Formal Protocol Design of ESIKE Based on Authentication Tests

Rui Jiang[1], Aiqun Hu[1], and Jianhua Li[2]
(Corresponding author: Rui Jiang)
School of Information Science and Engineering, Southeast University[1]
No. 2 Sipailou Road, Nanjing, Jiangsu, China 210096 (Email: R.Jiang@seu.edu.cn)
Department of Electronic Engineering, Shanghai Jiao Tong University, China 200030[2]

(Received Apr. 6, 2006; revised and accepted Aug. 1, 2006 & Nov. 8, 2006)

Abstract

In this paper, we first present a concrete formal protocol design approach, which is based on authentication tests, to create an Efficient and Secure Internet Key Exchange (ESIKE) protocol. Then we formally prove the secure properties of ESIKE with strand space model and authentication tests. The ESIKE protocol overcomes the security shortages of the Internet Key Exchange (IKE), and can provide secure negotiation of session key and Security Association (SA), protection of endpoints' identities, and mutual authentication between the initiator and the responder. It needs only three messages and less computational load, so it is simple and efficient.

Keywords: Authentication tests, formal method, internet key exchange, protocol design, strand space model

1 Introduction

The Internet Key Exchange (IKE) protocol specified in RFC2409 [7] is a key exchange protocol being developed for the Internet Community. It is designed to establish Security Associations (SAs) and obtain authenticated keying material for use with ISAKMP [9] and other security services such as AH and ESP for IPSec. It works in two phases. Phase 1 establishes an ISAKMP SA and derives shared secrets that will be used to protect Phase 2 exchanges. Phase 2 negotiates SAs for IPSec and generates fresh keying material. In addition, the IKE protocol defines three basic modes of exchanges: main mode and aggressive mode used in Phase 1, and quick mode used in Phase 2.

However, there are many security shortages in IKE. Meadows [10] used the NRL Protocol Analyzer to point out an attack on the authentication of IDs to digital signatures in aggressive mode in Phase 1, and a reflection attack to quick mode in Phase 2. Zhou pointed out in [15] an attack on the authentication of ISAKMP SA to main mode in Phase 1, and then pointed out in [16] the failure of identity protection with digital signature in main mode, the weakness of support for nomadic user with pre-shared key in main mode, and the weakness about use of certificates with public key encryption in main mode. Perlman and Kaufman pointed out in [11] that IKE is far too complex, the Phase 2 should be removed, the specifications are too difficult to understand, and it is only possible to hide one endpoint's identity in some modes. Aiello et al. [2] presented a new key exchange protocol named JFK, which stands for "Just Fast Keying", to overcome the deficiencies of IKE such as the high number of message exchange rounds, the complexity of the protocol and its specification. And now the design of IKEv2 [8] draft is still underway.

On the other hand, much work has been done on protocol design. The bulk of work on protocol design such as Abadi and Needham [1] seems to rely on the skill and ingenuity of the designer, but they make no claim to be systematic, nor do they base their advice on a theory of protocol goals and correctness. Woo and Lam [14] focused on how to safely remove information from a "full information" but inefficient version of a protocol to a less cluttered version. There are two limitations to their approach. One is no guidance for how to construct a full information protocol to achieve given goals. The other is the criteria for safely removing information seems fragile. Buttyan et al. [3] describes a BAN-style logic to motivate a design method, but it seems hard to abstract the method from the example they give. Based on the authentication tests [4], which are developed from the strand space model [6], Guttman [5] describes an abstract formal protocol design process, and correctly illustrates its use by creating AT-SPECT, an Authentication Test-based Secure Protocol for Electronic Commerce Transactions. Perrig and Song use their automated protocol generator APG [12], which is related to authentication tests method, to generate candidate three-party authentication and key agreement protocol, then call Athena [13] to use the strand space model to filter protocol, and obtain suitable protocol proved to

meet their specifications.

In this paper, we present our concrete formal protocol design approach, which is based on the authentication tests and strand space model, to design the Efficient and Secure Internet Key Exchange (ESIKE) protocol. Our protocol design approach gives a concrete approach rather than an abstract protocol design process in [5], and is simple and efficient to avoid the endless state search used in [12, 13]. The ESIKE protocol is secure in protection of negotiation of session key and SAs, in protection of endpoint's identity, is able to obtain mutual authentication between the initiator and the responder, and can withstand the above attacks to the IKE protocol. In addition, the ESIKE only contains three messages and needs less computational load, so it is more simple and efficient than IKE, JFK and IKEv2 (which is only a draft of IETF and is still working in progress). The remainder of the paper is organized as follows. In Section 2, we claim the ESIKE security goals. In Section 3, we first briefly introduce the strand space model and the authentication tests, then present our protocol design approach to create the ESIKE protocol. In Section 4, we formally prove the correctness of the ESIKE protocol with strand space model and authentication tests, then we discuss the efficiency of the protocol. Section 5 concludes the paper.

2 ESIKE Protocol Goals

The causes of security shortages of IKE described in Section 1 are the insecure negotiation of Security Association (SA) and session key, the disclose of endpoint's identity, and short of mutual authentication between the initiator and the responder. Therefore, our security goals in design ESIKE are to provide mutual authentication between the initiator and the responder, and provide the secure negotiation of session key and SAs, and the protection of two parties' identities, which lead to the confidentiality for certain value in two parties protocol exchange. In addition, the ESIKE must contain as fewer messages as possible.

2.1 Protocol Participants

Protocol participants play two different roles, typically a client and a server, or in other words, an initiator and a responder. We will refer to the two principals as I and R. All the secret data such as principal's identity, SA, and information of session key must remain confidential from principals other than these two.

The same principal may play different roles in different protocol executions. When different clients order service to each other, they alternately play the role of I and R.

