FAMILY SECRETS

James Heather and Jonathan Y. Clark
Department of Computing (H3)
University of Surrey
Guildford
GU2 7XH
United Kingdom
j.heather@eim.surrey.ac.uk j.y.clark@eim.surrey.ac.uk

Abstract

We consider the possibility of secure communications over an insecure channel when the two agents have no verifiable public keys, no shared cryptographic information, and no trusted third party to assist them.

We investigate two scenarios. In the first, the agents are biologically related, and use biological data to construct a shared key; the possibility of using DNA data, shared between the two parties but not readily available to others, is considered. The second concerns unrelated parties who have some printed material, such as a photograph, in common; we explore the possibility of scanning this material at both ends and constructing a secret key from the shared information.

In each case, the two parties can convert their information into approximately equal sequences of bits. We borrow results from coding theory to show how these approximate sequences can be turned into exactly equal shared keys without compromising security in the process.

1. Introduction

Alice and Bob, as is their wont, are looking to communicate securely over an insecure channel. Usually when they find themselves in this situation, they are spoilt for choice: they can use any of a whole host of well-known cryptographic protocols to agree on a session key by means of a long-term shared secret or their long-term public keys. This time, however, they have been somewhat negligent: they have no cryptographic shared secret and no public keys, and there is no trusted third party who can verify their identities and distribute any public keys that they may choose to create. What to do now?

If they really can find nothing to work with, they will of course be stuck. Alice cannot authenticate herself except by proving that she *knows* something that only Alice is supposed to know—for example, a key she shares with a

server; or proving that she can *do* something that only Alice is supposed to be able to do—for example, recovering something from a message encrypted with her public key.

However, they may, for any of several reasons, have access to some store of shared information that is not normally considered 'cryptographic' but is nonetheless accessible only to Alice and Bob. For two arbitrary members of the human race, this is optimistic, to say the least; but if Alice and Bob know each other well, or if they are closely biologically related, there may be cause for hope.

This paper explores these possibilities in detail, presenting two different solutions to this problem. In Section 2 we discuss the feasibility of recovering approximate 'shared secrets' from biological information stored within Alice and Bob themselves, in the case that they are closely related. We then discuss in Section 3 the possibility of generating approximate shared secrets from a shared photograph or similar. Section 4 considers how they can use results from information and coding theory to turn these approximate secrets into an exact shared secret, and provides some indications about the level of security provided by these methods. In Section 5, we briefly consider other possible methods of generating approximate shared secrets. Section 6 discusses future related work; and Section 7 then forms the conclusion.

2. Biological secrets

There are a number of possible avenues of exploration for finding information shared by siblings that is rare in others. Forensic science has a long history of using fingerprints to identify individuals uniquely. Similarly, retinal images are usually unique to the individual. However, it appears that, even if the individuals are closely related, their fingerprints and retinal images will be substantially different. (Even 'identical' twins have different fingerprints.)

DNA analysis, on the other hand, now in common use in forensic studies, might be exactly what we are looking for: it is very similar in close relatives, but different in those who are unrelated.

Although DNA is modelled as a double helix, the information on the second strand merely mirrors that on the first. The nucleotide bases along the strands are considered as *base pairs*, with **C** (Cytosine) always being paired with **G** (Guanine), and **T** (Thymine) always paired with **A** (Adenine). Each base pair is at a given 'locus' along the strand, and, since there are four possibilities for the nucleotide appearing on the first strand, it can be represented with two binary digits.

The idea is to perform a DNA analysis at each end, and use parts of the analysis that are likely to be similar between close relatives, but different between unrelated people. The DNA in the nucleus of each cell contains about 3.1 bil-

lion base pairs. Much of this is shared by other humans (and, indeed, other animals), and is therefore effectively public information. However, if Alice and Bob are siblings, then other possibilities become available. One such possibility involves the DNA contained within the 'powerhouses' of the cell, the *mitochondria*. Mitochondrial DNA ('mtDNA') is identical between siblings in the same family because mitochondria are inherited from the mother. Half-brothers and half-sisters also share the same mtDNA signature provided that they share the same mother. A grandmother and a great grandmother would also share this secret mtDNA code. Blood-relative aunts would have the same mtDNA sequence only if they were sisters of the mother.

