Network Working Group   D. Maughan
Request for Comments: 2408   National Security Agency
Category: Standards Track   M. Schertler
                                                       Securify, Inc.
                                                       M. Schneider
                                             National Security Agency
                                                          J. Turner
                                               RABA Technologies, Inc.
                                                       November 1998

     Internet Security Association and Key Management Protocol (ISAKMP)

Status of this Memo

Copyright Notice

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.

Table of Contents

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.

      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 various network
      security services, such as the IP layer services (such as header
      authentication and payload encapsulation), transport or application
      layer services, or self-protection of negotiation traffic. ISAKMP
      defines payloads for exchanging key generation and authentication
      data.   These formats provide a consistent framework for transferring
      key and authentication data which is independent of the key
      generation technique, encryption algorithm and authentication
      mechanism.

ISAKMP is distinct from key exchange protocols in order to cleanly
separate the details of security association management (and key
management) from the details of key exchange. There may be many
different key exchange protocols, each with different security
properties. However, a common framework is required for agreeing to
the format of SA attributes, and for negotiating, modifying, and
deleting SAs. ISAKMP serves as this common framework.

Separating the functionality into three parts adds complexity to the
security analysis of a complete ISAKMP implementation. However, the
separation is critical for interoperability between systems with
differing security requirements, and should also simplify the
analysis of further evolution of a ISAKMP server.

ISAKMP is intended to support the negotiation of SAs for security
protocols at all layers of the network stack (e.g., IPSEC, TLS, TLSP,
OSPF, etc.). By centralizing the management of the security
associations, ISAKMP reduces the amount of duplicated functionality
within each security protocol. ISAKMP can also reduce connection
setup time, by negotiating a whole stack of services at once.

The remainder of section 1 establishes the motivation for security
negotiation and outlines the major components of ISAKMP, i.e.
Security Associations and Management, Authentication, Public Key
Cryptography, and Miscellaneous items. Section 2 presents the
terminology and concepts associated with ISAKMP. Section 3 describes
the different ISAKMP payload formats. Section 4 describes how the
payloads of ISAKMP are composed together as exchange types to
establish security associations and perform key exchanges in an
authenticated manner. Additionally, security association
modification, deletion, and error notification are discussed.
Section 5 describes the processing of each payload within the context
of ISAKMP exchanges, including error handling and associated actions.
The appendices provide the attribute values necessary for ISAKMP and
requirement for defining a new Domain of Interpretation (DOI) within
ISAKMP.

1.1 Requirements Terminology

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD,
SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this
document, are to be interpreted as described in [RFC-2119].

1.2 The Need for Negotiation

ISAKMP extends the assertion in [DOW92] that authentication and key
exchanges must be combined for better security to include security
association exchanges. The security services required for

communications depends on the individual network configurations and
environments. Organizations are setting up Virtual Private Networks
(VPN), also known as Intranets, 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 within the
same 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 that the
user supports in an authenticated and protected manner for agreement
upon a common set of security attributes, i.e. an interoperable
security association.

1.3 What can be Negotiated?

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 exchange
   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.

1.4 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). [SEC-ARCH] provides details
   on IP Security Associations (SA) and Security Parameter Index (SPI)
   definitions.

1.4.1 Security Associations and Registration

   The SA attributes required and recommended for the IP Security (AH,
   ESP) are defined in [SEC-ARCH]. 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 requirements for
   SA attributes. The separation of ISAKMP from a specific SA
   definition is important to ensure ISAKMP can es tablish SAs for all
   possible security protocols and applications.

   NOTE: See [IPDOI] 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.

1.4.2 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 negotiating entities followed by the
   establishment of a security association by these negotiating entities
   in behalf of some protocol (e.g. ESP/AH). 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 between the negotiating
   server entities, the initial ISAKMP exchange may be skipped and the
   establishment of a security association can be done directly. After
   the basic set of security attributes has been agreed upon, initial
   identity authenticated, and required keys generated, the established
   SA can 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 A.

1.5 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. Sending cleartext keys or other unprotected
   authenticating information over a network is weak, due to the threat
   of reading them with a network sniffer. Additionally, sending one-
   way hashed poorly-chosen keys with low entropy is also weak, due to
   the threat of brute-force guessing attacks on the sniffed messages.
   While passwords can be used for establishing identity, they are not
   considered in this context because of recent statements from the
   Internet Architecture Board [IAB]. 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 public key digital signatures each entity requires a
   public key and a private key. Certificates are an essential part of
   a digital signature authentication mechanism. Certificates bind a
   specific entity's 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 [Schneier].

1.5.1 Certificate Authorities

Certificates require an infrastructure for generation, verification,
revocation, 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 [DNSSEC] 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.

1.5.2 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
[Berge] 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 associated 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 usually the entity's e-mail address which has
meaning to those, and only those, who understand e-mail. Another web
of trust could use an entirely different naming scheme.

1.5.3 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,
   which makes access control questionable. While encryption (e.g.
   ESP) and integrity (e.g. AH) will protect subsequent communications
   from passive eavesdroppers, without authentication it is possible
   that the SA and key may have been established with an adversary who
   performed an active man-in-the-middle attack and is now stealing all
   your personal data.

   A digital signature algorithm MUST be used within ISAKMP's
   authentication component. However, ISAKMP does not mandate a
   specific signature algorithm or certificate authority (CA). ISAKMP
   allows an entity initiating communications to indicate which CAs it
   supports. After selection of a CA, the protocol provides the
   messages required to support the actual authentication exchange. The
   protocol provides a facility for identification of different
   certificate authorities, certificate types (e.g. X.509, PKCS #7,
   PGP, DNS SIG and KEY records), and the exchange of the certificates
   identified.

   ISAKMP utilizes digital signatures, based on public key 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 its network
   domain. A client's proof that it holds its secret key provides
   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. The use of these additional services and protocols
   will be described in a Key Exchange specific document.

1.6 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], [ANSI], and [Oakley]). Properties
of key exchange protocols include the key establishment method,
authentication, symmetry, perfect forward secrecy, and back traffic
protection.

NOTE: 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.

1.6.1 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 than 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
and the independence of keys from one key exchange to another
provides perfect forward secrecy. Detailed descriptions of these
algorithms can be found in [Schneier]. 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 provided 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 echange. Authentication after
the protocol must occur in subsequent commu nications.
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 affecting the key that is generated. This is
desirable so that computation of the keys does not require either
party to know who initated the exchange. While key exchange symmetry
is desirable, symmetry in the entire key management protocol may
provide a vulnerablity to reflection attacks.

Perfect Forward Secrecy: As described in [DOW92], an authenticated
key exchange protocol provides perfect forward secrecy if disclosure
of longterm secret keying material does not compromise the secrecy of
the exchanged keys from previous communications. The property of
perfect forward secrecy does not apply to key exchange without
authentication.

1.6.2 ISAKMP Requirements

An authenticated 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.
However, [IKE] describes a proposal for using the Oakley key exchange
[Oakley] in conjunction with ISAKMP. Requirements that should be
evaluated when choosing a key establishment algorithm include
establishment method (generation vs. transport), perfect forward
secrecy, 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.

1.7 ISAKMP Protection

1.7.1 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. A "cookie" or anti-clogging token (ACT) is aimed at
protecting the computing resources from attack without spending
excessive CPU resources to determine its authenticity. 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). Absolute protection against denial of service is
impossible, but this anti-clogging token provides a technique for
making it easier to handle. The use of an anti-clogging token was
introduced by Karn and Simpson in [Karn].

It should be noted that in the exchanges shown in section 4, the
anticlogging mechanism should be used in conjuction with a garbage-
state collection mechanism; an attacker can still flood a server
using packets with bogus IP addresses and cause state to be created.
Such aggressive memory management techniques SHOULD be employed by
protocols using ISAKMP that do not go through an initial, anti-
clogging only phase, as was done in [Karn].

1.7.2 Connection Hijacking

ISAKMP prevents connection hijacking by linking the authentication,
key exchange and security association exchanges. This linking
prevents 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.

1.7.3 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 anyone other than 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.

1.8 Multicast Communications

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 [SEC-ARCH]. Multicast
security issues are also discussed in [RFC-1949] and [BC]. A future
extension to ISAKMP will support multicast key distribution. For an
introduction to the issues related to multicast security, consult the
Internet Drafts, [RFC-2094] and [RFC-2093], describing Sparta's
research in this area.

2 Terminology and Concepts

2.1 ISAKMP Terminology

   Security Protocol: A Security Protocol consists of an entity at a
   single point in the network stack, performing a security service for
   network communication. For example, IPSEC ESP and IPSEC AH are two
   different security protocols. TLS is another example. Security
   Protocols may perform more than one service, for example providing
   integrity and confidentiality in one module.

   Protection Suite: A protection suite is a list of the security
   services that must be applied by various security protocols. For
   example, a protection suite may consist of DES encryption in IP ESP,
   and keyed MD5 in IP AH. All of the protections in a suite must be
   treated as a single unit. This is necessary because security
   services in different security protocols can have subtle
   interactions, and the effects of a suite must be analyzed and
   verified as a whole.

   Security Association (SA): A Security Association is a security-
   protocol- specific set of parameters that completely defines the
   services and mechanisms necessary to protect traffic at that security
   protocol location. These parameters can include algorithm
   identifiers, modes, cryptographic keys, etc. The SA is referred to
   by its associated security protocol (for example, "ISAKMP SA", "ESP
   SA", "TLS SA").

   ISAKMP SA: An SA used by the ISAKMP servers to protect their own
   traffic. Sections 2.3 and 2.4 provide more details about ISAKMP SAs.

   Security Parameter Index (SPI): An identifier for a Security
   Assocation, relative to some security protocol. Each security
   protocol has its own "SPI-space". A (security protocol, SPI) pair
   may uniquely identify an SA. The uniqueness of the SPI is
   implementation dependent, but could be based per system, per
   protocol, or other options. Depending on the DOI, additional
   information (e.g. host address) may be necessary to identify an SA.
   The DOI will also determine which SPIs (i.e. initiator's or
   responder's) are sent during communication.

   Domain of Interpretation: A Domain of Interpretation (DOI) defines
   payload formats, exchange types, and conventions for naming
   security-relevant information such as security policies or
   cryptographic algorithms and modes. A Domain of Interpretation (DOI)
   identifier is used to interpret the payloads of ISAKMP payloads. A
   system SHOULD support multiple Domains of Interpretation
   simultaneously. The concept of a DOI is based on previous work by

the TSIG CIPSO Working Group, but extends beyond security label
interpretation to include naming and interpretation of security
services. A DOI defines:

o A "situation": the set of information that will be used to
   determine the required security services.

o The set of security policies that must, and may, be supported.

o A syntax for the specification of proposed security services.

o A scheme for naming security-relevant information, including
   encryption algorithms, key exchange algorithms, security policy
   attributes, and certificate authorities.

o The specific formats of the various payload contents.

o Additional exchange types, if required.

The rules for the IETF IP Security DOI are presented in [IPDOI].
Specifications of the rules for customized DOIs will be presented in
separate documents.

Situation: A situation contains all of the security-relevant
information that a system considers necessary to decide the security
services required to protect the session being negotiated. The
situation may include addresses, security classifications, modes of
operation (normal vs. emergency), etc.

Proposal: A proposal is a list, in decreasing order of preference, of
the protection suites that a system considers acceptable to protect
traffic under a given situation.

Payload: ISAKMP defines several types of payloads, which are used to
transfer information such as security association data, or key
exchange data, in DOI-defined formats. A payload consists of a
generic payload header and a string of octects that is opaque to
ISAKMP. ISAKMP uses DOI- specific functionality to synthesize and
interpret these payloads. Multiple payloads can be sent in a single
ISAKMP message. See section 3 for more details on the payload types,
and [IPDOI] for the formats of the IETF IP Security DOI payloads.

Exchange Type: An exchange type is a specification of the number of
messages in an ISAKMP exchange, and the payload types that are
contained in each of those messages. Each exchange type is designed
to provide a particular set of security services, such as anonymity
of the participants, perfect forward secrecy of the keying material,
authentication of the participants, etc. Section 4.1 defines the

default set of ISAKMP exchange types. Other exchange types can be
added to support additional key exchanges, if required.

2.2 ISAKMP Placement

Figure 1 is a high level view of the placement of ISAKMP within a
system context in a network architecture. An important part of
negotiating security services is to consider the entire "stack" of
individual SAs as a unit. This is referred to as a "protection
suite".

```
 +------------+        +--------+              +-------------+
 !    DOI     !        !        !              ! Application !
 ! Definition ! <----> ! ISAKMP !              !   Process   !
 +------------+   --> !        !              !-------------!
+--------------+   !   +--------+              ! Appl Protocol!
! Key Exchange !   !     ^  ^                  +-------------+
! Definition  !<--     !  !                         ^
+--------------+       !  !                         !
                       !  !                         !
      !----------------!  !                         !
      v                   !                         !
   +-------+              v                         v
   !  API  !       +---------------------------------------------+
   +-------+       !                 Socket Layer               !
      !            !---------------------------------------------!
      v            !      Transport Protocol (TCP / UDP)         !
 +----------+      !---------------------------------------------!
 ! Security ! <----> !                  IP                       !
 ! Protocol !      !---------------------------------------------!
 +----------+      !           Link Layer Protocol              !
                   +---------------------------------------------+
```
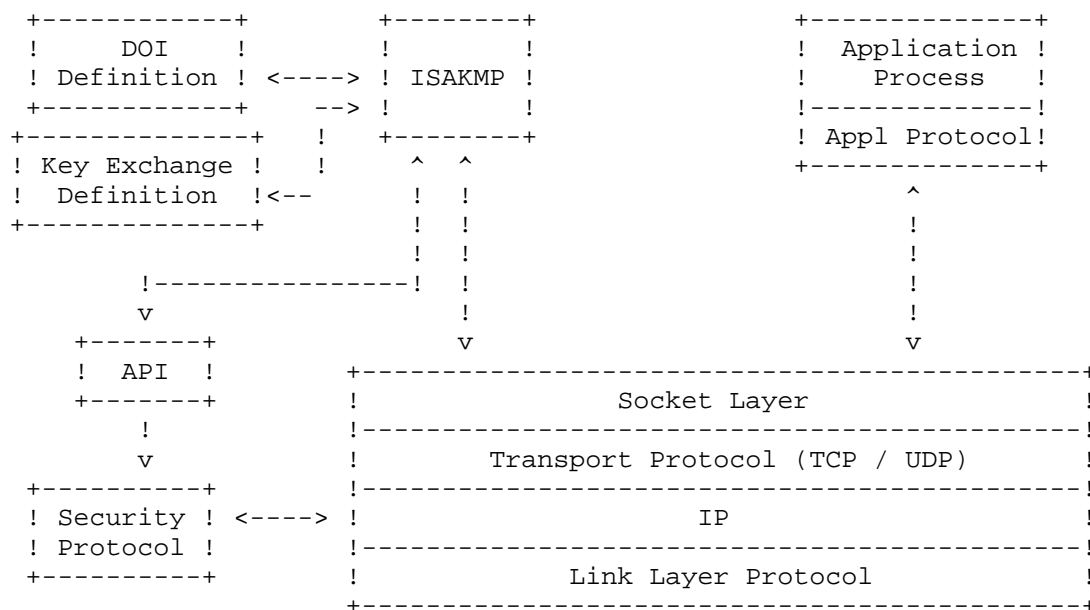
Figure 1: ISAKMP Relationships

2.3 Negotiation Phases

ISAKMP offers two "phases" of negotiation. In the first phase, two
entities (e.g. ISAKMP servers) agree on how to protect further
negotiation traffic between themselves, establishing an ISAKMP SA.
This ISAKMP SA is then used to protect the negotiations for the
Protocol SA being requested. Two entities (e.g. ISAKMP servers) can
negotiate (and have active) multiple ISAKMP SAs.

The second phase of negotiation is used to establish security
associations for other security protocols. This second phase can be
used to establish many security associations. The security
associations established by ISAKMP during this phase can be used by a
security protocol to protect many message/data exchanges.

While the two-phased approach has a higher start-up cost for most
simple scenarios, there are several reasons that it is beneficial for
most cases.

First, entities (e.g. ISAKMP servers) can amortize the cost of the
first phase across several second phase negotiations. This allows
multiple SAs to be established between peers over time without having
to start over for each communication.

Second, security services negotiated during the first phase provide
security properties for the second phase. For example, after the
first phase of negotiation, the encryption provided by the ISAKMP SA
can provide identity protection, potentially allowing the use of
simpler second-phase exchanges. On the other hand, if the channel
established during the first phase is not adequate to protect
identities, then the second phase must negotiate adequate security
mechanisms.

Third, having an ISAKMP SA in place considerably reduces the cost of
ISAKMP management activity - without the "trusted path" that an
ISAKMP SA gives you, the entities (e.g. ISAKMP servers) would have
to go through a complete re-authentication for each error
notification or deletion of an SA.