2.2 Protocol Goals

The goals of the participants are of four kinds:

- **Confidentiality**: All important data such as principal's identity, SA, and information of session key transmitted in the exchange are to remain secret, and data intended for a pair should not be disclosed to the others.
- **Authentication 1**: Each participant I should receive a guarantee that each partner R has received I's data and R accepted it.
- **Authentication 2**: Each participant R should receive a guarantee that data purportedly from a partner I in fact originated with R, freshly in a recent run of this protocol.
- **Efficiency**: The protocol must be efficient with respect to the number of message exchanges, computation and bandwidth in the communication.

3 Authentication Tests and ESIKE Design

In this section, we first introduce the basic ideas of the strand space model [4], then introduce the basic definitions of the authentication tests [6], which are based on the strand space theory. Finally, we present our formal protocol design approach to create the ESIKE protocol.

3.1 Strand Space Model

Consider a set A, the elements of which are the possible messages that can be exchanged between principals in a protocol. We will refer to the element of A as terms t . $t \sqsubset t$ means is a subterm of t. The set A is constrained further below in Section 3.1.2, and the subterm relation is defined there. We will represent sending a term as the occurrence of that term with positive sign, and receiving a term as its occurrence with a negative sign.

Definition 1. *A signed term is a pair* (σ, a) *with* $a \in A$ *and* σ *one of the symbols* $+, -$. *We will write a signed term as* $+t$ *or* $-t$. $(\pm A)^*$ *is the set of finite sequences of signed terms. We will denote a typical element of* $(\pm A)^*$ *by* $((\sigma_1, \sigma_1), \ldots, (\sigma_n, \sigma_n))$.

Definition 2. *A strand space is a set* Σ *with a trace mapping* $tr : \Sigma \rightarrow (\pm A)^*$.

Fix a strand space Σ.

1) A node is a pair (s, i), with $s \in \Sigma$ and i an integer satisfying $1 \leq i \leq length(tr(s))$. The set of nodes is denoted by N. We will say the node (s, i) belongs to the strand s. Clearly, every node belongs to a unique strand.

2) If $n = (s, i) \in$N then index$(n) = i$ and strand $(n) = s$. Define term(n) to be $((tr(s))_1)_2$, i.e. the ith signed term in the trace of s . Similarly, uns_term(n) is $((tr(s))_1)_2$, i.e. the unsigned part of the ith signed term in the trace of s.

3) If $n_1, n_2 \in$ N, $n_1 \rightarrow n_2$ means term$(n_1) = +a$ and term$(n_2) = -a$. It means that node n_1 sends the message a, which is received by n, creating a casual link between their strands.

4) If $n_1, n_2 \in$ N, then $n_1 \Rightarrow n_2$ means n_1, n_2 occur on the same strand with index(n_1) =index$(n_2) - 1$. It expresses that n_1 is an immediate causal predecessor of n_2 on the strand.

5) An unsigned term t occurs in $n \in$N iff $t \sqsubset$term(n).

6) An unsigned term t originates on $n \in$N iff: term(n) is positive; $t \sqsubset$term(n) ; and whenever n' precedes n on the same strand, $t \not\sqsubset term(n')$.

7) An unsigned term t is uniquely originating iff t originates on a unique $n \in$ N.

N becomes an ordered graph with both sets of edges $n_1 \rightarrow n_2$ and $n_1 \Rightarrow n_2$.

3.1.1 Bundles

A bundle is a finite subgraph of this graph, for which we can regard the edges as expressing the causal dependencies of the nodes.

Definition 3. *Let $\mathcal{C}$ be a set of edges, and let N_c be the set of nodes incident with any edge in $\mathcal{C}$. $\mathcal{C}$ is a bundle if:*

1) $\mathcal{C}$ is finite.

2) If $n_1 \in N_c$ and term(n_1) is negative, then there is a unique n_2 such that $n_2 \rightarrow n_1 \in \mathcal{C}$.

3) If $n_1 \in N_c$ and $n_2 \Rightarrow n_1$, then $n_2 \Rightarrow n_1 \in \mathcal{C}$.

4) $\mathcal{C}$ is acyclic.

Definition 4. *A node n is in a bundle $\mathcal{C}$, written $n \in \mathcal{C}$, if $n \in N_c$; a strand s is in a bundle if all of its nodes are in N_c.*

3.1.2 Terms and Encryption

The terms and encryption of strand space model are described as follows:

- A set $T \subset$A of texts (representing the atomic messages), and a disjoint set $K \subset$A of cryptographic keys.
- A unary operator inv: $K \rightarrow K$. The inv maps each member of a key pair for an asymmetric cryptosystem to the other, and that it maps a symmetric key to itself.
- Two binary operators:

$$encr : K \times A \longrightarrow A$$
$$join : A \times A \longrightarrow A.$$

As usual, inv(K) is denoted as K^{-1}, encr(K, m) as $\{|m|\}_K$, and join(a, b) as ab.

Definition 5. *The subterm relation $\sqsubset$ is defined inductively, as the smallest relation such that $a \sqsubset a$; $a \sqsubset \{|g|\}_K$ if $a \sqsubset g$; $a \sqsubset gh$ if $a \sqsubset g$ or $a \sqsubset h$.*

3.1.3 The Penetrator

The penetrator's powers are characterized by two ingredients, namely a set of keys known initially to the penetrator and a set of penetrator strands that allow the penetrator to generate new messages from messages he intercepts.

A penetrator set consists of a set of keys K_p, which are initially known to the penetrator.

The atomic actions available to the penetrator are encoded in a set of penetrator traces. A protocol attack typically requires hooking together several of these atomic actions.

