SPARTA, Inc.

H Harney (SPARTA)
A Colegrove (SPARTA)
E Harder (NSA)
U Meth (SPARTA)
R Fleischer (SPARTA)
SPARTA, Inc., National Security Agency

INTERNET-DRAFT
draft-harney-sparta-gsakmp-sec-01.ps
May 2000

Group Secure Association Key Management Protocol

Abstract

The Group Secure Association Key Management Protocol (GSAKMP) provides a security framework for creating cryptographic groups on a network. It provides mechanisms to disseminate group security policy, perform access control based upon PKI certificates, generate group keys, and recover from compromise. This framework addresses group scalability issues by facillitating delegation of process-intensive actions in a secure and controlled manner.

Contents

1 Overview **5**
- 1.1 GSAKMP Overview 5
- 1.2 Assumptions 5
- 1.3 Document Organization 6
- 1.4 References 6

2 Terminology **8**
- 2.1 GSAKMP Terminology 8

3 GROUP LIFE-CYCLE **10**
- 3.1 Group Establishment 10
  - 3.1.1 Create Group Key 11
  - 3.1.2 Distribute Group Key 11
- 3.2 Group Maintenance 14
  - 3.2.1 Member Joins/Leaves 14
  - 3.2.2 Rekey Events 15
- 3.3 Group Removal/Destruction 15

4 Message formats **16**
- 4.1 GSAKMP Header 16
- 4.2 Generic Payload Header 17
- 4.3 Data Attributes Payload 18
- 4.4 Policy Token Payload 19
- 4.5 Key Download Payload 20
  - 4.5.1 GTEK Key Packet 21
  - 4.5.2 Rekey Key Packet 21
- 4.6 Rekey Event Payload 22
- 4.7 Identification/Role Payload 23
- 4.8 Certificate Payload 24
- 4.9 Certificate Request Payload 25
- 4.10 Signature Payload 26
- 4.11 Notification Payload 27
  - 4.11.1 Notify Message Types 28
- 4.12 Vendor ID Payload 28
- 4.13 Key Creation Payload 29

5 GSAKMP State Diagram **31**

6 APPENDIX A – Rekey Packet data format **32**
- 6.1 Rekey Event Header 32
- 6.2 Rekey Event Packet Data(s) 33
- 6.3 Key Pack Data 33
- 6.4 Pack Data Formats 34
  - 6.4.1 GTEK Pack Data 34
  - 6.4.2 LKH Pack Data 34
- 6.5 Example 35

7 Authors Addresses **37**

List of Figures

1 Group Establishment Ladder Diagram . . . 11
2 GSAKMP Header Format . . . 16
3 Generic Payload Header . . . 17
4 Data Attributes Payload . . . 18
5 Policy Token Payload Format . . . 19
6 Key Download Payload Format . . . 20
7 Rekey Event Payload Format . . . 22
8 Identification/Role Payload Format . . . 23
9 Certificate Payload Format . . . 24
10 Certificate Request Payload Format . . . 25
11 Signature Payload Format . . . 26
12 Notification Payload Format . . . 27
13 Vendor ID Payload Format . . . 30
14 Key Creation Payload Format . . . 30
15 GSAKMP State Diagram . . . 31
16 A.1: Rekey Event Header Format . . . 32
17 A.2: Rekey Event Packet Data Format . . . 33
18 A.3: Key Pack Data Format . . . 33

List of Tables

1 Request To Join Message Definition . . . 12
2 Invitation to Join Message Definition . . . 12
3 Invitation Response Message Definition . . . 13
4 Key Download Message Definition . . . 13
5 Acknowledgment Message Definition . . . 14
6 Rekey Event Message Definition . . . 15
7 Group Removal/Destruction Message Definition . . . 15
8 Group Identification Types . . . 16
9 Payload Types . . . 17
10 Exchange Types . . . 18
11 Policy Token Types . . . 19
12 Key Download Data Types . . . 20
13 Rekey Event Types . . . 22
14 Identity/Role Types . . . 23
15 Certificate Payload Types . . . 24
16 Signature Types . . . 26
17 Notify Messages – Status Types . . . 28
18 Notify Messages Types . . . 29
19 Types Of Key Creation Information . . . 30
20 State Transition Events . . . 38

1 Overview

The Group Secure Association Key Management Protocol (GSAKMP) provides symmetric key to groups of users on a network. It provides mechanisms to disseminate group policy, perform access control decisions during group establishment, generate group keys, recover from the compromise of group members, delegate group security functions and destroy the group.

The goals of the GSAKMP are to create a protocol that:

1. Distributes group policy,
2. Provides mechanisms for distributing the group key, and
3. Provides mechanisms for a Rekey of the group.

1.1 GSAKMP Overview

Protecting information requires the definition of a security policy and the enforcement of that policy by capable parties. Control and access to the cryptographic key is the primary mechanism to enforce the access control policy. The GSAKMP provides these mechanisms to control access to the group key.

This document identifies the GSAKMP Message Passing Requirements. The group key(s) are created by the group controller. The group controller must start the group access control, policy enforcement process. The group controller needs to have the access rules defined for joining the group and should be able to identify and verify the permissions of the members to which they will distribute keys.

The potential group members need to have “knowledge” of the access control policy for the group, an unambiguous identification of any party downloading keys to them, and verifiable chain of authority for key download. The group members also need to verify that the key creator is authorized to act in that capacity.

In order to establish a group Secure Association (SA) to support these activities, the identity of each party in the security/access control process must be unambiguously stated/asserted and authenticated to ensure that they are authorized to be a member of the group as defined by the group’s security policy. The security characteristics of the establishment protocol for the SA should include:

1. Coherent permission topology,
2. Group policy,
3. Group policy dissemination,
4. Peer SA to protect data, and
5. Access control checking.