Negotiation during each phase is accomplished using ISAKMP-defined
exchanges (see section 4) or exchanges defined for a key exchange
within a DOI.

Note that security services may be applied differently in each
negotiation phase. For example, different parties are being
authenticated during each of the phases of negotiation. During the
first phase, the parties being authenticated may be the ISAKMP
servers/hosts, while during the second phase, users or application
level programs are being authenticated.

2.4 Identifying Security Associations

While bootstrapping secure channels between systems, ISAKMP cannot
assume the existence of security services, and must provide some
protections for itself. Therefore, ISAKMP considers an ISAKMP
Security Association to be different than other types, and manages
ISAKMP SAs itself, in their own name space. ISAKMP uses the two

cookie fields in the ISAKMP header to identify ISAKMP SAs. The Message ID in the ISAKMP Header and the SPI field in the Proposal payload are used during SA establishment to identify the SA for other security protocols. The interpretation of these four fields is dependent on the operation taking place.

The following table shows the presence or absence of several fields during SA establishment. The following fields are necessary for various operations associated with SA establishment: cookies in the ISAKMP header, the ISAKMP Header Message ID field, and the SPI field in the Proposal payload. An 'X' in the column means the value MUST be present. An 'NA' in the column means a value in the column is Not Applicable to the operation.

#	Operation	I-Cookie	R-Cookie	Message ID	SPI
(1)	Start ISAKMP SA negotiation	X	0	0	0
(2)	Respond ISAKMP SA negotiation	X	X	0	0
(3)	Init other SA negotiation	X	X	X	X
(4)	Respond other SA negotiation	X	X	X	X
(5)	Other (KE, ID, etc.)	X	X	X/0	NA
(6)	Security Protocol (ESP, AH)	NA	NA	NA	X

In the first line (1) of the table, the initiator includes the Initiator Cookie field in the ISAKMP Header, using the procedures outlined in sections 2.5.3 and 3.1.

In the second line (2) of the table, the responder includes the Initiator and Responder Cookie fields in the ISAKMP Header, using the procedures outlined in sections 2.5.3 and 3.1. Additional messages may be exchanged between ISAKMP peers, depending on the ISAKMP exchange type used during the phase 1 negotiation. Once the phase 1 exchange is completed, the Initiator and Responder cookies are included in the ISAKMP Header of all subsequent communications between the ISAKMP peers.

During phase 1 negotiations, the initiator and responder cookies determine the ISAKMP SA. Therefore, the SPI field in the Proposal payload is redundant and MAY be set to 0 or it MAY contain the transmitting entity's cookie.

In the third line (3) of the table, the initiator associates a Message ID with the Protocols contained in the SA Proposal. This Message ID and the initiator's SPI(s) to be associated with each protocol in the Proposal are sent to the responder. The SPI(s) will be used by the security protocols once the phase 2 negotiation is completed.

In the fourth line (4) of the table, the responder includes the same
Message ID and the responder's SPI(s) to be associated with each
protocol in the accepted Proposal. This information is returned to
the initiator.

In the fifth line (5) of the table, the initiator and responder use
the Message ID field in the ISAKMP Header to keep track of the in-
progress protocol negotiation. This is only applicable for a phase 2
exchange and the value MUST be 0 for a phase 1 exchange because the
combined cookies identify the ISAKMP SA. The SPI field in the
Proposal payload is not applicable because the Proposal payload is
only used during the SA negotiation message exchange (steps 3 and 4).

In the sixth line (6) of the table, the phase 2 negotiation is
complete. The security protocols use the SPI(s) to determine which
security services and mechanisms to apply to the communication
between them. The SPI value shown in the sixth line (6) is not the
SPI field in the Proposal payload, but the SPI field contained within
the security protocol header.

During the SA establishment, a SPI MUST be generated. ISAKMP is
designed to handle variable sized SPIs. This is accomplished by
using the SPI Size field within the Proposal payload during SA
establishment. Handling of SPIs will be outlined by the DOI
specification (e.g. [IPDOI]).

When a security association (SA) is initially established, one side
assumes the role of initiator and the other the role of responder.
Once the SA is established, both the original initiator and responder
can initiate a phase 2 negotiation with the peer entity. Thus,
ISAKMP SAs are bidirectional in nature.

Additionally, ISAKMP allows both initiator and responder to have some
control during the negotiation process. While ISAKMP is designed to
allow an SA negotiation that includes multiple proposals, the
initiator can maintain some control by only making one proposal in
accordance with the initiator's local security policy. Once the
initiator sends a proposal containing more than one proposal (which
are sent in decreasing preference order), the initiator relinquishes
control to the responder. Once the responder is controlling the SA
establishment, the responder can make its policy take precedence over
the initiator within the context of the multiple options offered by
the initiator. This is accomplished by selecting the proposal best
suited for the responder's local security policy and returning this
selection to the initiator.

2.5 Miscellaneous

2.5.1 Transport Protocol

   ISAKMP can be implemented over any transport protocol or over IP
   itself. Implementations MUST include send and receive capability for
   ISAKMP using the User Datagram Protocol (UDP) on port 500. UDP Port
   500 has been assigned to ISAKMP by the Internet Assigned Numbers
   Authority (IANA). Implementations MAY additionally support ISAKMP
   over other transport protocols or over IP itself.

2.5.2 RESERVED Fields

   The existence of RESERVED fields within ISAKMP payloads are used
   strictly to preserve byte alignment. 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 a zero (0) value and
   discard the packet if other values are found.

2.5.3 Anti-Clogging Token ("Cookie") Creation

   The details of cookie generation are implementation dependent, but
   MUST satisfy these basic requirements (originally stated by Phil Karn
   in [Karn]):

      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 to help prevent replay attacks, therefore, the date and
   time MUST be added to the information hashed. The generated cookies
   are placed in the ISAKMP Header (described in section 3.1) Initiator

and Responder cookie fields. These fields are 8 octets in length,
thus, requiring a generated cookie to be 8 octets. Notify and Delete
messages (see sections 3.14, 3.15, and 4.8) are uni-directional
transmissions and are done under the protection of an existing ISAKMP
SA, thus, not requiring the generation of a new cookie. One
exception to this is the transmission of a Notify message during a
Phase 1 exchange, prior to completing the establishment of an SA.
Sections 3.14 and 4.8 provide additional details.

3 ISAKMP Payloads

ISAKMP payloads provide modular building blocks for constructing
ISAKMP messages. The presence and ordering of payloads in ISAKMP is
defined by and dependent upon the Exchange Type Field located in the
ISAKMP Header (see Figure 2). The ISAKMP payload types are discussed
in sections 3.4 through 3.15. The descriptions of the ISAKMP
payloads, messages, and exchanges (see Section 4) are shown using
network octet ordering.

3.1 ISAKMP Header Format

An ISAKMP message has a fixed header format, shown in Figure 2,
followed by a variable number of payloads. 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 possibly prevent denial of service or replay attacks.

The ISAKMP Header fields are defined as follows:

   o Initiator Cookie (8 octets) - Cookie of entity that initiated SA
      establishment, SA notification, or SA deletion.

   o Responder Cookie (8 octets) - Cookie of entity that is responding
      to an SA establishment request, SA notification, or SA deletion.

```
                          1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      !                          Initiator                            !
      !                            Cookie                             !
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      !                          Responder                            !
      !                            Cookie                             !
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      !  Next Payload ! MjVer ! MnVer ! Exchange Type !     Flags     !
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      !                          Message ID                           !
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      !                            Length                             !
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
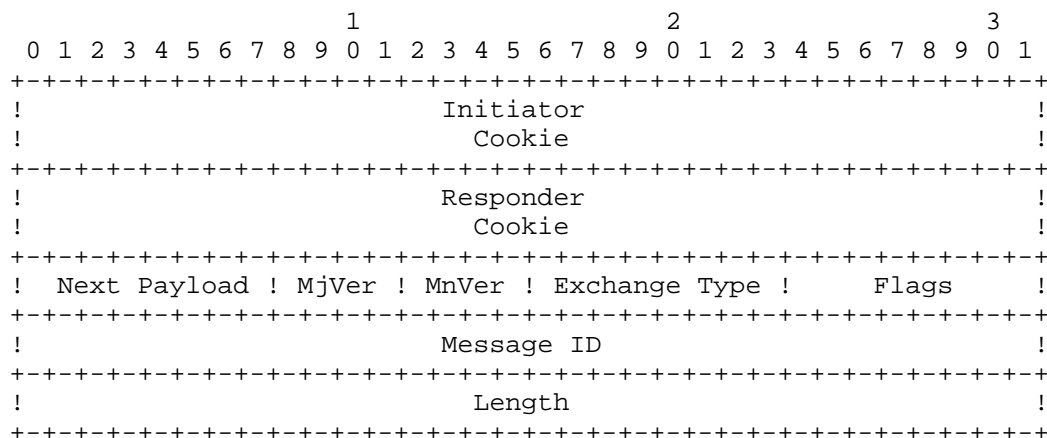
                   Figure 2: ISAKMP Header Format

   o Next Payload (1 octet) - Indicates the type of the first payload
      in the message. The format for each payload is defined in
      sections 3.4 through 3.16. The processing for the payloads is
      defined in section 5.

                 Next Payload Type   Value
              NONE   0
              Security Association (SA)   1
              Proposal (P)   2
              Transform (T)   3
              Key Exchange (KE)   4
              Identification (ID)   5
              Certificate (CERT)   6
              Certificate Request (CR)   7
              Hash (HASH)   8
              Signature (SIG)   9
              Nonce (NONCE)   10
              Notification (N)   11
              Delete (D)   12
              Vendor ID (VID)   13
              RESERVED   14 - 127
              Private USE   128 - 255

   o Major Version (4 bits) - indicates the major version of the ISAKMP
      protocol in use. Implementations based on this version of the
      ISAKMP Internet-Draft MUST set the Major Version to 1.
      Implementations based on previous versions of ISAKMP Internet-
      Drafts MUST set the Major Version to 0. Implementations SHOULD

          never accept packets with a major version number larger than its
          own.

     o Minor Version (4 bits) - indicates the minor version of the
        ISAKMP protocol in use. Implementations based on this version of
        the ISAKMP Internet-Draft MUST set the Minor Version to 0.
        Implementations based on previous versions of ISAKMP Internet-
        Drafts MUST set the Minor Version to 1. Implementations SHOULD
        never accept packets with a minor version number larger than its
        own, given the major version numbers are identical.

     o Exchange Type (1 octet) - indicates the type of exchange being
        used. This dictates the message and payload orderings in the
        ISAKMP exchanges.

                      Exchange Type   Value
                   NONE   0
                   Base   1
                   Identity Protection   2
                   Authentication Only   3
                   Aggressive   4
                   Informational   5
                   ISAKMP Future Use   6 - 31
                   DOI Specific Use   32 - 239
                   Private Use   240 - 255

     o Flags (1 octet) - indicates specific options that are set for the
        ISAKMP exchange. The flags listed below are specified in the
        Flags field beginning with the least significant bit, i.e the
        Encryption bit is bit 0 of the Flags field, the Commit bit is bit
        1 of the Flags field, and the Authentication Only bit is bit 2 of
        the Flags field. The remaining bits of the Flags field MUST be
        set to 0 prior to transmission.

        -- E(ncryption Bit) (1 bit) - If set (1), all payloads following
           the header are encrypted using the encryption algorithm
           identified in the ISAKMP SA. The ISAKMP SA Identifier is the
           combination of the initiator and responder cookie. It is
           RECOMMENDED that encryption of communications be done as soon
           as possible between the peers. For all ISAKMP exchanges
           described in section 4.1, the encryption SHOULD begin after
           both parties have exchanged Key Exchange payloads. If the
           E(ncryption Bit) is not set (0), the payloads are not
           encrypted.

-- C(ommit Bit) (1 bit) - This bit is used to signal key exchange
   synchronization. It is used to ensure that encrypted material
   is not received prior to completion of the SA establishment.
   The Commit Bit can be set (at anytime) by either party
   participating in the SA establishment, and can be used during
   both phases of an ISAKMP SA establishment. However, the value
   MUST be reset after the Phase 1 negotiation. If set(1), the
   entity which did not set the Commit Bit MUST wait for an
   Informational Exchange containing a Notify payload (with the
   CONNECTED Notify Message) from the entity which set the Commit
   Bit. In this instance, the Message ID field of the
   Informational Exchange MUST contain the Message ID of the
   original ISAKMP Phase 2 SA negotiation. This is done to
   ensure that the Informational Exchange with the CONNECTED
   Notify Message can be associated with the correct Phase 2 SA.
   The receipt and processing of the Informational Exchange
   indicates that the SA establishment was successful and either
   entity can now proceed with encrypted traffic communication.
   In addition to synchronizing key exchange, the Commit Bit can
   be used to protect against loss of transmissions over
   unreliable networks and guard against the need for multiple
   re-transmissions.

   NOTE: It is always possible that the final message of an
   exchange can be lost. In this case, the entity expecting to
   receive the final message of an exchange would receive the
   Phase 2 SA negotiation message following a Phase 1 exchange or
   encrypted traffic following a Phase 2 exchange. Handling of
   this situation is not standardized, but we propose the
   following possibilities. If the entity awaiting the
   Informational Exchange can verify the received message (i.e.
   Phase 2 SA negotiation message or encrypted traffic), then
   they MAY consider the SA was established and continue
   processing. The other option is to retransmit the last ISAKMP
   message to force the other entity to retransmit the final
   message. This suggests that implementations may consider
   retaining the last message (locally) until they are sure the
   SA is established.

-- A(uthentication Only Bit) (1 bit) - This bit is intended for
   use with the Informational Exchange with a Notify payload and
   will allow the transmission of information with integrity
   checking, but no encryption (e.g. "emergency mode"). Section
   4.8 states that a Phase 2 Informational Exchange MUST be sent
   under the protection of an ISAKMP SA. This is the only
   exception to that policy. If the Authentication Only bit is
   set (1), only authentication security services will be applied
   to the entire Notify payload of the Informational Exchange and

the payload will not be encrypted.

o Message ID (4 octets) - Unique Message Identifier used to
   identify protocol state during Phase 2 negotiations. This value
   is randomly generated by the initiator of the Phase 2
   negotiation. In the event of simultaneous SA establishments
   (i.e. collisions), the value of this field will likely be
   different because they are independently generated and, thus, two
   security associations will progress toward establishment.
   However, it is unlikely there will be absolute simultaneous
   establishments. During Phase 1 negotiations, the value MUST be
   set to 0.

o Length (4 octets) - Length of total message (header + payloads)
   in octets. Encryption can expand the size of an ISAKMP message.

3.2 Generic Payload Header

Each ISAKMP payload defined in sections 3.4 through 3.16 begins with
a generic header, shown in Figure 3, which provides a payload
"chaining" capability and clearly defines the boundaries of a
payload.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
! Next Payload  !   RESERVED    !         Payload Length        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
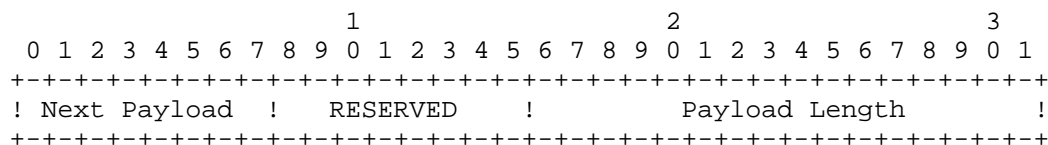
                  Figure 3: Generic Payload Header

The Generic Payload Header fields are defined as follows:

o Next Payload (1 octet) - Identifier for the payload type of the
   next payload in the message. If the current payload is the last
   in the message, then this field will be 0. This field provides
   the "chaining" capability.

o RESERVED (1 octet) - Unused, set to 0.

o Payload Length (2 octets) - Length in octets of the current
   payload, including the generic payload header.

3.3 Data Attributes

There are several instances within ISAKMP where it is necessary to
represent Data Attributes. An example of this is the Security
Association (SA) Attributes contained in the Transform payload

(described in section 3.6). These Data Attributes are not an ISAKMP
payload, but are contained within ISAKMP payloads. The format of the
Data Attributes provides the flexibility for representation of many
different types of information. There can be multiple Data
Attributes within a payload. The length of the Data Attributes will
either be 4 octets or defined by the Attribute Length field. This is
done using the Attribute Format bit described below. Specific
information about the attributes for each domain will be described in
a DOI document, e.g. IPSEC DOI [IPDOI].

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !A!        Attribute Type        !    AF=0  Attribute Length    !
 !F!                              !    AF=1  Attribute Value     !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 .                     AF=0  Attribute Value                     .
 .                     AF=1  Not Transmitted                     .
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
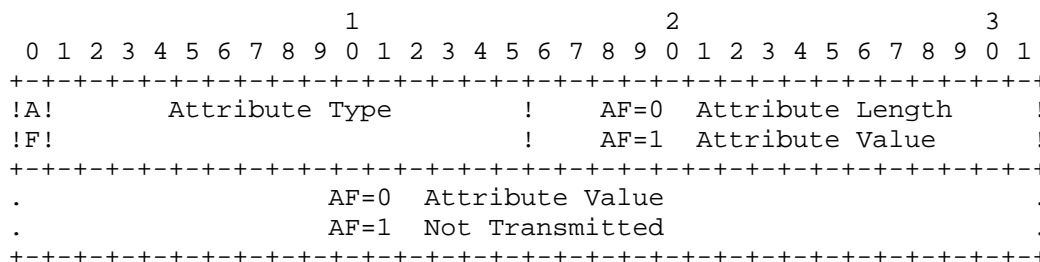
                       Figure 4: Data Attributes

The Data Attributes fields are defined as follows:

 o Attribute Type (2 octets) - Unique identifier for each type of
    attribute. These attributes are defined as part of the DOI-
    specific information.

    The most significant bit, or Attribute Format (AF), indicates
    whether the data attributes follow the Type/Length/Value (TLV)
    format or a shortened Type/Value (TV) format. If the AF bit is a
    zero (0), then the Data Attributes are of the Type/Length/Value
    (TLV) form. If the AF bit is a one (1), then the Data Attributes
    are of the Type/Value form.

 o Attribute Length (2 octets) - Length in octets of the Attribute
    Value. When the AF bit is a one (1), the Attribute Value is only
    2 octets and the Attribute Length field is not present.

 o Attribute Value (variable length) - Value of the attribute
    associated with the DOI-specific Attribute Type. If the AF bit
    is a zero (0), this field has a variable length defined by the
    Attribute Length field. If the AF bit is a one (1), the
    Attribute Value has a length of 2 octets.

3.4 Security Association Payload

   The Security Association Payload is used to negotiate security
   attributes and to indicate the Domain of Interpretation (DOI) and
   Situation under which the negotiation is taking place. Figure 5
   shows the format of the Security Association payload.

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! Next Payload  !   RESERVED    !         Payload Length        !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !              Domain of Interpretation  (DOI)                  !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !                                                               !
 ~                           Situation                           ~
 !                                                               !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
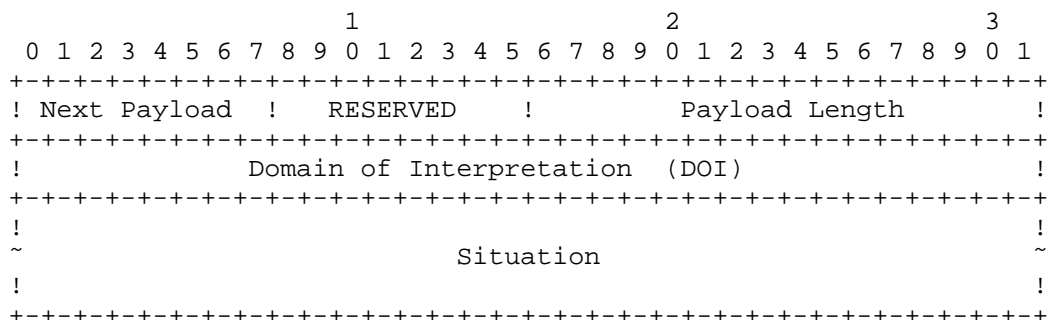
                  Figure 5: Security Association Payload

   o Next Payload (1 octet) - Identifier for the payload type of the
      next payload in the message. If the current payload is the last
      in the message, then this field will be 0. This field MUST NOT
      contain the values for the Proposal or Transform payloads as they
      are considered part of the security association negotiation. For
      example, this field would contain the value "10" (Nonce payload)
      in the first message of a Base Exchange (see Section 4.4) and the
      value "0" in the first message of an Identity Protect Exchange
      (see Section 4.5).

   o RESERVED (1 octet) - Unused, set to 0.

   o Payload Length (2 octets) - Length in octets of the entire
      Security Association payload, including the SA payload, all
      Proposal payloads, and all Transform payloads associated with the
      proposed Security Association.

   o Domain of Interpretation (4 octets) - Identifies the DOI (as
      described in Section 2.1) under which this negotiation is taking
      place. The DOI is a 32-bit unsigned integer. A DOI value of 0
      during a Phase 1 exchange specifies a Generic ISAKMP SA which can
      be used for any protocol during the Phase 2 exchange. The
      necessary SA Attributes are defined in A.4. A DOI value of 1 is
      assigned to the IPsec DOI [IPDOI]. All other DOI values are
      reserved to IANA for future use. IANA will not normally assign a
      DOI value without referencing some public specification, such as

an Internet RFC. Other DOI's can be defined using the description
in appendix B. This field MUST be present within the Security
Association payload.

o Situation (variable length) - A DOI-specific field that
   identifies the situation under which this negotiation is taking
   place. The Situation is used to make policy decisions regarding
   the security attributes being negotiated. Specifics for the IETF
   IP Security DOI Situation are detailed in [IPDOI]. This field
   MUST be present within the Security Association payload.

3.5 Proposal Payload

The Proposal Payload contains information used during Security
Association negotiation. The proposal consists of security
mechanisms, or transforms, to be used to secure the communications
channel. Figure 6 shows the format of the Proposal Payload. A
description of its use can be found in section 4.2.

