Journal of King Saud University – Computer and Information Sciences 31 (2019) 318–326

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

A formal framework for verifying IKA property of protocols

Shyam P. Joy *, Priya Chandran

Department of Computer Science and Engineering, National Institute of Technology Calicut, Kerala 673601, India

ARTICLE INFO

Article history:
Received 19 October 2016
Revised 7 January 2017
Accepted 8 January 2017
Available online 28 January 2017

Keywords:
Group Key Agreement Protocols
Implicit Key Authentication
Strand Spaces
Formal Framework

ABSTRACT

A group key agreement (GKA) protocol generates a secret key (session key) shared among the members of the group, from the contributions made by group members. GKA protocols are expected to satisfy the property of Implicit Key Authentication (IKA) which assures group members that the key generated by the protocol is not accessible to any member outside the group. In this article, we propose a technique to prove the correctness of GKA protocols with respect to IKA. We establish the soundness of our proposal and also illustrate its application.

Normally IKA property of protocols is established by proving that the protocol satisfies authentication of participants and secrecy of the session keys. Most formal models would be able to analyze a GKA protocol with respect to IKA, using the above approach. However analysis of two security properties, namely authentication and secrecy, would increase the chances of errors. We propose a single condition for verifying whether the GKA protocol satisfies IKA.

1. Introduction

Rapid developments in the field of computer networks and communications have led to increase in the number of applications based on the Internet. These applications need to be secure, and hence utilize security protocols. Security protocols define rules for communication, to achieve security objectives like authentication and secrecy. Earlier methods of ascertaining whether a protocol achieves its security objectives was by trial and error. The difficulty in detecting the flaws in security protocols by the above method is exemplified by the fact that it took seventeen years after the Needham–Schroeder public key authentication protocol (Needham and Schroeder, 1978) was proposed, that an attack on the protocol was discovered by Lowe (1995). Formal modeling and analysis helps to overcome the above difficulty. Techniques used for formal modeling and analysis of protocols include BAN Logic (Burrows et al., 1990), Communicating Sequential Processes (Schneider, 1996), Spi Calculus (Abadi and Gordon, 1997), Strand Spaces (Fabrega et al., 1999), Inductive Approach (Paulson, 1997), Multi-set Re-writing approach (Durgin and Mitchell, 1999) and a model proposed by Ramanujam and Suresh (2003). All of the above use symbolic approach toward modeling while Bellare and Rogaway (1993) use a computational approach.

One of the functions performed by security protocols is to distribute a shared secret key among members of a group. Key distribution in a group can be done either through a centralized or a distributed approach. In the centralized approach, a trusted server that is not a member of the group distributes the key to the members of the group. In distributed approach, each group member computes a shared secret key from the contributions made by each member of the group. The latter class of protocols is referred to as key agreement or group key agreement (GKA) protocols (Manulis, 2006). GKA protocols are suitable for peer-to-peer applications (Amir et al., 2002). A number of key agreement protocols have been proposed in the literature (Ateniese et al., 2000; Bresson et al., 2002; Burmester and Desmedt, 1994; Just and Vaudenay, 1996; Kim et al., 2001; Kim et al., 2004; Steer et al., 1990; Mishra et al., 2014). Most of the GKA protocols are based on Diffie–Hellman (DH) Key Exchange scheme (Diffie and Hellman, 1976).

Key agreement protocols must satisfy the security requirement of *Implicit Key Authentication (IKA)* or Key Authentication. *IKA is the property whereby one party is assured that no other party aside from a specifically identified second party (and possibly additional identified trusted parties) may gain access to a particular secret key* (Menezes et al., 1996). Normally, IKA property of protocols are established

* Corresponding author.
E-mail addresses: shyampjoy@gmail.com (S.P. Joy), priya@nitc.ac.in (P. Chandran).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2017.01.002
1319-1578/

by proving that the protocol satisfies authentication of participants and secrecy of the group key. In this article, we propose a formal technique for proving the correctness of a group key agreement protocol with respect to IKA. Strand Spaces framework (Fabrega et al., 1999) is used in modeling and proving the correctness of protocols. The reader, if new to Strand Spaces, would find it useful to go through the references (Fabrega et al., 1999; Herzog, 2003) before reading the rest of the text.

The organization of the rest of the paper is as follows: Section 2 briefly discusses the challenges involved in modeling and analyzing DH protocols and various models proposed to overcome those challenges. In Section 3, we present our contributions which includes tweaking Strand Spaces to our requirements, defining the notion of correctness of protocols with respect to IKA, and establishing its soundness. Section 4 highlights the contributions in our proposal. The proposal is illustrated in Section 5. A comparison of our proposal with existing proposal is presented in Section 6.

2. Related work

This section presents the state of research on the formal analysis of GKA protocols using Strand Spaces. Since this article is concerned with establishing the IKA properties of GKA protocols using Strand Spaces, and most GKA protocols are based on DH, this section presents a survey on the challenges involved in the symbolic analysis of DH protocols and the various approaches adopted in solving them.

The secrecy of the key exchanged in DH protocol relies on Discrete Logarithm problem being hard. The messages exchanged in the protocol consist of terms which are elements of abelian group Z_p^*. We will refer to these terms as DH terms. Modeling DH terms is more challenging as the algebra on terms needs to be considered.

In formal modeling, if a term t_1 can be derived from t_2, then t_1 is defined as a sub-term of t_2, denoted as $t_1 \sqsubset t_2$ (Fabrega et al., 1999). To decide whether a term t remains secret, after execution of a protocol, it is sufficient to examine if the intruder has access to any term whose sub-term is t. However in the case of DH terms, relations like $x \sqsubset \alpha^x$ or $\alpha^x \sqsubset \alpha^{x.y}$ are not valid. Therefore models whose analysis is based on sub-term relation needs to be modified for analysis of DH protocols. The following paragraphs present two modifications to Strand Spaces to overcome this challenge.

Herzog proposed extensions to Strand Spaces (Herzog, 2003) for analyzing security protocols based on DH primitive. He overcame the above problem by defining a term as an *ingredient* of other, if it is used in constructing the latter or if it is a sub term. According to the model, a DH term remains a secret if a security protocol is *silent* and *conservative*. These conditions, which the protocols had to satisfy to assure the security property, were defined in terms of *ingredient* relation. However the model is limited to the secrecy analysis of two-party DH based protocols. The paper does not formalize authentication. Moreover the definition of *silent* is not appropriate to group key agreement protocols like AT-GDH-2 protocols which have been proved to be secure but violate the *silent* condition.

Another approach to overcome the above problem was presented in Pereira (2003) which proposed a model for analyzing Cliques protocols (Ateniese et al., 1998; Ateniese et al., 2000). The primitive root α was modeled as $\{1\}_{K_\alpha}$, where 1 and K_α are keys, but K_α does not have any inverse. $\{h\}_k$ denotes encryption of h with key k. Using the above notation α^r is represented as $\{\{1\}_{K_\alpha}, r\}_{K_\alpha}$. With this notation, the relation $x \sqsubset \alpha^x$ is valid as $h \sqsubset \{h\}_k$ and $h \sqsubset hg$ or $g \sqsubset hg$. However, it contradicts the intuition that sub-term can be derived from a term, as deriving r from $\{\{1\}_{K_\alpha}, r\}_{K_\alpha}$ is prevented as K_α is assumed to have no inverse. The model establishes the secrecy of term by checking the consistency of a collection of linear equations. The model is specific to Cliques protocol suite.

