Logical Protocol Analysis for Authenticated Diffie-Hellman*

Daniel J. Dougherty Joshua D. Guttman

Worcester Polytechnic Institute
{dd,guttman}@wpi.edu

Abstract. Draft of August 15, 2011. Diffie-Hellman protocols for authenticated key agreement construct a shared secret with a peer using a minimum of communication and using limited cryptographic operations. However, their analysis has been challenging in computational models and especially in symbolic models.

In this paper, we develop a framework for protocol analysis that combines algebraic and strand space ideas. We show that it identifies exact assumptions on the behavior of a certifying authority. These assumptions establish the confidentiality and authentication properties for two protocols, the Unified Model and Menezes-Qu-Vanstone (MQV). For MQV, we establish a stronger authentication property than previously claimed, using a stronger (but realistic) assumption on the certifying authority.

Verification within our framework implies that the adversary has no strategy that works uniformly, independent of the choice of the cyclic group in which the protocol operates. Indeed, we provide an equational theory which constitutes an analysis of these uniform strategies. We provide an abstraction, the notion of indicator, which leads to easy proofs of protocol correctness assertions. Computational soundness awaits further investigation.

1 Introduction

The Diffie-Hellman key exchange [8] is a widely used cryptographic idea. Each principal A, B chooses a random value, x, y resp., raising a base g to this power modulo a suitable prime p:

$$A, x \qquad \bullet \xrightarrow{g^x} \qquad \xleftarrow{g^y} \bullet \qquad B, y \tag{1}$$

They can then both compute the value $(g^y)^x = g^{xy} = (g^x)^y$ (modulo p, as we will no longer explicitly repeat). Since there is reason to believe that g^{xy} is indistinguishable from g^z for randomly chosen z, we can treat g^{xy} as a new shared secret for A, B. The protocol is thus secure against a passive adversary,

* Supported by the National Science Foundation under grant CNS-0952287.

who observes what the compliant principals do, but can neither create messages nor alter (or misdirect) messages of compliant principals.

It is however certainly not secure against an active adversary, which can choose its own x', y', sending $g^{y'}$ to A instead of g^y, and sending $g^{x'}$ to B instead of g^x. In this case, each of A, B actually shares one key with the adversary, who can act as a man in the middle, re-encrypting messages in any conversation between A and B. So as not to prejudice ourselves in evaluating the possible attacks, we will write R_B for the public value that A receives, purportedly from B, and R_A for the public value that B receives, purportedly from A. In the intended case, $R_A = g^x$ and $R_B = g^y$.

One approach to authenticating a Diffie-Hellman exchange, originating in the Station-to-Station protocol STS [9], is digitally to sign parts of the exchange. For instance, in a simplified STS, the exchange in Eqn. 1 is followed by the signed messages:

$$A \quad \bullet \xrightarrow{[\![\, g^x,\ R_B \,]\!]_A} \quad \xleftarrow{[\![\, g^y,\ R_A \,]\!]_B} \bullet \quad B \tag{2}$$

where the signatures exclude a man in the middle.[1] STS requires an additional message transmission and reception for each participant, in each session. Moreover, each participant must also prepare one digital signature and also verify one digital signature specifically for that session. STS requires some public key infrastructure to certify the signature verification keys of A and B.

An alternative to using per-session digital signatures is *implicit verification* [2]. Here the goal is to ensure that any principal that can compute the same value as A *can only* be B. To implement this idea, each principal maintains a long-term secret, which we will write a for principal A, b for B; they publish the long-term public values g^a, g^b, which we will refer to as Y_A, Y_B, etc. The trick is to build the use of the private values a, b into the computation of the shared secret, so that only A, B can do it. For instance, in the "Unified Model" UM of Ankney, Johnson, and Matyas, the principals combine long term values with short term values by concatenating and hashing. They follow the exchange of Eqn. 1 with these computations, where $H(x)$ is a hash of x:

$$A:\ k = H({Y_B}^a,\ {R_B}^x) \quad B:\ k = H({Y_A}^b,\ {R_A}^y), \tag{3}$$

obtaining a shared value if $R_A = g^x$ and $R_B = g^y$. We will write $\mathsf{k_{um}}(c, Y, z, R) = H(Y^c,\ R^z)$. A and B respectively compute

$$\mathsf{k_{um}}(a, Y_B, x, R_B) \quad \text{and} \quad \mathsf{k_{um}}(b, Y_A, y, R_A). \tag{4}$$

Here again some public key infrastructure is required so that each principal knows to associate the intended peer P with the right public value to Y_P. However, no digital signature needs to be generated or checked specific to this run. If A frequently has sessions with B, A can amortize the cost of verifying B's certificate once, by keeping Y_B in secure storage.

[1] We use $t,\ t'$ for the result of concatenating t with t'. A digitally signed message $[\![\, t \,]\!]_A$ means the message t concatenated with a digital signature algorithm's output when applied to a hash of t, using a signing key associated with the principal A.

One might prefer a protocol in which the operations are only algebraic, as distinguished from UM's combination of algebraic operations such as exponentiation with the rather different operations of concatenation and hashing. Indeed Menezes-Qu-Vanstone (MQV) [23] does exactly this, computing the key via the rules:

$$A:\ k = (R_B \cdot {Y_B}^{[R_B]})^{s_A} \qquad B:\ k = (R_A \cdot {Y_A}^{[R_A]})^{s_B} \tag{5}$$

where $s_A = x + a[R_A]$ and $s_B = y + b[R_B]$. The "box" operator coerces numbers mod p to a convenient form in which they can be used as exponents. In the literature this is generally written with a bar, as $\overline{R_B}$, which is typographically more cumbersome. We will discuss it more below. Now, in a successful run, A obtains the value

$$(g^y \cdot (g^b)^{\overline{g^y}})^{s_A} = (g^{(y+b\overline{g^y})})^{(x+a\overline{g^x})} = g^{(s_B \cdot s_A)} \tag{6}$$

and B obtains $g^{s_A \cdot s_B}$, which is the same value. MQV differs from UM and other related protocols only in the function that the principals use to compute the key. MQV uses the definition:

$$\mathsf{k}_{\mathsf{mqv}}(a, Y, x, R) = (R \cdot Y^{[R]})^{(x+a[g^x])} \tag{7}$$

This key computation makes MQV algebraically challenging to model and to analyze. There is indeed some controversy about its security [18, 20, 21, 24].

Rigorous treatment in computational models is hard, and until now, they have been out of reach in symbolic models [11, 19, 22]. These symbolic approaches to protocol analysis have relied on unification as a central part of their reasoning. Unifiability, in the presence of the ring structure used in Eqn. 6, is undecidable, essentially by the unsolvability of Hilbert's tenth problem.

Nevertheless, interesting and relevant problems could be clarified by a more suitable symbolic approach. Exactly what constraints do the protocols assume about the authority that certifies public values Y_P? When do the protocols require an additional step of key confirmation, or including the principals' identities in the key computation? How can we define the authentication that "implicit authentication" provides?

Our contributions. We formalize UM and MQV within the strand space model [16, 27], adapted to this purpose, and especially to use a message algebra rich enough to express their operations. We have given special attention to the constraints on the certifying authority.

We establish confidentiality in this model, i.e. that a session key is not disclosed to an active adversary in either UM or MQV, on reasonable assumptions. One can also prove in a similar manner that session keys are not disclosed when other session keys become disclosed, or when long term keys are disclosed, although we will not do so here. Our notion of disclosure is, however, a weaker assertion than indistinguishability of a real session key from a random value, to which our model is not currently well-suited.

More interestingly, for MQV we identify a slightly tighter requirement on a certifying authority. This states that a full run of the certification protocol

may never be included between the beginning of a user session yielding a session key, and its end. This is easily implemented. If user sessions always time out before some maximum time t elapses, then the CA only needs to ensure that at least time t elapses between receiving a certification request and transmitting the corresponding certificate. Other implementation strategies could also be imagined.

We will prove that this assumption about the ordering of user sessions and CA sessions eliminates attacks similar to Kaliski's [18]; as a consequence, we can prove an authentication result stronger than claimed by Blake-Wilson and Menezes [2] and Law et al. [23]. When two compliant principals A, B actually share a session key generated via MQV, do they necessarily know their partners? The Kaliski attack on MQV finds a situation in which a non-compliant E convinces B that the key is shared with E, even though it is shared with A. However, we will prove that our CA ordering assumption eliminates such executions.

