The SSL Reference Implementation Project

Author: Jeremy Bradley
Supervisor: Neil Davies

Abstract

Information is now the most valuable resource in the world. Whether it is a personal letter or an industrial secret, all information has a worth to someone. This thesis considers issues of security and privacy for such information. It discusses the reasons for wishing to protect data and the methods available for doing so.

Specifically, the project concerns cryptography and secure communications, in particular the Secure Sockets Layer (SSL) protocol. It contains a communications and security analysis of SSL as well as details of the construction of a secure reference implementation of the protocol.

"*We have it in our power to begin the world over again*"

Thomas Paine, 1737–1809

Declaration

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stage was any collaboration entered into with any other party.

Any views expressed within or implied by the dissertation are those of the author and should be attributed as such. They should not be taken as representative of any policy or official statement of the University of Bristol.

This thesis has been submitted in accordance with the rules of the Faculty of Engineering and has not previously been entered for any other research qualification at any other institution.

Jeremy Bradley,
2nd October 1995.

Contents Page

Appendices

List of Figures and Tables

1. Introduction

With the explosion in public interest in the Internet comes a corresponding fear for privacy. Is someone recording everything I do? Are hackers reading my e-mail? The media portrayal of modern computer networks has not exactly helped clear the gloom either. On the one hand we are told that he Internet will "revolutionise your poultry existence", and on the other it becomes apparent that it is festooned with "anarchistic criminals" bent on the destruction of civilisation and the misuse of your credit card!

It is unheard of these days to see articles in the media extolling the virtues of the printing press or condemning the existence of the radio set. And yet these were new and unknown technologies in their time and both had their proponents and their detractors. It is only with the gradual building of public trust does a technology become accepted. It is not until a box of gadgets can sit in someone's living room without attracting comment that a technology can claim to have become part of society.

This is the hurdle which computer technology and the Internet in particular have to overcome. The public at large have to be convinced that it is a safe and benign place to tread. It should be possible for a newcomer to the technology to open a bank account electronically in the certain knowledge that it is as secure as if the account had been opened in person. It should be possible for someone to send a personal e-mail to a friend with the assurance that no one else can read it on the way. It should be possible for people to connect their computers to the Internet without fear of anyone reading private files.

While this project does not in any way pretend to achieve these feats of social engineering, it does recount a possible technological step in that direction. The project describes a technology which may offer some of the assurances which businesses and the public have been looking for. It concerns the use of cryptography to protect private information; information belonging to companies; information belonging to governments; information belonging to individuals; information that we decide not to disclose or information that we only wish to be shared with people of our choice.

Chapter 1 deals with the rights and wrongs, advantages and disadvantages and different points of view concerning encryption. It is meant to act as an apolitical justification of the project, although as the reader will see, it is an area which tends to polarize into extreme points of view with very little in the middle. Chapter 2 summarizes the technical state of cryptography today. It is not meant to be a detailed expose of the subject, however it is presented in order to give readers a brief overview so that they can better understand the issues that follow.

Chapter 3 introduces, then describes and analyses in detail, the Secure Sockets Layer Protocol which is the main subject of this thesis. The project culminates with the software engineering of a product based on SSL and presented in Chapter 4. The Appendices are reserved for technical or very detailed documentation which would not add a great deal to the main body of the report, but is nevertheless related to and heavily referenced by the remainder of the thesis.

2. The Need for Secure Communications

2.1 Secure Communications

The distribution of electronic media over the last few decades has seen an equivalent rise in the need for privacy on such channels. Whether communicating via telephone, fax or email, there is an unseen and largely unprotected line of transmission which neither the communicating party or the carrier can wholly vouch for. There are many areas of society which have shown an interest in communication privacy and in recent times, these areas have largely driven research into cryptography as a means of protecting information in transit.

2.2 Network Insecurities

It is always going to be difficult to assess the true extent to which private information maybe being intercepted. However it is not entirely unreasonable to expect that at one stage or another attempts will probably be made to gain information by illicit means. Whether these threats are perceived or real, this is an area in which erring on the side of caution is definitely the safest path. Although accurate statistical evidence is hard to come by there has been much in the way of anecdotal evidence to suggest that current channels are being infiltrated.

There are three main public communications networks to consider:

1. Radio and Satellite Transmission
Many analogue cellular phone systems have no scrambling or encoding mechanism at all such that any third party with a radio receiver can pick up mobile phone calls. In recent times many cases have come to light of such calls being overheard and recorded.

Most digital transmissions, while not as easy to pick up, currently use no encryption and are transmitted using publicly available protocols.

2. Terrestrial Telecoms
Public terrestrial telephone networks have been shown to be vulnerable as well. In some networks, any unauthorised individual can tap into the controlling digital exchanges and

reroute, shutdown or listen in on any conversation or data transmission. This phenomenon called "Phreaking" is especially prevalent in large cities such as New York, which have a large hacker community. Many phone companies are unwilling to address these problems in public, as to do so would be seen as an admission that these activities occur. Such an admission might cause a loss of confidence in a service, leading to a reduction of business.

3. High Speed Data Networks (ISDN, Internet)
High speed data networks are certainly not immune either. The Internet is a widely distributed network and it would not be a difficult task to surreptitiously record data packets as they pass through a particular node. Although this attack is complicated by having to recombine the packets into the correct order to regain the full transmission.

2.3 Information At Risk

Now that the potential for network attack has been established, it is necessary to discuss the different types of information that might require secure networks and data encryption.

2.3.1 Commercial Information

> [BAR92.3] "We have the largest information based economy in the world. We have lots of reasons for wanting to protect information..."

As companies become more effective by using the latest communications technologies, so more and more "critical" information is being committed to computers and networks. This information might include:

1. details of a takeover bid and share price
2. future business strategy
3. commercially secret product details

If information like this gets intercepted by corporate rivals then it can cause great harm to the company. However it is important to quantify more precisely the possible consequences of such disclosures in a society which relies heavily on the free market to regulate itself.

In the case of (1), premature disclosure of a possible takeover bid can have dramatic effects on a stock market. Invariably the share price of the desired company rises sharply as demand for its shares increases. This may cost the predator company billions more than it would otherwise have had to spend in order to achieve the takeover. Such extra expenditure is either picked up by the customer or the share-holder and either way the cost would filter down to society, whether through poorly performing pension schemes or increased prices.

Alternatively, if the information pertaining to a takeover is obtained by an individual who does not make it public, then that person has the opportunity to make a large amount of money.

Without risk of being found out, the individual can indulge in “insider-dealing” in the shares of the target company and thus make the stock exchange carry the cost of the crime. After many such incidents a market might collapse, as has happened in some South American economies where drugs barons have used market influence and information to insider-deal to excess.

It is worth repeating that in the past, obtaining such corporate secrets by intercepting communications was not as likely because that sort of information would not have been kept on computer or knowingly transmitted insecurely.

2.3.2 Financial Transactions

> [LEV93.6] “Privacy and authenticity technologies are key to reliable and trustworthy social and business interactions over networks.”

The detection of financial transactions, if not encrypted, poses a much more immediate threat to society. Obviously, the power behind an attack here, lies not so much in reading of data but changing of data. In an instance where Alice is sending an electronic money transfer to Bob, if careful cryptographic protocols are not used, Eve could intercept the message. It would then be a simple matter for Eve to change the recipient of the transfer and possibly even the amount, so that she gets credited with the money.

However most inter-bank electronic transfers are being encrypted (albeit with only medium grade encryption, DES-56), thus changing data and passing it off as a real transfer is much more difficult, but not impossible (see Section 3.2.4 on Key Length). But the mere encryption of financial transfer data is often not enough to protect against more sophisticated attacks. A “replay” attack involves catching a message or sequence of messages and replaying them many times to the recipient. This message might instruct a bank to increment an account by £10 (a genuine message), however if intercepted by the owner of the account and replayed, say 10,000 times, it would add £100,000 to the account - which is effectively robbing the bank.

Thus financial transactions require, not just encryption, but an entire communications protocol to counteract such attacks.

Potentially more damaging for the individual is the sending of credit-card details over unprotected networks like the Internet. A lot of the insecurity in credit card transactions lies in the credit card system itself. So, for instance, if a customer wants to buy a product over the phone or by post, no customer identification or authorization is required, only the information that is on the card: number and expiry date. So what makes information transmitted over a computer network any more vulnerable than the same information over the telephone or in the post? The difference lies in the technology over which they run. It would be relatively easy to set up a data-packet filter at a busy Internet node, which could scan, extract and save any packet with phrases such as “credit card” or “expiry date” in them. Even though such messages might only occur in 1 in every 1000 (a fact which makes a similar attack on the postal or telephone systems impractical) if a node gets 1,000,000 messages a day, that would give a fraudster 1000 sets of credit card details a day.

The outlook for share transactions over the Internet is just as bleak if proper encryption standards are not used. Currently only very weak encryption is available for anyone who is not a financial institution (see Section 2.5.2 on Export Regulations).

2.3.3 Individual Privacy

> [SCH94] “He may be planning a political campaign, discussing his taxes or having an illicit affair. He may be designing a new product, discussing a marketing strategy or planning a hostile business takeover. He may be living in a country that does not respect the rights of privacy of its citizens. He may be doing something that he feels shouldn’t be illegal but is. Whatever his reasons, his data and communications are personal, private, and nobody’s business but his own.”

Justifying the use of encryption as a de facto right for individuals lies more in the political arena of libertarianism than in the realm of Computer Science. However many argue that this is where encryption can possibly have the most beneficial effect.

The argument goes as follows:

People that take the view that freedom of expression, freedom to communicate with whosoever you wish and a right to privacy, are basic tenets for living in a modern democracy, see encryption of all public communication networks simply as a means to protecting that democracy.

Experience has shown that given the opportunity, a ruling political party may succumb to temptation and attempt to gain advantage by eves-dropping on its political adversaries. It is not a very large step from trying to listen to what other people say about you, to trying to prevent them saying it in public. Once this road has been trod, a country loses the free exchange and discussion of ideas which maintain democratic rule and more importantly, society’s acceptance of democratic rule.

The fear of the loss of democracy is a real one, expressed by many informed academics, including recently Ross Anderson of Cambridge University, in a television interview:

> [HEL95.3, EQX95.3] “The transactions which make up our daily lives are rapidly becoming electronic. If we’re denied the means to protect them, we’re not just talking about hackers and electronic crime. We’re talking about a surveillance society in which authority will know our every detail of our lives. Even Hitler and Stalin couldn’t have dreamt of that.”

This is backed up by anecdotal evidence from countries with oppressive regimes. In places like China where political activism is regarded by the incumbent dictatorships as a crime, personal use of cryptography is the only means to achieve freedom of speech, the free discussion of ideas and privacy from the government. In some of these countries the freedom which cryptography brings is the only thing that has kept the democratic movements alive. It seems natural to extend this to a free society where the widespread use of cryptography would act as a safety mechanism against the day that a tyrannical government did try to seize power, as happened in Germany in the 1930s, for instance.

This view was taken by John Gilmore, an american millionaire computer programmer who was threatened with imprisonment when he discovered a classified cryptography book in an american library. He gave evidence to Congress at the time the US government was debating communications technology policy:

> [LEV93.6] "Government investment invariably brings government control, which is harmful to the development of a communications medium in a free and open society."

2.3.4 Military Information

The dangers involved if military information is compromised during time of crisis, are far more clear cut and obvious. It is perhaps unsurprising that governments are more easily persuaded of such threats and so have no hesitation in allowing the strongest security measures to be put in place for military standard communications. It is important that a nation implements its own cryptography in this instance as it has become apparent that cryptographic material imported from another country is likely to be rigged in such a way that it can be broken or in some way compromised.

2.4 Perceived Disadvantages of Encryption

2.4.1 Law Enforcement

When it became apparent that encrypted communications were set to become common-place, due mainly to the increase in desktop computing power, law enforcement agencies started lobbying government to prevent widespread use. The case made by many law enforcement agencies is very similar and is well summarized below by various advisors to the US government:

> [LEV93.2] "We have the capability of 100% privacy, but if we use this I don't think society can survive."

> [LEV93.3] "If we fail to enact legislation that will ensure a continued capability for court-ordered electronic surveillance, systems... would become sanctuaries for criminality wherein organized crime leaders, drug dealers, terrorists, and other criminals. Eventually we could find ourselves with an increase in major crimes against society, a greatly diminished capacity to fight them, and no timely solution."

Similar sentiments were presented by former CIA Director, Admiral Stansfield Turner:

> [BAR92.1] "A threat could develop... terrorism, narcotics, whatever... where the public would be pleased that all electronic traffic was open to decryption. You can't legislate something which forecloses the possibility of meeting that kind of emergency."

The same government advisors are suggesting banning encryption altogether. This from an former NSA Director, Admiral Bobby Ray Inman:

> [BAR92.2] "My answer would be legislation which would make it a criminal offence to use encrypted communication to conceal criminal activity."

