User Guide - English

FUJITSU

IPSec V1.4

Internet Security in BS2000/OSD

Edition November 2010

Comments… Suggestions… Corrections…

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to:
manuals@ts.fujitsu.com

Certified documentation according to DIN EN ISO 9001:2008

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2008.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright and Trademarks

Contents

1 Preface

The selectable unit openNet Server contains the BS2000/OSD transport system. In addition to the proprietary NEA protocols, the BS2000/OSD Communication Manager BCAM also supports the ISO protocol and the TCP/IP protocols IPv4 and IPv6.

As of version 3.0, openNet Server supports the IP Security protocol IPSec, which provides comprehensive security mechanisms for connectionless, packet-switched IP communication.

1.1 Brief description of IPSec (IP Security protocol)

The IP protocol itself provides only very limited data security. To ensure the integrity of data packets, the IP protocol calculates the 16-bit header checksum. This mechanism is inadequate, since it makes it easy for hackers and other unauthorized persons to:

- Forge IP addresses (IP spoofing)
- Read and change the contents of IP packets during transfer
- Intercept IP packets and insert changed packets into the data stream (i.e. replay them)

By contrast, IPSec offers comprehensive, high-quality, interoperable protection for IPv4 and IPv6 packets based on cryptographic algorithms.

It covers the following aspects of data security:

- Access control
- The integrity of connectionless IP data transfer
- Authentication of the source of IP packets (ascertaining that the data really originates from the specified sender)
- Protection against interception of data packets and insertion of changed packets (anti-replay mechanism)
- Data confidentiality
- Prevention of traffic flow analyses

The security mechanisms supported by IPSec are implemented at the network layer of the TCP/IP protocol stack.

1.2 Target group

This manual is intended for the following readers:

- Network administrators
- Developers of network applications in BS2000/OSD
- Anyone interested in questions of Internet security, particularly in the BS2000/OSD environment

It is assumed that readers have knowledge of the BS2000/OSD operating system, the basic TCP/IP concepts and cryptographic security mechanisms.

1.3 License provisions

The following copyright notices concern the programs Racoon2, SPMD und CTRLPROG.

LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

iked/ikev1/evt.h

```
/*
 * Copyright (C) 2004 Emmanuel Dreyfus
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. Neither the name of the project nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE PROJECT AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE PROJECT OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */
```

---

iked/ikev1/genlist.c
iked/ikev1/genlist.h
iked/ikev1/ikev1_natt.c
iked/ikev1/ikev1_natt.h

```
/*
 * Copyright (C) 2004 SuSE Linux AG, Nuernberg, Germany.
 * Contributed by: Michal Ludvig <mludvig@suse.cz>, SUSE Labs
```

1.4 Summary of contents

The manual is subdivided into two parts:

- A general section (chapters 2 to 4), which provides information on the concept behind IPSec
- A BS2000/OSD-specific section (chapters 5 and 6), which describes the implementation of IPSec in BS2000/OSD and the installation, configuration and generation of the IPSec subsystem.

Experienced readers can skip chapters 2 to 4 and go directly to chapter 5.

The chapters of the manual deal with the following topics:

- Chapter 2 provides an overview of the threats to Internet security and the measures designed to combat the associated risks.
- Chapter 3 outlines the history of the development of IPSec and describes the benefits and uses of IPSec.
- Chapter 4 explains the concept behind the security architecture of the Internet protocols and describes the associated Security Association Database (SAD) and Security Policy Database (SPD). It also explains the Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols in transport mode and tunnel mode and the main processes involved in IPSec.
- Chapter 5 describes the implementation of IPSec in BS2000/OSD.
- Chapter 6 describes the installation, configuration and startup of the IPSec subsystem in BS2000/OSD and how to use and configure the programs Racoon2 and SPMD. It also contains brief instructions on IPSec operation to help you get started.
- Chapter 7 contains information on IPSec messages.
- The appendix provides more detailed descriptions of the Security Association Database (SAD); the Security Policy Database (SPD); the Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols in transport mode and tunnel mode; the structure of the AH and the processing of the AH protocol; and the structure of the ESP and the processing of the ESP protocol.
- The glossary defines the important concepts and terms in relation to IPSec.
- The list of abbreviations explains the abbreviations used in this manual.

1.5 Changes since IPSec V1.3

New functions

IPSec V1.4 and higher supports NAT traversal, DNS and IPCompression for IKEv2.

Current RFC basis

The implementation status has been brought in line with the current RFC basis.

IKEv1 implementations must provide the following functionality (RFC4109):

- Encryption algorithm 3DES-CBC
- Hash algorithms SHA-1
- Authentication with a preshared secret key
- Diffie-Hellman Group 2

IKEv2 implementations must provide the following functionality (RFC4307):

- Encryption algorithm 3DES-CBC
- Hash algorithms SHA-1
- Authentication with a preshared secret key
- Diffie-Hellman Group 2

IPSec examples

Examples of NAT traversal and DNS have been included.

1.6 Changes since IPSec V1.2

Sections and functions which have been removed

- The previous section 3.4 “IPSec RFCs” has been removed.
- Automatic generation of keys has been removed (KEY-VALUE=*AUTOMATIC operand in the KEY and SIGNATURE statements).
- The MODE=IKE(..) operand in the SECURITY-ASSOCIATION statement has been removed.

TU subsystems Racoon2 and SPMD

IPSec V1.3 and higher supports not only the IKE Protocol Version 1 (IKEv1), but also IKE Protocol Version 2 (IKEv2), which is implemented via the TU Racoon2 and SPMD subsystems.

IPSec configuration file

The POLICY record has been changed:

- Specification of the prefix length is supported in IPv4 and IPv6 addresses.
- The new operand COMPRESSION has been introduced.

IPSec administration commands

LOAD-IPSEC-SECRETS removed.

IPSec examples

Existing examples have been adapted to the new implementation, and new examples have been included.

Messages

- Modified messages:
  YIS0004
  YIS0401
  YIS0402
  YIS0403
- Deleted messages:
  YIS0404
  YIS0409

1.7 Changes since IPSec V1.1

The manual has been restructured since IPSec V1.1. It is now more compact, and less important details have been moved to the appendix.

The following changes have also been made:

TU subsystem Pluto

As of V1.2, IPSec supports the IKE protocol, which is implemented by the TU subsystem Pluto.

IPSec configuration file

New control statements have been introduced in the IPSec configuration file:

- INCLUDE
- FLUSH
- ADD
- DELETE

In addition, a new configuration record has been introduced, and the existing configuration records have changed:

- The PARTNER-SAS record has been introduced.
- The KEY record has been changed as follows:
  The KEY-VALUE keyword supports the value *AUTOMATIC.
- The SIGNATURE record has been changed as follows:
  The SIGNATURE-VALUE keyword supports the value *AUTOMATIC.
- The SIGNATURE-ASSOCIATION record has been changed as follows:
  The MODE keyword supports the value IKE, and thus also LIFETIME and CHECK-SEQUENCE-NUMBER.
- The POLICY record has been changed.