The circular human mitochondrial genome consists of about 16,569 base pairs. The best area to consider seems to be the so-called D-loop,because this is the region that contains the greatest variation. The most relevant parts are Hypervariable Region I (HVR-I), a sequence of about 342 base pairs, and Hypervariable Region II (HVR-II), comprising about 268 base pairs. These are the most highly variable regions and are of interest in forensic studies (Isenberg and Moore, 1999). In a famous case, mtDNA was used to confirm the identity of the remains of Tsar Nicholas II, since these were found to have a rare mixture of sequences (Gill et al., 1994; Massie, 1995). As a practical guideline, however, it has been reported by one forensic science laboratory (Tully et al., 1998) that more than one base difference in the mtDNA genome has not been observed by them in any one individual.

Germline mutation in mtDNA—that is, changes in mtDNA between successive generations—occurs about once in every thirty generations (Parsons et al., 1997), so two individuals with a common female ancestor up to thirty generations back would be likely to have roughly the same sequence, provided that there is an unbroken female line linking them. This means that there could be many people with the same sequence, but, one hopes, sufficiently many with different sequences to provide a 'family secret' to form at least one element of an encryption key. Where these mutations do occur, they take the form of a base insertion, a base deletion, or a substitution of one for another.

Of particular interest is an existing study reported in (Handt et al., 1998) in which the degree of variation in the HVR-I and HVR-II areas within a collection of 728 individual sequences has been determined, after careful alignment. Table 1 shows the number of variable loci, together with the number of possible nucleotide permutations. It should be noted that, although the figures are thought to be representative of the human population as a whole, finding exact figures would require all lineages to be represented within the data, which is unlikely to have been the case. The number of bits of entropy has been added by us. The optimistic figures take into account the number of possibilities observed at each locus, according to (Handt et al., 1998); the pessimistic figures assume equal probability of occurrence of any of the four DNA bases.

D-loop	Permutations	Variable loci	Bits (optimistic)	Bits (pessimistic)
HVR-I	2	188	188	376
	3	66	132	132
	4	21	42	42
Total		275	362	540
HVR-II	2	89	89	178
	3	15	30	30
	4	1	2	2
Total		105	121	210
Grand Total		**380**	**483**	**750**

Table 1. Human mtDNA D-loop: variation and representation

The latter would provide the maximum possible entropy. However, since the probabilities are not equal, and further research might improve the optimistic figures, it is more reasonable to consider a point somewhere midway between the two, which would give us around 616 bits of entropy in the variable data from the mtDNA. This is probably enough for our application, but since the data from which this was compiled might not be completely representative of the entire population, it is worthwhile also to consider other options.

Another potential source of information is the DNA in the cell nucleus. The difficulties arise when trying to find sequences unique to Alice and Bob. A possible rich source might be the so-called *short tandem repeats* (STRs), which are commonly used in forensic DNA fingerprinting. These areas are where two or more bases repeat, and the number of times they repeat can be counted. One might expect siblings to have similar lengths of STRs at these loci, and there is at least one study (Biondo, 2000) that showed it was necessary to consider nine different loci in order to separate the profile of two brothers. Although more studies like this are needed for complete confidence, if it is assumed that eight loci in common is typical for siblings, and each locus has a contribution (allele) from each parent, then that would mean 16 attributes in common between brothers out of a possible 26. If it is estimated that the maximum number of permutations per allele is twelve, then each could be represented by four bits. Thus 104 bits would be needed to represent the component of the data stream derived from the STRs in the 'normal' chromosomes.

However, if the two communicating agents are both male, then yet another possibility becomes available. Female humans have a pair of X-chromosomes, so recombination (shuffling or swapping) of genes occurs between the pair. However, the Y-chromosome in a male has no such partner. This makes it

as potentially useful a source of data as the mtDNA. A recent paper (Butler et al., 2002) highlights 20 useful loci and describes a method of investigation. Since these loci contain a contribution from only one parent (the father), 80 bits would thus be sufficient to represent this information, using the earlier assumption of around twelve different attribute values at each locus.

It is possible for two close relatives to have their mtDNA and their STRs sequenced (at a modest cost), and for each of them to turn the analysis into two bit sequences, one for the mtDNA and one for the STRs. Depending on decisions taken as to how much of the mtDNA information to use, the lengths of these sequences will be around 610 bits for the mtDNA, and 104 bits for the STRs. Whilst this data is applicable to all siblings wishing to exchange information, two brothers could also take advantage of STRs on the Y-chromosome, in which case another 80 bits could be used to provide a greater level of security.