1.2 Assumptions

GSAKMP makes the following assumptions of the underlying host:

1. The operating system can provide the process and data separation services to support software encryption.
2. A separate SA mechanism is present that is sufficient to protect the distribution of the group key.
3. The host and all the applications on that host share the same certificate identities (at least initially).

1.3 Document Organization

Section 1 presents an overview of secure group communications and identifies additional reference documents. Section 2 presents the terminology and concepts used to present the requirements of this protocol. Section 3 describes the group management life-cycle and Section 4 presents the message types and formats used during each phase of the life-cycle. Section 5 presents a discussion of the states encountered in the protocol.

1.4 References

The following references were used in the preparation of this document:

Wallner, D., Harder E., and Agee R., "Key Management for Multicast: Issues and Architectures," Internet Draft, Informational, September 1998.

"Multicast Security Management Protocol (MSMP) Requirements and Policy", SPARTA, October, 1998.

"Logical Key Hierarchy (LKH) Protocol", SPARTA, October, 1998.

[RFC 2093] Harney H., Muckenhirn C., and Rivers T., "Group Key, Management Protocol Specification," RFC 2093, Experimental, July 1997.

[RFC 2094] Harney H., Muckenhirn C., and Rivers T., "Group Key Management Protocol Architecture," RFC 2094, Experimental, July 1997.

[RFC 2408] Maughan D., Schertler M., Schneider M., and Turner J., "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, Proposed Standard, November 1998.

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

[RFC 2412] Orman H. K., "The OAKLEY Key Determination Protocol", RFC 2412, Informational, November 1998.

The Secure Multicast Research Group (SMuG), An Internet Research Task Force Group formed to discuss issues related to multicast security.

[RFC 2402] Kent S. and Atkinson, R., "IP Authentication Header", RFC 2402, November 1998, Proposed Standard.

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

[RFC 2406] Kent S. and Atkinson, R., “IP Encapsulating Security Payload (ESP)“, RFC 2406, November 1998, Proposed Standard.

Balenson D., McGrew D., Sherman A., “Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization”, Internet Draft, February 1999.

Bhattacharya P. and Pereira R., “IPSec Policy Data Model“, Internet Draft, February 1998.

2 Terminology

The following terminology is used throughout the GSAKMP paper.

2.1 GSAKMP Terminology

Group Member: A group member (GM) is any entity with access to the group keys. Regardless of how a member becomes a part of the group or how the group is structured, GMs will perform the following actions:

1. Validate the GC's authorization to perform actions;
2. Accept group keys from the GC;
3. Request group keys from the GC;
4. Maintain local Certificate Revocation Lists (CRLs);
5. Enforce the cooperative group policies as stated in the group policy token;
6. Perform peer review of key management actions; and
7. Manage their local key.

Group Secure Association (GSA): A cryptographic group is a logical association of users or hosts that share cryptographic key(s). This group may be established to support associations between applications or communication protocols.

Group Policy: The group policy completely describes the protection mechanisms and security relevant behaviors of the group. This policy must be commonly understood and enforced by the group for coherent secure operations.

Policy Token/Certificate: The policy token is a mechanism used to disseminate the group policy. The policy token is issued and signed by an authorized source. Each member of the group must verify the token, meet the group join policy , and enforce the policy of the group. The group policy data element will contain a variety of information including:

1. GSAKMP protocol format,
2. Key creation method,
3. Key dissemination policy,
4. Access control policy,
5. Group architecture policy, and
6. Compromise recovery policy.

The policy token layout will be fully presented in the *Group Policy Token Specification* document.

Group Controller: The Group Controller (GC) is a group member with authority to perform any critical protocol actions including:

1. Creating and distributing keys;
2. Maintain the Rekey infrastructure; and
3. Building and maintaining the Rekey arrays.

As the group evolves, it may become desirable to have multiple controllers perform these functions (e.g., Rekey Controller and Group Key Controller).

Subordinate Controller: Any group member, as defined in the group policy, has the capability to act as a Subordinate Controller (SC) thus allowing the group processing and communication requirements to be distributed equitably throughout the network. If the group is structured in such a way, the delegated group members would be identified via the policy token. The SCs may perform actions delegated to them by the GC including:

1. Dissemination of the group key and
2. Management of the status of the local group.

The ease of managing a very large group may also be improved by delegating the creation of subordinate LKH arrays to the SCs. The SCs would have the authority and mechanisms necessary to create and disseminate the LKH arrays for the members under their control. A more detailed discussion of LKH arrays may be found in the Logical Key Hierarchy (LKH) Protocol document.

Peer-to-Peer SA: Peer-to-Peer SA keys can be created by using any number of key generation protocols including the Internet Secure Association Key Management Protocol (ISAKMP)/IPSec and HS/SSL. These protocols rely on cooperative key generation algorithms and on peer review of permissions. Modern SA protocols are specifically developed to support this task. Once the peer-to-peer SA is established, the group protocol can use that SA mechanism for secure confidential peer communications throughout the life of the group.

GSA Keys: GSA keys can be created using strong randomization key generation protocols. These protocols rely on a cooperatively conferred policy. Once the group keys are created and disseminated to the group members, the group protocol can use that SA mechanism for secure confidential group communications throughout the life of the group.

Group Traffic Encryption Key (GTEK): The key or keys created for encrypting the group data.

Logical Key Hierarchy (LKH) array: The group of keys created to facilitate the LKH compromise recovery methodology.

Compromise Recovery: The act of recovering a secure operating state after detecting that a group member cannot be trusted.

3 GROUP LIFE-CYCLE

The management of a cryptographic group follows a life-cycle: group definition, group establishment, group maintenance, and group removal. Each of these life-cycle phases is discussed in the following sections. A cryptographic group is established based on some need for secure communications among a group of individuals. The activities involved in creating a cryptographic group include:

1. Determine Access Policy: Group Join
2. Determine Authorization Policy: Key Dissemination, Computer Trust, and Architecture Authorization
3. Determine Mechanisms: Algorithms and Infrastructure
4. Determine Architecture: Key Dissemination and Compromise Recovery
5. Create Group Policy Token

For the purposes of this document, it is assumed that the group definition activity has occurred and the group information has been broadcast on a key management channel or through a directory service.

