International Journal of Information Security manuscript No.
(will be inserted by the editor)

Rob Delicata · **Steve Schneider**

An algebraic approach to the verification of a class of Diffie-Hellman protocols

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 protocols, and derive conditions under which each protocol in the suite can be considered secure.

Keywords Protocol Verification · Diffie-Hellman

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 [11] 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 [5], 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 [16,19,18,1] 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.[1] Although our approach is quite general we begin by presenting it in the context of the MTI A(0) protocol of Matsumoto, Takashima and Imai [15]. 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 $g^x \cdot g^y = g^{x+y}$). The design of the MTI protocols is such that we are able to model them at a very abstract level. The class A protocols are described in Section 2. Our model revolves around the idea of a message-template which, suitably instantiated, can represent any value that an attacker 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, applied to the MTI A(0) protocol in Section 4, and the remaining protocols in Section 5. Although we do not describe it in such language, our approach shares a conceptual origin with the notion of a rank function [20], and is informed by the approach of Pereira and Quisquater [19]; we explore these relationships, and conclude, in Section 6.

2 The MTI protocols

Three infinite classes of authenticated key agreement protocols fall under the banner of MTI [15]. All of the MTI protocols are amenable to analysis in our framework but, in the first instance, 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 [4], is given in

Department of Computing, University of Surrey, Guildford, Surrey, GU2 7XH, UK.
Tel.: +44 (0)1483 300 800
E-mail: {R.Delicata, S.Schneider}@surrey.ac.uk

[1] A preliminary version of this work appeared as [9].

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

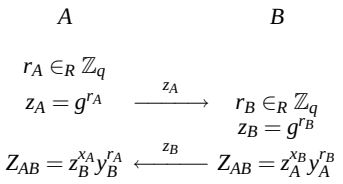

Fig. 2 MTI A(0) protocol

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 protocol participant, can learn the shared-secret Z_{AB}. This property is often termed *implicit key authentication*:

Definition 21 *Let P be a 2-party key agreement protocol involving principals A and B, and let k be a secret jointly generated as a result of P. We say that P provides* implicit key authentication *if A and B are assured that no principal $C \notin \{A, B\}$ can learn k (unless aided by a dishonest $D \in \{A, B\}$).*

In line with previous work [19,8] we formalise implicit key authentication as the inability of an attacker to learn a shared secret.

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 [3], where a principal B may receive an input m from A, apply some function to m and send the result on to C.) We will see in the next section that this fact enables us to model protocols at a very abstract level.

3 A model for Diffie-Hellman protocols

In this section we present a model for Diffie-Hellman protocols based around the idea of a *message-template* which defines the general form of any message generable by an attacker 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:

Assumption 31 *All values passing on the network are powers of g where the exponent can be expressed as a sum of products of integers, and computation takes place in a group in which the Diffie-Hellman problem is hard.*

In the protocol, principals make use of public-key certificates, such as g^{x_A}, but we do not specify how such certificates are registered and obtained. Instead, we assume the following:

Assumption 32 *There exists a certification authority, or some other means by which a principal A can obtain B's public-key certificate and be sure that B knows the corresponding private-key.*

Finally, we divide the principals into disjoint sets of honest and dishonest agents, such that:

Assumption 33 *Honest principals do not deviate from the protocol and do not (knowingly) divulge their secret keys or any previously established session-keys.*

In fact, we assume the presence of a single dishonest principal: the *attacker*.

3.1 The attacker

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 agent, C, whose goal is to subvert the protocol.

Some elements of (i) and (ii) (from above) will be known initially to the attacker (such as random numbers he has chosen himself, and their corresponding powers), and some elements of (ii) will become known to the attacker during the course of the protocol. The design of the MTI protocols means that an active attacker cannot influence any of the values sent by honest participants, since the functions which produce these values are not dependent on any external input (they may, however, be influenced by the perceived sender of the incoming message). This is important, since it is then sufficient to assume that the attacker knows these values from the start.

Following [19], we divide the attacker'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 attacker can perform:

Definition 31 (attacker capabilities) *Given a set P of initially known powers of g and a set E of initially known exponents, the attacker 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 attacker 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 attacker 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 attacker is not able to use $m_1 \in P$ and $m_2 \in P$ to deduce $m_1 m_2$. ■

In this model, the attacker's entire knowledge can be defined as the closure of P under the deductions of Definition 31 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 attacker 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 attacker using rules 1–3, above.

Definition 32 *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 33 *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 34 (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. Our goal is to find F and h such that:

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

We begin by rewriting the right-hand side of the equation to include all multipliers of P elements and powers of E elements:

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

The term is a linear combination of two components, each of which makes reference to a different power of x_C (-1 and 1). This guides us to a definition of F, as:

$$F = \{\{x_C \mapsto -1\}, \{x_C \mapsto 1\}\}$$

In the first component of the linear combination, the -1st power of x_C is multiplied by $-1 \cdot r_A$, leading us to:

$$h(\{x_C \mapsto -1\}) = \{r_A \mapsto -1\}$$

In the second component, the 1st power of x_C is multiplied by $5 \cdot r_A$, leading to:

$$h(\{x_C \mapsto 1\}) = \{r_A \mapsto 5\}$$

So, $v(F,h) = -r_A x_C^{-1} + 5 r_A x_C$ precisely when $F = \{\{x_C \mapsto -1\}, \{x_C \mapsto 1\}\}$ and h is defined such that $h(\{x_C \mapsto -1\}) = \{r_A \mapsto 1\}$ and $h(\{x_C \mapsto 1\}) = \{r_A \mapsto 5\}$. ■

As a more complex example, consider the following:

Example 3 Let $P = \{1, r_A, r_B\}$, $E = \{x_C, r_C\}$. Expanding the definition of $v(F,h)$ in terms of P and E results in:

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

Taking a different approach this time, consider how the value $g^{r_C - r_A}$ (from Example 1) can be represented. We are looking for F and h such that:

$$\sum_{f \in F} \left(h_{f,1} \cdot 1 + h_{f,r_A} \cdot r_A + h_{f,r_B} \cdot r_B \right) \left(x_C^{f_{x_C}} \cdot r_C^{f_{r_C}} \right) = (-1 \cdot r_A) \cdot (r_C^0) + (1 \cdot 1) \cdot (r_C^1)$$

Note that, on the left-hand side of the equation, each component is in terms of the product of some powers of x_C and r_C. Since x_C does not appear in the target value $r_C - r_A$, this leads us to:

$$\sum_{f \in F} \left(h_{f,1} \cdot 1 + h_{f,r_A} \cdot r_A + h_{f,r_B} \cdot r_B\right) \left(x_C^{f_{X_C}} \cdot r_C^{f_{r_C}}\right) = (-1 \cdot r_A) \cdot (x_C^0 \cdot r_C^0) + (1 \cdot 1) \cdot (x_C^0 \cdot r_C^1)$$

This, in turn leads us to a definition of F:

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

We also note that $r_B \in P$ does not appear in $r_C - r_A$. Expanding the right-hand side in terms of P we get:

$$\sum_{f \in F} \left(h_{f,1} \cdot 1 + h_{f,r_A} \cdot r_A + h_{f,r_B} \cdot r_B\right) \left(x_C^{f_{X_C}} \cdot r_C^{f_{r_C}}\right) = (0 \cdot 1 + -1 \cdot r_A + 0 \cdot r_B) \cdot (x_C^0 \cdot r_C^0) + (1 \cdot 1 + 0 \cdot r_A + 0 \cdot r_B) \cdot (x_C^0 \cdot r_C^1)$$

Finally, the coefficients of each P term in the above lead us to define 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: $v(F,h) = (0 \cdot 1 + -1 \cdot r_A + 0 \cdot r_B)(x_C^0 \cdot r_C^0) + (1 \cdot 1 + 0 \cdot r_A + 0 \cdot r_B)(x_C^0 \cdot r_C^1) = -r_A + r_C$. ■

Note that, in the definition of the message-template, h is defined in terms of F. Therefore, any values $v(F_1,h_1)$ and $v(F_2,h_2)$ exhibit the following property:

Lemma 35 *Given $v(F_1,h_1)$ and $v(F_2,h_2)$, we have that:*

$$F_1 \neq F_2 \Longrightarrow h_1 \neq h_2 \qquad \square$$

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 attacker, from P and E, by appeal to the deduction rules of Definition 31. We embed the ability of a polynomial to take a certain value in the concept of realisability:

Definition 36 *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 37 (Faithfulness) *Fix some P and E and Pub as defined above. Pub is closed under the deductions of Definition 31.*

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 attacker deduction rules:

(i) $realisable(m_1) \wedge realisable(m_2) \implies realisable(m_1 + m_2)$

(ii) $realisable(m_1) \wedge n \in E \implies realisable(m_1 n) \wedge realisable(m_1 n^{-1})$

(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

$$h_2(f)(p) = -(h_1(f)(p)) \qquad \square$$

Our intention is for the model to respect the fact that some values are impossible for an attacker 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 34 $(P \cup E) \cap \mathbb{Z} = \emptyset$

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 34 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 38 $1 \in P \implies P \cap E = \emptyset$

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.

Even in the context of Condition 38, the important property of *unique realisability*—that a value can be realised in at most one way—is not generally true. Consider the following counter-example:

Example 5 Let $E = \{x_C\}$, $P = \{x_A\}$ and note that E and P satisfy Condition 38. The value $x_A x_C$ can be realised in at least two ways:
(i) Define $v(F_1,h_1)$ where:

$$F_1 = \{\{x_C \mapsto 0\}, \{x_C \mapsto 1\}\}$$
$$h_1(\{x_C \mapsto 0\}) = \{x_A \mapsto 0\}$$
$$h_1(\{x_C \mapsto 1\}) = \{x_A \mapsto 1\}$$

Then $v(F_1,h_1) = 1x_A x_C^1 + 0x_A x_C^0 = x_A x_C$.
(ii) Define $v(F_2,h_2)$ where:

$$F_1 = \{\{x_C \mapsto 1\}, \{x_C \mapsto 2\}\}$$
$$h_1(\{x_C \mapsto 1\}) = \{x_A \mapsto 1\}$$
$$h_1(\{x_C \mapsto 2\}) = \{x_A \mapsto 0\}$$

Then $v(F_2,h_2) = 1x_A x_C^1 + 0x_A x_C^2 = x_A x_C$. ■

In this example, the property of unique realisability is violated by the presence, in the linear combinations, of components equalling 0, in (i) we have $0x_A x_C^0 = 0$ and, in (ii), $0x_A x_C^2$. We introduce a normalisation procedure which removes such degenerate components.

Definition 39 *Given $v(F,h)$, define $norm(v(F,h)) = v(F',h')$ where:*

$$F' = F \setminus \{f \in F \mid ran(h(f)) = \{0\}\}$$
$$h' = h \setminus \{h(f) \mid f \in F \wedge ran(h(f)) = \{0\}\}$$

This would yield, for example, $norm(v(F_1,h_1)) = v(F_1',h_1')$, where $F_1' = \{\{x_c \mapsto 0\}\}$ and $h_1'(\{x_C \mapsto 1\}) = \{x_A \mapsto 1\}$. For the remainder of this paper, we assume that all values are normalised.