IPSec administration commands

The LOAD-IPSEC-SECRETS command has been introduced.

Messages

- Changed messages:

  YIS0218
  YIS0251
  YIS0252
  YIS0253
  YIS0405
  YIS0406
  YIS0407

- New messages:

  YIS0220
  YIS0254
  YIS0255
  YIS0256

1.8 Notational conventions

Typographic conventions

This manual uses the following notational conventions to highlight important parts of the text:

For notes

CAUTION!

For warnings

Italics

Italics are used to denote variables you have to replace with values.

`Fixed-pitch text`

This is used to denote entries made in the system, system outputs and file names in examples.

Command

In the syntax descriptions for commands, the parts you have to enter unchanged (the names of commands and parameters) are shown in bold.

► This symbol draws your attention to entries to be made and steps to be performed by the user.

Syntax for the command descriptions

An SDF-like representation is used to describe the syntax of some IPSec commands. The characters, representation and data types are explained in the following two tables.

Element/ formatting	Purpose	Examples
UPPERCASE LETTERS	Uppercase letters denote keywords (command, statement or operand names and keyword values) and constant operand values. Keyword values begin with *.	**HELP-SDF** **SCREEN-STEPS** = ***NO**
UPPERCASE LETTERS in boldface	Uppercase letters shown in boldface denote guaranteed or suggested abbreviations of keywords.	**GUID**ANCE**-MODE** = ***Y**ES
=	The equals sign connects an operand name with the associated operand values.	**GUID**ANCE**-MODE** = ***NO**
< >	Angle brackets denote variables whose range of values is described by data types and their suffixes.	**SYNTAX-F**ILE = <filename 1..54>
Underlining	Underlining denotes the default value of an operand.	**GUID**ANCE**-MODE** = ***NO**
/	A slash separates alternative operand values.	**NEXT-FIELD** = ***NO** / ***Y**ES
(...)	Parentheses denote operand values that initiate a structure.	,**UNGUID**ED**-DIA**LOG = ***Y**ES (...) / ***NO**
[ ]	Square brackets denote operand values that introduce a structure and are optional. The subsequent structure can be specified without the initiating operand value.	**SELECT** = [***BY-ATTR**IBUTES](...)
Indentation	Indentation indicates dependence on a higher-ranking operand.	,**GUID**ED**-DIA**LOG = ***Y**ES (...) ***Y**ES(...) **SCREEN-STEPS** = ***NO** / ***Y**ES

Element/ formatting	Purpose	Examples
|	A vertical bar indicates related operands within a structure. Its length marks the beginning and end of a structure. A structure may contain further structures. The number of vertical bars preceding an operand indicates the depth of the structure.	**SUP**PORT = ***TAPE**(...) ***TAPE(...)** | **VOL**UME = ***ANY**(...) | ***ANY**(...) | | ...
,	A comma precedes further operands on the same structure level.	**GUID**ANCE**-MODE** = ***NO** / ***Y**ES ,**SDF-COM**MANDS = ***NO** / ***Y**ES
list-poss(n):	The entry list-poss indicates that a list of operand values can be given at this point. If (n) is specified, the list must not have more than n elements. A list containing more than one element must be enclosed in parentheses.	list-poss: ***SAM** / ***ISAM** list-poss(40): <structured-name 1..30> list-poss(256): ***OMF** / ***SYSLST**(...) / <filename 1..54>
Alias:	The name that follows is a guaranteed alias (abbreviation) for the command or statement name.	**HELP-SDF** Alias: **HPSDF**

Data type	Character set	Points to note
alphanum-name	A…Z 0…9 $, #, @	
command-rest	Any	
composed-name	A…Z 0…9 $, #, @ Hyphen Period Catalog ID	Alphanumeric string that can be split into multiple substrings by means of a period or hyphen. If a file name can also be specified, the string may begin with a catalog ID in the form :cat: (see the filename data type).
c-string	EBCDIC characters	Must be enclosed in quotes; may be preceded by the letter C; any single quotes occurring within the string must be entered twice.

Data type	Character set	Points to note
filename	A…Z 0…9 $, #, @ Hyphen Period	Input format: [:cat:][$user.] { file / file(no.) / group / group { (*abs) / (+rel) / (-rel) } } :cat: Catalog ID (optional); the character set is limited to A…Z and 0…9; max. 4 characters; it must be enclosed in colons; the default is the catalog ID assigned to the user ID based on the entry in the user catalog. $user. User ID (optional); the character set is A…Z, 0…9, $, #, @; max. 8 characters; it must not begin with a digit; the $ and period must be specified; the default is your own user ID. $. (Special case) System default ID #file (Special case) @file (Special case) # or @ used as the first character indicates temporary files or job variables, depending on the system parameters. file(no.) Tape file name no.: version number; the character set is A...Z, 0...9, $, #, @. Parentheses must be specified. group Name of a file generation group (character set as for “file”) group { (*abs) / (+rel) / (-rel) } (*abs) Absolute generation number (1..9999); * and parentheses must be specified.

Data type	Character set	Points to note
filename (continued)		(+rel) (-rel) Relative generation number (0..99); the preceding sign and parentheses must be specified.
integer	0…9	+ or - must be the first character (preceding sign).
name	A…Z 0…9 $, #, @	Must not begin with a digit.
partial-filename	A…Z 0…9 $, #, @ Hyphen Period	Input format: [:cat:][$user.][partname.] :cat: See filename $user. See filename partname Common first part of the name of files and file generation groups (optional entry), specified in the following form: $name_1$.[$name_2$.[...]] $name_i$: see filename. The last character of partname must be a period. At least one of the parts :cat:, $user. or partname must be specified.
text	Any	See the relevant operand descriptions for the input format.
x-string	Hexadecimal: 00…FF	Must be enclosed in single quotes; must be preceded by the letter x or X; it is possible for there to be an odd number of characters.

1.9 Readme file

Information on functional changes and additions to the current product version described in this manual can be found in the product-specific readme file. You will find the readme file on your BS2000/OSD computer. The name of the file is `SYSRME.IPSEC.014.E`. Ask your system administrator to tell you the user ID under which the readme file is located. You can view the readme file by using the `/SHOW-FILE` command or an editor. Alternatively, you can print it out on a standard printer by using the following command:

```
/PRINT-DOCUMENT SYSRME.IPSEC.014.E,LINE-SPACING=*BY-EBCDIC-CONTROL
```

2 Internet security

As a result of the explosive growth of the Internet and its advance into practically all areas of daily life, security issues are becoming increasingly important:

- Communications security (data authenticity, data integrity, data confidentiality, etc.)
- Reliability (the general availability of the systems involved)
- Protection against viruses, worms, Trojan horses, backdoors, etc.

The focus of IPSec, and thus of this chapter as well, is communications security.

2.1 Threats to Internet security

The threats to Internet security can be subdivided into active and passive attacks.