```
                    1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
! Next Payload  !   RESERVED    !         Payload Length        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!  Proposal #   !  Protocol-Id  !    SPI Size   !# of Transforms!
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                        SPI (variable)                         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                 Figure 6: Proposal Payload Format

The Proposal Payload fields are defined as follows:

o Next Payload (1 octet) - Identifier for the payload type of the
   next payload in the message. This field MUST only contain the
   value "2" or "0". If there are additional Proposal payloads in
   the message, then this field will be 2. If the current Proposal
   payload is the last within the security association proposal,
   then this field will be 0.

o RESERVED (1 octet) - Unused, set to 0.

o Payload Length (2 octets) - Length in octets of the entire
   Proposal payload, including generic payload header, the Proposal
   payload, and all Transform payloads associated with this
   proposal. In the event there are multiple proposals with the
   same proposal number (see section 4.2), the Payload Length field

only applies to the current Proposal payload and not to all
Proposal payloads.

o Proposal # (1 octet) - Identifies the Proposal number for the
   current payload. A description of the use of this field is found
   in section 4.2.

o Protocol-Id (1 octet) - Specifies the protocol identifier for the
   current negotiation. Examples might include IPSEC ESP, IPSEC AH,
   OSPF, TLS, etc.

o SPI Size (1 octet) - Length in octets of the SPI as defined by
   the Protocol-Id. In the case of ISAKMP, the Initiator and
   Responder cookie pair from the ISAKMP Header is the ISAKMP SPI,
   therefore, the SPI Size is irrelevant and MAY be from zero (0) to
   sixteen (16). If the SPI Size is non-zero, the content of the
   SPI field MUST be ignored. If the SPI Size is not a multiple of
   4 octets it will have some impact on the SPI field and the
   alignment of all payloads in the message. The Domain of
   Interpretation (DOI) will dictate the SPI Size for other
   protocols.

o # of Transforms (1 octet) - Specifies the number of transforms
   for the Proposal. Each of these is contained in a Transform
   payload.

o SPI (variable) - The sending entity's SPI. In the event the SPI
   Size is not a multiple of 4 octets, there is no padding applied
   to the payload, however, it can be applied at the end of the
   message.

The payload type for the Proposal Payload is two (2).

3.6 Transform Payload

The Transform Payload contains information used during Security
Association negotiation. The Transform payload consists of a
specific security mechanism, or transforms, to be used to secure the
communications channel. The Transform payload also contains the
security association attributes associated with the specific
transform. These SA attributes are DOI-specific. Figure 7 shows the
format of the Transform Payload. A description of its use can be
found in section 4.2.

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! Next Payload  !   RESERVED    !         Payload Length        !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! Transform #   ! Transform-Id  !           RESERVED2           !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !                                                               !
 ~                        SA Attributes                          ~
 !                                                               !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
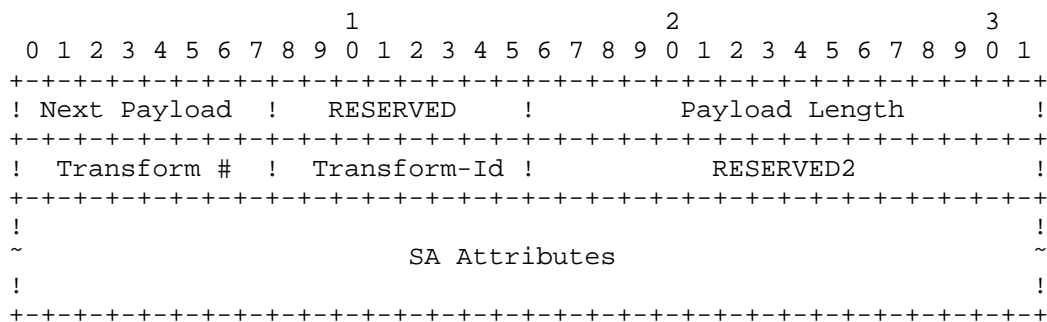
                 Figure 7: Transform Payload Format

   The Transform Payload fields are defined as follows:

   o Next Payload (1 octet) - Identifier for the payload type of the
      next payload in the message. This field MUST only contain the
      value "3" or "0". If there are additional Transform payloads in
      the proposal, then this field will be 3. If the current
      Transform payload is the last within the proposal, then this
      field will be 0.

   o RESERVED (1 octet) - Unused, set to 0.

   o Payload Length (2 octets) - Length in octets of the current
      payload, including the generic payload header, Transform values,
      and all SA Attributes.

   o Transform # (1 octet) - Identifies the Transform number for the
      current payload. If there is more than one transform proposed
      for a specific protocol within the Proposal payload, then each
      Transform payload has a unique Transform number. A description
      of the use of this field is found in section 4.2.

   o Transform-Id (1 octet) - Specifies the Transform identifier for
      the protocol within the current proposal. These transforms are
      defined by the DOI and are dependent on the protocol being
      negotiated.

   o RESERVED2 (2 octets) - Unused, set to 0.

   o SA Attributes (variable length) - This field contains the
      security association attributes as defined for the transform
      given in the Transform-Id field. The SA Attributes SHOULD be
      represented using the Data Attributes format described in section
      3.3. If the SA Attributes are not aligned on 4-byte boundaries,

then subsequent payloads will not be aligned and any padding will
be added at the end of the message to make the message 4-octet
aligned.

The payload type for the Transform Payload is three (3).

3.7 Key Exchange Payload

The Key Exchange Payload supports a variety of key exchange
techniques. Example key exchanges are Oakley [Oakley], Diffie-
Hellman, the enhanced Diffie-Hellman key exchange described in X9.42
[ANSI], and the RSA-based key exchange used by PGP. Figure 8 shows
the format of the Key Exchange payload.

The Key Exchange Payload fields are defined as follows:

 o Next Payload (1 octet) - Identifier for the payload type of the
    nextpayload in the message. If the current payload is the last
    in the message, then this field will be 0.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Next Payload  !   RESERVED    !          Payload Length       !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !                                                               !
   ~                    Key Exchange Data                          ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                Figure 8: Key Exchange Payload Format

 o RESERVED (1 octet) - Unused, set to 0.

 o Payload Length (2 octets) - Length in octets of the current
    payload, including the generic payload header.

 o Key Exchange Data (variable length) - Data required to generate a
    session key. The interpretation of this data is specified by the
    DOI and the associated Key Exchange algorithm. This field may
    also contain pre-placed key indicators.

The payload type for the Key Exchange Payload is four (4).

3.8 Identification Payload

   The Identification Payload contains DOI-specific data used to
   exchange identification information. This information is used for
   determining the identities of communicating peers and may be used for
   determining authenticity of information. Figure 9 shows the format
   of the Identification Payload.

   The Identification Payload fields are defined as follows:

   o Next Payload (1 octet) - Identifier for the payload type of the
      next payload in the message. If the current payload is the last
      in the message, then this field will be 0.

   o RESERVED (1 octet) - Unused, set to 0.

   o Payload Length (2 octets) - Length in octets of the current
      payload, including the generic payload header.

   o ID Type (1 octet) - Specifies the type of Identification being
      used.

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! Next Payload  !   RESERVED    !         Payload Length        !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !   ID Type     !             DOI Specific ID Data              !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !                                                               !
 ~                    Identification Data                        ~
 !                                                               !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
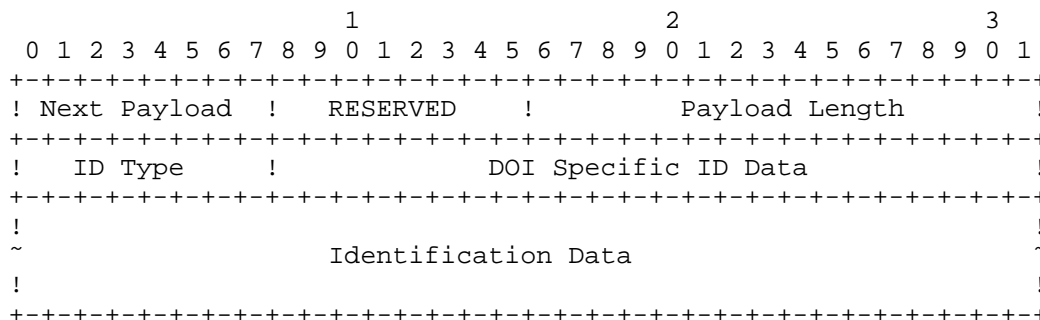
             Figure 9: Identification Payload Format

      This field is DOI-dependent.

   o DOI Specific ID Data (3 octets) - Contains DOI specific
      Identification data. If unused, then this field MUST be set to
      0.

   o Identification Data (variable length) - Contains identity
      information. The values for this field are DOI-specific and the
      format is specified by the ID Type field. Specific details for
      the IETF IP Security DOI Identification Data are detailed in
      [IPDOI].

The payload type for the Identification Payload is five (5).

3.9 Certificate Payload

   The Certificate Payload provides a means to transport certificates or
   other certificate-related information via ISAKMP and can appear in
   any ISAKMP message. Certificate payloads SHOULD be included in an
   exchange whenever an appropriate directory service (e.g. Secure DNS
   [DNSSEC]) is not available to distribute certificates. The
   Certificate payload MUST be accepted at any point during an exchange.
   Figure 10 shows the format of the Certificate Payload.

   NOTE: Certificate types and formats are not generally bound to a DOI
   - it is expected that there will only be a few certificate types, and
   that most DOIs will accept all of these types.

   The Certificate Payload fields are defined as follows:

    o Next Payload (1 octet) - Identifier for the payload type of the
       next payload in the message. If the current payload is the last
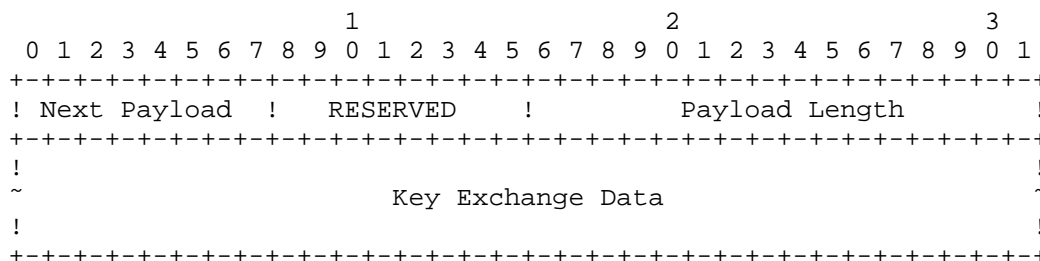   in the message, then this field will be 0.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Next Payload  !   RESERVED    !         Payload Length        !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Cert Encoding !                                               !
   +-+-+-+-+-+-+-+-+                                               !
   ~                       Certificate Data                       ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                 Figure 10: Certificate Payload Format

    o RESERVED (1 octet) - Unused, set to 0.

    o Payload Length (2 octets) - Length in octets of the current
       payload, including the generic payload header.

    o Certificate Encoding (1 octet) - This field indicates the type of
       certificate or certificate-related information contained in the
       Certificate Data field.

```
                    Certificate Type           Value
              NONE                               0
              PKCS #7 wrapped X.509 certificate  1
              PGP Certificate                    2
              DNS Signed Key                     3
              X.509 Certificate - Signature      4
              X.509 Certificate - Key Exchange   5
              Kerberos Tokens                    6
              Certificate Revocation List (CRL)  7
              Authority Revocation List (ARL)    8
              SPKI Certificate                   9
              X.509 Certificate - Attribute      10
              RESERVED                           11 - 255
```

   o Certificate Data (variable length) - Actual encoding of
      certificate data. The type of certificate is indicated by the
      Certificate Encoding field.

   The payload type for the Certificate Payload is six (6).

