# A storage complexity based analogue of Maurer key establishment using public channels

C.J. Mitchell[1]

Royal Holloway, University of London, Egham, Surrey TW20 0EX, U.K.

**Abstract.** We describe a key agreement system based on the assumption that there exists a public broadcast channel transmitting data at such a rate that an eavesdropper cannot economically store all the data sent over a certain time period. The two legitimate parties select bits randomly from this channel, and use as key bits those which they have selected in common. The work is inspired by recent work of Maurer, [3].

## 1 Introduction

In a recent paper, [3], Maurer has described a number of related methods for providing secret key agreement between two parties using only publicly available information. These methods are based on a development of Wyner's ideas, [4].

The particular method which has inspired the work described here relies on the two parties wishing to agree a secret key, and any eavesdropper, all receiving a signal from some channel, for which each party only receives a noisy version of the originally transmitted signal. For the system to operate, the common information derived from the channel by the two legitimate parties, $A$ and $B$ say, must not be a subset of the information derived from the channel by any eavesdropper. Note that it is not necessary for $A$ or $B$ to receive a 'better' version of the signal than the eavesdropper.

This requirement is very simply met in any situation where the channel errors are statistically independent for each of the three parties. However, in practice this requirement may be rather difficult to guarantee. As an example consider a radio channel: the eavesdropper might be able to position his antenna close to one of the legitimate parties, and hence obtain a strictly better version of the signal than one of the legitimate parties.

It is also necessary for the parties $A$ and $B$ to share an error-free 'authenticated channel', although this may also be intercepted by the eavesdropper. By an authenticated channel we mean one in which $B$ is able to verify that all the data received on the channel originated from $A$ in exactly the same form as was received, and vice versa. This may also be non-trivial to provide in practice.

In this paper we consider a slightly different key agreement system, which avoids the first requirement above. This system is based on the assumption that there exists a public broadcast channel transmitting data at such a rate that an eavesdropper cannot economically store all the data sent over a certain time period. The two legitimate parties randomly choose which bits to store from

this broadcast channel, and then compare notes (using an authenticated channel which need not provide privacy) after the chosen time period has passed as to which bits they have both selected, which then constitute the shared secret key. Unless the eavesdropper has stored a high proportion of all the bits transmitted on the public broadcast channel, it will almost certainly have no more than a small proportion of the secret key bits. This idea is analogous to the encryption system described by Maurer in [2], where the idea of a noise source producing data in quantities which cannot economically be stored is also exploited. However, the idea in [2] is rather different, in that the legitimate users do not divulge which of the bits they have used.

In the scheme described here the provable guarantees of security which can be obtained for Maurer's schemes, [3], are thus exchanged for arguments regarding the cost (in providing large amounts of data storage) to a third party of obtaining the key agreed by $A$ and $B$. It is important to note that this scheme, like all the schemes in [3], still requires the error-free authenticated channel.

Before proceeding observe that, for clarity and brevity of presentation, we only provide informal arguments to support certain of our main results. Formal proofs can be constructed using information theoretic arguments.

## 2 The basic scheme

In order to describe our key agreement system we first need to describe our model of the communicating parties. We suppose that $A$ and $B$, the two legitimate parties wishing to establish a shared secret key, both have access to an (error free) public broadcast channel sending random binary data at a high rate, say $R$ bits/sec. We suppose also that $A$ and $B$

- share an authenticated error-free channel,
- have the means to store $n_A$ and $n_B$ bits respectively, and
- are 'synchronised' with respect to the broadcast channel, i.e. they have the means to refer to a single bit sent on this channel.

By an authenticated channel shared by $A$ and $B$ we mean a channel for which $A$ can be sure that any bits received claiming to be from $B$ are genuinely from $B$, and have not been manipulated in transit (and vice versa).

The eavesdropper, who we call $C$, also has error-free access to both the public broadcast channel and the authenticated channel between $A$ and $B$.

The basic key agreement system works as follows. Note that in a subsequent section we describe some improvements on this basic scheme.

*Algorithm K*

1. $A$ and $B$ both monitor the broadcast channel for an agreed interval of time of duration $T$. The start and end points of this interval can be agreed using the authenticated channel. We assume that the exact details of the time interval are also known to the eavesdropper $C$. Note that this means that $N = TR$ bits are transmitted during the agreed interval.