We believe this establishes that our method delivers results that would be hard to obtain in other ways. This result combines information about the possible orderings of events and the algebraic properties of the values transmitted and received in those events. The strand space framework has always provided a reliable framework for reasoning about the causal relations among events, and thus their possible temporal orderings. In this paper, we weld that together with a message algebra for DH operations.

Since reasoning about equations between messages is hard in this algebra, for the MQV analysis we have introduced the notion of an *indicator*. An indicator is a vector of integers that describes how many occurrences each of a number of critical variables has in a particular level of exponent of a message value. We prove (Thm. 18) that these values are preserved under adversary actions when the critical variables are unavailable to the adversary. This means that the adversary is unable to create a message with an indicator different from any he was given. The indicators also provide a convenient fingerprint to use to determine whether messages can be equated by instantiating only non-critical variables. If they have different indicators, then they cannot be. The indicators provide the main proof technique in Section 6.

Structure of this paper. We start by introducing the strand space theory, as we will use it here (Section 2). We then turn to the Universal Model protocol in Section 3, which we use to introduce our specification style, to define a set of constraints on the CA, and to introduce the security properties that we will also study in MQV.

After this we characterize the algebra of basic messages more mathematically, and introduce the notion of *indicator* (Section 4). Section 5 provides more detail on the adversary, and establishes that indicators are preserved by adversary actions.

In Section 6, we establish the main results about MQV confidentiality and authenticity. Section 7 provides some conclusions and comments on related (and future) work.

2 Strands and Bundles

In this paper, we will adapt the strand space ideas to our context, in which messages may form a more complex algebraic structure than [16, 27] envisage.

Strands. A *strand* is a sequence of local actions called *nodes*. Each local action in a strand is called a *node*, and a node may be either a message *transmission*, a message *reception*, or else a local or *neutral* node. Neutral nodes are events in which a principal consults or updates its local state [15]. We write bullets $\bullet$ for transmission and reception events and circles $\circ$ for "neutral" events, in which a principal consults or updates its long-term state. Double arrows indicate successive events on the same strand, e.g. $\circ \Rightarrow \bullet \Rightarrow \bullet$.

Each strand is either a *regular strand*, which represents the sequence of local actions made be a single principal in a single local session of a protocol, or else an *adversary strand*, which represents a single action of the adversary. Only regular strands use neutral nodes $\circ$. An adversary strand consists of zero or more reception nodes followed by one or more transmission nodes. It represents the adversary obtaining these newly transmitted values as a function of the values received; or creating it, if there are no reception nodes. All values that the adversary handles are received and transmitted; none are silently obtained from any long-term state.

We regard the messages transmitted and received on $\bullet$ nodes, and obtained from long-term state on neutral nodes $\circ$, as forming an abstract algebra. *Concatenation* and *encryption* are operators that construct values in the algebra from a pair of given values, and we regard v_0, v_1 as equal to u_0, u_1 just in case $v_0 = u_0$ and $v_1 = u_1$. Similarly, $\{|v_0|\}_{v_1}$ equals $\{|u_0|\}_{u_1}$ just in case $v_0 = u_0$ and $v_1 = u_1$. That is, they are *free* operators.

The *basic* values that are neither concatenations nor encryptions include principal names; keys of various kinds; group elements x, $x \cdot y$, and g^x; and text values. We regard variables ("indeterminates") such as x as values distinct from values of other forms, e.g. products $z \cdot y$, or from other variables. A variable represents a "degree of freedom" in a description of some executions, which can be instantiated or restricted. It may also represent an independent choice, as A's choice of a group element x to build g^x is independent of B's choice of y. The algebra of basic message values is defined in Section 4 as the normal forms of an AC rewriting system.

If n is a node, and the message t is transmitted, received, or coordinated with the state on n, then we write $t = \mathsf{msg}(n)$.

Ingredients and origination. A value t_1 is an *ingredient* of another value t_2, written $t_1 \sqsubseteq t_2$, if t_1 contributes to t_2 via concatenation or as the plaintext of encryptions: $\sqsubseteq$ is the least reflexive, transitive relation such that:

$$t_1 \sqsubseteq t_1,\ t_2, \qquad t_2 \sqsubseteq t_1,\ t_2, \qquad t_1 \sqsubseteq \{|t_1|\}_{t_2}.$$

By this definition, $t_2 \sqsubseteq \{|t_1|\}_{t_2}$ implies that (anomalously) $t_2 \sqsubseteq t_1$. Also, for basic values v_1, v_2, if $v_1 \sqsubseteq v_2$ then $v_1 = v_2$.

A value t *originates* on a transmission node n if $t \sqsubseteq \mathsf{msg}(n)$, so that it is an ingredient of the message sent on n, but it was not an ingredient of any message earlier on the same strand. That is, $m \Rightarrow^+ n$ implies $t \not\sqsubseteq \mathsf{msg}(m)$.

A basic value is *uniquely originating* in an execution if there is exactly one node at which it originates. Freshly chosen nonces or DH values g^x are typically assumed to be uniquely originating. A value is *non-originating* if there is no node at which it originates. A long term secret such as a signature key or a private decryption key is assumed to be non-originating to formalize being uncompromised. Because the adversary strands always receives their arguments as incoming messages, an adversary strand that encrypts a message requires that the key has been transmitted. Similarly, a decryption strand requires the same for the symmetric key, or for the inverse decryption member of a key pair.

Very often in DH-style protocols unique origination and non-origination are used in tandem. When a compliant principal generates a random x and transmits g^x, the former will be non-originating and the latter uniquely originating.

Executions are bundles. We formalize the idea of a protocol execution by *bundles*. A bundle is a directed, acyclic graph. Its vertices are nodes on some strands (which may include both regular and adversary strands). Its edges include the strand succession edges $n_1 \Rightarrow n_2$, as well as *communication edges* written $n_1 \rightarrow n_2$. Such a dag $\mathcal{B} = (V, E_{\Rightarrow} \cup E_{\rightarrow})$ is a *bundle* if it is causally self-contained, meaning:

- If $n_2 \in V$ and $n_1 \Rightarrow n_2$, then $n_1 \in V$ and $(n_1, n_2) \in E_{\Rightarrow}$;
- If $n_2 \in V$ is a reception node, then there is a unique transmission node $n_1 \in V$ such that $\mathsf{msg}(n_2) = \mathsf{msg}(n_1)$ and $(n_1, n_2) \in E_{\rightarrow}$;
- The precedence ordering $\preceq_{\mathcal{B}}$ for $\mathcal{B}$, defined to be $(E_{\Rightarrow} \cup E_{\rightarrow})^*$, is a well-founded relation.

The first clause says that a node has a causal explanation from the occurrence of the earlier nodes on its strand. The second says that any reception has the causal explanation that the message was obtained from some particular transmission node. The last clause says that causality is globally well-founded. It holds automatically in *finite* dags $\mathcal{B}$, which are the only ones we consider here.

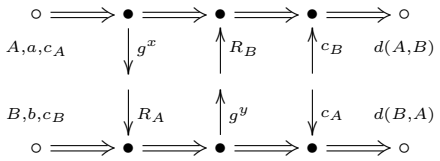
When we assume that a value is non-originating, or uniquely originating, we are effectively constraining which bundles $\mathcal{B}$ are of interest to us, namely those in which the value originates on no node of $\mathcal{B}$, or on one node of $\mathcal{B}$, respectively.

3 The Unified Model

We first consider the Unified Model UM. In this section, we first (Section 3.1) describe the protocol, both the way that the session keys are agreed, and also the interaction between the clients and the certifying authority. We then describe the certification process. We turn then in Section 3.3 to defining the security goals. In Sections 3.4 and 3.5 we then establish a confidentiality property and an authentication property.

3.1 The Protocol

We regard UM as involving several events that were left implicit in the introduction. We summarize this in Fig. 1. The top and bottom rows shows the



$k_A = \mathsf{k_{um}}(a, Y_B, x, R_B)$ $\qquad$ $k_B = \mathsf{k_{um}}(b, Y_A, y, R_A)$

$c_P = [\![\, \mathsf{cert}\ Y_P,\ P \,]\!]_{\mathsf{CA}}$ $\qquad$ $d(P, P') = \mathsf{keyrec}\ P,\ P',\ k_P$

Fig. 1. The UM Protocol

initiator's actions and the responder's actions, respectively. In the initiator we have successively:

1. A neutral node consulting its principal's local state to obtain its own name, long term secret DH value, and certificate;
2. A transmission node sending R_A, where R_A is g^x for a freshly chosen value x;
3. A reception node receiving the ephemeral value R_B;
4. A reception node for the peer's certificate c_B associating Y_B with B's identity; and
5. A neutral node that deposits the principals' names and the resulting session key as a new *key record* into the principal's local state database.