3.1 Group Establishment

The Group Establishment Ladder diagram, Figure 1, is presented to illustrate the process of establishing a cryptographic group. The left side of the diagram represents the actions of the GC. The right side of the diagram represents the actions of the GMs. The components of each message shown in the diagram are presented in the Message Definitions sections following the diagram.

Potential GMs may join a group in two ways: by invitation (push) or request (pull). For purposes of illustration, the diagram presents a "Request to Join Group", a "pull", message sent from a potential GM.

At this point, the GC must accept or deny the request. "Process RTJ" indicates a provision for refusing the connection due to some specified reason (e.g., no group, group full, repetitive attempts to join). If the results of "Process RTJ" indicate that the GC should reject the request, the session is terminated.
If the results of Processing the Request To Join indicate that the GC should accept the request, the session continues. The message traffic to an invited potential member also begins at this point on the diagram.

The area of the diagram specified as "Outside GSAKMP" is merely illustrative to show the confidentiality between the GC and GM. It is assumed, for the purposes of this document, that the GC and GM are able to establish a SA using protocols like ISAKMP and IPSec. The GC will specify the security characteristics of the SA to the outside application. The level of protection shall be as good or stronger than the SA characteristics specified in the group policy token. A suggested minimal SA security level is confidentiality with integrity.

To facilitate a well ordered group creation, security policy information must be passed between the GC and the GMs using a group policy token. The group policy token must include the group's address, group permissions, group join policy, group controller identity, group management information, and digital signature of the policy creation authority.

```
          CONTROLLER                  MESSAGE                 MEMBER

                    !<------------Request to Join-------------!
     <Process RTJ>  !                                         !
                    !<============SA ESTABLISHMENT===========>! (Outside GSAKMP)
                    !                                         !
                    !-------------Invitation----------------->!
                    !                                         !<Process Inv.>
                    !<------------Invitation Response-------->!
<Process Inv. Rsp.> !                                         !
                    !-------------Key Download--------------->!
                    !                                         !<Prcs. Key DL>
                    !<------------Acknowledgment--------------!
     <Process ACK>  !                                         !
                    !<=======SHARED KEYED GROUP SESSION======>!
```

Figure 1: Group Establishment Ladder Diagram

3.1.1 Create Group Key

There are two options: key generation at a single point and shared generation. In shared generation, the first member must cooperate with the GC to create the group key. There are several established software-based key creation protocols, including Diffie-Hellman and RSA, that support two group members cooperating to create a cryptographic key. However, for this document, the following discussion presents single-point key generation. Prior to the first member join, the GC will have created the GTEK and the Rekey array.

3.1.2 Distribute Group Key

Potential GMs may join a group and receive the group key in two ways: by invitation (push) or request (pull). The following message definition shows a Request to Join message from a potential GM. The initial message from the GM would contain the following:

1. GSA request and
2. GM Certificate (optional).

The components of a Request to Join message are are shown in Table 1:

The following message definition shows an “Invitation to Join” message from the GC to a potential GM. The initial message from the GC would contain the following:

1. Signed group policy token,
2. GSA request, and

Table 1: Request To Join Message Definition

Message Name	:	Request to Join
Dissection	:	{HDR, Grp ID, GSA RQ} SigM, [CertM]
Payload Types	:	GSAKMP Header, Notification, Signature, [Certificate], [Certificate Request], [Vendor ID], [Identification/Role]
Legend	:	
SigM	:	Signature of Group Member
CertM	:	Certificate of Group Member
{}SigX	:	Indicates minimum fields used in Signature
[]	:	Indicate an optional data item

3. GC Certificate (optional).

The components of an Invitation to Join message are shown in Table 2:

Table 2: Invitation to Join Message Definition

Message Name	:	Invitation
Dissection	:	{HDR, Policy Token, [Key Creation], GSA RQ}SigC, [CertC], [SigSC], [CertSC]
Payload Types	:	GSAKMP Header, Policy Token, Notification, Signature, [Certificate], [Signature], [Certificate], [Key Creation], [Certificate Request], [Vendor ID], [Identification/Role]
Legend	:	
SigC	:	Signature of Group Controller
SigSC	:	Signature of Subordinate Group Controller
CertC	:	Certificate of Group Controller
CertSC	:	Certificate of Subordinate Group Controller
{}SigX	:	Indicates minimum fields used in Signature
[]	:	Indicate an optional data item

For purposes of discussion, this section presents a "Invitation to Join" as presented in Table 2.

The GM will receive this message and process it according to the provisions of Processing the Invitation. The GSA RQ contains the identity of the message source in enough detail to allow the potential member to verify the signature. The GSA RQ also contains the ID of the invited member. In "Process Inv.", the potential GM will initially verify that the signature on the message is authentic. If the message signature does not verify, the session is terminated. GSAKMP sends a properly authenticated message with a Notification Payload of type NACK to indicate termination.

If the message signature is authentic, then the potential GM will look at who signed the message, verify the signer's authorization, and make a decision to proceed. If the potential GM decides not to proceed, the session is terminated. GSAKMP sends a properly authenticated message with a Notification Payload of type NACK to indicate termination.

If the potential GM has decided to continue, they will examine the information within the policy token to determine if this is a group they are authorized and interested in joining. If the decision is not to join, the session is terminated. GSAKMP sends a properly authenticated message with a Notification Payload of type NACK to indicate termination.