2. During this interval $A$ selects $n_A$ of the bits sent over the broadcast channel at random and stores them. Similarly, and independently, $B$ selects $n_B$ bits at random and stores them.
3. At the end of the interval (and not before) $A$ sends $B$ the 'indices' of the bits that it has stored, where the bits sent over the public channel during the time interval are successively given the indices $0, 1, \ldots$, etc. Having received these indices, $B$ examines them and compares them with the indices of the bits it has stored and makes special note of any coincidences.
4. $B$ now sends back to $A$ a list of all coincidences, and the bits corresponding to these coincident indices become the key bits (which both $A$ and $B$ have).

As we now show, given that $A$ and $B$ make genuinely random selections, and $T$, $R$, $n_A$ and $n_B$ are chosen appropriately (with $N = TR$), the eavesdropper will be obliged to store many more bits than either $A$ or $B$ to have a good probability of knowing more than a very few of the key bits. To demonstrate this we first establish the following simple result.

***Theorem 1.*** *Suppose $A$ and $B$ follow Algorithm K, and an eavesdropper stores $n_C$ bits randomly selected from the broadcast channel during the agreed time interval. Suppose also that $n_A$ and $n_B$ are both very much smaller than $N$ (the number of bits sent during the agreed time interval). Then the following will hold.*

(i) *The expected number of bits of key shared by $A$ and $B$ at the end of the process will be approximately $n_A n_B / N$.*
(ii) *The expected number of key bits available to $C$ is approximately $n_A n_B n_C / N^2$.*

*Proof.* Both results follow from elementary probability considerations.

*(i)* For any given bit of the $n_A$ selected by $A$, the probability that it is also selected by $B$ is $n_B/N$. Given that $n_A$ is very small with respect to $N$, we may ignore the fact that the probabilities are not independent and hence say that the expected size of the set of bits selected by both $A$ and $B$ will be *approximately* $n_A$ times the above probability, and *(i)* follows.
*(ii)* follows by a precisely analogous argument.

Before proceeding note that $A$ will actually need $(L+1)n_A$ bits of storage, where $L = \lceil \log_2 N \rceil$, i.e. $L$ is the number of bits in the index values. Similarly $B$ will need $(L+1)n_B$ bits of storage. In addition $A$ will need to send $Ln_A$ bits over the authenticated channel as part of the key agreement process.

We have thus presented a system which provides a cost difference between legitimate key agreement and unauthorised interception of key material.

## 3 A simple example of the system

To illustrate how such a system might operate we consider a simple example.

Suppose $T$ is $10^5$ seconds (i.e. approximately one day) and $R$ is $10^{10}$ bits per second, i.e. 10 Gbits/sec, and hence $N = TR = 10^{15}$ and $L = 50$. Now suppose

that $n_A = n_B = 3 \times 10^8$ and hence $A$ and $B$ will need to have $51 \times 3 \times 10^8$ bits of storage, i.e. a little under 2 Gigabytes. At current prices, high speed magnetic disk storage of this capacity will cost approximately £500, and prices are likely to continue to fall. At the same time, $A$ will need to send a little under 2 Gigabytes of information to $B$ over the shared authenticated channel. By Theorem 1, at the end of the process $A$ and $B$ will expect to share a key of approximately $(3 \times 10^8)^2/10^{15} = 90$ bits.

We next consider what strategy the eavesdropper might adopt to try and learn significant amounts of key information. In order to obtain, say, 10% of the key bits, by Theorem 1 the eavesdropper will need to store 10% of the bits sent over the public broadcast channel. This will require $10^{14}$ bits of storage, i.e. approximately 12,000 Gbytes. At today's prices, low cost storage (e.g. magnetic tape) still costs significantly more than £10 per Gbyte, and hence such storage will cost the eavesdropper well in excess of £120,000, and, by similar arguments, to obtain 50% or 100% of the key bits would cost in excess of £600,000 or £1,200,000 respectively.

If $A$ and $B$ were concerned about the possibility that an eavesdropper could make use of a small number of the key bits, then security could be increased by using a one-way hash function to produce, say, a 64-bit key from the 90 bits derived from the key exchange process.

## 4 Extensions of the basic scheme

There are a number of ways in which the basic system can be modified to increase the cost differential between the legitimate parties and the eavesdropper. We consider two such possibilities. Both offer methods by which the storage requirements for $A$ and $B$ can be reduced from $(L+1)n_A$ and $(L+1)n_B$ to around $n_A$ and $n_B$ respectively.

