would displace 20% of cash transactions within three years. But by the end of the decade, the outcome was disappointing. Even where an e-purse chip has been built into standard bank cards and issued to the entire customer base, as with Proton in Belgium and the Geldkarte in Germany, usage remains disappointing. (The business is surveyed in [763].) The slow take-off might have been expected, given the history of credit cards; like them, e-purses suffer from what economists call *network externalities*, meaning that the more the cards are used, the more merchants accept them, and the more useful they become—not just to their existing customers but to potential adopters. But when few people use them, merchants don't have a motive to buy terminals; and with few terminals, they attract few users. At best, such schemes have some way to go before they reach critical mass; they may get there eventually, driven by applications such as payphones, and then they could grow very quickly indeed. I discuss the underlying economics further in Section 19.6.

The one exception to the slow start imposed on new payment mechanisms by network externalities may be the micropayment mechanism that might be shipped with the second phase of third-generation mobile phones, which I described in Section 17.3.4. As all purchasers of next-generation phones will be using this system to pay for calls, as well as for value-added services, the extra cost involved in adding a new service should be low, while the customer base should be substantial within a few years. There is a risk that, as with Geldkarte, this will end up as a system that's widely deployed but has few active users; but there's also the possibility that it might fly. (Phone-based payments could also become a serious problem if, as seems likely, third-generation phones get the capability to run Java applets—with all the potential for malicious code, feature interaction, and general confusion that this would bring.)

It's also conceivable that third-generation phones could provide a close thing to a universal open PKI, if certification authorities are used to sign users' public key certificates in a way that's usable by other services. On the other hand, if the top-level public key protocol is the Royal Holloway one, and (as in its established government uses), this escrows signing keys as well as confidentiality keys [50], then it will be much less useful, because the evidential value of signatures produced by the system will be undermined.

19.6 Network Economics

The network externalities mentioned above don't just apply to people trying to launch new payment systems, but are of very much wider importance. Many communications systems, and the downstream applications they support, suffer from them. They dictate

not just e-commerce business models, but also many of the problems that the security engineer must wrestle with.

The key observation is that the more users there are on a system, the more people there are to talk to, and so the more useful it is to each user. (This is sometimes referred to as *Metcalfe's law.*) There are many documented examples of how the resulting positive feedback can give a large advantage to the early players in a market, and lead toward a monopoly. For example, in the early years of the twentieth century, AT&T's dominance of long-distance telephone communication enabled it to crush local competition and to establish itself as the near-monopoly provider in the United States until its break-up in 1984. Compatability is also crucial. In the 1950s, there was a battle between CBS and RCA over whose standard for color television would prevail; although the FCC endorsed CBS, its standard wasn't backward-compatible with the millions of installed black-and-white TV sets. RCA's standard was, so it won.

The most obvious effect of this positive feedback is that once a network passes a certain critical size, it grows rapidly. The telegraph, the telephone, the fax machine and, most recently, the Internet, have all followed this model. Another effect is that there are enormous rewards for being first. This happens to some extent in traditional businesses such as cars where supplyside economies of scale help big firms to grow bigger. But there are limits: once they became too large, companies such as General Motors couldn't react quickly enough to compete against upstarts such as Toyota. With networks, the economies of scale occur on the demand side, and so there is no upper limit on growth.

Network effects aren't limited to the kind of networks that involve shipping electrical or optical signals, which economists refer to as *real networks*. They also apply to *virtual networks*, of which the classic example is software. Recall the battle for supremacy between the PC and the Mac in the mid-1980s: once it was clear that there were going to be more users of PCs than Macs, software houses concentrated on shipping their products for the PC first and the Mac only afterward, if at all. This meant that there was more software for the PC, so people were more likely to buy PCs, and the positive feedback continued.

This isn't limited to PC architecures, but works at all sorts of levels including application software and file formats. Once most people started using Microsoft Word for documents, that was even more reason for everybody else to (regardless of the risk from Word macro viruses). And such a network effect can be enhanced by arranging that your file formats are difficult for other companies' programs to read, and by changing them often; I'll discuss this in the next section.

