IPSEC Working Group
INTERNET-DRAFT
draft-ietf-ipsec-isakmp-03.txt, .ps

Douglas Maughan, Mark Schertler
National Security Agency
November 21, 1995

Internet Security Association and Key Management Protocol (ISAKMP)

Abstract

This memo describes a protocol utilizing security concepts necessary for establishing Security Associations (SA) and cryptographic keys in an Internet environment. A Security Association protocol that negotiates, establishes, modifies and deletes Security Associations and their attributes is required for an evolving Internet, where there will be numerous security mechanisms and several options for each security mechanism. The key management protocol must be robust in order to handle public key generation for the Internet community at large and private key requirements for those private networks with that requirement.

The Internet Security Association and Key Management Protocol (ISAKMP) defines the procedures for authenticating a communicating peer, creation and management of Security Associations, key generation techniques, and threat mitigation (e.g. denial of service and replay attacks). All of these are necessary to establish and maintain secure communications (via IP Security Service or any other security protocol) in an Internet environment.

Status of this memo

This document is being submitted to the IETF Internet Protocol Security (IPSEC) Working Group for consideration as a method for the establishment and management of security associations and their appropriate security attributes. Additionally, this document proposes a method for key management to support IPSP and IPv6. Publication of this document does not imply acceptance of the concepts discussed by the IPSEC Working Group. Comments are solicited and should be addressed to the authors and/or the working group mailing list at `ipsec@ans.net`.

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as "working draft" or "work in progress."

To learn the current status of any Internet-Draft, please check the "1id-abstracts.txt" listing contained in the Internet- Drafts Shadow Directories on ds.internic.net (US East Coast), nic.nordu.net (Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific Rim).

Distribution of this document is unlimited.

Contents

1 Introduction **4**
1.1 Authentication . . . 4
1.1.1 Certificate Authorities . . . 5
1.1.2 Entity Naming . . . 5
1.1.3 ISAKMP Requirements . . . 5
1.2 Security Associations and Management . . . 6
1.2.1 Security Associations and Registration . . . 6
1.2.2 ISAKMP Requirements . . . 7
1.3 Public Key Cryptography . . . 7
1.3.1 Key Exchange Properties . . . 7
1.3.2 ISAKMP Requirements . . . 8
1.4 ISAKMP Protection . . . 8
1.4.1 Anti-Clogging (Denial of Service) . . . 8
1.4.2 Connection Hijacking . . . 9
1.4.3 Man-in-the-Middle Attacks . . . 9
1.5 Multicast Communications . . . 9

2 Description of the Protocol **9**
2.1 ISAKMP Header Format . . . 10
2.1.1 General Message Processing . . . 12
2.2 ISAKMP Packet Exchanges . . . 13
2.2.1 Base Exchange . . . 13
2.2.2 Identity Protection Exchange . . . 13
2.2.3 Authentication Only Exchange . . . 14
2.3 ISAKMP Details . . . 14
2.3.1 Security Association Attributes . . . 14
2.3.2 Transport Protocol . . . 16
2.3.3 RESERVED Fields . . . 16
2.3.4 Anti-Clogging Token ("Cookie") Creation . . . 17
2.3.5 SA Flags Field . . . 17

3 Security Association Establishment **17**
3.1 Security Association Initialization . . . 18
3.1.1 SA Initialization Procedures . . . 19
3.2 Authentication and Key Exchange . . . 20
3.2.1 Authentication Payload Format . . . 20
3.2.2 Key Exchange Payload Format . . . 21
3.2.3 Authentication and Key Exchange Procedures . . . 22
3.3 Security Association Negotiation . . . 23
3.3.1 SA Negotiation Procedures . . . 24
3.4 SA Negotiation Conclusion . . . 25
3.4.1 SA Negotiation Conclusion Procedures . . . 26

4 Security Association Modification **27**
4.1 Modification Procedures . . . 27

5 Security Association Deletion **27**
5.1 Deletion Procedures . . . 28

6 Notification Message **29**

6.1 Notification Procedures 29

7 Conclusions **30**

A ISAKMP Scenarios **32**
A.1 Initial ISAKMP Daemon Scenerio 32
A.2 Virtual Private Network Scenario 32

B Security Association Attributes **35**

C Security Association Examples **37**
C.1 ISAKMP SA Definition 37
C.1.1 ISAKMP SA Examples 37
C.2 ESP SA and AH SA Definitions 38
C.2.1 ESP and AH SA Examples 39
C.2.2 Fortezza SA Examples 40

1 Introduction

This document describes an Internet Security Association and Key Management Protocol (ISAKMP). ISAKMP combines the security concepts of authentication, key management, and security associations to establish the required security for government, commercial, and private communications on the Internet. ISAKMP extends the assertion in [DOW92] that authentication and key exchanges must be combined for better security to include security association exchanges. The security required for communications depends on the individual network configurations and environments. Organizations are setting up Virtual Private Networks (VPN) that will require one set of security functions for communications within the VPN and possibly many different security functions for communications outside the VPN to support geographically separate organizational components, customers, suppliers, sub-contractors (with their own VPNs), government, and others. Departments within large organizations may require a number of security associations to separate and protect data (e.g. personnel data, company proprietary data, medical) on internal networks and other security associations to communicate inter-department. Nomadic users wanting to "phone home" represent another set of security requirements. These requirements must be tempered with bandwidth challenges. Smaller groups of people may meet their security requirements by setting up "Webs of Trust". ISAKMP exchanges provide these assorted networking communities the ability to present peers with the security functionality it supports in an authenticated and protected manner for agreement upon a common interoperable security association.

Security associations must support different encryption algorithms, authentication mechanisms, and key establishment algorithms for other security protocols, as well as IP Security. Security associations must also support host-oriented certificates for lower layer protocols and user-oriented certificates for higher level protocols. Algorithm and mechanism independence is required in applications such as e-mail, remote login, and file transfer, as well as in session oriented protocols, routing protocols, and link layer protocols. ISAKMP provides a common security association and key establishment protocol for this wide range of security protocols, applications, security requirements, and network environments.

ISAKMP is not bound to any specific cryptographic algorithm, key generation technique, or security mechanism. This flexibility is beneficial for a number of reasons. First, it supports the dynamic communications environment described above. Second, the independence from specific security mechanisms and algorithms provides a forward migration path to better mechanisms and algorithms. When improved security mechanisms are developed or new attacks against current encryption algorithms, authentication mechanisms and key exchanges are discovered, ISAKMP will allow the updating of the algorithms and mechanisms without having to develop a completely new KMP or patch the current one.

ISAKMP has basic requirements for its authentication and key exchanges components. These requirements guard against denial of service, replay / reflection, man-in-the-middle, and connection hijacking attacks. This is important because these are the types of attacks that are targeted against protocols. Complete Security Association (SA) support, which provides mechanism and algorithm independence, and protection from protocol threats are the strengths of ISAKMP.

1.1 Authentication

A very important step in establishing secure network communications is authentication of the entity at the other end of the communication. Many authentication mechanisms are available. Authentication mechanisms fall into two catagories of strength - weak and strong. Passwords are an example of a mechanism that provides weak authentication. Reasons for this include the fact that most users pick easy to guess passwords and when

used over an unprotected network are easily read by network sniffers. Digital signatures, such as the Digital Signature Standard (DSS) and the Rivest-Shamir-Adleman (RSA) signature, are public key based strong authentication mechanisms. When using digital signatures each entity requires a public and a private key. Certificates are an essential part of a digital signature authentication mechanism. Certificates bind a specific entities identity (be it host, network, user, or application) to its public keys and possibly other security-related information such as privileges, clearances, and compartments. Authentication based on digital signatures requires a trusted third party or certificate authority to create, sign and properly distribute certificates. For more detailed information on digital signatures, such as DSS and RSA, and certificates see [Schn94].

1.1.1 Certificate Authorities

Certificates require an infrastructure for generation, verification, management and distribution. The Internet Policy Registration Authority (IPRA) [RFC-1422] has been established to direct this infrastructure for the IETF. The IPRA certifies Policy Certification Authorities (PCA). PCAs control Certificate Authorities (CA) which certify users and subordinate entities. Current certificate related work includes the Domain Name System (DNS) Security Extensions [EK94] which will provide signed entity keys in the DNS. The Public Key Infrastucture (PKIX) working group is specifying an Internet profile for X.509 certificates. There is also work going on in industry to develop X.500 Directory Services which would provide X.509 certificates to users. The U.S. Post Office is developing a (CA) hierarchy. The NIST Public Key Infrastructure Working Group has also been doing work in this area. The DOD Multi Level Information System Security Initiative (MISSI) program has begun deploying a certificate infrastructure for the U.S. Government. Alternatively, if no infrastructure exists, the PGP Web of Trust certificates can be used to provide user authentication and privacy in a community of users who know and trust each other.

1.1.2 Entity Naming

An entity's name is its identity and is bound to its public keys in certificates. The CA MUST define the naming semantics for the certificates it issues. See the UNINETT PCA Policy Statements [Berg] for an example of how a CA defines its naming policy. When the certificate is verified, the name is verified and that name will have meaning within the realm of that CA. An example is the DNS security extensions which make DNS servers CAs for the zones and nodes they serve. Resource records are provided for public keys and signatures on those keys. The names associatied with the keys are IP addresses and domain names which have meaning to entities accessing the DNS for this information. A Web of Trust is another example. When webs of trust are set up, names are bound with the public keys. In PGP the name is usaully the entities e-mail address which has meaning to those, and only those, who understand e-mail (Do MCI and AOL e-mail addresses tell the casual e-mailer anything about identity?). Another web could use an entirely different naming scheme.

1.1.3 ISAKMP Requirements

Strong authentication MUST be provided on ISAKMP exchanges. Without being able to authenticate the entity at the other end, the Security Association (SA) and session key established are suspect. Without authentication you are unable to trust an entity's identification, this makes access control questionable. Encryption (e.g. ESP) and integrity (e.g. AH) will protect subsequent communications from passive eavesdroppers, but the SA and key may be established with an adversary who performed an active man-in-the-middle attack and is now stealing all your personnal data.

A digital signature algorithm MUST be used within ISAKMP's authentication component. However, ISAKMP does not mandate a specific mechanism. ISAKMP allows an entity initiating communications to indicate which signature algorithms it supports. After selection of a common algorithm, the protocol provides the messages required to support the actual authentication exchange. As an example, if the DSA is selected as the signature algorithm, then the protocol provides a facility for identification of different certificate authorities, certificate types (e.g. X.509v1 certificates, PKCS #7), and the exchange of the certificates identified.

ISAKMP utilizes digital signatures, based on public cryptography, for authentication. There are other strong authentication systems available, which could be specified as additional optional authentication mechanisms for ISAKMP. Some of these authentication systems rely on a trusted third party called a key distribution center (KDC) to distribute secret session keys. An example is Kerberos, where the trusted third party is the Kerberos server, which holds secret keys for all clients and servers within it's network domain. A clients proof it holds it's secret key provides its authenticaton to a server.

The ISAKMP specification does not specify the protocol for communicating with the trusted third parties (TTP) or certificate directory services. These protocols are defined by the TTP and directory service themselves and are outside the scope of this specification.

1.2 Security Associations and Management

A Security Association (SA) is a relationship between two or more entities that describes how the entities will utilize security services to communicate securely. This relationship is represented by a set of information that can be considered a contract between the entities. The information must be agreed upon and shared between all the entities. Sometimes the information alone is referred to as an SA, but this is just a physical instantiation of the existing relationship. The existence of this relationship, represented by the information, is what provides the agreed upon security information needed by entities to securely interoperate. All entities must adhere to the SA for secure communications to be possible. When accessing SA attributes, entities use a pointer or identifier refered to as the Security Parameter Index (SPI).