3.10 Certificate Request Payload

   The Certificate Request Payload provides a means to request
   certificates via ISAKMP and can appear in any message. Certificate
   Request payloads SHOULD be included in an exchange whenever an
   appropriate directory service (e.g. Secure DNS [DNSSEC]) is not
   available to distribute certificates. The Certificate Request
   payload MUST be accepted at any point during the exchange. The
   responder to the Certificate Request payload MUST send its
   certificate, if certificates are supported, based on the values
   contained in the payload. If multiple certificates are required,
   then multiple Certificate Request payloads SHOULD be transmitted.
   Figure 11 shows the format of the Certificate Request Payload.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Next Payload  !   RESERVED    !         Payload Length        !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !  Cert. Type   !                                               !
   +-+-+-+-+-+-+-+-+                                               !
   ~                     Certificate Authority                    ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

         Figure 11: Certificate Request Payload Format

The Certificate Payload fields are defined as follows:

o Next Payload (1 octet) - Identifier for the payload type of the
  next payload in the message. If the current payload is the last
  in the message, then this field will be 0.

o RESERVED (1 octet) - Unused, set to 0.

o Payload Length (2 octets) - Length in octets of the current
  payload, including the generic payload header.
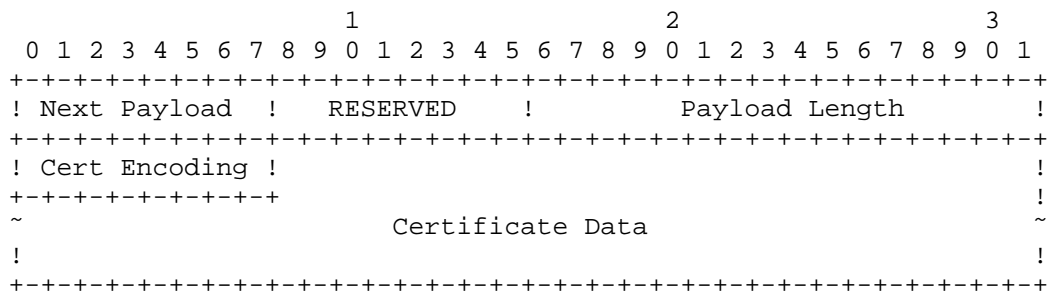
o Certificate Type (1 octet) - Contains an encoding of the type of
  certificate requested. Acceptable values are listed in section
  3.9.

o Certificate Authority (variable length) - Contains an encoding of
  an acceptable certificate authority for the type of certificate
  requested. As an example, for an X.509 certificate this field
  would contain the Distinguished Name encoding of the Issuer Name
  of an X.509 certificate authority acceptable to the sender of
  this payload. This would be included to assist the responder in
  determining how much of the certificate chain would need to be
  sent in response to this request. If there is no specific
  certificate authority requested, this field SHOULD not be
  included.

The payload type for the Certificate Request Payload is seven (7).

3.11 Hash Payload

   The Hash Payload contains data generated by the hash function
   (selected during the SA establishment exchange), over some part of
   the message and/or ISAKMP state. This payload may be used to verify
   the integrity of the data in an ISAKMP message or for authentication
   of the negotiating entities. Figure 12 shows the format of the Hash
   Payload.

```
                         1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ! Next Payload  !   RESERVED    !         Payload Length        !
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    !                                                               !
    ~                           Hash Data                           ~
    !                                                               !
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
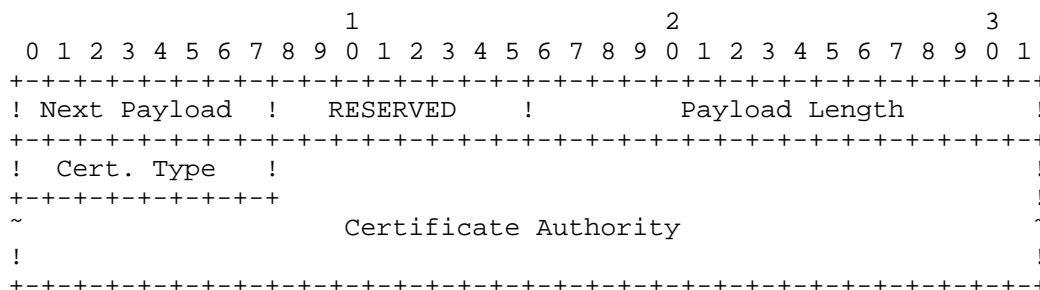
                    Figure 12: Hash Payload Format

   The Hash Payload fields are defined as follows:

    o Next Payload (1 octet) - Identifier for the payload type of the
       next payload in the message. If the current payload is the last
       in the message, then this field will be 0.

    o RESERVED (1 octet) - Unused, set to 0.

    o Payload Length (2 octets) - Length in octets of the current
       payload, including the generic payload header.

    o Hash Data (variable length) - Data that results from applying the
       hash routine to the ISAKMP message and/or state.

3.12 Signature Payload

   The Signature Payload contains data generated by the digital
   signature function (selected during the SA establishment exchange),
   over some part of the message and/or ISAKMP state. This payload is
   used to verify the integrity of the data in the ISAKMP message, and
   may be of use for non-repudiation services. Figure 13 shows the
   format of the Signature Payload.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Next Payload  !   RESERVED    !         Payload Length        !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !                                                               !
   ~                       Signature Data                          ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
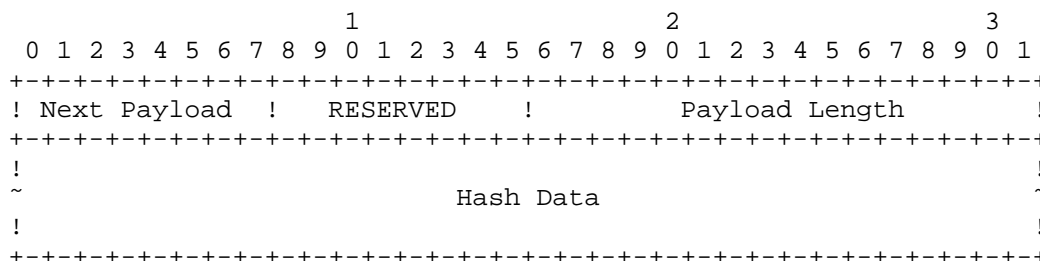
                Figure 13: Signature Payload Format

   The Signature Payload fields are defined as follows:

   o Next Payload (1 octet) - Identifier for the payload type of the
      next payload in the message. If the current payload is the last
      in the message, then this field will be 0.

   o RESERVED (1 octet) - Unused, set to 0.

   o Payload Length (2 octets) - Length in octets of the current
      payload, including the generic payload header.

   o Signature Data (variable length) - Data that results from
      applying the digital signature function to the ISAKMP message
      and/or state.

   The payload type for the Signature Payload is nine (9).

3.13 Nonce Payload

   The Nonce Payload contains random data used to guarantee liveness
   during an exchange and protect against replay attacks. Figure 14
   shows the format of the Nonce Payload. If nonces are used by a
   particular key exchange, the use of the Nonce payload will be
   dictated by the key exchange. The nonces may be transmitted as part
   of the key exchange data, or as a separate payload. However, this is
   defined by the key exchange, not by ISAKMP.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Next Payload  !   RESERVED    !          Payload Length       !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !                                                               !
   ~                          Nonce Data                          ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
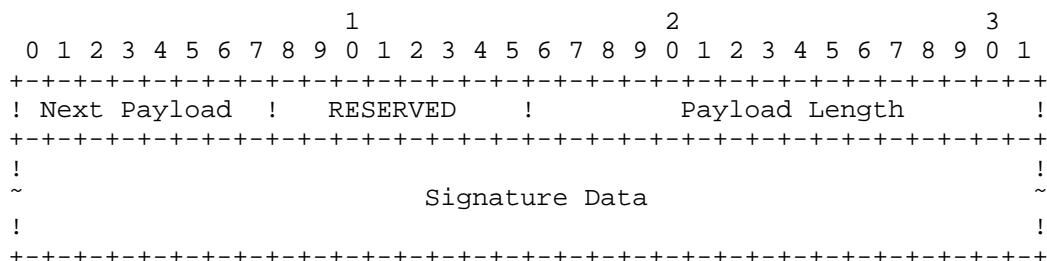
                 Figure 14: Nonce Payload Format

   The Nonce Payload fields are defined as follows:

   o Next Payload (1 octet) - Identifier for the payload type of the
      next payload in the message. If the current payload is the last
      in the message, then this field will be 0.

   o RESERVED (1 octet) - Unused, set to 0.

   o Payload Length (2 octets) - Length in octets of the current
      payload, including the generic payload header.

   o Nonce Data (variable length) - Contains the random data generated
      by the transmitting entity.

   The payload type for the Nonce Payload is ten (10).

3.14 Notification Payload

   The Notification Payload can contain both ISAKMP and DOI-specific
   data and is used to transmit informational data, such as error
   conditions, to an ISAKMP peer. It is possible to send multiple
   Notification payloads in a single ISAKMP message. Figure 15 shows
   the format of the Notification Payload.

   Notification which occurs during, or is concerned with, a Phase 1
   negotiation is identified by the Initiator and Responder cookie pair
   in the ISAKMP Header. The Protocol Identifier, in this case, is
   ISAKMP and the SPI value is 0 because the cookie pair in the ISAKMP
   Header identifies the ISAKMP SA. If the notification takes place
   prior to the completed exchange of keying information, then the
   notification will be unprotected.

Notification which occurs during, or is concerned with, a Phase 2
negotiation is identified by the Initiator and Responder cookie pair
in the ISAKMP Header and the Message ID and SPI associated with the
current negotiation. One example for this type of notification is to
indicate why a proposal was rejected.

```
                        1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Next Payload  !   RESERVED    !         Payload Length        !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !              Domain of Interpretation  (DOI)                  !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ! Protocol-ID  !   SPI Size    !      Notify Message Type       !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !                                                               !
   ~                Security Parameter Index (SPI)                 ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   !                                                               !
   ~                      Notification Data                        ~
   !                                                               !
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
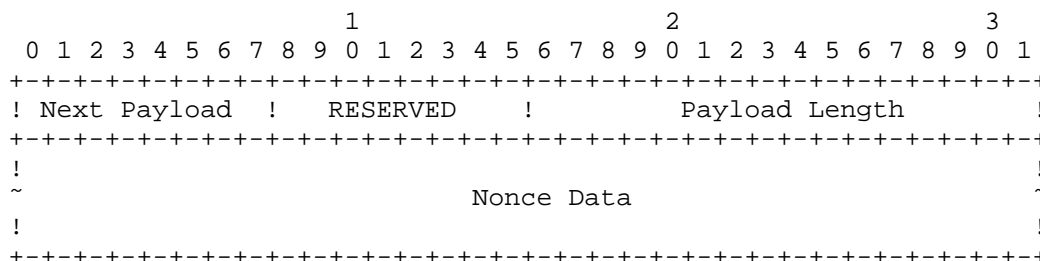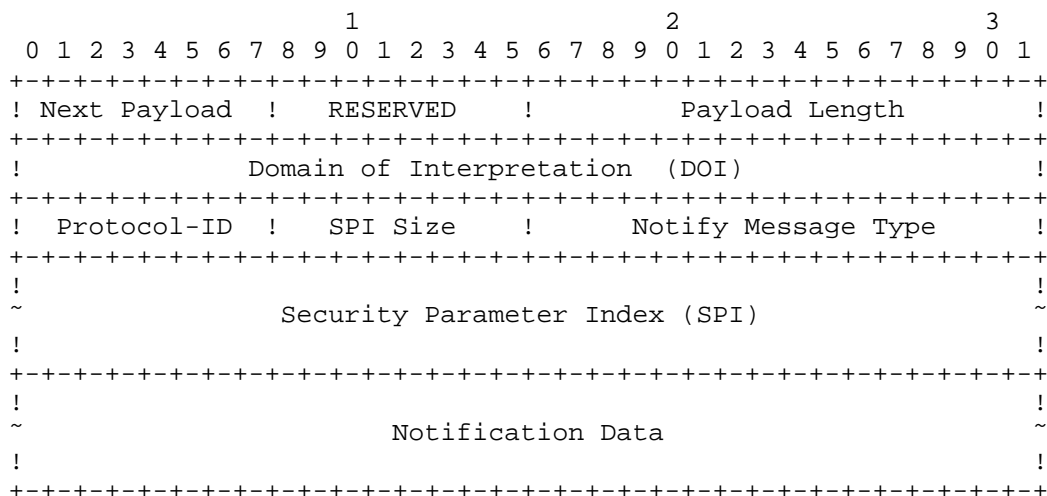
Figure 15: Notification Payload Format

The Notification Payload fields are defined as follows:

 o Next Payload (1 octet) - Identifier for the payload type of the
    next payload in the message. If the current payload is the last
    in the message, then this field will be 0.

 o RESERVED (1 octet) - Unused, set to 0.

 o Payload Length (2 octets) - Length in octets of the current
    payload, including the generic payload header.

 o Domain of Interpretation (4 octets) - Identifies the DOI (as
    described in Section 2.1) under which this notification is taking
    place. For ISAKMP this value is zero (0) and for the IPSEC DOI
    it is one (1). Other DOI's can be defined using the description
    in appendix B.

 o Protocol-Id (1 octet) - Specifies the protocol identifier for the
    current notification. Examples might include ISAKMP, IPSEC ESP,
    IPSEC AH, OSPF, TLS, etc.

    o SPI Size (1 octet) - Length in octets of the SPI as defined by
       the Protocol-Id. In the case of ISAKMP, the Initiator and
       Responder cookie pair from the ISAKMP Header is the ISAKMP SPI,
       therefore, the SPI Size is irrelevant and MAY be from zero (0) to
       sixteen (16). If the SPI Size is non-zero, the content of the
       SPI field MUST be ignored. The Domain of Interpretation (DOI)
       will dictate the SPI Size for other protocols.

    o Notify Message Type (2 octets) - Specifies the type of
       notification message (see section 3.14.1). Additional text, if
       specified by the DOI, is placed in the Notification Data field.

    o SPI (variable length) - Security Parameter Index. The receiving
       entity's SPI. The use of the SPI field is described in section
       2.4. The length of this field is determined by the SPI Size
       field and is not necessarily aligned to a 4 octet boundary.

    o Notification Data (variable length) - Informational or error data
       transmitted in addition to the Notify Message Type. Values for
       this field are DOI-specific.

   The payload type for the Notification Payload is eleven (11).

3.14.1 Notify Message Types

   Notification information can be error messages specifying why an 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
   Notify message to synchronize SA communication. The table below
   lists the Nofitication messages and their corresponding values.
   Values in the Private Use range are expected to be DOI-specific
   values.

                      NOTIFY MESSAGES - ERROR TYPES

                      Errors   Value
                INVALID-PAYLOAD-TYPE   1
                DOI-NOT-SUPPORTED   2
                SITUATION-NOT-SUPPORTED   3
                INVALID-COOKIE   4
                INVALID-MAJOR-VERSION   5
                INVALID-MINOR-VERSION   6
                INVALID-EXCHANGE-TYPE   7
                INVALID-FLAGS   8
                INVALID-MESSAGE-ID   9
                INVALID-PROTOCOL-ID   10
                INVALID-SPI   11

```
                INVALID-TRANSFORM-ID              12
                ATTRIBUTES-NOT-SUPPORTED          13
                NO-PROPOSAL-CHOSEN                14
                BAD-PROPOSAL-SYNTAX               15
                PAYLOAD-MALFORMED                 16
                INVALID-KEY-INFORMATION           17
                INVALID-ID-INFORMATION            18
                INVALID-CERT-ENCODING             19
                INVALID-CERTIFICATE               20
                CERT-TYPE-UNSUPPORTED             21
                INVALID-CERT-AUTHORITY            22
                INVALID-HASH-INFORMATION          23
                AUTHENTICATION-FAILED             24
                INVALID-SIGNATURE                 25
                ADDRESS-NOTIFICATION              26
                NOTIFY-SA-LIFETIME                27
                CERTIFICATE-UNAVAILABLE           28
                UNSUPPORTED-EXCHANGE-TYPE         29
                UNEQUAL-PAYLOAD-LENGTHS           30
                RESERVED (Future Use)        31 - 8191
                Private Use                8192 - 16383


                   NOTIFY MESSAGES - STATUS TYPES
                       Status                 Value
                 CONNECTED                     16384
                 RESERVED (Future Use)  16385 - 24575
                 DOI-specific codes     24576 - 32767
                 Private Use            32768 - 40959
                 RESERVED (Future Use)  40960 - 65535
```

3.15 Delete Payload

   The Delete Payload contains a protocol-specific security association
   identifier that the sender has removed from its security association
   database and is, therefore, no longer valid. Figure 16 shows the
   format of the Delete Payload. It is possible to send multiple SPIs
   in a Delete payload, however, each SPI MUST be for the same protocol.
   Mixing of Protocol Identifiers MUST NOT be performed with the Delete
   payload.

   Deletion which is concerned with an ISAKMP SA will contain a
   Protocol-Id of ISAKMP and the SPIs are the initiator and responder
   cookies from the ISAKMP Header. Deletion which is concerned with a
   Protocol SA, such as ESP or AH, will contain the Protocol-Id of that
   protocol (e.g. ESP, AH) and the SPI is the sending entity's SPI(s).