One of the features of markets exposed to network effects is lock-in. This can be either technology lock-in or vendor lock-in. Technology lock-in often involves complementary suppliers, as with the software vendors whose bandwagon effect carried Microsoft to victory over Apple. A side effect of this is that successful networks appeal to complementary suppliers rather than to users—the potential creators of "killer apps" need to be courted. Once the customers have a substantial investment in complementary assets, they will be locked in. Andrew Odlyzko observes that much of the lack of user-friendliness of both Microsoft software and the Internet is due to the fact that both Microsoft and the Internet achieved success by appealing to developers. The support costs that Microsoft dumps on users—and, in fact, even the cost of the time wasted waiting for PCs to boot up and shut down—greatly exceed its turnover [595].

So there are three particularly important features of information technology markets.

- *First, technology often has high fixed costs and low marginal costs.* The first copy of a chip or a software package may cost millions to produce, but subsequent copies may cost very little. This isn't unique to information markets; it's also seen in business sectors such as airlines and hotels. In all such sectors, pricing at marginal cost will tend to drive revenues steadily down toward the cost of production (which in the case of information is zero).
- *Second, there are often high costs to users from switching technologies, which leads to lock-in.* Such markets may remain very profitable, despite the low marginal costs.
- *Third, there are often network externalities of the sort discussed earlier.* The value of a product to a user depends on how many other users adopt it.

All three of these effects tend to lead to winner-take-all market structures with dominant firms. Indeed, firms will attempt to manipulate these effects to gain competitive advantage in various ways.

One common strategy, for example, is differentiated pricing. This means pricing the product or service not to its cost but to its value to the customer. This is familiar from the world of air travel: you can spend $200 to fly the Atlantic in coach class, $2,000 in business class, or $5,000 in first. (As noted in Section 19.3.3, this is also a classic case of Goldilocks pricing: the main function of the first-class fare is to enable people who fly business class to claim they are being economical.) This business model is spreading widely in the software and online services sectors. A basic program or service may be available free; a much better one for a subscription; and a "gold" service at a ridiculous price. In many cases, the program is the same except that some features are disabled for the budget user while the "gold" user gets a high-quality helpline. Many of the protection mechanisms you will come across have as their real function the maintenance of this differential.

Another strategy is to manipulate *switching costs.* The long-term value to your ISP of your account, in the sense of the discounted future earnings, should be equal to the total amount of money (and hassle) involved in the customers' switching to a competitor. So the ISP will do its utmost to make it easy for you to switch to them, but difficult to switch from them. This applies with particular force to dominant-firm markets where the incumbent tries to build a monopoly which its competitors try to attack. Incumbents try to increase the cost of switching, whether by indirect methods, such as controlling marketing channels and building industries of complementary suppliers, or by direct methods, such as making systems incompatible and hard to reverse-engineer. Meanwhile, market entrants try to do the reverse: they look for ways to reuse the base of complementary products and services, and to reverse-engineer whatever protection the incumbent builds in. They may use penetration pricing—selling cheaply to subsidize switching—and the incumbent may respond with vaporware, designed to increase users' perception of the opportunity cost of switching.

As technology advances, even seemingly impregnable monopolies can be replaced, so the competition can be vicious. Extensive use is made of protection mechanisms, from tamper-resistant devices to proprietary encryption algorithms.

19.7 Competitive Applications and Corporate Warfare

This leads us to the applications of information security mechanisms whose goal isn't to protect the customers or their data, but to either entrench or attack a monopoly.

Sometimes the mechanisms in use are obvious. For example, manufacturers of game consoles try to keep their platforms closed so that they can monopolize sales of accessories and impose conditions on games software vendors—which may involve not just royalties but also exclusivity agreements. Legal solutions such as copyrighted interfaces aren't enough, as in many countries there is an exemption from copyright law for firms doing reverse-engineering in order to build a compatible product. So cryptographic challenge-response protocols are used to authenticate genuine accessories and game cartridges; competitors hire reverse-engineering labs to dig out the keys. I mentioned other applications of accessory control in Section 2.2.

Another example comes from Microsoft Passport. This is a system whose ostensible purpose is single sign-on: a Passport user doesn't have to think up separate passwords for each participating Web site, with all the attendant hassle and risk. Instead, some sites use Passport, a central authentication server run by Microsoft, to which users log on. Servers use Web redirection to connect their Passport-carrying visitors to this server; authentication requests and responses are passed between them by the user's browser in encrypted cookies. So far, so good.