Theorem 310 (Unique realisability) *Fix some E and P satisfying Condition 38. Given some $v(F_1,h_1)$ and $v(F_2,h_2)$,*

$$v(F_1,h_1) = v(F_2,h_2) \implies F_1 = F_2 \wedge h_1 = h_2$$

Proof. For a contradiction assume some F_1, F_2, h_1 and h_2 such that $v(F_1,h_1) = v(F_2,h_2)$ but $\neg(F_1 = F_2 \wedge h_1 = h_2)$. Since:

$$\neg(F_1 = F_2 \wedge h_1 = h_2) = F_1 \neq F_2 \vee h_1 \neq h_2$$

there appear to be two cases to consider, $F_1 \neq F_2$ and $h_1 \neq h_2$. However, from Lemma 35 we have that $F_1 \neq F_2 \implies h_1 \neq h_2$, so it will be sufficient to treat the case $h_1 \neq h_2$.

In this case, $h_1 \neq h_2$ tells us that (a) there exists some $f \in F$, $p \in P$ s.t. $h_1(f)(p) \notin ran(h_2)$ or, (b) there exists some $f \in F$, $p \in P$ s.t. $h_2(f)(p) \notin ran(h_1)$. We will only treat case (a) here, noting that the same proof (with subscripts altered) also applies to (b).

Given $h_1(f)(p) \notin ran(h_2)$ either $f \notin F_2$ or $f \in F_2$. In this first case, there exists some coefficient of the linear combination of $v(F_1,h_1)$ which does not appear in $v(F_2,h_2)$. Since variables are symbolic, it is not possible for any combination of components from $v(F_2,h_2)$ to equal (and therefore cancel with) the presence of f in $v(F_1,h_1)$. Therefore, the equality $v(F_1,h_1) = v(F_2,h_2)$ cannot hold, yielding a contradiction.

In the case that $f \in F_2$, $h_1(f)(p) \notin ran(h_2)$ tells us that there exists some $y \in \mathbb{Z}$ s.t. $y \cdot p \cdot f$ is a component of the linear combination of $v(F_1,h_1)$. Furthermore, since variables are symbolic (and $y \cdot p \cdot f$ is thus distinct from all other terms on either side of the equation $v(F_1,h_1) = v(F_2,h_2)$), there are no values of the coefficients of $v(F_2,h_2)$ which will allow the equality to be met, yielding a contradiction which establishes the theorem. □

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 6 In the standard Diffie-Hellman protocol [11], a principal A, apparently running with a principal claiming to be B, and using the ephemeral secret x_A, performs the key computation $k_{AB}(z) = zx_A$ representing $Z_{AB} = g^{zx_A}$. ■

Definition 311 (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 attacker 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 a 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. $a = A \wedge b = C$
2. $a = B \wedge b = C$
3. $a = C \wedge b = A$
4. $a = C \wedge b = B$
5. $a = A \wedge b = A$
6. $a = B \wedge b = B$
7. $a = A \wedge b = B$
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 37 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 attacker 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 attacker's legitimate participation in the protocol. Any honest principal who willingly engages in a protocol run with the attacker 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 attacker 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:

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

As a result, the attacker 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 [13]. 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 A's willingness to engage in the protocol with herself, 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 41 *For protocol MTI A(0), given* $E^{A(0)} = \{r_C, x_C\}$, $P^{A(0)} = \{1, r_A, r_B, x_A, x_B\}$,

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

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 attacker C.

5 Further examples: A(i), B(i) and C(i)

In the previous section we saw how message-templates can be used to reason about the MTI A(0) protocol. In fact, A(0) is just one of an infinite number of protocols that fall under the banner of MTI. In this section we investigate these protocols and describe their corresponding security analyses. In particular, we note in passing that the correctness conditions for A(0) appear to apply equally well in the general case A(i). In the following, we concentrate instead on the remaining MTI protocol classes: B(i) and C(i).

5.1 General form of the MTI protocols

All MTI protocols are of the same basic form, involving the exchange of two messages, and aiming to provide implicit authentication of the resulting shared secret. The three classes of MTI protocols: A, B and C, differ in the precise format of the exchanged messages and the computation which each principal performs to derive the shared secret. The general form for each class is given in terms of a parameter $i \in \mathbb{Z}$ where, for instance, B(i) is the ith protocol of class B. For comparison, Table 1 summarises the exponent of the shared secret derived in each protocol.[2] Protocols A(i), B(i) and C(i) are given in Figures 3, 4 and 5, respectively.

[2] Reproduced from [4].

i	A(i)	B(i)	C(i)
-1	$x_A x_B^{-1} r_B + x_B x_A^{-1} r_A$	$x_A^{-1} r_A + x_B^{-1} r_B$	$x_A^{-1} r_A x_B^{-1} r_B$
0	$x_A r_B + x_B r_A$	$r_A + r_B$	$r_A r_B$
1	$x_A x_B r_B + x_B x_A r_A$	$x_A r_A + x_B r_B$	$x_A r_A x_B r_B$
2	$x_A x_B^2 r_B + x_B x_A^2 r_A$	$x_A^2 r_A + x_B^2 r_B$	$x_A^2 r_A x_B^2 r_B$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
k	$x_A x_B^k r_B + x_B x_A^k r_A$	$x_A^k r_A + x_B^k r_B$	$x_A^k r_A x_B^k r_B$

Table 1 Exponent of shared secret in the MTI protocols

A		B
$r_A \in_R \mathbb{Z}_q$		
$z_A = g^{x_A^i r_A}$	$\xrightarrow{z_A}$	$r_B \in_R \mathbb{Z}_q$
		$z_B = g^{x_B^i r_B}$
$Z_{AB} = z_B^{x_A} y_B^{r_A x_A^i}$	$\xleftarrow{z_B}$	$Z_{AB} = z_A^{x_B} y_A^{r_B x_B^i}$