1.2.1 Security Associations and Registration

The SA attributes required and recommended for the IP Security (AH, ESP) are defined in [RFC-1825]. The attributes specified for an IP Security SA include, but are not limited to, authentication mechanism, cryptographic algorithm, algorithm mode, key length, and Initialization Vector (IV). Other protocols that provide algorithm and mechanism independent security MUST define their SA attributes requirements. The separation of ISAKMP from a specific SA definition is important to ensure ISAKMP can establish SAs for all possible security protocols and applications.

NOTE: See Appendix B for a discussion of SA attributes that should be considered when defining a security protocol or application.

In order to facilitate easy identification of specific attributes (e.g. a specific encryption algorithm) among different network entites the attributes must be assigned identifiers and these identifiers must be registered by a central authority. The Internet Assigned Numbers Authority (IANA) provides this function for the Internet.

1.2.2 ISAKMP Requirements

Security Association (SA) establishment MUST be part of the key management protocol defined for IP based networks. The SA concept is required to support security protocols in a diverse and dynamic networking environment. Just as authentication and key exchange must be linked to provide assurance that the key is established with the authenticated party [DOW92], SA establishment must be linked with the authentication and the key exchange protocol.

ISAKMP provides the protocol exchanges to establish a security association between entities. First, an initial protocol exchange allows a basic set of security attributes to be agreed upon. This basic set provides protection for subsequent ISAKMP exchanges. It also indicates the authentication method and key exchange that will be performed as part of the ISAKMP protocol. If a basic set of security attributes is already in place on the communicating entities the initial ISAKMP exchange may be skipped and the key and authentication exchanges issued directly. After the basic set of security attributes has been agreed upon, initial identity authenticated, and required keys generated, another security attribute exchange takes place to establish the complete SA which will be used for subsequent communications by the entity that invoked ISAKMP. The basic set of SA attributes that MUST be implemented to provide ISAKMP interoperability are defined in Appendix C. *These atributes will be moved to a separate document to maintain separation of protocol and attributes.*

1.3 Public Key Cryptography

Public key cryptography is the most flexible, scalable, and efficient way for users to obtain the shared secrets and session keys needed to support the large number of ways Internet users will interoperate. Many key generation algorithms, that have different properties, are available to users (see [DOW92] and [ANSI94]). Properties of key exchange protocols include the key establishment method, authentication, symmetry, perfect forward secrecy, and back traffic protection.

1.3.1 Key Exchange Properties

Key Establishment (Key Generation / Key Transport) The two common methods of using public key cryptography for key establishment are key transport and key generation. An example of key transport is the use of the RSA algorithm to encrypt a randomly generated session key (for encrypting subsequent communications) with the recipient's public key. The encrypted random key is then sent to the recipient, who decrypts it using his private key. At this point both sides have the same session key, however it was created based on input from only one side of the communications. The benefit of the key transport method is that it has less computational overhead then the following method. The Diffie-Hellman (D-H) algorithm illustrates key generation using public key cryptography. The D-H algorithm is begun by two users exchanging public information. Each user then mathematically combines the other's public information along with their own secret information to compute a shared secret value. This secret value can be used as a session key or as a key encryption key for encrypting a randomly generated session key. This method generates a session key based on public and secret information held by both users. The benefit of the D-H algorithm is that the key used for encrypting messages is based on information held by both users. Assuming checks for weak values neither party can force the session key to a predetermined value. Detailed descriptions of these algorithms can be found in [Schn94]. There are a number of variations on these two key generation schemes and these variations do not necessarily interoperate.

Key Exchange Authentication Key exchanges may be authenticated during the protocol or after protocol completion. Authentication of the key exchange during the protocol is provide when each party provides proof it has the secret session key before the end of the protocol. Proof can be provided by encrypting known data in the secret session key during the protocol exchange. Authentication after the protocol must occur in subsequent communications. Authentication during the protocol is preferred so subsequent communications are not initiated if the secret session key is not established with the desired party.

Key Exchange Symmetry A key exchange provides symmetry if either party can initiate the exchange and exchanged messages can cross in transit without effecting the key that is generated. This is desirable so that computation of the keys does not require either party to know who initiated the exchange. While key exchange symmetry is desirable, symmetry in the entire KMP may provide a vulnerablity to reflection attacks. The entire ISAKMP SA establishment is asymetrical.

Back Traffic Protection / Perfect Forward Secrecy Perfect forward secrecy is provided by a key exchange protocol if disclosure of long-term cryptographic keying material (e.g. public signature keys) does not compromise previously generated keys. Back traffic protection is provided by the independent generation of each key such that subsequent keys are not dependent on any previous key. There is a subtle difference. Past session keys will NOT be obtainable is the long-term key is compromised in perfect forward secrecy; Past session keys will NOT be obtainable if the current session key is compromised in back traffic protecion.

The difficulty of numerical factoring of large numbers has proven that cryptographic keys can protect information for a considerable length of time. However, this is based on the assumption that keys used for protection of communications are destroyed after use and not kept for any reason.

1.3.2 ISAKMP Requirements

An authenticate key exchange MUST be supported by ISAKMP. Users SHOULD choose additional key establishment algorithms based on their requirements. ISAKMP does not specify a specific key exchange. Requirements that should be evaluated when choosing a key establishment algorithm include establishment method (generation vs. transport), perfect forward secrecy, back traffic protection, computational overhead, key escrow, and key strength. Based on user requirements, ISAKMP allows an entity initiating communications to indicate which key exchanges it supports. After selection of a key exchange, the protocol provides the messages required to support the actual key establishment.

1.4 ISAKMP Protection

1.4.1 Anti-Clogging (Denial of Service)

Of the numerous security services available, protection against denial of service always seems to be one of the most difficult to address. Phil Karn in his Internet-Draft [Karn95] has introduced a mechanism to provide a measure of denial of service protection through the use of a "cookie" exchange. This anti-clogging token (ACT) is aimed at protecting the computing resources from attack without spending excessive CPU resources to determine its authenticity. As described in [Karn95], an exchange prior to CPU-intensive public key operations can thwart some denial of service attempts (e.g. simple flooding with bogus IP source addresses).

As noted by Karn, absolute protection against denial of service is impossible, but this anti-clogging token provides a technique for making it easier to handle.

1.4.2 Connection Hijacking

ISAKMP prevents connection hijacking by linking the authentication, key exchange and security association exchanges. This linking prevent an attacker from allowing the authentication to complete and then jumping in and impersonating one entity to the other during the key and security association exchanges.

1.4.3 Man-in-the-Middle Attacks

Man-in-the-Middle attacks include interception, insertion, deletion, and modification of messages, reflecting messages back at the sender, replaying old messages and redirecting messages. ISAKMP features prevent these types of attacks from being successful. The linking of the ISAKMP exchanges prevents the insertion of messages in the protocol exchange. The ISAKMP protocol state machine is defined so deleted messages will not cause a partial SA to be created, the state machine will clear all state and return to idle. The state machine also prevents reflection of a message from causing harm. The requirement for a new cookie with time variant material for each new SA establishment prevents attacks that involve replaying old messages. The ISAKMP strong authentication requirement prevents an SA from being established with other then the intended party. Messages may be redirected to a different destination or modified but this will be detected and an SA will not be established. The ISAKMP specification defines where abnormal processing has occurred and recommends notifying the appropriate party of this abnormality.

1.5 Multicast Communications

While future Internet communications will increasingly be of a multicast nature, this document is presenting a security association and key management protocol from the unicast point of view. It is expected that multicast communications will require the same security services as unicast communications and may introduce the need for additional security services. The issues of distributing SPIs for multicast traffic are presented in [RFC-1825]. Upon agreement and implementation of a security association protocol for the Internet unicast environment, we fully intend to examine any additional security requirements for multicast communications. For an introduction to the issues related to multicast security consult the Internet Drafts, [Spar94a] and [Spar94b], describing Sparta's research in this area.

2 Description of the Protocol

The Internet Security Association and Key Management Protocol (ISAKMP) defines procedures and packet formats to establish, negotiate, modify and delete Security Associations (SA). SAs contain all the information required for execution of IP security services, such as the IP Authentication Header (AH), the IP Encapsulating Security Payload (ESP), and routing protocol authentication mechanisms. ISAKMP includes packet formats for exchanging key generation and authentication data. These formats provide a consistent method of transferring key and authentication data that is independent of the key generation technique, encryption algorithm or authentication mechanism.

The following sections contain the details of ISAKMP. Sections 2.1 through 2.3 cover details that are pertinent to the entire protocol. Sections 3 through 6 define the establishment, modification, and deletion services, defined as exchanges, offered by the protocol. The appendices provide examples of SAs and key exchanges.

2.1 ISAKMP Header Format

ISAKMP has a fixed header format (shown in Figure 1) followed by a variable length payload, optional digital signature, and optional padding. A fixed header simplifies parsing, providing the benefit of protocol parsing software that is less complex and easier to implement. The fixed header contains the information required by the protocol to maintain state, process payloads and prevent attacks (e.g. denial of service and replay). Based on the message type, each header is followed by a payload specific to the message type. The payload for each message is defined in sections 3 through 6. Following the payload portion of the ISAKMP packet is a digital signature. This field is dependent on the negotiation of Security Association attributes and may not be present.

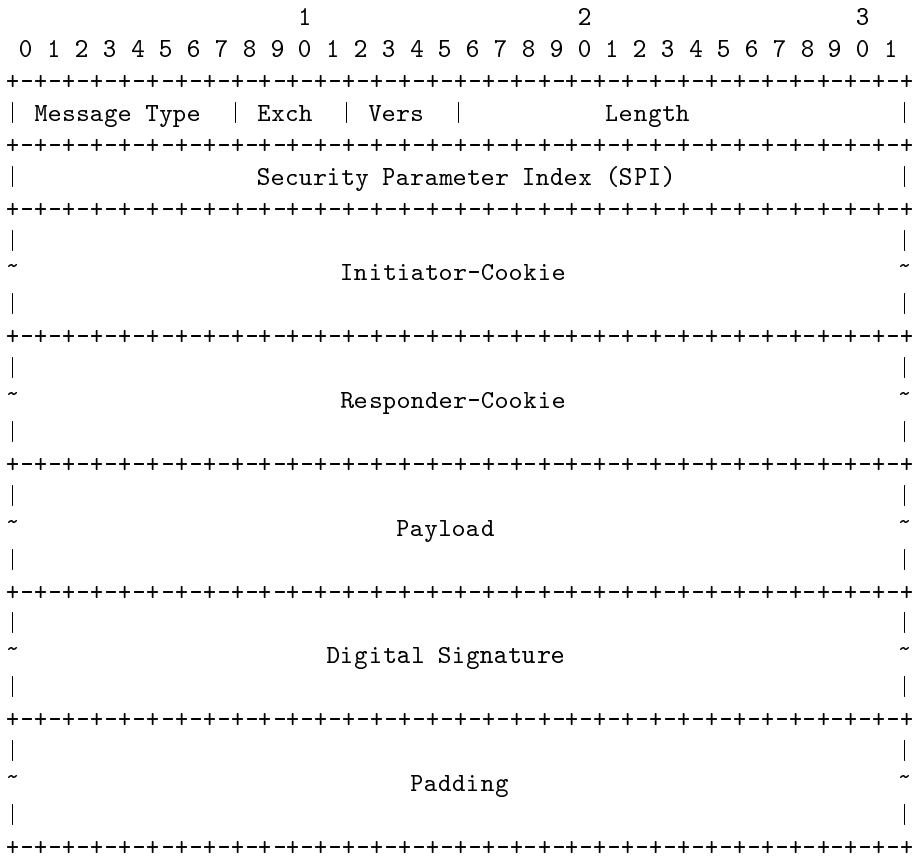


Figure 1: ISAKMP Header Format

- Message Type (1 octet) - Indicates the type of message. A suffix of REQ denotes a Request message type and an RESP suffix denotes a *Response* message type. The format and processing for each message is defined in sections 3 through 6.