### 4.1 Pseudo-random selection of bits by $A$ and $B$

Suppose $A$ and $B$ have agreed in advance the choice of a cryptographically secure pseudo-random number generator. By this we mean a generator which, given a secret key as input, produces a sequence of pseudo-random numbers as output and for which, given knowledge of some of the output sequence, it is computationally infeasible to compute any more information regarding the output sequence (in particular it will be computationally infeasible to deduce the key used to generate this sequence).

We now describe a modified version of the basic system described in Algorithm K, which makes use of such a pseudo-random number generator.

*Algorithm L*

1. Before starting the process $A$ and $B$ select random keys for their chosen pseudo-random number generator, which we call $R_A$ and $R_B$ respectively.

2. $A$ and $B$ both monitor the broadcast channel for an agreed interval of time of duration $T$ (and $N = TR$ as previously). The start and end points of this interval can be agreed using the authenticated channel. We assume that the exact details of the time interval are also known to the eavesdropper $C$. For the purposes of this discussion we suppose that the bits sent over the broadcast channel during the selected time interval are labelled

$$b_0, b_1, \ldots, b_{N-1}.$$

3. Before starting the monitoring $A$ and $B$ also choose *step values* $s_A$ and $s_B$ respectively, where $s_A = \lfloor N/n_A \rfloor$ and $s_B = \lfloor N/n_B \rfloor$. If necessary, at some point (either before, during or after the monitoring period) $A$ and $B$ exchange these step values.
4. During the time interval, $A$ uses the chosen pseudo-random number generator and its secret key $R_A$ to produce a sequence $t_0, t_1, \ldots, t_{n_A-1}$ of pseudo-random numbers, where each pseudo-random number $t_i$ is chosen from the range $0, 1, \ldots, s_A - 1$ (with uniform probabilities). $A$ then selects the following $n_A$ bits sent over the broadcast channel and stores them:

$$b_{t_0}, b_{s_A+t_1}, b_{2s_A+t_2}, \ldots, b_{(n_A-1)s_A+t_{n_A-1}}.$$

Similarly, and independently, $B$ uses the pseudo-random number generator and its secret key $R_B$ to produce a sequence $u_0, u_1, \ldots, u_{n_B-1}$ of pseudo-random numbers, where each pseudo-random number $u_i$ is chosen from the range $0, 1, \ldots, s_B - 1$ (with uniform probabilities). $B$ then selects the following $n_B$ bits sent over the broadcast channel and stores them:

$$b_{u_0}, b_{s_B+u_1}, b_{2s_B+u_2}, \ldots, b_{(n_B-1)s_B+u_{n_B-1}}.$$

5. At the end of the interval (and not before) $A$ sends $B$ its secret key $R_A$, used to help select which bits from the public channel it has stored during the time interval. Similarly $B$ sends $A$ its secret key $R_B$.
6. $A$ can then use $R_A$ and $R_B$ to find those values of $i$ and $j$ ($0 \leq i < n_A$, $0 \leq j < n_B$) for which $is_A + t_i = js_B + u_j$. This can be done with the minimum of storage by at any point retaining the values of $i$, $t_i$, $j$ and $u_j$, and then
   - replacing the pair $(i, t_i)$ with the pair $(i+1, t_{i+1})$ if $is_A + t_i < js_B + u_j$,
   - replacing the pair $(j, u_j)$ with the pair $(j+1, u_{j+1})$ if $is_A + t_i > js_B + u_j$, and
   - storing the values $is_A + t_i$ whenever $is_A + t_i = js_B + u_j$.

   $B$ can do precisely the same calculations, leaving $A$ and $B$ with a known set of mutually held bits (which can be used to create a key).

Before attempting to describe the performance of this modified scheme, we first consider the best strategy for an eavesdropper wishing to find as many key bits as possible using the minimum amount of storage. There would appear to be three obvious strategies for the eavesdropper, $C$. Firstly $C$ could choose to store a random selection of bits from the channel (without regard to the

'step values'). Secondly $C$ could store a fixed number of bits from each range, $\boldsymbol{b}_t = (b_{ts_A}, b_{ts_A+1}, \ldots, b_{(t+1)s_A-1})$, for $t = 0, 1, \ldots, n_A - 1$. Thirdly, $C$ could select a number of ranges $\boldsymbol{b}_t$ for various values of $t$, $(0 \leq t < n_A)$, and store all the bits for the selected ranges. Note that, alternative versions of the second and third strategies would involve replacing $s_A$ and $n_A$ with $s_B$ and $n_B$.

