And now for something completely different: looking ahead to new encryption and secrecy protocols.

Helen Ashman[1,2], Martyn Gilbert[2]
1 Computer Science and IT, University of Nottingham , U.K. *hla@cs.nott.ac.uk*
2 Amino Communications, Cambridge, U.K. *mgilbert@aminocom.com*

Transmitting sensitive data over non-secret channels has always required encryption technologies to ensure that the data arrives without exposure to eavesdroppers. The Internet has made it possible to transmit vast volumes of data more rapidly and cheaply and to a wider audience than ever before. At the same time, strong encryption makes it possible to send data securely, to digitally sign it, to prove it was sent or received, and to guarantee its integrity.

The Internet and encryption make bulk transmission of data a commercially viable proposition. However, there are implementation challenges to solve before commercial bulk transmission becomes mainstream. Powerful have a performance cost, and may affect quality of service. Without encryption, intercepted data may be illicitly duplicated and re-sold, or its commercial value diminished because its secrecy is lost. Performance degradation and potential for commercial loss discourage the bulk transmission of data over the Internet in any commercial application.

This paper outlines technical solutions to these problems. We develop new technologies and combine existing ones in new and powerful ways to minimise commercial loss without compromising performance or inflating overheads.

1 INTRODUCTION

Bulk transmission of data in a commercially secure yet reasonably fast way requires encryption technology that is both secure and fast. However more powerful encryption algorithms and longer secret keys are slower computationally, while faster encryption is inevitably less secure. This is a natural consequence of the nature of encryption – the speed of encryption is proportional to the speed of breaking the code by doing a brute-force search. This leaves developers in a quandary when choosing encryption technologies, needing to balance quality of service with security.

In this paper we look at two solutions to this performance versus security problem. The solutions do not propose new encryption algorithms, but rather they change the way we use them. The two new technologies are hierarchical encryption, discussed in section 2, and data fragmentation combined with signal diversity, discussed in section 3. Section 4 concludes with an outline of further work in support of these solutions. Video transmission is used as an example application, but the same technologies can be used in many other applications. Whenever speed and secrecy of transmission are required at the same time, these technologies can help make a commercially-viable application.

2 HIERARCHICAL ENCRYPTION

The stronger the encryption, the harder it is to break but also the more computationally expensive it is. A hierarchical approach to key exchange means that simple and relatively weak encryption and keys are used to encrypt small chunks of data, for example 10 seconds of video. Each chunk has its own key. New keys for this bottom-level encryption are exchanged using a slightly stronger encryption, e.g. whole-video keys could govern the exchange of the 10-second chunk keys. At a higher level, there could be weekly keys, securing the exchange of whole-video keys, and at a yet higher

level, a subscriber key could govern the exchange of weekly keys. At higher levels, the encryption is stronger but is used less frequently, so the overall computational cost is low.

A primary observation is that there are two main things governing the strength of the encryption technology used to secure some data. These are:

1) *the value of each encrypted item determines the strength of the encryption algorithm used to secure it*. An eavesdropper could decode a chunk in a reasonable time but the time to decode every key for the whole video makes it not worth the effort. The *value* of the data partly determines the strength of the security required.

2) *the duration of secrecy required dictates the lowest cost of breaking the encryption algorithm*. Military or political data whose secrecy must last for many years require strong encryption algorithms with long keys. Conversely, a stock market feed whose data retains commercial value only for a few hours, needs only to withstand cryptanalysis until the value of the data has expired. Complexity theory can estimate how long it will take to break a given encryption with a key of a given size, given current algorithms and processor speeds.

Any security solution should be either to expensive or should deliver the decrypted data too slowly for the eavesdropper to use. So it is necessary to choose encryption algorithms and key lengths suitable to the material being ciphered.

In a video stream, the entire video will have a larger value than small sections of the video. An eavesdropper will put in more effort to break a cipher that hides an entire video than merely a few parts of it, as the value of the video may be the resale value of pirated copies. Similarly, an eavesdropper would put more effort again into breaking a cipher that governed the issue of all videos to a given subscriber, and more yet again to breaking into the security surrounding the database of all videos or of all subscribers.