NOTE: The Delete Payload is not a request for the responder to delete
an SA, but an advisory from the initiator to the responder. If the
responder chooses to ignore the message, the next communication from
the responder to the initiator, using that security association, will
fail. A responder is not expected to acknowledge receipt of a Delete
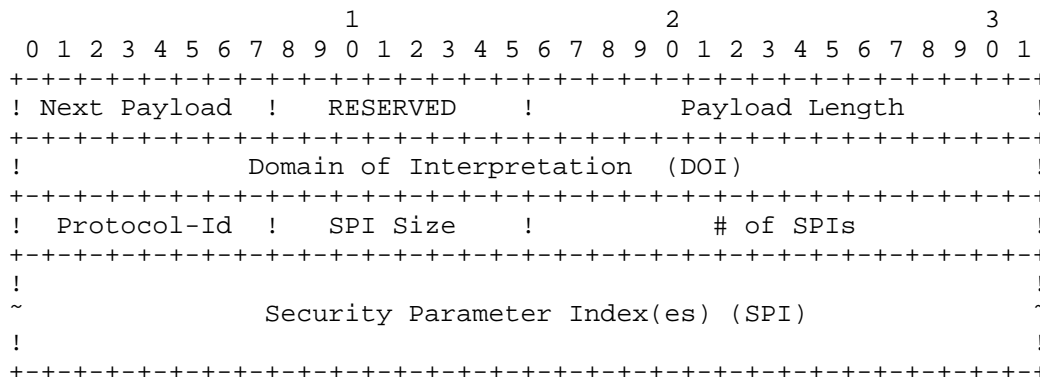payload.

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! Next Payload  !   RESERVED    !         Payload Length        !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !              Domain of Interpretation  (DOI)                  !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !  Protocol-Id  !  SPI Size     !           # of SPIs           !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !                                                               !
 ~              Security Parameter Index(es) (SPI)               ~
 !                                                               !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 16: Delete Payload Format

The Delete Payload fields are defined as follows:

  o Next Payload (1 octet) - Identifier for the payload type of the
     next payload in the message. If the current payload is the last
     in the message, then this field will be 0.

  o RESERVED (1 octet) - Unused, set to 0.

  o Payload Length (2 octets) - Length in octets of the current
     payload, including the generic payload header.

  o Domain of Interpretation (4 octets) - Identifies the DOI (as
     described in Section 2.1) under which this deletion is taking
     place. For ISAKMP this value is zero (0) and for the IPSEC DOI
     it is one (1). Other DOI's can be defined using the description
     in appendix B.

  o Protocol-Id (1 octet) - ISAKMP can establish security
     associations for various protocols, including ISAKMP and IPSEC.
     This field identifies which security association database to
     apply the delete request.

   o SPI Size (1 octet) - Length in octets of the SPI as defined by
      the Protocol-Id. In the case of ISAKMP, the Initiator and
      Responder cookie pair is the ISAKMP SPI. In this case, the SPI
      Size would be 16 octets for each SPI being deleted.

   o # of SPIs (2 octets) - The number of SPIs contained in the Delete
      payload. The size of each SPI is defined by the SPI Size field.

   o Security Parameter Index(es) (variable length) - Identifies the
      specific security association(s) to delete. Values for this
      field are DOI and protocol specific. The length of this field is
      determined by the SPI Size and # of SPIs fields.

   The payload type for the Delete Payload is twelve (12).

3.16 Vendor ID Payload

   The Vendor ID Payload contains a vendor defined constant. The
   constant is used by vendors to identify and recognize remote
   instances of their implementations. This mechanism allows a vendor
   to experiment with new features while maintaining backwards
   compatibility. This is not a general extension facility of ISAKMP.
   Figure 17 shows the format of the Vendor ID Payload.

   The Vendor ID payload is not an announcement from the sender that it
   will send private payload types. A vendor sending the Vendor ID MUST
   not make any assumptions about private payloads that it may send
   unless a Vendor ID is received as well. Multiple Vendor ID payloads
   MAY be sent. An implementation is NOT REQUIRED to understand any
   Vendor ID payloads. An implementation is NOT REQUIRED to send any
   Vendor ID payload at all. If a private payload was sent without
   prior agreement to send it, a compliant implementation may reject a
   proposal with a notify message of type INVALID-PAYLOAD-TYPE.

   If a Vendor ID payload is sent, it MUST be sent during the Phase 1
   negotiation. Reception of a familiar Vendor ID payload in the Phase
   1 negotiation allows an implementation to make use of Private USE
   payload numbers (128-255), described in section 3.1 for vendor
   specific extensions during Phase 2 negotiations. The definition of
   "familiar" is left to implementations to determine. Some vendors may
   wish to implement another vendor's extension prior to
   standardization. However, this practice SHOULD not be widespread and
   vendors should work towards standardization instead.

   The vendor defined constant MUST be unique. The choice of hash and
   text to hash is left to the vendor to decide. As an example, vendors
   could generate their vendor id by taking a plain (non-keyed) hash of
   a string containing the product name, and the version of the product.

A hash is used instead of a vendor registry to avoid local
cryptographic policy problems with having a list of "approved"
products, to keep away from maintaining a list of vendors, and to
allow classified products to avoid having to appear on any list. For
instance:

"Example Company IPsec. Version 97.1"

(not including the quotes) has MD5 hash:
48544f9b1fe662af98b9b39e50c01a5a, when using MD5file. Vendors may
include all of the hash, or just a portion of it, as the payload
length will bound the data. There are no security implications of
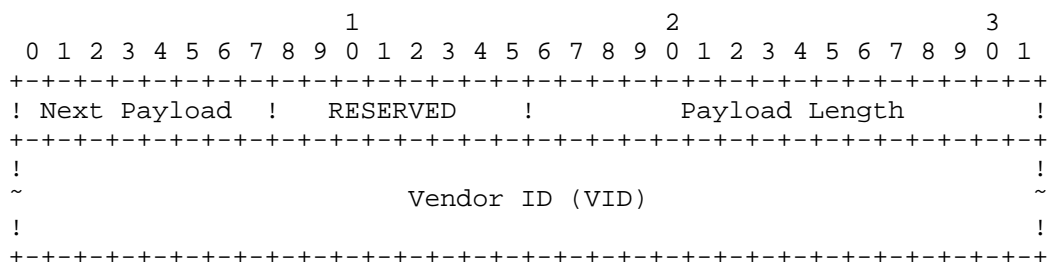this hash, so its choice is arbitrary.

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 ! Next Payload  !   RESERVED    !         Payload Length        !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 !                                                               !
 ~                     Vendor ID (VID)                           ~
 !                                                               !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                  Figure 17: Vendor ID Payload Format

The Vendor ID Payload fields are defined as follows:

  o Next Payload (1 octet) - Identifier for the payload type of the
     next payload in the message. If the current payload is the last
     in the message, then this field will be 0.

  o RESERVED (1 octet) - Unused, set to 0.

  o Payload Length (2 octets) - Length in octets of the current
     payload, including the generic payload header.

  o Vendor ID (variable length) - Hash of the vendor string plus
     version (as described above).

The payload type for the Vendor ID Payload is thirteen (13).

4 ISAKMP Exchanges

  ISAKMP supplies the basic syntax of a message exchange. The basic
  building blocks for ISAKMP messages are the payload types described
  in section 3. This section describes the procedures for SA

establishment and SA modification, followed by a default set of
exchanges that MAY be used for initial interoperability. Other
exchanges will be defined depending on the DOI and key exchange.
[IPDOI] and [IKE] are examples of how this is achieved. Appendix B
explains the procedures for accomplishing these additions.

4.1 ISAKMP Exchange Types

   ISAKMP allows the creation of exchanges for the establishment of
   Security Associations and keying material. There are currently five
   default Exchange Types defined for ISAKMP. Sections 4.4 through 4.8
   describe these exchanges. Exchanges define the content and ordering
   of ISAKMP messages during communications between peers. Most
   exchanges will include all the basic payload types - SA, KE, ID, SIG
   - and may include others. The primary difference between exchange
   types is the ordering of the messages and the payload ordering within
   each message. While the ordering of payloads within messages is not
   mandated, for processing efficiency it is RECOMMENDED that the
   Security Association payload be the first payload within an exchange.
   Processing of each payload within an exchange is described in section
   5.

   Sections 4.4 through 4.8 provide a default set of ISAKMP exchanges.
   These exchanges provide different security protection for the
   exchange itself and information exchanged. The diagrams in each of
   the following sections show the message ordering for each exchange
   type as well as the payloads included in each message, and provide
   basic notes describing what has happened after each message exchange.
   None of the examples include any "optional payloads", like
   certificate and certificate request. Additionally, none of the
   examples include an initial exchange of ISAKMP Headers (containing
   initiator and responder cookies) which would provide protection
   against clogging (see section 2.5.3).

   The defined exchanges are not meant to satisfy all DOI and key
   exchange protocol requirements. If the defined exchanges meet the
   DOI requirements, then they can be used as outlined. If the defined
   exchanges do not meet the security requirements defined by the DOI,
   then the DOI MUST specify new exchange type(s) and the valid
   sequences of payloads that make up a successful exchange, and how to
   build and interpret those payloads. All ISAKMP implementations MUST
   implement the Informational Exchange and SHOULD implement the other
   four exchanges. However, this is dependent on the definition of the
   DOI and associated key exchange protocols.

As discussed above, these exchange types can be used in either phase
of negotiation. However, they may provide different security
properties in each of the phases. With each of these exchanges, the
combination of cookies and SPI fields identifies whether this
exchange is being used in the first or second phase of a negotiation.

4.1.1 Notation

The following notation is used to describe the ISAKMP exchange types,
shown in the next section, with the message formats and associated
payloads:

   HDR is an ISAKMP header whose exchange type defines the payload
        orderings
   SA is an SA negotiation payload with one or more Proposal and
        Transform payloads. An initiator MAY provide multiple proposals
        for negotiation; a responder MUST reply with only one.
   KE is the key exchange payload.
   IDx is the identity payload for "x". x can be: "ii" or "ir"
        for the ISAKMP initiator and responder, respectively, or x can
        be: "ui", "ur" (when the ISAKMP daemon is a proxy negotiator),
        for the user initiator and responder, respectively.
   HASH is the hash payload.
   SIG is the signature payload. The data to sign is exchange-specific.
   AUTH is a generic authentication mechanism, such as HASH or SIG.
   NONCE is the nonce payload.
   '*' signifies payload encryption after the ISAKMP header. This
        encryption MUST begin immediately after the ISAKMP header and
        all payloads following the ISAKMP header MUST be encrypted.

   => signifies "initiator to responder" communication
   <= signifies "responder to initiator" communication

4.2 Security Association Establishment

   The Security Association, Proposal, and Transform payloads are used
   to build ISAKMP messages for the negotiation and establishment of
   SAs. An SA establishment message consists of a single SA payload
   followed by at least one, and possibly many, Proposal payloads and at
   least one, and possibly many, Transform payloads associated with each
   Proposal payload. Because these payloads are considered together,
   the SA payload will point to any following payloads and not to the
   Proposal payload included with the SA payload. The SA Payload
   contains the DOI and Situation for the proposed SA. Each Proposal
   payload contains a Security Parameter Index (SPI) and ensures that
   the SPI is associated with the Protocol-Id in accordance with the
   Internet Security Architecture [SEC-ARCH]. Proposal payloads may or
   may not have the same SPI, as this is implementation dependent. Each

Transform Payload contains the specific security mechanisms to be
used for the designated protocol. It is expected that the Proposal
and Transform payloads will be used only during SA establishment
negotiation. The creation of payloads for security association
negotiation and establishment described here in this section are
applicable for all ISAKMP exchanges described later in sections 4.4
through 4.8. The examples shown in 4.2.1 contain only the SA,
Proposal, and Transform payloads and do not contain other payloads
that might exist for a given ISAKMP exchange.

The Proposal payload provides the initiating entity with the
capability to present to the responding entity the security protocols
and associated security mechanisms for use with the security
association being negotiated. If the SA establishment negotiation is
for a combined protection suite consisting of multiple protocols,
then there MUST be multiple Proposal payloads each with the same
Proposal number. These proposals MUST be considered as a unit and
MUST NOT be separated by a proposal with a different proposal number.
The use of the same Proposal number in multiple Proposal payloads
provides a logical AND operation, i.e. Protocol 1 AND Protocol 2.
The first example below shows an ESP AND AH protection suite. If the
SA establishment negotiation is for different protection suites, then
there MUST be multiple Proposal payloads each with a monotonically
increasing Proposal number. The different proposals MUST be
presented in the initiator's preference order. The use of different
Proposal numbers in multiple Proposal payloads provides a logical OR
operation, i.e. Proposal 1 OR Proposal 2, where each proposal may
have more than one protocol. The second example below shows either
an AH AND ESP protection suite OR just an ESP protection suite. Note
that the Next Payload field of the Proposal payload points to another
Proposal payload (if it exists). The existence of a Proposal payload
implies the existence of one or more Transform payloads.

The Transform payload provides the initiating entity with the
capability to present to the responding entity multiple mechanisms,
or transforms, for a given protocol. The Proposal payload identifies
a Protocol for which services and mechanisms are being negotiated.
The Transform payload allows the initiating entity to present several
possible supported transforms for that proposed protocol. There may
be several transforms associated with a specific Proposal payload
each identified in a separate Transform payload. The multiple
transforms MUST be presented with monotonically increasing numbers in
the initiator's preference order. The receiving entity MUST select a
single transform for each protocol in a proposal or reject the entire
proposal. The use of the Transform number in multiple Transform
payloads provides a second level OR operation, i.e. Transform 1 OR
Transform 2 OR Transform 3. Example 1 below shows two possible
transforms for ESP and a single transform for AH. Example 2 below

shows one transform for AH AND one transform for ESP OR two
transforms for ESP alone. Note that the Next Payload field of the
Transform payload points to another Transform payload or 0. The
Proposal payload delineates the different proposals.

When responding to a Security Association payload, the responder MUST
send a Security Association payload with the selected proposal, which
may consist of multiple Proposal payloads and their associated
Transform payloads. Each of the Proposal payloads MUST contain a
single Transform payload associated with the Protocol. The responder
SHOULD retain the Proposal # field in the Proposal payload and the
Transform # field in each Transform payload of the selected Proposal.
Retention of Proposal and Transform numbers should speed the
initiator's protocol processing by negating the need to compare the
respondor's selection with every offered option. These values enable
the initiator to perform the comparison directly and quickly. The
initiator MUST verify that the Security Association payload received
from the responder matches one of the proposals sent initially.

4.2.1 Security Association Establishment Examples

This example shows a Proposal for a combined protection suite with
two different protocols. The first protocol is presented with two
transforms supported by the proposer. The second protocol is
presented with a single transform. An example for this proposal
might be: Protocol 1 is ESP with Transform 1 as 3DES and Transform 2
as DES AND Protocol 2 is AH with Transform 1 as SHA. The responder
MUST select from the two transforms proposed for ESP. The resulting
protection suite will be either (1) 3DES AND SHA OR (2) DES AND SHA,
depending on which ESP transform was selected by the responder. Note
this example is shown using the Base Exchange.

```
                               1                   2                   3
          0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       /+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      / ! NP = Nonce    !   RESERVED    !        Payload Length         !
     /  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
SA Pay !                  Domain of Interpretation (DOI)               !
     \  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      \ !                            Situation                         !
       >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      / ! NP = Proposal !   RESERVED    !        Payload Length         !
     /  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Prop 1 ! Proposal # = 1! Protocol-Id   !    SPI Size   !# of Trans. = 2!
Prot 1 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     \ !                          SPI (variable)                       !
      >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      / ! NP = Transform!   RESERVED    !        Payload Length         !
```

```
      /  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Tran 1 ! Transform # 1 ! Transform ID  !           RESERVED2           !
      \  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       \ !                        SA Attributes                        !
       >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      / ! NP = 0       !  RESERVED    !          Payload Length         !
      /  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Tran 2 ! Transform # 2 ! Transform ID  !           RESERVED2           !
      \  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       \ !                        SA Attributes                        !
       >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      / ! NP = 0       !  RESERVED    !          Payload Length         !
      /  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Prop 1 ! Proposal # = 1!  Protocol ID  !   SPI Size   !# of Trans. = 1!
Prot 2 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      \ !                         SPI (variable)                       !
       >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      / ! NP = 0       !  RESERVED    !          Payload Length         !
      /  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Tran 1 ! Transform # 1 ! Transform ID  !           RESERVED2           !
      \  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       \ !                        SA Attributes                        !
       \+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   This second example shows a Proposal for two different protection
   suites. The SA Payload was omitted for space reasons. The first
   protection suite is presented with one transform for the first
   protocol and one transform for the second protocol. The second
   protection suite is presented with two transforms for a single
   protocol. An example for this proposal might be: Proposal 1 with
   Protocol 1 as AH with Transform 1 as MD5 AND Protocol 2 as ESP with
   Transform 1 as 3DES. This is followed by Proposal 2 with Protocol 1
   as ESP with Transform 1 as DES and Transform 2 as 3DES. The responder
   MUST select from the two different proposals. If the second Proposal
   is selected, the responder MUST select from the two transforms for
   ESP. The resulting protection suite will be either (1) MD5 AND 3DES
   OR the selection between (2) DES OR (3) 3DES.