Now, since there is at most one key bit within any range $\boldsymbol{b}_t$, $(0 \leq t < n_A)$, the second strategy would seem to be the best (although all strategies yield very similar results when $n_A$ and $n_B$ are small relative to $N$). In the following simple result, in which we derive the performance of this revised scheme, we therefore assume that the eavesdropper is using the second of the above strategies.

***Theorem 2.*** *Suppose $A$ and $B$ follow Algorithm L, and an eavesdropper $C$ stores $n_C$ bits selected from the public broadcast channel during the agreed time interval. Suppose also that $n_C = dn_A$ for some integer $d$, and that $C$ stores $d$ bits from each range $\boldsymbol{b}_t$, $(0 \leq t < n_A)$. As previously we assume that $n_A$ and $n_B$ are both very much smaller than $N$. Then the following will hold.*

(i) *The expected number of bits of key shared by $A$ and $B$ at the end of the process will be approximately $n_A n_B / N$.*
(ii) *The expected number of key bits available to $C$ is approximately $n_A n_B n_C / N^2$.*

*Proof. (i)* Consider any range: $\boldsymbol{b}_t$ (for some value of $t$ satisfying $0 \leq t < n_A$). At the end of the agreed time interval, $A$ will store exactly one bit from this range. The probability that $B$ will also store this bit is equal to $n_B/N$. Given that there are $n_A$ such ranges, and assuming that these probabilities are independent (which is a reasonable approximation given $n_A$ and $n_B$ are small with respect to $N$), we see that the expected number of bits held by both $A$ and $B$ at the end of the agreed time interval is approximately equal to $n_A n_B / N$, as required.

*(ii)* As previously, consider any range: $\boldsymbol{b}_t$ (for some value of $t$ satisfying $0 \leq t < n_A$). At the end of the agreed time interval, $A$ will store exactly one bit from this range. The probability that $B$ and $C$ will also store this bit is equal to $(n_B/N)(d/s_A)$. Given that there are $n_A$ such ranges, and assuming that these probabilities are independent (which is a reasonable approximation given $n_A$ and $n_B$ are small with respect to $N$), we see that the expected number of bits held by all of $A$, $B$ and $C$ at the end of the agreed time interval is approximately equal to $n_A n_B d / N s_A = n_A n_B n_C / N^2$, as required.

Before proceeding note that $A$ will only need $n_A$ bits of storage. Similarly $B$ will only need $n_B$ bits of storage. In addition $A$ and $B$ will only need to send their respective secret keys $R_A$ and $R_B$ over the authenticated channel as part of the key agreement process.

Hence this amended procedure reduces the storage for $A$ and $B$ to $n_A$ and $n_B$ respectively, minimises use of the authenticated channel, and also makes the match-finding process a simple one. This is at the cost of making the security dependent on the computational security of the pseudo-random number generator employed by $A$ and $B$. To show how effective this improvement is we consider a

modified version of our previous example; we use the same cost assumptions as in Section 3.

Suppose $T$ is $10^5$ seconds (i.e. approximately one day) and $R$ is $10^{12}$ bits per second, i.e. 1000 Gbits/sec, and hence $N = TR = 10^{17}$. Now suppose that $n_A = n_B = 3 \times 10^9$ and hence $A$ and $B$ will need to have $3 \times 10^9$ bits of storage, i.e. a little under 400 Megabytes, costing no more than £100. At the same time, $A$ will need to send only one key (of say 128 bits) to $B$ over the shared authenticated channel (and vice versa). By Theorem 2, at the end of the process $A$ and $B$ will expect to share a key of approximately $(3 \times 10^9)^2/10^{17} = 90$ bits.

We next consider the position of the eavesdropper. In order to obtain, say, 10% of the key bits, the eavesdropper will need to store 10% of the bits sent over the public broadcast channel. This will require $10^{16}$ bits of storage, i.e. approximately 1,200,000 Gbytes. Such storage will cost the eavesdropper well in excess of £12,000,000, and, by similar arguments, to obtain 50% or 100% of the key bits would cost in excess of £60,000,000 or £120,000,000 respectively.

### 4.2 Block-wise selection of bits

In the previous section we described a system which minimises both the storage requirements for $A$ and $B$ and the use of the authenticated channel for $A$ and $B$. This was at the cost of making the security depend on the cryptographic properties of a pseudo-random number generator. We now consider a slightly different modification of the basic scheme which retains many of the advantages of the scheme described in Section 4.1, but which does not rely on any computational security assumptions. The procedure is as follows.

