A formal approach for reasoning about a class of Diffie-Hellman protocols

Rob Delicata and Steve Schneider
{R.Delicata, S.Schneider}@surrey.ac.uk

Department of Computing, University of Surrey, Guildford, GU2 7XH, UK.

Abstract. We present a framework for reasoning about secrecy in a class of Diffie-Hellman protocols. The technique, which shares a conceptual origin with the idea of a rank function, uses the notion of a message-template to determine whether a given value is generable by an intruder in a protocol model. Traditionally, the rich algebraic structure of Diffie-Hellman messages has made it difficult to reason about such protocols using formal, rather than complexity-theoretic, techniques. We describe the approach in the context of the MTI A(0) protocol, and derive the conditions under which this protocol can be considered secure.

1 Introduction

Formal protocol analysis techniques have a simplicity which is due, in part, to the high level of abstraction at which they operate. Such abstractions are justified since any attack discovered at the abstract level will tend to be preserved in a more concrete model. In general, however, failure to discover an attack does not imply correctness, and in seeking to establish correctness we must be mindful of the assumptions on which our abstractions are based.

Protocols based on the Diffie-Hellman scheme [7] present an interesting verification challenge since, in this context, we cannot assume such an abstract view of cryptography. Certain algebraic properties (such as the homomorphism of exponentiation in $(g^x)^y = (g^y)^x$) must be represented for the protocol to reach its functional goal, and other properties (such as the cancellation of multiplicative inverses) must also be considered if we wish to prove a meaningful security result. As a consequence, such protocols have tended to be evaluated in complexity-theoretic models (see [4], for example) which aim to reduce the correctness of the protocol to some well-defined hard problem, such as the computation of discrete logarithms in a finite field. The resulting proofs tend to be difficult to conduct and evaluate, and a small change in the protocol will often require an entirely new proof to be constructed.

With some exceptions [11,12,1] formal techniques have been slow in rising to the challenge of Diffie-Hellman. This paper presents a theorem-proving approach to the verification of a class of Diffie-Hellman protocols. Although our

r_A, r_B, r_C	Random integers, chosen by A, B and C respectively
t_A, t_B	Ephemeral public-keys, $t_A = g^{r_A}$, $t_B = g^{r_B}$
x_A, x_B, x_C	Private long-term keys of A, B and C respectively
y_A, y_B	Public keys of A and B: $y_A = g^{x_A}$, $y_B = g^{x_B}$
Z_{AB}	The shared secret between A and B
$x \in_R X$	An element x chosen at random from the set X

Fig. 1. Protocol notation

approach is quite general we present it in the context of the MTI A(0) protocol of Matsumoto, Takashima and Imai [10]. This protocol is chosen for the simplicity of its messages and non-standard use of Diffie-Hellman (in particular, the computation of a shared key as $g^x \cdot g^y = g^{x+y}$). Some of the MTI protocols satisfy an interesting property — which we call I/O-independence — that enables us to model the protocols at a very abstract level. The protocols and the concept of I/O-independence are described in Section 2. Our model revolves around the idea of a message-template which, suitably instantiated, can represent any value that an intruder can deduce (under a defined set of capabilities). A particular value remains secret if it cannot be realised via any instantiation of the message-template. This model, and its associated definition of secrecy, is described in Section 3 and applied to the MTI A(0) protocol in Section 4. Although we do not describe it in such language, our approach shares a conceptual origin with the notion of a rank function [13], and is informed by the approach of Pereira and Quisquater [12]; we explore these relationships, and conclude, in Section 5.

2 The MTI protocols

Three infinite classes of authenticated key agreement protocols fall under the banner of MTI [10]. All of the MTI protocols appear amenable to analysis in our framework but, in this paper, we focus on one particular protocol, A(0). The protocol combines long-term and ephemeral key contributions to provide authentication in the Diffie-Hellman scheme. A summary of notation, following [3], is given in Figure 1. In protocol A(0) (Figure 2) principal A (who wishes to establish a shared-secret with B) generates a long-term secret, x_A, and publishes the corresponding public-key $y_A = g^{x_A}$. B does the same with x_B and y_B. A randomly chooses r_A, computes $z_A = g^{r_A}$ and sends it to B. In response, B randomly chooses r_B, computes $z_B = g^{r_B}$ and sends it to A. B then computes $Z_{AB} = z_A^{x_B} y_A^{r_B} = (g^{r_A})^{x_B} \cdot (g^{x_A})^{r_B} = g^{r_A x_B + x_A r_B}$ and A computes $Z_{AB} = g^{r_B x_A + x_B r_A}$. The protocol aims to convince each principal that no one, aside from the other