ISAKMP Message	Message Type
RESERVED	0
ISA_INIT_REQ	1
ISA_INIT_RESP	2
ISA_KE_REQ	3
ISA_KE_RESP	4
ISA_AUTH_REQ	5
ISA_AUTH_RESP	6
ISA_AUTH&KE_REQ	7
ISA_AUTH&KE_RESP	8
ISA_NEG_REQ	9
ISA_NEG_RESP	10
ISA_MODIFY_REQ	11
ISA_MODIFY_RESP	12
ISA_NOTIFY	13
ISA_DELETE	14
ISA_COMMIT	15
IANA Use	16-127
Future Use	128-255

- Exchange (4 bits) - indicates the type of exchange, see section 2.2 for a description of the Message Types exchanged in each of these Exchange Types.

ISAKMP Exchange	Exchange Type
RESERVED	0
Base	1
Identity Protection	2
Authentication Only	3
Future Use	4 - 15

- Version (4 bits) - indicates the version of the ISAKMP protocol in use.
- Length (2 octets) - Length of total message (header + payload) in octets.
- SPI (4 octets) - Security Parameter Index. The receiving entity's SPI is always in this field, except for the ISA_INIT packets. The ISA_INIT packets contain the SPI the initiator expects to receive in all subsequent packets.
- Initiator Cookie (16 octets) - Cookie of entity that initiated SA establishment, SA modify or SA delete.
- Responder Cookie (16 octets) - Cookie of entity that is responding to an SA establishment, SA modify or SA delete request.
- Payload (variable) - The format of the payload is based on the message type. These are defined in sections 3 through 6.
- Signature - The digital signature of the initiator of the ISAKMP message. This field will not be included on all packets and will be determined by the negotiated SA attributes.
- Padding - This is an optional field that may be added depending on the type of encryption algorithm. If the encryption mechanism is based on block encryption, then this field may be necessary to ensure the packet is a specific size.

2.1.1 General Message Processing

Every ISAKMP message has basic processing applied to insure protocol reliability, and to minimize threats, such as denial of service and replay attacks.

When transmitting an ISAKMP packet, the transmitting entity (initiator or responder) does the following:

1. Sets a timer and initializes a retry counter.
2. If the timer expires, the ISAKMP packet is resent and the retry counter is decremented.
3. If the retry counter reaches zero (0), the event, *RETRY LIMIT REACHED*, is logged in the appropriate system audit file.
4. The ISAKMP protocol machine clears all states and returns to IDLE.

When an ISAKMP packet is received, the receiving entity (initiator or responder) does the following:

1. Verifies the Initiator and Responder "cookies". If the cookie validation fails, the message is discarded and the following actions are taken:
   (a) The event, *INVALID COOKIE*, is logged in the appropriate system audit file.
   (b) No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
2. Check the Message Type field to confirm it is valid. If the Message Type field validation fails, the message is discarded and the following actions are taken:
   (a) The event, *INVALID MESSAGE TYPE*, is logged in the appropriate system audit file.
   (b) No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
3. Check the Exchange field to confirm it is valid for the Message Type requested. If the Exchange field validation fails, the message is discarded and the following actions are taken:
   (a) The event, *INVALID EXCHANGE TYPE*, is logged in the appropriate system audit file.
   (b) No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
4. Check SPI to ensure it is valid for the Message Type and Exchange being performed. If the SPI validation fails, the message is discarded and the following actions are taken:
   (a) The event, *INVALID SPI*, is logged in the appropriate system audit file.
   (b) No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
5. The message payload is processed. Individual message processing is described in sections 3 through 6. Depending on the Message Type, a valid message results in a response being sent to the transmitting entity (message originator). The procedures for sending these responses are also outline in sections 3 through 6.

2.2 ISAKMP Packet Exchanges

The Exchange field in the ISAKMP header currently has three values defined: the base exchange, the identity protection exchange, and the authentication only exchange. These exchanges define the flow of the ISAKMP packets during SA establishment. The diagrams in 2.2.1, 2.2.2, and 2.2.3 show the packet exchange ordering for each exchange type and provide basic notes describing what has happened after each packet exchange.

2.2.1 Base Exchange

Sections 3.1 through 3.3 describe the three basic phases: SA Initialization, Key Exchange and Authentication, and SA Negotiation, that comprise the base exchange. The base exchange contains the minimum number of packet exchanges in order to reduce latency associated with SA establishment.

	Initiator	Base Exchange Direction	Responder	Note
	ISA_INIT_REQ	=>		
		<=	ISA_INIT_RESP	
				Basic SA selected
	ISA_AUTH&KE_REQ	=>		
		<=	ISA_AUTH&KE_RESP	
				Identity Verified Key Generated Encryption Begun
	ISA_NEG_REQ	=>		
		<=	ISA_NEG_RESP	SA Completed
(optional)	ISA_COMMIT	=>		

2.2.2 Identity Protection Exchange

The identity protection exchange starts and ends the same as the base exchange, but separates the key exchange payload and the authentication payloads into separate packets. In this exchange, the key exchange is transmitted first followed by the strong authentication exchange. The benefit of this exchange is the ability to communicate with a person without disclosing either party's identity to passive attackers on the network.

The ISA_KE_REQ and ISA_KE_RESP packets used for the key exchange portion of this exchange contain an ISAKMP header followed by the key exchange payload. The ISA_AUTH_REQ and ISA_AUTH_RESP packet used for the authentication portion of this exchange contain an ISAKMP header followed by the authentication payload.

Identity Protection Exchange

	Initiator	Direction	Responder	Note
	ISA_INIT_REQ	=>		
		<=	ISA_INIT_RESP	
				Basic SA selected
	ISA_KE_REQ	=>		
		<=	ISA_KE_RESP	
				Key Generated
				Encryption Begun
	ISA_AUTH_REQ	=>		
		<=	ISA_AUTH_RESP	
				Identity Verified
	ISA_NEG_REQ	=>		
		<=	ISA_NEG_RESP	SA Completed
(optional)	ISA_COMMIT	=>		

2.2.3 Authentication Only Exchange

The authentication only exchange starts and ends the same as the base exchange. In this exchange, the authentication information is the only information transmitted. The benefit of this exchange is the ability to perform only an authentication exchange without the computational expense of computing keys. Using this exchange, none of the transmitted information will be encrypted.

The ISA_AUTH_REQ and ISA_AUTH_RESP packet used for the authentication only exchange contain an ISAKMP header followed by the authentication payload.

Identity Protection Exchange

	Initiator	Direction	Responder	Note
	ISA_INIT_REQ	=>		
		<=	ISA_INIT_RESP	
				Basic SA selected
	ISA_AUTH_REQ	=>		
		<=	ISA_AUTH_RESP	
				Identity Verified
	ISA_NEG_REQ	=>		
		<=	ISA_NEG_RESP	SA Completed
(optional)	ISA_COMMIT	=>		

2.3 ISAKMP Details

2.3.1 Security Association Attributes

A Security Association (SA) is a relationship between two entities that describes how they will utilize security services. This relationship is represented by a collection of security related information. The SA Attributes are the individual elements that comprise all security relevant information necessary to form the SA.

The following syntax defines the security attributes to be exchanged by ISAKMP. This syntax is used in the

ISA_INIT_REQ, ISA_INIT_RESP, ISA_NEG_REQ, ISA_NEG_RESP, ISA_MOD_REQ, and ISA_MOD_RESP messages. The syntax groups security attributes needed to perform a security function into either an SA set or SA list format. The set format MUST be supported by ISAKMP implementations. The list format is an optional format.

Security Associations Sets The set format groups all necessary attributes together. Each set has a unique identifier (Set Number), supported security service (Supports), such as IP AH, IP ESP, OSPF authentication, and a list of Attribute Classes and Attribute Types. The Attribute Class is the broad category of Attribute Type, such as encryption algorithms. Attribute Type is a specific attribute identifier. DES is an example of an attribute type for the encryption algorithm attribute class. A set has only one instance of an Attribute Class and one type for that class. This syntax maintains flexibility by allowing many different (and some still undefined) types of SA attributes to be exchanged.

Figure 2 depicts the syntax for exchanging security attributes using the set format. It shows a single set from which multiple sets would be grouped for a specific message type.

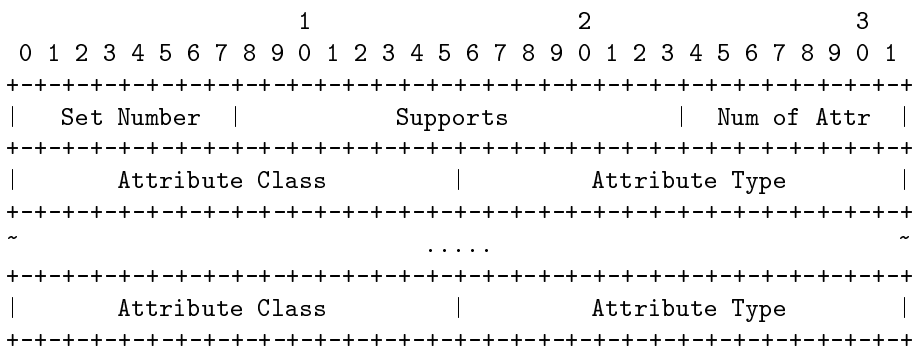


Figure 2: Generic Set Exchange Format

- Set number (1 octet) - Unique identifier for each proposed set
- Supports (2 octets) - Security service proposed set supports. Examples are IP AH, IP ESP, and OSPF authentication
- Number of Attributes (1 octet) - Number of attribute classes contained in the proposed set
- Attribute Class (2 octets) - examples are Encryption Algorithms, Key Exchange Algorithms, Authentication Mechanisms
- Attribute Type (2 octets) - examples of attribute types for the encryption algorithms attribute class are DES, Triple DES, and IDEA.

The size of a set formatted exchange is 4 octets + (Number of Attribute Classes * 4 octets). Computing the size of a particular set allows the determination of the beginning of the next set without completely parsing the current set. This is necessary when it is determined that the current set is not an acceptable SA set. This will improve the performance of SA Attribute determination.

Security Association Lists The SA list format presents several attribute types for an attribute class. Each type within the class is then ordered to indicate its precedence. Specifically, the first attribute type is the highest priority type, followed by other choices. Each subsequent choice is listed in descending priority order. An attribute type must be chosen for each attribute class to establish a complete SA.

Figure 3 shows the syntax for the optional list exchange format. The number of types is determined by the Count field. The number of Attribute Types within an Attribute Class will depend on what is supported by the local machine.