Active attacks on Internet security

Active attacks on Internet security include:

- Forging the address of the message sender (address spoofing)
- Forging the contents of messages
- Intercepting messages and replaying them with changed data packets
- Changing the sequence of the messages sent

Passive attacks on Internet security

Passive attacks on Internet security include:

- Reading the contents of messages
- Analyzing the traffic flow of messages:
  - Who are the communication partners?
  - The volume of messages transferred within certain periods
  - The length and frequency of the messages

2.2 Measures for maintaining Internet security

To ward off the threats to Internet security described in the previous section, a whole host of strategies and mechanisms are available. These address the issue in different ways.

Figure 1 provides an overview of the currently available measures for maintaining Internet security.

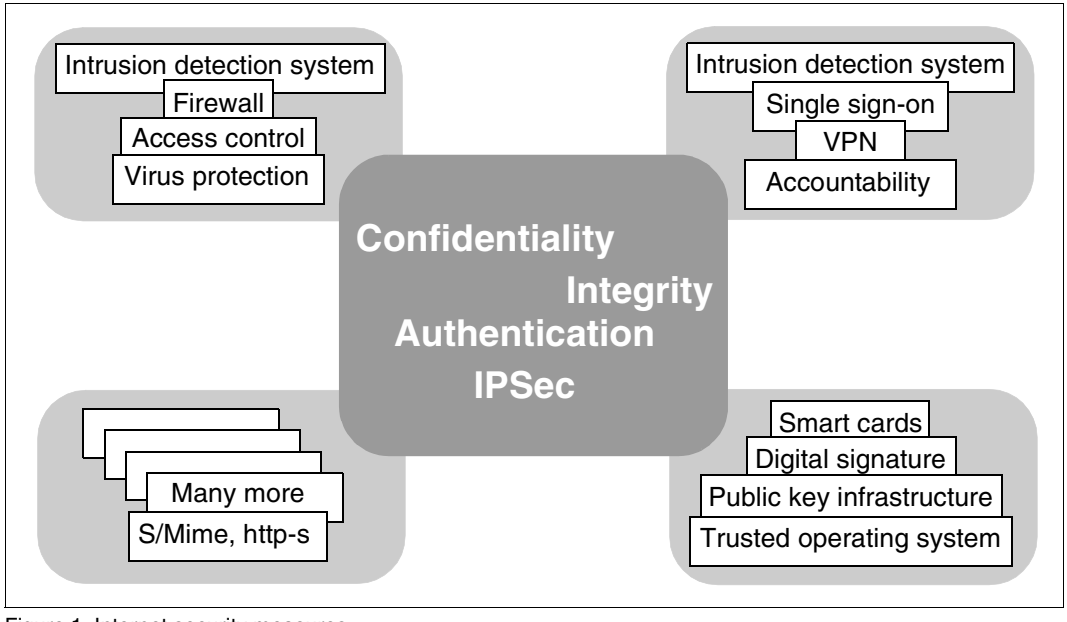


Figure 1: Internet security measures

2.2.1 Aims of the security measures

The aims of these measures are as follows:

- Authenticity
  This involves ascertaining whether the specified source of the data really is the sender.
- Data confidentiality
  This involves ensuring that data cannot be read by those not authorized to access it.
- Data integrity
  This involves ensuring that data has not been modified.
- Replay prevention
  This involves preventing data from being intercepted, changed and replayed by an intruder.
- Confidentiality of the traffic flow
  This involves preventing unauthorized analysis of the message traffic.
- Access protection for parts of the network
  This involves protecting parts of a network against unauthorized access.

IPSec allows you to implement either some or all of these measures, depending on your requirements. IPSec offers a high degree of flexibility, both with regard to the security mechanisms used and to the granularity of the IP traffic protected.

2.3 IPSec in the context of the other security protocols

IPSec offers its security services at the network layer (layer 3) of the TCP/IP protocol stack. In addition, there is a raft of further standardized security protocols at the application layer (layer 5 of the TCP/IP stack or or layer 7 of the OSI Reference Model).

The following table shows examples of standardized security protocols:

Name	Description	Defined by	Network layer TCP/IP (OSI)
IPSec	Internet Protocol Security	IETF	L3
SSL	Secure Socket Layer	Netscape Communications	L5 (L7)
TLS	Transport Layer Security	IETF	L5 (L7)
S-HTTP	Secure Hypertext Transfer Protocol	Enterprise Integration Technologies, IETF	L5 (L7)
SET	Secure Electronic Transaction	Visa and MasterCard	L5 (L7)
HBCI	Home Banking Computer Interface	Association of German Banks	L5 (L7)

Internet security standards

Figure 2 shows where the various security protocols fit into the TCP/IP protocol stack.

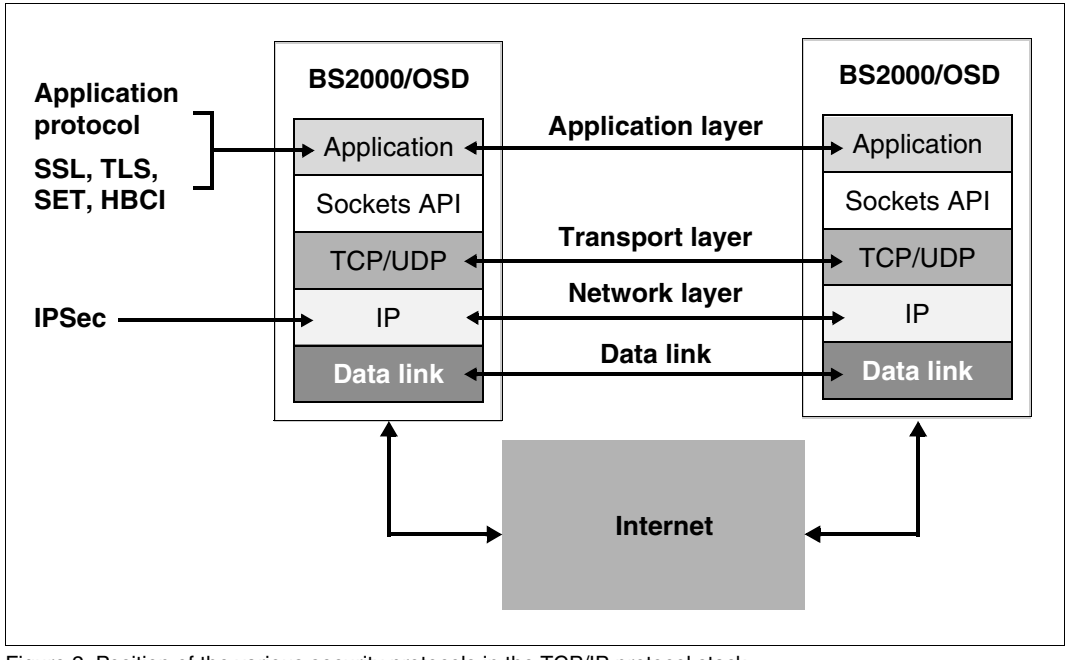


Figure 2: Position of the various security protocols in the TCP/IP protocol stack

SSL – Secure Socket Layer

The Secure Socket Layer (SSL) permits mutual authentication between two communicating applications and guarantees the confidentiality, integrity and authenticity of the application data exchanged. Client/server systems can thus communicate via SSL without running the risk of the data they transfer being intercepted or forged.