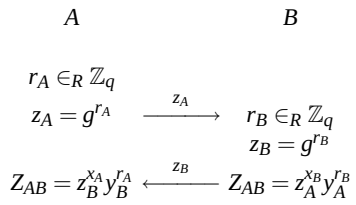

Fig. 2. MTI A(0) protocol

protocol participant, can learn the shared-secret Z_{AB}. This property is often termed *implicit key authentication*; here we simply refer to it as *secrecy*.

All of the MTI protocols involve the exchange of two messages, z_A and z_B, each of which is computed within the principal and not as a function of a previously received message. (Contrast this with protocols like Cliques, where a principal B may receive an input m from A, apply some function to m and send the result on to C.) We capture this notion in the property of Input/Output-independence:

Definition 1. *In a Diffie-Hellman protocol a principal P is I/O-independent if P does not transmit any message which is dependent on the value of a previously received message.*

We say that a protocol is I/O-independent if every honest principal is I/O-independent.

Proposition 1. *Protocol A(0) is I/O-independent.*

We will see in the next section that the property of I/O-independence enables us to model protocols at a very abstract level.

3 A model for I/O-independent Diffie-Hellman protocols

In this section we present a model for I/O-independent protocols based around the idea of a *message-template* which defines the general form of any message generable by an intruder in a given protocol.

We begin by noting that transmitted messages are elements of some group G in which the Decisional Diffie-Hellman problem is believed to be hard. A generator g of G is agreed by all principals and there exists an identity element 1 such that $1 \cdot x = 1$, for all $x \in G$. We assume that elements of G can be expressed as g raised to the power of a sum of products of random numbers. This assumption permits, for example, g^{xy+z}, where x, y and z are random numbers,

but excludes values such as $g^{(g^x)}$ since the exponent is itself a group element. The users of the system therefore manipulate two types of element, (i) random exponents, and (ii) powers of g, and we assume that only the latter will be sent on the network.

3.1 The intruder

We divide the users of the system into a set of honest principals, $\{A, B\}$, who will always adhere to the protocol, and a malevolent intruder, C, whose goal is to subvert the protocol.

Some elements of (i) (from above) will be known initially to the intruder (such as random numbers he has chosen himself), and some elements of (ii) will become known to the intruder during the course of the protocol. The I/O-independent nature of the protocols means that an active intruder cannot influence any of the values sent by honest participants, since the functions which produce these values are not dependent on any external input. This is important, since it is then sufficient to assume that the intruder knows these values from the start.

Following [12], we divide the intruder's initial knowledge into a set E of exponents and a set P of known powers of g, where $x \in P$ indicates knowledge of g^x but not of x (unless $x \in E$). We then define the computations that the intruder can perform

Definition 2 (intruder capabilities). *Given a set P of initially known powers of g and a set E of initially known exponents, the intruder can* grow *P based on the following operations:*
1. given $m_1 \in P$ and $m_2 \in P$ add $m_1 + m_2$ to P
2. given $m \in P$ and $n \in E$ add mn, $m(n^{-1})$ to P
3. given $m \in P$ add $-m$ to P

In other words, we allow the intruder to (1) compute $g^{m_1} \cdot g^{m_1} = g^{m_1+m_2}$ given knowledge of g^{m_1} and g^{m_2}, (2) compute the exponentiations $(g^m)^n$, $(g^m)^{n^{-1}}$ given knowledge of g^m and n, and (3) compute the inverse $\frac{1}{g^m} = g^{-m}$ given g^m. Moreover, these capabilities can be combined:

Example 1. Suppose that $P = \{1, r_A\}$ and $E = \{r_C\}$. The intruder can deduce (i) $-r_A \in P$ by rule 3 from r_A, (ii) $1r_C \in P$ by rule 2 and $-r_A + 1r_C \in P$ by rule 1 from (i) and (ii), representing the computation of $g^{r_C - r_A}$.

Crucially, the intruder is not able to use $m_1 \in P$ and $m_2 \in P$ to deduce $m_1 m_2$.