Fig. 3 MTI A(i) protocol

A		B
$r_A \in_R \mathbb{Z}_q$		
$z_A = y_B^{r_A x_A^i}$	$\xrightarrow{z_A}$	$r_B \in_R \mathbb{Z}_q$
		$z_B = y_A^{r_B x_B^i}$
$Z_{AB} = z_B^{x_A^{-1} r_A} g^{r_A x_A^i}$	$\xleftarrow{z_B}$	$Z_{AB} = z_A^{x_B^{-1}} g^{r_B x_B^i}$

Fig. 4 MTI B(i) protocol

A		B
$r_A \in_R \mathbb{Z}_q$		
$z_A = y_B^{r_A x_A^i}$	$\xrightarrow{z_A}$	$r_B \in_R \mathbb{Z}_q$
		$z_B = y_A^{r_B x_B^i}$
$Z_{AB} = z_B^{x_A^{-1} r_A x_A^i}$	$\xleftarrow{z_B}$	$Z_{AB} = z_A^{x_B^{-1} r_B x_B^i}$

Fig. 5 MTI C(i) protocol

Note that the format of exchanged messages does not differ between the B(i) and C(i) protocols; instead, the distinction lies in the computation performed to derive the shared secret in each case.

It is also interesting to note that the A(i) protocols are rather different in nature to the B(i)/C(i) protocols. In the A(i) protocols, any message $M = g^{x_A^i r_A}$ sent by a principal A is independent of both:

- the claimed identity of the sender of any previously received message,
- the intended recipient of M.

A's intention to share a secret k with a principal B is captured in the key computation itself, $Z_{AB} = z_B^{x_A} y_B^{r_A x_A^i}$, which makes mention of B's public-key, y_B. (A's computation of the shared secret can be interpreted as a statement of belief:

at the point at which A performs the key computation she believes that Z_{AB} is a secret shared with no-one other than B.)

The opposite is true of the B(i)/C(i) protocols, where A's output message, $y_B^{r_A x_A^i}$, is dependent on the identity of the intended recipient (B). The key computations, on the other hand:

- Protocol B(i): $Z_{AB} = z_B^{x_A^{-1} r_A} g^{r_A x_A^i}$
- Protocol C(i): $Z_{AB} = z_B^{x_A^{-1} r_A x_A^i}$

are independent of the principal with whom A wishes to establish a shared secret since A only adds her own exponents (r_A and x_A) to the received value z_B. Whilst this difference does not appear to have security implications, it does affect the way in which we represent the protocols in our model.

5.2 Message-templates for B(i) and C(i)

We use the message-template of Definition 34, where we define the set E (of known exponents) as:

$$E^B = E^C = E = \{x_C, r_C\}$$

where x_C is the attacker C's long-term secret-key and r_C is some random exponent generated by C. Note that the set E^B is independent of the protocol parameter, i. For each $i \in \mathbb{Z}$, we define the set $P^B(i)$ (of known powers of g) as follows:

$$\begin{aligned} P^B(i) = & \{1, x_A, x_B\} \\ & \cup \\ & \{x_b r_a x_a^i \mid b \in \{A,B,C\}, a \in \{A,B\}\} \end{aligned}$$

The intuition behind this definition is based on several observations:

- An attacker can query a certification authority to obtain the public-keys, g^{x_A} and g^{x_B}, of A and B, respectively,
- When a principal $a = A$ wishes to establish a shared secret with $b = B$, she does so by sending a message with the exponent:

  $$x_b r_a x_a^i = x_B r_A x_A^i$$

- Since b may choose to run the protocol with any principal, we form P as a comprehension over the set $\{A,B,C\}$ of principals
- The attacker, C, is excluded from the comprehension over a since any message of the form $x_b r_C x_C^i$ ($b \in \{A,B,C\}$) can be formed using the deductions rules of Definition 31 on the values $x_b \in P^B(i)$, $x_C \in E^B$, $r_C \in E^B$.

This definition of P^B allows the attacker to learn a larger set of values than is typically assumed since it allows for the reuse of the random exponents. For instance, $P^B(i)$ includes the values $x_B r_A x_A^i$ and $x_C r_A x_A^i$ which respectively represent a run between A and B and a run between A and C; the session variable, r_A, is used in both cases. However, proving correctness in this context will enable us to conclude correctness in a more restricted context where r_A is used just once.

5.3 Reasoning about B(i)

Given an input z, a principal a in protocol B(i) computes the shared secret with exponent $z x_a^{-1} + r_a x_a^i$. As noted above, a's key computation is independent of the identity of the other protocol participant. The resulting key computation function is therefore:

$$k_a^{B(i)}(z) = z x_a^{-1} + r_a x_a^i$$

We wish to prove that, for any $i \in \mathbb{Z}$, $k_a^{B(i)}$ maintains secrecy. The case $a = C$ (representing the attacker's key computation function) is redundant since C can achieve the same result using his deduction rules on the values x_C, r_C. We therefore restrict ourselves to the following goal:

$$\forall i \in \mathbb{Z}, a \in \{A,B\}.k_a^{B(i)} \text{ maintains secrecy}$$

A first restriction

We begin by noting that, as with protocol A(0), we cannot expect the shared secret to be unknown to C if either A or B actually *choose* to run the protocol with C. In such a run A and B will generate and send the values $x_C r_A x_A^i \in P^B(i)$ and $x_C r_B x_B^i \in P^B(i)$, respectively, and $k_a^{B(i)}$ will not maintain secrecy. We therefore consider the restricted set:

$$\begin{aligned} P'^B(i) = & \{1, x_A, x_B\} \\ & \cup \\ & \{x_b r_a x_a^i \mid b, a \in \{A,B\}\} \end{aligned}$$