It transpires that many law enforcement agencies wish to have untrammelled access to digital communication transmissions without recourse to the communication provider company. Apparently wiretaps have proved difficult to get when telephone company employees are in the pay of criminals too.

2.4.2 Intelligence Gathering

Obtaining evidence, or even details of instances where cryptography has hindered intelligence gathering activity is very difficult. By its very nature such activities are kept under close guard and even denied completely, even though it is widely reported in the press that agencies like the NSA in the US, GCHQ in the UK and DGSE in France regularly monitor communications which leave and enter their respective countries. It is even reported from sources close to these agencies that they actively record all electronic traffic on the Internet and other digital media.

One would expect these agencies to keep quiet, if they have developed analytical methods to enable them to break codes which are thought of as strong in the academic community. On the other, if they found that cryptographic usage was making their work more difficult, one would look for accounts of pressure being brought to bear on individuals and companies who use such methods.

It would appear that, in the UK anyway, the latter is the case, as a recent anonymous post to an Internet newsgroup testified. (PGP is a cryptographic protocol, freely available and widely believed to be extremely secure)

> "I've been telling this company for 18 months or so about the advantages of PGP and email rather than faxes, and they finally tried it, liked it, and decided to use it a month or so ago. Last week they got a visit from the Department of Trade and Industry and MI5... telling them to stop using PGP or they might find difficulties getting export licences for their products in future. The visitors wouldn't say how they knew this company had been using it."

Similar pressure has purportedly also been put on communications service providers in this country:

> [LEV93] "Britain's two cellular operators, Vodafone and Cellnet, both say they have heard hints... that various police and security services... would be happier if they could eavesdrop on domestic conversations carried on [cellular phone networks]..."

Another interesting anomaly that must arise in this industry, is that since many of the agencies, described above, also advise their governments on how to keep their own governmental communications secure, it is difficult to see how such agencies are able to intercept the communications of other nation states. It would appear that the attention of the intelligence organizations must be turning to the communications of individual citizens and businesses who cannot use, or are not allowed to use, strong encryption. This is certainly born out by this anecdote which appears in many texts:

> [LEV93] "Recently, the head of the French Intelligence service quite cheerfully admitted intercepting confidential IBM documents and handing them over to French-government-backed competitors."

And again from Martin Hellman, a well respected cryptographer.

> [SCH94.2] "The situation is very different with [the] NSA and its foreign equivalents, especially the French Direction Generale de la Securite Exterieure (DGSE), which has openly boasted of using commercial intelligence to help French companies win bidding wars for large contracts. They are legally encouraged to spy, have large budgets, and routinely intercept vast amounts of data."

The above might well explain the NSA's self-contradictory and newly-revised view of cryptography for US businesses, although interestingly not US citizens:

> [LEV93.5] "As for domestic use of cryptography, we have always supported the use of cryptographic products by US businesses operating domestically and overseas to protect their sensitive and proprietary information."

2.5 Preventing the Spread of Cryptography

2.5.1 Restriction of Academic Research

> [LEV93.6] "Current government policies actively prohibit and inhibit the research, design, manufacture, sale and use of [encryption] technologies."

Rather worryingly governments have, for a long time, tried to restrict the study of cryptography in universities. Since the second world war, this has taken the form of employing anyone who looked like they might be any good at the discipline. This was very successful and between 1949 and 1967 nothing was published in the field!

However in the late 1960s it became more difficult to prevent external research, as the number of people studying the subject rapidly grew. This was mainly due to a ground-breaking book by David Kahn [KAH67] which contained a "remarkably complete history" [SCH94.1] of codes, and inspired many people including famous contemporary cryptographers.

When open research into cryptography did start to take off in the 1970s, the NSA decided to act and tried to prevent arguably the most important cryptographic paper ever, the RSA paper [RSA78], from being presented at an IEEE conference in 1977:

> [BAM83, LEV93.1] "...the first salvo in the conflict was a letter written in July 1977 by an NSA employee named Joseph A. Meyer. It warned those planning to attend an upcoming symposium on cryptography that participation might be unlawful under an Arms Regulation law (ITAR), which controls weapons found on the US munitions list."

The NSA had to retreat fast when it came under withering attack for trying to inhibit the academic process and attempting to remove the American constitutional right of "Freedom of Speech". So the paper was given. However the NSA was not finished as Whitfield Diffie describes:

> [SCH94.1] "A more serious attempt occurred in 1980, when [the] NSA funded the American Council on Education to examine the issue with a view to persuading Congress to give it legal control of publications in the field of cryptography."

This legislative attempt failed as well, when Congress only allowed a voluntary submission scheme to be set up. This was a process whereby a researcher could voluntarily ask the NSA if their paper would "serve the National Interest" by being published. To the best of anyone's knowledge this process has never been used.

2.5.2 Legislation

Export Restrictions

The first line of attack for any government trying to restrict the spread of encryption software is to institute a ban on its export. Some countries notably France and Russia also put bans on import as well. The position of the UK with respect to the legalities of export, with regard to this project, is discussed in Section 5.6 (UK Laws on Export of Cryptography).

One country which has had export bans in place for a number of years is the US where, on the recommendation of the NSA, export licences have to be applied for, and software reviewed if any export product contains cryptographic material.

The US implementation of these export restrictions was called ITAR (International Traffic in Arms Regulation) [STA89], under which cryptography is classified as a weapon of war, and restricted accordingly. Under ITAR, with few exceptions, the export of strong and weak cryptography was illegal. In July 1992, in an agreement with the Software Publishers

Association (SPA) [WAS94], the NSA agreed to the export of cryptography with weak session keys, but only for two proprietary ciphers (RSADSI's RC4 and RC2 with 40 bit key schedules, see Section 3.2 on Symmetric Key Cryptography). This concession was made only after significant pressure from the US computer industry which had seen many security related contracts won by European rivals who were unfettered by such legislation. The new US export regulations, which coincided with the SPA allowance, are called DTR (Defense Trade Regulations) [STA92] and are still in place today.

The reasoning given for ITAR control on cryptography was ostensibly to prevent such technology getting into the hands of foreign adversaries. It has since emerged that a useful side-effect of the law has been to inhibit the spread of secure communications within the US. American companies are unwilling to pay the cost of development and distribution for a domestic (US) version and an international version of a piece of software. The result has been that a single version is sold worldwide which conforms to the weakened export requirements. The irony has been that the original intention, that of preventing opponents outside the US of using cryptography, has completely failed. So much so, that some political observers maintain that it was never the US governments aim in the first place, especially given the fact that every nation state in the world almost certainly uses strong cryptography at a government level and did so when the first regulations were published. In a study commissioned by the SPA [HOF94], on the availability of strong cryptography overseas, the conclusion stated:

> [HOF94] "It cannot be any clearer: the existence of widespread and affordable cryptographic products overseas is an indisputable fact. Based on that fact, unilateral US export controls keep US firms from competing in the global marketplace."

This scenario, of export regulation to limit internal use, seems to be backed up by successive Directors of the NSA, notably Admiral Bobby Ray Inman [BAR92.2] and Vice-Admiral DM McConnell [SST94.1], who when questioned about the necessity of export controls on cryptography, reply with answers which concern domestic law enforcement, not intelligence gathering:

> [BAR92.2] "The Department of Justice has a very legitimate worry. The major weapon against white collar crime has been the court-ordered wiretap. If the criminal elements go to using a high quality cipher, the principal defense against narcotics traffic is gone."

This is odd given that it is expressly forbidden in law for the NSA to wiretap its own citizens on US soil and it was intended that the NSA be barred from undertaking any domestic surveillance at all. However this has been undermined by recent revelations about NSA bases in the UK such as that on Menwith Hill, North Yorkshire:

> [GOL95] "The NSA's British sites can evade US domestic legislation against phone tapping... Menwith could easily listen to UK citizens as well."

GCHQ is under no such legal obligations:

> [GOL95] "...the recent Intelligence Services Act allows GCHQ to 'monitor or interfere with electromagnetic, acoustic and other emissions'."

US Digital Telephony Bill

A second offensive against cryptography being used in standard telephone networks emerged in 1991. The FBI lobbied the industry and then Senate very heavily, to allow them to be able to require the telephone companies to provide an unencrypted transcript of telephone conversations and data transmissions. The first attempt at this came in the form of a clause entered into the 1991 Anti-Crime Bill [GAR95.1] by Senator Joseph R. Biden (Democrat-Delaware):

> **Cooperation of Telecommunications Providers with Law Enforcement Agencies**
> It is the sense of Congress that providers of electronic communications services and manufacturers of electronic communications service equipment shall ensure that communications systems permit the government to obtain the plain text contents of voice, data, and other communications when appropriately authorized by law.

After huge pressure from the computing industry, the clause was removed. A further attempt was made in 1992, this time in the form of The Digital Telephony Bill. This too failed but in 1994, a revised Digital Telephony Bill was passed by the US government. The new bill places all technological developments and standards (including cryptography) used by the private communications companies in the hands of the US Justice Department at a cost of $500 million.

Key Escrow and Encryption Bans

The third and possibly most direct form of legislation which many governments are considering and some have already partially instituted is the banning of strong cryptography altogether and the forced introduction of key escrow systems.

Key escrow is a form of encryption in which the secret key is shared by a third party. It was initially devised for the purposes of an electronic will, where in the event of death, the executor could use his copy of the key to decrypt the will. However in this case the third party is the government or law enforcement agency. The government stores all these private keys in a "secure" archive somewhere, then in the event of needing to tap someone's phone they use this key to decrypt the conversation.

Key escrow is seen by government as a good compromise between the needs of business, the individual and law enforcement as this statement form a DTI official shows:

> [HEL95.2, EQX95.2] "Such a policy will need to balance the legitimate requirements of industry, commerce and individuals for a range of cryptographic services with the national security needs of the authorities in fighting terrorism and crime."

However the NSA, again talking about matters of domestic law enforcement not intelligence, told a Senate committee that escrow technologies would not catch criminals:

> [SST94.1] "Key escrow encryption is not meant to be a tool to catch criminals. It will make excellent encryption available to legitimate businesses and private citizens without allowing criminals to use the telecommunications system to plan and commit crimes with impunity."

Clipper

The first version of this came to light on 16 April, 1993 when the US government introduced a hardware encryption system, which supported key escrow, called the Clipper chip. Rather than ban strong encryption as well, they tried to get it into widespread use by using "market forces". This was meant to be achieved by subsidising the mass manufacture of the chips which reduced the price per unit cost to roughly $20 and insisting that anyone that wished to communicate with government agencies would have to use Clipper. There were also lots of rules and regulations regarding the distribution of keys to law enforcement agents, but it turned out that these were only recommendations and not legally binding.

In 1994 an AT&T Bell Labs researcher called Mat Blaze, was reviewing the security of the Clipper system when he discovered that it was possible to use the Clipper to provide excellent encryption (80 bit Skipjack, a classified bulk cipher algorithm) as intended but by doing some insignificant pre-processing the key escrow feature could be disabled. This revelation has meant the scrapping of millions of dollars worth of Clipper chips which in turn has angered many in the computer industry who invested heavily in it. Further this point was picked up by many academics as the result of forbidding research and peer-review of government encryption standards. Previously the NSA had thought that their staff of cryptographers was large enough and technically adept enough for them to be able to avoid public external review procedures. But as Bruce Schneier notes:

> [SCH94] "Encryption is too important to be left to the government."

European Council Recommendations

Meanwhile disclosures in the British press suggest that UK government is considering similar proposals:

> [HEL95.1, EQX95.1] "The project [funded by the Government and Vodafone] is being used as an excuse to try to develop the British key escrow system..."

Also very recently it has come to light that on 9 September 1995 the 34 nation European Council decided on a policy on cryptography which will role key escrow, digital telephony and banning strong encryption into one European Directive.

In 18 September 1995 issue of "Communications Week International" magazine, in an article entitled "Euro-Clipper Chip Scheme Proposed", it states that the Council "has agreed to outlaw strong encryption products which do not make keys available to governments" as well as "[making] telecoms operators responsible for decrypting traffic and supplying it to governments when asked" and finally "[putting] restrictions on the possession, distribution, or use of cryptography."

2.6 The Real National Interest

Cryptography as a science cannot be uninvented, anyone with a personal computer, access to a library and the inclination to do so, can implement the very strongest encryption systems.

Many people who support the use of encryption think that law enforcement officials should be worried if encryption is not allowed to take off. As has been discussed earlier in the report, the results of electronic crime if our communications remain unencrypted is potentially devastating to our economies and democracies.

Alternatively, reducing the security of allowable cryptography is not safe since it has already been shown that if one country can decrypt a coded message then nearly every other country in the world can too. This is especially true since it seems that many country's use their intelligence agencies to obtain commercial and industrial secrets, amongst other things. This view is also backed up by the famous cryptographer Ron Rivest:

> [BAR92.3] "...weakening our encryptions for the convenience of law enforcement doesn't serve the national interest."