The responder's actions are identical except that in place of nodes 2 and 3, we have their duals; i.e. a reception node receiving some ephemeral value R_A, followed by a node transmitting R_B, where R_B is g^y for a freshly chosen value y.

3.2 The Certificate Authority

We identify a protocol role for certificate authorities also. It receives a request containing the principal name P and the public value Y_P, and may then emit a public key certificate c_P.

It does so after some procedure we will not represent, which is intended to ensure that the principal P possesses an a such that $g^a = Y$. The same Y should never be certified with two different principals P, P', lest a participant A not know whether a particular session key was shared with P or P'. For instance, if a priest had the same public value Y as a district attorney, a confession meant for one might be received by the other. However, if each principal chooses his long term secret at random from a set far larger than the set of principals, then this event is unlikely.

To request certification, a compliant principal transmits:

$$\bullet \xrightarrow{\mathsf{cert\_req}\ P,\ g^a} ,$$

having made sure to choose a freshly generated long term secret a. Thus, whenever we have a certification request from a compliant principal, we will assume that a is non-originating, and that g^a originates uniquely at this transmission. The CA's (unrepresented) procedure is intended to ensure P has met these conditions, before the CA emits the certificate.

Even if a CA cannot check for all collisions, we can still avoid the bad consequences of colliding certificates by altering the protocol. We could include an additional step of key confirmation, using a different hash function H' to generate a confirmation key $k' = H'({Y_A}^b,\ {R_A}^y)$. By exchanging messages containing a MAC of the ephemeral values and principal identities, each principal can ensure that no confusion has occurred.

$$A:\ \bullet \xrightarrow{\mathsf{mac}_{k'}(3,\ A,\ B,\ g^x,\ R_B)} \qquad \xleftarrow{\mathsf{mac}_{k'}(2,\ B,\ A,\ R_a,\ g^y)} \bullet :\ B \tag{8}$$

Instead of the key confirmation messages of Eqn. 8, one can also diversify the session key using the intended principal identities:

$$k = H(A,\ B,\ {Y_B}^a,\ {R_B}^x) = H(A,\ B,\ {Y_A}^b,\ {R_A}^y).$$

This key computation prevents the district attorney (if complying with the protocol) from receiving a message intended for the priest.

The CA can ascertain whether Y is a genuine group element via the little Fermat theorem, ensuring that there exists an a such that $Y = g^a$, and in particular that $Y \neq g^1$. We embody these comments in some assumptions about the client and server behaviors in certification:

Principal Certification. When a compliant principal P successfully requests a certificate with long term public value Y, we will assume that Y uniquely originates with this request. Since the protocol requires that Y is of the form g^a, this means that P has chosen a value a independent of other values chosen elsewhere. Thus, we will also assume that a is non-originating.

Certificate Authority. A certificate $c_P = [\![\ \mathsf{cert}\ Y,\ P\ ]\!]_{\mathsf{CA}}$ is useful only if certificates cannot be constructed by the adversary, i.e. the Certificate Authority's signature key K_{CA} is uncompromised. We will assume that $\mathsf{sk}(K_{\mathsf{CA}})$ is non-originating.

We also assume that the CA ensures that there exists an a such that: (1) $a \neq 1$; (2) $Y = g^a$; and (3) if a is non-originating and there is also another certificate

$$c_{P'} = [\![\mathsf{cert}\ Y,\ P']\!]_{\mathsf{CA}}$$

for the same Y, then $P = P'$.

Any other party has a negligible likelihood of selecting the same a for any purpose, and the adversary has a negligible likelihood of inferring it from the later usage of Y. The CA establishes (3) by a proof-of-possession interaction. If $Y = g^v$, and a principal P possesses v, then there are only two cases. One is that P is compliant, in which case v is a freshly chosen value a, and thus with overwhelming probability distinct from any previously chosen a'. The other case is that P in non-compliant ("the adversary"), in which case in our model v is not non-originating: the adversary transmits v to itself to perform operations with it.

3.3 Security Properties of UM

In particular, an adversary cannot obtain a key shared with a compliant A or B, who has engaged in a session apparently with a B or A, if the latter's long term secret is used only in accordance with the protocol. Writing A for either the initiator or responder, we have the confidentiality property:

Security Goal 1 (UM Confidentiality) *Suppose that A ran a session yielding $k = H({Y_B}^a,\ {R_B}^y)$, and $a, b \neq 1$ are non-originating.*
Then k is not disclosed to the adversary.

The ephemeral values x, y make the keys in different sessions independent, so that key re-establishment for A, B is still possible even if a session key becomes compromised. We will discuss such properties later in a separate paper.

Moreover, we also have an authentication property. If two compliant principals A, B obtain the same key k, then each believes it to be shared with the other:

Security Goal 2 (UM Authentication) *Suppose A, B both ran sessions yielding*

$$H({Y_D}^a,\ {R_D}^x) = k = H({Y_C}^b,\ {R_C}^y),$$

and A and B received certificates

$$[\![\mathsf{cert}\ Y_D,\ D]\!]_{\mathsf{CA}} \quad \textit{and} \quad [\![\mathsf{cert}\ Y_C,\ C]\!]_{\mathsf{CA}}$$

respectively. If a, b are non-originating, then $C = A$ and $D = B$.

This is a stronger authentication property than Blake-Wilson and Menezes claim [2, Sec. 4.3], in reference to MQV, namely, "implicit key authentication is only considered in the case where B engages with an honest entity (which E isn't)."[2] In our terminology, their claim means,

[2] See also [23, Sec. 5.2], where the same point is made.

Security Goal 3 (UM Weak Authentication) *Suppose A, B both ran sessions yielding*

$$H({Y_D}^a,\ {R_D}^x) = k = H({Y_C}^b,\ {R_C}^y),$$

and A and B received certificates

$$[\![\,\mathsf{cert}\ Y_D,\ D\,]\!]_{\mathsf{CA}} \quad \textit{and} \quad [\![\,\mathsf{cert}\ Y_C,\ C\,]\!]_{\mathsf{CA}}$$

respectively. If a, b are non-originating, and if some c is non-originating such that $g^c = Y_C$, then $C = A$ and $D = B$.

So in this goal, A, B, *and* C, namely B's apparent peer, must be assumed compliant. Using our stronger goal, even a non-compliant E cannot cause authentication failure. It is in this sense that we have discovered a more exact property that a suitable CA definition can achieve.

3.4 Confidentiality for UM

The confidentiality property Goal 1 states that the adversary cannot obtain a session key in a session satisfying reasonable assumptions.

Suppose that B has engaged in a run of UM as responder. Moreover, suppose that the other parameters of the run are:

$$A, R_A, y, Y_A, b, \mathsf{CA}$$

where, Y_A having been certified, we will infer that Y_A is of the form g^a. Then if $a, b \neq 1$ are non-originating, k is not disclosed to the adversary.

Proof sketch. If there were such a node n on which k is transmitted, and thereby disclosed to the adversary, it would be either a regular node or an adversary node. No regular node, however, transmits a value of this form. Thus, n would be an adversary node. However, in that case, the adversary must have constructed $H({Y_A}^b,\ {R_A}^y)$ from an earlier transmission of ${Y_A}^b,\ {R_A}^y$, which in turn must have been constructed from earlier transmissions of ${Y_A}^b$ and ${R_A}^y$.

1. No regular node can transmit ${Y_A}^b$.

In particular, Y_A having been certified, we have from CA assumption (1) that $Y_A \neq g^1$, hence ${Y_A}^b \neq Y_B$. Moreover, since $Y_B = g^b$ was certified, $b \neq 1$, whence ${Y_A}^b \neq Y_A$. Thus ${Y_A}^b$ was not transmitted on the certification nodes for A, Y_A or B, Y_B.

Moreover, ${Y_A}^b$ was not transmitted on any other certification request by a compliant principal. A compliant principal selects a value g^c, but we know that Y_A is of the form g^a, so that ${Y_A}^b = g^{ab}$, hence distinct from any g^c at all.[3] Nor, for the same reason, does ${Y_A}^b$ originate as the ephemeral value of any compliant strand.

[3] We formalize this below, Section 4. However, a, b being choices made independently of each other and of c, betting on $ab = c$ is a strategy that loses for the adversary with overwhelming probability.