In this model, the intruder's entire knowledge can be defined as the closure of P under the deductions of Definition 2 and set E. In any useful protocol, E and

P will initially be non-empty, and the resulting knowledge sets will be infinite. For this reason, it will be infeasible to enumerate these sets by *growing* P via successive application of rules 1–3.

3.2 System definition

An examination of the sorts of values that can be deduced by an intruder leads to the following observation: a generable value can be written as some number of elements of P multiplied by some product of (possibly inverted) elements from E. For instance, the value derived in Example 1 can be written as $-1(r_A)(r_C^0)+1(1)(r_C^1)$ (noting the difference between the group identity 1 and the integer 1). In fact, we can go further by defining a polynomial over the variables of E and P which represents any value generable by the intruder using rules 1–3, above.

Definition 3. *Let F be a finite family of functions that map elements of E to integer powers: $F \subseteq_{fin} E \to \mathbb{Z}$.*

Given $E = \{x_C\}$, for example, we may define $F = \{\{x_C \mapsto -1\}\}$.

Definition 4. *Let h be a higher-order function which, for a member of F, maps elements of P to integers:$h : F \to (P \to \mathbb{Z})$.*

As an example, given $P = \{r_A\}$ and $F = \{\{x_C \mapsto -1\}\}$, we might choose to define $h(\{x_C \mapsto -1\}) = \{r_A \mapsto 1\}$.

Definition 5 (message-template). *Fix some E and P. Then:*

$$v(F,h) = \sum_{f \in F} \left(\sum_{p \in P} h_{f,p} \cdot p \right) \left(\prod_{e \in E} e^{f_e} \right)$$

We call v the *message-template* for a system defined by E and P. Intuition is little help here, so consider a simple example:

Example 2. Given the system defined by $P = \{r_A\}$ and $E = \{x_C\}$, consider how the value $g^{-r_A x_C^{-1} + 5 r_A x_C}$ can be expressed. P and E result in the following polynomial:

$$v(F,h) = \sum_{f \in F} \left(h_{f,r_A} \cdot x_C^{f_{x_C}} \right)$$

To express a particular generable value we must define F and h. Recall that F is a family of functions. Suppose that $F = \{\{x_C \mapsto -1\}, \{x_C \mapsto 0\}, \{x_C \mapsto 1\}\}$, then we have:

$$v(F,h) = (h_{\{x_C \mapsto -1\},r_A} \cdot x_C^{-1}) + (h_{\{x_C \mapsto 0\},r_A} \cdot x_C^0) + (h_{\{x_C \mapsto 1\},r_A} \cdot x_C^1)$$

Finally, suppose that h is defined such that $h(\{x_C \mapsto -1\}) = \{r_A \mapsto 1\}$, $h(\{x_C \mapsto 0\}) = \{r_A \mapsto 0\}$ and $h(\{x_C \mapsto 1\}) = \{r_A \mapsto 5\}$. This results in:

$$v(F,h) = (-1 \cdot r_A) \cdot (x_C^{-1}) + (0 \cdot r_A) \cdot (x_C^0) + (5 \cdot r_A)(x_C^1)$$

which is the value $-r_A x_C^{-1} + 5 r_A x_C$.

As a more complex example, consider the following:

Example 3. Let $P = \{1, r_A, r_B\}$, $E = \{x_C, r_C\}$. Then:

$$v(F,h) = \sum_{f \in F} (h_{f,1} \cdot 1 + h_{f,r_A} \cdot r_A + h_{f,r_B} \cdot r_B) \left(x_C^{f_{x_C}} \cdot r_C^{f_{r_C}} \right)$$

In this polynomial, the value $g^{r_C - r_A}$ from Example 1 can be represented by defining:

$$F = \{\{x_C \mapsto 0, r_C \mapsto 0\}, \{x_C \mapsto 0, r_C \mapsto 1\}\}$$

and h such that:

$$h(\{x_C \mapsto 0, r_C \mapsto 0\}) = \{1 \mapsto 0, r_A \mapsto -1, r_B \mapsto 0\}$$
$$h(\{x_C \mapsto 0, r_C \mapsto 1\}) = \{1 \mapsto 1, r_A \mapsto 0, r_B \mapsto 0\}$$

We then obtain: $v(F,h) = (0 \cdot 1 + -1 \cdot r_A + 0 \cdot r_B)(x_C^0 \cdot r_C^0) + (1 \cdot 1 + 0 \cdot r_A + 0 \cdot r_B)(x_C^0 \cdot r_C^1) = -r_A + r_C$.