3. Photography

Another potential source of shared data that does not require the communicants to be related is possession of a shared photograph (or other printed material). Clearly two agents who each have a copy of the same photograph should be considered to share information in some sense; and if the photograph is unavailable to others, they may be able to use this information to generate a shared key. The question is how they are to 'unlock' the photograph and agree on a shared key.

As with the biological data,what we give here is a method for obtaining approximately equal bit strings. The technique for turning these into an exact agreed secret key will be left for Section 4.

The procedure is very simple: roughly speaking, each end should scan the photograph, at an agreed resolution, reduce the colour depth, and treat the pixel information as a string of bits.

In experiments, we have managed to construct bit strings of about 700,000 bits in length, with an error rate of around 1.6%, using a moderate-sized photograph. This is more than adequate here.

There are, however, various considerations that need to be taken into account when attempting this, in order to lower the error rate and increase the chances of successfully agreeing on a shared secret. For one, each scanner has different characteristics, and care must be taken to reduce the discrepancies. Paint Shop Pro's *Histogram Equalize* filter is particularly helpful here: it normalises the data in each colour plane in the image, removing any colour or contrast bias.

Secondly, it is best to reduce each colour plane to a depth of two bits per pixel before constructing the sequence. This is still enough information to

provide for 'unguessability', but drastically reduces the error rate. It is possible even to reduce this to one bit per pixel per colour plane.

Thirdly, by pushing the photograph up into the corner of the scanner's bed, it is easy enough to avoid inadvertent rotation of the image. However, it is not so easy to avoid a small translation of the image. But since the procedure outlined below for converting these approximate sequences into an exact secret is fairly quick, it is possible to try various different offsets into the image in order to find a good match.

In Section 4, we shall find Alice sending Bob a relatively short message from which Bob can discover Alice's exact sequence, thus enabling them to agree on a shared key. If Bob ends up with the wrong key, he will need to try the procedure a few times trying different offsets until he alights on the correct key. In our experience, the required offsets have been very small (2 or 3).

Fourthly, one can reduce the effect of rotations and translations by blurring the image a little before converting into a bit sequence. Paint Shop Pro's *Gaussian Blur* has proved useful here.

4. Approximations

Let us suppose that Alice and Bob have followed the suggestions of Section 2 or Section 3. By this point, they each have a sequence of bits of roughly the same length. These sequences will be approximately the same, in the sense that it will be possible to convert one into the other by a short sequence of steps, each step involving inserting a new bit, deleting a bit, or substituting one bit for another. (A substitution can be thought of as a deletion followed by an insertion, but it will help us to consider this case separately.)

Of course, they cannot directly use these sequences to construct a shared key, because the sequences are not exactly the same, and there is nothing to be gained by encrypting a message under similar but not equal keys. What is required is to find a method of using these approximately equal sequences to agree on an exactly equal sequence, but without giving too much information away to eavesdroppers.

The exact sequence on which Alice and Bob will eventually agree need not be equal to Alice's sequence or Bob's sequence; it may be some amalgamation of the two. However, the easiest approach is to get them to agree on either Alice's sequence or Bob's sequence; we shall assume henceforth, without loss of generality, that they will attempt to agree on Alice's sequence.

4.1 Dealing with substitutions

Let us start by considering a situation in which there are known to be no insertions or deletions, but only substitutions. In that case, the sequences will be of the same length l, and if the approximations have worked well, we shall

be able to choose some k with $k \ll l$ such that there is a high probability of not more than k differences between the two sequences.

Alice has the exact sequence; Bob has an approximation to it; Alice wants to send as little as possible to Bob to enable him to establish the exact sequence. This situation is exactly the same as if Bob had known nothing about Alice's sequence, and Alice had just sent her sequence to Bob over a *noisy channel*—that is, over a channel that has a tendency to corrupt data.

The usual way of communicating over a noisy channel is to use an *error-correcting code*. (See (Welsh, 1988) for a good introduction to the general topic of coding theory.) If we wish to communicate a message from $\{\mathbf{0},\mathbf{1}\}^l$, we can construct a set $C \subseteq \{\mathbf{0},\mathbf{1}\}^{l+p}$ of *codewords* such that changing d elements of a codeword $x \in C$ never produces another codeword. (The number of changes required to turn codeword c_1 into codeword c_2 is termed the *Hamming distance* of c_1 and c_2.) By doing this, we increase the length of the transmission along the channel (from l to $l + p$), but provide some error detection and error correction capabilities.