2. No adversary action can produce ${Y_A}^b$. Although the adversary has the values $Y_A = g^a$ and $Y_B = g^b$, in order to incorporate b into the exponent of g^a, or in order to incorporate a into the exponent of g^b, the adversary would have to have access to a (non-originating) value a or b. ///

In a world with no compromise, x, y would have no role in ensuring confidentiality. Their only role is to ensure that the confidentiality of the key for one secure conversation is independent of the compromise of other session keys. We will not pause here to establish this in more detail.

In step 2 we use a central principle, which we will formalize in a stronger form in Section 5. For now, we codify it as:

Principle 4 *If the adversary originates g^e, then $e = e_1 \cdot e_2$ where both g^{e_1} and e_2 have been previously transmitted.*

Principle 4 also leads to a result when A has engaged in a full run of UM as initiator similar to this result about B as responder.

3.5 Authentication for UM

The authentication goal for UM, Goal 2 states that, on reasonable assumptions, if two compliant principals agree on a key, then they agree on their identities.

Suppose that A, B have engaged in a full runs of UM as initiator, yielding

$$H({Y_D}^a,\ {R_D}^x) = k = H({Y_C}^b,\ {R_C}^y),$$

and A and B received certificates

$$[\![\, \mathsf{cert}\ Y_D,\ D \,]\!]_{\mathsf{CA}} \quad \text{and} \quad [\![\, \mathsf{cert}\ Y_C,\ C \,]\!]_{\mathsf{CA}}$$

respectively. Since Y_C, Y_D have been certified, we will infer that they are of the form g^c, g^d. If a, b are non-originating, then $C = A$ and $D = B$.

A may have executed either an initiator session or a responder session. The protocol permits two principals, each of whom starts a session as initiator, to successfully complete a run with each other. However, if one executes a local session as responder, the other must have acted as initiator.

Proof sketch. Since B's session yields $k = H(g^{ab},\ g^{xy})$ and $Y_D^b = g^{ab}$, we may infer that B's private value is b. Since A's session yields $k = H(g^{ab},\ g^{xy})$ and $Y_D^a = g^{ab}$, $Y_D = g^b = Y_B$. Since the CA certifies Y_C with at most one principal's identity, and since B's session involves the certificate $[\![\, \mathsf{cert}\ Y_B,\ B \,]\!]_{\mathsf{CA}}$, it follows that $D = B$.

The other result is similar. ///

4 A Theory of Groups with Exponentiation

In this section we present an axiomatization of a theory of equational theory of groups with exponentiation. It is appropriate to say "a theory" rather than "the

theory" for the following reason. Our goal is to model the actions of the adversary as the construction of terms built from terms that he receives as messages. From this perspective the operations of the signature embody the computational power that we model for the adversary. For example the adversary should be able to multiply two values he possesses, and take inverses as well. He should be able to do exponentiation, but *not* retrieve an exponent from a value that might have been constructed and sent as an exponentiation. This of course is a reflection of the Computational Diffie-Hellman Assumption.

This assumption is reflected in a completely straightforward way in our formalism. Namely, we do not provide a logarithm function in our signature.

Capturing adversary ability by writing terms is the standard insight underlying symbolic analysis of protocols. When terms are considered modulo some equations, of course syntactic analysis is more complex. Algebraically rich settings such as those of interest in this paper call for careful consideration of which equations to work with. A crucial point is that we want to model what the adversary can do *uniformly* over all values of q. The sense in which equations $\mathsf{AG}^{\wedge}$ below capture this uniformity is the content of Theorem 13.

Definition 5. *The theory* $\mathsf{AG}^{\wedge}$ *is the equational theory determined by the following data.*

- *Three sorts* G, E, *and* NZE, *with* NZE *a subsort of* E.
- *Operators:*

$$\begin{aligned} \circ &: G \times G \to G \\ id &:\to G \\ inv &: G \to G \end{aligned}$$

$$\begin{aligned} * &: E \times E \to E \\ 1 &:\to NZE \\ i &: NZE \to NZE \end{aligned}$$

$$\begin{aligned} exp &: G \times E \to G \\ bar &: G \to NZE \\ ** &: NZE \to NZE \end{aligned}$$

- *Equations: as given in Table 1*

The inclusion of the operator $**$ is a device for ensuring that the sort NZE is closed under ordinary E-multiplication (by writing the equation that states that $**$ coincides with $*$ on NZE).

Note that no equations involve bar. So the operations in G and in E are not bound to be related in any way.

4.1 Rewriting with theory $\mathsf{AG}^{\wedge}$

In order to work with the theory $\mathsf{AG}^{\wedge}$ we will present it as a rewrite theory, orienting the equations of $\mathsf{AG}^{\wedge}$. To achieve confluence we will also add new rules,

$(G, \circ, i, id)$ is an abelian group

$$(a \circ b) \circ c = a \circ (b \circ c)$$
$$a \circ b = b \circ a$$
$$b \circ id = b$$
$$b \circ inv(b) = id$$

$(E, +, 0, -, *, \mathbf{1}, i)$ is a commutative unitary ring

$$(x + y) + z = x + (y + z)$$
$$x + y = y + x$$
$$x + 0 = x$$
$$x + (-x) = 0$$
$$(x * y) * z = x * (y * z)$$
$$x * y = y * x$$
$$x * (y + z) = (x * y) + (z + z)$$
$$x * 1 = x$$

Multiplicative inverse is defined for elements of sort NZE

$$i(u * v) = i(u) * i(v)$$
$$i(1) = 1$$
$$i(i(w)) = w$$

Exponentiation makes G a unitary right E-module

$$a^{x * y} = (a^x)^y$$
$$a^1 = a$$
$$(a \circ b)^x = a^x \circ b^x$$
$$a^{(x+y)} = a^x \circ a^y$$
$$id^x = id$$

NZE is closed under multiplication

$$u * *v = u * v$$

Table 1. The theory $\mathsf{AG}^{\hat{}}$

corresponding to equations that are derivable from $\mathsf{AG}^{\hat{}}$ yet are are necessary to join critical pairs.

Definition 6. *Let R be the set of rewrite rules given in Table 2. The rewrite relation $\rightarrow_{\mathsf{AG}^{\hat{}}}$ is rewriting with R modulo associativity and commutativity of $id, +$, and $*$.*

Termination and Irreducible Forms

Lemma 7. *The reduction $\rightarrow_{\mathsf{AG}^{\hat{}}}$ is terminating and confluent modulo AC.*

At sort G			At sort E		
$b \circ id$	$\to$	b	$x + 0$	$\to$	x
$b \circ inv(b)$	$\to$	id	$x + (-(x))$	$\to$	0
$inv(id)$	$\to$	id	$x * (y + z)$	$\to$	$(x * y) + (x * z)$
$inv(a \circ b)$	$\to$	$inv(a) \circ inv(b)$	$x * 1$	$\to$	x
$inv(inv(b))$	$\to$	b	$-(0)$	$\to$	0
id^x	$\to$	id	$-(x + y)$	$\to$	$-(x) + (-(y))$
$(a \circ b)^x$	$\to$	$(a^x) \circ (b^x)$	$-(-(x))$	$\to$	x
$(inv(a))^x$	$\to$	$inv(a^x)$	$0 * x$	$\to$	0
$(a^x)^y$	$\to$	$a^{(x * y)}$	$-(x) * y$	$\to$	$-(x * y)$
a^0	$\to$	id	At sort NZE		
$a^{(x+y)}$	$\to$	$(a^x) \circ (a^y)$	$u * *v$	$\to$	$u * v$
$a^{-(x)}$	$\to$	$inv(a^x)$	$i(u * v)$	$\to$	$i(u) * i(v)$
a^1	$\to$	a	$i(1)$	$\to$	1
			$i(i(w))$	$\to$	w

Table 2. Rewrite rules for $\to_{\mathsf{AG}^\wedge}$

Proof. Termination can be established using the AC-recursive path order defined by Rubio [26] with a precedence in which exponentiation is greater than inverse, which is in turn greater than multiplication (and 1). This has been verified with the Aprove termination tool [13].

Then confluence follows from local confluence, which is established via a verification that all critical pairs are joinable. We do not give details of the critical-pair checking here, in part because confluence is a direct consequence of Theorem 13 below.