Earlier models for analyzing security protocols assumed that the cryptographic algorithms are perfect, in the sense that it is impossible to get the knowledge about the plain text from the corresponding cipher text, without the knowledge of the key. In such cases, the algebra of terms was assumed to be free, where only syntactic equivalence between terms were considered. However the attacker can exploit the algebraic relationships between the terms leading to attacks. Such attacks would not be captured by free algebra models. A recursive authentication protocol proposed by Bull (Bull and Otway, 1997) was proved correct using Inductive Approach (Paulson et al., 1997) assuming that encryption is perfect. However an attack on the protocol was discovered when the protocol was implemented using XOR for encryption, owing to self cancelation property of XOR (Ryan and Schneider, 1998). Like XOR, modular multiplication also forms an abelian group and attacks have been discovered on protocols based on DH protocols, which exploit the commutative property of modular multiplication. As a consequence, protocol verification accommodating algebraic properties of operators were studied. Some attempts at solving the above issues are presented in the following paragraphs.

Rob and Steve (Delicata and Schneider, 2003) has used CSP (Hoare, 2004) and rank functions to model DH protocols. The method uses rank functions to map messages to a set $\{0, 1\}$, so that the messages that an intruder should not get, is assigned a rank of 0 and others are assigned 1. The model overcame the issues due to commutativity by proposing a normal form for DH terms, so that messages that are structurally different, but identical in value, may be assigned the same ranks.

A verification model for DH protocols based on spi-calculus (Abadi and Gordon, 1997) was presented in Boreale and Buscemi (2003). To model the commutativity and associativity of abelian groups, the model uses non-determinism. For example, the operation $mult(\alpha_1 \times \ldots \alpha_k, \alpha_{k+1} \times \ldots \alpha_n) \rightarrow \alpha_{i_1} \times \ldots \times \alpha_{i_n}$, where $i_1 \ldots i_n$ denotes any permutation of $1 \ldots n$ and $\alpha_1 \times \ldots \times \alpha_n$ denotes $\alpha_1 \times (\alpha_2 \times \ldots \alpha_n)$.

The model (Goubault-Larrecq et al., 2005) uses resolution theorem proving with ordering and selection to verify DH based protocols. The protocol and the security conditions are modeled using a unary function e, with α^r being represented as $e(r)$. $\alpha^{r_1 r_2}$ is represented as $e(r_1 \oplus r_2)$, such that $e(r_1 \oplus r_2) = e(r_2 \oplus r_1)$, to account for commutativity.

Jonathan Millen and Vitaly Shmaitikov introduced constraint solving for analysis of security protocols involving ground terms and for constructed keys in Millen and Shamatikov (2005) and with modular exponentiation with arbitrary base. The model considers its terms to be in normal form if they cannot be reduced with respect to a set of reduction rules even after rearranging using associativity and commutativity. Delaune et al. (2008) generalized the approach for monoidal equational theories.

It was observed that, while modeling messages communicated between the principals of protocols, it is necessary to introduce variables, as a receiving honest principal does not know apriori the term he will receive. Moreover it is possible that, a message received by a participant has been sent by an intruder. Therefore, if the message exchanges are as per the protocol and if there is a consistent unification of terms such that the attacker is able to derive the secret from his knowledge, then the protocol is insecure. Therefore unification problem under equational theory of abelian groups assumes importance, while modeling DH protocols. However unification problem of modular exponents under equational

theory is not always decidable. A summary of decidability results on unification problem under various equational theories can be found in Kapur et al. (2003).

Maude-NRL Protocol Analyzer (Santiago et al., 2007) is a tool for analyzing protocols that satisfy algebraic properties. It performs a backward narrowing reachability analysis from an attack state to the initial state. It finds an attack if one exists and whenever it terminates without attacks, it gives a proof of security. However, there is no guarantee of termination.

Analysis of DH based exponentiation was carried out using Pro-Verif by Ralf Kusters and Truderung (2009). The protocol and the intruder were modeled using Horn Theory modulo the algebraic properties of DH. The model imposed a restriction on exponential terms of the form s^t that s should not be of the form a^{-1} and t should be a ground term. Horn theory under such a restriction was called as Exponent Ground Horn Theory. Once the protocol model is reduced to Exponential Ground Horn Theory it is solved using Pro-Verif.

Schmidt et al. (2012) proposed a symbolic model for analyzing DH protocols. The protocol and the adversary are modeled using multi set term re-writing. The model considers DH exponents to form an abelian group. However the model does not support addition of exponents.

The decidability of secrecy for protocols based on DH was proved to be NP Complete (Chevalier et al., 2008). They defined a set of re-write rules which were confluent modulo associativity and commutativity with respect to multiplication.

Dougherty and Guttman (2013) has proposed a re-writing theory for analyzing DH based protocols. The paper recognizes that the exponents in DH scenario forms a finite field, where as previous symbolic approaches considered the algebraic properties of abelian group. They also established that secrecy was decidable for light weight Diffie–Hellman protocols (Dougherty and Guttman, 2014). The algebraic properties of fields is being used by protocols like MQV (Blake-Wilson and Menezes, 1999; Krawczyk, 2005).

Models presented above would be able to verify IKA by separately verifying that the protocols satisfy authentication and secrecy. This approach has been used in Herzog (2003) and Pereira (2003). However, two separate verifications increase the chances of errors. In this paper, we propose a single condition for proving correctness of GKA protocols with respect to IKA. It would be sufficient to verify the proposed condition instead of verifying authentication and secrecy separately. Although many models have used Strand Spaces representation for protocols, the techniques for analysis were different from that of Strand Spaces. We represent the protocol using Strand Spaces with some modifications and analyze the property using the conventional Strand Spaces approach, i.e., partial order theory. Our proposal is a unified approach to solve the same problem, but using a single verification step. The proposal is presented in the following section:

3. Framework for proving IKA

This section presents our proposal for proving IKA property of GKA protocols. We briefly present Strand Spaces (Fabrega et al., 1999; Herzog, 2003) and the proposed extensions. After adapting Strand Spaces to our requirements we present the notion of correctness, establish the soundness of the proposal and illustrate the proposal on sample GKA protocols.

3.1. Strand Spaces with the proposed Extensions

Since our proposal is an extension of Strand Spaces, we have interleaved our proposal within the necessary Strand Spaces literature. We have made modifications to the representation of protocol using Strand Spaces. An extension has also been made to the penetrator model. The major modification has been in the term algebra.

Strand Spaces (Fabrega et al., 1999) provide a framework to model protocols and prove the correctness of protocols. Failure to prove the correctness gives insight into possible flaws in protocols. The model uses a graphical representation to show the causal precedence relationships between terms.

Let the set T represent all possible terms that can be sent and received in a protocol. The trace of all activities performed by an instance of a principal (or penetrator) is represented as a *strand* (Fabrega et al., 1999). If a protocol is considered as a collection of roles played by participants, the strands can be modified as *role strands* i.e. strands with parameters. An instance of the protocol results by binding the parameters. Similar representation have been used by Millen and Shamatikov (2005) and Song (1999). We use the following notation to differentiate between a parameter and its instance: the instance of a parameter x is represented as $x_{<numeral>}$.

For verification of IKA property of protocols, we propose that the trace parameters be bound only if the messages are signed by their respective senders, as signature identifies sender and assures integrity of the message.

Each strand consists of a sequence of nodes connected by the symbol '$\Rightarrow$'. $\langle s, i\rangle$ denotes ith node on strand s. Each node represents *action* taken by a principal. The action can be that of 'send' or 'receive'. If a principal sends a term t from node n_1, and the same is received by node n_2, then $term(n_1) = +t$ and $term(n_2) = -t$ and node n_1 is connected to node n_2 by the symbol '$\rightarrow$' (Fabrega et al., 1999).