A second restriction

We fix some arbitrary $i \in \mathbb{Z}$ and consider the case $a = A$, noting that analogous results hold for $a = B$. For secrecy to fail there must exist some $z = v(F_1, h_1)$ and $k_A^{B(i)}(z) = v(F_2, h_2)$ such that:

$$v(F_1, h_1) x_A^{-1} + r_A x_A^i = v(F_2, h_2)$$

Consider the coefficient of $x_C^p r_C^q$, where p and q are arbitrary integers:

$$\begin{aligned} & h_2(\{x_C \mapsto p, r_C \mapsto q\}) = \{ \\ & \quad 1 \mapsto n_1, x_A \mapsto n_2, x_B \mapsto n_3, r_A x_A^{i+1} \mapsto n_4, x_A r_B x_B^i \mapsto n_5, \\ & \quad x_B r_A x_A^i \mapsto n_6, r_B x_B^{i+1} \mapsto n_7\} \end{aligned}$$

$$\begin{aligned} & h_1(\{x_C \mapsto p, r_C \mapsto q\}) = \{ \\ & \quad 1 \mapsto m_1, x_A \mapsto m_2, x_B \mapsto m_3, r_A x_A^{i+1} \mapsto m_4, x_A r_B x_B^i \mapsto m_5, \\ & \quad x_B r_A x_A^i \mapsto m_6, r_B x_B^{i+1} \mapsto m_7\} \end{aligned}$$

for some $m_1, \ldots, m_7 \in \mathbb{Z}$, $n_1, \ldots, n_7 \in \mathbb{Z}$ where the coefficients on both sides are the same:

$$\begin{array}{c} m_1x_A^{-1}+m_2+m_3x_Bx_A^{-1}+m_4r_Ax_A^i+m_5r_Bx_B^i \\ +m_6x_Br_Ax_A^{i-1}+m_7r_Bx_B^{i+1}x_A^{-1}+r_Ax_A^i \\ = \\ n_1+n_2x_A+n_3x_B+n_4r_Ax_A^{i+1}+n_5x_Ar_Bx_B^i \\ +n_6x_Br_Ax_A^i+n_7r_Bx_B^{i+1} \end{array}$$

One solution that presents itself is $m_4 = -1$, with all other coefficients set to 0, resulting in:

$$-1r_Ax_A^i + r_Ax_A^i = 0$$

As with the impersonation attack on A(0), this attack exploits a scenario in which A runs the protocol with herself, generating the value $x_Ar_Ax_A^i$. The attacker returns the inverse of this value, $-x_Ar_Ax_A^i$, causing A to compute the exponent of the shared secret as 0 ($g^0 = 1$):

$$k_A^{B(i)}(-x_Ar_Ax_A^i) = -x_Ar_Ax_A^ix_A^{-1} + r_Ax_A^i = 0$$

As with A(0), this attack suggests that any implementation should ensure that the protocol participants are distinct. However, it is worth considering whether this is strictly necessary. The impersonation attack on A(0) allows the attacker to force the shared secret to be computed as one of a number of values. Here, the attack only appears to work when the resulting exponent is 0 (i.e., the attacker cannot inject his own value into the shared secret). We formalise this in the property of *non-degenerate-key secrecy*:

Definition 51 (non-degenerate-key secrecy) *Given a system defined by E and P, a key computation function k maintains* non-degenerate-key secrecy *iff:*

$$\forall m.realisable(m) \wedge k(m) \neq 0 \implies \neg realisable(k(m))$$

non-degenerate-key secrecy is weaker than the secrecy property of Definition 311 since it disregards cases where the exponent of the shared secret is computed as 0. Note that this definition does not prevent the exponent of values *passing on the network* from being 0.

Results

Restricting the output of $k_A^{B(i)}$ to non-zero values translates to a constraint on the coefficients of $v(F_2,h_2)$: that at least one has to be initialised to a non-zero value. Under this constraint, it can be shown that the key computation function $k_a^{B(i)}$ maintains non-degenerate-key secrecy, given E^B and P'^B:

Theorem 52 *For protocol MTI B(i), given* $E^B = \{r_C, x_C\}$ *and* $P'^B(i) = \{1, x_A, x_B\} \cup \{x_b r_a x_a^i \mid b,a \in \{A,B\}\}$*:*

$\forall i \in \mathbb{Z}, a \in \{A,B\}.k_a^{B(i)}$ maintains non-degenerate-key secrecy.

Proof. We fix some arbitrary $i \in \mathbb{Z}$ and consider the case $a = A$, (an analogous result holds for $a = B$). For non-degenerate secrecy to fail there must exist some $z = v(F_1,h_1)$ and $k_A^{B(i)}(z) = v(F_2,h_2)$ such that:

$$v(F_1,h_1)x_A^{-1} + r_Ax_A^i = v(F_2,h_2)$$

where $v(F_2,h_2) \neq 0$. The constraint that the resulting secret be non-zero means that there must exist at least one non-zero coefficient in $v(F_2,h_2)$. Let $p \in \mathbb{Z}$, $q \in \mathbb{Z}$ be the powers of x_C and r_C (respectively) in $v(F_2,h_2)$ in which a non-zero coefficient occurs. the coefficient of $x_C^p r_C^q$, where p and q are arbitrary integers:

$$\begin{array}{l} h_2(\{x_C \mapsto p, r_C \mapsto q\}) = \{ \\ \quad 1 \mapsto n_1, x_A \mapsto n_2, x_B \mapsto n_3, r_Ax_A^{i+1} \mapsto n_4, x_Ar_Bx_B^i \mapsto n_5, \\ \quad x_Br_Ax_A^i \mapsto n_6, r_Bx_B^{i+1} \mapsto n_7\} \end{array}$$