But the real functions of Passport are somewhat more subtle [727]. First, by patching itself into all the Web transactions of participating sites, Microsoft can collect a huge amount of data about online shopping habits, and enable participants to swap it. The redirection and cookie mechanisms mean that, in effect, all the browsing sessions you have at participating sites become one single session, managed by Microsoft. If every site can exchange data with every other site, then the value of a network of Web sites is the square of the number of sites, and there is a strong network externality. So one such network may come to dominate, and Microsoft hopes to own it. Second, the authentication protocols used between the merchant servers and the Passport server are proprietary variants of Kerberos, meaning the Web server must use Microsoft software rather than Apache or Netscape. In short, Passport isn't as much a security product as a play for control of both the Web server and purchasing information markets. It comes bundled with services such as Hotmail, is already used by 40 million people, and does 400 authentications per second on average. Its known flaws include that Microsoft keeps all the users' credit card details, creating a huge target; various possible middleperson attacks; and that a user can be impersonated by someone who steals their cookie file. Passport has a "logout" facility that's supposed to delete the cookies for a particular merchant, so users can use a shared PC with less risk, but this feature doesn't work properly for Netscape users [473].

The constant struggles to entrench or undermine monopolies, and to segment and control markets, determine many of the environmental conditions that make the security engineer's work harder. The markup language XML enables document content to be processed easily, and has the potential to build in a rich syntax of protection attributes [46]. However, it's not taking off as many people had hoped, because if Web pages become easily machine-readable, then comparison shopping bots are easier to build. In addition, many online merchants gauge demand by discounting perhaps every hundredth transaction by a random amount. This wouldn't work if shoppers had tools to hit the site again and again until they got a bargain. In general, there is a constant

struggle between the designers of intermediaries—from Web caches to anonymous communication services—who wish to control a user's transactions is various ways, and merchant sites, that wish to break this control and "own" the user directly.

I'll come back to the effects of network economics on security in Section 22.6.

19.8 What Else Goes Wrong

An important survey of the things that went wrong with First Virtual, one of the first online banks, revealed that the typical problem was an upset customer calling at three in the morning and wanting to speak to someone in Korean about a missing payment. Investigation would typically reveal that the cause was an obscurely broken implementation of one or more Internet protocols, or a mistake made in typing an email address. Solving such problems is not currently a core competence of the typical financial institution [129].

This pattern—that most of the problems come from unanticipated bugs and blunders—was to be expected from other application areas; and it has continued. A number of large online stores, including `Buy. com`, `Staples. com` and Amazon-backed crafts retailer `eZiba. com`, were hit by pricing errors. At `Buy. com`, a coupon meant to be worth $50 off any order of $500 or more actually gave $50 off any purchase, making any item sold for $50 or less free to the buyer. At `eZiba. com`, each customer was offered a $20 voucher; people logged on with multiple names and found that they could use the discount many times [671]. Errors like these are nothing new; there have been numerous cases in the past of a special offer being carelessly designed, or even just being more popular than planned for. The difference is that, on the Net, an error can become widely known very rapidly and lead to large losses.

There have also been some interesting attacks on specific systems. The *Radio Data System* (RDS) adds a data channel to broadcast radio so the receiver can tell which station it is and what sort of content is being broadcast. This enables the radio to switch automatically to the strongest transmitter for your favorite network when you're driving along in your car; you can also program the radio to interrupt you if traffic information appears on another channel. (An RDS radio has two tuners, one of which constantly sweeps the band looking for a higher-strength or higher-priority signal). Pirate radio stations have developed the trick of marking their content falsely as traffic news, so lots of car radios switch to them automatically [306]. This isn't always obvious if the pirate is playing the same general kind of music as the station you were just listening to. It's also not obvious how genuine radio stations could be authenticated; perhaps the FCC would act as a certification authority, and issue certificates along with spectrum licenses. But then how would a genuine station that turned pirate be revoked? The moral is that often it is necessary to protect the integrity of distribution channels; and, more generally, that interesting new features often turn out to be vulnerable to interesting new exploits.