The set of nodes along with the set of edges, $\rightarrow \cup \Rightarrow$, form a directed graph. A finite, acyclic, subgraph of the above directed graph is called a *bundle* if: for all the receiver nodes in the bundle, the corresponding sender nodes must also be in the bundle and for all nodes present in the bundle, their immediate causal predecessors must also be present in the bundle (Fabrega et al., 1999).

If $C \subset \rightarrow \cup \Rightarrow$, then the symbol '$\preceq_C$' means the reflexive, transitive closure of C. Suppose B is a bundle. Then $\preceq_B$ is a partial order relation and every non-empty subset of nodes in B has $\preceq_B$ minimal members. It has been proved in Fabrega et al. (1999) that if $C \subseteq B$ is a set of nodes such that $\forall m, m', unsigned(m) = unsigned(m') \Rightarrow (m \in C \text{ iff } m' \in C)$ and if n is a $\preceq_B$ minimal member of C, then the sign of n is positive.

In the following section, we present the penetrator model used in this article.

3.1.1. Penetrator model

In Strand Spaces, the capability of penetrator is represented by a set of penetrator strands, as shown below:

- **M** – Text message: $\langle +t\rangle$ where $t \in T$
- **F** – Flushing a message $\langle -g\rangle$
- **R** – Fresh Nonces: $\langle +r\rangle$ where $r \in R_P$, set of nonces known to penetrator.
- **C** – Concatenation: $\langle -g, -h, +gh\rangle$
- **S** – Separation into components: $\langle -gh, +g, +h\rangle$
- **K** – Key: $\langle +K\rangle$ where $K \in K_P$, set of keys known to the penetrator.
- **E** – Encryption: $\langle -K, -h, +\{h\}_K\rangle$
- **D** – Decryption: $\langle -K^{-1}, -\{h\}_K, +h\rangle$
- $\boldsymbol{\sigma}$ – Signing: $\langle -K, -h, +[h]_K\rangle$ where $K \in K_{Sig}$. K_{Sig} is the set of keys which are used by participants to sign messages.
- **X** – Extraction of plaintext from signatures: $\langle -[h]_K, +h\rangle$
- **H** – Hashing: $\langle -g, +\text{hash}(g)\rangle$

- **FF** – Fresh Diffie–Hellman value: $\langle +\alpha^{p} \rangle$
- **Exp** – $\langle -\alpha^{s_1 \ldots s_{i-1} s_{i+1} \ldots s_n}, -s_i, +\alpha^{s} \rangle$, where $s_1 \ldots s_n$ represents permutation of factors of s and s_i is known to the penetrator.

The strands **M**, **R**, **K** and **FF** represent penetrator's ability to guess a text, nonce, key and Diffie–Hellman term respectively. Strand **C** allows the penetrator to concatenate two received terms g and h. Similarly, strand **S** models the penetrator's ability to separate concatenated term gh into components g and h. **E** represents the ability of penetrator to encrypt a known term h with a known key K and **D** represents his ability to decrypt an encrypted term with a key known to the penetrator. The strand σ and **X** represents penetrator's ability to sign a message with a key known to the penetrator and extract plain text from signature respectively. Penetrator's ability to Hash is represented by **H** strand. **F** represents the penetrator's ability to delete a message from the protocol.

In addition, we introduce the **Exp** strand which models the ability of the penetrator to perform Diffie–Hellman exponentiation. **Exp** strand models the ability of the penetrator to exponentiate a received term $\alpha^{s_1 \ldots s_{i-1} s_{i+1} \ldots s_n}$ using an integer s_i. Since $s_1 \ldots s_n$ are factors of s, in any permutation, exponentiation results in α^{s}, i.e. the Diffie–Hellman terms are normalized with respect to associativity and commutativity. This would mean that the terms $\alpha^{(a.b)} = \alpha^{(b.a)}$ and $\alpha^{(a.b).c} = \alpha^{a.(b.c)}$. The idea is similar to the one used in Boreale and Buscemi (2003).

To illustrate the penetrator model, consider an arbitrary protocol in which participant A sends the message t_1, t_2, t_3 to participant B. The message is a concatenation of terms t_1, t_2 and t_3. The Strand Spaces with strands of A and B, along with attacker realizing the action of replacing t_2 in the message by t_2' is modeled as in Fig. 1. Strand **S** separates t_1, t_2, t_3 to three components t_1, t_2 and t_3. Node **F** deletes the term t_2, and strand **M** introduces a term t_2'. Strand **C** represents the concatenation of t_1, t_2' and t_3.

Fig. 2 represents the session of the penetrator with the initiator, A, during Man-in-Middle attack on two-party Diffie–Hellman Key Exchange, using Strand Spaces. The message α^{a} sent from A to B is captured by the penetrator. The penetrator generates a random integer c using **R** and computes α^{c} by exponentiating α, using the **Exp** strand and sends it back to A. The penetrator computes α^{ac} by exponentiating α^{a} using the **Exp** Strand.

The penetrator strands that we are considering in this model are similar to the ones in Herzog (2003). Herzog (2003) had an additional f strand which gave the penetrator the ability to compute Diffie–Hellman terms from other terms using probabilistic polynomial time algorithm. Herzog (2003) used f strand to establish a relationship between computational and formal domains. Since we assume that Diffie–Hellman problem is hard, **FF** and **Exp** strands are sufficient to model the capabilities of the penetrator with respect to Diffie–Hellman primitive.

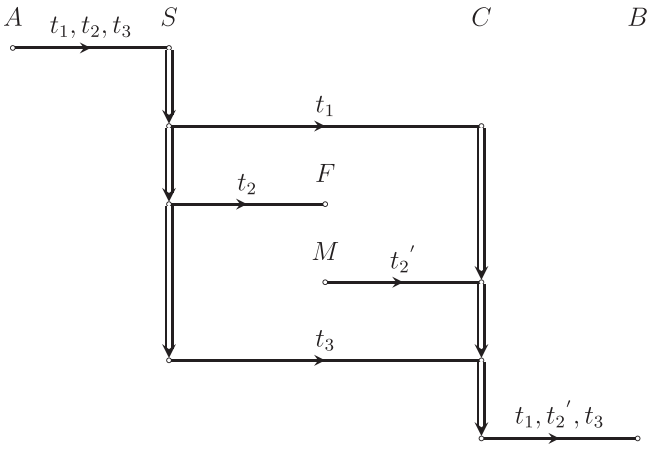

Fig. 1. Strand Space representation of replacement of a term in a message by attacker.

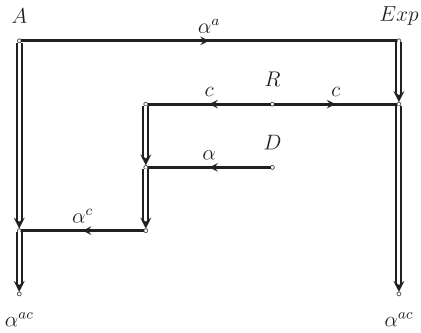

Fig. 2. Strand Space representation of one session of Man-In-Middle attack on Diffie–Hellman Key Exchange.

The major extensions proposed to the Strand Spaces are in the Term Algebra of the model, which is discussed below:

3.1.2. Term algebra

Let set T consists of all messages transmitted during the run of the protocol (Herzog, 2003). The set T can be classified into four mutually disjoint subsets namely:

- $A \subseteq T$, set containing predictable text messages
- $K \subseteq T$, set containing Keys
- $R \subseteq T$, set containing nonces
- $D \subseteq T$, set containing Diffie–Hellman terms.