Hierarchical encryption [Gilbert 2000] applies the principle of cryptographic strength appropriate to the value and duration of secrecy of the data. It breaks data into small chunks, e.g. breaking a video stream into 10-second excerpts. It encrypts each chunk with a different key. This is the lowest level of encryption, and because the value of one chunk is small, it is where we use the fastest and weakest level encryption.

At this lowest level, the encryption is applied to every chunk of data. It is also the **only** encryption that is applied to the actual transmitted data (e.g. the video). However, there are a substantial number of keys that an eavesdropper would require in order to decipher the entire data file. For example a 90-minute video broken into ten-second chunks would be encrypted with 540 chunk keys. If a single chunk key was compromised, then all that would be exposed is a single ten-second excerpt of video.

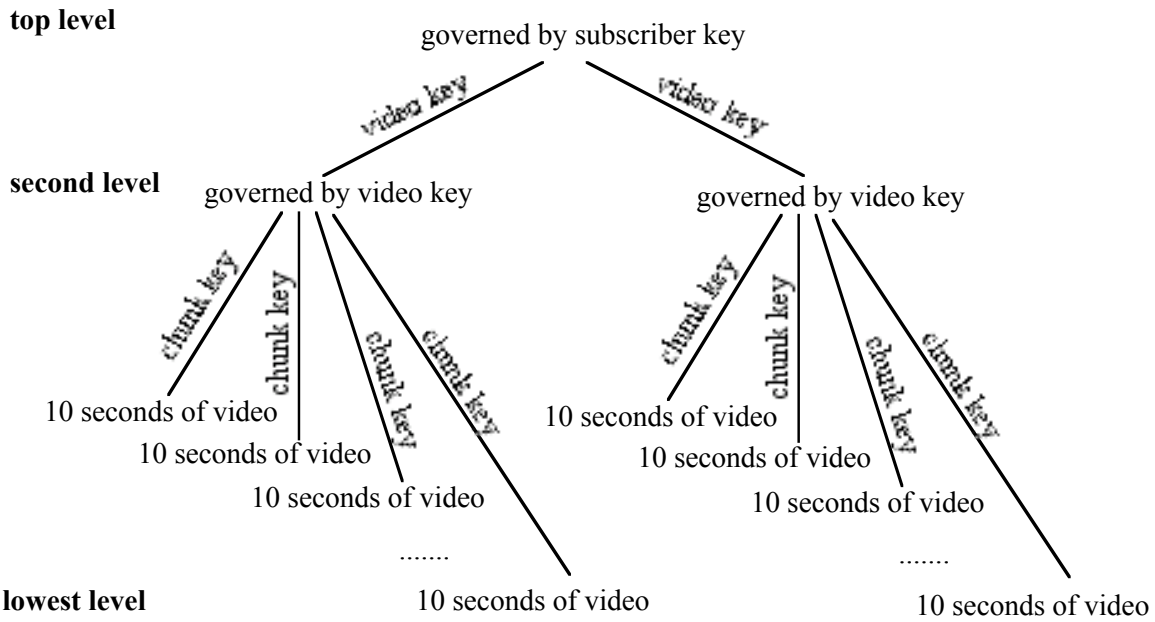
The distribution of chunk keys between sending and receiving hosts is managed by a second level of encryption. The value of the data encrypted at this level is higher, as it secures all chunk keys. Because of the higher value of the data, a stronger encryption is used, but the performance penalty is minimised because the encryption and decryption are performed less frequently. Only the chunk key is encrypted and only when it changes,

for example every ten seconds. If a second-level key was compromised, all chunk keys would be compromised until the second-level key itself was changed. Inn our video example, the second level key could be changed on a per-video basis, with a new "video" key for every video requested by the subscriber. Compromising one video key would mean that the chunk keys for the video would be easily recovered by an eavesdropper, but that the video key would become worthless at the completion of viewing the video.

The third-level key governs the distribution of second-level keys. Distribution of second-level keys will depend on the application, for example, the number of video keys would be the same as the number of different videos requested by a given subscriber. In our example this would be the "subscriber key" that secures all the subscriber's video keys. If a single third-level key was compromised, all videos requested by the subscriber would be compromised, hence the need for stronger encryption at this level.

There is inevitably a top level of encryption, which may be the third level. It governs all primary transactions, such as requesting and paying for individual videos, changing account details, and so on. It should be a form of public-key encryption as this allows the subscriber to identify themselves securely.