$$\begin{array}{l} h_1(\{x_C \mapsto p, r_C \mapsto q\}) = \{ \\ \quad 1 \mapsto m_1, x_A \mapsto m_2, x_B \mapsto m_3, r_Ax_A^{i+1} \mapsto m_4, x_Ar_Bx_B^i \mapsto m_5, \\ \quad x_Br_Ax_A^i \mapsto m_6, r_Bx_B^{i+1} \mapsto m_7\} \end{array}$$

for some $m_1,\ldots,m_7 \in \mathbb{Z}$, $n_1,\ldots,n_7 \in \mathbb{Z}$ where the coefficients on both sides are the same:

$$\begin{array}{c} m_1x_A^{-1}+m_2+m_3x_Bx_A^{-1}+m_4r_Ax_A^i+m_5r_Bx_B^i \\ +m_6x_Br_Ax_A^{i-1}+m_7r_Bx_B^{i+1}x_A^{-1}+r_Ax_A^i \\ = \\ n_1+n_2x_A+n_3x_B+n_4r_Ax_A^{i+1}+n_5x_Ar_Bx_B^i \\ +n_6x_Br_Ax_A^i+n_7r_Bx_B^{i+1} \end{array}$$

One of $n_1,\ldots,n_7$ must be non-zero; let this coefficient be n_r . By assumption we have that variables are symbolic and that a given symbol x is distinct from all others. Specifically, there is no value of the coefficient n_r which can enable the equality to be met. For example, if $n_r = n_4$, then the component $n_rr_Ax_A^{i+1}$ is distinct from all terms on either side of the equation.

As a result, we conclude that, for any $i \in \mathbb{Z}$, $k_a^{B(i)}(z)$ is unrealisable (under the assumption that $k_a^{B(i)}(z) \neq 0$), and the theorem holds. □

5.4 Reasoning about C(i)

The C(i) protocols involve the same exchange of messages as the class B protocols; the difference lies in the key computation. In an ideal run of C(i), each principal computes the key:

$$Z_{AB} = (g^{x_Ar_Bx_B^i})^{x_A^{-1}r_Ax_A^i} = (g^{x_Br_Ax_A^i})^{x_B^{-1}r_Bx_B^i} = g^{x_A^ir_Ax_B^ir_B}$$

In our model, a principal a, on receipt of an input z, will perform the following key computation:

$$k_a^{C(i)}(z) = zx_a^{-1}r_ax_a^i$$

Following the approach that we used for B(i), we consider whether there exist realisable values, $v(F_1,h_1)$ (whose coefficients are represented by subscripted m values) and $v(F_2,h_2)$ (represented by subscripted n values), such that:

$$k_a^{C(i)}(v(F_1,h_1)) = v(F_1,h_1)x_a^{-1}r_a x_a^i = v(F_2,h_2)$$

To exclude cases where honest principals willingly engage with the attacker, we define the P set as per the first restriction in the B(i) analysis, above. Specifically, we have that:

$$\begin{aligned} P^C(i) = &\{1, x_A, x_B\} \\ &\cup \\ &\{x_b r_a x_a^i \mid b,a \in \{A,B\}\} \end{aligned}$$

We also define $E^C = E^B = \{x_C, r_C\}$. As before, we fix $a = A$ and consider the coefficients of $x_C^p r_C^q$, for arbitrary integers p and q. For secrecy to fail, the following equality must hold:

$$\begin{gathered} m_1 x_A^{-1} r_A x_A^i + m_2 x_A x_A^{-1} r_A x_A^i + m_3 x_B x_A^{-1} r_A x_A^i \\ + m_4 r_A x_A^{i+1} x_A^{-1} r_A x_A^i + m_5 x_A r_B x_B^i x_A^{-1} r_A x_A^i \\ + m_6 x_B r_A x_A^i x_A^{-1} r_A x_A^i + m_7 r_B x_B^{i+1} x_A^{-1} r_A x_A^i \\ = \\ n_1 + n_2 x_A + n_3 x_B + n_4 r_A x_A^{i+1} + n_5 x_A r_B x_B^i \\ + n_6 x_B r_A x_A^i + n_7 r_B x_B^{i+1} \end{gathered}$$

In contrast to the A(i) and B(i) protocols, the left-hand side of the above equation does not contain any component that is not multiplied by an m coefficient. This arises from the difference in key computation between A(i)/B(i) and C(i). In the former cases, the key computation relies on the multiplication of two elements of G, and results in the presence of addition in the exponent of the shared secret ($g^x \cdot g^y = g^{x+y}$). In the latter case, the key is computed by exponentiating the received value, and the exponent of the shared secret is therefore a simple product ($(g^x)^y = g^{xy}$). For this reason, the above equation can be solved, very simply, by setting all coefficients to 0, resulting in $0 = 0$. This solution corresponds to a well-known attack on standard Diffie-Hellman, where the attacker simply replaces the exchanged values, z_A and z_B, with $g^0 = 1$. The principals then compute the shared secret:

$$Z_{AB} = 1^{x_A^{-1} r_A x_A^i} = 1^{x_B^{-1} r_B x_B^i} = 1$$

which is realisable by the attacker. This attack can be prevented in a straightforward manner; each principal must check incoming messages and reject those which have the degenerate value $g^0 = 1$. This translates to a constraint on the coefficients of $v(F_1,h_2)$: that at least one has to be initialised to a non-zero value. We formalise this property in the concept of *non-degenerate-input secrecy*:

Definition 53 (non-degenerate-input secrecy) *Given a system defined by E and P, a key computation function k maintains* non-degenerate-input secrecy *iff:*

$$\forall m.realisable(m) \wedge m \neq 0 \implies \neg realisable(k(m))$$

Note the difference between the three definitions of secrecy that we have introduced in this paper—the first (secrecy) is unconstrained; the second (non-degenerate-key secrecy) looks at cases where the exponent of the value *output* by the key computation function is non-zero; the third (non-degenerate-input-secrecy) looks at cases where the exponent of the value *input* to the key computation function is non-zero.