SSL implements authentication, data integrity and data confidentiality with the aid of two subordinate protocols:

- The SSL Record Protocol
- The SSL Handshake Protocol

The SSL Record Protocol defines the format to be used for the transfer of the data. The SSL Handshake Protocol enables the SSL client and SSL server to authenticate themselves to each other and to exchange encryption algorithms together with the cryptographic key before a protocol of the application layer starts transferring the data.

TLS – Transport Layer Security

V1.0 of the Transport Layer Security protocol is based on SSL. Like SSL, TLS is a layer-5 protocol. Although no major differences exist between TLS V1.0 and SSL V3.0, it should not be assumed that the two protocols are interoperable.

The extent to which TLS can be used for the additional security of application protocols such as FTP, SMTP or LDAP is currently a matter of some debate, since SSL was developed exclusively for the HTTP protocol.

S-HTTP – Secure HTTP

Secure HTTP (S-HTTP) is an extension to the HTTP protocol (HyperText Transfer Protocol) that permits secure data transfer on the World Wide Web. Each message transferred with S-HTTP can be secured by any combination of data encryption, a digital signature and authentication. An S-HTTP message consists of an encapsulated HTTP message and a header that describes the format of the encapsulated data.

SET – Secure Electronic Transaction

The Secure Electronic Transaction™ protocol SET is an open technical standard developed by Visa and MasterCard for the secure handling of payment transactions on the Internet.

HBCI – Home Banking Computer Interface

The Internet banking standard Home Banking Computer Interface (HBCI) permits the secure handling of bank transactions over the Internet. This technology is based on the encryption of all transactions using cryptographic methods. The encryption technology used in the HBCI is an electronic signature to ensure reliable protection against attacks by hackers.

2.4 Firewalls

When the Internet protocols were initially developed, security aspects such as authenticity, data integrity and confidentiality were not considered. Consequently, technologies such as firewalls had to be developed to control access from the Internet to private local networks and vice versa. Firewalls monitor the source, destination, type and direction of the data transferred.

They also implement rules designed to control this traffic. These rules can come into effect at the network layer (packet filters or screening routers), session layer (circuit-level gateways) or application layer (application-level gateways, ALG). A firewall generally uses combinations of these filter technologies.

3 Overview of IPSec

IP Security (IPSec) offers high-quality security, based on cryptographic mechanisms, for IPv4 and IPv6 datagrams. These mechanisms are implemented at the network layer of the TCP/IP protocol stack.

This chapter deals with the following topics:

- The history of the development of IPSec
- The benefits and uses of IPSec
- The security extensions to the IP protocol provided by IPSec
- IPSec RFCs

3.1 The history of the development of IPSec

The rapid growth of the Internet has meant that, in addition to the problem of assigning IP addresses, a variety of Internet security issues have become increasingly important. These include, in particular:

- Access control to protect parts of a network
- The authenticity, integrity and confidentiality of the data transferred
- Prevention of the interception of data packets and insertion of changed data packets (replaying)
- The confidentiality of the traffic flow

It was already apparent at the beginning of the 90s that a solution had to be found to these problems. Research focused not only on enlarging the address space and on Internet security but on performance and on simplifying routing.

The result of this work was the Internet Protocol version 6 (IPv6) specification. The increased demands on Internet security were taken into account in IPv6 by, among other things, the introduction of two new headers: the Authentication Header (AH) and the Encapsulating Security Payload Header (ESP). The AH and ESP are part of the IPv6 extension header concept, a major innovation since IPv4.

However, since the introduction of IPv6 took longer than originally expected, the security mechanisms envisaged for IPv6 also had to be made available in IPv4. To this end, the IP Security Working Group of the Internet Engineering Task Force (IETF) defined security mechanisms outside the IPv6 RFCs. These security mechanisms, which are defined in RFCs of their own and apply to both IPv4 and IPv6, include, above all, the headers AH and ESP.

3.2 The benefits and uses of IPSec

IPSec offers comprehensive security at the level of the IP protocol that can be utilized for both IPv4 and IPv6. With the aid of additional headers (the Authentication Header and Encapsulating Security Payload Header), IPSec ensures the security of data transfer on the Internet.

The IPSec protocol provides the following benefits:

- IPSec applies to both IPv4 and IPv6.
- IPSec is transparent for applications.
- IPSec is independent of other security mechanisms on the Internet.
- IPSec does not define a rigid security architecture.
- IPSec permits the definition of a variable security policy.

IPSec is transparent for applications

The IPSec security mechanisms do not require any modifications to be made to existing applications; they are integrated in the system at the administrative level.

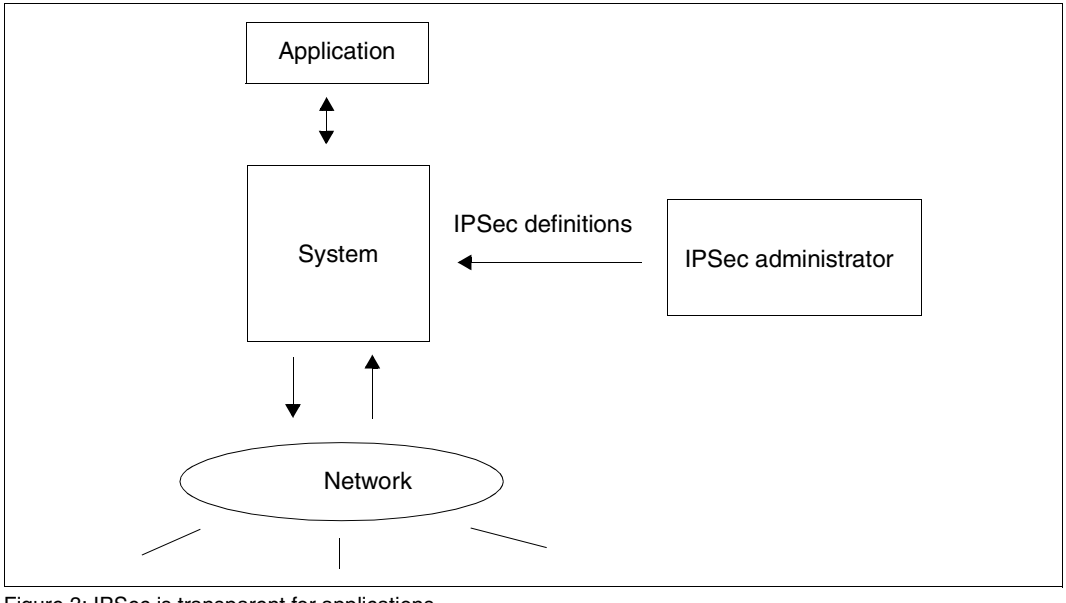


Figure 3: IPSec is transparent for applications

IPSec is independent of other security mechanisms

The security mechanisms of IPSec are defined at the network layer within the IP protocol. IPSec includes no specifications relating to the other communication layers, nor does it define any dependency on these layers. Consequently, the IPSec security mechanisms can be used in parallel with other security mechanisms, either in cooperation with them or entirely independently of them.