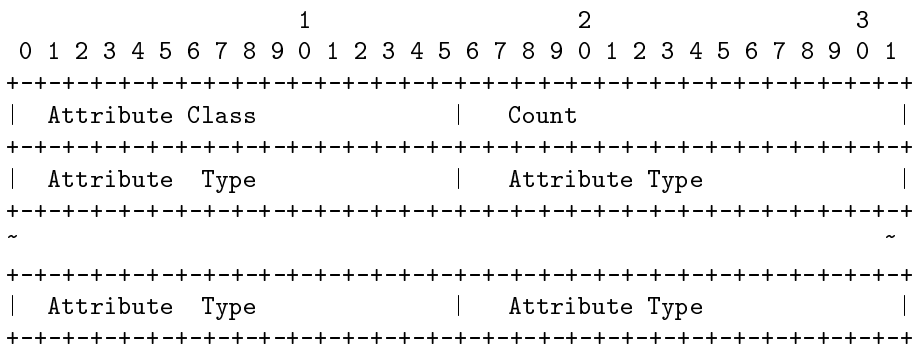


Figure 3: Generic List Exchange Format

- Attribute Class (2 octets) - Examples are Encryption Algorithms, Key Exchange Algorithms
- Count - Number of proposed Attribute Types for the given Attribute Class
- Attribute Type (2 octets) - Presented in descending priority order

Appendix B presents an outline containing a comprehensive listing of SA attributes. This listing of attributes are initial definitions and are presented to stimulate thought and discussion on SAs. The final SA for a protocol SHOULD be defined in that protocol so additions or modifications to the attributes do not require a modification to the Internet Key Management Protocol (IKMP) RFC and vice versa. For example, Appendix C describes the sample security associations for ISAKMP and IPSP ESP and AH.

2.3.2 Transport Protocol

The User Datagram Protocol (UDP) is the transport protocol for ISAKMP. UDP checksumming discards UDP packets with an incorrect or zero (0) checksum. ISAKMP is unaware of the discard, but will resend the packet when its resend timer expires.

2.3.3 RESERVED Fields

The existence of RESERVED fields are strictly used to preserve byte alignement. All RESERVED fields in the ISAKMP protocol MUST be set to zero (0) when a packet is issued. The receiver SHOULD check the RESERVED fields for zero (0) and discard the packet if other values are found.

2.3.4 Anti-Clogging Token ("Cookie") Creation

Phil Karn's Internet Draft [Karn95] states that cookie generation is implementation dependent, but must satisfy some basic requirements:

```
1.    The cookie must depend on the specific parties.  This prevents
      an attacker from obtaining a cookie using a real IP address and
      UDP port, and then using it to swamp the victim with Diffie-
      Hellman requests from randomly chosen IP addresses or ports.

2.    It must not be possible for anyone other than the issuing
      entity to generate cookies that will be accepted by that
      entity.  This implies that the issuing entity must use local
      secret information in the generation and subsequent
      verification of a cookie.  It must not be possible to deduce
      this secret information from any particular cookie.

3.    The cookie generation function must be fast to thwart attacks
      intended to sabotage CPU resources.
```

Karn's suggested method for creating the cookie is to perform a fast hash (e.g. MD5) over the IP Source and Destination Address, the UDP Source and Destination Ports and a locally generated secret random value. ISAKMP requires that the cookie be unique for each SA establishment, SA modify and SA delete to help prevent replay attacks, therefore the date and time MUST be added to the information hashed.

2.3.5 SA Flags Field

The SA Flags field may be set by the entity that initiated the negotiation to indicate that the ISA_COMMIT packet will follow the completion of the protocol exchange. The SA Flags field exists only in the ISA_INIT and ISA_NEG packets. If the initiating entity sets the flag, the responding entity cannot reset it. If the initiating entity does not set the flag, the responding entity may set it, thereby, forcing the initiating entity to issue an ISA_COMMIT packet. If neither entity sets the flag, then the ISA_COMMIT packet will not be issued. To set the flag the Least Significant Bit (LSB) in the SA Flags field is set to one (1) . All other bits in the SA Flags field are zero (0).

3 Security Association Establishment

Security Association (SA) Establishment is the process of agreeing upon and exchanging all the security information that is required in an SA. The following sections, 3.1 to 3.3, describe the three basic phases that comprise SA Establishment: SA Initialization, Key and Authentication information exchange, and SA Negotiation.

3.1 Security Association Initialization

The initialization exchange of SA establishment is composed of the ISA_INIT_REQ and ISA_INIT_RESP packets shown in figure 4. The ISA_INIT packets exchange "cookies", and options for a key generation technique, an encryption algorithm and an authentication mechanism. The "cookies" are used to prevent replay and denial of service attacks. Authentication and encryption for the ISAKMP exchanges are provided by the authentication mechanism and encryption algorithm selected. The key generation technique selected creates keys for use by the authentication mechanism and encryption algorithm. The keys may also be used as any of the following: actual session keys, to create the session keys, or to protect the exchange of the actual session keys for the SA. If the key, encryption algorithm, and authentication mechanism are only used to protect ISAKMP exchanges, then new options can be picked during the negotiation phase (described in Section 3.3) for use in protecting the actual data communications. If encryption is not required for the SA, the encryption algorithm options are not exchanged.

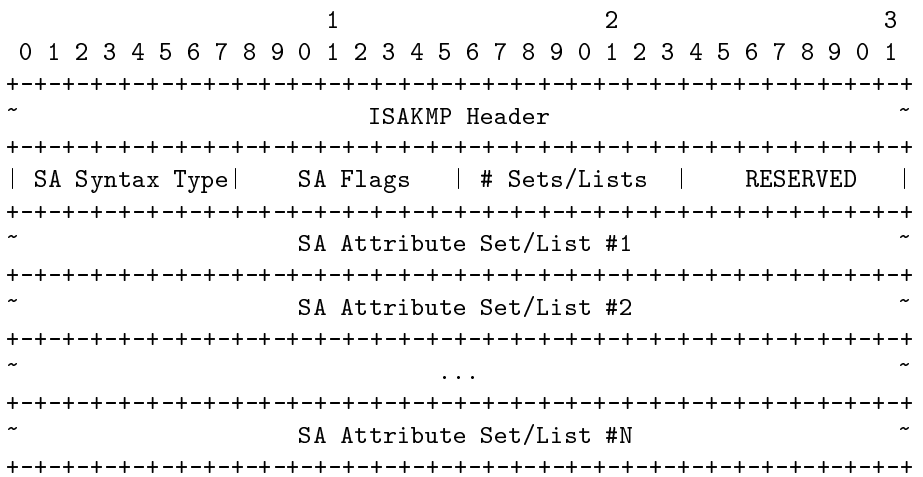


Figure 4: ISA_INIT_REQ and ISA_INIT_RESP Packet Format

- ISAKMP Header - Described in Section 2.1
- SA Syntax Type (1 octet) - Presentation format of SAs

SA Syntax	SA Syntax Type
RESERVED	0
Set	1
List	2

- SA Flags (1 octet) - Flags specific to an SA service. See section 2.3.5 for details.
- Number of Sets (1 octet) - Number of SA Attribute Sets being proposed
- SA Attributes (variable) - A list of SA Attributes. The SA Attribute specifications are discussed in Section 2.3.1.

3.1.1 SA Initialization Procedures

When issuing an ISA_INIT_REQ message, the initiating entity does the following:

1. Create initiator cookie. See Section 2.3.4 for details.
2. Generate a unique pseudo-random SPI. See Section 2.1 for details.
3. Construct an ISA_INIT_REQ packet. If the initiator will send an ISA_COMMIT message upon completion of the SA establishment, then the SA Flags field MUST be set (see section 2.3.5 and 3.4).
4. Transmit the packet to the destination host as described in Section 2.1.1.

When an ISA_INIT_REQ message is received, the receiving entity does the following:

1. Check the ISAKMP header as described in Section 2.1.1.
2. Unpack the ISA_INIT_REQ payload and determine the highest priority attribute set (or attribute list) supported. If the proposed attribute set (or list) is rejected, then the protocol machine must clear all state and return to IDLE.
3. Create responder cookie. See Section 2.3.4 for details.
4. Generate a unique pseudo-random SPI. See Section 2.1 for details.
5. Construct an ISA_INIT_RESP packet. If the responder wants to request that an ISA_COMMIT message be sent upon completion of the SA establishment, then the SA Flags field MUST be set (see section 2.3.5 and 3.4).
6. Transmit the packet to the initiating host as described in Section 2.1.1.

When an ISA_INIT_RESP message is received, the receiving entity (original initiator) does the following:

1. Check the ISAKMP header as described in Section 2.1.1.
2. Unpack the ISA_INIT_RESP payload.
3. Determine if the attribute set (or list) selected by the responder is valid. If the attribute set (or list) is invalid or the responder rejected all proposed attribute sets (or lists), the receiving entity does the following:

   (a) The event, *INVALID ATTRIBUTES*, is logged in the appropriate system audit file.
   (b) Clear all state and return to IDLE. Any further communication must start the SA initialization procedures from the beginning.

   If the attribute set (or list) is valid, the receiving entity does the following:

   (a) Configure protocol machine based on attribute set selected.
   (b) Transition to Authentication and Key Exchange (see Section 3.2).

3.2 Authentication and Key Exchange

During the authentication and key exchange phase, information required to confirm the identities of the parties wishing to establish the SA and establish a session key for use during the SA establishment is exchanged. Depending on the key exchange algorithm, the original key may be used during data communications or a new one may be created and exchanged during the negotiation phase (described in section 3.3). This original or new key would be used in protecting the actual data communications.

The packets that carry the authentication and key exchange payloads have the format shown in Figure 5. When the ISA_AUTH&KE_REQ and ISA_AUTH&KE_RESP packets are used, the Authentication Payload SHOULD be processed first to strongly authenticate the packet issuer, followed by the processing of the Key Exchange Payload. The authentication and key exchange payloads (shown in Figures 6 and 7) are general formats which support many types of authentication and key exchange mechanisms. The detailed specification of these fields will be specified in companion RFCs. These companion RFCs will define the standard authentication and key exchange mechanisms that need to be implemented to assure compliance with this specification.

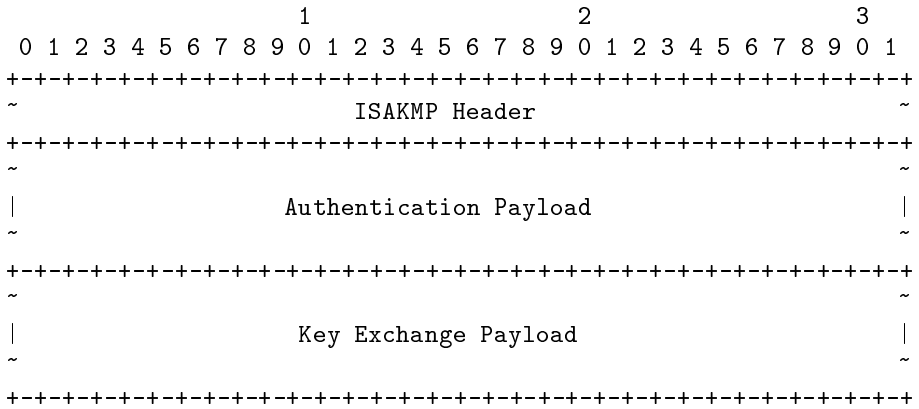


Figure 5: ISA_AUTH&KE_REQ and ISA_AUTH&KE_RESP Packet Format

3.2.1 Authentication Payload Format

This section specifies the encoding of the authentication payload for the ISA_AUTH_REQ, ISA_AUTH_RESP, ISA_AUTH&KE_REQ, and ISA_AUTH&KE_RESP messages. As described in section 2.2.3, when the ISA_AUTH_REQ and ISA_AUTH_RESP packets are transmitted alone, the key exchange payload is not present. The format of these messages is shown in Figure 6.