Definition 6. *A penetrator trace is one of the following:*

M	*Text message: $+t$ where $t \in T$;*
F	*Flushing: $(-l)$;*
T	*Tee: $(-l, +l, +l)$;*
C	*Concatenation: $(-l, -h, +lh)$;*
S	*Separation into components: $(-lh. + l, +h)$;*
K	*Key: $(+K)$ where $k \in K_p$;*
E	*Encryption: $(-K, -h, +\{|H|\}_K)$;*
D	*Decryption: $(-K^{-1}, -\{|H|\}_K, +h)$.*

Definition 7. *An infiltrated strand space is a pair (Σ, Π) with Σ a strand space and $\Pi \subseteq \Sigma$ such that $tr(P)$ is a penetrator trace for all $P \in \Pi$. A strand $s \in \Sigma$ is a penetrator strand if it belongs to Π, and a node is a penetrator node if the strand it lies on is a penetrator strand. Otherwise we will call it a regular strand or node.*

3.2 The Authentication Tests

Fix some strand space Σ. We identify segments of regular strands that amount to tests. Their presence will guarantee the existence of other regular strands in the bundle.

Definition 8. *A term t_0 is a component of t if $t_0 \sqsubset t$, t_0 is not a concatenated term, and every $t_1 \neq t_0$ such that $t_0 \sqsubset t_1 \sqsubset t$ is a concatenated term.*

Components are either atomic values or encryptions. For instance, the three components of the concatenated term $B\{|N_aK\{|KN_b|\}_{K_B}|\}_{K_A}N_a$ are B, $\{|N_aK\{|KN_b|\}_{K_B}|\}_{K_A}$, and N_a. We say t is a component of a node n if t is a component of $term(n)$.

Definition 9. *The edge $n_1 \Rightarrow^+ n_2$ is a transformed edge [respectively, a transforming edge] for $a \in A$ if n_1 is positive and n_2 is negative [respectively, n_1 is negative and n_2 is positive], $a \sqsubset term(n_1)$, and there is a new component t_2 of n_2 such that $a \sqsubset t_2$.*

Thus, a transformed edge emits and later tests for its presence in a new form. A transforming edge receives and later emits it in a new form.

Definition 10. *$t = \{|h|\}_K$ is a test component for a in n if:*

1) *$a \sqsubset t$ and t is a component of n;*
2) *The term is not a proper subterm of a component of any regular node $n' \in \Sigma$.*

The edge $n_0 \Rightarrow^+ n_1$ is a test for a if a uniquely originates at n_0 and $n_0 \Rightarrow^+ n_1$ is a transformed edge for a.

Definition 11. *The edge $m_0 \Rightarrow^+ m_1$ is an outgoing test for a in $t = \{|h|\}_K$ if it is a test for a in which: $K^{-1} \notin K_p$; a does not occur in any component of m_0 other than t; and t is a test component fora in m_0. The edge $m_0 \Rightarrow^+ m_1$ is an incoming test for a in $t_1 = \{|h|\}_K$if it is a test for a in which $K \notin K_p$ and t_1 is a test component for a in m_1.*

(Outgoing test) Let $\mathcal{C}$ be a bundle with $m_1 \in \mathcal{C}$, and let $m_0 \Rightarrow^+ m_1$ be an outgoing test for a in t.

1) *There exist regular nodes n_0, $n_1 \in \mathcal{C}$ such that t is a component of n_0 and $n_0 \Rightarrow^+ n_1$ is a transforming edge for a.*
2) *Suppose in addition that a occurs only in component $t_1 = \{|h_1|\}K_1$ of n_1, that t_1 is not a proper subterm of any regular component, and that $K_1^{-1} \notin K_p$. then there is a negative regular node with t_1 as a component.*

The meaning of this assertion is illustrated in Figure 1.

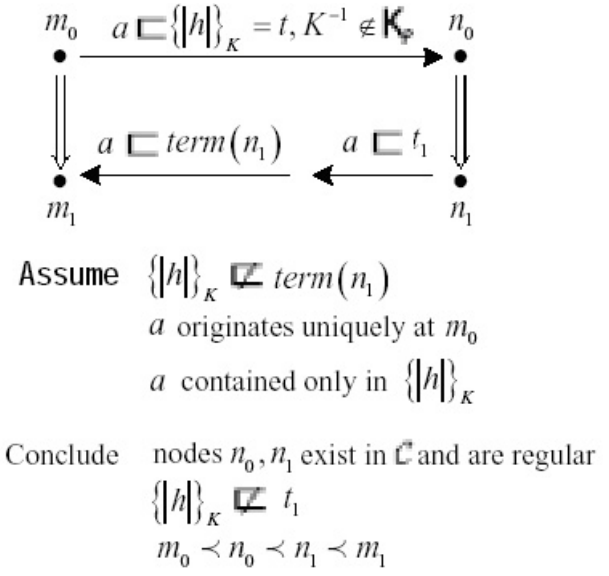


Figure 1: Outgoing authentication test

(Incoming test) Let $\mathcal{C}$ be a bundle with $m_1 \in \mathcal{C}$, and let $m_0 \Rightarrow^+ m_1$ be an incoming test for a in t'. Then there exist regular nodes n_0, $n_1 \in \mathcal{C}$ such that t' is a component of n_1 and $n_0 \Rightarrow^+ n_1$ is a transforming edge for a.

The meaning of this assertion is illustrated in Figure 2.

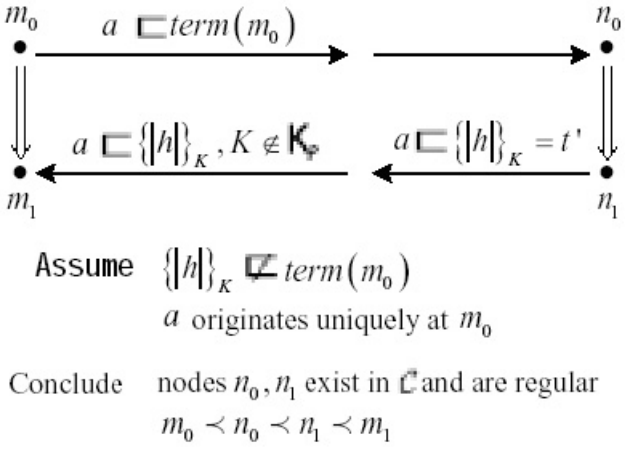


Figure 2: Incoming authentication test

Definition 12. *A negative node m is an unsolicited test for $t = \{|h|\}_K$ if is a test component for any a in m and $K \notin K_p$.*

(Unsolicited test) Let $\mathcal{C}$ be a bundle with $m \in \mathcal{C}$ and let m be an unsolicited test for $t = \{|h|\}_K$. Then there exists a positive regular node $n \in K$ such that t is a component of n.