The following picture gives an overview of the interactions between the levels of encryption, using the three-level video stream as an example:



The lowest level of encryption, the chunk level, will almost always be performed using a classical (or secret-key) encryption technique, as these are the fastest algorithms for both encryption and decryption. Similarly, the top level of encryption should use a strong public-key encryption with a large key, as the data value at this level is very high.

Intermediate levels of encryption should be chosen to complement the combined strength of the lower levels, as described in the next paragraph. In every case, the encryption chosen should make the effort required to illicitly decipher the communications more than a suspected eavesdropper would be willing to invest for the return.

The effort required for an eavesdropper to decipher all chunk keys should be roughly similar to the effort required to decipher the next level key, and should cost more than the data's potential benefit to the eavesdropper. The same principle applies to second- and higher-level keys – the effort required to discover all second-level keys should be similar to that required for the single higher-level key, and so on for all subsequent levels.

At no stage does the subscriber need to know the individual keys used in this process, except perhaps the top-level key which will change infrequently. The lower level keys are exchanged between the sending and receiving host machines, and the user need not be aware of any encryption taking place at all.

The following table applies three-level encryption to video stream data:

Level	**Unit of encryption**	**Key for encryption**	**Type of encryption**	**Data hidden by key**	**Key life**
Lowest	all video data	chunk key	Classical (secret-key)	10 seconds of video chunk	10 seconds
Second	every chunk key	Video key	Any appropriate	All chunk keys, i.e. entire video	running time of video
Top	every second level key	subscriber key	public-key e.g. RSA	All second-level keys, i.e. all subscriber transactions	lifetime of subscription

The example above is of a three-layer encryption protocol, but there is no reason to restrict the number of levels to three. The actual number of encryption levels depends on the application data and the requirements of the vendors. For example, video transmission could use four levels of encryption, with the second level of keys changing for each entire video as before, then the third level changed weekly, and the top level per subscriber.

Hierarchical encryption has antecedents in the Enigma cipher machines used by the German military during World War II. Each operator had what was effectively a one-time password system, where the day's password for the entire military was listed in a book, and was in turn used to transmit "message keys". While the Enigma cipher was frequently broken by codebreakers at Bletchley Park, this was achieved only with a heroic effort, vast human resources and highly specialised codebreaking equipment

(including the first ever electronic computer), and was motivated by the very valuable data recovered in this way.

The Pretty Good Privacy (PGP) mail encryption tool likewise follows a two-level encryption technology. It is a more modern tool devised in the 1990s to compromise between the speed of secret-key ciphers and the key ownership characteristics of the slower public-key ciphers. A message is encrypted with a secret-key cipher, then the key to the cipher is itself encrypted with the receiver's public key and appended to the message which is then transmitted. Hierarchical encryption generalises and improves on the two-level PGP encryption process, as it intentionally tailors the strength of the encryption at any level to the value and lifespan of the data being encrypted.

In summary, hierarchical encryption makes it possible to securely transmit data with minimum commercial loss and maximum speed. We can apply relatively weak encryption technology to low-value portions of the data, while preserving the overall value of the data by using strong encryption for the more valuable but less time-critical facets.

3 NON-SYMBOLIC FRAGMENTATION WITH SIGNAL DIVERSITY

While encryption technologies offer good protection against eavesdropping, there are other ways to improve security without much performance degradation. If security of transmission must be better than hierarchical encryption alone, then one complementary solution is a combination of *non-symbolic fragmentation* and *signal diversity*.

3.1 Non-symbolic fragmentation

Non-symbolic fragmentation is the breaking up of data into fragments which are usually (but not necessarily) different in size to the size of the base unit of data (the "symbol") of the application. For example, ASCII data is represented in 8-bit symbols and Unicode is represented in 16-bit symbols. Breaking up data in this way ensures that whole symbols are spread across more than one fragment. Fragments are then disordered and/or divided into one or more fragment streams, usually with no two fragments adjacent in the fragment stream if they had been adjacent in the original data.

3.2 Signal diversity

Network and path diversity break up a file or data stream into fragments which are then sent over many different channels, either in the same network or different networks. For example, a message could be transmitted partly over the phone network and partly via satellite. TCP/IP does a similar thing in sending packets over different paths, doing so for load-balancing purposes and is invisible to the end application.