Similarly the idea of key escrow is hardly a safeguard against future or present administrations misusing the technology to spy on political adversaries or interfere with the democratic process. Key escrow has other draw backs as well:

1. Key escrow systems are not exportable, so there would be no business growth in this market. The US would like to sell Clipper technology abroad, but other nations should regard placing the security of their industries in the hands of a foreign power as not in their National Interest.

2. Escrow databases would be very vulnerable to attack from criminals or other intelligence agencies because of their massive value. The ability to decrypt the entire information infrastructure of a nation would be worth a lot of money to a lot of people, and the existence of a single database which would comprise such an infrastructure would be a target worth expending a lot of money on. If an individual did obtain an escrow database, it could be sold to another intelligence agency, used to black-mail the country of origin with, or simply published on a newsgroup thus crippling a country's ability to communicate securely. The conclusion must be that the very existence of such databases would not be in the National Interest either.

Finally the complete bans on cryptography, as proposed by some governments, would be completely unenforceable and would in no way prevent those criminals that wished to pursue illegal activities by using cryptography, from doing so. Not that law enforcement does not have a place in a modern society, but in the same way that law enforcement officials have to obtain court orders before searching a premises, so a court order should be required before forced data decryption should be allowed. In that way a legitimate discussion of the facts of a case would occur, the proper procedures would be gone through and the rights of the individual protected.

Meanwhile, as author Tim May notes, the people whose privacy is at stake from legislation like this have two very different concepts to deal with:

> [LEV93.4] "[People] have two dichotomous views held exactly at the same time, one view is 'None of your damn business, a man's home is his castle. What I do is my business', and the other is 'What have you got to hide? If you didn't have anything to hide, you wouldn't be using cryptography.' There's a deep suspicion of people who want to keep things secret."

Many people, including journalist John Barlow, feel that this "suspicion" does tend to be fed with nightmare scenarios of nuclear terrorism and uncontrollable drug trafficking, by political adversaries of encryption:

> [BAR92] "I think we can expect our drifting and confused hardliners to burn the Reichstag repeatedly until they have managed to extract from our induced alarm the sort of government which makes them feel safe."

The last word on the subject is left to Stephen Levy who writes for Wired magazine:

> [LEV93] "If privacy is outlawed, only outlaws will have privacy."

3. The Current State of Encryption

3.1 Definitions

Cryptography	the science of making ciphers
Cryptanalysis	the science of breaking ciphers
Cryptology	the combination of cryptography and cryptanalysis
Plaintext	the original message prior to encryption
Ciphertext	the scrambled plaintext after encryption

3.2 Symmetric Key Cryptography

Most of the cryptographic research this century has been into symmetric key technology. A symmetric key cipher uses a secret sequence of characters or key to scramble any amount of plaintext into a unique ciphertext. A different key, used to scramble the same plaintext, will yield a completely different ciphertext. The plaintext is recovered by rerunning the same algorithm with the same key over the ciphertext.

There are two types of symmetric cipher in common usage:

The **block cipher** takes a fixed length of plaintext (called the block size) and generates the same amount of ciphertext. If the total length of the plaintext is not a multiple of the block size, then padding data may be used to make up the difference on the last block of plaintext.

The **stream cipher** converts plaintext to ciphertext one bit at a time.

3.2.1 Common Symmetric Ciphers

DES (Data Encryption Standard) emerged from IBM in the early 1970s as an algorithm called Lucifer and was modified by the NSA to produce the “Federal Standard” [NBS77]. It is a block cipher with a block size of 64 bits and a fixed key length of 56 bits.

IDEA (International Data Encryption Algorithm) was developed by Xuejia Lai and James Massey in 1990 and later improved in 1991 [LAI91]. It is a block cipher with 64 bit block size and has a fixed key length of 128 bits.

RC2 (Ron's Code 2) was originally designed by Ron Rivest, 1 to 2 years ago and has since been used exclusively by RSA Data Security Inc. It has never been published and has not been patented by RSADSI. RC2 is a block cipher with block size of 64 bits and variable key length. As such it can be used as a drop in replacement for DES, although with a larger key, it could be significantly more secure. Common key lengths are 40 bit, 56 bit and 128 bit.

RC4 (Ron's Code 4) differs from RC2 by being a stream cipher. It also has variable key length. Common key lengths are 40 bit and 128 bit. Although RC4 has never been published, an implementation was reverse engineered in 1994 and posted to a newsgroup, thus placing it in the public domain. The legal position of using such an implementation in the US is still unclear.

3.2.2 Cryptanalytic Attacks

1. Ciphertext-only attack
A cryptanalyst has the ciphertext from one or more messages, encrypted using the same cipher and is able to deduce the the plaintext or the key.

2. Known-plaintext attack
In this case, both the ciphertext and corresponding plaintext for one or more messages are available from which plaintext or key are obtained.

3. Chosen-plaintext attack
This differs from the known-plaintext attack in that the cryptanalyst is allowed to choose the plaintext which is encrypted.

4. Adaptive-chosen-plaintext attack
Similar to the chosen-plaintext attack, except that the cryptanalyst can choose subsequent plaintext on the basis of previously selected plaintext/ciphertext pairs.

5. Chosen-ciphertext attack
For a symmetric cipher, identical to chosen-plaintext attack. For an asymmetric (public-key) cipher this differs by letting the cryptanalyst choosing the ciphertext which is decrypted.

6. Chosen-key or Brute-force attack
This is not really cryptanalysis, but it is still a valid attack. If the cipher forces the cryptanalyst to resort to this method then the cipher is regarded as strong. The attack consists of going through every possible key, deciphering the ciphertext with each key, and attempting to find some recognisable plaintext in the result. So for a 40 bit key, there would be 2^{40} possible keys of which, on average, 2^{39} would have to be tested.

3.2.3 Security

An encryption algorithm is **unconditionally secure** if, given any amount of ciphertext, there is never enough information to recover the original plaintext, even with infinite computational resources. The one-time pad [DEN82, SCH94] is the only known unconditionally secure cipher.

An encryption algorithm is **computationally secure** if, given all the finite resources conceivably available now or in the future, it still cannot be broken.

A useful tenet, when considering the strength of a new algorithm is given below. It nicely sums up the state of mind one should adopt when considering the cryptanalytical potency of an adversary. Assume the worst; that way, if you are wrong, security is maintained.

> [SCH94] "Assume that some cryptanalysts have access to all the details of your algorithm. Assume they... can mount an intensive ciphertext-only attack. Assume they can mount a plaintext attack... Even assume they can mount a chosen-plaintext attack. If your cryptosystem can remain secure even in the face of all that knowledge, then you've got the security you need."

3.2.4 Key Length

The benefit of forcing an opponent to fall back on a brute-force attack to crack your code is that the key length can be used as a quantitative metric for comparison with other algorithms. For instance, a 16 bit key would only take seconds to search, on any modern computer, no matter which of the common symmetric ciphers was used. Whereas a 128 bit key would provide secure encryption for hundreds of years, even when future advances in computer technology are taken into account.

When considering a particular encryption algorithm, a decision has to made over the value of the original data and the time period over which it must be kept secure. The key length provides the means to assess these issues.

Consider a hardware based attack, where the algorithm has been programmed onto a dedicated processor. Assuming the following specialised encryption hardware speeds are possible for the given year and that an array of 4000 such chips is produced to work in parallel on the problem.

Year	Encryptions per Second ('000,000)	Cost of 4000 Chip Array (£)
1995	4	520,000
2000	32	550,000
2005	256	490,000

Fig 1. Table of encryption hardware and estimated cost

Key Length (bits)	1995	2000	2005
40	68 seconds	8.6 seconds	1.07 seconds
56	7.4 weeks	6.5 days	19 hours
64	36.7 years	4.6 years	6.9 months
128	6.7e17 millennia	8.4e16 millennia	1.1e16 millennia

Fig 2. Table of time needed to break certain key sizes using hardware

These figures are estimates derived from the more detailed tables in [SCH94]. Note that the cost of such an attack is well within the means of a medium sized company's R&D budget. Also, it is widely believed that the current state of the art allows for the manufacture of chips with encryptions speeds of up to 256 million encryptions per second. Although such technology is probably only available in the range of hundreds of millions of pounds, this is still well within the range of major intelligence organizations.

The existence of such dedicated hardware devices might seem unlikely. However the following reasoning certainly suggests that a dedicated DES cracker already exists, given the perceived dangers during the Cold War.

[SCH94] "In the 1980s, the East German company Robotron fabricated hundreds of thousands of DES chips for the Soviet Union (they claimed it was for encryptor chips in IBM-PC XT clones). This implies one of two things: the Soviet Union used these chips to build a DES cracker, or the Soviet Union used DES to encrypt its own communications - which means the NSA built one."

An organization may well fear an attack from a group of individuals more than it does from a major government or rival company. In which case a software attack on current standard personal computers is more likely.

Now consider a distributed software attack, where the cipher, the ciphertext and an allocation of key-space have been given to 200 computers, which can each muster the following computer power.

Year	Encryptions per Second ('000)
1995	50
2000	400
2005	3200

Fig 3. Table of speeds of a software attack

Key Length (bits)	1995	2000	2005
40	1.3 days	3.8 hours	28.6 minutes
56	228.5 years	28.6 years	3.6 years
64	58.5 millennia	7.3 millennia	914.0 years
128	1.1e21 millennia	1.3e20 millennia	1.7e19 millennia

Fig 4. Table of time needed to break certain key sizes using software

A software attack of this nature is not fanciful. In August 1995, a group of just over 100 people donated the idle time from the computers at their disposal to attempt to break a 40 bit key. The attempt was successful in that only 30 hours later, the key was found and the message decrypted. Furthermore only half the keyspace had been searched. It is important to remember that the figures quoted in this section are for searching the entire key-space and thus guaranteeing a hit; on average it will take half the time stated; if lucky a key could be guessed at the first attempt!

The large figures relating to completion time for software and hardware attacks are very deceptive. They do not necessarily mean that a large-key cipher is safe for that amount of time. For instance, if I have a £500,000 budget and have the task of breaking a 64 bit key in the shortest amount of time, I do not go out and buy £500,000 worth of hardware immediately. If I did it would take 37 years to break the key. To achieve an optimal time period, I would wait 5 years for the price of 32-million-encryptions-per-second chips to fall to within my budget, and then mount an attack with hardware that is 8 times faster than anything I could buy today for the same price. In this way I actually spend a maximum of 4.7 years breaking the cipher and have taken less than 10 years in total to complete the required task.

This reasoning gives totally different figures for the security of the larger key lengths, to both software and hardware attacks. The figures below represent the estimated amount of time that the relevant key sizes will be safe for. The big assumption made in generating the figures is that technological advancement will continue at the same fast pace over the next 100 years (approximately a factor of 8 increase in speed every 5 years). The theoretical justification behind this can be found in [BRD95b], which is reprinted in Appendix C.

Key Length (bits)	Software Attack	Hardware Attack	Intelligence Attack
40	1 day	1 minute	1 second
56	13 years	2 months	1 day
64	26 years	10 years	7 months
128	130 years	115 years	105 years

Fig 5. Table of secure periods of usage for different key sizes

Unfortunately many businesses have failed to realise, or have been mislead about, the strength of the encryption systems which protect their transactions. This quote came from the head of Technology Development at Barclays Bank, while talking about a a system used to protect commercial transactions over the Internet.

> [HAW95.1] "...it would take 64 years, using the world's most powerful computers to break [a 40 bit key]."

As has been seen the unfortunate truth is that a 40 bit key will not survive a concerted software attack for more than a day and a hardware attack for much more than a minute. The misunderstanding came from a figure which stated that it would take 64 mips years to crack the 40 bit key. 1 mips year is equivalent to a 1 mips computer running for a year.

3.3 Random Number Generators for Cipher Keys

> [ECS94] "Generation of unguessable 'random' secret quantities for security use is an essential but difficult task."

Random numbers play a very important part in the security of any cipher system. At some stage a key has to be generated and if it is at all possible to guess the key then the strength of the cipher has been compromised. Generation of random numbers for use in cryptography is one of the hardest security issues to accomplish correctly.

In the RFC [ECS94], many ways are discussed of generating random numbers including using specialised hardware, hardware devices which are commonly found in computers (keyboards, disc drives, mice, etc) and software techniques. The main problem lies in developing a random number system which is cross-platform compatible. Different architectures have very few hardware devices in common. Those that are common to nearly all platforms provide random numbers fairly slowly.

It should be understood that for the purposes of security, random numbers do not just have to conform to a uniform distribution but it also has to be impossible for other people to use the same source of random numbers.

Given the difficulty in obtaining a continuous stream of secret random numbers, a common practice is to use a cryptographically strong pseudo-random number generator. These have the property that, although the numbers are generated from a repeatable formula, it is impossible to recover the formula, just by analysing the numbers (in reasonable time). The seed of the generator has to be known before the sequence can be repeated by another party. It is therefore important that the seed be truly random and secret, not pseudo-random or guessable.