Definition 13. *(Recency) A node n is recent for a regular node m_1 in $\mathcal{C}$ if there is a regular node $m_0 \in \mathcal{C}$ such that $m_0 \Rightarrow^+ m_1$ and $m_0 \leq_c np_c m_1$.*

The incoming test and the outgoing test entail recency. That is, if $m_0 \Rightarrow^+ m_1$ is a test edge, and $n_0 \Rightarrow^+ n_1$ is the corresponding transforming edge in $\mathcal{C}$, then $m_0 p n_0 p n_1 p m_1$, so that n_0 and n_1 are recent for m_1. By contrast, the unsolicited test establishes nothing about recency.

Definition 14. *(n-Recency) A node n is 1-recent for m_1 if n is recent for m_1 as in Definition 13. A node n is $i+1$-recent for m_1 if there exists a node m_0 such that n is i-recent for m_0 and m_0 is recent for m_1.*

3.3 Protocol Design

The authentication tests suggest a protocol design process. At this level of abstraction, authentication protocol design is a matter of selecting authentication tests, and constructing a unique regular transforming edge to satisfy each. We now consider how to get them with authentication tests according to our concrete ESIKE protocol security goals.

3.3.1 Assumptions and Notations

It is reasonable that we assume that each principal has at least one public-private key pair. The public key may be used to encrypt the messages or verify the signature, and the private key may be used either to decrypt the encryptions or sign the messages. We assume that the public keys for any participant can be determined reliably through a public key infrastructure. When | is a principal with public encryption key K_I, we write $\{|t|\}_I$ to stand for $\{|t|\}_{K_I}$. Assuming K_I is uncompromised (i.e. $K_I \in S$), only | can tractably recover t from this encryption. Likewise, when | is a principal with private signature key S_I, we write $[t]_I$ to stand for $\{|t|\}|_{S_I}$. We assume that only | can tractably construct $[t]_I$ from a new message t.

We need one cryptographic quality primitive $h(x)$, which is a one-way hash function. $h(t)$ is the result of applying the hash function to t. We assume that no principal can tractably find a pair of values t_1, t_2 such that $h(t_1) = h(t_2)$, or, given v, can tractably find t such that $h(t) = v$.

3.3.2 Payloads and Confidentiality

The purpose of ESIKE is to generate secure session key, protect two partners' identities, securely negotiate SA for IPSec in communication process, and obtain the mutual authentications for two principals. Therefore, according to the security goals described in Section 2.2, we specify the message components as payloads used in ESIKE protocol to satisfy the above purpose. We describe them as follows:

SA_I Cryptographic and service properties of the security association (SA) that the initiator wants to establish.

SA_R SA information the responder may need to give to the initiator (e.g., the responder's SPI in IPSec).

N_I Initiator nonce, a random bit-string.

N_R Responder nonce, a random bit-string.

KE_I Initiator's current Diffie-Hellman (DH) exponential.

KE_R Responder's current Diffie-Hellman (DH) exponential.

ID_I Initiator's certificates or public-key identifying information.

ID_R Responder's certificates or public-key identifying information.

$h(M)$ hash of message M. It also implies that $h(x)$ is a secure message authentication code (MAC) function.

In order to satisfy the confidentiality goal of ESIKE, we determine the data of SA_I, SA_R, ID_I, ID_R, KE_I, KE_R should be securely protected in the communication. Then we can derive the session key $K_{IR} = H_{DH}(N_I, N_R)$ securely, where $H_k(M)$ is a keyed hash of message M using key k. In ESIKE, the K is the DH exchanged key, which can be derived form KE_I and KE_R.

3.3.3 Designing the ESIKE Protocol

In this section, we design the ESIKE protocol step by step to satisfy the protocol goals one by one.

Achieving Confidentiality

The confidentiality goal is the assertion: All important data transmitted in the exchange are to remain secret, and data intended for a pair should not be disclosed to the others.

Therefore, we should not let the data of SA_I, SA_R, ID_I, ID_R, KE_I, KE_R transmitted in plaintext in the exchange. We should transmit them in encryption form with public key K_I or K_R, or with a hash function of $h(t)$, which satisfy the assumption described in Section 3.3.1.

Achieving Authentication 1

The first authentication goal is the assertion: Each participant | should receive a guarantee that each partner R has received |'s data and R accepted it.

|'s data means the data of SA_I, ID_I and KE_I, which we know must be transmitted in the form of $\{|SA_I, KE_I, ID_I|\}_R$. The incoming authentication test tells us that one way to ensure "authentication 1" is to prepare a nonce N_I, transmitting n_I with $\{|SA_I, KE_I, ID_I|\}_R$. After receiving and processing this unit, R returns an authentication message taking the form $[\ldots N_I \ldots]_R$, which prove that N_I reached R and was accepted as part of a successful strand.

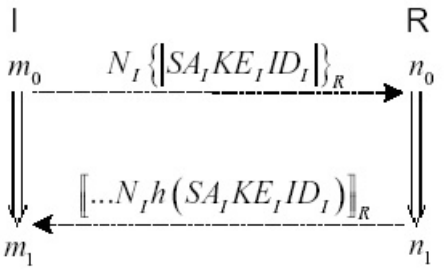


Figure 3: Edges achieving Authentication 1

We also want to ensure that N_I was accompanied by the payloads SA_I, ID_I and KE_I when it was processed. Therefore, we will require the authentication message taking the form $[\ldots N_I t]_R$, where t contains the payloads in some other form. Specially, in order to maintain the confidentiality of the payloads, they should be hashed or encrypted. We decide to use $h(SA_I KE_I ID_I)$ rather than the encrypted component $\{|SA_I KE_I ID_I|\}$ in authentication message to make the protocol more efficient and need less computation. So, we may have the authentication message $[\ldots N_I h(SA_I KE_I ID_I)]_R$. We now have the behavior shown in Figure 3. This is evidently an incoming test for | assuming that R's private signature key is uncompromised and N_I is uniquely originating.