If the potential GM is satisfied with the received information and decides to join the group, he will pass back a message containing the following:

1. Signed GSA response, and

2. GM's certificate (optional).

The components of an Invitation Response message are shown in Table 3:

Table 3: Invitation Response Message Definition

Message Name	:	Invitation Response
Dissection	:	{HDR, [Key Creation], GSA RS}SigM, [CertM], [RuleM]
Payload Types	:	GSAKMP Header, Notification, Signature, [Key Creation], [Certificate], [Vendor ID], [Identification/Role]
Legend	:	
SigM	:	Signature of Group Member
CertM	:	Certificate of Group Member
{}SigX	:	Indicates minimum fields used in Signature
[]	:	Indicate an optional data item

The GC receives this message and processes it according to the provisions of Processing the Invitation Response. In this procedure, the GC will verify the signature on the message to ensure its authenticity. If the message signature does not verify, the session is terminated. GSAKMP sends a properly authenticated message with a Notification Payload of type NACK to indicate termination.

If the message signature is verified, and the GM passes the GC's Access Control checks, the GC will create and send a signed message containing the GTEK and the Rekey array to the GM.

The components of a Key Download message are shown in Table 4:

Table 4: Key Download Message Definition

Message Name	:	Key Download
Dissection	:	{HDR, (GTEK, Rekeys)SigC}, [SigSC], [CertSC]
Payload Types	:	GSAKMP Header, Key Download, Signature, [Key Creation], [Identification/Role], [Vendor ID]
Legend	:	
SigC	:	Signature of Group Controller
SigSC	:	Signature of Subordinate Group Controller
CertC	:	Certificate of Group Controller
CertSC	:	Certificate of Subordinate Group Controller
{}SigX	:	Indicates minimum fields used in Signature
[]	:	Indicate an optional data item

The GM receives this message and processes it according to the provisions of Processing the Key Download. In this procedure, the GM will verify the signature on the message to ensure its authenticity. If the message signature does not verify, the session is terminated. GSAKMP sends a properly authenticated message with a Notification Payload of type NACK to indicate termination.

If the message signature is verified, the GM will create a signed acknowledgment message to return to the GC.

The components of an Acknowledgment message are shown in Table 5:

Table 5: Acknowledgment Message Definition

Message Name	:	Acknowledgment
Dissection	:	{HDR, ACK}SigM, [CertM]
Payload Types	:	GSAKMP Header, Notification, Signature, [Certificate], [Vendor ID]
Legend	:	
SigM	:	Signature of Group Member
CertM	:	Certificate of Group Member
{}SigX	:	Indicates minimum fields used in Signature
[]	:	Indicate an optional data item

The GC receives the signed acknowledgment and processes it according to the provision of Processing the Acknowledgement. In this procedure, the GC will verify the signature on the message to ensure its authenticity. If the message signature does not verify, the session is terminated. GSAKMP sends a properly authenticated message with a Notification Payload of type NACK to indicate termination.

If the message signature is verified, then the GC and GM have established a Shared Keyed Group Session.

3.2 Group Maintenance

The Group Maintenance phase includes member joins and leaves, group rekey activities, and the management of Rekey events. These activities are presented in the following sections.

3.2.1 Member Joins/Leaves

The addition of group members to a previously established group will closely follow the processing presented in Section 3.1 – Group Establishment. With the exception of the pure group establishment tasks (e.g., creation of policy token, GTEK, and Rekey array), an entity becomes a GM using the same message exchanges described in Section 3.1.

A member who elects to voluntarily leave the group will be responsible for destroying his key. Any further action for a voluntary leave should be specifically addressed in the group's security policy.

3.2.2 Rekey Events

A Rekey event is any action, including compromises, that involves the creation and dissemination of a new group key and/or Rekey information.

Once it has been identified, using the group's security policy, that a Rekey event has occurred, the GC must create and send a signed message containing the GTEK and Rekey array to the group.

Each GM who receives this message must verify the signature on the message to ensure its authenticity. If the message signature does not verify, the session is terminated. GSAKMP sends a properly authenticated message with a Notification Payload of type NACK to indicate termination. Upon verification the GM will find the appropriate Rekey download packet and decrypt the information with a stored Rekey key.

The components of a Rekey Event message are shown in Table 6:

Table 6: Rekey Event Message Definition

Message Name	:	Rekey Event
Dissection	:	{HDR, Grp ID, [Policy Token], Rekey Array}SigC, [CertC]
Payload Types	:	GSAKMP Header, [Policy Token], Rekey Event, Signature, [Certificate], [Vendor ID], [Identification/Role]
Legend	:	
SigC	:	Signature of Group Controller
CertC	:	Certificate of Group Controller
{}SigX	:	Indicates minimum fields used in Signature
[]	:	Indicate an optional data item

3.3 Group Removal/Destruction

At this point in the group's life-cycle, there has been a decision to destroy the group and the notification is broadcast on a key management channel or through a directory service.

The components of a Group Removal/Destruction message are shown in Table 7:

Table 7: Group Removal/Destruction Message Definition

Message Name	:	Group Removal/Destruction
Dissection	:	{HDR, Grp ID, [Policy Token], Destruct}SigC, [CertC]
Payload Types	:	GSAKMP Header, [Policy Token], Notification, Signature, [Certificate], [Vendor ID], [Identification/Role]
Legend	:	
SigC	:	Signature of Group Controller
CertC	:	Certificate of Group Controller
{}SigX	:	Indicates minimum fields used in Signature
[]	:	Indicate an optional data item

4 Message formats

4.1 GSAKMP Header

The GSAKMP Header fields are defined in Figure 2:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!Group ID Type  !      Group ID Value                           ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
~                                                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
~               ! Next Payload  !   Version     ! Exchange Type !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Message ID                                                    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Length                                                        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 2: GSAKMP Header Format

Group Identification Type (1 octet) - Table 8 presents the group identification types.