The code described above is *d-error-detecting*. If there are at most d errors in a codeword, the receiver will be able to detect that errors have occurred, because the errors cannot result in a distinct codeword having been received.

If $d = 2z$ then the code is also said to be *z-error-correcting*. If there are no more than z errors in a transmission, the receiver will be able to determine which codeword was intended. The received vector cannot be within z errors of two distinct codewords, or otherwise these codewords would differ by at most $2z = d$ places, and the code has been constructed to make this impossible.

Alice can make use of such a code to tell Bob how to convert his approximation into the exact sequence that Alice holds. Their sequences are of length l, and there is a high probability that there are no more than k errors. They need to make use of a code that has l possible codewords and that is k-error-correcting. They will want to make the codeword length as short as possible. They also need a code in which the codeword representing c is of the form $(c|e)$; that is, where the codeword always starts with the message that is to be communicated, and then follows this up with the error-correcting information. Codes with this property are called *systematic*; not all codes are systematic.

As an example, let us take $l = 4$, $p = 7$, $k = 1$. (The numbers in practice will, of course, be much bigger.) Consider the following code, discovered by Hamming, and first published in (Shannon, 1949):

$\langle 0,0,0,0,0,0,0\rangle$	$\langle 0,1,0,0,1,0,1\rangle$	$\langle 1,0,0,0,1,1,0\rangle$	$\langle 1,1,0,0,0,1,1\rangle$
$\langle 0,0,0,1,1,1,1\rangle$	$\langle 0,1,0,1,0,1,0\rangle$	$\langle 1,0,0,1,0,0,1\rangle$	$\langle 1,1,0,1,1,0,0\rangle$
$\langle 0,0,1,0,0,1,1\rangle$	$\langle 0,1,1,0,1,1,0\rangle$	$\langle 1,0,1,0,1,0,1\rangle$	$\langle 1,1,1,0,0,0,0\rangle$
$\langle 0,0,1,1,1,0,0\rangle$	$\langle 0,1,1,1,0,0,1\rangle$	$\langle 1,0,1,1,0,1,0\rangle$	$\langle 1,1,1,1,1,1,1\rangle$

The first $l = 4$ bits in each codeword are the message (making this code systematic); the last $p - l = 3$ are the error-correcting bits. Any two distinct codewords differ in at least three places, so that if there is a single error in a codeword transmission, it can be corrected. The code is thus 1-error-correcting.

4.2 Security by photograph

If Alice and Bob are using a scanned photograph as the basis for their shared key, this is all they need. Following the principles set out in Section 3, they can easily each get a large sequence of bits, with an error rate of under 10%.

Alice must then choose the systematic code that they will use, and tell Bob (over a possibly insecure channel) what the code is. She should then send him the error-correcting digits corresponding to the message formed by her sequence; Bob can append these error-correcting digits to the sequence that he has, and decode the resulting codeword to recover Alice's.

Alice and Bob now have the same sequence. It is likely that this sequence will be too long for use directly as a shared key; however, they can each apply a hash function to the sequence to get something of a more appropriate length. The output of a hash function is often of the order of 100–200 bits, which is ideal for symmetric cryptography. A good hash function will also ensure, provided there is enough entropy in the bit sequence, that the resulting shared key is uniformly distributed in the key space.

This, we believe, is secure against a Dolev-Yao-style intruder who can engage not only in passive attacks but also in attacks requiring spoofing, interception or manipulation of messages. The intruder cannot break security by changing the code, or modifying the codeword. If he changes anything, Bob's decoding will not result in the same sequence as Alice's, and they will not agree on the shared key. However, this will quickly become obvious to them when they try to use it, and the intruder will not have managed to learn either of the keys in any case. Clearly if the intruder has control over the communications medium then he can stop them from successfully agreeing on a shared key; but there is nothing that can be done about this. An intruder who has control over the communications medium can always stop agents from communicating.

4.3 Choosing a code

If the codewords are p bits long, and Alice's sequence itself is l bits long, she will need to transmit $p - l$ bits over the insecure channel in order for Bob to be able to establish her sequence.

It is not immediately clear exactly how much information about Alice's sequence can be deduced by the intruder from these $p - l$ bits. However, if only $p - l$ bits have been transmitted, then certainly at most $p - l$ bits of information can have been leaked to the intruder concerning Alice's sequence.