- Authentication Authority (2 octets) - This field identifies the party that generated the certificates used for authentication. Authorities must be assigned an identifier by the Internet Assigned Numbers Authority (IANA). Before being assigned an identifier, an authority must publish an RFC defining the authority's domain. [RFC-1422] describes the Internet Policy Registration Authority (IPRA) and the procedures for achieving this registration.

  If PGP certificates, based on the "web of trust", are carried in the authentication payload the Authentication Authority value is one (1).

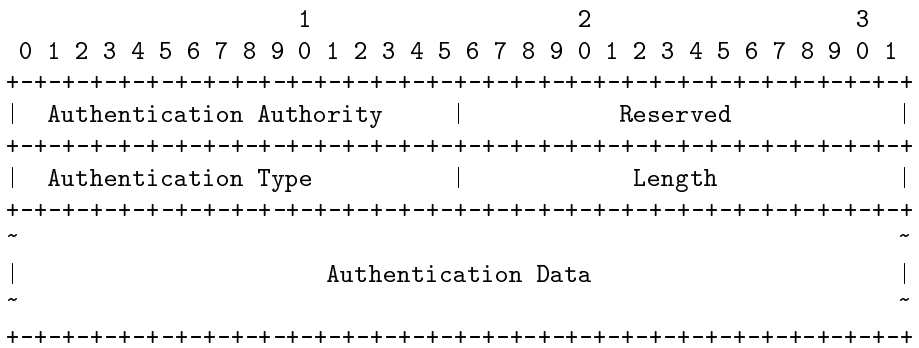


Figure 6: Authentication Payload Format

Example certificate authorities that would have to register for an identifier are:

- RSA Commercial Certificate Authority (http://www_csc.rsa.com/netsite)
- Stable Large E-mail Database (SLED) (http://www.four11.com)
- U.S. Postal Service.

- Authentication Type (2 octets) - This field indicates the authentication payload format. This field is used by authentication authorities that support more than one certificate type. The authentication types supported by an authentication authority must be defined in the RFC required for authentication authority registration. Examples are:
  - RSA certificates
  - PGP certificates
  - DSS certificates
  - DNS Signed Keys
  - Kerberos Tokens
  - X.509 certificates
- Length (2 octets) - Length of the Authentication Data field in octets.
- Authentication Data (variable) - Actual authentication data. The type of certificate is indicated by the Authentication Type field.

3.2.2 Key Exchange Payload Format

A variety of key exchanges can be supported in the key exchange phase. Some examples of key exchanges which may be used in this protocol are Diffie-Hellman, the enhanced Diffie-Hellman key exchange described in X9.42 [ANSI94], the key exchange on the FORTEZZA card, and the RSA-based key exchange used by PGP. This protocol will also support key exchanges that include key escrow or data recovery techniques, but does not mandate their use.

The encoding of the key exchange payload is dependent on the specific key exchange and, therefore, is not specified in this Internet draft. Each key exchange must define the following information: (a) System

parameters, (b) Key establishment algorithm, and (c) Key derivation procedure (dependent on key exchange type).

There can be both public and private key generation techniques. Both types must register with IANA to obtain a Key Exchange Identifier (KEI). Before published public key exchanges can obtain a KEI, an RFC defining the key exchange payload format and key generation procedures MUST be submitted. Private key exchanges SHOULD be documented in an RFC when registering for a KEI.

As described in section 2.2.2, when the ISA_KE_REQ and ISA_KE_RESP packets are transmitted alone, the authentication payload is not present. Once the key exchange is completed, then the authentication payload is sent separately using the format described in section 3.2.1

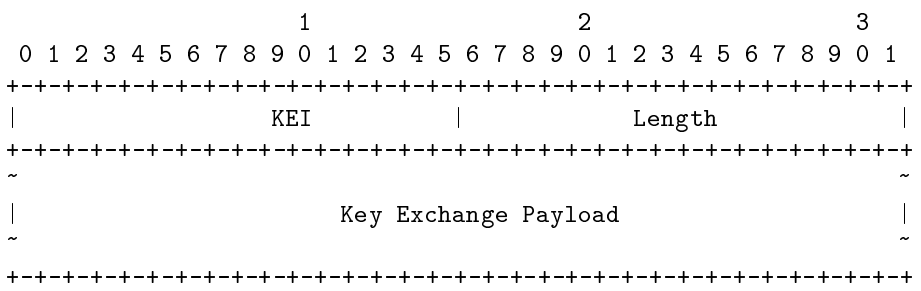


Figure 7: Key Exchange Payload Format

- KEI (2 octets) - Key Exchange Identifier
- Length (2 octets) - Length of payload in octets
- Key Exchange Payload (variable) - Data (i.e. public values) required to create session key.

3.2.3 Authentication and Key Exchange Procedures

When issuing an ISA_AUTH&KE_REQ packet, the initiating entity will do the following:

1. Create the ISAKMP Header.
2. Create the authentication payload.
3. Create the key exchange payload based on KEI.
4. Construct an ISA_AUTH&KE_REQ packet.
5. Generate an authentication signature using the authentication attributes and options selected in the initialization phase.
6. Transmit the packet to the responding host as described in Section 2.1.1.

When an ISA_AUTH&KE_REQ packet is received, the receiving entity will do the following:

1. Check the ISAKMP header as described in Section 2.1.1.

2. Verify the initiator's signature. The ISA_AUTH&KE_REQ packet is processed and the calculated signature is compared to the signature contained in the ISA_AUTH&KE_REQ packet. If these signatures are not identical, the message is discarded and the following actions are taken:
   (a) The event, *INVALID SIGNATURE*, is logged in the appropriate system audit file.
   (b) No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
3. Unpack the ISA_AUTH&KE_REQ packet.
4. Create the ISAKMP Header.
5. Create the authentication payload.
6. Create the key exchange payload based on KEI.
7. Construct an ISA_AUTH&KE_RESP packet.
8. Generate an authentication signature, to authenticate responder to initiator, using the authentication attributes and options selected.
9. Transmit the packet to the initiating host as described in Section 2.1.1.
10. Begin key calculation in the background, if necessary.

When an ISA_AUTH&KE_RESP message is received, the receiving entity (original initiator) will do the following:

1. Check the ISAKMP header as described in Section 2.1.1.
2. Verify the initiator's signature. The ISA_AUTH&KE_RESP packet is processed and the calculated signature is compared to the signature contained in the ISA_AUTH&KE_RESP packet. If these signatures are not identical, the message is discarded and the following actions are taken:
   (a) The event, *INVALID SIGNATURE*, is logged in the appropriate system audit file.
   (b) No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
3. Calculate key, if necessary.
4. Transition to Security Association Negotiation.

3.3 Security Association Negotiation

The SA Negotiation phase allows the initiating entity to present SA attributes that it wishes to use for secure communications to a responding entity. These SA attributes may include additional options for the attributes agreed upon during the initialization phase, as well as additional attributes required for an SA. As an example, the SA parameters for the IP AH and IP ESP security mechanisms are cited in the Security Architecture for the Internet Protocol [RFC-1825]. The format for the ISA_NEG_REQ and ISA_NEG_RESP packets is the same as the ISA_INIT_REQ and ISA_INIT_RESP shown in Figure 4. All fields shown in Figure 4 exist for the ISA_NEG_REQ and ISA_NEG_RESP packets.

3.3.1 SA Negotiation Procedures

When issuing an ISA_NEG_REQ packet, the initiating entity does the following:

1. Determine SA attributes to be negotiated. This may include changing some attributes from the original SA initialization.
2. Construct an ISA_NEG_REQ packet. If the initiator will send an ISA_COMMIT message upon completion of the SA establishment, then the SA Flags field MUST be set (see section 2.3.5 and 3.4).
3. Depending on the SA Attributes established in the SA initialization phase, apply the agreed upon security services.
   (a) If the SA requires authentication, the ISA_NEG_REQ packet is processed (or signed) and the signature placed as noted in Figure 1.
   (b) If the SA requires encryption and the encryption algorithm is a block encryption algorithm, then padding up to the block size MUST be placed as noted in Figure 1.
   (c) If the SA requires encryption, the ISA_NEG_REQ payload and Signature are encrypted.
4. Transmit the packet to the responding host as described in Section 2.1.1.

When an ISA_NEG_REQ packet is received, the receiving entity does the following:

1. Check the ISAKMP header as described in Section 2.1.1.
2. Depending on the SA Attributes, apply the agreed upon security services.
   (a) If the SA requires encryption, decrypt the ISA_NEG_REQ payload and Signature. If the decryption fails, the message is discarded and the following actions are taken:
      i. The event, *DECRYPTION FAILED*, is logged in the appropriate system audit file.
      ii. No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
   (b) If the SA requires authentication, the ISA_NEG_REQ packet is processed and the calculated signature is compared to the signature contained in the ISA_NEG_REQ packet. If these signatures are not identical, the message is discarded and the following actions are taken:
      i. The event, *INVALID SIGNATURE*, is logged in the appropriate system audit file.
      ii. No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
3. Unpack the ISA_NEG_REQ packet payload and determine the highest priority SA attributes supported. If none of the SA attribute options are supported, the ISA_NEG_RESP packet will have the value zero (0) in the Number of Sets field and an SA will not be established.
4. If the SA negotiation is requesting a key change or new authentication mechanism, then generate the appropriate information and include it as an attribute in the ISA_NEG_RESP payload.
5. Construct an ISA_NEG_RESP packet. If the responder wants to request that an ISA_COMMIT message be sent upon completion of the SA establishment, then the SA Flags field MUST be set (see section 2.3.5 and 3.4).
6. Depending on the SA Attributes, apply the agreed upon security services.
   (a) If the SA requires authentication, the ISA_NEG_RESP packet is processed and the signature placed as noted in Figure 1.

(b) If the SA requires encryption and the encryption algorithm is a block encryption algorithm, then padding up to the block size MUST be placed as noted in Figure 1.
(c) If the SA requires encryption, the ISA_NEG_RESP payload and Signature are encrypted.

7. Transmit the packet to the initiating host as described in Section 2.1.1.
8. If required, begin calculation of the new session key in the background.
9. Transition to SA Negotation Conclusion (see Section 3.4).

When an ISA_NEG_RESP message is received, the receiving entity (original initiator) does the following:

1. Check the ISAKMP header as described in Section 2.1.1.
2. Depending on the SA Attributes, apply the agreed upon security services.
   (a) If the SA requires encryption, decrypt the ISA_NEG_RESP payload and Signature. If the decryption fails, the message is discarded and the following actions are taken:
      i. The event, *DECRYPTION FAILED*, is logged in the appropriate system audit file.
      ii. No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
   (b) If the SA requires authentication, the ISA_NEG_RESP packet is processed and the calculated signature is compared to the signature contained in the ISA_NEG_RESP packet. If these signatures are not identical, the message is discarded and the following actions are taken:
      i. The event, *INVALID SIGNATURE*, is logged in the appropriate system audit file.
      ii. No response is sent to the initiating entity. This will cause the transmission timer of the initiating entity to expire and force retransmission of the message.
3. Unpack the ISA_NEG_RESP payload and verify the SA attributes selected by responder are valid. If the attribute sets (or lists) are invalid or the responder rejected all proposed attribute sets (or lists), the receiving entity does the following:
   (a) The event, *INVALID ATTRIBUTES*, is logged in the appropriate system audit file.
   (b) Clear all state and return to IDLE.