As stated, our intention is that, for a given system (defined by E and P), the polynomial $v(F,h)$ expresses the general form of all values deducible by an intruder, from P and E, by appeal to the deduction rules of Definition 2. We embed the ability of a polynomial to take a certain value in the concept of realisability:

Definition 6. *A value m is realisable (written $realisable(m)$) if there exists functions F and h such that $v(F,h) = m$.*

That is, a value m is realisable if there exists a solution to the equation $v(F,h) - m = 0$. If m is not realisable we write $\neg realisable(m)$. Define *Pub* to be a closure containing all possible polynomials for a given system. *Pub* is the set containing all realisable values of that system: the set of *public* messages.

Theorem 1 (Faithfulness). *Fix some P and E and Pub as defined above. Pub is closed under the deductions of Definition 2.*

Proof. By induction. For the base case we show that, whenever $p \in P$, p is realisable.

Base case: Given some $p \in P$, p is realisable with $v(F,h)$ by defining

$$F = \{\{e \mapsto 0 \mid e \in E\}\}$$

and:

$$h(\{e \mapsto 0 \mid e \in E\}) = \{p \mapsto 1\} \cup \{q \mapsto 0 \mid q \in P \setminus \{p\}\}$$

Inductive step: There are three cases, corresponding to the three intruder deduction rules: (i) $realisable(m_1) \wedge realisable(m_2) \implies realisable(m_1 + m_2)$, (ii) $realisable(m_1) \wedge n \in E \implies realisable(m_1 n) \wedge realisable(m_1 n^{-1})$, and (iii) $realisable(m_1) \implies realisable(-m_1)$.

(i) Assume $m_1 = v(F_1, h_1)$ and $m_2 = v(F_2, h_2)$. Then $m_1 + m_2$ is realisable with $v(F_3, h_3)$ by defining $F_3 = F_1 \cup F_2$ and h such that:

$$h_3(f) = \begin{cases} h_1(f) & \text{if } f \in dom(h_1) \setminus dom(h_2) \\ h_2(f) & \text{if } f \in dom(h_2) \setminus dom(h_1) \\ \lambda p.h_1(f)(p) + h_2(f)(p) & \text{if } f \in dom(h_1) \cap dom(h_2) \end{cases}$$

(ii) For the first conjunct assume $m_1 = v(F_1, h_1)$ and $n \in E$. Then, $m_1 n$ is realisable with $v(F_2, h_2)$ by defining:

$$F_2 = \{f \oplus \{n \mapsto (F_1(n) + 1)\} \mid f \in F_1\}$$

and h_2 such that:

$$h_2(f) = h_1(f \oplus \{n \mapsto (f(n) - 1)\})$$

The second conjunct follows the above, with addition in place of the subtraction in the definition of h_2.

(iii) Assume $m_1 = v(F_1, h_1)$. Then $-m_1$ is realisable with $v(F_1, h_2)$ where h_2 is defined such that $h_2(f)(p) = -(h_1(f)(p))$.

□

Our intention is for the model to respect the fact that some values are impossible for an intruder to guess. We achieve this by assuming that the variables (r_A, x_C etc.) are *symbolic*, that each is distinct from all others, and that the set of variables is disjoint from the set of integers.

Assumption 1 $(P \cup E) \cap \mathbb{Z} = \emptyset$

The following example makes clear why this restriction is necessary:

Example 4. Consider the system defined by $P = \{1\}$ and$E = \{x_C\}$. If variables are numbers, then any group value g^X can be realised by defining $X = v(F,h)$, where $F = \{\{x_C \mapsto 0\}\}$ and $h(\{x_C \mapsto 0\}) = \{1 \mapsto X\}$, yielding $v(F,h) = (1 \cdot X)x_C^0 = X$.

Assumption 1 means that, for the group identity 1, we have that $1 \notin \mathbb{Z}$ and, in particular, $1 \neq 1$. However, we grant special privileges to the group identity such that $1 \cdot m = m$, for all m. Note that an element $n \in E \setminus P$ will typically only be realisable if $1 \in P$. That is, n is realisable by $v(F,h)$, where $F = \{\{n \mapsto 1\}\}$ and $h(\{n \mapsto 1\}) = \{1 \mapsto 1\}$, giving $1 \cdot (1 \cdot n^1) = n$.