Results

It can be shown that the key computation function $k_a^{C(i)}$ maintains non-degenerate-input secrecy, given E^B and P'^B:

Theorem 54 *For protocol C(i), given* $E^C = \{r_C, x_C\}$ *and* $P^C(i) = \{1, x_A, x_B\} \cup \{x_b r_a x_a^i \mid b,a \in \{A,B\}\}$*:*

$\forall i \in \mathbb{Z}, a \in \{A,B\}.k_a^{C(i)}$ maintains non-degenerate-input secrecy.

Proof. The argument used in this proof is similar to that used in the proof of Theorem 52. We consider whether there exist realisable values, $v(F_1,h_1)$ (whose coefficients are represented by subscripted m values) and $v(F_2,h_2)$ (represented by subscripted n values), such that $k_a^{C(i)}(v(F_1,h_1)) = v(F_2,h_2)$. Specifically, we fix some arbitrary $i \in \mathbb{Z}$, $a = A$, and consider the coefficients of $x_C^p r_C^q$ for arbitrary integers p and q. For secrecy to fail, the following equality must hold:

$$\begin{gathered} m_1 x_A^{-1} r_A x_A^i + m_2 x_A x_A^{-1} r_A x_A^i + m_3 x_B x_A^{-1} r_A x_A^i \\ + m_4 r_A x_A^{i+1} x_A^{-1} r_A x_A^i + m_5 x_A r_B x_B^i x_A^{-1} r_A x_A^i \\ + m_6 x_B r_A x_A^i x_A^{-1} r_A x_A^i + m_7 r_B x_B^{i+1} x_A^{-1} r_A x_A^i \\ = \\ n_1 + n_2 x_A + n_3 x_B + n_4 r_A x_A^{i+1} + n_5 x_A r_B x_B^i \\ + n_6 x_B r_A x_A^i + n_7 r_B x_B^{i+1} \end{gathered}$$

In the case of non-degenerate-input secrecy, we only consider cases where one of the m coefficients has a non-zero value. By assumption we have that variables are symbolic and that a given symbol x is distinct from all others. In particular, since each component of the linear combination is distinct from all others (on either side of the equation), there can be no values of the coefficients which cause the equality to hold. Therefore, the theorem holds. □

6 Discussion

6.1 Summary of results

The three classes of MTI protocols are conceptually similar, yet the security results yielded by our model are subtly different in each case:

A(i)

The model of the A(i) protocol maintains the secrecy of the shared secret if the run involves distinct, and honest, protocol participants. If a principal is willing to run the protocol with herself an attacker can manipulate that run to learn the (possibly non-degenerate) session-keys.

B(i)

The B(i) protocol maintains the secrecy of the shared secret if the run involves honest protocol participants, and each rejects any shared secrets computed as 1 (i.e., when the exponent of the shared secret is 0). Note, in particular, that this does not preclude situations where a principal runs the protocol with herself.

C(i)

The C(i) protocol maintains the secrecy of the shared secret if the run involves honest protocol participants and each rejects incoming values of 1 (i.e., g^0). If a principal is willing to accept 1 as an input to the key computation function she will compute a shared secret ($Z_{AB} = 1$) which is known to the attacker.

6.2 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 [20]. 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 [21], 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 [10]. 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.[3]

One could extend the analogy by defining the set *Sec*, of secret messages, as the range of a given key computation function (whose input is restricted to values in *Pub*). The set *Sec* would then correspond to the set of messages assigned a rank of sec by a rank function. In this view secrecy is maintained if the sets *Pub* and *Sec* contain no common elements.

[3] In fact, *Pub* is similar to Heather's concept of a minimal rank function [12].

6.3 Pereira and Quisquater's approach

Recently, Pereira and Quisquater [19] developed a formal model of the Cliques conference key agreement protocols [3], 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 attacker 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 attacker to learn a pair of messages separated by the ratio $\frac{y}{x}$. The model allows the attacker 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:

$$\frac{zx_A + x_B r_A}{z} = x_A + \frac{x_B r_A}{z}$$

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 provides *services*, in which protocol participants receive a message, perform some computation on that message and send out the result. 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 attacker 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. One could envisage weakening the assumptions of the current work by internalising such services in the attacker (in the style of Broadfoot and Roscoe [6]) where, for example, the multiplication of a value with yz is encoded as an additional attacker 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.

6.4 Assumptions and attacks

In light of the above, some care is needed in establishing the assumptions upon which our proofs of the the MTI protocols are based.

Consider the three assumptions of Section 3. These assumptions are necessary, since attacks exist when one or more of them are relaxed. For example, Menezes et al. [17] discovered an unknown key-share attack on all classes of the MTI protocols under the assumption that an attacker can register a public-key y_C which is related to A's public-key by the equation $y_C = y_A^{x_C} = g^{x_A x_C}$. However, C does not know the corresponding secret key (in violation of Assumption 32).[4] As another example, Lim and Lee devised attacks which apply to MTI variants [14]. These attacks depend on the attacker sending a value to an honest principal B which is not in the group G, and so violates the Assumption 31. Lastly, Burmester proposed an attack which requires the attacker, C, to run the protocol with both A and B and later induce A and B to reveal the keys used in sessions between them [7]. This violates Assumption 33.

These attacks are computational in flavour, and it is not clear whether a symbolic approach should seek to reason about such attacks. It is clear, however, that care should be taken—when claiming a proof of correctness—to state the assumptions on which that proof is founded.