The set A would consist of names of principals and any other text values as needed to model the protocol, the set K consists of keys for encryption, decryption and signature, R represents the set of nonces and D consists of elements of the form α^{x} where α is the generator of a finite cyclic group G and x is a natural number less than the order of the group. New terms can be built from the existing terms using the following functions (Herzog, 2003):

- *encrypt* : $K \times T \rightarrow T$. Function *encrypt* performs encryption on a plain text element of T using a key from K, resulting in a cipher text which is also an element of T. Strand Spaces assumes that two distinct plain texts or two distinct keys cannot produce same cipher texts on encryption.
- *concat* : $T \times T \rightarrow T$. Function *concat* performs the concatenation of terms from T and the resulting term is also an element of set T. It is assumed that if two strings resulting from the concatenation of two pairs of sub-strings are equal, then the corresponding sub-strings must also be equal.
- *exponent* : $D \times Z^{+} \rightarrow D$, where Z^{+} is set of positive integers less than the order of the group. If $\alpha^{x} \in D$ and $t \in Z^{+}$ then $exponent(\alpha^{x}, t) \rightarrow \alpha^{xt} \in D$. This is similar to *DH* operation defined in Herzog (2003) using more explicit representation.
- *hashing* : $T \rightarrow K$. *hashing* is a one way function which converts a term of T to an atomic key. We assume that the hash operation is injective. The assumption is valid as hash functions are expected to provide collision resistance which is assured by injectivity.
- *signature* : $T \times K \rightarrow T$. *signature* is a function similar to encryption. It signs the term of T by a key known only to the participant who signs.

It is assumed that a key, text or random term cannot be created from concatenation or encryption. These assumptions are collectively referred to as free algebra assumptions (Fabrega et al., 1999). Encryption of a term M by an encryption key K is denoted by $\{M\}_K$. Signature of a term M by a signature key K is denoted by the notation $[M]_K$.

Some of the definitions of Strand Spaces (Fabrega et al., 1999; Herzog, 2003) that would be used to explain our proposal is presented next.

The Terms that can be derived from the existing terms are called sub-terms.

Sub-term relation is the smallest relation defined inductively by the following rules (Fabrega et al., 1999):

- $a \sqsubset a$.
- $a \sqsubset \{b\}_k$ if $a \sqsubset b$.
- $a \sqsubset bc$ if $a \sqsubset b$ or $a \sqsubset c$.

A set of terms, such that h is a sub term of any term in the set, is called the ideal of h. *Ideals* have been defined in Herzog (2003) as follows: If $K' \subseteq K$, a K'-Ideal of T is a subset I of T such that for all $h \in I$, $g \in T$ and $k \in K'$, $hg \in I$, $gh \in I$, $\{h\}_k \in I$ and $[h]_k \in I$. The smallest K'-Ideal that contains h is denoted as $I_{K'}[h]$. All terms in $I_{K'}[h]$ has h as a sub-term.

The *entry point* and *origination* of a term is defined in Fabrega et al. (1999) as follows: A node n is defined as an *entry point* for a set of unsigned terms I iff $term(n) = +t$ for some $t \in I$ and whenever n' precedes n on a strand, $term(n') \notin I$ (Fabrega et al., 1999).

A term t *originates* at a node n iff the node n is an entry point to the set $\{t' : t \sqsubset t'\}$ (Fabrega et al., 1999).

Herzog (2003) defined *ingredient* relation on terms to capture the notion of the terms that can either be derived from other terms or those that are used in building other terms.

X is an *ingredient* of Y, written $X \prec Y$, if:

- $X = Y$, or
- if $Y = hash(Y')$, then $X \prec Y'$
- if $Y = g^{ab}$, then $X \prec g^a$ or $X \prec g^b$
- if $Y = \{Y'\}_K$, then $X \prec Y'$ *or* $X \prec K$
- if $Y = [Y']_K$, then $X \prec Y'$ *or* $X \prec K$

Similar to *origination* for sub-term relation, *arise* has been defined (Herzog, 2003) for ingredient relation as follows:

Arise: A term t *arises* on a node n iff n is an entry point to the set $I = \{t' : t \prec t'\}$ (Herzog, 2003).

In the following paragraphs, the proposed extensions to Strand Spaces are presented. We modify the ingredient relation so as to develop the framework for proving IKA:

Modified Ingredient Relation: The **modified ingredient relation**, denoted by <, is defined as the smallest non-empty relation defined inductively by the following properties:

1. $m < h \iff m < hash(h)$
2. $m < h \vee m < k \iff m < \{h\}_k$
3. $m < h \vee m < k \iff m < [h]_k$
4. $m < g \vee m < h \iff m < gh$
5. $m < g^{s_1} \vee \ldots \vee m < g^{s_n} \iff m < g^s$ where $s_1, \ldots, s_n$ are factors of s.

The modified ingredient relation is different from ingredient relation (Herzog, 2003). In addition to the implication defined in Ingredient relation, Modified Ingredient relations include their converse too. This is valid because, if a pair of terms related by the ingredient relation imply the presence of another pair of terms related by the same relation, then the presence of the latter pair would imply the former, as the set is built inductively. Moreover, Relation 1 holds as the hash function is assumed to be injective and hashing results in an atomic value. Relations $2, 3, 4$ hold due to the free algebra assumptions about encryption and concatenation functions. Relation 5 holds as the elements of the form g^s belong to cyclic group. These relations are essential for the correctness of our proofs.

It may be noted that equality is not included in the modified ingredient relation. This is because including equality can lead to ambiguous relations such as the one below:

- $hash(m) < hash(m) \Rightarrow hash(m) < m$

To account for equality of terms, we propose a **constituent** relation between terms and denote it by $\succ$. The relation is defined as follows:

Constituent: The constituent relation holds if and only if modified ingredient relation holds or the case that the two terms are identical holds, but not both. Formally,

- $m \succ n \iff (m = n) \oplus (m < n)$

Similar to the definition of arise (Herzog, 2003) for ingredient relation, we define *rise* for constituent relation.

Rise: A term t *rise* on a node n iff n is an entry point to the set $I = \{t' : t \succ t'\}$.

The following lemma relates the partial order structure of the bundle and the constituent relation between the terms.

Lemma 1. *Suppose B is a bundle, $t \in T$ and $n \in B$ is a $\preceq_B$ minimal element of $S = \{m \in C : t \succ term(m)\}$. Then t rises on node n.*

Proof. Since n is a member of S, $t \succ term(n)$. If a node n' precedes n on the same strand, then $n' \in B$, as bundle is backward closed on '$\Rightarrow$'. By minimality property of n, $t \not\succ term(n')$. The proof is similar to the proof relating origination and minimal node in Fabrega et al. (1999).

It has already been proved in Fabrega et al. (1999) that minimal node of a subset of the bundle is positive. This notion along with the above lemma prove that the node on which a term t *rises* would be positive.

We next propose a formal condition which when satisfied assures that the protocol satisfies IKA property.