Table 8: Group Identification Types

Grp ID Type	Value
IPSec IPv4	0
IPSec IPv6	1
TLS	2
SMIME	3
Other	4-255

Group Identification Value (8 octets) - Indicates the name/title of the group.

Next Payload (1 octet) - Indicates the type of the first payload in the message. The format for each payload is defined in the following sections. Table 9 presents the payload types.

Version (1 octet) - Indicates the version of the GSAKMP protocol in use.

Exchange Type (1 octet) - Indicates the type of exchange (also known as the message type). Table 10 presents the exchange type values.

Message ID (4 octets) - Unique Message Identifier used to identify protocol state during negotiations. This value is randomly generated by the initiator of communications for each group join activity. In

Table 9: Payload Types

Next_Payload_Type	Value
None	0
Policy Token	1
Key Download Packet	2
Rekey event	3
Identification/ Roles (ID)	4
Certificate (CERT)	5
Certificate Request (CR)	6
Hash (HASH)	7
Signature (SIG)	8
Notification (N)	9
Delete (D)	10
Vendor ID (VID)	11
Key Creation	12
Reserved	13 - 127
Private Use	128 – 255

the event of simultaneous GSA establishments (i.e., collisions), the value of this field may be different because they are independently generated and, thus, two group security associations will progress toward establishment. However, it is unlikely there will be absolute simultaneous establishments.

Length (4 octets) - Length of total message (header + payloads) in octets. Encryption can expand the size of a GSAKMP message.

4.2 Generic Payload Header

Each GSAKMP payload defined in the following sections begins with a generic header, shown in Figure 3, which provides a payload "chaining" capability and clearly defines the boundaries of a payload. The Generic Payload Header fields are defined as follows:

```
 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        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 3: Generic Payload Header

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.

Table 10: Exchange Types

Exchange_Type	Value
Request to Join	0
Invitation	1
Invitation Response	2
Key Download	3
Acknowledgement	4
Rekey Event	5
Group Removal/Destruction	6
Other	7-255

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

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

4.3 Data Attributes Payload

There are instances within GSAKMP where it is necessary to represent Data Attributes. These Data Attributes are not a GSAKMP payload, but are contained within GSAKMP 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 in Figure 4.

```
  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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 !        Attribute Type         !     AF=0  Attribute Length    !
 !                               !     AF=1  Attribute Value     !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 !                   AF=0  Attribute Value                       ~
 !                   AF=1  Not Transmitted                       ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 4: Data Attributes Payload

The Data Attributes fields are defined as follows:

Attribute Type (2 octets) - Unique identifier for each type of attribute. 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.

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.

Attribute Value (variable length) - Value of the attribute associated with the GSAKMP-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.

4.4 Policy Token Payload

The Policy Token Payload contains group specific information that describes the group security relevant behaviors, access control parameters, and security mechanisms. This information may contain a digital signature(s) to prove authority and integrity of the information. Figure 5 shows the format of the payload.

```
 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        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!   ID Type     !              Policy Token Data                ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 5: Policy Token Payload Format

The Policy Token Payload fields are defined as follows:

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.

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

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

ID Type (1 octet) - Specifies the type of Policy Token being used. Table 11 identifies the types of policy tokens.

Table 11: Policy Token Types

ID_Type	Value
Group	0
Auxiliary	1
Reserved	2-63
Unassigned	64-255

Policy Token Data (variable length) - Contains Policy Token information. The values for this field are group specific and the format is specified by the ID Type field.

The payload type for the Policy Token Payload is one (1).

```
 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 Download Data                          ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 6: Key Download Payload Format

4.5 Key Download Payload

The Key Download Payload contains group keys. These key download payloads can have several security attributes applied to them based upon the security policy of the group. Figure 6 shows the format of the payload.

If the security policy of the group dictates, the key download payload may be encrypted with a key exchange key (KEK). The type of encryption used is specified in the Policy Token. The group members may create the KEK using the key creation method identified in the Key Creation Payload.

The Key Download Payload fields are defined as follows:

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.

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

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

Key Download Data (variable length) - Contains Key Download information.

Number of Key Packets (2 octets) – Contains the total number of both GTEK and Rekey arrays being passed in this data block.

For each Key Packet, the data format is as follows:

Key Download Data (KDD) Type (1 octet) – Identifier for the Key Data field of this Key Packet. See Table 12 for the possible values of this field.

Table 12: Key Download Data Types

Key Download Data Type	Value
GTEK	0
Rekey	1
Unassigned	2-255

Key Download Length (2 octets) – Length in octets of the Key Packet data following this field.

Key Packet Data (variable length) – Contains Key information. The format of this field is specific depending on the value of the Key Download Data field.

4.5.1 GTEK Key Packet

For a Key Download Data value of GTEK, the Key Packet Data field is formatted as follows:

Key Type (1 octet) – This is the encryption algorithm for which this key data is to be used. This value is specified in the Policy Token.

Key Creation Date (4 octets) – This is the UNIX time value of when this key data was originally generated.

Key Expiration Date (4 octets) – This is the UNIX time value of when this key is no longer valid for use.

Key Handle (4 octets) – This is the randomly generated value to uniquely identify a key.

Key Data (variable length) – This is the actual encryption key data, which is dependant on the Key Type algorithm for its format.

4.5.2 Rekey Key Packet

GSAKMP currently uses the Logical Key Hierarchy (LKH) protocol for Rekey operations. This Key Packet Data is assumed to contain LKH Array data of the following format:

LKH Version (1 octet) – Contains the version of the LKH protocol which the data is formatted in.

Leaf ID (2 octets) – This is the Leaf Node ID of the LKH sequence contained in this Key Packet Data block.

Number of LKH Keys (2 octets) – This value is the number of distinct LKH keys in this sequence.

For each LKH key in the sequence, the data format is as follows:

LKH ID (2 octets) – This is the position of this key in the binarytree structure used by LKH.

Key Type (1 octet) – This is the encryption algorithm for which this key data is to be used. This value is specified in the Policy Token.