In the same way that the strength of a good cipher should lie in the length of the key, so the strength of a secure pseudo-random number generator should lie in the length of the seed. So if such a generator is seeded with a truly secret and random 128 bit number, then there is said to be 128 bits of uncertainty or entropy in the sequence generated. The danger lies in only having a small amount of entropy in a seed, in this case a systematic search of the uncertain bits will reveal the required sequence of numbers.

When generating such a seed, the danger lies in thinking that just because a 128 bit number has been obtained that necessarily means that it contains 128 bits of entropy. See Section 4.6.2 on the design flaw of SSLref where such a mistake was made leading to much reduced security.

3.4 Public-Key Cryptography

Public-key cryptography was invented by Whitfield Diffie and Martin Hellman in 1976 [DIF76]. The most popular public-key system called RSA was developed by Ron Rivest, Ali Shamir and Leornard Adleman in 1978 [RSA78].

Public-key cryptosystems differ from their symmetric key counterparts in that they use a private key for decrypting and a public key for encrypting rather than having a single key to do both.

Although the public and private keys are mathematically related, the private key cannot feasibly be computed from the public key, so making it safe to publish the public key anywhere. This is achieved by relating the two keys through a mathematical “trap-door” function. A trap-door function is one which is easy to compute in one direction, but which has an inverse which is much more difficult to calculate. So in the case of public-key cryptography, creating the two keys from scratch is easy but attempting to find one from the other is hard.

In the Diffie-Hellman system, the trapdoor function used is the discrete logarithm.
Whereas the RSA setup uses factorisation of large numbers. It is important to realise that although both these problems are thought to be as hard as each other, no one has ever proved that they are mathematically “hard” to break. On the other hand no one has ever proved that they are mathematically “easy” to find either. A theoretical breakthrough along either path would significantly strengthen or weaken public-key cryptography.

RSA relies on the use of large numbers for its security, these large numbers are measured in bit length. A similar discussion to that in the symmetric cipher realm exists concerning how many bits make up a secure RSA key [RIV93]. Recently a 129 digit number was factorised in an 8 month distributed software effort; 129 digits equates to a 429 bit key. It is now recommended people use at least 700 bit keys for security. Encryption and decryption are also inverse operations, no matter which order they are carried out in. This is useful for digital signatures (see Section 3.6 on Certificates).

3.5 Secure Communications Protocols

In implementation, public-key systems are far slower than symmetric ciphers. It is this fact which has lead secure communications protocol designers to use a combination of both to achieve security and speed. The problem with symmetric ciphers has always been one of how the key can be transmitted securely. The advent of public-key cryptography provided the solution to this problem by encrypting the main message using a bulk symmetric cipher and then encrypting the symmetric key using the recipients public key.

Examples of such secure communications protocols include PGP and SSL, the latter being the concern of the remainder of this project.

PGP (Pretty Good Privacy) was designed, originally implemented by Phil Zimmermann and finally released in 1991. It has since been through many revisions and is well described in [GAR95]. PGP supports IDEA as the bulk cipher and RSA public keys of length 512, 768 or 1024 bits.

3.6 Certificates

After securing a communication channel, the next task is authentication. Authentication is perhaps the hardest of all the security issues to solve, being really a matter of trust, no matter how it is implemented.

Authentication is confirming that the person that you are talking to is actually the person you wanted to talk to. For instance, you might find Alice's public-key in a directory somewhere, but how do you know that Eve did not place the key there with Alice's name attached. This type of attack is called a Man-In-The-Middle attack and is just as compromising as if someone had successfully cryptanalysed your bulk cipher.

The Man-In-The-Middle attack occurs when an adversary acts as a third party in a two party conversation. Both legitimate parties assume that they are talking securely with each other; in fact the adversary is intercepting the entire conversation, decrypting it, re-encrypting it and sending it on to the intended recipient.

For example, in public-key protected communication: Bob wants to talk to Alice, but Bob needs Alice's public-key first. Eve intercepts Bob's request for Alice's public-key and instead sends Bob her public-key. Bob uses Eve's public key to encrypt his conversation, thinking that it is Alice's. Eve can now intercept all Bob's messages to Alice, decrypt them, re-encrypt them using Alice's real public-key and send them on to Alice. An entire conversation can pass like this without anyone realising what is happening.

To get round this problem a certificate is used to identify a person (or e-mail address) with a public key. What makes the certificate binding is that it is digitally signed by a certification authority. A certificate will typically consist of the person's name and public key, the certification authorities name and public key and the digital signature. The signature is calculated by hashing the data on the certificate and encrypting the result with the certification authority's private key. The certificate can then be verified by a third party by encrypting the signature block with the certification authority's public key (this amounts to decryption) and comparing the result with the third party's own hash calculation of the certificate. Note that more than one person can sign a certificate and there is nothing to prevent someone signing their own certificate.

In reality all a certificate does is transfer the question of trust from the person to the certification authority. So now the question is how does anyone know that the certificate authority's public key actually does belong to the certification authority in question. The answer, unsurprisingly, lies in a certificate for the certification authority.

3.6.1 PEM Certification

The PEM (Privacy Enhanced Mail) certificate system [KEN93] defines a whole hierarchy of certification. The root authority, the institution everyone should trust since it signs its own certificate, is called the IPRA (Internet PCA Registration Authority). The IPRA only signs certificates of PCAs (Policy Certification Authorities). PCAs in turn sign institutional or corporate certificates who in turn sign individual certificates. The PEM system is still in its infancy and although the IPRA does now exist, it will not disclose the names of any PCAs whose certificates it has signed. People suggest that such intransigence does not inspire the trust necessary to maintain the system's credibility.

An unfortunate consequence of a centralised certification system is that all the necessary administration is expensive, which means that a charge has to be levied on every certificate. As soon as certificates cost money, individuals and non-profit making organizations are going to be excluded from a secure community.

This does not necessarily mean that the notion of centralised certification authorities is unworkable, however. In an electronic society, where commerce becomes common place, it will certainly become necessary to have publicly trusted official certification authorities backed up by a certification hierarchy, in order to protect the consumer against fraud.

3.6.2 PGP Certification

PGP, on the other hand, uses a decentralised certification system called "The Web of Trust". The system is self-certifying, people who already trust each other, sign each others keys. In this way small rings of trusted groups build up, and start to join forming a larger web.

Someone who is not part of the web, or is part of an unconnected subsection of the web, can still communicate with other people, they just have to take it on trust that they are who they say they are. This is not a huge risk when you consider the concept of identity on the Internet. If you are mailing someone you have never met before, in response to a post to a newsgroup for instance, then you do not actually need to know who they are; you are only responding to what they have written. On the other hand if you are passing confidential data to a counterpart in another company, that you have never met before, then you would naturally ring them or meat them in person. Once you actually trust the person, PGP employs a fingerprint method (hash of the public key), to check that the certificate you have got is actually that of the person you met or rang.

The PGP Web of Trust is much the same as the trust method that society uses in the physical world.

The ideal certification system would employ a hybrid of the Web of Trust and the hierarchical system.

3.7 Patents

Most cryptographic algorithms are patented. For details of patent holders on ciphers like IDEA, and all the public-key ciphers see [GAR95, SCH94]. Some patents, in particular those on public-key cryptography, apply only in the US and are not recognised internationally. Most of the non-US patents freely allow any non-commercial usage of the algorithm. However the PKP (Public Key Partners) who hold many american cryptography patents will not consider licensing deals for any non-commercial software (with the exception of RSAref2.0), despite assurances given to standards organisations like ANSI that "non-discriminatory licences would be granted". It is widely thought that such generalized patents have restricted the growth of cryptography in commonplace software in the US.

4. The Secure Sockets Layer Protocol

4.1 Netscape Communications Corporation and The SSL Protocol

Netscape Communications Corporation is a firm which originated in academia. In 1993 Marc Andreessen, then a 22-year-old undergraduate at the University of Illinois, designed and wrote Mosaic, the first graphical browser for the World Wide Web. In April 1994 Andreessen and Jim Clark, a former Stamford University Professor, co-founded Netscape with a much improved browser, the Netscape Navigator. This software, which was distributed free of charge for personal use, has dominated the Internet to the point where an estimated 80% of the total on-line population now use Netscape. Their market success culminated in August 1995 when the company was floated on the stock exchange and was valued at $565 million or $58.25 per share [QUI95], despite never having made an operating profit. Since being formed, Netscape has attracted some notable academics to work for them, including the cryptographer Taher Elgamal, well known for developing the Elgamal public-key signature and encryption system.

Last year Netscape decided to cash in on the potential for electronic commerce on the Internet. This opportunity had not been realised before because of the Internet's perceived insecurity. To rectify this, Netscape decided to develop a secure communications protocol which would allow for the safe transmission of sensitive information such as personal details or credit card numbers. Their commercial plan was to build this security into the popular browsers for free, but to charge for secure servers which could then be used by companies as off-the-shelf "Internet cash registers".

The result of their design activities was the Secure Sockets Layer (SSL) protocol, which was published in Request-For-Comments (RFC) form in October 1994. It has since been through a review process, externally as well as internally, and stabilised as an official protocol design in February 1995 after five revisions [HIC95].

Netscape would like SSL to become the standard secure Internet protocol and have suggested it as such to the Security Committee of the World Wide Web Consortium. A decision by the committee on SSL and other such protocols is expected in early 1996.

4.2 Protocol Export and Patent Issues

The SSL protocol is freely available, exportable and implementable and, as far as I can ascertain, is not patented.

As discussed earlier (see Section 3.5.2 on Export Restrictions), any US company which publishes a truly secure communications protocol will never see that protocol achieve widespread usage due to US export law. Since Netscape are keen to see SSL become a worldwide standard and the lifeblood of any standard is how widely it is used, Netscape made special allowance for weak bulk ciphers in the protocol.

To achieve this, the protocol specifies variants of the RC4-128 and RC2-128 ciphers, which in effect use only 40 bit keys. Now any implementation which conforms to the protocol can still be exported as long as it only uses the weakened ciphers. Note that the keys for the export-grade ciphers are still actually 128 bits in length, however 88 bits of the key is sent in the clear and only 40 bits are RSA encrypted. The reason for this is discussed in the Section 4.4.3 on Strengths of The SSL Protocol.

4.3 Technical Overview of The SSL Protocol

4.3.1 Introduction to SSL

Netscape describe SSL as follows:

> [HIC95] "The SSL Protocol is designed to provide privacy between two communicating applications (a client and a server). Second, the protocol is designed to authenticate the server, and optionally the client."

SSL can layer on top of any transport protocol, it is not TCP/IP dependant, and can run under application protocols such as HTTP, FTP and TELNET.

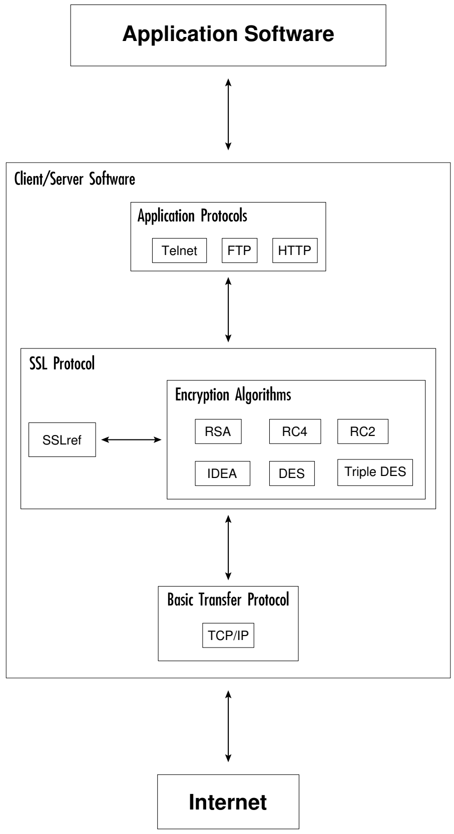

Fig 6. Diagram of relation between SSL, Applications and the Network

SSL uses X.509 certificates for authentication, RSA as its public-key cipher and one of RC4-128, RC2-128, DES, Triple DES or IDEA as its bulk symmetric cipher.

4.3.2 SSL and The ISO Reference Model

It is important that any new communications protocol conform to a standard model if it is to easily replace or become part of an existing protocol structure. The ISO Reference Model for Open Systems Interconnection or 7-Layer Model is the most popular abstraction [HAL93].