Achieving Authentication 2

The second authentication goal is the assertion: Each participant R should receive a guarantee that data purportedly from a partner | in fact originated with R, freshly in a recent run of this protocol.

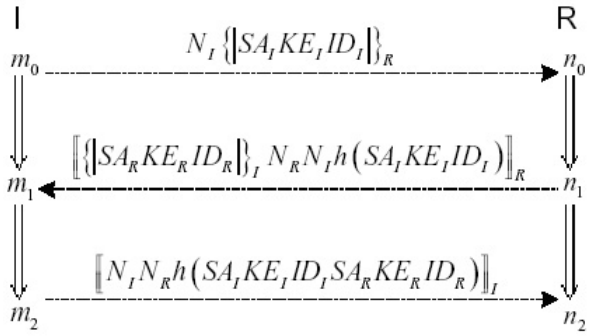


Figure 4: Edges achieving Authentication 2

In order to achieve this authentication goal, we must extend the protocol. In particular, it originates at a 2-recent node (Definition 14). R's data means the data of SA_R, KE_R and ID_R, which we know must be transmitted with encryption by |'s public key. The incoming authentication test gives us a way to ensure the authentication of the R's payload is to prepare a nonce N_R, transmitting N_R with $\{|SA_R KE_R ID_R|\}_I$. We can combine R's sending message with |'s authentication 1 message, therefore, we enrich the protocol exchange displayed in Figure 3 by having R emit a uniquely originating value with R's secret data $\{|SA_R KE_R ID_R|\}_I$ in addition to the authentication message in Figure 3. So, the message has the form of $[\{|SA_R KE_R ID_R|\}_I N_R N_I h(SA_I KE_I ID_I)]_R$. After receiving and processing this unit, | signs N_I, N_R, and the hash of payloads in a recency certificate, taking the form $[N_I N_R h(SA_I KE_I ID_I SA_R KE_R ID_R)]_I$. This transforming edge completes an incoming test for R, assuming |'s private signature key is uncomprised and N_R is uniquely originating, as shown (right-to-left) in the lower rectangle in Figure 4.

R knows that this signed message was generated after N_R was created. Moreover, if | is behaving properly, then this signature is emitted only in a run that also caused the origination of N_I. Thus, m_2 is recent for n_2, and m_0 is recent for m_2. Therefore, m_0 is 2-recent for n_2.

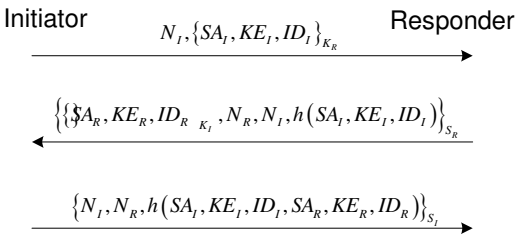


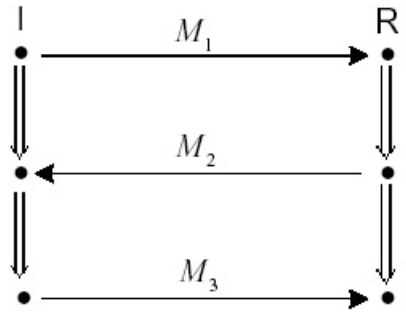
Figure 5: Our ESIKE protocol

Now, we can get the designed ESIKE protocol, which is illustrated in Figure 5. In Figure 5, $\{M\}_{K_R}$ and $\{K\}_{K_I}$ denote the message M encrypted with public key K_R and K_I respectively. $\{M\}_{S_I}$ denotes the pair $(M, Sig_{S_I}(M))$, where $Sig_{S_I}(M)$ denotes the signature of message with private key S_I. $\{M\}_{S_R}$ is similar to $\{M\}_{S_I}$, in which the private key is $\{M\}_{S_R}$. In the protocol, the initiator and the responder only need three messages to negotiate the SA, protect each principal's identity, obtain the session key, which can be derived from KE_I, KE_R, N_I and N_R, and make sure the mutual authentication between each other.

4 Proving ESIKE Protocol Correct

In this section, we formally prove the correctness of ESIKE protocol. We apply the strand space model and the authentication tests to prove the ESIKE satisfies the security goals it want to have.

In the form we consider, the ESIKE protocol involves two types of regular strands and is depicted in Figure 6.



Where: $M_1 = N_I \{|SA_I KE_I ID_I|\}_{K_R}$

$M_2 = \{|\{|SA_R KE_R ID_R|\}_{K_I} N_R N_I h(SA_I KE_I ID_I)|\}_{S_R}$

$M_3 = \{|N_I N_R h(SA_I KE_I ID_I SA_R KE_R ID_R)|\}_{S_I}$

Figure 6: Regular bundle in ESIKE protocol

1) Initiator strands with trace: $(+M_1, -M_2, +M_3)$, where $ID_I, ID_R \in T_{name}$, $N_I, N_R, KE_I, KE_R, SA_I, SA_R \in T$ but $N_I, KE_I, SA_I \notin T_{name}$, $K_I, K_R \in K$ are public keys, $S_I, S_R \in K$ are private keys for signature, $K_I^{-1}, K_R^{-1} \in K$ are private keys for decryption. $Init[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ will denote the set of all strands with the trace shown.