The quantity of information that remains secret from the intruder is, then, at least the amount of secret information in Alice's sequence less $p - l$. Getting an acceptable level of security depends, therefore, on keeping this quantity of information as high as possible. Alice and Bob will want to find a code to minimise $p - l$.

Bounds on $p - l$. Hamming's lower bound on the codeword size gives

$$2^l \leqslant \frac{2^p}{V(p,k)} \qquad \text{where} \qquad V(n,r) = \sum_{i=0}^{r} \binom{n}{i}$$

Rough working figures for this are that if Alice's sequence is l bits long, with $10^2 < l < 10^5$, a lower bound on the number of error-correcting bits required will be somewhere around $0.66l$ to correct an 8% error rate, $0.4l$ to correct a 5% error rate, or $0.25l$ to correct a 3% error rate. These factors get smaller as l increases, but not significantly so in the range that fits our purposes.

A code that meets exactly this theoretical lower bound is called *perfect*. Perfect codes do exist for many values of l, p and k, but they are hard to find.

Gilbert, Shannon and Varshamov's upper bound, alongside work in (Garcia and Stichtenoth, 1995), gives a method for constructing codes such that

$$2^l \geqslant \frac{2^p}{V(p,2k)}$$

This will halve the allowable error rate in the figures given above, so that if Alice's sequence is l bits long, with $10^2 < l < 10^5$, we can construct a code with the number of error-correcting bits at somewhere around $0.66l$ to correct a 4% error rate, $0.4l$ to correct a 2.5% error rate, or $0.25l$ to correct a 1.5% error rate.

Practical implications. In practice, for the photograph technique, results similar to ours would result in Alice and Bob agreeing on a sequence of around 700,000 bits, by sending somewhat less than 250,000 bits of error-correcting data. Running a suitable hash function on the agreed 700,000-bit sequence will create a secret key of about 128 bits, depending on the hash function; this is a decent size of key to use for symmetric-key cryptography.

Whether this method should be regarded as secure hinges critically on the information content of a photograph. Unfortunately, objective answers are difficult to find here: it largely depends on what sort of photograph is used—and how can one possibly determine the information content of, say, a photograph of a tree by a lake?

The information content is certainly much lower than the simple size of the photograph in pixels multiplied by the colour depth: JPEG compression can

reduce the file size by a factor of ten without significant loss of image quality. In addition, much of the information will be lost by the reduction in colour depth, and the Gaussian blur applied to the image. This notwithstanding, our belief is that the number of bits in the sequence is so high that the information content will still exceed the 128-bit key that results from the hash function. For instance, if one imagines a photograph of a man standing in a busy high street, it would not be difficult to think of 128 independent factors that could be varied (angle of shot, distance to subject, number of people in background, name of third shop on the left) and that would affect the image and hence the agreed bit sequence. However, these things are admittedly difficult to quantify.

Additionally, of course, it relies on the accuracy of Alice and Bob's beliefs about who holds (or can get hold of) what photograph. If the photograph that they believe to be secret is in fact available to an attacker, or if an attacker can persuade Alice to use a photograph that he holds, then clearly security is lost. This procedure can be used only with a photograph where its distribution is known to the participants.

4.4 Security by biology

The codes referred to above will not be of great help with genetic 'secrets', where the differences between one sequence and the other are attributable not only to substitutions but also to insertions and deletions. Clearly an insertion or deletion near the start of the sequence will cause a match failure from that point onwards, and the Hamming distance between the two sequences will be high even though the sequences may be intuitively very similar.

Fortunately, there are measures of distance between sequences other than the Hamming distance. The *Levenshtein distance*, first proposed in (Levenshtein, 1966), takes into account the minimum number of substitutions, insertions, deletions and transpositions (swapping of two consecutive elements) required to turn one sequence into the other. It is the basis of automatic spelling correction techniques, where, for instance, we should want to be able to correct `URPLE` to `PURPLE` without naïvely attempting to match the words letter by letter and rejecting the match. Work in (Schulman and Zuckerman, 1997), among others, building on previous work in (Levenshtein, 1966; Levenshtein, 1992), (Spielman, 1995), (Okuda et al., 1976), (Varshamov and Tenengolts, 1965), and (Calabi and Hartnett, 1969), allows for construction of efficient error-correcting codes that correct insertions and deletions as well as substitutions. The detail of these codes is beyond the scope of this paper; however, Alice does by means of these codes have an efficient means of transmitting error-correcting information to Bob, again enabling him to recover her genetic sequence.