The scenario described below provides an example of the use of IPSec and SSL in cooperation.

Example of the use of IPSec and SSL in cooperation

The FTP server of a BS2000/OSD system is protected against access by unauthorized partners by means of IPSec authentication. Confidential data transfer between the FTP server and FTP client is handled at the level of SSL encryption. One advantage of this strategy over authentication at SSL level is that “denial of service” attacks on the FTP server are intercepted at the IP layer (i.e. the network layer). In this way, hung TCP connections, which are frequently used in “denial of service” attacks, can be avoided. In a “denial of service” attack, the attacker attempts to prevent a legitimate user of particular services from using these services by, for example, interrupting the network connection between the client and server computers.

IPSec does not define a rigid security architecture

IPSec does not define a closed security architecture. Such a security architecture is often referred to as a public key infrastructure (PKI). Instead, IPSec provides the Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols and the security association (SA) and security policy (SP) concepts, which allow security architectures to be tailored to meet specific requirements.

IPSec permits the definition of a variable security policy

The Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols can be combined, as required, with transport mode and tunnel mode. Consequently, IPSec allows great flexibility when creating security strategies.

3.3 Security extensions to the IP protocol provided by IPSec

Without IPSec, the integrity check for an IP packet is restricted to the formation of the header checksum, which only validates the IP header and is also easy to forge.

IPSec extends IP security by adding the following security services:

- Access control
- Authentication of the source of the data
- The integrity of connectionless IP data transfer
- Protection against replaying of an IP packet (interception and modification of the packet before inserting it into the data stream again)
- Data confidentiality
- Confidentiality of the traffic flow (to some extent)

IPSec implements these services with the help of the following mechanisms:

- The concept of the security association
- The Authentication Header (AH) and Encapsulating Security Payload (ESP) security protocols in transport mode and tunnel mode
- Algorithms for authentication and encryption

The IPSec security services secure data communication between two hosts, between two gateways and between a security gateway and a host. Interoperability is ensured. In other words, the IPSec security services have no negative affects on users, hosts and network components that do not support IPSec.

Since IPSec provides its security services at the level of the network layer of the IP protocol stack, protocols and services of the higher layers can use these services without problems and without the need for any changes.

4 The security architecture of the Internet protocols

IPSec provides systems with security services at the network layer of the IP protocol. In order to deliver the security services, certain resources and mechanisms are required. It must be possible, for example, to select security mechanisms and protocols. It must also be possible to specify the keys to be used, their period of validity and the mechanism by which a communiation partner receives the key information. The data to which the services are to be applied must be specified. In other words, it must be specified which data streams to which partners are to be protected.

The security services can be negotiated between a pair of end systems, a pair of security gateways or an end system and a security gateway. The systems on which the security services are provided affect how the components that provide the services work.

The following sections introduce both the concepts required for an understanding of IPSec and the components of the IPSec architecture.

The BS2000/OSD implementation of IPSec is based on this architecture. Any differences in points of detail are intended.

4.1 The selectors

In the context of IPSec, a selector is a set of descriptive resources used to define the data traffic to be handled by IPSec. Selectors make it possible to select those IP segments for which IPSec is to provide security functions. The combination of selectors with their values determines whether the set of data to which the security functions are to apply is large or small. It is said that the selectors define the granularity of the security rules. For example, if the selectors include all the data traffic between a pair of systems, they are said to be course-grained. By contrast, when the data traffic between two applications is selected, the selectors are said to be fine-grained.

The comparison values for the selectors must be obtained from the protocol fields of the packets.

- Destination IP address
  This selector refers to the address of the innermost IP protocol. IP addresses of possible IP tunnel protocols may not be used for comparison purposes. It is possible to specify a single address (a unicast address, for example), an address range, an address with a netmask or prefix length or a wildcard address. The options for specifying addresses depend on the implementation and impact on the ease of use of the operating system. However, every system must support the destination IP address as a selector.
- Source IP address
  The description of the destination IP address applies here by analogy.
- Names
  IPSec supports two name types:
  - DNS user names (fully qualified user names) and X.500 user names (distinguished names)
  - System names (also either DNS or X.500)

  The values for the name selector cannot be read from protocol fields. A mechanism is required that can map the IP address to the name (in the case of a DNS or X.500 directory query, for example).
- Transport protocol type
  The transport protocol is the protocol that follows the IP protocol (including its extensions, which include the IPSec security protocols).
- Port number
  In conjunction with the specification of TCP or UDP in the transport protocol selector, it is possible to specify the data traffic more precisely by specifying the destination and/or source port number. It should be possible to specify a wildcard port number.
- Data sensitivity level (confidentiality)
  The sensitivity of the data within a system is used by security concepts such as MAC and MLS.

  Mandatory Access Control - MAC is for monitoring and controlling access rights to IT systems in which decisions about whether to grant access are made not on the basis of the identity of a user (or process) and of the object (file, device), but on the basis of general rules and attributes of the user (or process) and object. Programs also often receive their own rights, which can further restrict the rights of the user. The reason for this is that users (or processes) can never access objects directly; they access them through a reference monitor.

The multi-level security systems (MLSs) correspond to Mandatory Access Control in its original form. When the IP segments are processed, suitable security services are selected, or it is checked which security services are being used.
IPSec requires assistance for this – from the IP protocol architecture, for example – so that the data identifiers can be transferred and/or so that an instance ensures that the required security mechanisms are in place at the right time (i.e. when sensitive data is accessed and after the rules for the object and the user or process have been checked).

When an IP segment is processed, the selector information may be unclear to the higher protocol layer (e.g. when the data of the segment is encrypted). The IPSec implementation must take this into account.

4.2 Security policy

Before security services can be used between a pair of systems, certain agreements must be reached between the instances of the systems involved. Agreements about the scope and type of the protection mechanism to be used are referred to as the security policy (SP). A security policy defines how the data packets to and from a particular partner are to be handled and which protection mechanisms have to be used.

A security policy contains instructions about how the packets to which the security policy is to be applied are to be handled. IPSec offers three alternatives:

- Bypass IPSec
  The relevant IP segments are passed on to the next layer of the protocol stack without further processing by IPSec.
- Discard IPSec
  IP segments to which the Discard policy is to be applied are not processed any further. They are deleted. The neighboring protocol layers are not informed. The resulting packet loss must be detected by a higher protocol layer. It is not possible to correct or get around this situation within IPSec.
- Apply IPSec
  Security services are provided for the selected IP segments.