   If the attribute set (or list) is valid, the receiving entity does the following:

   (a) Configure the protocol machine based on the attribute set (or list) selected.
4. If required, begin calculation of the new session key in the background.
5. Transition to SA Negotiation Conclusion (see Section 3.4).

3.4 SA Negotiation Conclusion

The SA negotiation concludes with the transmittal of the optional SA_COMMIT packet. This is determined by the setting of the SA Flags field. The SA_COMMIT message allows the initiating entity to inform the responding party that it has completed the processing required to set-up the SA and therefore, secure communications may begin. If the entity initiating the SA establishment does not have the ability to queue incoming data it may receive prior to its completion of SA establishment processing, then it requires the responding entity to wait for an SA_COMMIT message before sending data. The transmittal of the

ISA_COMMIT packet is optional and determined by the policy of the parties establishing the SA. All implementations MUST be able to generate, transmit, and receive this message.

The ISA_COMMIT packet is the ISAKMP header, described in section 2.1, with no payload.

3.4.1 SA Negotiation Conclusion Procedures

When issuing an ISA_COMMIT packet, the initiating entity does the following:

1. Construct an ISA_COMMIT packet (ISAKMP Header).
2. Depending on the SA Attributes established in the SA initialization phase, apply the agreed upon security services.
   (a) If the SA requires authentication, the ISA_COMMIT packet is processed (or signed) and the signature placed as noted in Figure 1.
   (b) If the SA requires encryption and the encryption algorithm is a block encryption algorithm, then padding up to the block size MUST be placed as noted in Figure 1.
   (c) If the SA requires encryption, the ISA_COMMIT Signature is encrypted.
3. Transmit the packet to the responding host as described in Section 2.1.1.
4. Clear all state and return to IDLE.

When an ISA_COMMIT packet is received, the receiving entity does the following:

1. Check the ISAKMP header as described in section 2.1.1.
2. Depending on the SA Attributes, apply the agreed upon security services.
   (a) If the SA requires encryption, decrypt the ISA_COMMIT Signature. If the decryption fails, the message is discarded and the following actions are taken:
      i. The event, *DECRYPTION FAILED*, is logged in the appropriate system audit file.
      ii. Because the ISA_COMMIT packet is a unidirectional message a retransmission will not be performed. Because the SA is established, we recommend that communications can proceed, however, the local security policy will dictate the procedures for continuing. We recommend that an ISA_NOTIFY packet with an Error Message Type (see Section 6) be sent to the originator of the ISA_COMMIT packet.
   (b) If the SA requires authentication, the ISA_COMMIT packet is processed and the calculated signature is compared to the signature contained in the ISA_COMMIT packet. If these signatures are not identical, the message is discarded and the following actions are taken:
      i. The event, *INVALID SIGNATURE*, is logged in the appropriate system audit file.
      ii. Because the ISA_COMMIT packet is a unidirectional message a retransmission will not be performed. Because the SA is established, we recommend that communications can proceed, however, the local security policy will dictate the procedures for continuing. We recommend that an ISA_NOTIFY packet with an Error Message Type (see Section 6) be sent to the originator of the ISA_COMMIT packet.
3. Clear all state and return to IDLE.

4 Security Association Modification

Security Association modification provides the ability to update security association attributes and parameters within an existing SA without having to establish a new SA. The use of this exchange can provide performance benefits without sacrificing the security of the existing communication. The most common use of this exchange will be to re-key an existing SA. The format for the ISA_MODIFY packet is the same as the ISA_INIT_REQ and ISA_INIT_RESP shown in Figure 4. All fields shown in Figure 4 exist for the ISA_MODIFY packets.

4.1 Modification Procedures

The procedure for exchanging information to modify an SA are similiar to the SA negotiation exchange. The details of SA modification will be described in this section as they are solidified during prototype development.

5 Security Association Deletion

During communications it is possible that hosts may be compromised or that information may be intercepted during transmission. Determining whether this has occurred is not an easy task and is outside the scope of this Internet-Draft. However, if it is discovered that transmissions are being compromised, then it is necessary to delete the current SA and establish a new SA.

The ISA_DELETE packet (shown in Figure 8) provides a controlled method of informing a peer entity that the initiating entity has deleted an SA(s). The ISA_DELETE packet allows for the deletion of any number of SAs with a single message. The receiving entity SHOULD clean up its local SA database. The receiving entity may be using the SA for secure communications with more than one party and would not want to actually delete the SA from its database in this case. However, upon receipt of an ISA_DELETE packet the SAs listed in the SPIs field of the packet cannot be used with the initiating entity. The SA Establishment procedure must be invoked to re-establish secure communications.

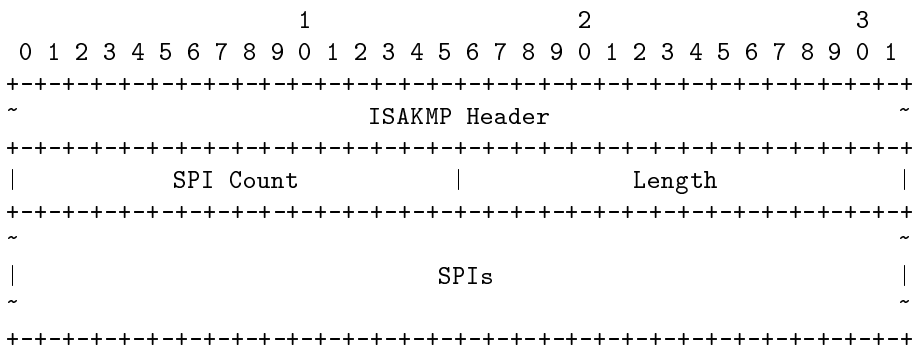


Figure 8: SA Delete Payload Format

- SPI Count - Number of security associations to be deleted
- Length - length of payload in octets
- SPIs - Initiator's Security Parameter Index(s) to be deleted

5.1 Deletion Procedures

When issuing an ISA_DELETE packet, the issuing entity (initiator or responder) does the following:

1. Create initiator cookie. See Section 2.3.4 for details.
2. Determine SPI of receiving entity.
3. Construct the ISA_DELETE packet.
4. Depending on the SA Attributes, apply the agreed upon security services.
   (a) If the SA requires authentication, the ISA_DELETE packet is processed and the signature placed as noted in Figure 1.
   (b) If the SA requires encryption, the ISA_DELETE payload and Signature are encrypted.
5. Transmit the packet to the destination host as described in Section 2.1.1.
6. Update the local SA database to reflect the SPI deletions.

Upon receipt of an ISA_DELETE packet, the receiving entity (initiator or responder) does the following:

1. Check the ISAKMP header as described in Section 2.1.1.
2. Depending on the SA Attributes, apply the agreed upon security services in the following order.
   (a) If the SA requires encryption, decrypt the ISA_DELETE payload and Signature. If the decryption fails, the message is discarded and the following actions are taken:
      i. The event is logged in the appropriate system audit file.
      ii. Because the ISA_DELETE packet is a unidirectional message a retransmission will not be performed. The local security policy will dictate the procedures for continuing. However, we recommend that the SPIs in the ISA_DELETE packet be checked to see if the originator was the communicating party. If so, then these SAs can be deleted from the local SA database. We also recommend that an ISA_NOTIFY packet with an Error Message Type (see Section 6) be sent to the originator of the ISA_DELETE packet. If the SPIs do not match those of the originator, then no further action should be taken.
   (b) If the SA requires authentication, the ISA_DELETE packet is processed and the calculated signature is compared to the signature contained in the ISA_DELETE packet. If these signatures are not identical, the message is discarded and the following actions are taken:
      i. The event is logged in the appropriate system audit file.

ii. Because the ISA_DELETE packet is a unidirectional message a retransmission will not be performed. The local security policy will dictate the procedures for continuing. However, we recommend that the SPIs in the ISA_DELETE packet be checked to see if the originator was the communicating party. If so, then these SAs can be deleted from the local SA database. We also recommend that an ISA_NOTIFY packet with an Error Message Type (see Section 6) be sent to the originator of the ISA_DELETE packet. If the SPIs do not match those of the originator, then no further action should be taken.

3. Unpack the ISA_DELETE payload.
4. Update the local SA database to reflect the SPI deletions.

6 Notification Message

The ISAKMP ISA_NOTIFY packet contains information one party wants to send to another. Notification information can be error messages specifying why a SA could not be established. It can also be status data that a process managing an SA database wishes to communicate with a peer process. For example, a secure front end or security gateway may use the ISA_NOTIFY message to synchronize SA communication (see Appendix A.2). The ISA_NOTIFY packet is unidirectional.

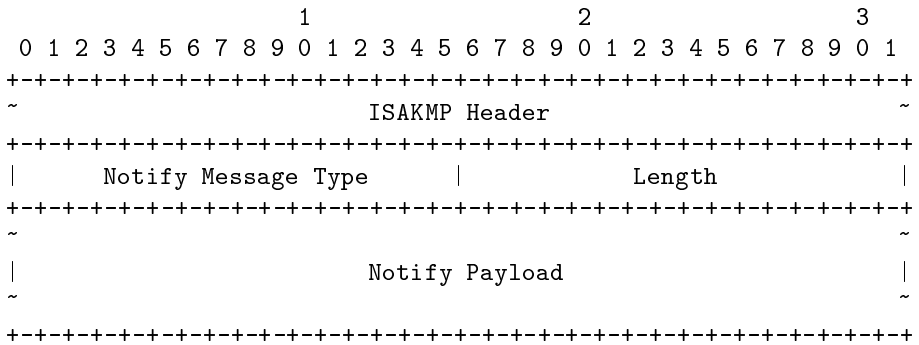


Figure 9: ISA NOTIFY Payload Format

- Notify Message Type (2 octets)

Notification	Notify Message Type
Error	1
Status	2

- Length (2 octets) - length of payload in octets
- Notify Payload (variable) - Value dependent on the Notify Message Type

6.1 Notification Procedures

When issuing an ISA_NOTIFY message, the issuing entity (initiator or responder) does the following:

1. Create initiator cookie. See Section 2.3.4 for details.
2. Determine SPI of receiving entity.
3. Construct ISA_NOTIFY packet.
4. Depending on the SA Attributes, apply the agreed upon security services.
   (a) If the SA requires authentication, the ISA_NOTIFY packet is processed and the signature placed as noted in Figure 1.
   (b) If the SA requires encryption, the ISA_NOTIFY payload and Signature are encrypted.
5. Transmit the packet to the destination host as described in Section 2.1.1.

Upon receipt of an ISA_NOTIFY message, the receiving entity (initiator or responder) does the following:

1. Check the ISAKMP header as described in Section 2.1.1.
2. Depending on the SA Attributes, apply the agreed upon security services in the following order.
   (a) If the SA requires encryption, decrypt the ISA_NOTIFY payload and Signature. If the decryption fails, the message is discarded and the following actions are taken:
      i. The event is logged in the appropriate system audit file.
      ii. Because the ISA_NOTIFY packet is a unidirectional message a retransmission will not be performed. The local security policy will dictate the procedures for continuing.
   (b) If the SA requires authentication, the ISA_NOTIFY packet is processed and the calculated signature is compared to the signature contained in the ISA_NOTIFY packet. If these signatures are not identical, the message is discarded and the following actions are taken:
      i. The event is logged in the appropriate system audit file.
      ii. Because the ISA_NOTIFY packet is a unidirectional message a retransmission will not be performed. The local security policy will dictate the procedures for continuing.
3. Unpack the ISA_NOTIFY payload.
4. Depending on the Notify Message Type, additional processing may be necessary.