*Algorithm M*

1. $A$ and $B$ both monitor the broadcast channel for an agreed interval of time of duration $T$. The start and end points of this interval can be agreed using the authenticated channel. We assume that the exact details of the time interval are also known to the eavesdropper $C$. Suppose that the bits sent over the broadcast channel during the selected time interval are labelled

   $$b_0, b_1, \ldots, b_{N-1}.$$

   To make our discussions simpler we also assume that $n_A | N$ and $n_B | N$, and hence define $s_A$ and $s_B$ by $s_A n_A = s_B n_B = N$. Suppose moreover that $s_A s_B | N$, and define $w$ by $s_A s_B w = N$. We assume that all these parameters are known to $A$ and $B$ before the start of the agreed time interval.
2. During the time interval $A$ randomly chooses $w$ values $p_0, p_1, \ldots, p_{w-1}$, where each value $p_i$ satisfies $0 \leq p_i < s_A$. $A$ then stores the following $w$ sets of $s_B$ bits during the agreed time interval (i.e. a total of $n_A$ bits):

   $$b_{is_As_B+p_is_B}, b_{is_As_B+p_is_B+1}, \ldots, b_{is_As_B+p_is_B+s_B-1}$$

   for $i = 0, 1, \ldots, w-1$.

Similarly, and independently, $B$ randomly chooses $w$ values $q_0, q_1, \ldots, q_{w-1}$, where each value $q_i$ satisfies $0 \leq q_i < s_B$. $B$ then stores the following $w$ sets of $s_A$ bits during the agreed time interval (i.e. a total of $n_B$ bits):

$$b_{is_As_B+q_i}, b_{is_As_B+s_B+q_i}, \ldots, b_{is_As_B+(s_A-1)s_B+q_i}$$

for $i = 0, 1, \ldots, w-1$.

3. At the end of the agreed time interval it should be clear that $A$ and $B$ will share precisely one bit from each range of bits

$$\boldsymbol{b}'_i = (b_{is_As_B}, b_{is_As_B+1}, \ldots, b_{(i+1)s_As_B-1})$$

for $i = 0, 1, \ldots, w-1$. I.e. $A$ and $B$ will share precisely $w$ bits.

4. At the end of the agreed time interval (and not before) $A$ sends $B$ its random values $p_0, p_1, \ldots, p_{w-1}$, used to help select which bits from the public channel it has stored during the time interval. Similarly $B$ sends $A$ its secret random values $q_0, q_1, \ldots, q_{w-1}$.
5. $A$ and $B$ can then both very easily determine which key bits they share.

As in the previous section, before attempting to describe the performance of this scheme, we first consider the best strategy for an eavesdropper wishing to find as many key bits as possible using the minimum amount of storage. There would appear to be three obvious strategies for the eavesdropper, $C$. Firstly $C$ could choose to store a random selection of bits from the channel. Secondly $C$ could store a fixed number of bits from each range, $\boldsymbol{b}'_t$, for $t = 0, 1, \ldots, w-1$. Thirdly, $C$ could select a number of ranges $\boldsymbol{b}'_t$ for various values of $t$, $(0 \leq t < w)$, and store all the bits for the selected ranges.

Now, since there is exactly one key bit within any range $\boldsymbol{b}'_t$, $(0 \leq t < w)$, all strategies would appear to yield similar results. In the following simple result, in which we derive the performance of this revised scheme, we therefore arbitrarily assume that the eavesdropper is using the second of the above strategies.

***Theorem 3.*** *Suppose $A$ and $B$ follow Algorithm M, and an eavesdropper $C$ stores $n_C$ bits selected from the public broadcast channel during the agreed time interval. Then the following will hold.*

(i) *The number of bits of key shared by $A$ and $B$ at the end of the process will be exactly $w = n_An_B/N$.*
(ii) *The expected number of key bits available to $C$ is $wn_C/N = n_An_Bn_C/N^2$.*

*Proof. (i)* This follows from the discussion given as part of Algorithm M.
*(ii)* Consider any range: $\boldsymbol{b}'_t$ (for some value of $t$ satisfying $0 \leq t < w$). At the end of the agreed time interval, $A$ and $B$ will store exactly one bit from this range. The probability that $C$ will store this particular bit is equal to $n_C/N$. Given that there are $w$ such ranges, and given that these probabilities are independent, we see that the expected number of bits held by all of $A$, $B$ and $C$ at the end of the agreed time interval is equal to $wn_C/N$, as required.