Lemma 8. *If $e : E$ is irreducible then e is a sum $(m_1 + \ldots + m_n)$ where each m_i is of the form*

$$e_1 \ * \ \ldots \ * \ e_k \qquad k \geq 0$$

where

- *the case $n = 0$ is taken to mean $e = 0$*
- *the case $k = 0$ is taken to mean $m_i = 1$*
- *no e_i is of the form $i(e_j)$, and*
- *each e_i is one of:*

$$x \qquad i(x) \qquad [t] \qquad i([t])$$

where x is a G-variable and $t : G$ is a G-normal form

We will call terms of the form m_i above irreducible monomials

Lemma 9. *If $t : G$ is irreducible then t is a product*

$$t_1 \circ \ldots \circ t_n \qquad n \geq 0$$

where

- *the case $n = 0$ is taken to mean $t = id$*
- *no t_i is of the form $inv(t_j)$*
- *each t_i is one of:*

$$v \qquad inv(v) \qquad v^e \qquad inv(v^e)$$

where v is a G-variable $e : E$ is an irreducible monomial.

Note that in an irreducible G-term, the symbols $+$ and 1 do not occur.

4.2 A Completeness Result for AGˆ

The following models for the language of AGˆ will be of interest.

Definition 10.

- *For prime q, the model $\mathcal{M}_q$ is given by the following data.*
  Let $\mathbb{Z}_q$ denote the additive group of integers mod q and $\mathbb{F}_q$ denote the field of order q (with domain $\{0, 1, \ldots, q-1\}$). The interpretation of the sort G is $\mathbb{Z}_q$, (that, is, with $\circ$ interpreted as addition), the interpretation of the sort E is $\mathbb{F}_q$, with the arithmetic operators are interpreted in the canonical way, and the interpretation of the sort NZE is the set of non-0 elements of $\mathbb{F}_q$. The "bar" function symbol $[\cdot]$ is interpreted by the following function

$$[a] = \begin{cases} a & \text{if } a \neq 0 \\ 1 & \text{if } a = 0 \end{cases}$$

  Here we exploit the fact that the underlying sets for $\mathbb{Z}_q$ and $\mathbb{F}_q$ are the same.
- *If D is a non-principal ultrafilter over the set of prime numbers, $\prod_D \mathcal{M}_q$ denotes the ultraproduct structure $\prod_D \{\mathcal{M}_q \mid q \text{ prime }\}$. Here we view each $\mathcal{M}_q$ as an ordinary structure in unsorted first-order logic, treating the sorts G, E, and NZE as unary predicates.*
- *The model $\mathbb{Q}^{[]}$ comprises the additive group of rational numbers considered as a vector space over itself, with the "bar" function symbol is interpreted via the same recipe as for the $\mathcal{M}_q$. That is, the interpretation of the sort G is $\mathbf{Q}$, (with $\circ$ interpreted as addition), the interpretation of the sort E is $\mathbf{Q}$, with the arithmetic operators are interpreted in the canonical way, and the interpretation of the sort NZE is the set of non-0 elements of $\mathbb{F}_q$.*

For the notion of ultraproduct of structures see, for example, [6]; the crucial facts about ultraproducts for our purposes are (i) a first-order sentence is true in $\prod_D \mathcal{M}_q$ if and only if the set of indices at which it is true is a set in D, and (ii) when D is non-principal, every cofinite set is in D.

Lemma 11. *The structure $\mathbb{Q}^{[]}$ can be embedded as a submodel of $\prod_D \mathcal{M}_q$.*

Proof. Each $\mathcal{M}_q$ is a field, so $\prod_D \mathcal{M}_q$ is a field, since ultraproducts preserve all first-order sentences. The characteristic of $\prod_D \mathcal{M}_q$ is 0, since for each prime p, the equation $1 + 1 + \cdots + 1 = 0$ (p occurrences of 1) holds only at $\mathcal{M}_p$, and D contains no finite sets. Since $\mathbf{Q}$ is the prime field of characteristic 0, it is embeddable in $\prod_D \mathcal{M}_q$, and this field embedding respects the function $[\cdot]$, since $[\cdot]$ is definable by the same first-order formula in $\mathbb{Q}^{[]}$ and in each $\mathcal{M}_q$. ///

Lemma 12. *If $t : G$ is in normal form, $t \not\equiv id$, then there exists $\eta : Vars \to (\mathbf{Q} \setminus \{0\})$ such that for every subterm t' of t, $\eta(t') \neq 0$ in $\mathbb{Q}^{[]}$.*

Proof. First note that in an irreducible term not identically *id* or 0, the constants *id* and 0 do not occur. Now, in the structure $\mathbb{Q}^{[]}$, exponentiation is interpreted as multiplication, so it suffices to consider the expression obtained by replacing $\circ$ and *inv* by $+$ and $-$, and the exponentiation operator by $*$, and viewing t as an ordinary rational expression over the rationals, albeit with the $[\cdot]$ operator still allowed to appear.

First consider the case where there are no occurrences of $[\cdot]$. Then we are concerned with an ordinary rational expression t in several variables $x_1, \ldots, x_k$, not identically 0. We may view t as determining a real function $f_t : \mathbb{R}^k \to \mathbb{R}$. In fact each subterm t' of t similarly determines a function from $\mathbb{R}^k$ to $\mathbb{R}$. Let $U \subseteq \mathbb{R}^k$ be the set of points $p = (p_1, ...p_k)$ such that $f_{t'}(p) \neq 0$ for every subterm t' of t. Since t is not identically 0 and is irreducible, U is not empty, and, as the preimage of $\mathbb{R} \setminus \{0\}$, is open. So U contains a rational point $r = (r_1, ..., r_k)$ with each $f_{t'}(r) \neq 0$. We take η to map each x_i to r_i.

Now for the general case, when $[\cdot]$ subterms can arise, consider the term t_1 obtained from t by simply forgetting the $[\cdot]$ operator, that is, treating it as the identity function. The η that works for t_1 will in fact suffice for the original term t. This is because, for each subterm $[u]$ of t, the value of $\eta(u)$ in $\mathbb{Q}^{[]}$ will not be 0, so the value of $[\eta(u)]$ in $\mathbb{Q}^{[]}$ will be $\eta(u)$. ///

The next theorem characterizes the sense in which the theory $\mathsf{AG}^{\hat{}}$ functions as an analysis of uniform equality.

Theorem 13. *For each pair of G-terms s and t, the following are equivalent*

1. $\mathsf{AG}^{\hat{}} \vdash s = t$
2. *the equation $s = t$ holds in every model based on $\mathbb{Z}_q$, $\mathbb{F}_q$ where the $[\cdot]$ is interpreted as any function from $\mathbb{Z}_q$ to the non-0 elements of $\mathbb{F}_q$*
3. *the equation $s = t$ holds in every model $\mathcal{M}_q$*
4. *for infinitely many q, the equation $s = t$ holds in $\mathcal{M}_q$*
5. *there exists an ultrafilter D such that $\prod_D \mathcal{M}_q \models s = t$*
6. $\mathbb{Q}^{[]} \models s = t$
7. *if s reduces to s' with s' irreducible, and t reduces to t' with t' irreducible, then s' and t' are identical modulo associativity and commutativity of $\circ$, $+$, and $*$.*

Proof. It suffices to establish the cycle of entailments 1 implies 2 implies 3 ... implies 7 implies 1. The first three of these are immediate.The implication 4 implies 5 follows from the fact that for any infinite set I there exists an ultrafilter D containing I. The fact that 5 implies 6 follows from the fact that $\mathbb{Q}^{[]}$ can be embedded as a submodel of $\prod_D \mathcal{M}_q$. For 6 implies 7, suppose that $\mathbb{Q}^{[]} \models s = t$ and that s and t reduce to irreducible terms s' and t' respectively. Form the term $u \equiv s' \circ inv(t')$. By hypothesis $\mathbb{Q}^{[]} \models u = id$. By Lemma 12, any irreducible form of $s' \circ inv(t')$ is id. Since s' and t' are themselves irreducible it must be the case that $s' \equiv t'$.

It is immediate that 7 implies 1. ///

As a corollary of Theorem 13 we note that the above equivalences hold for E-term equations as well. Since: given a pair of terms e and e', we may form the equation $g^e = g^{e'}$, and note that this is provable iff $e = e'$ is provable, and is true in a given model $\mathcal{M}$ iff $e = e'$ is.

Another consequence of Theorem 13—specifically the implication from 1 to 7—is the fact that the reduction relation $\rightarrow$ is Church-Rosser. This discharges the proof obligation remaining from Lemma 7

Discussion. The equivalence of $\mathsf{AG}\hat{}$-provability with equality in the models $\mathcal{M}_q$ (items 2, 3, 4 in Theorem 13) is the technical content of our claim that $\mathsf{AG}\hat{}$ captures "uniform equality" between terms with exponentiation.