3.2. Notion of correctness of IKA:

In this section we propose a single correctness condition which can be used for verification of IKA. Formally, we establish the following implication:

- Correctness Condition $\Rightarrow$ IKA

The proposal is based on the following assumptions: GKA protocols using DH primitive are based on three related assumptions namely discrete logarithm assumption, computational Diffie–Hellman assumption and Decisional Diffie Hellman assumption. Consider a cyclic group with generator α and order q. Discrete logarithm problem assumes that, it is computationally infeasible to compute a knowing α^a, where a is a random integer less than q. Similarly, computational Diffie–Hellman assumption states that, it is computationally infeasible to compute α^{ab}, knowing α, α^a and

α^b, where a, b are random integers less than q. Decisional Diffie–Hellman assumes indistinguishabilty between α, α^a, α^b, α^{ab}. Diffie–Hellman key Exchange Scheme and ElGamal Encryption algorithm are both based on the above assumptions. In addition to the above mentioned assumptions, our method is applicable only to those protocols that satisfy the following condition:

3.2.1. Secrecy condition:

Secrecy condition ensures that the regular participants do not transmit the session key in a form from which penetrator can deduce the key. The condition is satisfied by all GKA protocols.

Formally we state the condition as follows: No regular node is an entry point to $I_{K_P}[k_g]$, where $I_{K_P}[k_g]$ is the smallest K_P-ideal containing k_g, K_P is the set of keys known to the penetrator and k_g is the computed group key.

A node n is defined as an entry point to a set of unsigned terms I iff $term(n) = +t$ for some $t \in I$ and whenever n' precedes n on a strand, $term(n') \notin I$ (Fabrega et al., 1999). We define a predicate $EP(n, I)$, read as node n is an entry point to set I, to represent entry point as follows:

$$EP(n, I) \stackrel{def}{=} (\exists t : t \in I, term(n) = +t) \wedge \left(\forall n' : n, n' \in C : n' \stackrel{*}{\Rightarrow} n : term(n') \notin I\right)$$

where C is a bundle, and $n' \stackrel{*}{\Rightarrow} n$ denotes n' preceding n with zero or more nodes in between.

Based on the above predicate, the secrecy condition can be formally stated as follows: $\forall n \in N \setminus N_P(\neg EP(n, I_{K_P}[k_g]))$. A protocol satisfies secrecy condition whenever the above condition is true.

Consider the case when the *Secrecy Condition* is violated i.e. a regular node n being an entry point to set $I_{K_P}[k_g]$. Then $term(n) \in I_{K_P}[k_g]$. All elements of set $I_{K_P}[k_g]$ are such that the penetrator can deduce the group key from them. This would be violating the secrecy of the group key trivially.

Under the assumption stated above, a GKA protocol satisfies IKA property if it satisfies the following condition:

3.2.2. Correctness condition

A protocol satisfies IKA property, if no **constituent** *of the computed key* **rises** *on a penetrator strand.*

To formulate the correctness condition, we define predicates. We stated that a term t is said to *rise* on node n iff n is an entry point to the set $J = \{t' : t \succ t'\}$, a set of all terms for whom t is a constituent. We define a predicate $RISE(t, n)$, read as t rises on the node n, as follows:

$$RISE(t, n) \stackrel{def}{=} EP(n, J)$$

Let E be the set containing all constituents of the group key $k_g, E = \{t : t \succ k_g\}$. If N_P represents the set of penetrator nodes in a bundle, the correctness condition stated above can be formally stated as follows:

$$[\forall a : a \in E, \forall n : n \in N_P, \neg RISE(a, n)] \rightarrow IKA \quad (1)$$

In the following section we establish the soundness of our proposal.

3.3. Soundness of proposal

In mathematical logic a proof system is sound if a formula can be deduced from a set of axioms, then the formula is a valid consequence of the set of axioms. Verification of security protocols requires a framework (proof system) for expressing the protocols (axioms) and a correctness condition (formula) corresponding to the property being verified. The soundness of the framework is established by proving that whenever the condition is proved then the property holds, for all protocols.

We have proposed a framework for representing protocols and a correctness condition to be satisfied to guarantee that the protocol satisfies IKA. The proposed framework is sound if whenever the condition is proved then IKA has to be satisfied for any protocol.

IKA property of protocols has been established by proving that the protocols satisfy authentication and secrecy Pereira (2003), i.e. Authentication $\wedge$ Secrecy $\rightarrow$ IKA. To prove that the proposed Correctness Condition $\rightarrow$ IKA, it is enough to prove that Correctness Condition $\rightarrow$ Authentication $\wedge$ Secrecy or formally

$$[\forall a : a \in E, \forall n : n \in N_P, \neg RISE(a, n)] \rightarrow \text{Authentication} \wedge \text{Secrecy}$$

To prove the above implication, assume that authentication fails. This allows penetrator to introduce terms of his choice into the protocol. Then at least one constituent of the key would rise in penetrator strand and hence the correctness condition is violated.

On the other hand assume that the secrecy is violated, i.e. computed group key is not a secret, then there are two possibilities which are analyzed below:

1. The penetrator originates the key. As every key is a constituent of itself, the condition is violated trivially.
2. Key originates on the regular node and penetrator can deduce the key. In such a case, the corresponding term would be an element of $I_{K_P}[k_g]$ and a regular node would be an entry point for $I_{K_P}[k_g]$, which is prevented by Secrecy Condition defined in Section 3.2.1.

This establishes the soundness of our proposal. Before illustrating the proposal, the contributions of this paper are summarized below:

4. Contributions

1. A single correctness condition to verify the IKA property of GKA has been proposed. This reduces the chances of errors that might occur if IKA were to be verified by verifying authentication and secrecy separately. To define the correctness condition, the following modifications were made to Strand Spaces:
   (a) Normally, terms received in a protocol are represented by variables. It is proposed that whenever a message is signed by the sender, the variable has to be bound to the ground terms.
   (b) The ingredient relation was defined in Herzog (2003) as a conditional. It was modified to a bi-conditional so that the proof holds.
   (c) To avoid the ambiguity arising from the presence of equality in ingredient relation, a new relation namely constituent was proposed, which was used to define the correctness condition in Section 3.2.2.
2. The soundness of proposal has been established.
3. The proposal has been illustrated on two sample protocols.

In the following section we illustrate our proposal on Simplified TLS and SAT-GDH protocol.

5. Illustration of the proposal

5.1. Proving IKA property of Simplified TLS protocol

A simplified form of TLS protocol was presented in Herzog (2003). The protocol is similar to conventional DH key exchange

combined with entity authentication. Standard representation of the protocol, quoted from Herzog (2003), is as follows:

1. $A \rightarrow B : A$
2. $B \rightarrow A : B[\alpha^b]_{K_B}$
3. $A \rightarrow B : [\alpha^a]_{K_A}\{T_1AB\}_{K'}$
4. $B \rightarrow A : \{T_2AB\}_{K'}$

where A, B are unique identifiers for Client and Server respectively, T_1, T_2 are tags to distinguish message 3 from 4, K' is a symmetric key created by hashing α^{ab}, where α is the generator of a cyclic group G of order q and $a, b < q$. It has been proved that the symmetric key K' remains secret and that mutual authentication is assured by this protocol (Herzog, 2003). Strand Space representation of the protocol is shown in Fig. 3.

To prove that the protocol satisfies IKA property, consider the session key α^{ab}. Since the messages exchanged are signed, the instance of session key can be considered as $\alpha^{a_0b_0}$. To prove that the protocol satisfies IKA, we must show that no constituent of $\alpha^{a_0b_0}$ rises on penetrator strand.

Proposition 1. *Assume a_0 and b_0 are not known to the penetrator. α^{a_0} and α^{b_0} are the constituents of the key $\alpha^{a_0b_0}$. Neither α^{a_0} nor α^{b_0} rise on penetrator node.*

Proof. Consider a bundle B consisting of nodes $\langle s_1, 1\rangle, \langle s_2, 1\rangle, \langle s_2, 2\rangle, \langle s_1, 2\rangle, \quad \langle s_1, 3\rangle, \langle s_2, 3\rangle, \langle s_2, 4\rangle, \langle s_1, 4\rangle$. Let $F = \{n \in B : \alpha^{a_0} \succ term(n)\}$. F is non-empty as we have nodes like $\langle s_1, 3\rangle$ and so minimal node exists and is positive. We examine each of the penetrator strands for possible occurrence of the minimal node with α^{a_0} as its constituent.