This system then achieves a comparable performance to the system in the previous section. The only disadvantage is the slightly increased communication cost in transferring the values $p_i$ and $q_i$ across the authenticated channel. However, this is a very small cost since the number of these values will only be the same as the number of key bits agreed by $A$ and $B$, and each value will only require $\log_2 s_A$ or $\log_2 s_B$ bits of storage.

Moreover this variant of the basic scheme has two significant advantages.

- Unlike the first variant (described in Section 4.1), its security is not dependent on the cryptographic properties of a pseudo-random number generator.
- Unlike both the other schemes, it yields a key of guaranteed length to $A$ and $B$, and not just a varying number of key bits with an associated expected value. The disadvantage of this latter case is that on some occasions the number of key bits provided to $A$ and $B$ may be somewhat less than the expected value, potentially causing problems.

## 5 Summary and conclusions

We have thus described systems which provide secret key agreement between $A$ and $B$ and whose security rests solely on the following two assumptions.

- The cost of storage remains high relative to the bandwidth of one or more publicly available broadcast channels.
- $A$ and $B$ share an error-free authenticated channel.

It is particularly interesting to note that, with the exception of the scheme described in Section 4.1, the systems' security does not depend on any assumptions regarding the computational difficulty of any problems. In that sense the systems are provably secure (given the two key assumptions listed above).

We now briefly consider possible practical circumstances in which the above two assumptions might be satisfied. There are various ways in which the two legitimate parties might be provided with an authenticated channel (but not with the means to securely agree a key). Two of the more likely are as follows.

- The users may purchase a communications facility which provides an authenticated channel as a premium service (which can be obtained simply by paying the appropriate rate). The communications service provider may, for example, provide this by using digital signatures, MACs or some other type of cryptographic check function. It is certainly conceivable that this could be provided in such a way that the users have no access to the keys used, and hence no direct means to exchange secret keys.
- The users may have access to an implementation of a digital signature function such as DSS, which cannot be used for data encryption. They could then use this digital signature function, in conjunction with authenticated keys for each other, to provide the authenticated channel.

We next briefly describe two possible sources for a high bit rate (say greater than 10 Gbit/sec) public broadcast channel.

- The first is to make use of a high rate public satellite data channel. In this case the bits sent over the channel will not be random—instead they will consist of many data streams intermingled. However, in practice they may be 'random enough' for our purposes (especially if a hashing operation is performed on any agreed set of key bits).
- The second is to employ a purpose-designed high speed random data source, the output of which is made publicly available by some means (e.g. by fibre-optic cable). Although this will now guarantee randomness for the bits, there are obvious problems with this approach if it is simultaneously used by many pairs of parties to agree a secret key (as would almost certainly be necessary to justify the cost of providing such a channel). In such an event there are much greater incentives for third parties to invest resources in storing the channel output, since it could yield many secret keys simultaneously.

Finally, it could be argued that, given that $A$ and $B$ share an authenticated channel, then they can achieve secret key agreement by using the well-known Diffie-Hellman key exchange protocol, (see, for example, [1]). Whilst this is certainly true, there may be situations where users do not wish to take such an approach. For example, users may not choose to trust a system whose security depends entirely on a single mathematical function remaining hard to compute (i.e. the discrete logarithm problem), and they may prefer to trust in arguments about the likely cost of data storage. It is certainly of theoretical interest to observe that key agreement schemes can be devised which rely only on the two assumptions listed above, and which do not require any assumptions about the computational difficulty of certain calculations.

## Acknowledgements

The work in this paper is part of the DTI/EPSRC Link Personal Communications Programme project: *Security studies for third generation mobile telecommunications systems (3GS3)*, the colloborating partners in which are Vodafone, GPT and RHUL, and which has been partly funded by the UK EPSRC under Grant Number GR/J17173. The author would like to acknowledge the invaluable advice and encouragement of colleagues in the project.

## References

1. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. on Information Theory **IT-22** (1976) 644–654
2. Maurer, U.M.: Conditionally-perfect secrecy and a provably-secure randomized cipher. J. of Cryptology **5** (1992) 53–66
3. Maurer, U.M.: Secret key agreement by public discussion from common information. IEEE Trans. on Information Theory **39** (1993) 733–742
4. Wyner, A.D.: The wire-tap channel. Bell System Tech. J. **54** (1975) 1355–1387