Key Creation Date (4 octets) – This is the UNIX time value of when this key data was originally generated.

Key Expiration Date (4 octets) – This is the UNIX time value of when this key is no longer valid for use.

Key Handle (4 octets) – This is the randomly generated value to uniquely identify a key.

Key Data (variable length) – This is the actual encryption key data, which is dependant on the Key Type algorithm for its format.

The payload type for the Key Download Packet is two (2).

4.6 Rekey Event Payload

The Rekey Event Payload contains multiple keys encrypted in Rekey keys. These Rekey Event payloads can have several security attributes applied to them based upon the security policy of the group. Figure 7 shows the format of the payload.

```
  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        !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 !   ID Type     !           Rekey Event Data                    ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 7: Rekey Event Payload Format

The Rekey Event Payload fields are defined as follows:

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.

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

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

ID Type (1 octet) - Specifies the type of Rekey Event being used. Table 13 presents the types of Rekey events.

Table 13: Rekey Event Types

ID_Type	Value
None	0
Group Recovery	1
Individual Recovery	2
Maintenance	3
Delete Group Key	4
Unassigned	5-255

Rekey Event Data (variable length) - Contains Rekey Event information. The values for this field are group specific and the format is specified by the ID Type field. The format for the LKH type of Rekey Event Data is located in the appendix section.

The Rekey Event payload type is three (3).

4.7 Identification/Role Payload

The Identification/Role Payload contains group-specific data used to exchange identification information. This information is used for determining the identities of subordinate controllers and may be used for determining authenticity of information. Figure 8 shows the format of the Identification Payload.

```
 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        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!   ID Type     !            Identification/Role Data           ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 8: Identification/Role Payload Format

The Identification/Role Payload fields are defined as follows:

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.

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

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

ID Type (1 octet) - Specifies the type of Identification/Role being used. Table 14 identifies the types of identities/roles.

Table 14: Identity/Role Types

ID_Type	Value
Group Controller	0
Group and Rekey Controller	1
Rekey Controller	2
Subordinate Group Controller	3
Subordinate Group and Rekey Controller	4
Subordinate Rekey Controller	5
Member ID	6
Unassigned	7-255

Identification Data (variable length) - Contains identity information. The values for this field are group-specific and the format is specified by the ID Type field.

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

4.8 Certificate Payload

The Certificate Payload provides a means to transport certificates or other certificate-related information via GSAKMP and can appear in any GSAKMP 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 9 shows the format of the Certificate Payload.

```
 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 9: Certificate Payload Format

The Certificate Payload fields are defined as follows:

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.

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

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

Certificate Encoding (1 octet) - This field indicates the type of certificate or certificate-related information contained in the Certificate Data field. Table 15 presents the types of certificate payloads.

Table 15: Certificate Payload Types

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

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 five (5).

4.9 Certificate Request Payload

The Certificate Request Payload provides a means to request certificates via GSAKMP 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 10 shows the format of the Certificate Request Payload.

```
 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 Type    !           Certificate Authority               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 10: Certificate Request Payload Format

The Certificate Payload fields are defined as follows:

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.

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

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

Certificate Type (1 octet) - Contains an encoding of the type of certificate requested.

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 six (6).

4.10 Signature Payload

The Signature Payload contains data generated by the digital signature function. The digital signature covers the Signature Payload Span and the Signature Payload up to the Signature Data. The exception to this is if the signature algorithm used is DSS with ASN.1/DER encoding. Due to the variable length of a DER encoding, the signature span across the signature payload itself only extends up to the signature data length field, not the signature data. Figure 11 shows the format of the Signature Payload.

```
  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        !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 !  Sig Type     !        Signature Payload Span                 ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 ~               ! Sig ID Role   !     Signature Timestamp       ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 ~                               !  Signer ID Length             !
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 ~                    Signer ID Data                             ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 !     Signature Length          !     Signature Data            ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
 ~                                                               ~
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 11: Signature Payload Format

The Signature Payload fields are defined as follows:

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.

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

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

Signature Type (1 octet) – Indicates the type of signature. Table 16 presents the Signature Types.

Table 16: Signature Types

Signature Type	Value
DSS with ASN.1/DER encoding	0
DSS without encoding	1
Other	2-255

Signature Payload Span (4 octets) - Identifies the information included in the signature. The first two octets define the first signature payload. The third and fourth octet define the last payload. The payloads in the message are an ordered sequence beginning at the header, with a value of 0. If the signature payload itself is not in the signature span, you must still sign over the signature payload up to the signature data.

Signature ID Role (1 octet) – Specifies the type of Identification/Role being used. Refer to Table 14 for the types of identities/roles.

Signature Timestamp (4 octets) – Date and time that the digital signature was applied.

Signer ID Length (2 octets) - Length in octets of the Signer' ID.

Signer ID (variable length) – Data identifying the Signer's ID (e.g., DN).

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

The payload type for the Signature Payload is eight (8).

4.11 Notification Payload

The Notification Payload can contain both GSAKMP and group specific data and is used to transmit informational data, such as error conditions, to a GSAKMP peer. It is possible to send multiple Notification payloads in a single GSAKMP message. Figure 12 shows the format of the Notification Payload.

```
 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         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!     Notify Message Type       !  STATUS TYPE  !               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!                        Notification Data                      ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 12: Notification Payload Format

The Notification Payload fields are defined as follows:

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.

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

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

Notify Message Type (2 octets) - Specifies the type of notification message.

Status Type (1 octet) - Specifies the status of group with respect to originator of notification.

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

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

4.11.1 Notify Message Types

Notification information specifies status data and can be used by a process managing a SA database to communicate with a peer process. For example, a secure front end or security gateway may use the Notify message to synchronize SA communication. Table 17 and Table 18 list the Notification Messages and their corresponding values.