- **M,R,K,FF** – These nodes correspond to guessing a text term, a key and a DH term and hence will not be considered.
- **C** – $\langle -g, -h, +gh\rangle$ – $\alpha^{a_0} \neq gh$ as concatenation cannot create a DH term.Therefore, if $\alpha^{a_0} \succ gh$ then $\alpha^{a_0} \succ g$ or $\alpha^{a_0} \succ h$.
- **S** – $\langle -gh, +g, +h\rangle$ – If $\alpha^{a_0} \succ g$ or $\alpha^{a_0} \succ h$ then $\alpha^{a_0} \succ gh$.
- **E** – $\langle -K, -h, +\{h\}_K\rangle$ – $\alpha^{a_0} \neq \{h\}_k$ as encryption cannot create a DH term.Therefore, if $\alpha^{a_0} \succ \{h\}_k$ then $\alpha^{a_0} \succ h$ or $\alpha^{a_0} \succ K$.
- **D** – $\langle -K^{-1}, -\{h\}_K, +h\rangle$ – If $\alpha^{a_0} \succ h$ then $\alpha^{a_0} \succ \{h\}_k$.

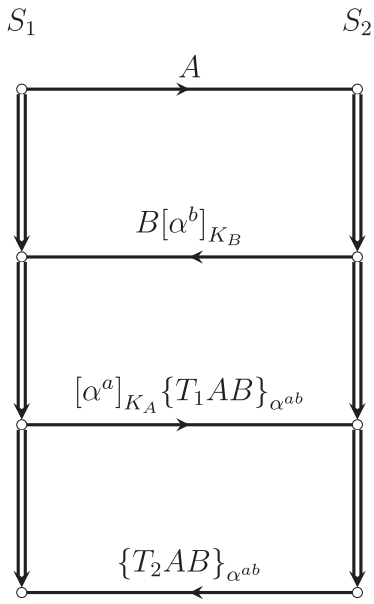

Fig. 3. Strand Space representation of two-party key agreement protocol.

- σ – $\langle -K, -h, +[h]_K\rangle$ – $\alpha^{a_0} \neq [h]_K$ as signing cannot create a DH term. Therefore, if $\alpha^{a_0} \succ [h]_K$ then $\alpha^{a_0} \succ h$ or $\alpha^{a_0} \succ K$.
- **X** – $\langle -[h]_K, +h\rangle$ – If $\alpha^{a_0} \succ h$ then $\alpha^{a_0} \succ [h]_K$.
- **H** – $\langle -h, +\text{hash}(h)\rangle$ – $\alpha^{a_0} \neq hash(h)$ as hashing cannot create a DH term.Therefore, if $\alpha^{a_0} \succ hash(h)$ then $\alpha^{a_0} \succ h$.
- **Exp**– $\langle -\alpha^{s_1 \ldots s_{i-1} s_{i+1} \ldots s_n}, -s_i, +\alpha^s\rangle$ – If $\alpha^{a_0} \succ \alpha^s$ then $\alpha^{a_0} \succ \alpha^{s_1 \ldots s_{i-1} s_{i+1} \ldots s_n}$.

It can be observed that, in none of the above cases α^{a_0} is a constituent of positive node. Hence, there is no penetrator strand where α^{a_0} can arise. The proof for α^{b_0} is symmetric and so the protocol assures IKA. Thus Simplified TLS protocol satisfies the correctness condition and hence IKA is assured. □

Simplified TLS protocol was a two-party protocol. In the following section we analyze a GKA protocol.

5.2. Proving IKA property of SAT-GDH protocol

Authenticated Group Diffie–Hellman (AGDH-2) protocol is a GKA protocol proposed by Ateniese et al. (2000). The protocol failed to provide IKA and a new protocol, AT-GDH was proposed and proved to satisfy IKA property (Pereira, 2003). Even a simpler version of AT-GDH protocol, referred to as Simplified AT-GDH protocol (SAT-GDH) hereafter, would satisfy the IKA property (Pereira, 2003). SAT-GDH protocol is presented below:

Let $\mathscr{U} = \{U_1 \ldots U_n\}$ be a set of n users wishing to share a key K_n. The group of users agree on a cyclic group G and generator α. G is a cyclic subgroup of Z_p^* of order q, such that $q|(p-1)$, where p, q are prime. Each group member U_i selects a new secret random value $r_i \in Z_q^*$ during each session of the protocol.

Each U_i selects an r_i and sends a message which is a sequence of $(i+1)$ elements of group G. U_1 initiates the protocol by sending α, α^{r_1} to U_2 signed by U_1's private key. Each receiver U_j exponentiates all the received terms by r_j and inserts the last term of the received message as the j^{th} term in message, signs the message by his private key, K_{R_j} and sends to U_{j+1}. U_n exponentiates all the received terms with r_n and uses the last term as its key. The remaining $(n-1)$ terms signed by U_n is broadcast to all users. A graphical representation of SAT-GDH protocol is shown in Fig. 4. The protocol is quoted (Pereira, 2003) below using standard notation:

- Round $i (1 \leqslant i < n)$

$$-U_i \rightarrow U_{i+1} : \left\{\alpha^{\frac{r_1 \ldots r_i}{r_j}} | j \in [1, i], \alpha^{r_1 \ldots r_i}\right\}_{K_{R_i}}$$

- Round n

$$-U_n \rightarrow \text{ All } U_i : \left\{\alpha^{\frac{r_1 \ldots r_n}{r_i}} | i \in [1, n]\right\}_{K_{R_n}}.$$

- Upon receipt of the above, every U_i computes: $K_n = \alpha^{(\frac{r_1 \ldots r_n}{r_i}).r_i} = \alpha^{(r_1 \ldots r_n)}$

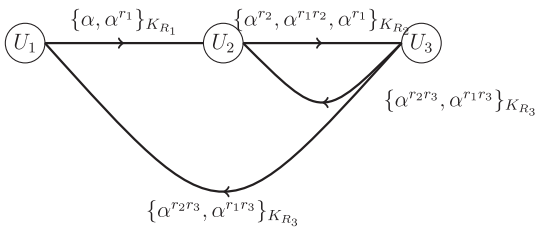

Fig. 4. Representation of SAT-GDH protocol for three participants.

The fact that SAT-GDH protocol satisfies IKA can be proved by adapting the proof for AT-GDH protocol (Pereira, 2003). The strategy is to prove that protocol satisfies authentication and that the computed group key remains secret. Authentication is proved using authentication tests proposed in Guttman and Fabrega (2000) and secrecy is proved as in Fabrega et al. (1999). Strand Space representation of the protocol is shown in Fig. 5.

As in the case of simplified TLS protocol, for SAT-GDH protocol to satisfy IKA, it has to be proved that no constituent of $\alpha^{r_{1_0} r_{2_0} r_{3_0}}$ should arise on penetrator node.