Condition 1 $1 \in P \implies P \cap E = \emptyset$

We require that the above condition be true of any protocol model. To see why this is necessary consider the system given by $E = \{x_C\}$, $P = \{1, x_C\}$. The value x_C can be realised in two ways, $x_C = v(F,h_1) = v(F,h_2)$, where $F = \{\{x_C \mapsto 0\}, \{x_C \mapsto 1\}\}$, and h_1, h_2 are defined such that:

- $h_1(\{x_C \mapsto 0\}) = \{1 \mapsto 0, x_C \mapsto 1\}$, $h_1(\{x_C \mapsto 1\}) = \{1 \mapsto 0, x_C \mapsto 0\}$
- $h_2(\{x_C \mapsto 0\}) = \{1 \mapsto 0, x_C \mapsto 0\}$, $h_2(\{x_C \mapsto 1\}) = \{1 \mapsto 1, x_C \mapsto 0\}$

The first case yields $v(F,h_1) = (x_C)x_C^0 + (0)x_C^1 = x_C$ and the second results in $v(F,h_2) = (0)x_C^0 + (1)x_C^1 = x_C$. Since $h_1 \neq h_2$, but $v(F,h_1) = v(F,h_2)$, the example allows the same value to be derived in two separate ways.

3.3 Secrecy

In a Diffie-Hellman protocol, a principal u performs some key computation function on an input z to derive a secret Z_{uv} believed to be shared with v. We denote this function k_{uv} with $Z_{uv} = k_{uv}(z)$.

Example 5. In the standard Diffie-Hellman protocol [7], a principal A, apparently running with B and using the ephemeral secret x_A performs the key computation $k_{AB}(z) = zx_A$representing the shared secret $Z_{AB} = g^{zx_A}$.

Definition 7 (Secrecy). *Given a system defined by E and P, a key computation function k maintains secrecy iff:*

$$\forall m.realisable(m) \implies \neg realisable(k(m))$$

Intuitively, secrecy is defined as an anti-closure property of the set of generable values: the result of applying k to a realisable value should never result in a realisable value. If this property does not hold then an intruder will possess two values, x and y, such that, if x is sent to some principal she will compute y, wrongly believing it to be secret.

4 Reasoning about the MTI A(0) protocol

A complete model of an I/O-independent protocol is a combination of the message-template with an appropriate key computation function. In this section we present a model of the MTI A(0) protocol and use it to deduce the conditions under which the protocol guarantees the secrecy of a shared key.

Define $E^{A(0)} = \{r_C, x_C\}$, $P^{A(0)} = \{1, r_A, r_B, x_A, x_B\}$, representing a run of the MTI A(0) protocol. We wish to show that the key computation function $k_{ab}^{A(0)}(z) = zx_a + x_b r_a$ maintains secrecy. There are eight cases to consider:

1. $a = A \wedge b = C$	5. $a = A \wedge b = A$
2. $a = B \wedge b = C$	6. $a = B \wedge b = B$
3. $a = C \wedge b = A$	7. $a = A \wedge b = B$
4. $a = C \wedge b = B$	8. $a = B \wedge b = A$

We treat each in turn.

Cases 1–4

Let $a = A$ and $b = C$. We are trying to show that, for any z where $realisable(z)$, $\neg realisable(k_{AC}^{A(0)}(z))$. There exists some F_1 and h_1 such that $v(F_1, h_1) = z$. If we can find some F_2 and h_2 such that $v(F_2, h_2) = k_{AC}^{A(0)}(z)$ we will have shown that $k_{AC}^{A(0)}(z)$ is realisable and is therefore, not secret.

Note that $k_{AC}^{A(0)}(z) = zx_A + x_C r_A$ is a linear combination, and that the linear combination will be realisable if each of its components is realisable. In general zx_A will be realisable if z does not mention x_A (since $x_A \in P$ but $x_A \notin E$). Consider, then, $z = r_C$, given by $v(F_1, h_1)$ where:

$$F_1 = \{\{r_C \mapsto 1\}\}$$
$$h_1(\{r_C \mapsto 1\}) = \{1 \mapsto 1\} \cup \{p \mapsto 0 \mid p \in P \setminus \{1\}\}$$

then $zx_A = r_C x_A$ is realisable by $v(F_1, h_3)$ where $h_3(\{r_C \mapsto 1\}) = \{x_A \mapsto 1\}$. Similarly, $x_C r_A$ is realisable by $v(F_3, h_4)$, where:

$$F_3 = \{\{x_C \mapsto 1\}\}$$
$$h_4(\{x_C \mapsto 1\}) = \{r_A \mapsto 1\} \cup \{p \mapsto 0 \mid p \in P \setminus \{r_A\}\}$$