2) Responder strands with trace: $(-M_1, +M_2, -M_3)$, where $ID_I, ID_R \in T_{name}$, $N_I, N_R, KE_I, KE_R, SA_I, SA_R \in T$ but $N_R, KE_R, SA_R \notin T_{name}$, $K_I, K_R \in K$ are public keys, $S_I, S_R \in K$ are private keys for signature, $K_I^{-1}, K_R^{-1} \in K$ are private keys for decryption. $Resp[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ will denote the set of all strands with the trace shown.

We now formulate the protocol goals as theorems about the ESIKE protocol, and prove that the initiator and the responder can securely exchange the session key, securely negotiate the SA, protect each other's identity, and obtain the mutual authentication to each other respectively.

Proposition 1. *(Confidentiality for \|'s data) Suppose $\mathcal{C}$ is a bundle in Σ, $ID_I, ID_R \in T_{name}$; $K_R^{-1} \notin K_p$; and $\mathcal{C}$ has an Init-strand $s \in Init[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ with $\mathcal{C}$-height 3. If $S_1 = \{SA_I, KE_I, ID_I\}$ is uniquely originating, then for every node $n \in \mathcal{C}$, $term(n) \notin S_1$.*

Proof. Let k be the set of the inverses of unsafe keys, i.e. $k = (K \backslash S)^{-1}$, where S is the set of secret keys. Let π be $S_1 \bigcup S$, i.e. $\pi = S_1 \bigcup S$.

According to the honest ideal of Corollary 6.12 in [4], if there is a node $m \in \mathcal{C}$ with $term(m) \in S_1$, then $term(m) \in I_k[\pi]$, and there is a regular node that is an entry point for $I_k[\pi]$. However, inspecting the positive regular nodes of bundle in $\sum$, we see from Figure 6 that no value in π is ever sent, only the value protected by a key K_R(in the form of $\{|SA_I, KE_I, ID_I|\}_{K_R}$) whose inversed key is safe according the assumption $K_R^{-1} \notin K_p$, or protected by a one-way hash function (in the form of $h(SA_I, KE_I, ID_I)$ and $h(SA_I, KE_I, ID_I, SA_R, KE_R, ID_R)$), in which who can not find t from $h(t) = \nu$ according to the assumption in Section 3.3.1. So, it causes the contradiction. Therefore, for every node $n \in \mathcal{C}$, $term(m) \notin S_1$. □

Proposition 1 proves that the secret information $S_1 = \{SA_I, KE_I, ID_I\}$ in ESIKE protocol sent by | will not be disclosed unless the penetrator possesses the private key of the responder, i.e. $K_R^{-1} \in K_p$.

Proposition 2. *(Confidentiality for R's data) Suppose $\mathcal{C}$ is a bundle in Σ, $ID_D, ID_R \in T_{name}$; $K_I^{-1} \notin K_p$; and $\mathcal{C}$ has a Resp-strand $s \in Resp[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ with $\mathcal{C}$-height 3. If $S_1 = \{SA_R, KE_R, ID_R\}$ is uniquely originating, then for every node $n \in \mathcal{C}$, $term(n) \notin S_1$.*

Proof. Let k be the set of the inverses of unsafe keys, i.e. $k = (K \backslash S)^{-1}$, where S is the set of secret keys. Let be $\pi = S_1 \bigcup S$.

According to the honest ideal of Corollary 6.12 in [4], if there is a node $m \in \mathcal{C}$ with $term(m) \in S_1$, then $term(m) \in I_k[\pi]$, and there is a regular node that is an entry point for $I_k[\pi]$. However, inspecting the positive regular nodes of bundle in Σ, we see from Figure 6 that no value in π is ever sent, only the value protected by a key K_I(in the form of $\{|SA_R, KE_R, ID_R|\}_{K_I}$) whose inversed key is safe according the assumption $K_R^{-1} \notin K_p$, or protected by a one-way hash function (in the form of $h(SA_I, KE_I, ID_I, SA_R, KE_R, ID_R)$) in which who can not find t from $h(t) = \nu$ according to the assumption in Section 3.3.1. So, it causes the contradiction. Therefore, for every node $n \in \mathcal{C}$, $term(n) \notin S_1$. □

Proposition 2 proves that the secret information $S_1 = \{SA_R, KE_R, ID_R\}$ in ESIKE protocol sent by R will not be disclosed unless the penetrator possesses the private key of the initiator, i.e. $K_I^{-1} \in K_p$.

Proposition 3. *(Authentication 1) Suppose $\mathcal{C}$ is a bundle in Σ, $ID_D, ID_R \in T_{name}$; $K_R^{-1} \notin K_p$; and $\mathcal{C}$ has a Resp-strand $s \in Init[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ with $\mathcal{C}$-height at least 2. If N_I is uniquely originating, the $\mathcal{C}$ has a matching Resp-strand $s' \in Resp[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ of $\mathcal{C}$-height at least 2.*

Proof. In Figure 6, we show first that the first and the second nodes on s form an incoming authentication test for N_I. We can know $\{|\{|SA_R, KE_R, ID_R|\}_{K_I} N_R N_I h(SA_I KE_I ID_I)|\}_{S_R}$ is a test component for N_I in $(s, 2)$, because it contains N_I, and no regular node has any term of this form as a proper subterm. Checking the assumptions, $S_R \notin K_p$, it follows that $(s, 1) \Rightarrow^+ (s, 2)$ is an incoming test for N_I in $\{|\{|SA_R, KE_R, ID_R|\}_{K_I} N_R N_I h(SA_I KE_I ID_I)|\}_{S_R}$ according to Definition 11.

By incoming test, there exist regular nodes n_0, $n_1 \in \mathcal{C}$ such that $\{|\{|SA_R, KE_R, ID_R|\}_{K_I} N_R N_I h(SA_I KE_I ID_I)|\}_{S_R}$ is a component of n_1 and $n_0 \Rightarrow^+ n_1$ is a transforming edge for N_I.