Proposition 2. *Assume r_{1_0}, r_{2_0} and r_{3_0} are not known to the penetrator. $\alpha^{r_{1_0}}, \alpha^{r_{2_0}}, \alpha^{r_{3_0}}, \alpha^{r_{1_0} r_{2_0}}, \alpha^{r_{1_0} r_{3_0}}$ and $\alpha^{r_{2_0} r_{3_0}}$ are constituents of the key $\alpha^{r_{1_0} r_{2_0} r_{3_0}}$. The constituents of the key do not rise on penetrator node.*

Proof. Let B represent a SAT-GDH bundle consisting of $\langle s_1, 1\rangle, \langle s_2, 1\rangle, \langle s_2, 2\rangle, \langle s_3, 1\rangle, \langle s_3, 2\rangle, \langle s_2, 3\rangle$. Let $F = \{n \in B : \alpha^{r_{1_0}} \succ term(n)\}$. F is non-empty as we have nodes like $\langle s_1, 1\rangle$ and so minimal nodes exist and are positive. We examine each of the penetrator strands for possible occurrence of the minimal node with $\alpha^{r_{1_0}}$ as its constituent.

- **M,R,K,FF** – These nodes correspond to guessing a text term, a key and a DH term and hence will not be considered.
- **C** – $\langle -g, -h, +gh\rangle$ – $\alpha^{r_{1_0}} \neq gh$ as concatenation cannot create a DH term. Therefore, if $\alpha^{r_{1_0}} \succ gh$ then $\alpha^{r_{1_0}} \succ g$ or $\alpha^{r_{1_0}} \succ h$.
- **S** – $\langle -gh, +g, +h\rangle$ – If $\alpha^{r_{1_0}} \succ g$ or $\alpha^{r_{1_0}} \succ h$ then $\alpha^{r_{1_0}} \succ gh$.
- **E** – $\langle -K, -h, +\{h\}_K\rangle$ – $\alpha^{r_{1_0}} \neq \{h\}_K$ as encryption cannot create a DH term. Therefore, if $\alpha^{r_{1_0}} \succ \{h\}_K$ then $\alpha^{r_{1_0}} \succ h$ or $\alpha^{r_{1_0}} \succ K$.
- **D** – $\langle -K^{-1}, -\{h\}_K, +h\rangle$ – If $\alpha^{r_{1_0}} \succ h$ then $\alpha^{r_{1_0}} \succ \{h\}_K$.
- σ – $\langle -K, -h, +[h]_K\rangle$ – $\alpha^{r_{1_0}} \neq [h]_K$ as signing cannot create a DH term. Therefore, if $\alpha^{r_{1_0}} \succ [h]_K$ then $\alpha^{r_{1_0}} \succ h$ or $\alpha^{r_{1_0}} \succ K$.
- **X** – If $\alpha^{r_{1_0}} \succ h$ then $\alpha^{r_{1_0}} \succ [h]_K$.
- **H** – $\langle -h, +\text{hash}(h)\rangle$ – $\alpha^{r_{1_0}} \neq hash(h)$as hashing cannot create a DH term. Therefore, if $\alpha^{r_{1_0}} \succ hash(h)$ then $\alpha^{r_{1_0}} \succ h$.
- **Exp** – $\langle -\alpha^{s_1 \ldots s_{i-1} s_{i+1} \ldots s_n}, -s_i, +\alpha^s\rangle$ – If $\alpha^{r_{1_0}} \succ \alpha^s$ then $\alpha^{r_{1_0}} \succ \alpha^{s_1 \ldots s_{i-1} s_{i+1} \ldots s_n}$.

It can be observed that in all the above cases $\alpha^{r_{1_0}}$ is not a constituent of a positive node. Hence there is no penetrator strand where $\alpha^{r_{1_0}}$ can arise. The proof for $\alpha^{r_{2_0}}, \alpha^{r_{3_0}}, \alpha^{r_{1_0} r_{2_0}}, \alpha^{r_{1_0} r_{3_0}}$ and $\alpha^{r_{2_0} r_{3_0}}$ is similar and so the protocol assures IKA. The generalization of the proof for n participants is a straight forward extension of the above proof. Thus SAT-GDH protocol satisfies IKA. □

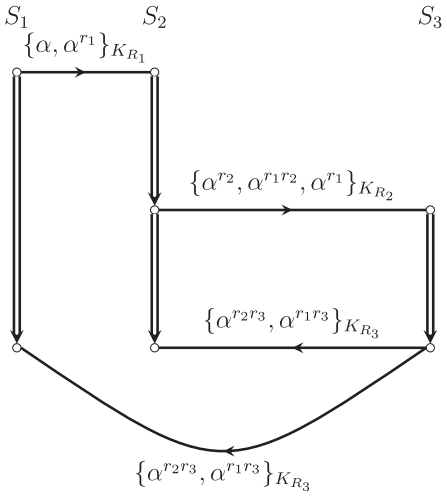

Fig. 5. Strand Space representation of three-party SAT-GDH protocol.

Table 1
Comparison of security properties of protocols.

Protocol	Secrets of participants	Secrecy	Authentication	IKA
Two-party DH	Short-term keys	✓	×	×
Simplified TLS	Short-term keys, Signature Keys	✓	✓	✓
GDH-2	Short-term keys	✓	×	×
A-GDH-2	Short-term keys, Long term keys shared with the Group Controller	✓	×	×
SAT-GDH-2	Short-term keys, Signature Keys	✓	✓	✓

A comparison of the security properties satisfied by Two-party DH protocol (Diffie and Hellman, 1976), Simplified TLS (Herzog, 2003), GDH-2 (Ateniese et al., 1998), A-GDH-2 (Ateniese et al., 1998), SAT-GDH-2 (Pereira, 2003) protocols are shown in the Table 1.

6. Discussion

Our work extends the proposals made in Herzog (2003). In this section, we compare our proposals with that of Herzog (2003).

The correctness condition that we have defined under Section 3.2.2 is similar to that of Herzog (2003). However the corresponding property being verified is IKA in our proposal, whereas it is Secrecy in Herzog (2003). This is achieved as we make use of parametric strands and the values are bound to the parameters whenever the terms are signed.

The correctness condition proposed in Herzog (2003) is for two party Diffie–Hellman based protocols, whereas our proposal is applicable for group key agreement protocols.

Similarly, the secrecy condition can be compared to the definition of silent protocols of Herzog (2003). But the definition of silent protocols (Herzog, 2003) cannot be applied to SAT-GDH or similar protocols, whereas they do satisfy secrecy condition. Hence our model can be used for analysis of such protocols too.

7. Conclusion

In this article, we have proposed a technique to verify the correctness of Implicit Key Authentication property of group key agreement protocols. Normally IKA property of protocols are established by proving that the protocol satisfies authentication of participants and secrecy of the group key. We have defined a single correctness condition exclusively for IKA. The proposed technique is based on conventional Strand Spaces analysis i.e. partial ordering. We have illustrated our proposal on a simple protocol based on TLS and simplified version of AT-GDH protocol. The soundness of our proposal has been established. The proposal enhances the ability of Strand Spaces model without sacrificing its existing capabilities.

References

Abadi, M., Gordon, A.D., 1997. A calculus for cryptographic protocols: the Spi Calculus. In: Proceedings of the 4th ACM Conference on Computer and Communications Security, pp. 36–47.
Amir, Y., Kim, Y., Nita-Rotaru, C., Tsudik, G., 2002. On the performance of group key agreement protocols, distributed computing systems. In: Proceedings. 22nd International Conference on (2002), pp. 463–464.
Ateniese, G., Steiner, M., Tsudik, G., 1998. Authenticated group key agreement and friends. In: ACM Conference on Computer and Communications Security. ACM, pp. 17–26.