If the security policy contains the Apply statement, the security policy also describes which security association is to be used. The security association is logically specified by the following:

- Security protocol
- Transfer mode
- Cryptographic algorithms

Security policies must be organized for a variety of search algorithms. Finally, a check must be carried out, on the basis of the security policies, to ascertain whether data packets have been handled correctly or how they have to be handled. The security policies are organized in the Security Policy Database (SPD).

Characteristics of the Security Policy Database (SPD)

The SPD and its elements must have the following characteristics:

- Fully organized

  The entries in the SPD must be managed in such a way that repeated searches with the same search argument result in the same entry. The search argument consists of one or more selectors consisting of protocol fields of the protocols surrounding IPSec. When filtering data traffic, you search for a suitable security policy by specifying an explicit value in the various selector fields, although it is explicitly permissible to define security policies by using wildcards and specifying value ranges in individual selectors. This results implicitly in the requirement that precisely specified entries must be found more quickly than other entries when the SPD is searched. A search is concluded when the first matching SPD entry is found.

- Derivation rules for security associations (SAs)

  Security policies specify the security association(s) to be used. It is important how SAs are derived from the SP when the set of data traffic to be covered is specified by means of wildcards or value ranges. The SA selectors can be derived from a single data packet or from the selectors of the security policy.

- Management of the Security Policy Database (SPD)

  The SPD must offer an administration interface so that security policies can be defined, manipulated and deleted. The interface must be accessible to the system administrator and can be offered to applications. This requires the SP definition to be prioritized. It must be specified whether applications are permitted to disable security policies of the system administrator.

- Combination of security associations

  A security policy can combine a number of security associations. The result is a security association bundle. The security policy specifies the sequence in which the security services are to be provided for the specified data traffic.

- Default arrangement

  Whenever IPSec is used, the Security Policy Database controls all data traffic to and from this system. It is therefore interesting to know what happens to the packets for which no security policy is defined. The SPD must offer the configuration option that either the Discard or Bypass action can be specified for such packets.

4.3 Security association

The system instances that provide the security services and ensure compliance with the specified security policy use security associations (SAs) for this purpose.

An SA is unidirectional. In other words, the parameters of an SA apply only to packets of a particular data stream to a partner system or to the packets coming from a partner system.

An SA describes the scope of the security service, which depends on different criteria. One key factor is the security protocol selected, which determines whether confidentiality and/or authenticity is ensured. The security services of an SA are provided by a single security protocol. If more than one security protocol is required to comply with a security policy, more SAs are needed (an SA bundle).

The security association describes which cryptographic algorithms are used to provide the required security.

If certain services can be used optionally within the framework of the security protocol, such as protection against replaying of packets, the information is stored in the SA. In addition, configuration parameters controlling key management (e.g. the key's period of validity or the exchange method) are saved.

The properties of the key depend on cryptographic methods and on the quality of the protection to be provided.

Depending on the positioning of the IPSec instances, the security service is provided in transport or tunnel mode. Transport and tunnel mode are transfer modes. The transfer mode is stored as a parameter of the SA. The following applies:

– An SA in transport mode can only be set up between a pair of end systems.

– If a security gateway (SG) is involved in providing the required security services, an SA must be set up in tunnel mode. This is because a security gateway offers its security services instead of an end system, and the data is forwarded to the end system after the security services are used.

– On the other hand, this also means that an end system has to support SAs in both transport mode and tunnel mode. Consequently, SAs in tunnel mode are possible between a pair of end systems.

Security Association Database (SAD)

The security associations form the Security Association Database (SAD). Unlike the Security Policy Database, the SAD does not have to be sorted. Nevertheless, it is expected that the right SA will be found using the available selectors, and that should not change with every packet. There is a further key difference from the SPD when it comes to searches.

Whereas in the SPD the same set of selectors is used to search for inbound and outbound packets, other search arguments have to be used in the SAD for inbound packets. A security association for inbound packets is uniquely identified by the following triple:

- The destination address contained in the (outer) IP protocol
- The type of the security protocol that comes after the IP protocol
- The Security Parameter Index (SPI) contained in the IPSec protocol
  The SPI is a bit string that serves to distinguish between SAs when the destination address and security protocol are the same.

All of the security associations taken together represent the current scope of the services of an IPSec system. Their period of validity is limited. They are not tied to the existence of a security policy, but they can only work when an associated security policy exists. Services are defined by IPSec with the help of the security policy and executed with the help of the security associations.

4.4 Combinations of security associations (SA bundles)

A security architecture should have a flexible design so that it can be adapted to suit the requirements of users. It may be that the required level of security cannot be achieved with a single protection function; it may have to be combined with an additional mechanism. Existing or planned network infrastructures with routers, gateways, firewalls and, together with security services, security gateways or security proxies also require the protocol architecture to be flexible. There is thus scope to decide on the systems in which security functions are to be provided, and on which functions.

Two transfer modes are available, and security associations can also be combined.

Transfer mode

Transfer mode determines which data of an IP segment is secured and where the security protocol is positioned within the protocol stack.

- Transport mode

  The security protocols are inserted immediately after the (end-to-end) IP protocol. (In the case of IPv6, that means after the IPv6 protocol and the extension protocols used by routers to provide their services.) In transport mode, only the data of the IP segment and, if need be, the fields of the IP protocol that cannot be changed by routers are secured.

  The security protocols defined in IPSec can be combined. The encryption service must be applied to the data of the IP segment first, then the authentication service.

  Security associations in transport mode can only be used between end systems.

- Tunnel mode

  In tunnel mode, the original IP segment represents the data of a surrounding IP segment. The security protocol is inserted between the two IP protocols. The security function thus also applies to a whole IP segment. Security associations in tunnel mode may be repeatedly applied to an end-to-end IP segment that has already been secured by a transport security association.

Combination options

The following combinations must be supported for end systems, referred to below as host 1 and host 2. The logical protocol stack is shown from the perspective of host 1.

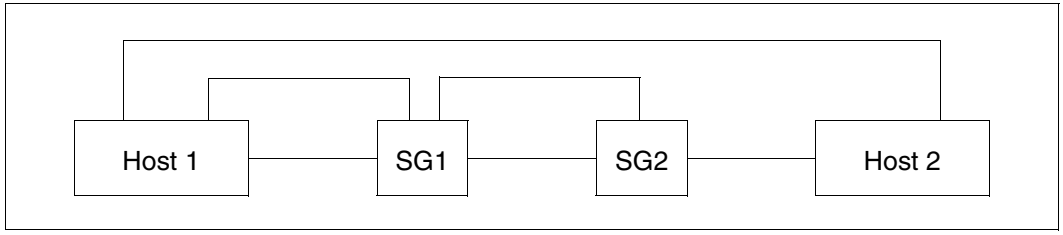


Figure 4: Representation of end systems and security gateways (SG1, SG2)

The following combinations are possible between host 1 and SG1 and between SG1 and SG2 (ULP stands for upper layer protocol):

IP2	AH	IP1	ULP

IP2	ESP	IP1	ULP

Figure 5: Combinations between host and security gateway

The following combinations are possible between host 1 and host 2:

IP	AH	ULP

IP	ESP	ULP

IP	AH	ESP	ULP

IP	AH	IP1	ULP

IP	ESP	IP1	ULP

Figure 6: Combinations between two end systems

The SA between the two security gateways SG1 and SG2 is not visible to the end systems. However, the variety of combinations possible for SAs between end systems results in complex configurations and processes.