7 Conclusions

The Internet Security Association and Key Management Protocol (ISAKMP) is a well designed protocol aimed at the Internet of the future. The massive growth of the Internet will lead to great diversity in network utilization, communications, and security requirements. ISAKMP contains all the features that will be needed for this dynamic and expanding communications environment.

ISAKMP's Security Association (SA) feature coupled with authentication and key establishment provides the security and flexibility that will be needed for future growth and diversity. This security diversity of multiple key exchange techniques, encryption algorithms, authentication mechanisms, security services, and security attributes will allow users to select the appropriate security for their network, communications, and security needs. The SA feature allows users to specify and negotiate security requirements with other users. An additional benefit of supporting multiple techniques in a single protocol is that as new techniques are developed they can easily be added to the protocol. This provides a path for the growth of Internet security services. ISAKMP supports both publicly or privately defined SAs, making it ideal for government, commercial, and private communications.

ISAKMP provides the ability to establish SAs for multiple security protocols and applications. These protocols and applications may be session-oriented or sessionless. Having one SA establishment protocol that supports multiple security protocols eliminates the need for multiple, nearly identical authentication, key exchange and SA establishment protocols when more than one security protocol is in use or desired. Just as IP has provided the common networking layer for the Internet, a common security establishment protocol is needed if security is to become a reality on the Internet. ISAKMP provides the common base that allows all other security protocols to interoperate.

ISAKMP follows good security design principles. It is not coupled to other insecure transport protocols, therefore it is not vulnerable or weakened by attacks on other protocols. Also, when more secure transport protocols are developed, ISAKMP can be easily migrated to them. ISAKMP also provides protection against protocol related attacks. This protection provides the assurance that the SAs and keys established are with the desired party and not with an attacker.

ISAKMP also follows good protocol design principles. Protocol specific information only is in the protocol header, following the design principles of IPv6. The data transported by the protocol is separated into functional payloads. As the Internet grows and evolves, new payloads to support new security functionality can be added without modifying the entire protocol.

A ISAKMP Scenarios

Examples scenerios are are presented to help illustrate the ISAKMP's ability to support multiple authentication methods and key exchanges.

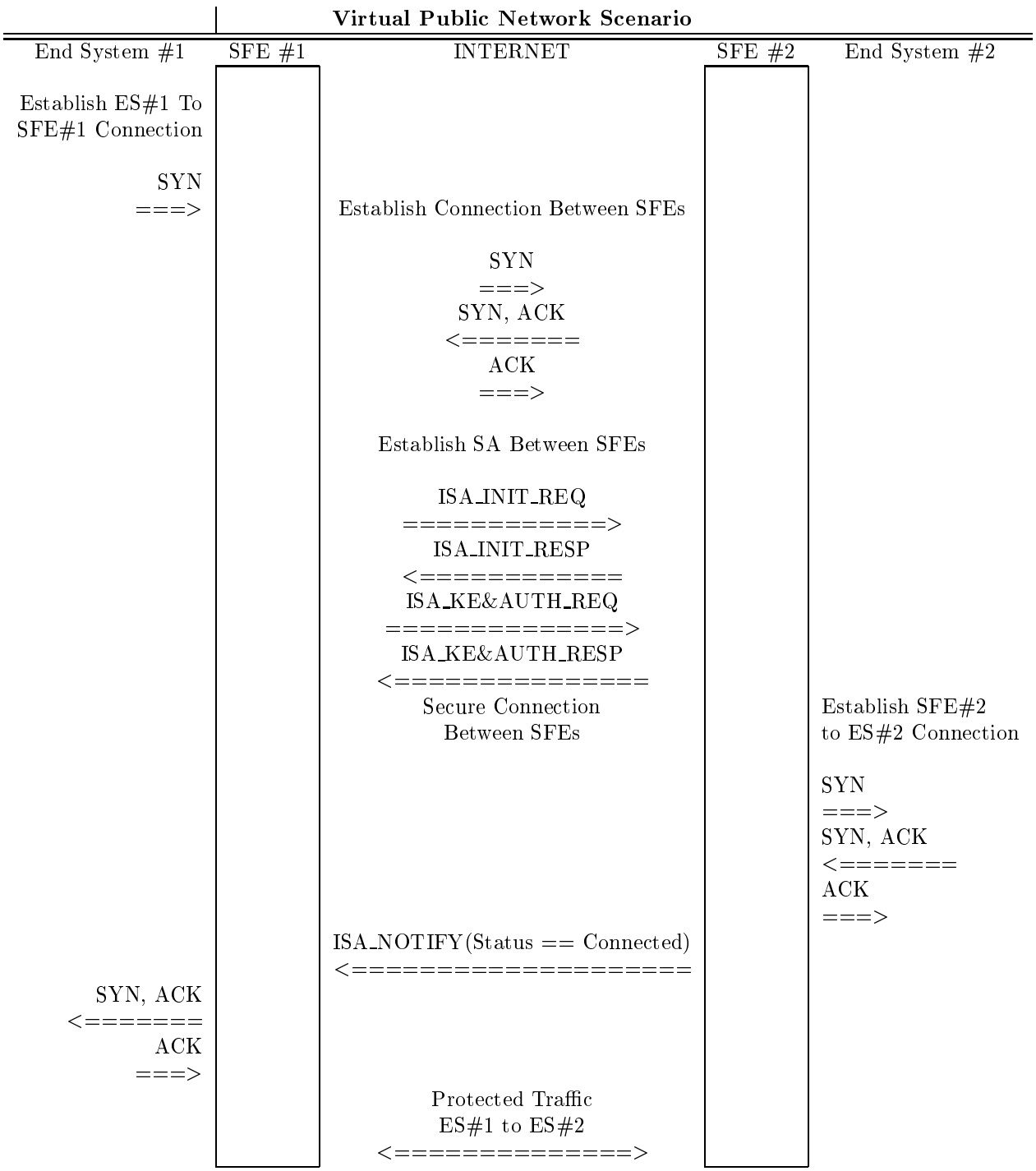
A.1 Initial ISAKMP Daemon Scenerio

This example steps through two ISAKMP daemons establishing an SA between themselves. This SA uses DNS Security Extentions [EK94] for authentication and a Photuris [Karn95] compliant key exchange. Following the SA establishment between the daemons, SAs are established for ESP and AH communications between user processes.

1. The initiating daemon sends an ISA_INIT_REQ messages with ISAKMP SA #3, #2, and #1 (in priority order). These SAs are defined in C.1.1.
2. The responding daemon sends an ISA_INIT_RESP message indicating that ISAKMP SA #2 was selected, which requires DNS Signature and Key Records and a Photuris compliant key exchange [DOW92].
3. The initiating daemon sends an ISA_KE_REQ packet with an index into well-known table of generator / prime pairs and it's public value.
4. Upon receipt of ISA_KE_REQ packet the responding daemon computes the shared secret and session key.
5. The responding daemon sends an ISA_KE_RESP packet with an its public value and both the initiator and responders public values signed using its Private (Signature) Key and encrypted in the session key created.
6. Upon receipt of ISA_KE_REQ packet the initiating daemon computes the shared secret and session key.
7. The initiating daemon sends an ISA_AUTH_REQ packet with both the initiator and responders public values signed using its Private (Signature) Key and it's DNS name and Public (Verification) Key signed by it nameserver. All encrypted in the session key created.
8. The responding daemon sends an ISA_AUTH_RESP packet with it's DNS name and Public (Verification) signed by it Secure DNS nameserver and encrypted in the session key created.
9. The initiating daemon sends an ISA_NEG_REQ packet with ESP SA #2, ESP SA #1, AH SA #1, and AH SA #2. These SAs are defined in C.2.1.
10. The responding daemon sends an ISA_NEG_RESP packet indicating that ESP SA #2, and AH SA #1 was selected.

A.2 Virtual Private Network Scenario

This scenario show how ISAKMP can be used in a Virtual Public Network (VPN). The ability to establish SAs for more than just ESP and AH and one of the uses of the ISA_NOTIFY message are also illustrated.



The diagram shows an End System (ES) using a connection oriented protocol (we use TCP as an example) establishing a connection with another ES. Both ES are behind Secure Front Ends (SFE) (e.g. firewalls). The connection establishment from End System #1 (ES#1) is intercepted by its Secure Front End (SFE #1). SFE#1 establishes a connection and then a Security Association (SA), using normal ISAKMP SA

establishment procedures, with SFE #2. Next SFE #2 establishes a connection with ES #2. Upon successful completion SFE #2 sends an SA_NOTIFY with Status equal Connected. SFE #1 completes it's connection with ES #1 and normal end to end communications takes place secured between SFE #1 and SFE #2. If SFE #2 had been unable to establish a connection with ES #2 it would have returned an SA_NOTIFY with Status equal Not Connected with an optional reason code.

B Security Association Attributes

This appendix contains a list of security attributes that should be considered when defining a Security Association (SA) for a security protocol or application. As an example, the security attributes culled from this list and required for an IP Security (AH, ESP) SA are defined in [RFC-1825]. The separation of ISAKMP from a specific SA definition is important to ensure ISAKMP can establish SAs for all possible security functionality. Each security function will be required to maintain a database of current SAs. This list is based upon an e-mail message [Kent94] to the IPSEC mail list from Steve Kent.

The authors welcome input on what are meaningful security attributes for an SA.

1. SAID.INBOUND
2. SAID.OUTBOUND
3. ENCAPSULATION
4. INBOUND-CRITERIA
   (a) IP-DESTINATION-ADDRESS
   (b) IP-SOURCE-ADDRESS
   (c) NEXT-PROTOCOL
   (d) IP-SECURITY-LABEL
   (e) TRANSPORT-DESTINATION-PORT
   (f) TRANSPORT-SOURCE-PORT
5. PEER-ADDRESS
6. AUTHENTICATION
   (a) ENABLED
   (b) MECHANISM
      - DIGITAL SIGNATURE
        i. KEY.INBOUND (Peer's Public Key)
        ii. KEY.OUTBOUND (Initator's Private Key)
7. ENCRYPTION
   (a) ENABLED
   (b) ALGORTIHM
   (c) KEY.INBOUND
   (d) KEY.OUTBOUND
   (e) IV.INBOUND
   (f) IV.OUTBOUND
8. INTEGRITY
   (a) ENABLED
   (b) PLAINTEXT
   (c) DIRECTION.ENABLED

(d) DIRECTION.VALUE
(e) ALGORITHM
(f) KEY.OUTBOUND
(g) KEY.INBOUND

9. COMPRESSION
   (a) ENABLED
   (b) ALGORITHM
10. REPLAY
   (a) ENABLED
   (b) SIZE
   (c) NUMBER.OUTBOUND
   (d) NUMBER.INBOUND
   (e) WINDOW.SIZE
   (f) WINDOW
11. FRAGMENTATION
   (a) INBOUND
   (b) OUTBOUND
12. KEY-MANAGEMENT
   (a) NEGOTIATED
   (b) TECHNIQUE
   (c) PARAMETERS
   (d) REKEY
      - GRACE
      - NEXT-SA
      - TIME-BASED
         i. ENABLE
         ii. TRIGGER
      - TRAFFIC-BASED
         i. ENABLE
         ii. PACKET-COUNT.INBOUND
         iii. PACKET-COUNT.OUTBOUND
         iv. TRIGGER.INBOUND
         v. TRIGGER.OUTBOUND

C Security Association Examples

C.1 ISAKMP SA Definition

The ISAKMP SA contains the SA attributes that are exchanged in the ISA_INIT messages.