Ateniese, G., Steiner, M., Tsudik, G., 2000. New multiparty authentication services and key agreement protocols. IEEE J. Sel. Areas Commun., 628–639
Bellare, M., Rogaway, P., 1993. Entity authentication and key distribution, lecture notes in computer science. In: Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology, pp. 232–249.
Blake-Wilson, S., Menezes, A., 1999. Authenticated diffe-hellman key agreement protocols. Sel. Areas Cryptogr., 339–361
Boreale, M., Buscemi, M.G., 2003. On the symbolic analysis of low-level cryptographic primitives: modular exponentiation and the Diffie–Hellman protocol. In: Proceedings of Workshop on the Foundations of Computer Security (FCS), p. 85.
Bresson, E., Chevassut, O., Pointcheval, D., 2002. Dynamic group Diffie–Hellman key exchange under standard assumptions. In: Proceedings of EUROCRYPT 2002, LNCS 2332. Springer-Verlag, pp. 321–336.
Bull, J.A., Otway, D.J., 1997. The Authentication Protocol, Technical Report DRA/CIS3/PROJ/CORBA/SC/1/CSM/436-04/03. Defence Research Agency, Malvern, UK.
Burmester, M., Desmedt, Y., 1994. A secure and efficient conference key distribution system. Lect. Notes Comput. Sci., 274–286
Burrows, M., Abadi, M., Needham, R., 1990. A logic of authentication. ACM Trans. Comput. Syst. (TOCS) 426 (1871), 18–36.
Chevalier, Y., Kusters, R., Rusinowitch, M., Turuani, M., 2008. Complexity results for security protocols with Diffie–Hellman exponentiation and commuting public key encryption. ACM Trans. Comput. Logic (TOCL) 9, 1–52.
Delaune, S., Lafourcade, P., Lugiez, D., Treinen, R., 2008. Symbolic protocol analysis for monoidal equational theories, information and computation. Elsevier 206 (2), 312–351.
Delicata, R., Schneider, S., 2003. A Formal Model of Diffie–Hellman Using CSP and Rank Functions, Technical Report CSD-TR-03-05. University of London, Royal Holloway.
Diffie, W., Hellman, M.E., 1976. New directions in cryptography. IEEE Trans. Inf. Theory IT-22, 644–654.
Dougherty, D.J., Guttman, J.D., 2013. An Algebra for Symbolic Diffie-Hellman Protocol Analysis. In: Proceedings of the 7th International Symposium on Trustworthy Global Computing, pp. 164–181.
Dougherty, D.J., Guttman, J.D., 2014. Decidability for Lightweight Diffie-Hellman Protocols. In: Proceedings of IEEE 27th Computer Security Foundations Symposium, pp. 217–231.
Durgin, N.A., Mitchell, J.C., 1999. Analysis of security protocols. Calculational Syst. Des., 369–395
Fabrega, F.J.T., Herzog, J.C., Guttman, J.D., 1999. Strand spaces: proving security protocols correct. J. Comput. Secur. 7 (2/3), 191–230.
Goubault-Larrecq, J., Roger, M., Neeraj Verma, Kumar, 2005. Abstraction and resolution modulo AC: How to verify Diffie–Hellman-like protocols automatically. J. Logic Algebraic Program. 64 (2), 219–251.
Guttman, J.D., Fabrega, F.J.T., 2000. Authentication tests. In: Proceedings of the 2000 IEEE Symposium on Security and Privacy, pp. 96–109.
Herzog, J.C., 2003. The Diffie–Hellman key-agreement scheme in the strand-space model. In: Proceedings of the 16th IEEE Computer Security Foundations Workshop, pp. 234–247.
Hoare, C.A.R., 2004. Communicating Sequential Processes. Prentice Hall International.
Just, M., Vaudenay, S., 1996. Authenticated multi-party key agreement. In: Proceedings of Advances in Cryptology ASIACRYPT'96, pp. 36–49.
Kapur, D., Narendran, P., Wang, L., 2003. An E-unification algorithm for analyzing protocols that use modular exponentiation. In: International Conference on Rewriting Techniques and Applications, pp. 165–179.
Kim, Y., Perrig, A., Tusidk, G., 2001. Communication-efficient grop key agreement. In: The Proceedings of International Fedaration for Information Processing (IFIP SEC), pp. 229–244.
Kim, Y., Perrig, A., Tsudik, G., 2004. Tree-based group key agreement. ACM Trans. Inf. Syst. Secur. 7, 60–96.
Krawczyk, H., 2005. HMQV: a high-performance secure Diffie–Hellman protocol. In: Proceedings of Annual International Cryptology Conference, pp. 546–566.
Lowe, G., 1995. An attack on Needham–Schroeder public-key authentication protocol. Inf. Process. Lett. 56 (3), 131–133.
Manulis, M., 2006. Survey on Security Requirements and Models for Group Key Exchange, Technical Report TR-HGI-2006-002. Horst Gortz Institute for IT Security, Bochum, Germany. 1–47.
Menezes, A.J., van Oorschot, P.C., Vanstone, S.A., 1996. Handbook of Applied Cryptography. CRC Press.
Millen, J., Shamatikov, V., 2005. Symbolic protocol analysis with an abelian group operator or Diffie–Hellman exponentiation. J. Comput. Secur. 13 (3), 515–564.
Mishra, D., Das, A., Mukhopadhyay, S., 2014. A secure user anonymity-preserving biometric-based multi-server authenticated key agreement scheme using smart cards. Expert Syst. Appl., 8129–8143
Needham, R.M., Schroeder, M.D., 1978. Using encryption for authentication in large network of computers. Commun. ACM 21 (12), 993–999.
Paulson, L., 1997. Mechanized proofs for a recursive authentication protocol. In: Computer Security Foundations Workshop, 1997. Proceedings., 10th, pp. 84–94.
Paulson, L.C., 1997. Proving properties of security protocols by induction. In: Proceedings of 10th Computer Security Foundations Workshop, pp. 70–83.
Pereira, O., 2003. Modelling and Security Analysis of Authenticated Group Key Agreement Protocols (Ph.D. thesis). Universite Catholique de Louvain, Belgique.
Ralf Kusters, R., Truderung, T., 2009. Using proverif to analyze protocols with Diffie–Hellman exponentiation. In: Proceedings of the 22nd IEEE Computer Security Foundations Symposium (CSF 2009), pp. 157–171.
Ramanujam, R., Suresh, S., 2003. A decidable subclass of unbounded security protocols. In: Proceedings of Workshop on Issues in the Theory of Security (WITS 2003), pp. 11–20.
Ryan, P., Schneider, S., 1998. An attack on a recursive authentication protocol A cautionary tale. Inf. Process. Lett. 65, 7–10.
Santiago, E., Catherine, M., Jos, M., 2007. Maude-npa: Cryptographic protocol analysis modulo equational properties. Lect. Notes Comput. Lect., Springer 5705, 1–50.
Schmidt, B., Meier, S., Cremers, C., Basin, D., 2012. Automated analysis of Diffie–Hellman protocols and advanced security properties. In: The Proceedings of 25th IEEE Computer Security Foundation (CSF) Symposium, pp. 78–94.
Schneider, S., 1996. Using CSP for Protocol Analysis: The Needham–Schroeder Public-key Protocol, Technical Report, CSD-TR-96-14 Royal Holloway. University of London.
Song, D., 1999. Athena: a new efficient model checker for security protocol analysis. In: Proceedings of the Twelth IEEE Computer Security Foundations Workshop, pp. 192–202.
Steer, D., Strawczynski, L., Diffie, W., Weiner, M., 1990. A secure audio conferencing system. In: The Proceedings of Advances in Cryptology, CRYPTO 80, pp. 520–528.