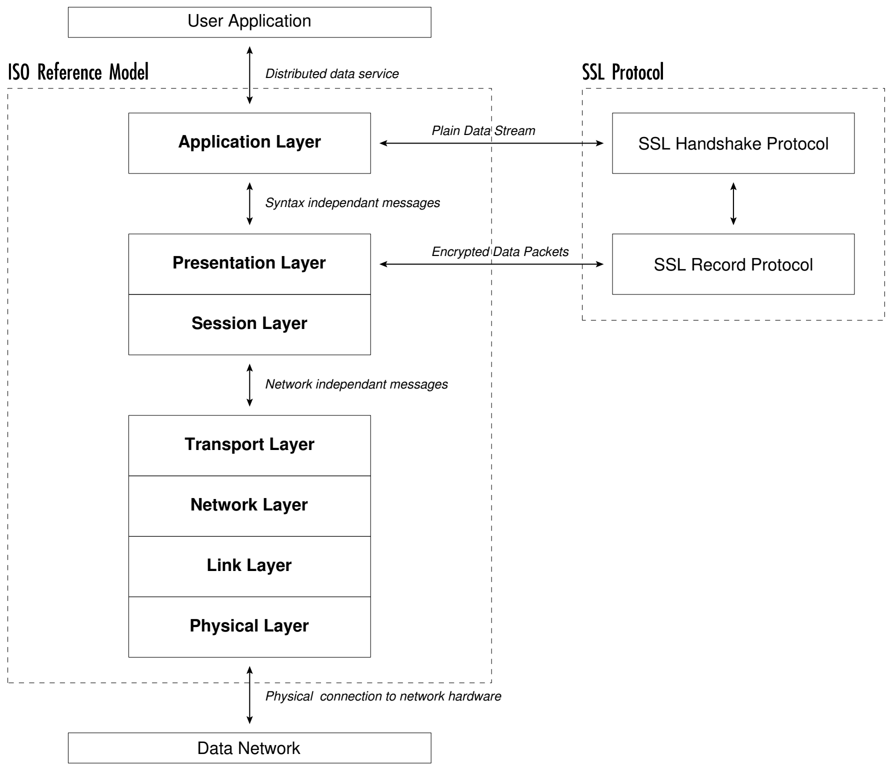

Fig 7. Diagram of relationship between SSL and ISO reference model

The Application Layer of the ISO model with respect to security is described as follows:

> [HAL93] "In addition to information transfer, the application layer provides such services as... agreement on privacy (encryption) mechanisms [and] authentication of an intended communication partner...".

Whereas the Presentation Layer handles the following security-related functionality:

> [HAL93] "Another function of the presentation layer is concerned with data security. In some applications, data sent by an application is first encrypted (enciphered) using a key, which is... known only by the intended recipient presentation layer. The latter decrypts (deciphers) any received data using the corresponding key before passing it on to the intended recipient. ...this is not currently part of the standard..."

This compares to SSL which is conceptually split into two parts, the SSL Handshake Protocol (SSLHP) and the SSL Record Protocol (SSLRP). The SSLHP negotiates which bulk cipher to be used and performs authentication of server and client, if requested. The SSLRP packets the data into records and performs the agreed encryption on them or receives records and decrypts them. The SSLHP and application data layer on top of the SSLRP.

So it can be seen (as in the diagram) that SSL actually rather neatly straddles the two ISO layers with SSLHP being at the Application Layer level and SSLRP at the Presentation Layer level. Since security functions have not been implemented in many protocols, SSL therefore acts as an add-on to such protocols and not a replacement. Also it can be seen that the use of SSL does not preclude the use of other security protocols which operate at a higher level; for instance SHTTP (Secure HyperText Transfer Protocol) [RES94] which is a document or data level security protocol, specifically tailored for financial transactions, maybe layered over SSL.

4.4 Detailed Design Summary

The full specification of the SSL protocol [HIC95] is reprinted in Appendix A. What follows is a detailed summary of the protocol.

4.4.1 The SSL Record Protocol

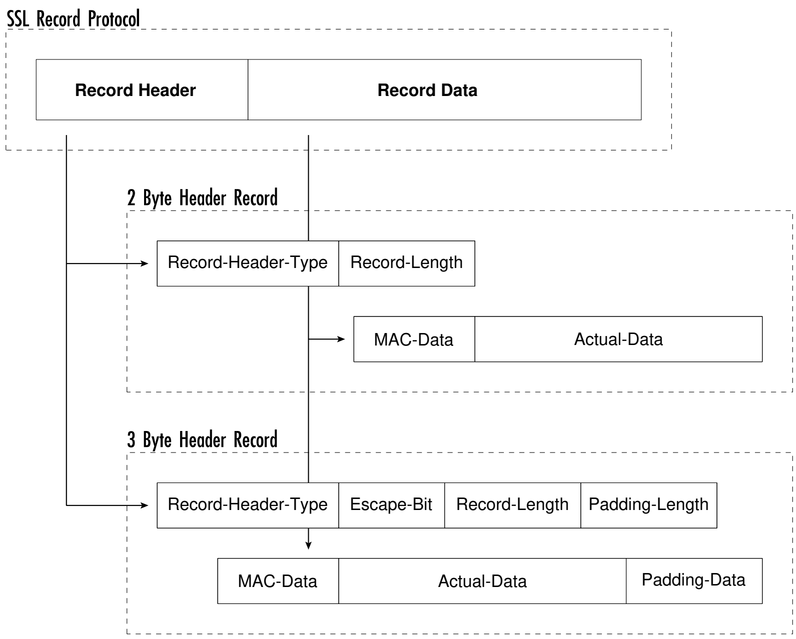

Fig 8. Diagram of SSL Record structures

An SSL record consists of two parts, the header and the data. The header can either be 3 bytes in length or 2 bytes in length, the latter being employed if there is no padding data. The escape bit is not used in version 2 of the protocol but it is suggested that it is used to designate Out-Of-Band data in future versions (see Section 4.7 on The Future of SSL). For a 2 byte header, the maximum record length is 32767 bytes whereas a 3 byte header will only allow a record length of up to 16383 bytes.

The data part of the record consists of a Message Authentication Code (MAC), the actual data itself and padding data, if required. It is the data part of the record which is encrypted in its entirety when encryption is necessary. Padding data is only required for use with block ciphers. It is used to pad out the length of the data block to be a multiple of the block size of the cipher. If a stream cipher is used or the data is already a multiple of the block size, then no padding is required and a 2 byte header record can be used. The MAC is a hash or message digest of the secret write key of the sending party, the actual data, the padding data and a sequence number in that order. The sequence number is a 32 bit integer which is incremented after each message is sent.

4.4.2 The SSL Handshake Protocol

The handshake protocol is composed of two phases. Phase 1 deals with the selection of a cipher, the exchange of a master key and the authentication of the server. Phase 2 handles client authentication, if requested and finishes the handshaking. After the handshake stage is complete, the data transfer between client and server begins. All messages during handshaking and after, are sent over the SSL Record Protocol layer.

No Session Identifier, No Client Authentication

Below is a typical exchange of messages for a first time connection with no client authentication. The following convention is used to indicate encrypted data: "{data}key" means that the "data" has been encrypted with the "key". The table is taken from the SSL specification document [HIC95] with corrections added:

Message Type	Direction	Data Transfered
client-hello	C > S	challenge-data, cipher-specs
server-hello	C < S	connection-id, server-certificate, cipher-specs
client-master-key	C > S	cipher-kind, clear-master-key, {secret-master-key}server-public-key
client-finish	C > S	{connection-id}client-write-key
server-verify	C < S	{challenge-data}server-write-key
server-finish	C < S	{session-id}server-write-key

Fig 9. Table of SSLHP messages: no session-id, no client authentication

The **client-hello** message sends the server some challenge-data and a list of ciphers which the client can support. The challenge-data is used to authenticate the server later on.

The **server-hello** message returns a connection-id, a server certificate and a modified list of ciphers which the client and server can both support. The server certificate is used by the client to obtain the servers public key and verify the identity of the server using any certification authority certificates it has.

The **client-master-key** message is sent in the clear with respect to the SSL record protocol. In this message, the client sends the final choice of cipher (chosen from the cipher list it received from the server-hello message) and the master key sent in two parts. If the chosen cipher is an export grade cipher then 88 bits of the master key are sent unencrypted and only 40 bits are sent encrypted with the servers public key. If on the other hand the client has selected a full strength cipher then the entire key is sent encrypted with the server's public key and the clear-master-key field is empty.

From this point in the handshaking all messages are encrypted at the SSL Record Protocol level. It should be noted that the master key is not actually used as a key for the bulk encryption directly, rather it is used to calculate two sets of keys which are. From now on when the client sends a message to the server, the client uses the client-write-key to encrypt and the server uses the server-read-key to decrypt (client-write-key is the same as server-read-key). Similarly when the server sends a message to the client, the server uses the server-write-key to encrypt and the client uses the client-read-key to decrypt (server-write-key is the same as client-read-key). It is these two sets of keys which are calculated from the client-master-key. Some ciphers require more than one actual key to encrypt and decrypt (Triple DES requires 3 keys each way), so the various write keys and read keys may consist of more than one actual key. This is the reason that the actual session key is not sent and a master key is used.

The **client-finish** message consists of the connection-id originally sent by the server. The connection-id acts as a nonce value which prevents certain attacks (see Section 4.4.3 on Strengths of The SSL Protocol). The message is encrypted using the client-write-key.

The **server-verify** message contains the challenge-data originally sent by the client. The receipt and decryption of this message is the final stage of server authentication in that only the real server should have the private key necessary to decrypt the secret-master-key in the client-master-key message. This message is encrypted using the server-write-key.

Finally the **server-finish** message terminates the handshaking section. It contains a new piece of data generated by the server called the session-id. The session-id is used in subsequent handshakes between the same client and server to avoid having to go through all the cipher and master key negotiation again. Session-ids are cached by each party after a connection is closed and only reused for subsequent connections if they do not occur too far apart. It is recommended that a session-id have a 100 second life in a cache before being discarded.

Session Identifier Used, No Client Authentication

When a previous session identifier is used to re-establish a connection by the client, a different type of client-hello message is sent.

Message Type	Direction	Data Transfered
client-hello	C > S	challenge-data, session-id, cipher-specs
server-hello	C < S	connection-id, session-id-hit
client-finish	C > S	{connection-id}client-write-key
server-verify	C < S	{challenge-data}server-write-key
server-finish	C < S	{session-id}server-write-key

Fig 10. Table of SSLHP messages: session-id used, no client authentication

The **client-hello** message now consists of the challenge-data, the session-id and the cipher-specs. The challenge-data has the same use as in the initial case. The session-id is used by the server to look up the cipher kind and the master key. The cipher-specs are sent again in case the server has timed out the session-id by the time the message reaches it. If this is the case the server can continue with the previous type of message exchange, as if no session-id were used.

The **server-hello** message, which is returned if the session-id was found in the server's cache, contains the connection-id and the session-id-hit flag.

The rest of the message exchange is the same as in the previous case except that there is no client-master-key message.

Session Identifier Used, Client Authentication

When client authentication is requested, the message exchange differs from the previous example after the server-verify message is sent.

Message Type	Direction	Data Transfered
client-hello	C > S	challenge-data, session-id, cipher-specs
server-hello	C < S	connection-id, session-id-hit
client-finish	C > S	{connection-id}client-write-key
server-verify	C < S	{challenge-data}server-write-key
request-certificate	C < S	{auth-type, cert-chal-data}server-write-key
client-certificate	C > S	{cert-type, client-cert, resp-data}client-write-key
server-finish	C < S	{session-id}server-write-key

Fig 11. Table of SSLHP messages: session-id used, client authentication

The **request-certificate** message is sent by the server and contains the authentication-type and the certificate-challenge-data and is encrypted using the server-write-key. The authentication-type states a message digest function and a form of public key encryption. The certificate-challenge-data is created by the server, and is the data which the client uses to create a digital signature about.

The **client-certificate** message returns the certificate-type, the client-certificate and the response-data. The response-data is the signature of the certificate-challenge-data, in that the challenge is hashed then signed using the client's private key. To authenticate the client, the server extracts the public key from the client-certificate, decrypts the response-data and compares the result with its own calculation of the hash of the certificate-challenge-data.

At any stage during these message exchanges, error messages may be sent in response to received messages. These messages are **no-cipher error**, when a server cannot support any of the ciphers in the client cipher list; **no-certificate-error**, when a certificate is requested but one is not available; **bad-certificate-error**, if a certificate is not valid; **unsupported-certificate-type-error**, when a certificate type is not supported by either party. In some cases the errors are recoverable, in others these errors are fatal (see [HIC95] in Appendix A for details).

After the handshaking has been completed both sides then should expect records of application data in any order.

4.4.3 Strengths of The SSL Protocol

Dictionary Attack

This attack works in situations where some plaintext is known to make up part of the original message. In the case of a protocol, the very fact that a rigid structure of data exists is often enough information. Similarly if the application data being transferred has a few often used commands, in the case of HTTP the "get" command for instance. The attack works by taking the known plaintext and precomputing the encrypted form of the plaintext using every possible key. Given an encrypted message, a search is performed looking for occurrences of the precomputed ciphertexts in the message. When one is found it is then known that the key which was used to give that ciphertext is the key which was used to encrypt the whole message.

SSL is protected from this attack by having very large key spaces for all its ciphers. Even the export ciphers support 128 bit keys (albeit 88 bit transmitted in the clear) and this makes the dictionary prohibitively large. For the export ciphers a dictionary could be produced of size 2^{40} entries, but this would have to be done for each session and this in effect makes identical to a brute force attack (discussed later in Section 4.5.2 on Weaknesses of The SSL Protocol).