7 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 protocols. The work hinges around the idea of a message-template, a term which defines, in a highly abstract way, the values that can be deduced by an attacker 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 protocols beyond the MTI suite. In particular, we have used the approach to reason about two further key agreement protocols: one proposed independently by Just and Vaudenay [13] and Song and Kim [22], and another due to Ateniese, Steiner and Tsudik [3]; in both cases we can apply our approach without modification. Other protocols, however, may prompt us to extend the approach. The key establishment protocol of Agnew, Mullin and Vanstone, for instance [2], makes use of messages of the form $g^x \cdot y$, where y is an integer (and not a group element). This is inexpressible in our current model where we are limited to messages expressible as g raised to the power of a sum of products of integers. Relaxing the restrictions on our algebra to allow the expression of such messages seems particularly interesting. The consideration of further protocols (such as Cliques) may require us 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 [19], 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 [23]. Future work will investigate whether this correspondence enables us to deduce general principles with which a protocol can be proven correct.

[4] However, it is not clear that this attack actually violates implicit key authentication. Furthermore, as noted in [4], the importance of unknown key-share attacks is questionable as there exist well-understood methods of prevention.

References

1. Abadi, M., Cortier, V.: Deciding knowledge in security protocols under equational theories. In: 31st international Colloquium on Automata, Languages and Programming: ICALP'04, *Lecture Notes in Computer Science*, vol. 3142. Springer-Verlag (2004)
2. Agnew, G., Mullin, R., Vanstone, S.: An interactive data exchange protocol based on discrete exponentiation. In: Advances in cryptology: Proceedings of EUROCRYPT '88, *Lecture Notes in Computer Science*, vol. 0330. Springer-Verlag (1988)
3. Ateniese, G., Steiner, M., Tsudik, G.: Authenticated group key agreement and friends. In: Proceedings of the 5th ACM conference on Computer and Communication Security. ACM Press (2000)
4. Boyd, C., Mathuria, A.: Protocols for Authentication and Key Establishment. Springer-Verlag (2003)
5. Bresson, E., Chevassut, O., Pointcheval, D.: Provably authenticated group Diffie-Hellman key exchange—the dynamic case. In: Advances in Cryptology: Proceedings of ASIACRYPT '01, *Lecture Notes in Computer Science*, vol. 2248. Springer-Verlag (2001)
6. Broadfoot, P., Roscoe, A.W.: Internalising agents in CSP protocol models. In: Workshop on Issues in the Theory of Security: WITS '02 (2002)
7. Burmester, M.: On the risk of opening distributed keys. In: Advances in Cryptology: Proceedings of CRYPTO '94, *Lecture Notes in Computer Science*, vol. 0839. Springer-Verlag (1994)
8. Delicata, R., Schneider, S.: A formal model of Diffie-Hellman using CSP and rank functions. Tech. Rep. CSD-TR-03-05, Department of Computer Science, Royal Holloway, University of London (2003)
9. Delicata, R., Schneider, S.: A formal approach to the verification of a class of Diffie-Hellman protocols. In: 3rd international workshop on Formal Aspects of Security and Trust: FAST2005, Lecture Notes in Computer Science. Springer-Verlag (2005). To appear
10. Delicata, R., Schneider, S.: Temporal rank functions for forward secrecy. In: Proceedings of the 18th Computer Security Foundations Workshop: CSFW-18. IEEE Computer Society Press (2005)
11. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory **IT-22**(6) (1976)
12. Heather, J.: 'Oh! ... Is it *really* you?' *using rank functions to verify authentication protocols*. Ph.D Thesis, Royal Holloway, University of London (2001)
13. Just, M., Vaudenay, S.: Authenticated multi-party key agreement. In: Advances in Cryptology: Proceedings of ASIACRYPT '96, *Lecture Notes in Computer Science*, vol. 1163. Springer-Verlag (1996)
14. Lim, C., Lee, P.: A key recovery attack on discrete log-based schemes using a prime order subgroup. In: Advances in Cryptology: Proceedings of CRYPTO '97, *Lecture Notes in Computer Science*, vol. 1294. Springer-Verlag (1994)

15. Matsumoto, T., Takashima, Y., Imai, H.: On seeking smart public-key-distribution systems. Transactions of the IECE of Japan **E69**(2) (1986)
16. Meadows, C.: 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)
17. Menezes, A., Qu, M., Vanstone, S.: Some new key agreement protocols providing mutual implicit authentication. In: Workshop on Selected Areas in Cryptography: SAC '95 (1995)
18. Millen, J., Shmatikov, V.: Symbolic protocol analysis with products and Diffie-Hellman exponentiation. In: Proceedings of the 16th Computer Security Foundations Workshop: CSFW-16. IEEE Computer Society Press (2003)
19. Pereira, O., Quisquater, J.J.: Security analysis of the Cliques protocols suites. In: Proceedings of the 14th IEEE Computer Security Foundations Workshop: CSFW-14. IEEE Computer Society Press (2001)
20. Schneider, S.: Verifying authentication protocols with CSP. In: Proceedings of the 10th IEEE Computer Security Foundations Workshop: CSFW-10. IEEE Computer Society Press (1997)
21. Schneider, S.: Concurrent and Real-time Systems: The CSP Approach. John Wiley and Sons (2000)
22. Song, B., Kim, K.: Two-pass authenticated key agreement protocols with key confirmation. In: Progress in Cryptology: Proceedings of INDOCRYPT 2000, *Lecture Notes in Computer Science*, vol. 1977. Springer-Verlag (2000)
23. Thayer Fábrega, F.J., Herzog, J., Guttman, J.: Strand spaces: Proving security protocols correct. Journal of Computer Security **7**(2/3) (1999)