ISAKMP Security Association

SA Attributes	Requirement
Peer ISAKMP Daemon Address	REQUIRED
Security Association Lifetime	REQUIRED
Certificate Authority	REQUIRED
Digital Signature Algorithm	REQUIRED
Signature Key(s)	REQUIRED
Security Association Lifetime	REQUIRED
Key Establishment Algorithm	REQUIRED
Cookie Generation Algorithm	REQUIRED
Sensitivity Level (e.g. Secret, Unclassified)	RECOMMENDED
Encryption Algorithm	RECOMMENDED
Encryption Mode	RECOMMENDED
Encryption Transform	RECOMMENDED
Encryption Key(s)	RECOMMENDED
Key Lifetime or Key Rollover	RECOMMENDED
Presence / Absence of cryptographic synchronization or IV	RECOMMENDED
Size of cryptographic synchronization or IV	RECOMMENDED

C.1.1 ISAKMP SA Examples

ISAKMP SA #1

SA Class	SA Type
Peer ISAKMP Daemon Address	N/A
Security Association Lifetime	86400 seconds (1day)
Certificate Authority	DMS Root CAW
Certificate Type	X.509v1m
Digital Signature Algorithm	DSA
Signature Key(s)	N/A
Security Association Lifetime	86400 seconds (1day)
Key Establishment Algorithm	Fortezza KEA
Cookie Generation Algorithm	SHA_1
Sensitivity Level (e.g. Secret, Unclassified)	Unclassified
Encryption Algorithm	Skipjack
Encryption Mode	CDC
Encryption Transform	NULL
Encryption Key(s)	N/A
Key Lifetime or Key Rollover	3600 seconds (1 hour)
Presence / Absence of cryptographic synchronization or IV	Present
Size of cryptographic synchronization or IV	64 bits

ISAKMP SA #2

SA Class	SA Type
Peer ISAKMP Daemon Address	N/A
Security Association Lifetime	86400 seconds (1day)
Certificate Authority	DNSSEC janeway.ncsc.mil
Certificate Type	RR
Digital Signature Algorithm	RSA
Signature Key(s)	N/A
Security Association Lifetime	86400 seconds (1day)
Key Establishment Algorithm	X9.42_STS
Cookie Generation Algorithm	MD5
Sensitivity Level (e.g. Secret, Unclassified)	N/A
Encryption Algorithm	DES
Encryption Mode	CDC
Encryption Transform	RFC-1829
Encryption Key(s)	N/A
Key Lifetime or Key Rollover	600 seconds (10 minutes)
Presence / Absence of cryptographic synchronization or IV	Present
Size of cryptographic synchronization or IV	64 bits

ISAKMP SA #3

SA Class	SA Type
Peer ISAKMP Daemon Address	N/A
Security Association Lifetime	86400 seconds (1day)
Certificate Authority	IPRA PCA UNINETT
Certificate Type	X.509v1
Digital Signature Algorithm	RSA
Signature Key(s)	N/A
Security Association Lifetime	86400 seconds (1day)
Key Establishment Algorithm	STS
Cookie Generation Algorithm	MD5
Sensitivity Level (e.g. Secret, Unclassified)	N/A
Encryption Algorithm	DES
Encryption Mode	CDC
Encryption Transform	RFC-1829
Encryption Key(s)	N/A
Key Lifetime or Key Rollover	600 seconds (10 minutes)
Presence / Absence of cryptographic synchronization or IV	Present
Size of cryptographic synchronization or IV	64 bits

C.2 ESP SA and AH SA Definitions

The following SAs are defined in [RFC-1825] and are presented here for comparative and completeness purposes.

AH Security Association

SA Attributes	Requirement
Authentication Algorithm	REQUIRED
Authentication Mode	REQUIRED
Authentication Key(s)	REQUIRED
Key Lifetime or Key Rollover	RECOMMENDED
Security Association Lifetime	RECOMMENDED
Sensitivity Level (e.g. Secret, Unclassified)	RECOMMENDED

ESP Security Association

SA Attributes	Requirement
Encryption Algorithm	REQUIRED
Encryption Mode	REQUIRED
Encryption Transform	REQUIRED
Encryption Key(s)	REQUIRED
Presence / Absence of cryptographic synchronization or IV	REQUIRED
Size of cryptographic synchronization or IV	REQUIRED
Authentication Algorithm	RECOMMENDED
Authentication Mode	RECOMMENDED
Authentication Key(s)	RECOMMENDED
Key Lifetime or Key Rollover	RECOMMENDED
Security Association Lifetime	RECOMMENDED
Sensitivity Level (e.g. Secret, Unclassified)	RECOMMENDED

C.2.1 ESP and AH SA Examples

AH SA #1

SA Class	SA Type
Authentication Algorithm	MD5
Authentication Mode	Keyed
Authentication Key(s)	Photuris
Key Lifetime or Key Rollover	600 seconds (10 minutes)
Security Association Lifetime	3600 seconds (1 hour)
Sensitivity Level (e.g. Secret, Unclassified)	N/A

AH SA #2

SA Class	SA Type
Authentication Algorithm	SHA
Authentication Mode	NULL
Authentication Key(s)	NULL
Key Lifetime or Key Rollover	600 seconds (10 minutes)
Security Association Lifetime	3600 seconds (1 hour)
Sensitivity Level (e.g. Secret, Unclassified)	N/A

ESP SA #1

SA Class	SA Type
Encryption Algorithm	DES
Encryption Mode	CBC
Encryption Transform	RFC-1829
Encryption Key(s)	Phutoris Generated
Presence / Absence of cryptographic synchronization or IV	Present
Size of cryptographic synchronization or IV	64 bits
Authentication Algorithm	NULL
Authentication Mode	NULL
Authentication Key(s)	NULL
Key Lifetime or Key Rollover	600 seconds (10 minutes)
Security Association Lifetime	3600 seconds (1 hour)
Sensitivity Level (e.g. Secret, Unclassified)	N/A

ESP SA #2

SA Class	SA Type
Encryption Algorithm	DES
Encryption Mode	CBC
Encryption Transform	RFC-1829
Encryption Key(s)	X9.42_DH Generated
Presence / Absence of cryptographic synchronization or IV	Present
Size of cryptographic synchronization or IV	64 bits
Authentication Algorithm	NULL
Authentication Mode	NULL
Authentication Key(s)	NULL
Key Lifetime or Key Rollover	600 seconds (10 minutes)
Security Association Lifetime	3600 seconds (1 hour)
Sensitivity Level (e.g. Secret, Unclassified)	N/A

C.2.2 Fortezza SA Examples

Fortezza AH SA

SA Class	SA Type
Authentication Algorithm	SHA
Authentication Mode	NULL
Authentication Key(s)	DMS Root CAW
Key Lifetime or Key Rollover	86400 seconds (1day)
Security Association Lifetime	86400 seconds (1day)
Sensitivity Level (e.g. Secret, Unclassified)	N/A

Fortezza ESP SA

SA Class	SA Type
Encryption Algorithm	Skipjack
Encryption Mode	CBC
Encryption Transform	NULL
Encryption Key(s)	Fortezza KEA Generated
Presence / Absence of cryptographic synchronization or IV	Present
Size of cryptographic synchronization or IV	64 bits
Authentication Algorithm	DSA
Authentication Mode	NULL
Authentication Key(s)	DMS Root CAW
Key Lifetime or Key Rollover	3600 seconds (1 hour)
Security Association Lifetime	86400 seconds (1day)
Sensitivity Level (e.g. Secret, Unclassified)	Unclassified

Security Considerations

Cryptographic analysis techniques are improving at a steady pace. The continuing improvement in processing power makes once computational prohibitive cryptographic attacks more realistic. New cryptographic algorithms and public key generation techniques are also being developed at a steady pace. New security services and mechanisms are being developed at an accelerated pace. A consistent method of choosing from a variety of security services and mechanisms and to exchange attributes required by the mechanisms is important to security in the complex structure of the Internet. However a system that locks itself into a single cryptographic algorithm, key exchange technique, or security mechanism will become increasingly vulnerable as time passes.

UDP is an unreliable datagram protocol and therefore its use in ISAKMP introduces a number of security considerations. Since UDP is unreliable, but a key management protocol must be reliable, the reliability is built into ISAKMP. While ISAKMP utilizes UDP as its transport mechanism, it doesn't soley rely on any UDP information (e.g. checksum, length) for its processing.

Another issue that must be considered in the development of IKMP is the effect of firewalls on the protocol. Many firewalls filter out all UDP packets, making reliance on UDP questionable in certian environments.

A number of very important security considerations are presented in [RFC-1825]. One bares repeating. Once a private session key is created it must be safely stored. Failure to properly protect the private key from access both internal and external to the system completely nullifies any protect provided by the IP Security services.

Acknowledgements

Marsha Gross, Bill Kutz, Mike Oehler, Mark Schneider, and Pete Sell provided significant input and review to this document.

Thanks to Carl Muckenhirn of SPARTA, Inc. for his assistance with LaTeX.

References

[ANSI94] ANSI, *X9.42: Public Key Cryptography for the Financial Services Industry – Establishment of Symmetric Algorithm Keys Using Diffie-Hellman*, Working Draft, October 26, 1995.

[DOW92] W. Diffie, M.Wiener, P. Van Oorschot, *Authtication and Authenticated Key Exchanges*, Designs, Codes, and Cryptography, 2, 107-125, Kluwer Academic Publishers, 1992.

[Berg] Berge, N.H., *UNINETT PCA Policy Statements*, Internet-Draft, work in progress, November, 1995.

[EK94] Eastlake III, D. and C. Kaufman, *Domain Name System Protocol Security Extensions*, Internet-Draft, work in progress, Oct, 1995.

[Karn95] Karn P. and B. Simpson, *The Photuris Key Management Protocol*, Internet-Draft, work in progress, November, 1995.

[Kent94] Steve Kent, *IPSEC SMIB*, e-mail to ipsec@ans.net, August 10, 1994.

[RFC-1155] Rose M. and K. McCloghrie, *Structure and Identification of Management Information for TCP/IP-based Internets*, RFC-1155, May, 1990.

[RFC-1212] McCloghrie K. and M. Rose, *Concise MIB Definitions*, RFC-1212, March 26, 1991.

[RFC-1213] McCloghrie K. and M. Rose, *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II*, RFC-1213, March 26, 1991.

[RFC-1422] Steve Kent, *Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management*, RFC-1422, February 1993.

[RFC-1825] Randell Atkinson, *Security Architecture for the Internet Protocol*, RFC-1825, August, 1995.

[Secu] SECUREWARE INC., *Peer Authentication and Key Management Protocol Specification*, Version 2.2, October 27, 1995.

[Schn94] Bruce Schneier, *Applied Cryptography - Protocols, Algorithms, and Source Code in C*, John Wiley & Sons, Inc., 1994.

[Spar94a] Harney H., C. Muckenhirn, and T. Rivers, *Group Key Management (GKMP) Architecture*, SPARTA, Inc., Internet-Draft, September, 1994.

[Spar94b] Harney H., C. Muckenhirn, and T. Rivers, *Group Key Management (GKMP) Specification*, SPARTA, Inc., Internet-Draft, September, 1994.

Addresses of Authors

The two authors are with:

National Security Agency
ATTN: R23
9800 Savage Road
Ft. Meade, MD. 20755-6000

Douglas Maughan
Phone: 301-688-0847
E-mail: `wdmaugh@tycho.ncsc.mil`

Mark Schertler
Phone: 301-688-0849
E-mail: `mjs@tycho.ncsc.mil`