On the other hand the model $\mathbb{Q}^{[]}$ is very convenient technically: the fact that a single model serves to witness uniform equality can simplify analyses. See for example the analysis of MQV in Section 6 below. The theory of fields has no equational presentation, and indeed there are no free fields (every non-trivial field homomorphism is a monomorphism). The theory $\mathsf{AG}\hat{}$ has free models, of course, its term-models, but these are not fields. So the field $\mathbb{Q}^{[]}$ serves as a sort of stand-in for a field "term-model" in the sense that it that witnesses precisely the equations that hold in all of the finite fields.

It follows from results of Ax [1] that the model $\prod_D \mathcal{M}_q$ satisfies precisely those *first-order* sentences in the language of field theory true in each of the $\mathcal{M}_q$ (equivalently, in all but a finite number of the $\mathcal{M}_q$).

4.3 Indicators for terms

The notion of *indicator* is our key technical device for analyzing terms. Indicators are an invariant that allow us to measure the ways that certain atoms (for our purposes, non-originating values) are "present" in a term. They are a refinement of the standard notion of an atom occurring term, a necessary refinement given the fact that terms are considered modulo equations.

The *basic terms* are the terms built from atoms by the group operations and exponentiation. We will define indicators for all terms t. Let $\mathbf{Z}^k$ denote the set of all k-tuples of integers. For intuition about the following definition, think of N as being a set of *non-originating values* for a protocol. If m is a monomial occurring as a subterm of a term t, say that m is "maximal" if it occurs as an

exponent in t, that is, if the smallest subterm of t of which m is a proper subterm is of the form b^m.

Definition 14. *Fix a vector $N = \langle v_1, \ldots, v_d \rangle$ of E-variables. If m is an irreducible monomial, the N-*indicator vector *for m is $\langle z_1, \ldots, z_k \rangle$ where z_i is the multiplicity of v_i in m.*

If t is an arbitrary term in normal form, the set $\mathrm{Ind}_N(t)$ is the collection of all indicator vectors of the monomials m occurring as maximal-monomial subterms of t.

Example: For $N = \{x, y\}$, if t is

$$g^{x\ i(y)} \cdot g^{zx[g^x]} \cdot g^{xx[g^y]}$$

$\mathrm{Ind}_N(t) = \{\langle 1, -1\rangle, \langle 1, 0\rangle, \langle 2, 0\rangle\}$

Definition 15. *If $T = \{t_1, \ldots, t_k\}$ is a set of terms then the set $\mathrm{Gen}(T)$ of terms* generated *by T is the least set containing T and closed under the term-forming operations.*

Theorem 16. *Suppose T is a collection of terms such that every $e \in T$ of sort E is N-free. Then*

1. *every $e \in \mathrm{Gen}(T)$ of sort E is N-free, and*
2. *if $u \in \mathrm{Gen}(T)$ is of sort G and $\boldsymbol{z} \in \mathrm{Ind}(u)$ then for some $t \in T$, $\boldsymbol{z} \in \mathrm{Ind}(t)$.*

Proof. The proof of each assertion is by induction on the number of operations used to construct terms from elements of T.

The interesting cases are when u is of the form $u_1 u_2$ or t^e where t, u_1, u_2 and e are each in $\mathrm{Gen}(T)$. We may assume that these latter terms are in normal form.

In the first case, then, u is a product

$$t_1 \circ \ldots \circ t_n$$

where each factor comes from u_1 or u_2. Since each t_i is of the form v, $inv(v)$, v^e, or $inv(v^e)$ the normal form of this term is simply the result of cancelling any factors (from different u_i) that are inverses of each other. Clearly no new E-subterms are created by this, so no new indicator vectors are created, and our assertion follows.

The other case is when u is t^e. Note that since e is in $\mathrm{Gen}(T)$ we know that e is N-free. It suffices to show that $\mathrm{Ind}(t^e) = \mathrm{Ind}(t)$. We may assume that t is in normal form as usual, so that t^e is

$$(t_1)^e \circ \ldots \circ (t_n)^e$$

Each $(t_i)^e$ is of the form

$$v^e \quad (i(v))^e \quad (v^{e'})^e \quad (inv(v^{e'}))^e$$

The first two terms are N-free. The second kind of term reduces to $v^{e * e'}$, and the indicator set for this term is precisely $\mathrm{Ind}(e)$ since e' is N-free. The last term reduces to $inv(v^{e' * e})$ and we can argue just as in the previous case.

The cases for the operations of concatenation and encryption are immediate applications of the induction hypothesis, since these non-algebraic constructions simply perpetuate indicator vectors freely through terms.

5 A Limitation of the Adversary

As described in the Introduction, an adversary strand consists of zero or more reception nodes followed by a transmission node. Formally we have

Definition 17. *An* adversary strand *is a strand of one of the following forms:*

- *Emission of a basic value or indeterminate* a*:* $\langle +a \rangle$
- *A constructor strand:* $\langle -a_1 \Rightarrow \ldots \Rightarrow -a_n \Rightarrow +t \rangle$ *where is* t *is in* $\mathrm{Gen}(a_1, \ldots, a_n)$
- *A destructor strand* $\langle -t \Rightarrow +s_1 \ldots \Rightarrow +s_n \rangle$ *where* t *is a concatenation of the values* s_i*.*
- *An encryption strand* $\langle -K \Rightarrow -t \Rightarrow +\{\!|t|\!\}_K \rangle$
- *An decryption strand* $\langle -K^{-1} \Rightarrow -\{\!|t|\!\}_K \Rightarrow +t \rangle$

An adversary web *constructed from a collection of node-disjoint adversary strands* $S_1, \ldots, S_k$ *is an acyclic graph whose nodes are the nodes of the* S_i *with an edge from node* n *to node* n' *if either (i)* $n \Rightarrow n'$ *on some strand or (ii)* n *is a transmission node,* n' *is a reception node, and* $\mathsf{msg}(n) = \mathsf{msg}(n')$*.*

The crucial aspect of an adversary web is that no message received can contain a non-originating value.

Theorem 18. *Let* N *be the set of non-originating variables in a bundle* B *and let* W *be an adversary web of* B*. If* u *is a message transmitted on this web, then every* N*-indicator at level* k *in* u *is an* N*-indicator at a level* $k' \leq k$ *in some message transmitted by a regular strand.*

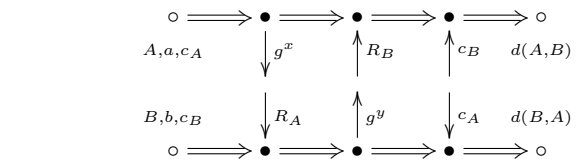
Proof. Let T_R be the set of messages received on W, and let T_M be the set of basic values emitted by W; set $T = T_R \cup T_M$. The message u is in $\mathrm{Gen}(T)$. The set T_R is N-free, as a consequence of the fact that every message received on W must have originated, and T_M is N-free since it is a set of basic values not in N (indeed, each term in T_M has an empty indicator set). So Theorem 16 applies, and any N-indicator at level k in u is an N-indicator at a level $k' \leq k$ in some message of T. Since each $t \in T_M$ has empty indicator set we conclude that every indicator in u comes from a message in T_R, as desired.

Corollary 19. *Let* N*,* B*, and* W *be as in Theorem 18. If* $\boldsymbol{z}$ *is a level-0 indicator in a message transmitted on* W *then* $\boldsymbol{z}$ *is a level-0 indicator of some message transmitted by a regular strand.*

Principle 4 from Section 3.1 is an immediate consequence of the theorem.

6 Completing the MQV Analysis

The MQV message flow (see Fig. 2) is the same as in UM. Only the key computation function $\mathsf{k}_{\mathsf{mqv}}(a, Y, x, R) = (R \cdot Y^{[R]})^{(x+a[g^x])}$ differs.