```
                      1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      /+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     / ! NP = Proposal !  RESERVED    !          Payload Length         !
      / +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Prop 1 ! Proposal # = 1!  Protocol ID  !   SPI Size   !# of Trans. = 1!
Prot 1 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      \ !                         SPI (variable)                       !
       >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      / ! NP = 0       !  RESERVED    !          Payload Length         !
```

```
   /   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Tran 1 ! Transform # 1 ! Transform ID  !             RESERVED2         !
   \   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    \ !                         SA Attributes                          !
     >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    / ! NP = Proposal !   RESERVED   !         Payload Length          !
   /   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Prop 1 ! Proposal # = 1! Protocol ID  !    SPI Size   !# of Trans. = 1!
Prot 2 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   \ !                         SPI (variable)                          !
    >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    / ! NP = 0        !   RESERVED   !         Payload Length          !
   /   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Tran 1 ! Transform # 1 ! Transform ID  !             RESERVED2         !
   \   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    \ !                         SA Attributes                          !
     >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    / ! NP = 0        !   RESERVED   !         Payload Length          !
   /   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Prop 2 ! Proposal # = 2! Protocol ID  !    SPI Size   !# of Trans. = 2!
Prot 1 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   \ !                         SPI (variable)                          !
    >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    / ! NP = Transform!   RESERVED   !         Payload Length          !
   /   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Tran 1 ! Transform # 1 ! Transform ID  !             RESERVED2         !
   \   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    \ !                         SA Attributes                          !
     >+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    / ! NP = 0        !   RESERVED   !         Payload Length          !
   /   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Tran 2 ! Transform # 2 ! Transform ID  !             RESERVED2         !
   \   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    \ !                         SA Attributes                          !
     \+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.3 Security Association Modification

   Security Association modification within ISAKMP is accomplished by
   creating a new SA and initiating communications using that new SA.
   Deletion of the old SA can be done anytime after the new SA is
   established. Deletion of the old SA is dependent on local security
   policy. Modification of SAs by using a "Create New SA followed by
   Delete Old SA" method is done to avoid potential vulnerabilities in
   synchronizing modification of existing SA attributes. The procedure
   for creating new SAs is outlined in section 4.2. The procedure for
   deleting SAs is outlined in section 5.15.

Modification of an ISAKMP SA (phase 1 negotiation) follows the same
procedure as creation of an ISAKMP SA. There is no relationship
between the two SAs and the initiator and responder cookie pairs
SHOULD be different, as outlined in section 2.5.3.

Modification of a Protocol SA (phase 2 negotiation) follows the same
procedure as creation of a Protocol SA. The creation of a new SA is
protected by the existing ISAKMP SA. There is no relationship between
the two Protocol SAs. A protocol implementation SHOULD begin using
the newly created SA for outbound traffic and SHOULD continue to
support incoming traffic on the old SA until it is deleted or until
traffic is received under the protection of the newly created SA. As
stated previously in this section, deletion of an old SA is then
dependent on local security policy.

4.4 Base Exchange

The Base Exchange is designed to allow the Key Exchange and
Authentication related information to be transmitted together.
Combining the Key Exchange and Authentication-related information
into one message reduces the number of round-trips at the expense of
not providing identity protection. Identity protection is not
provided because identities are exchanged before a common shared
secret has been established and, therefore, encryption of the
identities is not possible. The following diagram shows the messages
with the possible payloads sent in each message and notes for an
example of the Base Exchange.

```
                        BASE EXCHANGE

 #  Initiator Direction  Responder             NOTE
(1)  HDR; SA; NONCE  =>              Begin ISAKMP-SA or Proxy negotiation

(2)                    <=  HDR; SA; NONCE
                               Basic SA agreed upon
(3)  HDR; KE;         =>
     IDii; AUTH                    Key Generated (by responder)
                                   Initiator Identity Verified by
                                   Responder
(4)                    <=  HDR; KE;
                           IDir; AUTH
                                   Responder Identity Verified by
                                   Initiator Key Generated (by
                                   initiator) SA established
```

   In the first message (1), the initiator generates a proposal it
   considers adequate to protect traffic for the given situation. The
   Security Association, Proposal, and Transform payloads are included
   in the Security Association payload (for notation purposes). Random
   information which is used to guarantee liveness and protect against
   replay attacks is also transmitted. Random information provided by
   both parties SHOULD be used by the authentication mechanism to
   provide shared proof of participation in the exchange.

   In the second message (2), the responder indicates the protection
   suite it has accepted with the Security Association, Proposal, and
   Transform payloads. Again, random information which is used to
   guarantee liveness and protect against replay attacks is also
   transmitted. Random information provided by both parties SHOULD be
   used by the authentication mechanism to provide shared proof of
   participation in the exchange. Local security policy dictates the
   action of the responder if no proposed protection suite is accepted.
   One possible action is the transmission of a Notify payload as part
   of an Informational Exchange.

   In the third (3) and fourth (4) messages, the initiator and
   responder, respectively, exchange keying material used to arrive at a
   common shared secret and identification information. This
   information is transmitted under the protection of the agreed upon
   authentication function. Local security policy dictates the action
   if an error occurs during these messages. One possible action is the
   transmission of a Notify payload as part of an Informational
   Exchange.

4.5 Identity Protection Exchange

   The Identity Protection Exchange is designed to separate the Key
   Exchange information from the Identity and Authentication related
   information. Separating the Key Exchange from the Identity and
   Authentication related information provides protection of the
   communicating identities at the expense of two additional messages.
   Identities are exchanged under the protection of a previously
   established common shared secret. The following diagram shows the
   messages with the possible payloads sent in each message and notes
   for an example of the Identity Protection Exchange.

                          IDENTITY PROTECTION EXCHANGE

```
 #      Initiator       Direction     Responder         NOTE
(1)  HDR; SA                =>                           Begin ISAKMP-SA or
                                                         Proxy negotiation
(2)                         <=       HDR; SA
                                                         Basic SA agreed upon
(3)  HDR; KE; NONCE         =>
(4)                         <=       HDR; KE; NONCE
                                                         Key Generated (by
                                                         Initiator and
                                                         Responder)
(5)  HDR*; IDii; AUTH       =>
                                                         Initiator Identity
                                                         Verified by
                                                         Responder
(6)                         <=       HDR*; IDir; AUTH
                                                         Responder Identity
                                                         Verified by
                                                         Initiator
                                                         SA established
```

   In the first message (1), the initiator generates a proposal it
   considers adequate to protect traffic for the given situation. The
   Security Association, Proposal, and Transform payloads are included
   in the Security Association payload (for notation purposes).

   In the second message (2), the responder indicates the protection
   suite it has accepted with the Security Association, Proposal, and
   Transform payloads. Local security policy dictates the action of the
   responder if no proposed protection suite is accepted. One possible
   action is the transmission of a Notify payload as part of an
   Informational Exchange.

   In the third (3) and fourth (4) messages, the initiator and
   responder, respectively, exchange keying material used to arrive at a
   common shared secret and random information which is used to
   guarantee liveness and protect against replay attacks. Random
   information provided by both parties SHOULD be used by the
   authentication mechanism to provide shared proof of participation in
   the exchange. Local security policy dictates the action if an error
   occurs during these messages. One possible action is the
   transmission of a Notify payload as part of an Informational
   Exchange.

   In the fifth (5) and sixth (6) messages, the initiator and responder,
   respectively, exchange identification information and the results of
   the agreed upon authentication function. This information is

transmitted under the protection of the common shared secret. Local
security policy dictates the action if an error occurs during these
messages. One possible action is the transmission of a Notify
payload as part of an Informational Exchange.

4.6 Authentication Only Exchange

The Authentication Only Exchange is designed to allow only
Authentication related information to be transmitted. The benefit of
this exchange is the ability to perform only authentication without
the computational expense of computing keys. Using this exchange
during negotiation, none of the transmitted information will be
encrypted. However, the information may be encrypted in other
places. For example, if encryption is negotiated during the first
phase of a negotiation and the authentication only exchange is used
in the second phase of a negotiation, then the authentication only
exchange will be encrypted by the ISAKMP SAs negotiated in the first
phase. The following diagram shows the messages with possible
payloads sent in each message and notes for an example of the
Authentication Only Exchange.

```
                     AUTHENTICATION ONLY EXCHANGE

 #      Initiator      Direction       Responder       NOTE
(1)  HDR; SA; NONCE       =>                           Begin ISAKMP-SA or
                                                       Proxy negotiation
(2)                       <=      HDR; SA; NONCE;
                                  IDir; AUTH

                                                       Basic SA agreed upon
                                                       Responder Identity
                                                       Verified by Initiator
(3)  HDR; IDii; AUTH      =>

                                                       Initiator Identity
                                                       Verified by Responder
                                                       SA established
```

In the first message (1), the initiator generates a proposal it
considers adequate to protect traffic for the given situation. The
Security Association, Proposal, and Transform payloads are included
in the Security Association payload (for notation purposes). Random
information which is used to guarantee liveness and protect against
replay attacks is also transmitted. Random information provided by
both parties SHOULD be used by the authentication mechanism to
provide shared proof of participation in the exchange.

In the second message (2), the responder indicates the protection
suite it has accepted with the Security Association, Proposal, and
Transform payloads. Again, random information which is used to

guarantee liveness and protect against replay attacks is also
transmitted. Random information provided by both parties SHOULD be
used by the authentication mechanism to provide shared proof of
participation in the exchange. Additionally, the responder transmits
identification information. All of this information is transmitted
under the protection of the agreed upon authentication function.
Local security policy dictates the action of the responder if no
proposed protection suite is accepted. One possible action is the
transmission of a Notify payload as part of an Informational
Exchange.

In the third message (3), the initiator transmits identification
information. This information is transmitted under the protection of
the agreed upon authentication function. Local security policy
dictates the action if an error occurs during these messages. One
possible action is the transmission of a Notify payload as part of an
Informational Exchange.

4.7 Aggressive Exchange

The Aggressive Exchange is designed to allow the Security
Association, Key Exchange and Authentication related payloads to be
transmitted together. Combining the Security Association, Key
Exchange, and Authentication-related information into one message
reduces the number of round-trips at the expense of not providing
identity protection. Identity protection is not provided because
identities are exchanged before a common shared secret has been
established and, therefore, encryption of the identities is not
possible. Additionally, the Aggressive Exchange is attempting to
establish all security relevant information in a single exchange.
The following diagram shows the messages with possible payloads sent
in each message and notes for an example of the Aggressive Exchange.

                        AGGRESSIVE EXCHANGE

```
 #     Initiator   Direction      Responder        NOTE
(1)  HDR; SA; KE;      =>                           Begin ISAKMP-SA or
                                                    Proxy negotiation
     NONCE; IDii                                    and Key Exchange

(2)                    <=      HDR; SA; KE;
                               NONCE; IDir; AUTH
                                                    Initiator Identity
                                                    Verified by Responder
                                                    Key Generated
                                                    Basic SA agreed upon
(3)  HDR*; AUTH        =>
                                                    Responder Identity
                                                    Verified by Initiator
                                                    SA established
```

   In the first message (1), the initiator generates a proposal it
   considers adequate to protect traffic for the given situation. The
   Security Association, Proposal, and Transform payloads are included
   in the Security Association payload (for notation purposes). There
   can be only one Proposal and one Transform offered (i.e. no choices)
   in order for the aggressive exchange to work. Keying material used
   to arrive at a common shared secret and random information which is
   used to guarantee liveness and protect against replay attacks are
   also transmitted. Random information provided by both parties SHOULD
   be used by the authentication mechanism to provide shared proof of
   participation in the exchange. Additionally, the initiator transmits
   identification information.

   In the second message (2), the responder indicates the protection
   suite it has accepted with the Security Association, Proposal, and
   Transform payloads. Keying material used to arrive at a common
   shared secret and random information which is used to guarantee
   liveness and protect against replay attacks is also transmitted.
   Random information provided by both parties SHOULD be used by the
   authentication mechanism to provide shared proof of participation in
   the exchange. Additionally, the responder transmits identification
   information. All of this information is transmitted under the
   protection of the agreed upon authentication function. Local
   security policy dictates the action of the responder if no proposed
   protection suite is accepted. One possible action is the
   transmission of a Notify payload as part of an Informational
   Exchange.

In the third (3) message, the initiator transmits the results of the
agreed upon authentication function. This information is transmitted
under the protection of the common shared secret. Local security
policy dictates the action if an error occurs during these messages.
One possible action is the transmission of a Notify payload as part
of an Informational Exchange.

4.8 Informational Exchange

The Informational Exchange is designed as a one-way transmittal of
information that can be used for security association management.
The following diagram shows the messages with possible payloads sent
in each message and notes for an example of the Informational
Exchange.

                    INFORMATIONAL EXCHANGE

 #   Initiator Direction Responder NOTE
(1)   HDR*; N/D   =>   Error Notification or Deletion

In the first message (1), the initiator or responder transmits an
ISAKMP Notify or Delete payload.

If the Informational Exchange occurs prior to the exchange of keying
meterial during an ISAKMP Phase 1 negotiation, there will be no
protection provided for the Informational Exchange. Once keying
material has been exchanged or an ISAKMP SA has been established, the
Informational Exchange MUST be transmitted under the protection
provided by the keying material or the ISAKMP SA.

All exchanges are similar in that with the beginning of any exchange,
cryptographic synchronization MUST occur. The Informational Exchange
is an exchange and not an ISAKMP message. Thus, the generation of an
Message ID (MID) for an Informational Exchange SHOULD be independent
of IVs of other on-going communication. This will ensure
cryptographic synchronization is maintained for existing
communications and the Informational Exchange will be processed
correctly. The only exception to this is when the Commit Bit of the
ISAKMP Header is set. When the Commit Bit is set, the Message ID
field of the Informational Exchange MUST contain the Message ID of
the original ISAKMP Phase 2 SA negotiation, rather than a new Message
ID (MID). This is done to ensure that the Informational Exchange with
the CONNECTED Notify Message can be associated with the correct Phase
2 SA. For a description of the Commit Bit, see section 3.1.

5 ISAKMP Payload Processing

   Section 3 describes the ISAKMP payloads. These payloads are used in
   the exchanges described in section 4 and can be used in exchanges
   defined for a specific DOI. This section describes the processing for
   each of the payloads. This section suggests the logging of events to
   a system audit file. This action is controlled by a system security
   policy and is, therefore, only a suggested action.

5.1 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. All processing SHOULD include packet length checks
   to insure the packet received is at least as long as the length given
   in the ISAKMP Header. If the ISAKMP message length and the value in
   the Payload Length field of the ISAKMP Header are not the same, then
   the ISAKMP message MUST be rejected. The receiving entity (initiator
   or responder) MUST do the following:

   1. The event, UNEQUAL PAYLOAD LENGTHS, MAY be logged in the
       appropriate system audit file.

   2. An Informational Exchange with a Notification payload containing
       the UNEQUAL-PAYLOAD-LENGTHS message type MAY be sent to the
       transmitting entity. This action is dictated by a system
       security policy.

   When transmitting an ISAKMP message, the transmitting entity
   (initiator or responder) MUST do the following:

   1. Set a timer and initialize a retry counter.

       NOTE: Implementations MUST NOT use a fixed timer. Instead,
       transmission timer values should be adjusted dynamically based on
       measured round trip times. In addition, successive
       retransmissions of the same packet should be separated by
       increasingly longer time intervals (e.g., exponential backoff).

   2. If the timer expires, the ISAKMP message is resent and the retry
       counter is decremented.

   3. If the retry counter reaches zero (0), the event, RETRY LIMIT
       REACHED, MAY be logged in the appropriate system audit file.

   4. The ISAKMP protocol machine clears all states and returns to
       IDLE.

5.2 ISAKMP Header Processing

   When creating an ISAKMP message, the transmitting entity (initiator
   or responder) MUST do the following:

   1. Create the respective cookie. See section 2.5.3 for details.

   2. Determine the relevant security characteristics of the session
       (i.e. DOI and situation).

   3. Construct an ISAKMP Header with fields as described in section
       3.1.

   4. Construct other ISAKMP payloads, depending on the exchange type.

   5. Transmit the message to the destination host as described in
       section5.1.

   When an ISAKMP message is received, the receiving entity (initiator
   or responder) MUST do the following:

   1. Verify 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, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the INVALID-COOKIE message type MAY be sent to
            the transmitting entity. This action is dictated by a
            system security policy.

   2. Check the Next Payload field to confirm it is valid. If the Next
       Payload field validation fails, the message is discarded and the
       following actions are taken:

       (a) The event, INVALID NEXT PAYLOAD, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the INVALID-PAYLOAD-TYPE message type MAY be sent
            to the transmitting entity. This action is dictated by a
            system security policy.