Brute Force Attack Against Strong Ciphers

See Section 3.2.4 on Key Length for a description of the brute force attack and times required for successful attacks on different key lengths. A brute force attack against the ciphers with 128 bits or more is completely impractical in the foreseeable future. The only non-export cipher which would be susceptible to this attack is the standard DES 56 bit cipher - it is not recommended that this cipher be used if at all possible.

Replay Attack

A replay attack is one where a third party records an exchange of messages between a client and server and attempts to rerun the client messages at the server at a later date. SSL foils this possibility by introducing a nonce number, this is in the form of the connection-id which the server randomly generates and sends to the client. Since the nonce differs for each connection, no two connections are likely to have the same nonce and thus the old set of client messages do not satisfy the server. SSL nonces are 128 bits in length for added security.

Man-In-The-Middle Attack

The Man-In-The-Middle attack (described earlier in the Section 3.6 on Certificates) occurs when an adversary is able to intercept client messages read them and pass them on to the server and vice versa. This attack is prevented by SSL forcing the server to use its private key to decrypt the master key if it is to continue to handshake with the client. To be able to do this requires that it has a valid certificate which the client can verify. An adversary would have to fake a certificate and break the certification authorities key in order to compromise this chain; a task which is more difficult than doing a brute force attack on the cipher itself.

4.5 Protocol Analysis Summary

As part of this project the handshake protocol has been extensively analysed in [BRD95a], which is reprinted in Appendix B.

4.5.1 Omissions from The SSL Specification

Certification Management

The protocol states that certification systems are out of the scope of the document. However any security related protocol has to deal fully with all issues of negotiation and authentication. There are several certification standards already available, notably the PEM hierarchical system and the PGP web-of-trust system. It is recommended that a hybrid system be used, to cater for all communications requirement.

Error Messages

Two error messages have been omitted from the specification document. The first, an unsupported-authentication-type-error message, is a mistake which would prevent the protocol using different methods of authentication of a client.

The second is not essential but would let an implementation cater for all eventualities. An unexpected-message-error would allow an implementation to close the connection cleanly if an implementation sent an out-of-order message.

4.5.2 Weaknesses of The SSL Protocol

Brute Force Attack Against Weak Ciphers

The most obvious weakness of the protocol is the susceptibility of the ciphers which use small keys to brute force attack, in particular RC4-40, RC2-40 and to some extent DES-56. See Section 3.2.4 on Key Length for time and power requirements to break these ciphers. As has been discussed, this was a weakness which was forced upon Netscape by US export law. However it remains a serious flaw and has been exploited several times already. To the best of my knowledge successful brute force attacks against RC4-40 have been carried out at least five times since the protocol's release in February 1995.

Renegotiation of Session Keys

In the present situation, once a connection is established, the same session key (master key) is used throughout. If SSL were layered underneath a long running connection, such as a Telnet application, then this failure to alter session keys becomes a serious security hole. One of the best methods of increasing the security of a system such as this is to force renegotiation of the session key at regular intervals, thus multiplying the difficulty and cost of a brute force attack by the number times the session key is changed. This could actually be used to increase the security of the export ciphers. If an adversary were faced with the task of having to decrypt

100,000 records all encrypted using a different, albeit weak, key rather than the same key, then the task becomes 100,000 times more difficult.

4.6 Current Implementations of SSL

4.6.1 The Netscape Navigator

Export

In order to accommodate the export regulations, the Netscape browser comes in several different versions. The most widely available is the exportable international version which has no strong encryption at all. The versions which contains the fully enabled cryptography are only available in the US and cost between $40 and $60

Certificates

The other major problem with SSL as implemented within the browser is the certification issue. As there is no support for certification hierarchies in the protocol, a server or client certificate can only be verified by a certificate which resides in the browser. Netscape has hard-coded the VeriSign (RSADSI) certificate into the current version of the browser and does not allow any user certificate management, that is adding further certificates.

VeriSign is an off-shoot of RSADSI, which sells certificates to people who are prepared to provide full identification and conform with their various and complicated criteria. These certificates are not inexpensive, VeriSign charges $290 for the first certificate, $95 for subsequent certificates and a renewal charge of $75 per year, for this service.

The disadvantage with this system is that if you want to provide a secure service, you currently have no option other than to buy a certificate. Even if you reimplement the server using one of the SSL implementations below, there is no obvious way of getting the huge user base of Netscape browsers to communicate with your server, since they will not recognise any other certificate. Netscape have stated that they propose to allow people to add certificates of their choice in future versions.

Possible Man-In-The-Middle Attack

Another difficulty which has arisen from the current certificate situation was the possibility of a Man-In-The-Middle attack, as discovered by an independent cryptography consultant Donald Davis. The scenario is that an adversary tampers with an implementation of the Netscape browser, currently sitting in many FTP sites around the world, and replaces the hard coded certificate with the adversary's own certificate. When someone comes to use the browser for secure communications, the adversary can intercept the client request and pretend to be the real server, or passively read the communication and then pass it on to the intended server. Whichever way the client browser will always accept his public-key because the browser has the altered certificate.

This danger would be significantly averted by proper certificate management systems, which would allow the user to check certificates and replace them regularly if necessary.

4.6.2 SSLref

Export

SSLref is Netscape's SSL reference implementation with which it will allow US citizens to use to design SSL compliant non-commercial applications. Rather surprisingly and despite Netscape's best efforts, SSLref is itself non-exportable despite the fact that it does not contain any cryptographic material at all. The very fact that it contains references to libraries which do contain powerful encryption is itself enough, it seems, to have an export ban placed on it.

Design Flaw

Recently doubt has been cast on the design of SSLref. There are several instances in the protocol where it specifies that some important data items, like the symmetric session key, should be constructed using random numbers. For a computer this means pseudo-random although some sequences are considered more secure because of their unpredictable nature, if seeded properly. Such sequences are called cryptographically secure random number generators. The problem with SSLref is that it does not seed its random number generator with enough entropy (unpredictability). As the code below shows, on UNIX versions, it is seeded with the time, the process ID number and the parent process ID number, which are then all hashed through the MD5 algorithm to distribute the entropy more evenly; a standard technique.

```
/*
** Create a new random number generator that is "randomly" seeded.
** Few systems have good ways to do this so we try to get some
** external information besides just the time of day to use.
*/
RNGContext *RNG_CreateContext(void)
{
   /* Misc. declarations deleted */
   cx = (RNGContext*) calloc(1, sizeof(RNGContext));
   /* Get data for seeding the generator */
   gettimeofday(&now, &tz);
   pid = getpid();
   ppid = getppid();
   stat("/bin/ls", &ls);
   stat("/dev", &dev);
   stat("/tmp", &tmp);

   /* Create seed by hashing random numbers through a digest function */
   R_DigestUpdate(&digest, (unsigned char*) &now, sizeof(now));
   R_DigestUpdate(&digest, (unsigned char*) &pid, sizeof(pid));
   R_DigestUpdate(&digest, (unsigned char*) &ppid, sizeof(ppid));
   R_DigestUpdate(&digest, (unsigned char*) &ls, sizeof(ls));
   R_DigestUpdate(&digest, (unsigned char*) &dev, sizeof(dev));
   R_DigestUpdate(&digest, (unsigned char*) &tmp, sizeof(tmp));
   R_DigestFinal(&digest, cx->state, &statelen);
```

```
    return cx;
}
```

A lot of the variables that go into the hashing algorithm are very predictable and often take the same values (for instance, 'ppid' which is often 1). Netscape admitted that they have used only 30 bits of entropy to generate a 128 bit "random" number. What is worse is that the random number generator was only seeded once at the start of operations, so once the initial seed is discovered every random number generated can be predicted.

Although this just sounds academic, two students from UC Berkeley, David Wagner and Ian Goldberg were able to exploit this serious weakness. They wrote a program to exhaustively search the unknown bits in question, in much the same way as a brute force attack is done on a key space. Reputedly, the program can find the symmetric session key within one minute.

It should be noted that this was a flaw in the design of the implementation, not the protocol itself. Netscape are meant to be fixing the problem in the next month, however they have since stated that they will not in future publish how the random numbers are generated. This seems a strange reaction because, if one is to learn anything from these exploits, it is that exposing a protocol and implementation to the wider academic gaze serves to improve the security of the product, not diminish it. It is surely better to have an error like this publicised and fixed than to let it go unchecked and let an adversary use it against you later on. Maybe this is an instance where having an important standard in the hands of a company is not an advantage. A company, especially one with new share holders, will instinctively see such publicity as bad and try to prevent such revelations happening again. This is not in the best interests of the security of the product.

4.6.3 SSLeay

SSLeay was developed by a well known australian cryptographer, Eric Young. SSLeay was written in two months and has been through several revisions since then, but is not a reference implementation. It is freely available and exportable to anyone for any purpose, including commercial. It supports all the symmetric ciphers except RC2, which is not publicly available.

4.7 The Future of SSL

Since the original development of SSL, further suggestions for improvements have been made. SSL version 3 [ELG95] was released as an Internet-Draft in June 1995 and was subsequently discussed at the July meeting of the IETF (Internet Engineering Task Force) in Stockholm.

The new version is completely backwards compatible with SSL version 2, but also supports:

1. Alternative key-exchange algorithms, including Diffie-Hellman
2. Hardware "tokens", which implement some unspecified key exchange and bulk encryption
3. SHA, DSS and Fortezza

4. Out-Of-Band (OOB) data transfer

Rather controversially this includes support for Clipper-like key escrow systems. However an implementation could still be compliant with the protocol and leave such support out, so it is probable that this feature is only present for use within US government agencies.

The OOB data transfer is a possible solution of the certification problem. An entire certification chain could be sent using another data channel (again unspecified), so making it more difficult to introduce forgeries.

5. The SSLP Reference Project

5.1 Project Brief

> "Design and implement a reference implementation of the Secure Sockets Layer protocol. The design should be fully software engineered using formal techniques and should be the result of a thorough analysis of the protocol specification document."

The resulting implementation is called SSLPref (SSL Protocol Reference Implementation), to distinguish it from Netscape's implementation SSLref.

5.2 Motivation

Of the SSL implementations that have been released so far, SSLref suffers from export restrictions and some severe design flaws and SSLeay does not purport to be a reference implementation. So the requirements for SSLPref are that:

1. it is not restricted by export law
2. it is not inhibited by international patents
3. it does not suffer from the design flaws of SSLref
4. it has been through formal software analysis and design procedures
5. it will allow full certification management features

The main immediate beneficiaries of a good SSL implementation which conforms to the above requirements will be:

1. individuals who do not wish to pay for fully protected communications
2. educational institutions and individuals who cannot afford and do not need expensive certification procedures
3. companies who depend on strong security to maintain their public reputation for trustworthy transactions
4. companies who manufacture software and need to see demonstrable surety of design before they can incorporate third party software into their products
5. standards organizations who cannot endorse a reference implementation which is not freely distributable

Future developments which will be made possible will include:

1. client software capable of strong encryption which can support full certificate management features
2. secure server software, which does not force the user to buy a certificate
3. a fully functional secure proxy server capable of handling secure communications for many clients and many servers.

5.3 Integration Into Current Software

In order for the software to be used to its full potential, it must integrate with software which is currently in widespread use. Currently the most popular client web software is the Netscape Navigator, so any new client facilities should integrate with that. Similarly with servers and server software extensions.

This is where the idea of a protocol converter stems from. A useful application will be a piece of software which can translate between clear communications protocols and the SSL protocol and back again. The first two suggested applications provide this service for individual clients and servers, whereas the third does this for many clients and servers simultaneously.

SSLPref will be at the core of these applications; its job in effect is to do the protocol translation. So one of the design criteria for SSLPref is that it has to be flexible enough to be able to scale between the three applications without any code alteration. The only remaining question is how can the applications once written integrate into the current available software.

5.3.1 The Netscape Navigator

We require a facility in client web browsers which by-passes the in-built SSL implementation and redirects the browser to a proxy address which will handle the protocol conversion. Netscape browsers provide this facility in their Preferences dialogue box with the use of the Security Proxy dialogue box. So the client protocol converter must have a TCP/IP address and act as a proxy server for the one client, in this case the browser. This client proxy can run on the same machine for speed purposes, there is no need for it to be distributed across a network. This scenario solves three major problems with the current SSL situation:

1. The use of export-grade encryption can be eliminated, by providing the protocol converter with the necessary strong encryption tools.
2. The server certification problem is solved at the client end by by-passing the hard-coded RSADSI/VeriSign certificate. Thus leaving the decision of which certificates should be trusted up to the user and allowing the creation of free secure servers.
3. Netscape can continue to distribute their weak export version without infringing US export law and still know that people abroad will be able to use the full potential of the SSL protocol.

5.4 The Software Engineering Process

5.4.1 Planning SSLPref

The SSLPref project was subject to a time management methodology in the form of a preliminary task analysis of the project and subsequent use of Gantt Charts. Such a system allows the software developer to measure how well the original project timetable is being adhered to throughout the lifecycle of the project and to facilitate reallocation of tasks should slippages occur.