$c_P = [\![\, \mathsf{cert}\ Y_P,\ P \,]\!]_{\mathsf{CA}}$ $\qquad$ $d(P, P') = \mathsf{keyrec}\ P,\ P',\ \mathsf{k}_{\mathsf{mqv}}(\cdots)$

Fig. 2. The MQV Protocol Message Flow

Confidentiality of MQV. MQV is more challenging to analyze than UM, because it depends more intimately on the structure of the algebra. UM computes its key $\mathsf{k}_{\mathsf{um}}(a, Y, x, R) = H(Y^a,\ R^x)$, so there is no algebraic interaction between the long-term values and the ephemeral values. By contrast, the MQV key computation $\mathsf{k}_{\mathsf{mqv}}(a, Y, x, R) = (R \cdot Y^{[R]})^{(z+c[g^z])}$ mixes the long and short term values c, z and R, Y. In the normal form of Section 4, we have:

$$\mathsf{k}_{\mathsf{mqv}}(a, Y, x, R) = R^x \cdot R^{a[g^x]} \cdot Y^{x[R]} \cdot Y^{a[g^x][R]}.$$

Consider the case in which $Y = g^b$ and $a, b \neq 1$ are non-originating. Then we have:

$$\mathsf{k}_{\mathsf{mqv}}(a, g^b, x, R) = R^x \cdot R^{a[g^x]} \cdot g^{bx[R]} \cdot g^{ab[g^x][R]}.$$

Observe, if we take indicators relative to the vector of variables $\langle a, b \rangle$, then the last term gives an indicator of $\langle 1, 1 \rangle$. However, no regular node ever transmits a message with two non-zero positions in any indicator vector. Moreover, adversary strands never create new indicators. This immediately implies that the key cannot be disclosed:

Security Goal 20 (MQV Confidentiality) *Suppose that A ran a session yielding $k = \mathsf{k}_{\mathsf{mqv}}(a, g^b, x, R)$, and $a, b \neq 1$ are non-originating.*

Then k is not disclosed to the adversary.

Authentication in MQV. Let us turn now to the authentication goal for MQV. Here we would like to show that, when two compliant principals A, B share the same key at the end of a session, then each knows the identity of the other. This is in fact not true for MQV without a further constraint on the certification authority, as Kaliski points out [18].

The problem is that an adversary, observing A's ephemeral public value R_A, may generate a new ephemeral value R_E as well as a long-term value Y_E, which depend on R_A and Y_A:

$$R_E = R_A \cdot (Y_A)^{[R_A]} \cdot g^{-1} \qquad Y_E = g^{[R_E]^{-1}}$$

The adversary asks CA to certify the value Y_E, for which it can successfully prove possession of the exponent. The resulting certificate leads B to think that the key is shared with E, when in fact it is shared with A. The inverses cause E's operations to cancel out. Thus, a mischievous priest E can induce a criminal to confess to a compliant district attorney A, which could have led to an unexpected plot twist in the Hitchcock movie with Montgomery Clift [17]. We prove this scenario depends essentially on E's ability to incorporate an ephemeral value into E's certified long-term value.

The proof again relies on the invariance of indicator vectors (Cor. 19).

Consider indicators expressed in terms of the variable vector $\langle a, b, x, y\rangle$. Thus, $Y_A = g^a$ has indicator $\{\langle 1,0,0,0\rangle\}$, and $Y_B = g^b$ has indicator $\{\langle 0,1,0,0\rangle\}$. Every certified long-term public value not involving these variables has indicator $\{\langle 0,0,0,0\rangle\}$.

Security Goal 21 (MQV Authentication) *Suppose sessions of A, B yielded*

$$\mathsf{k}_{\mathsf{mqv}}(a, Y_D, x, R_D) = k = \mathsf{k}_{\mathsf{mqv}}(b, Y_C, x, R_C),$$

respectively, and A and B received certificates

$$[\![\, \mathsf{cert}\ Y_D,\ D \,]\!]_{\mathsf{CA}} \quad \textit{and} \quad [\![\, \mathsf{cert}\ Y_C,\ C \,]\!]_{\mathsf{CA}}$$

respectively. Suppose that the indicators of Y_C, Y_D are each among

$$\{\langle 1,0,0,0\rangle\}, \quad \{\langle 0,1,0,0\rangle\}, \quad \textit{and} \quad \{\langle 0,0,0,0\rangle\}.$$

If a, b are non-originating, then $C = A$ and $D = B$.

We will not want to instantiate any of the non-originating values a, b, x, y, since this would reflect an unreasonable constraint on the behavior of the regular participants.

If there were a counterexample to our claim—an execution in which the premises are satisfied, but the conclusion is not true—then the adversary uses some strategy to determine what messages to prepare and deliver hoodwink the regular participants. Our *uniformity* assumption implies that this strategy continues to work as the underlying algebra varies (i.e. as q increases), as the function $[\cdot]$ varies, and as the specific parameters of the run vary. That is, the adversary strategy may be specified using algebraic expressions of in the signature of $\mathsf{AG}^{\wedge}$. Moreover, whenever the adversary selects a value to deliver to a regular participant, the theory $\mathsf{AG}^{\wedge}$ must entail that the value computed by the adversary equals the value expected by the recipient.

Proof. We would like to show that the sessions agree on the long-term secrets, i.e. that $a = d$ and $b = c$, so by our CA assumption (3), $A = D$ and $B = C$.

Reducing k_A to normal form, we obtain:

$$g^{xz} \cdot g^{az[g^x]} \cdot g^{cx[g^z]} \cdot g^{ac[g^x][g^z]}$$

and k_B has the normal form:

$$g^{yw} \cdot g^{bw[g^y]} \cdot g^{dy[g^w]} \cdot g^{bd[g^y][g^w]}$$

Since we have assumed them equal, we must consider what permutations of the factors can equate the normal forms.

The indicators constrain which combinations are compatible. For instance, g^{xz} has an indicator of the form $\langle ?, ?, 1, ? \rangle$, since x is definitely present with multiplicity 1. However, we cannot be sure that z is a simple value, as it may have been built using a, b, y.

On the other hand, $g^{dy[g^w]}$ has an indicator of the form $\langle ?, 0, 0, 1 \rangle$ since y is definitely present, while b, x are definitely absent. The absence of x here depends crucially on our assumption that the certified value g^d is x-free; including a d in the exponent therefore cannot introduce any dependence on x. Similarly, $g^{bd[g^y][g^w]}$ has an indicator of the form $\langle ?, 1, 0, 0 \rangle$, since b is definitely present, and a is possibly present, as $d = a$ is possible. This reasoning yields the indicators for our terms of interest shown in Table 3. We ask now: what are the possible ways

g^{xz}	$\langle ?, ?, 1, ? \rangle$	g^{yw}	$\langle ?, ?, ?, 1 \rangle$
$g^{az[g^x]}$	$\langle 1, ?, ?, ? \rangle$	$g^{bw[g^y]}$	$\langle ?, 1, ?, ? \rangle$
$g^{cx[g^z]}$	$\langle ?, ?, 1, 0 \rangle$	$g^{dy[g^w]}$	$\langle ?, ?, 0, 1 \rangle$
$g^{ac[g^x][g^z]}$	$\langle 1, ?, 0, 0 \rangle$	$g^{bd[g^y][g^w]}$	$\langle ?, 1, 0, 0 \rangle$

Table 3. Factors and their indicators

that the "?" can be filled in to make the indicators match? A little investigation shows that there is only one solution.

Let us temporarily label the indicators in the left column as L1, L2, L3, and L4, and similarly R1, R2, R3, and R4 for the right column. We first notice that neither of L3 or L4 can match with R1 or R3 because of the conflict at index 4 of the vectors. So we must have {L1, L2 } match with {R1, R3 } and {L3, L4 } match with {R2, R4 }. But L1 cannot be paired with R3 due to the conflict at index 3. Similarly L3 cannot be paired with R4 due to the conflict at index 3.

Thus, the only scenario that matches the indicator variables correlates L1 with R1, L2 with R3, L3 with R2, and L4 with R4. *No other instantiations* are consistent with the information in the indicators. We must have:

$$g^{xz} = g^{yw} \tag{9}$$

$$g^{az[g^x]} = g^{dy[g^w]} \tag{10}$$

$$g^{cx[g^z]} = g^{bw[g^y]} \tag{11}$$

$$g^{ac[g^x][g^z]} = g^{bd[g^y][g^w]} \tag{12}$$

By Eqn. 9, we must identify $g^{xz} = g^{yw}$, but since we may not instantiate x or y, we have (for some v) $w \mapsto xv$ and $z \mapsto yv$. With this information, Eqn. 10 reduces to $a(yv)[g^x] = dy[g^{xv}]$. Eqn. 11 reduces to $cx[g^{yv}] = b(xv)[g^y]$.

If the adversary has any uniform strategy to satisfy these equations, then it must work in particular in the structure $\mathbb{Q}^{[]}$ (see Defn. 10). In that model, the "exponentiation" operator is interpreted by multiplication in the additive group of rationals, while the field operations in E are interpreted in the usual way. The box $[\cdot]$ is the identity (apart from 0). Hence we must have:

$$a(yv)gx = dygxv \tag{13}$$

$$cxgyv = b(xv)gy \tag{14}$$

By cancellation, Eqn. 13 yields $a = d$ and Eqn. 14 yields $c = b$. Eqn. 12 is compatible with this solution.