   3. Check the Major and Minor Version fields to confirm they are
       correct (see section 3.1). If the Version field validation
       fails, the message is discarded and the following actions are

taken:

   (a) The event, INVALID ISAKMP VERSION, MAY be logged in the
        appropriate system audit file.

   (b) An Informational Exchange with a Notification payload
        containing the INVALID-MAJOR-VERSION or INVALID-MINOR-
        VERSION message type MAY be sent to the transmitting entity.
        This action is dictated by a system security policy.

4. Check the Exchange Type field to confirm it is valid. If the
    Exchange Type field validation fails, the message is discarded
    and the following actions are taken:

   (a) The event, INVALID EXCHANGE TYPE, MAY be logged in the
        appropriate system audit file.

   (b) An Informational Exchange with a Notification payload
        containing the INVALID-EXCHANGE-TYPE message type MAY be
        sent to the transmitting entity. This action is dictated by
        a system security policy.

5. Check the Flags field to ensure it contains correct values. If
    the Flags field validation fails, the message is discarded and
    the following actions are taken:

   (a) The event, INVALID FLAGS, MAY be logged in the appropriate
        systemaudit file.

   (b) An Informational Exchange with a Notification payload
        containing the INVALID-FLAGS message type MAY be sent to the
        transmitting entity. This action is dictated by a system
        security policy.

6. Check the Message ID field to ensure it contains correct values.
    If the Message ID validation fails, the message is discarded and
    the following actions are taken:

   (a) The event, INVALID MESSAGE ID, MAY be logged in the
        appropriate system audit file.

   (b) An Informational Exchange with a Notification payload
        containing the INVALID-MESSAGE-ID message type MAY be sent
        to the transmitting entity. This action is dictated by a
        system security policy.

7. Processing of the ISAKMP message continues using the value in the
    Next Payload field.

5.3 Generic Payload Header Processing

   When creating any of the ISAKMP Payloads described in sections 3.4
   through 3.15 a Generic Payload Header is placed at the beginning of
   these payloads. When creating the Generic Payload Header, the
   transmitting entity (initiator or responder) MUST do the following:

   1. Place the value of the Next Payload in the Next Payload field.
       These values are described in section 3.1.

   2. Place the value zero (0) in the RESERVED field.

   3. Place the length (in octets) of the payload in the Payload Length
       field.

   4. Construct the payloads as defined in the remainder of this
       section.

   When any of the ISAKMP Payloads are received, the receiving entity
   (initiator or responder) MUST do the following:

   1. Check the Next Payload field to confirm it is valid. If the Next
       Payload field validation fails, the message is discarded and the
       following actions are taken:

       (a) The event, INVALID NEXT PAYLOAD, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the INVALID-PAYLOAD-TYPE message type MAY be sent
            to the transmitting entity. This action is dictated by a
            system security policy.

   2. Verify the RESERVED field contains the value zero. If the value
       in the RESERVED field is not zero, the message is discarded and
       the following actions are taken:

       (a) The event, INVALID RESERVED FIELD, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the BAD-PROPOSAL-SYNTAX or PAYLOAD-MALFORMED
            message type MAY be sent to the transmitting entity. This
            action is dictated by a system security policy.

   3. Process the remaining payloads as defined by the Next Payload
       field.

5.4 Security Association Payload Processing

   When creating a Security Association Payload, the transmitting entity
   (initiator or responder) MUST do the following:

   1. Determine the Domain of Interpretation for which this negotiation
       is being performed.

   2. Determine the situation within the determined DOI for which this
       negotiation is being performed.

   3. Determine the proposal(s) and transform(s) within the situation.
       These are described, respectively, in sections 3.5 and 3.6.

   4. Construct a Security Association payload.

   5. Transmit the message to the receiving entity as described in
       section 5.1.

   When a Security Association payload is received, the receiving entity
   (initiator or responder) MUST do the following:

   1. Determine if the Domain of Interpretation (DOI) is supported. If
       the DOI determination fails, the message is discarded and the
       following actions are taken:

       (a) The event, INVALID DOI, MAY be logged in the appropriate
            system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the DOI-NOT-SUPPORTED message type MAY be sent to
            the transmitting entity. This action is dictated by a
            system security policy.

   2. Determine if the given situation can be protected. If the
       Situation determination fails, the message is discarded and the
       following actions are taken:

       (a) The event, INVALID SITUATION, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the SITUATION-NOT-SUPPORTED message type MAY be
            sent to the transmitting entity. This action is dictated by
            a system security policy.

   3. Process the remaining payloads (i.e. Proposal, Transform) of the
       Security Association Payload. If the Security Association

        Proposal (as described in sections 5.5 and 5.6) is not accepted,
        then the following actions are taken:

        (a) The event, INVALID PROPOSAL, MAY be logged in the
             appropriate system audit file.

        (b) An Informational Exchange with a Notification payload
             containing the NO-PROPOSAL-CHOSEN message type MAY be sent
             to the transmitting entity. This action is dictated by a
             system security policy.

5.5 Proposal Payload Processing

   When creating a Proposal Payload, the transmitting entity (initiator
   or responder) MUST do the following:

   1. Determine the Protocol for this proposal.

   2. Determine the number of proposals to be offered for this protocol
       and the number of transforms for each proposal. Transforms are
       described in section 3.6.

   3. Generate a unique pseudo-random SPI.

   4. Construct a Proposal payload.

   When a Proposal payload is received, the receiving entity (initiator
   or responder) MUST do the following:

   1. Determine if the Protocol is supported. If the Protocol-ID field
       is invalid, the payload is discarded and the following actions
       are taken:

        (a) The event, INVALID PROTOCOL, MAY be logged in the
             appropriate system audit file.

        (b) An Informational Exchange with a Notification payload
             containing the INVALID-PROTOCOL-ID message type MAY be sent
             to the transmitting entity. This action is dictated by a
             system security policy.

   2. Determine if the SPI is valid. If the SPI is invalid, the
       payload is discarded and the following actions are taken:

        (a) The event, INVALID SPI, MAY be logged in the appropriate
             system audit file.

> (b) An Informational Exchange with a Notification payload
>   containing the INVALID-SPI message type MAY be sent to the
>   transmitting entity. This action is dictated by a system
>   security policy.

3. Ensure the Proposals are presented according to the details given
    in section 3.5 and 4.2. If the proposals are not formed
    correctly, the following actions are taken:

    (a) Possible events, BAD PROPOSAL SYNTAX, INVALID PROPOSAL, are
          logged in the appropriate system audit file.

    (b) An Informational Exchange with a Notification payload
          containing the BAD-PROPOSAL-SYNTAX or PAYLOAD-MALFORMED
          message type MAY be sent to the transmitting entity. This
          action is dictated by a system security policy.

4. Process the Proposal and Transform payloads as defined by the
    Next Payload field. Examples of processing these payloads are
    given in section 4.2.1.

5.6 Transform Payload Processing

   When creating a Transform Payload, the transmitting entity (initiator
   or responder) MUST do the following:

1. Determine the Transform # for this transform.

2. Determine the number of transforms to be offered for this
    proposal. Transforms are described in sections 3.6.

3. Construct a Transform payload.

   When a Transform payload is received, the receiving entity (initiator
   or responder) MUST do the following:

1. Determine if the Transform is supported. If the Transform-ID
    field contains an unknown or unsupported value, then that
    Transform payload MUST be ignored and MUST NOT cause the
    generation of an INVALID TRANSFORM event. If the Transform-ID
    field is invalid, the payload is discarded and the following
    actions are taken:

    (a) The event, INVALID TRANSFORM, MAY be logged in the
          appropriate system audit file.

    (b) An Informational Exchange with a Notification payload
          containing the INVALID-TRANSFORM-ID message type MAY be sent

          to the transmitting entity. This action is dictated by a
          system security policy.

     2. Ensure the Transforms are presented according to the details
         given in section 3.6 and 4.2. If the transforms are not formed
         correctly, the following actions are taken:

         (a) Possible events, BAD PROPOSAL SYNTAX, INVALID TRANSFORM,
              INVALID ATTRIBUTES, are logged in the appropriate system
              audit file.

         (b) An Informational Exchange with a Notification payload
              containing the BAD-PROPOSAL-SYNTAX, PAYLOAD-MALFORMED or
              ATTRIBUTES-NOT-SUPPORTED message type MAY be sent to the
              transmitting entity. This action is dictated by a system
              security policy.

     3. Process the subsequent Transform and Proposal payloads as defined
         by the Next Payload field. Examples of processing these payloads
         are given in section 4.2.1.

5.7 Key Exchange Payload Processing

   When creating a Key Exchange Payload, the transmitting entity
   (initiator or responder) MUST do the following:

   1. Determine the Key Exchange to be used as defined by the DOI.

   2. Determine the usage of the Key Exchange Data field as defined by
       the DOI.

   3. Construct a Key Exchange payload.

   4. Transmit the message to the receiving entity as described in
       section 5.1.

   When a Key Exchange payload is received, the receiving entity
   (initiator or responder) MUST do the following:

   1. Determine if the Key Exchange is supported. If the Key Exchange
       determination fails, the message is discarded and the following
       actions are taken:

       (a) The event, INVALID KEY INFORMATION, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the INVALID-KEY-INFORMATION message type MAY be

sent to the transmitting entity. This action is dictated by
a system security policy.

5.8 Identification Payload Processing

When creating an Identification Payload, the transmitting entity
(initiator or responder) MUST do the following:

1. Determine the Identification information to be used as defined by
    the DOI (and possibly the situation).

2. Determine the usage of the Identification Data field as defined
    by the DOI.

3. Construct an Identification payload.

4. Transmit the message to the receiving entity as described in
    section 5.1.

When an Identification payload is received, the receiving entity
(initiator or responder) MUST do the following:

1. Determine if the Identification Type is supported. This may be
    based on the DOI and Situation. If the Identification
    determination fails, the message is discarded and the following
    actions are taken:

    (a) The event, INVALID ID INFORMATION, MAY be logged in the
         appropriate system audit file.

    (b) An Informational Exchange with a Notification payload
         containing the INVALID-ID-INFORMATION message type MAY be
         sent to the transmitting entity. This action is dictated by
         a system security policy.

5.9 Certificate Payload Processing

When creating a Certificate Payload, the transmitting entity
(initiator or responder) MUST do the following:

1. Determine the Certificate Encoding to be used. This may be
    specified by the DOI.

2. Ensure the existence of a certificate formatted as defined by the
    Certificate Encoding.

3. Construct a Certificate payload.

   4. Transmit the message to the receiving entity as described in
      section 5.1.

   When a Certificate payload is received, the receiving entity
   (initiator or responder) MUST do the following:

   1. Determine if the Certificate Encoding is supported. If the
      Certificate Encoding is not supported, the payload is discarded
      and the following actions are taken:

      (a) The event, INVALID CERTIFICATE TYPE, MAY be logged in the
           appropriate system audit file.

      (b) An Informational Exchange with a Notification payload
           containing the INVALID-CERT-ENCODING message type MAY be
           sent to the transmitting entity. This action is dictated by
           a system security policy.

   2. Process the Certificate Data field. If the Certificate Data is
      invalid or improperly formatted, the payload is discarded and the
      following actions are taken:

      (a) The event, INVALID CERTIFICATE, MAY be logged in the
           appropriate system audit file.

      (b) An Informational Exchange with a Notification payload
           containing the INVALID-CERTIFICATE message type MAY be sent
           to the transmitting entity. This action is dictated by a
           system security policy.

5.10 Certificate Request Payload Processing

   When creating a Certificate Request Payload, the transmitting entity
   (initiator or responder) MUST do the following:

   1. Determine the type of Certificate Encoding to be requested. This
      may be specified by the DOI.

   2. Determine the name of an acceptable Certificate Authority which
      is to be requested (if applicable).

   3. Construct a Certificate Request payload.

   4. Transmit the message to the receiving entity as described in
      section 5.1.

   When a Certificate Request payload is received, the receiving entity
   (initiator or responder) MUST do the following:

1. Determine if the Certificate Encoding is supported. If the
    Certificate Encoding is invalid, the payload is discarded and the
    following actions are taken:

    (a) The event, INVALID CERTIFICATE TYPE, MAY be logged in
         the appropriate system audit file.

    (b) An Informational Exchange with a Notification payload
         containing the INVALID-CERT-ENCODING message type MAY be
         sent to the transmitting entity. This action is dictated by
         a system security policy.

    If the Certificate Encoding is not supported, the payload is
    discarded and the following actions are taken:

    (a) The event, CERTIFICATE TYPE UNSUPPORTED, MAY be logged in
         the appropriate system audit file.

    (b) An Informational Exchange with a Notification payload
         containing the CERT-TYPE-UNSUPPORTED message type MAY be
         sent to the transmitting entity. This action is dictated by
         a system security policy.

2. Determine if the Certificate Authority is supported for the
    specified Certificate Encoding. If the Certificate Authority is
    invalid or improperly formatted, the payload is discarded and the
    following actions are taken:

    (a) The event, INVALID CERTIFICATE AUTHORITY, MAY be logged in
         the appropriate system audit file.

    (b) An Informational Exchange with a Notification payload
         containing the INVALID-CERT-AUTHORITY message type MAY be
         sent to the transmitting entity. This action is dictated by
         a system security policy.

3. Process the Certificate Request. If a requested Certificate Type
    with the specified Certificate Authority is not available, then
    the payload is discarded and the following actions are taken:

    (a) The event, CERTIFICATE-UNAVAILABLE, MAY be logged in the
         appropriate system audit file.

    (b) An Informational Exchange with a Notification payload
         containing the CERTIFICATE-UNAVAILABLE message type MAY be
         sent to the transmitting entity. This action is dictated by
         a system security policy.

5.11 Hash Payload Processing

   When creating a Hash Payload, the transmitting entity (initiator or
   responder) MUST do the following:

   1. Determine the Hash function to be used as defined by the SA
       negotiation.

   2. Determine the usage of the Hash Data field as defined by the DOI.

   3. Construct a Hash payload.

   4. Transmit the message to the receiving entity as described in
       section 5.1.

   When a Hash payload is received, the receiving entity (initiator or
   responder) MUST do the following:

   1. Determine if the Hash is supported. If the Hash determination
       fails, the message is discarded and the following actions are
       taken:

       (a) The event, INVALID HASH INFORMATION, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the INVALID-HASH-INFORMATION message type MAY be
            sent to the transmitting entity. This action is dictated by
            a system security policy.

   2. Perform the Hash function as outlined in the DOI and/or Key
       Exchange protocol documents. If the Hash function fails, the
       message is discarded and the following actions are taken:

       (a) The event, INVALID HASH VALUE, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the AUTHENTICATION-FAILED message type MAY be
            sent to the transmitting entity. This action is dictated by
            a system security policy.

5.12 Signature Payload Processing

   When creating a Signature Payload, the transmitting entity (initiator
   or responder) MUST do the following:

   1. Determine the Signature function to be used as defined by the SA
       negotiation.

   2. Determine the usage of the Signature Data field as defined by the
       DOI.

   3. Construct a Signature payload.

   4. Transmit the message to the receiving entity as described in
       section 5.1.

   When a Signature payload is received, the receiving entity (initiator
   or responder) MUST do the following:

   1. Determine if the Signature is supported. If the Signature
       determination fails, the message is discarded and the following
       actions are taken:

       (a) The event, INVALID SIGNATURE INFORMATION, MAY be logged in
            the appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the INVALID-SIGNATURE message type MAY be sent to
            the transmitting entity. This action is dictated by a
            system security policy.

   2. Perform the Signature function as outlined in the DOI and/or Key
       Exchange protocol documents. If the Signature function fails,
       the message is discarded and the following actions are taken:

       (a) The event, INVALID SIGNATURE VALUE, MAY be logged in the
            appropriate system audit file.

       (b) An Informational Exchange with a Notification payload
            containing the AUTHENTICATION-FAILED message type MAY be
            sent to the transmitting entity. This action is dictated by
            a system security policy.

5.13 Nonce Payload Processing

   When creating a Nonce Payload, the transmitting entity (initiator or
   responder) MUST do the following:

   1. Create a unique random value to be used as a nonce.

   2. Construct a Nonce payload.

   3. Transmit the message to the receiving entity as described in
       section 5.1.

   When a Nonce payload is received, the receiving entity (initiator or
   responder) MUST do the following:

   1. There are no specific procedures for handling Nonce payloads.
       The procedures are defined by the exchange types (and possibly
       the DOI and Key Exchange descriptions).

5.14 Notification Payload Processing

   During communications it is possible that errors may occur. The
   Informational Exchange with a Notify Payload provides a controlled
   method of informing a peer entity that errors have occurred during
   protocol processing. It is RECOMMENDED that Notify Payloads be sent
   in a separate Informational Exchange rather than appending a Notify
   Payload to an existing exchange.