4.5 Interaction between the components

This section explains the interaction in the case of outbound and inbound data.

4.5.1 Outbound data

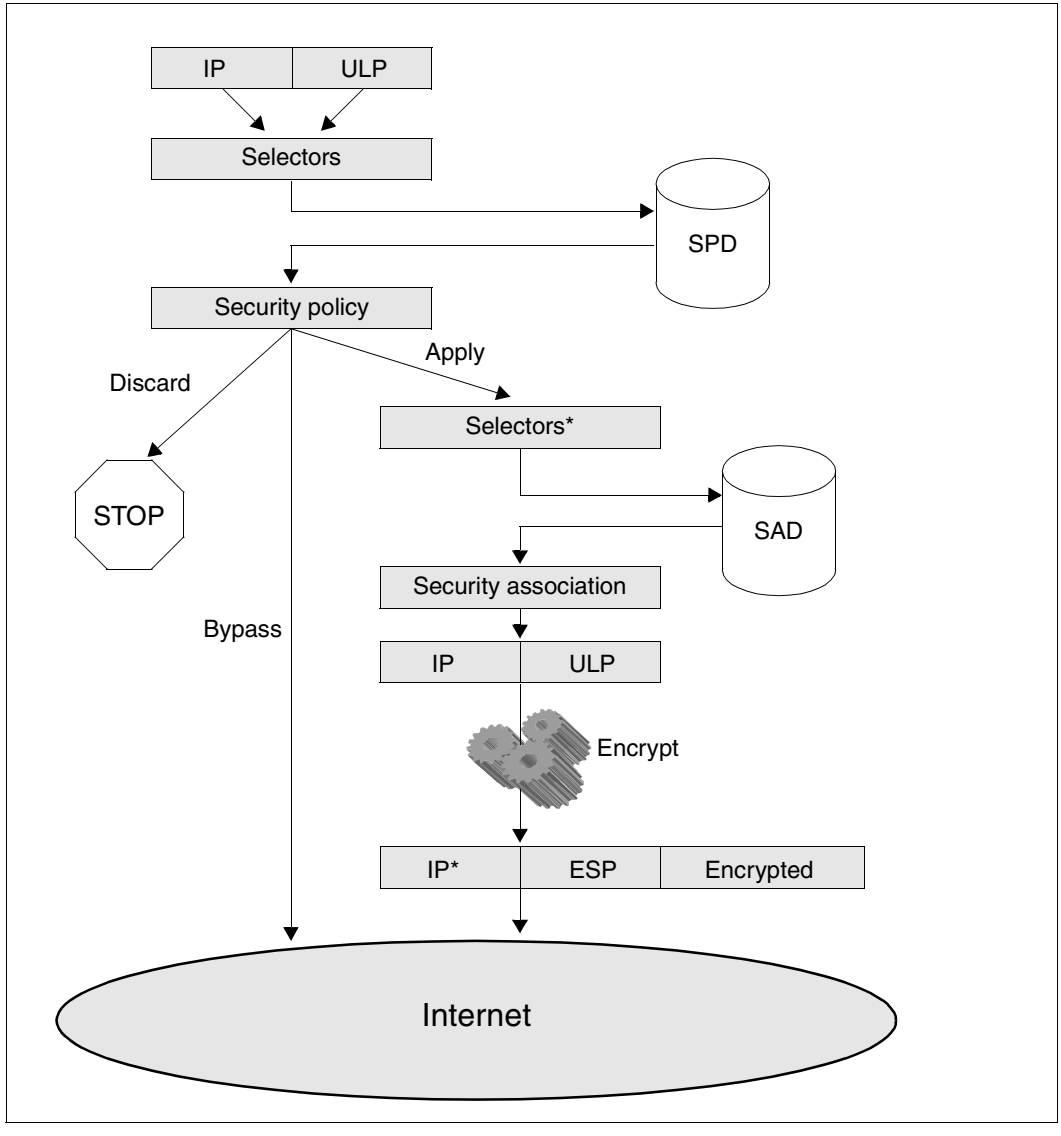


Figure 7: Interaction between the components in the case of outbound data

IPSec inspects each outbound IP segment to check whether security services are required for this segment. IPSec assembles the selectors from the protocol fields of the IP header and the upper layer protocol (ULP). A search is made of the Security Policy Database for a policy described as accurately as possible by the selectors. The security policy contains instructions for IPSec, indicating how the IP segment is to be processed:

– If the security policy defines the Discard statement, IPSec discards the IP segment.

– The Bypass statement causes IPSec to send the IP segment without any further action.

– The Apply statement instructs IPSec to search the Security Association Database for an outbound SA that represents as close a match as possible. The selectors must be adapted in accordance with the SA definition in the security policy. The transfer mode, possibly the IP address of the tunnel endpoint, the derivation rule for the SA, etc. therefore have to be taken into account as well as the security protocol. After a successful search for an SA, IPSec changes the original IP header, inserts the security protocol immediately after the IP header and applies the transformation specified in the SA. The SA-specific key is used.