To return now to the identities of the principals, we know from the first node of A's session that its own certificate took the form $[\![\mathsf{cert}\ g^a,\ A]\!]_{\mathsf{CA}}$. We also know that B received a certificate for its partner having the form $[\![\mathsf{cert}\ g^d,\ D]\!]_{\mathsf{CA}}$. Since we now know $a = d$, the assumption that the CA never recertifies the same long-term public value with different identities implies that $D = A$.

By a symmetric argument about B's session, $C = B$. ///

In fact, in "most" models we will also have $v = 1$, i.e. $z = y$ and $w = x$. By Eqns 10–11, this will hold in all models in which the interpretations of exponentiation and $[\cdot]$ do not compose to form a linear transformation.

7 Conclusion and Related Work

Related Work. Within the symbolic model, there has been substantial work on some aspects of DH, starting with Boreale and Buscemi [4], which provides a symbolic semantics [12, 25] for a process calculus with algebraic operations for DH. The symbolic semantics is based on unification. Goubault-Larrecq, Roger, and Verma [14] use a method based on Horn clauses and resolution modulo AC, providing automated proofs of passive security. Maude-NPA [10, 11] is also usable to analyze many protocols involving DH, again depending heavily on unification. All of these approaches appear to face a fundamental problem with a theory like the $\mathsf{AG}\hat{}$ theory of Section 4, in which it would be unwise to rely on the decidability of the unifiability problem (see Section 4.2).

Küsters and Truderung [22] finesse this issue by rewriting each protocol analysis problem using their AC theory involving exponentiation into a corresponding problem that does not require the AC property, and can work using standard ProVerif resolution [3]. Their approach covers a surprising range of protocols, although, like [7], not implicit authentication protocols.

Conclusion. In this paper, we have applied the strand space framework in a new setting, namely implicitly authenticated DH protocols, specifically UM and MQV. We have established that by choosing among different ordering constraints

on the CA, we can identify different authentication properties that MQV can achieve.

While there are certainly some properties that a computational analysis can clarify, to which our approach is unsuited—beginning with a stronger notion of security—we have also shed light on issues that would have been hard to uncover in computational models.

An important area of future work is to study the computational soundness of our approach, to determine exactly what conclusions (expressed in the computational model) a proof in this model establishes. For DH operations, there has been limited work on computational soundness even in the passive case, with Bresson et al. [5] giving a recent treatment.

Our completeness result in Section 4, Thm. 13, suggests that "uniformly algebraic" attacks are captured in our model. Let us call an attack *uniformly algebraic* if the actions of the adversary consist of a collection of algebraic expressions in the signature of Section 4; for every message received by a regular participant, the adversary uses one of these algebraic expressions to prepare the message to be delivered. After the adversary chooses its strategy, we instantiate the algebra with a finite cyclic group. The attack succeeds if there is, regardless of the choice of group, a non-negligible probability that the message will be accepted. *Non-negligibility* here is interpreted relative to the logarithm of the cardinality of the group.

It would be important to discover if, under the Computational Diffie-Hellman assumption, there are attacks that a polynomial-time probabilistic Turing machine could execute which, though not uniformly algebraic, has a non-negligible probability of success.

The work that we describe here is entirely handcrafted. It would be highly desirable to introduce techniques for automated symbolic protocol that are flexible enough to incorporate both the specifications (such as the CA ordering assumption) and reasoning in the style of Section 6. We believe that an approach using model-finding in *geometric logic*, a generalization of Horn logic, is promising.

Acknowledgments. We are grateful to Moses Liskov, Cathy Meadows, John Ramsdell, Paul Rowe, Paul Timmel, and Ed Zieglar for vigorous discussions.

References

1. James Ax. The elementary theory of finite fields. *The Annals of Mathematics*, 88(2):pp. 239–271, 1968.
2. Simon Blake-Wilson and Alfred Menezes. Authenticated Diffe-Hellman key agreement protocols. In *Selected Areas in Cryptography*, pages 630–630. Springer, 1999.
3. Bruno Blanchet. An efficient protocol verifier based on Prolog rules. In *14th Computer Security Foundations Workshop*, pages 82–96. IEEE CS Press, June 2001.
4. M. Boreale and M.G. Buscemi. Symbolic analysis of crypto-protocols based on modular exponentiation. *Mathematical Foundations of Computer Science 2003*, pages 269–278, 2003.

5. Emmanuel Bresson, Yassine Lakhnech, Laurent Mazaré, and Bogdan Warinschi. Computational soundness: The case of Diffie-Hellman keys. In Veronique Cortier and Steve Kremer, editors, *Formal Models and Techniques for Analyzing Security Protocols*, Cryptology and Information Security Series. IOS Press, 2011.
6. C.C. Chang and H.J. Keisler. Model Theory, volume 73 of Studies in Logic and the Foundations of Mathematics, 1990.
7. Yannick Chevalier, Ralf Küsters, Michaël Rusinowitch, and Mathieu Turuani. Deciding the security of protocols with Diffie-Hellman exponentiation and products in exponents. *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science*, pages 124–135, 2003.
8. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.
9. Whitfield Diffie, Paul C. van Oorschot, and Michael J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2(2):107–125, 1992.
10. Santiago Escobar, Catherine Meadows, and José Meseguer. State space reduction in the Maude-NRL protocol analyzer. *Computer Security-ESORICS 2008*, pages 548–562, 2008.
11. Santiago Escobar, Catherine Meadows, and José Meseguer. Maude-NPA: Cryptographic protocol analysis modulo equational properties. *Foundations of Security Analysis and Design V*, pages 1–50, 2009.
12. Marcelo Fiore and Martín Abadi. Computing symbolic models for verifying cryptographic protocols. In *Computer Security Foundations Workshop*, June 2001.
13. J. Giesl, P. Schneider-Kamp, and R. Thiemann. Aprove 1.2: Automatic termination proofs in the dependency pair framework. In *Proceedings IJCAR '06*, LNAI 4130, pages 281–286. Springer, 2006.
14. Jean Goubault-Larrecq, Muriel Roger, and Kumar Verma. Abstraction and resolution modulo AC: How to verify Diffie-Hellman-like protocols automatically. *Journal of Logic and Algebraic Programming*, 64(2):219–251, 2005.
15. Joshua D. Guttman. Fair exchange in strand spaces. In M. Boreale and S. Kremer, editors, *SecCo: 7th International Workshop on Security Issues in Concurrency*, EPTCS. Electronic Proceedings in Theoretical Computer Science, Sep 2009.
16. Joshua D. Guttman. Shapes: Surveying crypto protocol runs. In Veronique Cortier and Steve Kremer, editors, *Formal Models and Techniques for Analyzing Security Protocols*, Cryptology and Information Security Series. IOS Press, 2011.
17. Alfred Hitchcock. I confess. Warner Brothers, March 1953. http://www.imdb.com/title/tt0045897/.
18. Burton S. Kaliski. An unknown key-share attack on the MQV key agreement protocol. *ACM Transactions on Information and System Security*, 4(3):275–288, 2001.
19. Deepak Kapur, Paliath Narendran, and Lida Wang. An E-unification algorithm for analyzing protocols that use modular exponentiation. *RewritingTechniques and Applications*, pages 150–150, 2003.
20. H. Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. In *Advances in Cryptology–CRYPTO 2005*, pages 546–566. Springer, 2005.
21. Sebastian Kunz-Jacques and David Pointcheval. About the Security of MTI/C0 and MQV. *Security and Cryptography for Networks*, pages 156–172, 2006.
22. Ralf Küsters and Tomasz Truderung. Using ProVerif to analyze protocols with Diffie-Hellman exponentiation. In *IEEE Computer Security Foundations Symposium*, pages 157–171. IEEE, 2009.

23. L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone. An efficient protocol for authenticated key agreement. *Designs, Codes and Cryptography*, 28(2):119–134, 2003.
24. A. Menezes, University of Waterloo. Dept. of Combinatorics, Optimization, and University of Waterloo. Faculty of Mathematics. *Another look at HMQV*. Citeseer, 2005.
25. Jonathan K. Millen and Vitaly Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *8th ACM Conference on Computer and Communications Security (CCS '01)*, pages 166–175. ACM, 2001.
26. Albert Rubio. A fully syntactic AC-RPO. In Paliath Narendran and Michaël Rusinowitch, editors, *RTA*, volume 1631 of *Lecture Notes in Computer Science*, pages 133–147. Springer, 1999.
27. F. Javier Thayer, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(2/3):191–230, 1999.