Also, since there is a low expected error rate between the two sequences when using genetic data, the number of error-correcting bits will be small.

The method given in Section 2 allows Alice and Bob to create sequences totalling over 700 bits in length. The discussion there indicates that we can expect at most one error (insertion, deletion or substitution) in the mtDNA part of the sequence, and at most a 10% error rate in the STR part. Using any of several of the available codes that allow for correction of insertions and deletions, Alice should be able to communicate her exact mtDNA sequence to Bob by sending less than 50 bits of error-correcting data; she can then communicate the exact bit sequence constructed from her STRs (which may differ from Bob's by substitution, but not by insertion or deletion) by sending a similar amount of error-correcting data using the code suggested in Section 4.3.

Practical implications. As in the case of the photograph, it is difficult to quantify the information content in the bit sequences obtained by DNA analysis. Here, it is not any lack of objective criteria that cause the difficulty, but simply that although the science of DNA analysis has produced remarkable results, it is still too young for firm answers to many questions to be available. Not enough people have had their complete DNA sequenced. However, there are few indications at present that there are 'patterns' in the DNA considered here that would reduce the information content drastically. Until some drastic 'decoding' of this DNA demonstrates otherwise, it appears that the information content is reasonably high.

Alice and Bob would then, as with the photograph, apply a hash function to the bit sequence in order to generate a shared key. With current information, we believe that the information content of the secret data agreed upon by Alice and Bob is at least as high as the number of bits produced by the hash function for use as the secret key. However, it is acknowledged that future advances in human biological science may either support or contradict this position.

The security of this method clearly relies on making certain that an attacker cannot get hold of enough genetic material from Alice or Bob to be able to reproduce the DNA sequence for himself. In the short to medium term, depending on the circumstances and the importance of the secret to the attacker, this may well be realistic: the attacker would need a reasonable quantity of genetic material, and would also need to be able to collect it without contaminating the samples. In the long term, if the attacker has significant funds at his disposal or is geographically close to either party, it is possible that he could gather enough material to find the encryption key. This method is therefore probably most applicable in situations in which it is vital that something remain secret, but only for a limited time. It is an emergency measure rather than a suggested pattern for everyday life! It should also be noted that this is a procedure that Alice and Bob cannot use more than once to generate a session key. If they use the procedure twice, they will end up with the same key.

5. Other possibilities

The connection between the photograph and the DNA analysis is simply that each provides a mechanism for extracting an approximate secret shared between the two participants. It may be that there are other ways of constructing such an approximate sequence; however, it is worth noting that two of the 'easy' options have significant problems.

Text from a book (and similar). Alice and Bob cannot simply agree to use the text "on page 53 from Barbara Cartland's latest novel": the point is that they will be agreeing this over an insecure channel, and an attacker may be listening in. If he hears them decide which book to use, it will usually be a simple matter for him to obtain a copy of the book. Of course, Alice and Bob may have ways of alluding to the book without saying its title; but this just pushes the issue back a stage. Here, the allusion that they understand but the attacker doesn't is itself the shared secret. They will need to be sure how much information content there is in this shared secret before they commit to using it for cryptographic purposes.

Challenge and response. It may be that Alice and Bob can get somewhere by using information relating to past common experiences: maybe they are the only ones who know the answer to certain questions about their past.

Two scenarios need to be carefully distinguished here. In one, they authenticate each other by means of various questions until they are satisfied that they really are speaking to each other, after which they have their private conversation; in the other, they use answers to such questions to create a shared key.

The former is not appropriate here. If the approach is to be able to deal with Dolev-Yao attackers then there can be no guarantee that the attacker will not wait for the authentication to take place and then subsequently break in and masquerade as one party or the other. Indeed, even if he does nothing active, he may still overhear the private conversation.

Construction of a shared key using answers to such questions may work; however, it would require immense care and patience. To construct a key with, say, 128 bits of entropy may need a lot of questions. Even 128 yes/no questions will not be sufficient unless the probability of a yes for each question is 0.5 and is independent of probabilities for all the other questions. Other questions may have more information content, but assessing the information content and the secrecy of the answer will be very difficult.

6. Future work

It may be that there are more efficient ways of turning approximately equal sequences into exactly equal ones. Efficiency here is well worth striving for,

since its consequence is that less error-correcting data will be transmitted, and so less information about the secret key will be leaked to any eavesdropper.