Because n_1 is a positive regular node and $\{|\{|SA_R, KE_R, ID_R|\}_{K_I} N_R N_I h(SA_I KE_I ID_I)|\}_{S_R} = term(n_1)$, N_I is uniquely originated in $(s, 1)$, then there exists a negative regular node n_0 to receive N_I. For n_0 is a negative node, it is at $(s', 1)$ for some Resp-strand $s' \in Resp[N_I, N_R, SA_I, SA'_R, KE_I, KE'_R, ID_I, ID'_R]$. Since $(s', 1) \Rightarrow^+ (s', 2)$ and $term((s', 2)) = \{|\{|SA_R KE_R ID_R|\}_{K_I} N_R N_I h(SA_I KE_I ID_I)|\}_{S_R}$ in which $\{|SA_R KE_R ID_R|\}_{K_I}$ is contained, we see $ID'_R = ID_R, SA'_I = SA_I, KE'_R = KE_R$. The $\mathcal{C}$-height of s' is at least 2. □

Proposition 3 proves, in ESIKE protocol, the initiator can guarantee the authentication to the responder when the assumptions are satisfied. In addition, because the scheme contains an incoming test forN_I , it can entail recency of N_I according to Definition 13. Therefore, the initiator of the ESIKE protocol also can prevent malicious reply attacks.

Proposition 4. *(Authentication 2) Suppose $\mathcal{C}$ is a bundle in Σ, $ID_D, ID_R \in T_{name}$; $K_R^{-1} \notin K_p$; and $\mathcal{C}$ has a Resp-strand $s \in Resp[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ with $\mathcal{C}$-height 3. If N_R is uniquely originating, the $\mathcal{C}$ has a matching Init-strand $s' \in Init[N_I, N_R, SA_I, SA_R, KE_I, KE_R, ID_I, ID_R]$ of $\mathcal{C}$-height 3, and $(s, 2)p(s', 2)$.*

Proof. In Figure 6, we show first that the second and the third nodes on s form an incoming authentication test for N_R. We know $\{|N_I N_R h(ID_I ID_R SA_I SA_R KE_I KE_R)|\}_{S_I}$ is a test

component for N_R in $(s,3)$, because it contains N_R, and no regular node has any term of this form as a proper subterm. Checking the assumptions, $S_I \notin K_p$, it follows that $(s,2) \Rightarrow^+ (s,3)$ is an incoming test for N_R in $\{|N_I N_R h(ID_I id_R SA_I SA_R KE_I ke_R)|\}_{S_I}$ according to Definition 11.

By incoming test, there exist regular nodes $n_0, n_1 \in \mathcal{C}$ such that $\{|N_I N_R h(ID_I id_R SA_I SA_R KE_I ke_R)|\}_{S_I}$ is a component of n_1 and $n_0 \Rightarrow^+ n_1$ is a transforming edge for N_R.

Because n_1 is a positive regular node and $term(n_1) = \{|N_I N_R h(ID_I id_R SA_I SA_R KE_I ke_R)|\}_{S_I}$, N_R is uniquely originated in $(s,2)$, then there exists a negative regular node n_0 to receive N_R. For n_0 is a negative node, it is at $(s',2)$ for some Init-strand $s' \in Init[N_I, N_R, SA_I, SA'_R, KE_I, KE'_R, ID_I, ID'_R]$, thus $(s,2)p(s',2)$. Since $(s',2) \Rightarrow^+ (s',3)$ and $term((s',3)) = \{|N_I N_R h(ID_I id_R SA_I SA_R KE_I ke_R)|\}_{S_I}$ in which ID_I, SA_I, KE_I is contained, we see $ID'_R = ID_R, SA'_I = SA_I, KE'_R = KE_R$. The $\mathcal{C}$-height of s' is 3. □

Proposition 4 proves the responder can correctly authenticate the initiator in the ESIKE protocol. In addition, since $(s,2)p(s',2)$, the node $(s',1)$, where N_I, ID_I, SA_I and originate, is 2-recent for $(s,3)$ according to Definition 14. The 2-recency of the protocol can also prevent the responder from malicious reply attacks.

We have now established the security goals of ESIKE. We may discuss the efficiency of the ESIKE as follows.

Efficiency Discussion:

In many protocols, key setup must be performed frequently enough that it can became a bottleneck to communication. The key exchange protocol must minimize the number of message exchange as well as computation and total bandwidth. The number of message exchange can be an especially important factor when communicating over unreliable media. Using our ESIKE protocol, only three messages are needed to set up a working security association. This is a considerable saving in comparison with existing protocols such as IKE, JFK, and IKEv2 draft.

In addition, the ESIKE protocol rejects the notion of two different phases in IKE. Phase 2 of IKE is used for several reasons. One is generating the actual keying material used for security associations. It is expected that this will be done several times, to amortize the expense of the Phase 1 negotiation. The second reason is to permit periodically keys changing. It is permitted to do key rollover of a Phase 2 connection by doing another Phase 2 connection setup, which would be more cheap than that of restarting the Phase 1 connection setup. The third reason is to permit multiple connections with different security properties and keys between two nodes. But we do not think these apply. First, one phase of negotiation to generate keying material is sufficient, and use two phases to generate other keying material is not significantly cheaper than doing another Phase 1 exchange. To the second point, with modern underlying block cipher such as AES, there is no need for frequent key changes. AES keys are long enough that brute force attacks are infeasible. Also, to do a key rollover for perfect forward secrecy, the Phase 2 exchange is not significantly cheaper than doing another Phase 1 exchange. To the third reason, it is a relatively rare case, and set up a totally unrelated security association for each application would suffice. Therefore, we remove the Phase 2 in IKE and design the ESIKE only having one phase. This makes ESIKE protocol more efficiently.