Task Analysis

0	Learn Specification Language Z		
0.1		Obtain relevant books	
0.2		Obtain PS/TT/RO Z Fonts	
0.3		Learn Language	
1	SSL Reference Module D & I		
1.1		Encryption Algorithms D & I	
1.1.1			Obtain Documentation (UR)
1.1.2			Specify in Z
1.1.3			Animate Specification in Haskell
1.1.4			Implement in C/C++
1.2		Client/Server Communication Protocol D & I	
1.2.1			Obtain Documentation for SSL and TCP (UR)
1.2.2			Analyse formally using Process Algebra
1.2.3			Prototype in Parlog
1.2.4			Clarify design issues with author
1.2.5			Final Specification in Z
1.2.6			Implement in C/C++
2	HTTP Client/Server Interface D & I		
2.1		Obtain HTTP Documentation (UR)	
2.2		Specify in Z	
2.3		Implement in C/C++	
3	Project Documentation		
3.1		Organize Documentation/References/Books	
3.1.1			Learn LaTeX
3.1.2			Acquire necessary tools to typeset Z
3.2		Project Plan	
3.3		Design Overview	
3.4		Maintenance Guide	
3.5		Technical User Manual	
3.6		Test Plan	
3.7		Test Results	
3.8		Appendices	
3.8.1			Detailed Bibliography of References
3.8.2			Inclusion of external Protocol/Algorithm Specifications
3.8.3			Z specifications to accompany main algorithms
3.8.4			Haskell animations of 3.8.3
3.8.5			Process Algebra analysis of 1.2
3.8.6			Parlog prototype of 1.2
3.8.7			C/C++ code for system

Time Management Chart

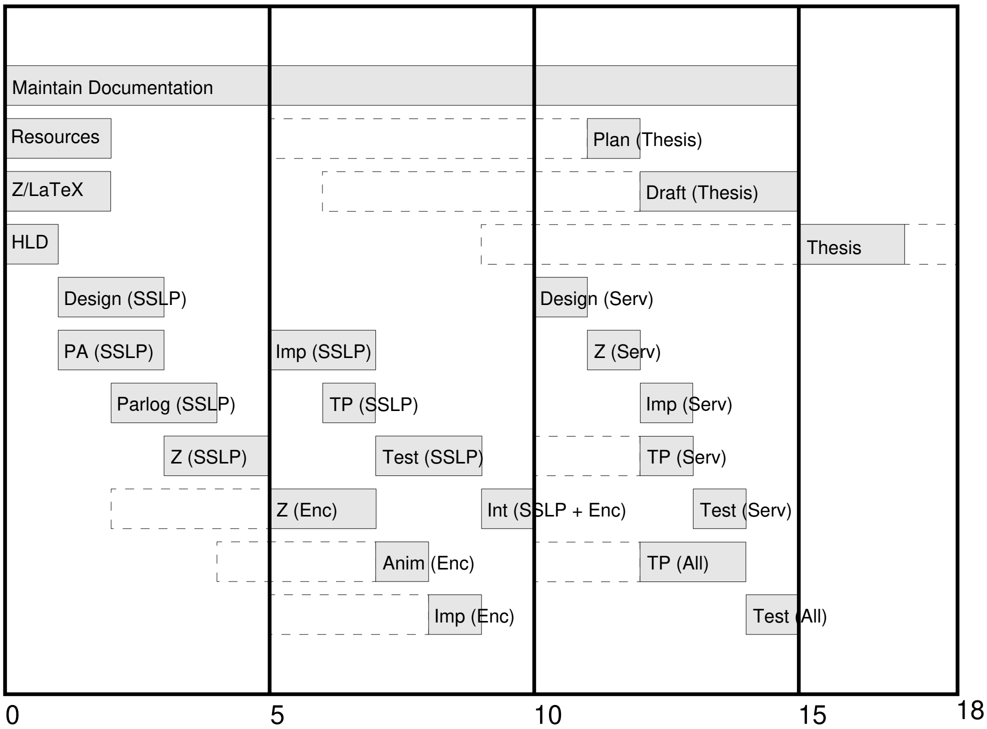

Fig 12. Diagram of Gantt chart used to map project tasks

Four milestones were established for the project lifecycle. It was decided that the design and implementation of server software between milestones 2 and 3 would be attempted only if time allowed and SSLPref had been fully completed. Similarly time was allocated for the design and implementation of further encryption algorithms that were specified but not implemented in the public domain. In the event that all ciphers were obtainable or alternatively the ciphers that were not available had not been published, then the time could be reallocated. The latter proved to be the case over the RC2 cipher.

5.4.2 From Analysis To Design

The first part of the project instituted an analysis of the protocol which culminated in the publication of a Technical Report [BRD95a] entitled “The Analysis of the SSL Protocol” (reprinted in Appendix B). By far the larger part of the report dealt with analysing the SSL Handshake Protocol specifically and the rest suggested security recommendations covering the protocol as a whole.

The analysis of the SSLHP took the form of examining the message dependencies and working out what constraints were placed on the message ordering within the protocol as a result. This was achieved through the use of message dependency diagrams. These were then converted into definitive generic finite state diagrams which could be used to model any compliant SSL implementation, not just SSLPref. It was a simple conversion process to produce concurrent process algebra from the diagrams.

The diagrams could then be analysed by Concurrency Programming Toolkits to check for protocol deadlock problems. It was even possible to implement this process algebra in a parallel language, Parlog to check that correct sequences of messages were generated by both sides.

5.4.3 Formal Methods Specification Using Z

A finite state machine was required to run through the finite state diagrams generated in the analysis. Now it was established that a finite state machine would represent the heart of SSLPref implementation, it was important to ensure that this critical subsection of the software was completely error free. This was achieved by specifying an FSM in the Z formal mathematical specification language [POT91]. The FSM Z specification is reprinted in Appendix D.

Similarly, design rigour was required when specifying the individual message and data components so Z data types were defined, taking into account all the constraints and requirements of the SSL protocol. The Z specification of the SSL data types is reprinted in Appendix E.

5.4.4 Technical Description of Programming Interface Design

The main data structure, that of the programming interface, was specified in ASN.1 (Abstract Syntax Notation 1) [HAL93]. This is reprinted in Appendix F.

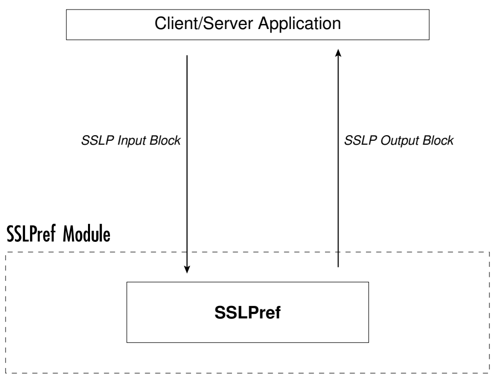

Fig 13. Diagram of SSLP interface

The basic intention of the programming interface was to make it as conceptually simple as possible. With this in mind the SSLPref module has a one function interface, which takes a SSLP Input Block as input and returns a SSLP Output Block. There are no other external operational function calls available as this system has been designed to handle every eventuality: certificate lists, cipher preferences, user certificates, as well as simple protocol conversion. Other non-operational function calls are provided in order to create and fill these interface data structures.

It is up to the application layer which uses SSLPref, to provide an interface to the user to allow the setting of such preferences and certification options. If SSLPref itself provided such an interface it would break the ISO layer abstraction rules which were discussed in Section 4.3.2 on SSL and The ISO Reference Model. This does not mean that every application has to use all the options, a simple client application might only use the most basic of the certification systems, similarly for the most basic server and indeed that is the situation with SSL-based applications at the moment. However for the full potential of SSL to be used, these options are provided.

An important point to note about the interface is that it has been designed to be completely scalable to fit any situation. So, as required by our initial application requirements, it can handle a one client to one server scenario or one client to many servers, or many clients to many servers.

Input Block

The Input Block consists of the following:

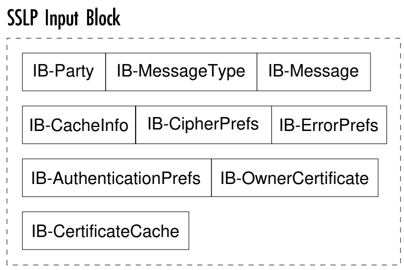

Fig 14. Diagram of SSLP input block data structure

The **IB-Party** flag can take one of two values, “client” or “server”, which instructs the module whether the incoming data is coming from the client or the server.

IB-MessageType takes one of three values, “init”, “data” or “close”. The init type is only set at the very beginning of every transaction. Together with the IB-Party flag, it instructs the SSLPref module as to which party, client or server, it is converting for. Technically, it allows the module to select the correct finite state diagram, which tells the module which messages to send and which messages to expect. The data flag is used for every other message that is sent to the module except the last one when the close flag is set. The close flag indicates that one of the parties has closed the connection. It allows the module to reset its finite state machine.

IB-Message consists of the actual message, either a SSL Record or a stream of clear data. Which type the module expects depends on how it was initialised and what IB-Party flag has been set. If the module is initialised for client then it will expect clear data from the client and SSL Records from the server, on the other hand if it is initialised for server then it will expect clear data from the server and SSL records from the client.

IB-CacheInfo represents either the client or server cache of previously used sessions or the session information for the current connection. The session information is described in greater detail in the Output Block section.

IB-CipherPrefs is the user's list of preferred ciphers. If a user does not wish to use a particular cipher at all, as might be the case with the weaker RC4-40, RC2-40 and DES ciphers, then they can be left out of the list altogether. On the other hand if for compatibility purposes, the weaker ciphers are required only as a last resort, then they can be placed at the end of the list. The recommended [SCH94] and default ordered list of ciphers, taking into account how much peer review and cryptanalysis has taken place on them, is:

IDEA, Triple DES, RC4-128, RC2-128, DES, RC4-40, RC2-40

IB-ErrorPrefs allow a server to decide which client errors should be fatal and which can allow the connection to continue. The SSL protocol specification allows the actions of some errors to be undefined. To some extent this is a security decision, it allows a server to restrict the information to a certain set of people who have certificates signed by a recognised certification authority. Because this is implemented on a connection-by-connection basis, it means that particular web pages might be restricted, for instance (if the application protocol using SSL is HTTP). The restriction does not have to be for all the information on a particular site. Similarly the domain of restriction depends on the certification system used within an organization. The domain can be restricted to one person, a small group of people, a company or a country.

IB-AuthenticationPrefs is used in conjunction with the IB-ErrorPrefs and IB-CertificateCache. It is server specific and defines whether the server requires the client to authenticate itself for a particular connection. Again this can be changed on a connection-by-connection basis, depending on the information that has been requested.

IB-OwnerCertificate is the certificate of the owner party. If the client initialized the module then it is the client certificate, if the client has one. If the server initialized the module then it is the server certificate.

IB-CertificateCache is the cache of recognised signatory certificates. This can be used in conjunction with IB-ErrorPrefs and IB-AuthenticationPrefs, to restrict client access to a subset of information as discussed earlier. Alternatively this can be used by the client to build a trusted hierarchy or commercial and personal certificate signatories.

SSLP Output Block

The Output Block consists of the following:

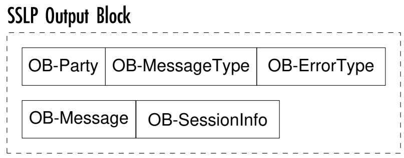

Fig 15. Diagram of SSLP output block data structure

OB-Party flag denotes who the result of the function should be sent to. It can take one of three values: “client”, “server” or “neither”. The host then knows to which party to send the data which the block contains. If it takes the “neither” value then the module is waiting for another response from the opposite party.

OB-MessageType can take one of several values: “ready”, “busy”, “fatalerror”, “nonfatalerror” or “close”. It takes the ready value when the handshake protocol has finished and indicates to the client that the module is ready to receive the application layer’s data in order to pass it on to the server. Conversely, the busy value indicates that the handshake protocol is still underway and that the module is not ready to take application data yet. The error values indicate a valid error; fatalerror means that the connection has to be closed; nonfatalerror means that the module can continue. The close message indicates that the connection should be terminated, in a condition other than that of an error state.

OB-ErrorType is only used if the OB-MessageType has returned one of the error conditions. The error type describes more fully what happened to cause the error, for example, no-cipher-error, if the parties do not share a common cipher, or bad-certificate-error if a false certificate was returned. As with the Input Block implementation it is up to the application layer to report these errors to the user, since an SSL module would break the ISO layer abstraction if it reported them directly.

OB-Message may consist of either an ordered list of SSL Records or a stream of clear data. The reason more than one SSL Record may be returned is that the protocol occasionally allows for more than one consecutive message to be passed from client to server or server to client in the SSLHP phase. See Analysis of The SSL Protocol in Appendix B. Similarly to the IB-Message, the clear data will be being sent to the host party and the SSL Records will be being sent to the remote party, depending on which party initialised the module and which OB-Party is declared at the beginning of the Output Block.