If improvements are to be found, there are four promising lines of enquiry.

In the first place, Alice and Bob's agreed sequence need not be either Alice's or Bob's original sequence. It may be that there is a way for them to agree on some combination of the sequences with less communication than is required for them to agree on Alice's sequence. It may also be possible for them to exchange a small quantity of information that would enable them to determine which parts of the sequence are likely to be the same and which parts are not. If so, then they would be able to drop the different parts of the sequence and exchange error-correcting data just on the parts likely to be the same; this would reduce the length of the agreed sequence, but with a possibly significantly reduced error rate.

Secondly, Wyner in (Wyner, 1975) and Csiszár and Körner in (Csiszár and Körner, 1978) conducted work on secure key agreement over noisy channels. Their models may shed light on the techniques discussed in this paper; for, as we have already shown, the problem of converting approximate shared secrets to exact shared secrets is closely related to the problem of communicating secrets over noisy channels.

In addition, there is recent work by Maurer and Wolf (see (Maurer and Wolf, 1997; Maurer and Wolf, 2000) among others) on *privacy amplification*, in which two agents hold a secret about which a Dolev-Yao attacker knows partial information; they discuss how to convert this into a secret about which almost nothing is known by the attacker. Their analysis also includes consideration of noisy channels. The work they present differs from ours in that in their scenario the information is fully shared but partially secret, whereas in ours the information is only partially shared. However, this, we believe, will provide interesting avenues for further exploration. In particular, it may be possible to use their results to allow for and remove any information known to the attacker resulting from less than perfect entropy in DNA and in photographs.

Thirdly, the codes discussed here are all designed to cope with errors that can appear anywhere in the codeword, including in the error-correction data. This is more robust than is required for our purposes. We need to allow for errors in Bob's version of Alice's sequence, but we do not need a code that allows for transmission errors in the error-correcting part of the code. This, of course, reduces the number of cases that our error-correcting part needs to be able to distinguish, and so might allow for a reduction in the size of the error-correcting part itself. A code of this nature would be highly specialised; in fact, it is difficult to think of any possible applications of it other than this one. It is not surprising, then, that there seems not to have been any work done on such codes. Investigation of this is the subject of planned future work.

Finally, the notion of *edit distance* may be useful here. Atallah, Kerschbaum and Du provide a way (Atallah et al., 2003) of enabling Alice and Bob to calculate the edit distance of their similar sequences without revealing to each other any more information than is contained in the edit distance itself. This could certainly be used by Alice and Bob initially to determine the similarity of their sequences and give a guide for how much error-correcting information is needed; the technique could possibly be adapted to find an efficient way of agreeing on a common sequence.

7. Conclusion

In this paper, we have given two methods by which two agents may be able to agree on a secret key even when they have no previously agreed cryptographic data with which to work, and no trusted third party who can verify their identities and distribute public keys for them. In one case, where the agents are closely biologically related, they can construct approximately equal bit sequences by using the information stored in certain parts of their DNA; in the other case, agents who have a shared photograph can extract the information from this photograph and manipulate the information so as to construct approximately equal bit sequences.

Regardless of which method is used, they can then use the techniques of Section 4 to convert this into an exact shared secret that can be used for cryptographic purposes. Although the approaches discussed here relate to DNA and to photographs, any other approach that generates approximate shared secrets could equally use the techniques of this section to construct a shared key.

It is worth observing that although the paper has been phrased in terms of two agents who wish to share secret information between them, the approach could be trivially extended to cover generation of a 'group key' for three or more siblings, or three or more agents who all have the same photograph.

Much of what is presented here is admittedly speculative, and may often be inapplicable to the two agents in question. However, it is sufficient to demonstrate that the usual assumption in this regard—essentially, that one cannot generate something from nothing—may not always hold.

Some of the more 'cloudy' issues will become clearer over time. There is still much to be learnt about DNA, its information content, and its variation throughout the population; the cost and difficulty of sequencing will also come down over time. Biometric information will undoubtedly come to play a large part in the world of security. Research is also ongoing into the discovery of efficient codes; new light shed on this area may allow agents to generate and use codes whose information rate is much closer to Hamming's lower bound.

Acknowledgements. Thanks are due to the anonymous referees for their insightful comments, and also to the FAST 2004 participants for a lively, interesting and useful discussion following presentation of the paper.