Computer games are another fertile field for finding hacks. An example is Quake, a distributed game whose source code is openly distributed. Some players have exploited

their access to the code to modify the Quake client so that they can cheat [633]. This raises a number of questions, particularly with e-commerce applications that employ Java applets or other code that is run on a potentially hostile customer machine. There's no obvious way such applets can protect themselves against virtualization. At the very least, it seems prudent to have a non-Java version of your site. This will not only mean that you can do business with people like me, who leave Java turned off for security reasons; it will also give you a fallback mechanism to block fraud based on bad applets without having to close down your site for redevelopment.

19.9 What Can a Merchant Do?

In general, the advice I give to people concerned about e-commerce risks is that they are not much different from normal business and IT risks.

You should make sure your developers understand the business model; use a structured development methodology; test your code thoroughly; don't be too clever; look around for proven ideas which you can adopt; by all means worry about the firewall through which your internal systems are connected to the Net, but pay particular attention to internal controls to discourage, prevent, and detect insider fraud. Expect that no matter what you do, things will go wrong and may have to be changed quickly. Don't be too greedy; if you put "We guarantee you can't lose money shopping online with us" on your Web page and "Our records are the sole and definitive evidence of all transactions between us" in the small print, you may expect to have the TV crew from a consumer rights program camping on your doorstep one day.

This much is motherhood and apple pie for any company IT director. The main way in which e-commerce appears different—at least as of mid-2000—is the seriously enhanced risk and cost of customers repudiating credit card transactions. There's no obvious fix; but a useful damage limitation strategy is to have a controlled procedure through which customers are invited to complain, rather than going directly to their credit card company on day one. For example, you can advertise a policy of allowing exchanges with a discount off the postage; and if customers do want a refund, ask them to print out and sign a form, and mail it to you by physical post, rather than offering a Web form for the purpose.

Above all, focus on the business risks, and have risk management documentation that you upgrade regularly in consultation with your auditors, insurers, and directors, rather than getting carried away with the latest technical security gizmos. Business is business, and just because there are now highly paid computer scientists designing business processes that used to be the domain of the work study clerks of old, this does not mean that things are any better. The risks associated with the "Net mentality" have to be carefully assessed and managed.

Finally, it may make sense to pay some attention to non-technical issues such as product liability. One critical advantage enjoyed by U.S. e-businesses is that foreign civil judgments aren't enforced by U.S. courts. So an e-business operating from the United States doesn't have to worry about being sued by consumers in faraway coun-

tries, even if the local law lets them sue locally, because the judgments they get cannot be enforced. But in Europe, there are international reciprocal agreements: a U.K. retailer sued in a local Greek court can have the judgment enforced in the U.K. On the other hand, U.S. courts are accessible and can give punitive damages, while courts in many European countries are so expensive and give such small awards that product liability suits are rare. These considerations interact with credit card chargeback issues, but are not subsumed by them. They can be so complex that all I'll say is that you should get legal advice: the best location for your business may depend on what you're selling, and to whom.

19.10 Summary

Most of the problems facing online businesses are no different from those facing other organizations, and the network security risks are not much different from those facing traditional businesses. The real increased risks to an e-business have to do with ways in which traditional risk management mechanisms don't scale properly from a world of local physical transactions to one of worldwide, dematerialized ones. Credit card transaction repudiation is the main example at present. There are also significant risks to rapidly growing companies that have hired a lot of new staff but that don't have the traditional internal controls in place.

Research Problems

Most of the research behind the e-commerce protection mechanisms that are already deployed, or about to be, was done around 1994–1996. It may well be time for a second wave now that we can see what has worked, what can work but failed in the marketplace, and where the real problems are.

Further Reading

The early history of the telegraph can be found in a book by Major General RFH Nalder [569], while Tom Standage tells the story of its rapid deployment in Victorian times [729]. There is a survey of organized credit card counterfeiting in [592]. The official specification of SSL is hard to read; a better exposition is in [604]. The SET protocol is described in a book by its chief architect Li Song [509]. The problems of public key certification and infrastructures are analyzed in [42, 268]. Finally, the best book I know on network economics is by Carl Shapiro and Hal Varian [696].