Table 17: Notify Messages – Status Types

Status	Value
Not connected	0
Establishing group	1
Connected to group	2
Previously member of group	3
Reserved (future use)	4-255

4.12 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 GSAKMP. Figure 13 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.

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.

The Vendor ID Payload fields are defined as follows:

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.

Table 18: Notify Messages Types

Information	Value
Invalid-Payload-Type	0
Situation-Not-Supported	1
Invalid-Major-Version	2
Invalid-Version	3
Invalid-Group-ID	4
Invalid-Message-ID	5
Payload-Malformed	6
Invalid-Key-Information	7
Invalid-ID-Information	8
Invalid-Cert-Encoding	9
Invalid-Certificate	10
Cert-Type-Unsupported	11
Invalid-Cert-Authority	12
Authentication-Failed	13
Invalid-Signature	14
Notify-GSA-Lifetime	15
Certificate-Unavailable	16
Unequal-Payload-Lengths	17
Unauthorized Request	18
Unable To Take Requested Role	19
Group Deleted	20
Request To Join	21
Acknowledgement	22
Invitation	23
Invitation-Response	24
Nack	25
Reserved (future use)	26 - 8191
Private Use	8192 – 16383

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

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

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

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

4.13 Key Creation Payload

The Key Creation Payload contains information used to create key encryption keys for the key download payload. These key creation payloads can have security attributes applied to them based upon the security policy of the group. Figure 14 shows the format of the payload.

```
 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        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!                          Vendor ID (VID)                      ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 13: Vendor ID Payload Format

```
 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        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!   ID Type     !           Key Creation Data                   ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 14: Key Creation Payload Format

The Key Creation Payload fields are defined as follows:

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.

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

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

ID Type (1 octet) - Specifies the type of Key Creation being used. Table 19 identifies the types of key download information.

Table 19: Types Of Key Creation Information

ID_Type	Value
Diffie-Hellman	0
other	1-255

Key Creation Data (variable length) - Contains Key Creation information. The values for this field are group specific and the format is specified by the ID Type field.

The payload type for the Key Creation Packet is twelve (12).

5 GSAKMP State Diagram

Figure 15 presents the states encountered in the use of this protocol.

```
 (1)
  !  -----------------------------------(17)----------------
  ! !                                                      !
  V V                                                      !
(    )---------------------(4)---------------->(      )    !
(idle)                                         (queued)    !
(     )<-------------------(5)-----------------(      )    !
 !   ^                                                     !
 !   !                                                     !
(2) (3)                                                    !
 V   !                                                     !
(Establishing Group) -(10)-> (GSA Established) -(16)->(Destroy GSA)
 ! ^         ^                  ! ^          ^
 ! !         !                  ! !          !----(15)----
 ! !         !                  !  -----(13)-            !
(6)!         ------(9)-----     --(12)--    !            !
 !(7)                     !           !     !            !
 V !                      !           V     !            !
(Establishing Group) (GSA Established) (Destroy GSA) (Destroy GSA)
```

Figure 15: GSAKMP State Diagram

Table 20 defines the transitions.

6 APPENDIX A – Rekey Packet data format

This appendix defines the format of the Rekey Event Data in the Rekey Event Payload, when using Logical Key Hierarchy (LKH) as the rekeying mechanism.

The Rekey Event Data consists of Rekey Event Header and Rekey Event Packet Data(s). A Packet Data is a complete set of information that an end-user requires to be Rekeyed. Packet Datas are comprised of new Key Packs of types GTEK and Rekey.

6.1 Rekey Event Header

The Rekey Event Data Header contains information about the rekey data being transmitted to the group. Figure 16 shows the format for the header.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
!                    Group ID Value                             ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
~                    Group ID Value                             !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Time/Date Stamp                                               !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Rekey Type    ! Algorithm Ver ! # of Rekey Packets            !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Rekey Event Packet Data(s)                                    ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 16: A.1: Rekey Event Header Format

Group Identification Value (8 octets) - Indicates the name/title of the group to be rekeyed. This is the same format as the Group Identification Value in the GSAKMP Message Header.

Time/Date Stamp (4 octets) - This is the UNIX time value of when the Rekey Event Data was generated.

Rekey Type (1 octet) - This is the Rekey algorithm being used for this group. This value is token specific. For this appendix, this value is LKH, which has a value of one (1).

Algorithm Version (1 octet) - Indicates the version of the Rekey Type being used. The value at this time is one (1).

# of Rekey Packets (2 octets) - The number of Rekey Packets contained in the Rekey Data.

Rekey Event Packet Data(s) (variable length) - Contains the packets of rekey event information being transmitted.

6.2 Rekey Event Packet Data(s)

As defined in the Rekey Event Header, # of Rekey Packets field, multiple pieces of information are sent in a Rekey Event Data. Each end user, will be interested in only one packet of the information sent. Each Packet, will contain all the Key Packs that a user requires. For each Packet, the data following the Security Header fields is encrypted with the key identified in the Security Header. Figure 17 shows the format of each Rekey Event Packet with respect to LKH.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Packet Length                 ! Security Header: LKH ID       !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Security Header: Key Handle                                   !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! # of Key Packs                ! Key Pack Data(s)              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 17: A.2: Rekey Event Packet Data Format

Packet Length (2 octets) - Length in octets of the Rekey Packet, which consists of the # of Key Packs and the Key Pack Data(s).

Security Header: LKH ID (2 octets) - This is the LKH ID of the Rekey Pack that is being used for encryption/decryption.

Security Header: Key Handle (4 octets) - This is a randomly generated value to uniquely identify the key defined by the LKH ID.

# of Key Packs (2 octets) - The number of key packs contained in this Packet Data.

Key Pack Data(s) (variable length) - Contains all the key pack data for this packet.