OB-SessionInfo returns the current state of the session, including: the session-id, used by the module to identify a unique connection; a time stamp which details when the session information was last updated; the cipher type; the session master key; and the finite state machine, relating to that session. When the OB-SessionInfo is returned by the module, it must be updated in the session cache which the application layer maintains. This allows the time stamp to be updated so that the information can be expired after a certain amount of time, if the connection is not reused.

5.4.5 Cryptographically Secure Algorithms

Encryption Routines
RSAref2.0, rc4 and idea are all implementations of ciphers which have been used in other products for some time. Of the three RSAref2.0 produced by RSADSI has been through the most rigorous user testing. Idea has been used for 2 years in PGP version 2 and rc4 is the implementation described in Section 3.2.1 on Common Symmetric Ciphers.

Pseudo-Random Number Generator Seeding
Given the problems that SSLref has encountered with seeding its pseudo-random number generator, this point is addressed specifically in this report. In accordance with the recommendations laid down by [ECS94] and US Department of Defense recommendations, an estimated 187 bits of entropy was obtained from the following sources:

1. extended sampling of the microsecond counter and other system counters
2. system interrupt registers
3. system status registers

This was de-skewed or unbiased by running the entropic data through the MD5 message digest algorithm to produce a 128 bit number. This number was then used to seed the cryptographically secure pseudo-random number generator, provided by RSAref2.0.

5.5 Possible Future Extensions

5.5.1 Changes to SSL Protocol

Given that SSL is likely to develop beyond version 2 which it is at present, one of the design criteria to which SSLPref was written was the ability to change with the protocol if necessary. This was largely made possible because the protocol document pointed out areas where additions or changes were likely to occur in future. For instance, the extension of the protocol to include support for more ciphers and other methods of key exchange, other than RSA.

Similarly if the SSLHP has to change to accommodate further changes, then SSLPref can be altered by manipulating the finite state diagrams in the appropriate way. The underlying finite state machine will remain completely unchanged whatever happens.

Another point which displays SSLPref's beneficial design features is that, even if a version 3 of the protocol is brought out, SSLPref will be able to run both version 2 and version 3 simultaneously through the same interface design. So for a proxy server application providing SSL conversion facilities for clients and servers, it would be possible for the same proxy server to provide SSL version 2 support for one client and SSL version 3 for another at the same time. The major benefit of this will be that once an application supports SSLPref, it will automatically be able to take advantage of SSL version 3, without modification to the application layer software.

5.5.2 Support for PGP Public Keys

Given SSL's flexibility towards different certification systems, it may well be possible to incorporate support for PGP public key certificates into SSLPref. This would tie in with the original project policy of trying to integrate and incorporate as many standards as possible, so as to increase user acceptance. In this case the existence of a large pre-certificated user population and the "Web of Trust" would present a huge advantage for any product wishing to use certificates.

5.6 UK Laws on Export of Cryptography

Although the UK does not institute punitive export laws on cryptographic software, it does have some cryptography-related export controls. The excerpts below are taken from [CUS89]. The first excerpt lists the relevant categories of export control:

> [Page 2]
>
> Prohibitions and restrictions on exportation
>
> 2. Subject to the provisions of this Order-
>
> (i) where scheduled goods are indicated by the letter "C", those goods are prohibited to be exported to any destination, and (whether or not express provision is made in relation to technology) technological documents the information in which relates to those goods are prohibited to be exported to any destination in a country listed in Schedule 2 to this Order;
>
> [Page 3] ...
>
> (viii) scheduled goods indicated by the letter "W" are prohibited to be exported to any destination;

The "Schedule 2" mentioned in Article 2.i is listed below. These are the countries to which export restrictions apply, if a Schedule is marked with the letter "C". Note that it is actually out of date, in that a number of these countries do not exist any more.

[Page 165]

SCHEDULE 2

List of countries referred to in Article 2:

Afghanistan
Albania
Bulgaria
China
Czechoslovakia
German Democratic Republic
Hungary
North Korea
People's Republic of Mongolia
Poland
Romania
Socialist Republic of Vietnam
Union of Socialist Soviet Republics

Now follows the law which is relevant to cryptographic material. Note that this Schedule is marked with restriction "C" and has exceptions.

[Page 77]

IL1527 Cryptographic equipment designed to ensure secrecy of communications (such as telegraphy, telephony, facsimile, video, and data communications) or of stored information; and specially designed components therefor, and software controlling or computers performing the functions of such cryptographic equipment C

except simple cryptographic devices or equipment ensuring only the privacy of communications, the following-

(a) equipment for voice transmission making use of fixed frequency inversions of fixed band scrambling techniques in which the transposition changes occur not more frequently than once every 10 seconds;

(b) standard civil facsimile and video equipment designed to ensure the privacy of communications by an analogue transmission using nonstandard practices for intended receivers only (video system equipment effecting the transposition of analogue data);

> (c) video systems for pay television and similar restricted audience television, including industrial and commercial television equipment using other than standard commercial sweep systems.
>
> Note 1. This entry includes video systems which, for secrecy purposes, use digital techniques (conversion of an analogue, ie video or facsimile signal into a digital signal).
>
> Note 2. Digital computers and digital differential analysers (incremental computers) designed or modified for, or combined with, any cypher machines, cryptographic equipment devices or techniques including software, microprogramme control (firmware).

Having quoted all that, it is still not clear whether something like SSLPref comes under the "C" restriction or is unrestricted. It is clear however that is does not come under the "W" restriction, so it can be exported to all of Western Europe and North America at the very least.

The uncertainty over the exact restriction category ("C" or none) stems from a passage in Schedule IL1527 which states exceptions from restriction "C":

> "...except simple cryptographic devices or equipment ensuring only the privacy of communications..."

SSLPref is certainly not complicated in that it does not use any new cryptographic techniques, and its purpose is solely for "privacy of communications", so it is possible that SSLPref does not even fall under restriction "C".

5.6.1 The Legal Publication of SSLPref

Until the legal position is clarified as to whether SSLPref falls under restriction "C" or not, any attempt to distribute or broadcast the implementation globally would be unwise. However publishing the implementation in the UK would not infringe any law.

In Internet terms, "distribute" or "broadcast", would include posting to a newsgroup with global distribution or e-mailing to people in other countries. "Publishing in the UK" would mean placing the implementation on a World Wide Web server which is situated in the UK. People from outside the country who attempt to export the implementation will be asked to read the above passages from British law and must make the decision as to whether they would be committing an offence under the act by exporting the implementation, given the country in which they are situated.

Bibliography

[BAM83] James Bamford
"The Puzzle Palace",
Penguin, 1983

[BAR92] John Perry Barlow
"Decrypting The Puzzle Palace",
Communications of the ACM, June 1992 (original),
EFFector Online Issue 3.1, 29 July 1992 (reprinted)

[BRD95a] Jeremy Bradley and Neil Davies
"Technical Report: Analysis of The SSL Protocol",
Communications Research Group,
Department of Computer Science,
University of Bristol, July 1995

[BRD95b] Jeremy Bradley
"Cryptography: Brute Force Attacks and Continual Technological Revolution",
Communications Research Group,
Department of Computer Science,
University of Bristol, September 1995

[CUS89] Customs and Excise
"The Export of Goods (Control) Order 1989",
HMSO, Statutory Intruments 1989 No. 2376, 15 December 1989

[DEN82] Dorothy Denning
"Cryptography and Data Security",
Addison-Wesley, 1982

[DIF76] Whitfield Diffie and Martin Hellman
"New Directions in Cryptography",
IEEE Transactions on Information Theory,
v. IT-22, n. 6, November 1976, pp. 644-654

[ECS94] Donald Eastlake, Stephen Crocker and Jeffrey Schiller
"Randomness Recommendations for Security",
RFC 1750, Network Working Group, IETF, December 1994

[ELG95] Taher Elgamal and Kipp Hickman
"The SSL Protocol (version 3)",
Netscape Communications Corporation, Internet Draft, June 1995

[EQX95] Equinox
"Cybersecrecy",
Channel 4, 3 September 1995

[GAR95] Simson Garfinkel
"PGP: Pretty Good Privacy",
O'Reilly & Associates, 1995

[GOL95] Steve Gold
"The Hills have Ears",
Wired 1.01, UK edition, Wired Ventures, 1995

[HAL93] Fred Halsall
"Data Communications, Computer Networks and Open Systems", Third Edition,
Addison-Wesley, 1993

[HAW95] Nigel Hawkes
"Barclays Opens First Mall on The Internet",
The Times, 3 June 1995

[HEL95] David Hellier
"Whitehall Snoops Exposed"
Independent On Sunday, 3 September 1995

[HIC95] Kipp Hickman
"The SSL Protocol (version 2)",
Netscape Communications Corportation, 9 February 1995

[HOF94] Lance Hoffman, George Washington University
"SPA Study of Foreign Availability of Cryptography",
SPA News, Vol. 13 No. 3, March 1994, Software Publishers Association

[KAH67] David Kahn
"The Codebreakers: The Story of Secret Writing",
Macmillan, 1967

[KEN93] Steve Kent
"Privacy Enhancement for Internet Electronic Mail,
Part II: Certificate-Based Key Management",
RFC1422, Network Working Group, IETF, February 1993

[LAI91] Xuejia Lai
"Detailed Description and a Software Implementation of the IPES Cipher"
Preprint, 8 November 1991

[LEV93] Stephen Levy
"Crypto Rebels",
Wired 1.2, Wired Ventures, 1993

[NBS77] National Bureau of Standards
"NBS FIPS PUB 46, Data Encryption Standard",
United States Department of Commerce, January 1977

[POT91] Ben Potter, Jane Sinclair and David Till
"An Introduction to Formal Specification and Z",
Prentice Hall, 1991

[QUI95] Joshua Quittner
"Browser Madness",
Time International, pp 38-39, 21 August, 1995

[RES94] Eric Rescorla and Allan Schiffman
"The Secure HyperText Transfer Protocol",
Enterprise Integration Technologies, Internet Draft, December 1994

[RIV93] Ron Rivest
"The Difficulty of Factoring",
The RSA Newsletter, v. 1, n. 1, Autumn 1993

[RSA78] Ron Rivest, Ali Shamir and Leornard Adleman
"A Method for Obtaining Digital Signatures and Public-Key Cryptosystems",
Communciations of the ACM, v. 21, n. 2, February 1978

[SCH94] Bruce Schneier
"Applied Cryptography: Protocols, Algorithms and Source Code in C",
John Wiley & Sons, 1994

[SST94] Senate Subcomittee on Technology and the Law
United States Senate, 3 May 1994

[STA89] United States Department of State
"International Traffic in Arms Regulations (ITAR)",
22 CFR 120-130, Office of Munitions Control, November 1989

[STA92] United States Department of State
"Defense Trade Regulations (DTR)",
22 CFR 120-130, Office of Defense Trade Controls, November 1992

[WAS94] Ken Wasch, SPA Executive Director
"SPA Spearheads Efforts to Soften Controls on Encryption Software",
SPA News, Vol. 13 No. 3, March 1994, Software Publishers Association

Quoted Sources

[BAR92.1]	Admiral Stansfield Turner, CIA Director, 1977-1981
[BAR92.2]	Admiral Bobby Ray Inman, NSA Director, 1977-1981
[BAR92.3]	Professor Ron Rivest, MIT, Co-Inventor of RSA Public Key system
[EQX95.1]	Professor Fred Piper, Mathematician, Royal Holloway College, London University
[EQX95.2]	DTI Spokesperson
[EQX95.3]	Ross Anderson, Senior Cryptography Researcher, Cambridge University
[GAR95.1]	Anti Crime Bill, 1991, S.226, Section 2201
[HAW95.1]	Steve Collins, Technology Development, Barclays" Emerging Markets Unit
[HEL95.1]	Professor Fred Piper, Mathematician, Royal Holloway College, London University
[HEL95.2]	DTI Spokesperson
[HEL95.3]	Ross Anderson, Senior Cryptography Researcher, Cambridge University
[LEV93.1]	James Bamford, Author of "The Puzzle Palace"
[LEV93.2]	Donn Parker, Computer Security Specialist, US Government Advisor
[LEV93.3]	Professor Dorothy Denning, Gorgetown University, US Government Advisor
[LEV93.4]	Tim May, ex-Intel Physicist, Author of "Crypto Anarchist Manifesto"
[LEV93.5]	NSA Spokesperson
[LEV93.6]	John Gilmore, ex-Sun Microsystems, Cofounder of Cygnus Support

[SCH94.1] Whitfield Diffie, Co-Inventor of Public Key cryptography in 1975

[SCH94.2] Martin Hellman, Co-Inventor of Public Key cryptography in 1975

[SST94.1] Vice-Admiral DM McConnell, NSA

Appendices

A. The SSL Protocol

B. Analysis of The SSL Protocol

C. Cryptography: Brute Force Attacks and Continual Technological Revolution

D. Z Specification of a Finite State Machine

E. Z Specification of SSLP Data Types

F. ASN.1 Specification of SSLP Data Types