Network and path diversity deliberately introduce the same principle as a secure communications mechanism - an eavesdropper would need to intercept not just one transmission path but all paths . Sub-symbolic fragmentation of data is also introduced to further confuse any intercepted stream of data. This technology can be used with or

without encryption and still remains secure. It is believed that signal diversity could help reduce unauthorised interception by international surveillance systems such as Echelon.

Signal diversity has been used previously in applications which assign different frequencies to transmission parts (a form of path diversity) [Globalstar, 2000]. Here path diversity is performed mostly for purposes of interference-free signal reception, and for avoiding or minimising network downtime, rather than security reasons. Globalstar [2000] give the following definition:

> Path Diversity is a patented method of signal reception that permits the combining of multiple signals of varying power strengths into a single coherent signal.

Network diversity exists in practices such as Amazon's online credit card purchasing, which allows users to email a part of their credit card number and to telephone through the rest. Path diversity (also called *spread spectrum*) is also a security mechanism and is known to be a strong defence against interception [All.net, 2000].

3.3. Combining non-symbolic fragmentation with signal diversity

Combining non-symbolic fragmentation with signal diversity is done by sending fragment streams over different paths or networks. Even if some or all of the fragment streams are intercepted, recovery of the original data is still very difficult. However, there are still some points of weakness, such as where any eavesdropper intercepting fragment streams at a single point of weakness (e.g. the site's firewall machine) can collect all fragment streams. Masquerading identities of the sending and receiving machines is one way of making a message's fragment streams seem unrelated to each other.

A stronger idea is to make very difficult the problem of discovering which fragments were adjacent in the original data, perhaps even where the actual fragment boundaries occur in the fragmented data. This can be done in a number of ways. Disordering the fragments and creating one or more fragment streams provides improved security if the actual fragmentation is performed using a secret-keyed process that generates apparently random fragment sizes within the permitted range. In one implementation, we use a pseudorandom number generator to determine the sequences of fragment sizes. The initial "seed" value for synchronisation of the pseudorandom number generator, must be secretly exchanged so that the recipient can duplicate the fragment size sequence. Alternatively, one can assign fixed-size fragments to fragment streams using the pseudorandom number sequence, although this is a less robust solution.

There are many benefits in non-symbolic fragmentation combined with signal diversity, as discussed in [Gilbert and Ashman, 2001], and the core of its benefit is the apparent randomisation of the data. We have performed experiments which show that fragmentation combined with signal diversity introduces apparent randomness in data. This is the same effect that is expected of a reliable encryption algorithm, suggesting that it is possible to implement fragmentation algorithms which are equivalent to encryption algorithms in terms of security provided.

It is also possible to combine fragmentation with existing encryption algorithms, either by applying ciphers before or after fragmentation [Gilbert and Ashman, 2001]. We are also investigating the properties of variations in fragmentation processes in order to determine the security and computation time of fragmentation as compared to encryption.

4 CONTINUING WORK

The solutions outlined here go some way to making it possible to transmit quantities of data with reasonable commercial security but not too much performance degradation. However, these solutions are specific to protection of the transmitted data en route to its destination. A commercial data provider might also be concerned with protection of the data from misuse by the recipient.

Hierarchical encryption and fragmentation with signal diversity are all the better when combined with technologies for ensuring that data, whether legitimately received or not, are not subsequently misused. Watermarking technologies discourage copyright breaches by enabling tracing of origin of misused data, or make it unpleasant or impossible to use illicitly-copied materials by corrupting the signal, as in [Wang, 2001].

Other supporting work includes the development of modified and new encryption algorithms and protocols for non-symbolic fragmentation [Gilbert and Ashman, 2001].

References

Gilbert, M., 2000, *Communication Method*, patent pending (Amino Holdings Ltd).

Gilbert, M. and Ashman, H., 2001, *Secure Communications Method*, patent pending (Amino Holdings Ltd).

Wang, K. and Gilbert, M., 2001, *Digital Image Watermarking*, patent pending (Amino Holdings Ltd).

All.net, 2000. *CID Security Database*, http://www.all.net/CID/Defense/Defense69.html (on path diversity) and http://www.all.net/CID/Defense/Defense68.html (on spread spectrum)

Globalstar, 2000, http://www.globalstar.ca/english/about/path.shtml *Path diversity*.