   When creating a Notification Payload, the transmitting entity
   (initiator or responder) MUST do the following:

   1. Determine the DOI for this Notification.

   2. Determine the Protocol-ID for this Notification.

   3. Determine the SPI size based on the Protocol-ID field. This
       field is necessary because different security protocols have
       different SPI sizes. For example, ISAKMP combines the Initiator
       and Responder cookie pair (16 octets) as a SPI, while ESP and AH
       have 4 octet SPIs.

   4. Determine the Notify Message Type based on the error or status
       message desired.

   5. Determine the SPI which is associated with this notification.

   6. Determine if additional Notification Data is to be included.
       This is additional information specified by the DOI.

   7. Construct a Notification payload.

   8. Transmit the message to the receiving entity as described in
       section 5.1.

   Because the Informational Exchange with a Notification payload is a
   unidirectional message a retransmission will not be performed. The
   local security policy will dictate the procedures for continuing.

However, we RECOMMEND that a NOTIFICATION PAYLOAD ERROR event be
logged in the appropriate system audit file by the receiving entity.

If the Informational Exchange occurs prior to the exchange of keying
material during an ISAKMP Phase 1 negotiation there will be no
protection provided for the Informational Exchange. Once the keying
material has been exchanged or the ISAKMP SA has been established,
the Informational Exchange MUST be transmitted under the protection
provided by the keying material or the ISAKMP SA.

When a Notification payload is received, the receiving entity
(initiator or responder) MUST do the following:

1. Determine if the Informational Exchange has any protection
    applied to it by checking the Encryption Bit and the
    Authentication Only Bit in the ISAKMP Header. If the Encryption
    Bit is set, i.e. the Informational Exchange is encrypted, then
    the message MUST be decrypted using the (in-progress or
    completed) ISAKMP SA. Once the decryption is complete the
    processing can continue as described below. If the
    Authentication Only Bit is set, then the message MUST be
    authenticated using the (in-progress or completed) ISAKMP SA.
    Once the authentication is completed, the processing can continue
    as described below. If the Informational Exchange is not
    encrypted or authentication, the payload processing can continue
    as described below.

2. Determine if the Domain of Interpretation (DOI) is supported. If
    the DOI determination fails, the payload is discarded and the
    following action is taken:

    (a) The event, INVALID DOI, MAY be logged in the appropriate
         system audit file.

3. Determine if the Protocol-Id is supported. If the Protocol-Id
    determination fails, the payload is discarded and the following
    action is taken:

    (a) The event, INVALID PROTOCOL-ID, MAY be logged in the
         appropriate system audit file.

4. Determine if the SPI is valid. If the SPI is invalid, the
    payload is discarded and the following action is taken:

    (a) The event, INVALID SPI, MAY be logged in the appropriate
         system audit file.

    5. Determine if the Notify Message Type is valid. If the Notify
        Message Type is invalid, the payload is discarded and the
        following action is taken:

        (a) The event, INVALID MESSAGE TYPE, MAY be logged in the
             appropriate system audit file.

    6. Process the Notification payload, including additional
        Notification Data, and take appropriate action, according to
        local security policy.

5.15 Delete Payload Processing

   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 memo. However, if it is discovered that transmissions
   are being compromised, then it is necessary to establish a new SA and
   delete the current SA.

   The Informational Exchange with a Delete Payload provides a
   controlled method of informing a peer entity that the transmitting
   entity has deleted the SA(s). Deletion of Security Associations MUST
   always be performed under the protection of an ISAKMP SA. The
   receiving entity SHOULD clean up its local SA database. However,
   upon receipt of a Delete message the SAs listed in the Security
   Parameter Index (SPI) field of the Delete payload cannot be used with
   the transmitting entity. The SA Establishment procedure must be
   invoked to re-establish secure communications.

   When creating a Delete Payload, the transmitting entity (initiator or
   responder) MUST do the following:

   1. Determine the DOI for this Deletion.

   2. Determine the Protocol-ID for this Deletion.

   3. Determine the SPI size based on the Protocol-ID field. This
       field is necessary because different security protocols have
       different SPI sizes. For example, ISAKMP combines the Initiator
       and Responder cookie pair (16 octets) as a SPI, while ESP and AH
       have 4 octet SPIs.

   4. Determine the # of SPIs to be deleted for this protocol.

   5. Determine the SPI(s) which is (are) associated with this
       deletion.

6. Construct a Delete payload.

7. Transmit the message to the receiving entity as described in
    section 5.1.

Because the Informational Exchange with a Delete payload is a
unidirectional message a retransmission will not be performed. The
local security policy will dictate the procedures for continuing.
However, we RECOMMEND that a DELETE PAYLOAD ERROR event be logged in
the appropriate system audit file by the receiving entity.

As described above, the Informational Exchange with a Delete payload
MUST be transmitted under the protection provided by an ISAKMP SA.

When a Delete payload is received, the receiving entity (initiator or
responder) MUST do the following:

1. Because the Informational Exchange is protected by some security
    service (e.g. authentication for an Auth-Only SA, encryption for
    other exchanges), the message MUST have these security services
    applied using the ISAKMP SA. Once the security service processing
    is complete the processing can continue as described below. Any
    errors that occur during the security service processing will be
    evident when checking information in the Delete payload. The
    local security policy SHOULD dictate any action to be taken as a
    result of security service processing errors.

2. Determine if the Domain of Interpretation (DOI) is supported. If
    the DOI determination fails, the payload is discarded and the
    following action is taken:

    (a) The event, INVALID DOI, MAY be logged in the appropriate
         system audit file.

3. Determine if the Protocol-Id is supported. If the Protocol-Id
    determination fails, the payload is discarded and the following
    action is taken:

    (a) The event, INVALID PROTOCOL-ID, MAY be logged in the
         appropriate system audit file.

4. Determine if the SPI is valid for each SPI included in the Delete
    payload. For each SPI that is invalid, the following action is
    taken:

    (a) The event, INVALID SPI, MAY be logged in the appropriate
         system audit file.

   5. Process the Delete payload and take appropriate action, according
       to local security policy. As described above, one appropriate
       action SHOULD include cleaning up the local SA database.

6 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, security
   requirements, and security mechanisms. 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 Security Association Attributes

A.1 Background/Rationale

   As detailed in previous sections, ISAKMP is designed to provide a
   flexible and extensible framework for establishing and managing
   Security Associations and cryptographic keys. The framework provided
   by ISAKMP consists of header and payload definitions, exchange types
   for guiding message and payload exchanges, and general processing
   guidelines. ISAKMP does not define the mechanisms that will be used
   to establish and manage Security Associations and cryptographic keys
   in an authenticated and confidential manner. The definition of
   mechanisms and their application is the purview of individual Domains
   of Interpretation (DOIs).

   This section describes the ISAKMP values for the Internet IP Security
   DOI, supported security protocols, and identification values for
   ISAKMP Phase 1 negotiations. The Internet IP Security DOI is
   MANDATORY to implement for IP Security. [Oakley] and [IKE] describe,
   in detail, the mechanisms and their application for establishing and
   managing Security Associations and cryptographic keys for IP
   Security.

A.2 Internet IP Security DOI Assigned Value

   As described in [IPDOI], the Internet IP Security DOI Assigned Number
   is one (1).

A.3 Supported Security Protocols

   Values for supported security protocols are specified in the most
   recent "Assigned Numbers" RFC [STD-2]. Presented in the following
   table are the values for the security protocols supported by ISAKMP
   for the Internet IP Security DOI.

                    Protocol Assigned Value
                    RESERVED   0
                    ISAKMP   1

   All DOIs MUST reserve ISAKMP with a Protocol-ID of 1. All other
   security protocols within that DOI will be numbered accordingly.

   Security protocol values 2-15359 are reserved to IANA for future use.
   Values 15360-16383 are permanently reserved for private use amongst
   mutually consenting implementations. Such private use values are
   unlikely to be interoperable across different implementations.

A.4 ISAKMP Identification Type Values

   The following table lists the assigned values for the Identification
   Type field found in the Identification payload during a generic Phase
   1 exchange, which is not for a specific protocol.

```
                   ID Type          Value
              ID_IPV4_ADDR            0
              ID_IPV4_ADDR_SUBNET     1
              ID_IPV6_ADDR            2
              ID_IPV6_ADDR_SUBNET     3
```

A.4.1 ID_IPV4_ADDR

   The ID_IPV4_ADDR type specifies a single four (4) octet IPv4 address.

A.4.2 ID_IPV4_ADDR_SUBNET

   The ID_IPV4_ADDR_SUBNET type specifies a range of IPv4 addresses,
   represented by two four (4) octet values. The first value is an IPv4
   address. The second is an IPv4 network mask. Note that ones (1s) in
   the network mask indicate that the corresponding bit in the address
   is fixed, while zeros (0s) indicate a "wildcard" bit.

A.4.3 ID_IPV6_ADDR

   The ID_IPV6_ADDR type specifies a single sixteen (16) octet IPv6
   address.

A.4.4 ID_IPV6_ADDR_SUBNET

   The ID_IPV6_ADDR_SUBNET type specifies a range of IPv6 addresses,
   represented by two sixteen (16) octet values. The first value is an
   IPv6 address. The second is an IPv6 network mask. Note that ones
   (1s) in the network mask indicate that the corresponding bit in the
   address is fixed, while zeros (0s) indicate a "wildcard" bit.

B Defining a new Domain of Interpretation

   The Internet DOI may be sufficient to meet the security requirements
   of a large portion of the internet community. However, some groups
   may have a need to customize some aspect of a DOI, perhaps to add a
   different set of cryptographic algorithms, or perhaps because they
   want to make their security-relevant decisions based on something
   other than a host id or user id. Also, a particular group may have a
   need for a new exchange type, for example to support key management
   for multicast groups.

   This section discusses guidelines for defining a new DOI. The full
   specification for the Internet DOI can be found in [IPDOI].

   Defining a new DOI is likely to be a time-consuming process. If at
   all possible, it is recommended that the designer begin with an
   existing DOI and customize only the parts that are unacceptable.

   If a designer chooses to start from scratch, the following MUST be
   defined:

     o A "situation": the set of information that will be used to
        determine the required security services.

     o The set of security policies that must be supported.

     o A scheme for naming security-relevant information, including
        encryption algorithms, key exchange algorithms, etc.

     o A syntax for the specification of proposed security services,
        attributes, and certificate authorities.

     o The specific formats of the various payload contents.

     o Additional exchange types, if required.

B.1 Situation

   The situation is the basis for deciding how to protect a
   communications channel. It must contain all of the data that will be
   used to determine the types and strengths of protections applied in
   an SA. For example, a US Department of Defense DOI would probably use
   unpublished algorithms and have additional special attributes to
   negotiate. These additional security attributes would be included in
   the situation.

B.2 Security Policies

   Security policies define how various types of information must be
   categorized and protected. The DOI must define the set of security
   policies supported, because both parties in a negotiation must trust
   that the other party understands a situation, and will protect
   information appropriately, both in transit and in storage. In a
   corporate setting, for example, both parties in a negotiation must
   agree to the meaning of the term "proprietary information" before
   they can negotiate how to protect it.

   Note that including the required security policies in the DOI only
   specifies that the participating hosts understand and implement those
   policies in a full system context.

B.3 Naming Schemes

   Any DOI must define a consistent way to name cryptographic
   algorithms, certificate authorities, etc. This can usually be done
   by using IANA naming conventions, perhaps with some private
   extensions.

B.4 Syntax for Specifying Security Services

   In addition to simply specifying how to name entities, the DOI must
   also specify the format for complete proposals of how to protect
   traffic under a given situation.

B.5 Payload Specification

   The DOI must specify the format of each of the payload types. For
   several of the payload types, ISAKMP has included fields that would
   have to be present across all DOI (such as a certificate authority in
   the certificate payload, or a key exchange identifier in the key
   exchange payload).

B.6 Defining new Exchange Types

   If the basic exchange types are inadequate to meet the requirements
   within a DOI, a designer can define up to thirteen extra exchange
   types per DOI. The designer creates a new exchange type by choosing
   an unused exchange type value, and defining a sequence of messages
   composed of strings of the ISAKMP payload types.

   Note that any new exchange types must be rigorously analyzed for
   vulnerabilities. Since this is an expensive and imprecise
   undertaking, a new exchange type should only be created when
   absolutely necessary.

Security Considerations

   Cryptographic analysis techniques are improving at a steady pace.
   The continuing improvement in processing power makes once
   computationally 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 rely on any UDP information (e.g.
   checksum, length) for its processing.

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

   A number of very important security considerations are presented in
   [SEC-ARCH]. One bears 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 protection provided by the IP Security
   services.

IANA Considerations

   This document contains many "magic" numbers to be maintained by the
   IANA. This section explains the criteria to be used by the IANA to
   assign additional numbers in each of these lists.

Domain of Interpretation

   The Domain of Interpretation (DOI) is a 32-bit field which identifies
   the domain under which the security association negotiation is taking
   place. Requests for assignments of new DOIs must be accompanied by a
   standards-track RFC which describes the specific domain.

Supported Security Protocols

    ISAKMP is designed to provide security association negotiation and
    key management for many security protocols. Requests for identifiers
    for additional security protocols must be accompanied by a
    standards-track RFC which describes the security protocol and its
    relationship to ISAKMP.

Acknowledgements

    Dan Harkins, Dave Carrel, and Derrell Piper of Cisco Systems provided
    design assistance with the protocol and coordination for the [IKE]
    and [IPDOI] documents.

    Hilarie Orman, via the Oakley key exchange protocol, has
    significantly influenced the design of ISAKMP.

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

    Scott Carlson ported the TIS DNSSEC prototype to FreeBSD for use with
    the ISAKMP prototype.

    Jeff Turner and Steve Smalley contributed to the prototype
    development and integration with ESP and AH.

    Mike Oehler and Pete Sell performed interoperability testing with
    other ISAKMP implementors.

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

References

    [ANSI]   ANSI, X9.42: Public Key Cryptography for the Financial
                Services Industry -- Establishment of Symmetric Algorithm
                Keys Using Diffie-Hellman, Working Draft, April 19, 1996.

    [BC]   Ballardie, A., and J. Crowcroft, Multicast-specific
                Security Threats and Countermeasures, Proceedings of 1995
                ISOC Symposium on Networks & Distributed Systems Security,
                pp. 17-30, Internet Society, San Diego, CA, February 1995.

    [Berge]   Berge, N., "UNINETT PCA Policy Statements", RFC 1875,
                December 1995.

   [CW87]   Clark, D.D. and D.R. Wilson, A Comparison of Commercial
               and Military Computer Security Policies, Proceedings of
               the IEEE Symposium on Security & Privacy, Oakland, CA,
               1987, pp. 184-193.

   [DNSSEC]   D. Eastlake III, Domain Name System Protocol Security
               Extensions, Work in Progress.

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

   [IAB]   Bellovin, S., "Report of the IAB Security Architecture
               Workshop", RFC 2316, April 1998.

   [IKE]   Harkins, D., and D. Carrel, "The Internet Key Exchange
               (IKE)", RFC 2409, November 1998.

   [IPDOI]   Piper, D., "The Internet IP Security Domain of
               Interpretation for ISAKMP", RFC 2407, November 1998.

   [Karn]   Karn, P., and B. Simpson, Photuris: Session Key
               Management Protocol, Work in Progress.

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

   [Oakley]   Orman, H., "The Oakley Key Determination Protocol", RFC
               2412, November 1998.

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

   [RFC-1949] Ballardie, A., "Scalable Multicast Key Distribution", RFC
               1949, May 1996.

   [RFC-2093] Harney, H., and C. Muckenhirn, "Group Key Management
               Protocol (GKMP) Specification", RFC 2093, July 1997.

   [RFC-2094] Harney, H., and C. Muckenhirn, "Group Key Management
               Protocol (GKMP) Architecture", RFC 2094, July 1997.

   [RFC-2119] Bradner, S., "Key Words for use in RFCs to Indicate
               Requirement Levels", BCP 14, RFC 2119, March 1997.

   [Schneier] Bruce Schneier, Applied Cryptography - Protocols,
              Algorithms, and Source Code in C (Second Edition), John
              Wiley & Sons, Inc., 1996.

   [SEC-ARCH] Atkinson, R., and S. Kent, "Security Architecture for the
              Internet Protocol", RFC 2401, November 1998.

   [STD-2]   Reynolds, J., and J. Postel, "Assigned Numbers", STD 2, RFC
              1700, October 1994. See also:
              http://www.iana.org/numbers.html

Authors' Addresses

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

   Phone: 301-688-0847
   EMail:wdm@tycho.ncsc.mil

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

   Phone: 301-688-0851
   EMail:mss@tycho.ncsc.mil

   Mark Schertler
   Securify, Inc.
   2415-B Charleston Road
   Mountain View, CA 94043

   Phone: 650-934-9303
   EMail:mjs@securify.com

   Jeff Turner
   RABA Technologies, Inc.
   10500 Little Patuxent Parkway
   Columbia, MD. 21044

   Phone: 410-715-9399
   EMail:jeff.turner@raba.com

Full Copyright Statement