Theorem 1 then tells us that, since $realisable(r_C x_A)$ and $realisable(x_C r_A)$, the sum $r_C x_A + x_C r_A$ is also realisable, and is given by $v(F_2, h_2)$, where:

$$F_2 = F_1 \cup F_3 = \{\{r_C \mapsto 1\}, \{x_C \mapsto 1\}\}$$
$$h_2(\{r_C \mapsto 1\}) = \{x_A \mapsto 1\} \cup \{p \mapsto 0 \mid p \in P \setminus \{x_A\}\}$$
$$h_2(\{x_C \mapsto 1\}) = \{r_A \mapsto 1\} \cup \{p \mapsto 0 \mid p \in P \setminus \{r_A\}\}$$

From this we conclude that the intruder can deduce a pair of values, r_C and $r_C x_A + x_C r_A$, related by the key computation function $k_{AC}^{A(0)}$, and so secrecy fails. This failure should come as no surprise since $b = C$ represents the intruder's legitimate participation in the protocol. Any honest principal who willingly engages in a protocol run with the intruder cannot hope to maintain secrecy of the resulting session-key. We note that similar conclusions can be reached in cases 2–4.

Cases 5 **and** 6 **(**$b = a$**)**

Let $a = A$, $b = A$. The corresponding key computation is given by $k_{AA}^{A(0)}(z) = zx_A + x_A r_A$. Note that $x_A r_A$ is the multiplication of two elements from P. The intruder model only allows the addition of elements from P and, since $x_A \notin E$ and $r_A \notin E$, the component $x_A r_A$ is unrealisable. Consequently, for $zx_A + x_A r_A$ to be realisable, zx_A must be a linear combination that includes $-x_A r_A$ (since $-x_A r_A + x_A r_A = 0$ is realisable). Consider the simplest case, where $z = -r_A$, which is realisable, since $r_A \in P$. The result of $k_{AA}^{A(0)}(-r_A) = -r_A x_A + x_A r_A = 0$ is realisable by $v(F_5, h_5)$, where, for instance:

$$\begin{aligned} F_5 &= \{\{r_C \mapsto 0\}, \{x_C \mapsto 0\}\} \\ h_5(\{r_C \mapsto 0\}) &= \{p \mapsto 0 \mid p \in P\} \\ h_5(\{r_C \mapsto 0\}) &= \{p \mapsto 0 \mid p \in P\} \end{aligned}$$

As a result, the intruder can deduce a pair of values $-r_A$ and 0 such that $0 = k_{AA}^{A(0)}(-r_A)$ and, again, secrecy fails. A similar result holds for case 6, where $a = b = B$. This attack is a simpler version of one discovered by Just and Vaudenay [9] and described by Boyd and Mathuria [3]. In the original attack, z was set to be $r_C - r_A$ and the resulting session-key computed as $g^{x_A r_C}$ (where $x_A r_C$ is realisable). The attack depends on the willingness of A to engage in the protocol with someone claiming her identity, and can be seen as stipulating a condition on an implementation: namely, that a principal should only engage in the protocol if the other party has a distinct identity.

Cases 7 **and** 8 **(**$b \neq a$**)**

For the final cases, assume $a = A$ and $b = B$ (a similar result holds for $a = B$ and $b = A$). The key computation is given by $k_{AB}^{A(0)}(z) = zx_A + x_B r_A$. For secrecy to fail there must exist some $z = v(F_1, h_1)$ and $k_{AB}^{A(0)}(z) = v(F_2, h_2)$ such that:

$$v(F_1, h_1) \cdot x_A + x_B r_A = v(F_2, h_2)$$

Consider the coefficient of $x_C^0 r_C^0$. We have:

$$h_2(\{x_C \mapsto 0, r_C \mapsto 0\}) = \{1 \mapsto n_1, r_A \mapsto n_2, r_B \mapsto n_3, x_A \mapsto n_4, x_B \mapsto n_5\}$$
$$h_1(\{x_C \mapsto 0, r_C \mapsto 0\}) = \{1 \mapsto m_1, r_A \mapsto m_2, r_B \mapsto m_3, x_A \mapsto m_4, x_B \mapsto m_5\}$$

for some $m_1 \ldots m_5 \in \mathbb{Z}$, $n_1 \ldots n_5 \in \mathbb{Z}$ where the coefficients on both sides are the same:

$$m_1 x_A + m_2 r_A x_A + m_3 r_B x_A + m_4 x_A^2 + m_5 x_B x_A + x_B r_A$$
$$=$$
$$n_1 + n_2 r_A + n_3 r_B + n_4 x_A + n_5 x_B$$

By assumption we have that variables are symbolic and that a given symbol x is distinct from all others. Specifically, we note that $x_B r_A$ is distinct from all other terms on either side of the equation and, therefore, there are no values of the coefficients which enable the equality to be met. We conclude that, for any realisable z, $k_{AB}^{A(0)}(z)$ is unrealisable.

Results

The analysis enables us to state the following result:

Theorem 2. *Given* $E^{A(0)} = \{r_C, x_C\}$, $P^{A(0)} = \{1, r_A, r_B, x_A, x_B\}$,

$$a \neq C \wedge b \neq C \wedge a \neq b \implies k_{ab}^{A(0)} \textit{ maintains secrecy}$$

□

This tells us that protocol A(0) maintains the secrecy of the session-key precisely when the initiator and responder are distinct entities and neither of them is the intruder C.

5 Discussion

5.1 The link with rank functions

Although we have not described our approach in such terms, it shares a conceptual origin with the notion of a *rank function*. In the context of protocol verification, a rank function describes an invariant property of a system [13]. This property will define the sorts of messages that may pass through the system, crucially distinguishing certain values that should remain secret. The rank function effectively partitions the message-space of a protocol by assigning a rank of pub to public and sec to secret messages. Traditionally a rank function

is defined over the message-space of a protocol model expressed in the process algebra CSP [14], and a central rank theorem gives a series of proof obligations on the rank function whose achievement allows us to conclude that only messages of rank pub ever appear on the network. Previous work has applied the rank function approach in the context of Diffie-Hellman protocols [6]. However, a fundamental difficulty with this approach is the necessity to statically assign a rank to messages. It is interesting to note that the present work side-steps this issue by defining (via the message-template) the set *Pub* of public messages. This set corresponds to the set of messages assigned a rank of pub by the rank approach.[1]

5.2 Pereira and Quisquater's approach

Recently, Pereira and Quisquater [12] developed a formal model of the Cliques conference key agreement protocols [2], based on linear logic, and discovered attacks on each of the claimed security properties. In the model, secrecy is defined as the inability of an intruder to discover a pair of values (g^x, g^y) such that, if a principal is sent g^x, he will compute the key g^y. Values are assumed to take the form of g raised to a product of exponents, and secrecy becomes the inability of an intruder to learn a pair of messages separated by the ratio $\frac{y}{x}$. The model allows the intruder to *grow* a set of known ratios, in the hope that some secret ratio(s) remain unobtainable. This ratio-centric view of secrecy seems particularly natural for Diffie-Hellman exchanges, and our initial attempts at modelling the MTI protocols sought to embrace this approach. However, it turns out that this view of secrecy does not generalise in the obvious way. Consider, for example, a value z in the A(0) protocol, and the key computation function $k_{ab}^{A(0)}(z) = zx_a + x_b r_a$. The ratio between $k_{ab}^{A(0)}(z)$ and z — $x_a + \frac{x_b r_a}{z}$ — is still in terms of z, due to the presence of addition in the exponents. This fact makes it difficult to derive the set of secret ratios, since a ratio cannot be stated without recourse to the argument to the key computation function. The present work can be viewed as an attempt to provide a more general view of Diffie-Hellman key computation.

In a different respect, Pereira's and Quisquater's model is more general than ours, since it applies to protocols which fail to satisfy the property of I/O-independence. This property, recall, tells us that no user of the protocol ever sends out any message which is dependent on a previously received message. In the Cliques protocols, protocol participants tend to receive a message, perform some computation on that message and send out the result. Pereira and

[1] In fact, *Pub* is similar to Heather's concept of a minimal rank function [8].