6.3 Key Pack Data

Each Key Pack contains all the information about the key.
Figure 18 shows the format for each type of key pack.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
! Pack Type     ! Pack Length                   ! Pack Data     ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-!
```

Figure 18: A.3: Key Pack Data Format

Pack Type (1 octet) - The type of key in this key pack. Legal values are GTEK (0) and LKH (1).

Pack Length (2 octets) - The length of the Pack Data.

Pack Data (variable length) - The actual data of the key, defined by the key type.

6.4 Pack Data Formats

There are 2 legal values for the Pack Type, GTEK and LKH. The formats for each Pack type are defined in this section.

6.4.1 GTEK Pack Data

This is data for the new GTEK being sent to the Rekeyed group.

Key Type (1 octet) - This is the encryption algorithm for which this key data is to be used. This value is specified in the Policy Token.

Key Creation Date (4 octets) - This is the UNIX time value of when this key data was originally generated.

Key Expiration Date (4 octets) - This is the UNIX time value of when this key is no longer valid for use.

Key Handle (4 octets) - This is the randomly generated value to uniquely identify a key.

Key Data (variable length) - This is the actual encryption key data, which is dependant on the Key Type algorithm for its format.

6.4.2 LKH Pack Data

This is the data to fix an Group Member Rekey sequence to recover from a compromise.

LKH ID (2 octets) – This is the position of this key in the binary tree structure used by LKH.

Key Type (1 octet) - This is the encryption algorithm for which this key data is to be used. This value is specified in the Policy Token.

Key Creation Date (4 octets) - This is the UNIX time value of when this key data was originally generated.

Key Expiration Date (4 octets) - This is the UNIX time value of when this key is no longer valid for use.

Key Handle (4 octets) - This is the randomly generated value to uniquely identify a key.

Key Data (variable length) - This is the actual encryption key data, which is dependant on the Key Type algorithm for its format.

6.5 Example

This section will give an example of the data. The data to be transmitted is:

| GroupID | Date/Time | Rekey Type | Algorithm Ver | # of Packets|
{ (GTEK)A, (GTEK, B, E)6, (GTEK, B)F }

This data shows that three packets are being transmitted. Read each packet as:
a) GTEK wrapped in LKH key A
b) GTEK, LKH keys B & E, all wrapped in LKH key 6
c) GTEK and LKH key B, all wrapped in LKH key F

We will show format for all header data, and packet (b).

Definition of values:

```
0xLLLL       -  length value
0xHHHHHHH#   -  handle value
0xTTTTTTTC   -  creation time
0xTTTTTTTE   -  expiration time
```

```
GroupID        - 0xAABBCCDD
                 0x12345678
Date/Time      - 0x34574509
Rekey Type     - 0x01 (LKH)
Algorithm Vers - 0x01
# of Packets   - 0x0003
For Packet (b):
Packet Length      - 0xLLLL
Sec HDR:LKH ID     - 0x0006
Sec HDR:Key Handle - 0xHHHHHHH1
# of Key Packs     - 0x0003
  Key Pack 1:
    Pack Type   - 0x00 (GTEK)
    Pack Length - 0xLLLL
      Key Type            - 0x02 (DES3)
      Key Creation Date   - 0xTTTTTTTC
      Key Expiration Date - 0xTTTTTTTE
      Key Handle          - 0xHHHHHHH2
      Key Data            - variable, based on key definition
  Key Pack 2:
    Pack Type   - 0x01 (LKH)
    Pack Length - 0xLLLL
      LKH ID              - 0x000B
      Key Type            - 0x02 (DES3)
      Key Creation Date   - 0xTTTTTTTC
      Key Expiration Date - 0xTTTTTTTE
      Key Handle          - 0xHHHHHHH3
      Key Data            - variable, based on key definition
```

```
Key Pack 3:
  Pack Type   - 0x01 (LKH)
  Pack Length - 0xLLLL
    LKH ID              - 0x000E
    Key Type            - 0x02 (DES3)
    Key Creation Date   - 0xTTTTTTTC
    Key Expiration Date - 0xTTTTTTTE
    Key Handle          - 0xHHHHHHH4
    Key Data            - variable, based on key definition
```

7 Authors Addresses

Hugh Harney (point-of-contact)
9861 Broken Land Parkway
Suite 300
Columbia, MD 21046
(410) 381-9400 ext 203
FAX (410) 381-5559
hh@columbia.sparta.com

Andrea Colegrove
9861 Broken Land Parkway
Suite 300
Columbia, MD 21046
(410) 381-9400 ext 232
FAX (410) 381-5559
acc@columbia.sparta.com

Eric J. Harder
R231 NSA
9800 Savage Rd
Suite 6534
Fort Meade, MD 20755
(301) 688-0847
FAX (301) 688-0255
ejharde@tycho.ncsc.mil

Uri Meth
9861 Broken Land Parkway
Suite 300
Columbia, MD 21046
(410) 381-9400 ext 233
FAX (410) 381-5559
umeth@columbia.sparta.com

Rod Fleischer
9861 Broken Land Parkway
Suite 300
Columbia, MD 21046
(410) 381-9400 ext 237
FAX (410) 381-5559
rodf@columbia.sparta.com

Document expiration: October 30, 2000

Table 20: State Transition Events

Transition 1	Request to Join is received from TCP/IP GUI Input Application Input
Transition 2	Group SA Required
Transition 3	Failure of Peer SA service Protocol Message failure Incorrect format Signature failed validation Certificate on CRL Access control invalid Authorization invalid Timeout
Transition 4	Session required, but tables full Session required, but processor busy
Transition 5	Timeout
Transition 6	Request Peer SA service Create Protocol Messages
Transition 7	Peer SA established
Transition 8,11,14	NA
Transition 9	Receipt of protocol messages
Transition 10	Group SA establishment complete
Transition 12	LKH event message completed
Transition 13	Group SA send failure notification
Transition 15	LKH event message
Transition 16	Delete Request validated
Transition 17	Destruction complete