References

Atallah, Mikhail J., Kerschbaum, Florian, and Du, Wenliang (2003). Secure and Private Sequence Comparisons. In *Proceedings of the ACM workshop on Privacy in the electronic society*, pages 39–44. ACM Press.

Biondo, R. (2000). The impact of CODIS software in criminal investigations in the Italian police. In *Eleventh International Symposium on Human Identification*. Available from `www.promega.com/geneticidproc/ussymp11proc/content/biondo.pdf`.

Butler, J. M., Schoske, R., Vallone, P. M., Kline, M. C., Redd, A. J., and Hammer, M. F. (2002). A novel multiplex for simultaneous amplification of 20 Y chromosome STR markers. *Forensic Science International*, 129:10–24.

Calabi, L. and Hartnett, W. E. (1969). A family of codes for the correction of substitution and synchronization errors. *IEEE Transactions on Information Theory*, IT-15:102–106.

Csiszár, I. and Körner, J. (1978). Broadcast channels with confidential messages. *IEEE Transactions on Information Theory*, 24(3):339–348.

Garcia, A. and Stichtenoth, H. (1995). A tower of Aritin-Schreier extensions of function fields attaining the Drinfed-Vladut bound. *Inventiones Mathematicae*, 121(1):211–222.

Gill, P., Ivanov, P. L., Kimpton, C., Piercy, R., Benson, N., Tully, G., Evett, L., Hagelberg, E., and Sullivan, K. (1994). Identification of the remains of the Romanov family by DNA analysis. *Nature Genetics*, 6:130–135.

Handt, O., Meyer, S., and von Haeseler, A. (1998). Compilation of human mtDNA control region sequences. *Nucleic Acids Research*, 26(1):126–129.

Isenberg, A. R. and Moore, J. M. (1999). Mitochondrial DNA Analysis at the FBI Laboratory. *Forensic Science Communications*, 1(2). Available from `http://www.fbi.gov/hq/lab/fsc/backissu/july1999/dnatext.htm`.

Levenshtein, V. I. (1966). Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics—Dolkady*, 10(8):707–710.

Levenshtein, V. I. (1992). On perfect codes in deletion and insertion metric. *Discrete Mathematics and its Applications*, 2(3):241–258.

Massie, R. K. (1995). *The Romanovs: The Final Chapter*. Random House, New York.

Maurer, Ueli and Wolf, Stefan (1997). Privacy amplification secure against active adversaries. In Jr., Burton S. Kaliski, editor, *Advances in Cryptology—CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 307–321. Springer-Verlag.

Maurer, Ueli and Wolf, Stefan (2000). Information-theoretic key agreement: From weak to strong secrecy for free. In *Advances in Cryptology—EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 351–368. Springer-Verlag.

Okuda, T., Tanaka, E., and Kasai, T. (1976). A method for the correction of garbled words based on the Levenshtein metric. *IEEE Transactions on Computing*, C-25(2):172–176.

Parsons, T. J., Muniec, D. S., Sullivan, K., Woodyatt, N., Alliston-Greiner, R., Wilson, M. R., Berry, D. L., Holland, K. A., Weedn, V. W., Gill, P., and Holland, M. M. (1997). A high observed substitution rate in the human mitochondrial dna control region. *Nature Genetics*, 15:363–368.

Schulman, Leonard J. and Zuckerman, David (1997). Asymptotically Good Codes Correcting Insertions, Deletions, and Transpositions. In *Symposium on Discrete Algorithms*, pages 669–674.

Shannon, Claude (1949). Communication Theory of Secrecy Systems. Technical report, Bell Systems.

Spielman, Daniel A. (1995). Linear-time encodable and decodable error-correcting codes. pages 388–397.

Tully, G., Morley, J. M., and Bark, J. E. (1998). Forensic analysis of mitochondrial DNA: application of multiplex solid-phase - fluorescent minisequencing to high throughput analysis. In *Second European Symposium on Human Identification*, pages 92–96. Online at `http://www.promega.com/geneticidproc/eusymp2proc/20.pdf`.

Varshamov, R. R. and Tenengolts, G. M. (1965). Codes which correct single asymmetric errors. *Automatika i Telemekhanika*, 26(2):288–292.

Welsh, Dominic (1988). *Codes and Cryptography*. Oxford University Press.

Wyner, A. D. (1975). The wire-tap channel. *Bell System Technical Journal*, 54(8):1355–1387.