Quisquater call such user operations *services*.[2] These services are encoded in terms of the values added to the exponent of an incoming message. For instance, a principal may receive a message g^x and generate and send the message g^{xyz} (where y and z are known to that principal). The intruder can then (with some restrictions) use the principal as an oracle, enabling him to send a spurious message g^c and receive g^{cyz} in return. The fact that the property of I/O-independence does not allow such services to be expressed in our model is not a fundamental limitation but a restriction which enables us to describe our work in a clean manner. One could envisage weakening this assumption by internalising such services in the intruder (in the style of Broadfoot and Roscoe[5]) where, for example, the multiplication of a value with yz is encoded as an additional intruder deduction. The message-template would need to be redesigned to account for these additional capabilities. In contrast to the present work, such a message-template would tend to be protocol specific.

5.3 Conclusion and further work

We have presented a framework for reasoning about secrecy in a class of Diffie-Hellman protocols, and demonstrated the approach by a consideration of secrecy in the MTI A(0) protocol. The work hinges around the idea of a message-template, an object which defines, in a highly abstract way, the values that can be deduced by an intruder under a given set of capabilities. A protocol model is given as a combination of a message-template and a function representing the key computation applied by a principal to derive a shared secret.

This work is nascent, but we are currently applying it to other protocols, both within and without the MTI suite. This requires us to relax the condition of I/O-independence and widen our model to address situations in which protocol participants provide *services*. In many cases, this extension appears straightforward. The *ad hoc* nature of the secrecy proof in Section 4 is unfortunate, and it would be useful to derive a general framework for such proof (as is achieved in [12], for instance). There also appears to be interesting links between the idea of a message-template and the concept of *ideal* used within the strand space approach [15]. Future work will investigate whether this correspondence enables us to deduce general principles with which a protocol can be proven correct.

Acknowledgements

Thanks to David Pitt, Joshua Guttman and James Heather for interesting discussions on this work and to the anonymous referees for their careful reviewing.

[2] In these terms, a principal is I/O-independent if it provides no services.

References

1. Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. In *31st International Colloquium on Automata, Languages and Programming: ICALP'04*, volume 3142 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.
2. Giuseppe Ateniese, Michael Steiner, and Gene Tsudik. Authenticated group key agreement and friends. In *Proceedings of the 5th ACM Conference on Computer and Communication Security*. ACM Press, 2000.
3. Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Springer-Verlag, 2003.
4. Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Provably authenticated group Diffie-Hellman key exchange — the dynamic case. In *Advances in Cryptology: Proceedings of ASIACRYPT '01*, volume 2248 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.
5. Philippa Broadfoot and A. W. Roscoe. Internalising agents in CSP protocol models. In *Workshop on Issues in the Theory of Security: WITS '02*, 2002.
6. Rob Delicata and Steve Schneider. Temporal rank functions for forward secrecy. In *Proceedings of the 18th Computer Security Foundations Workshop: CSFW-18*. IEEE Computer Society Press, 2005.
7. Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6), 1976.
8. James Heather. 'Oh! ... Is it *really* you?' *using rank functions to verify authentication protocols*. *Ph.D Thesis, Royal Holloway, University of London*, 2001.
9. Mike Just and Serge Vaudenay. Authenticated multi-party key agreement. In *Advances in Cryptology: Proceedings of ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.
10. Tsutomu Matsumoto, Youichi Takashima, and Hideki Imai. On seeking smart public-key-distribution systems. *Transactions of the IECE of Japan*, E69(2), 1986.
11. Catherine Meadows. Extending formal cryptographic protocol analysis techniques for group protocols and low-level cryptographic primitives. In *Workshop on Issues in the Theory of Security: WITS '00*, 2000.
12. Olivier Pereira and Jean-Jacques Quisquater. Security analysis of the Cliques protocols suites. In *Proceedings of the 14th IEEE Computer Security Foundations Workshop: CSFW-14*. IEEE Computer Society Press, 2001.
13. Steve Schneider. Verifying authentication protocols with CSP. In *Proceedings of The 10th Computer Security Foundations Workshop: CSFW-10*. IEEE Computer Society Press, 1997.
14. Steve Schneider. *Concurrent and Real-time Systems: The CSP Approach*. John Wiley and Sons, Ltd, 2000.
15. F. Javier Thayer Fábrega, Jonathan Herzog, and Joshua Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(2/3), 1999.