Table 1: Comparison of ESIKE, IKE, JFK, IKEv2 draft on computational load

Protocol	Computational Load	
	I	R
ESIKE	$2C_{EXP} + 1C_{HASH}$	$2C_{EXP} + 1C_{HASH}$
IKE*	$2C_{EXP} + 5C_{HASH}$	$2C_{EXP} + 5C_{HASH}$
JFK**	$3C_{EXP}$	$3C_{EXP} + 1C_{HASH}$
IKEv2***	$2C_{EXP} + 3C_{HASH}$	$2C_{EXP} + 3C_{HASH}$

Moreover, our ESIKE protocol has the advantage on computational load. The comparisons of IKE, JFK, IKEv2 draft and ESIKE on computational load are shown in Table 1, where C_{EXP}, C_{HASH} denote the exponentiation computation and the hash computation, respectively. I and R denote the protocol initiator and responder, respectively.

In Table 1, the example of IKE* is the pre-shared key main mode protocol, the example of JFK** is JFK$_r$, and the computational load of IKEv2*** draft only includes that in the initial exchange (known in IKE as Phase 1). Our ESIKE protocol has the relatively lower computational load and is more efficient.

5 Conclusion

Based on the authentication tests, a concrete formal protocol design approach is presented to create an Efficient and Secure Internet Key Exchange (ESIKE) protocol. At first, the secure goals of ESIKE are determined, then the protocol is designed step by step according to the secure goals one by one. Finally, it is formally proved with the strand space model and the authentication tests. The ESIKE protocol overcomes the security shortages of the Internet Key Exchange (IKE), and can provide secure negotiation of session key and Security Association (SA), protection of endpoints' identities, and mutual authentication between the initiator and the responder. It needs only three messages and less computational load, so it is simple and efficient.

Acknowledgements

This work was supported by National "863" High Technology Project of P. R. China under contract 2003AA142160, and now is supported by Natural Science Foundation of Jiangsu, P. R. China under grant no. BK2006108.

References

[1] M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," in *Proceedings of 1994 IEEE Symposium on Research in Security and Privacy*, pp. 122-136, May 1994.

[2] W. Aiello, S. M. Bellovin, M. Blaze, R. Canetti, J. Loannidis, A. D. Keromytis, and O. Reingold, "Efficient, DoS-resistant, secure key exchange for Internet protocols," in *ACM CCS'02*, pp. 27-39, Nov. 2002.

[3] L. Butty'an, S. Staamann, and U. Wilhelm, "A simple logic for authentication protocol design," in *11th IEEE Computer Security Foundations Workshop*, pp. 153-162, Rockport, Massachusetts, June 1998.

[4] F. J. T. Fabrega, J. C. Herzog, and J. D. Guttman, "Strand spaces: proving security protocols correct," *Journal of Computer Security*, vol. 7, no. 2-3, pp. 191-230, 1999.

[5] J. Guttman, "Security protocol design via authentication tests," in *Proceedings of the 15th IEEE Computer Security Foundation Workshop*, pp. 92-103, Grado, Italy, June 2002.

[6] J. D. Guttman and F. J. T. Fabrega, "Authentication tests and the structure of bundles," *Theoretical Computer Science*, vol. 283, no. 2, pp. 333-380, 2002.

[7] D. Harkins and D. Carrel, *The Internet Key Exchange (IKE)*, RFC 2409, Nov. 1998.

[8] C. Kaufman, *Internet Key Exchange (IKEv2) Protocol*, Internet Draft, Internet Engineering Task Force, Work in progress, Sep. 2005.

[9] D. Maughan, M. Schertler, M. Schneider, and J. Turner, *Internet Security Association and Key Management Protocol (ISAKMP)*, RFC 2408, Nov. 1998.

[10] C. Meadows, "Analysis of the Internet key exchange protocol using the NRL protocol analyzer," in *Proceedings of IEEE Symposium on Security and Privacy*, pp. 216-231, Oakland, CA, May 1999.

[11] R. Perlman and C. Kaufman, "Key exchange in IPSec: Analysis of IKE," *IEEE Internet Computing*, vol. 4, no. 6, pp. 50-56, Nov. 2000.

[12] A. Perrig and D. X. Song, "Looking for diamonds in the desert: Extending automatic protocol generation to three-party authentication and key agreement protocols," in *Proceedings of the 13th IEEE Computer Security Foundations Workshop*, pp. 64-76, Cambridge, England, July 2000.

[13] D. X. Song, "Athena: a new efficient automated checker for security protocol analysis," in *Proceedings of the 12th IEEE Computer Security Foundations Workshop*, pp. 192-202, Mordano, Italy, June 1999.

[14] T. Y. C. Woo and S. S. Lam, "A lesson on authentication protocol design," *Operating Systems Review*, vol. 28, no. 3, pp. 24-37 1994.

[15] J. Zhou, "Fixing a security flaw in IKE protocols," *Electronics Letters*, vol. 35, no. 13, pp. 1072-1073, 1999.

[16] J. Zhou, "Further analysis of the Internet key exchange protocol," *Computer Communications*, vol. 23, pp. 1606-1612, 2000.

Rui Jiang born in 1968, Ph. D.. Now he works in the School of Information Science and Engineering, Southeast University, Nanjing, China. He received his Ph.D. degree in Electronic Engineering from Shanghai Jiao Tong University, China in 2005. His current research interests include computer network security, next generation wireless network security and the broadband network.

Aiqun Hu born in 1964, Ph. D., professor. Now he is the director of Information Security Research Center in the School of Information Science and Engineering, Southeast University, Nanjing, China. He received his Ph. D. degree in Electronic Engineering from Southeast University in 1993. His current research interests include wireless network security, signal processing and wireless multimedia communication.

Jianghua Li born in 1965, Ph. D., professor. Now he is the executive vice dean of the School of Information Security, Shanghai Jiao Tong University, Shanghai, China. He received his Ph. D. degree in Electronic Engineering from Shanghai Jiao Tong University in 1994. His current research interests include information security and broadband multimedia communication.