4.5.2 Inbound data

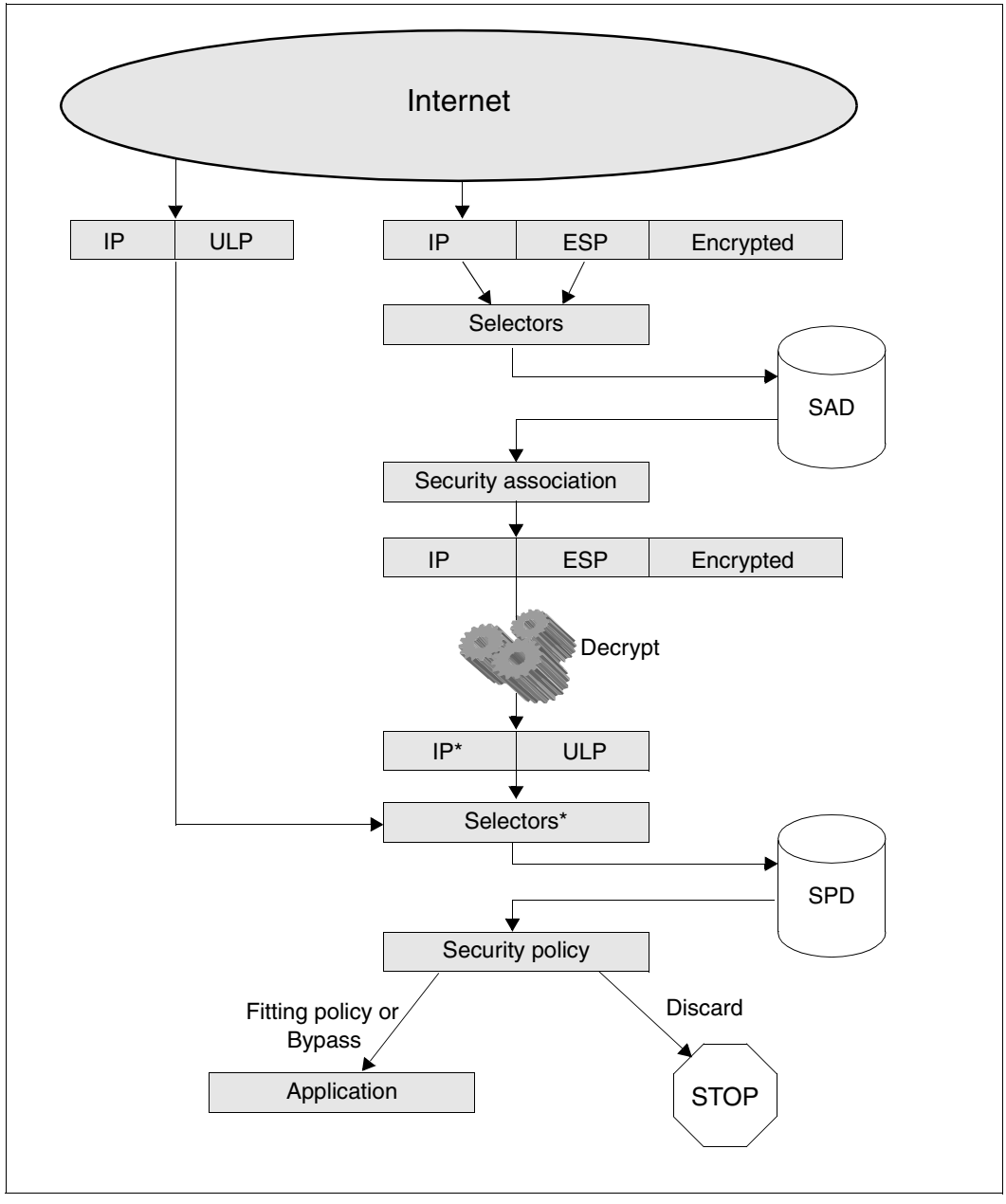


Figure 8: Interaction between the components in the case of inbound data

IPSec inevitably has to handle inbound IP segments differently from outbound IP segments. The search for the inbound security association is much simpler, but, on the other hand, the required selector information cannot be read when the data of the IP segment is encrypted. To search for the SA, IPSec uses the destination IP address of the segment, the type of the security protocol and the SPI contained in the security protocol. Once the SA has been found, IPSec applies the transformation specified in the SA. The SA-specific key is used. The security protocol is then removed and the IP header adapted.

IPSec uses the IP segment obtained in this way or the unprotected inbound IP segment to form the selectors for the search of the Security Policy Database. If a security policy is found that matches the selectors, IPSec checks whether the required security services match those of the IP segment received. For IP segments received without a security protocol, the alternatives of the Bypass and Discard policy are available, resulting in the further processing or discarding of the segment.

4.6 Security protocols

Two security protocols with different security functions are defined for the IPSec architecture. The security protocols represent the interface between the IPSec instances of the systems. The protocols contain the language resources required to provide the security services. Agreement on a security association covers the selection of a security protocol, the use of optional parameters within the protocol and the means to identify the security association. The two protocols have their own numbers within the IP protocol family.

4.6.1 Authentication Header

The security services provided using the Authentication Header (AH) protocol are:

- Authentication of the source of the data (i.e. the sender)
- Protection against data corruption
- Protection against replay attacks

The sender calculates a checksum, the Integrity Check Value (ICV), which is entered in the AH protocol and thus communicated to the recipient. The sender and recipient use the same secret key to calculate the checksum. This ensures that the data has not been corrupted and that it has indeed come from the specified sender.

Replay protection is provided on the basis of a sequence number and is optional. The service is only available when the corresponding SA has not been configured manually. During the dynamic establishment of the SA, the recipient informs the sender if it is not going to use replay protection. The sender then proceeds accordingly on the basis of this information.

The AH has the protocol number 51.

IP packet protected by AH tunnel mode

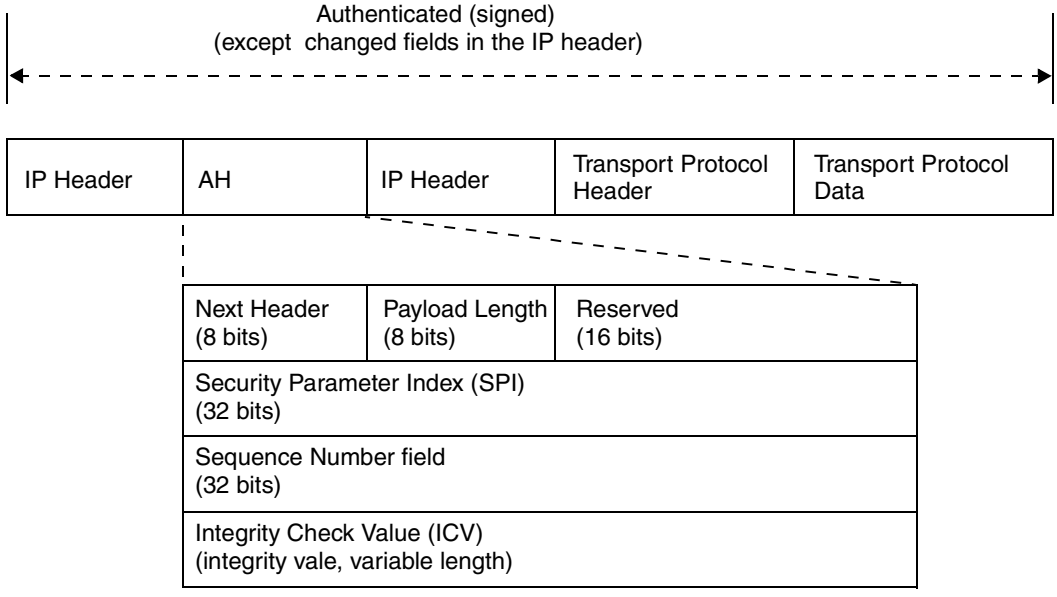


Figure 9: AH in tunnel mode

Explanation of the AH protocol fields

Next Header

This protocol field contains the protocol number of the protocol contained in the data.

When the AH is inserted, the sender takes this value from the corresponding protocol field of the header immediately before the AH.

The recipient transfers the contents of this field to the protocol header before the AH when it removes the Authentication Header.

Payload Length

This field specifies the length of the Authentication Header. The length is specified in multiples of 32 bits, not including the first 64 bits.

Reserved

16 bits, which are provided for protocol extensions. As long as they are not used, the sender must set them to zero. The recipient must ignore them.

Security Parameter Index (SPI)

The Security Parameter Index identifies the Authentication Header SA on the destination system. The SPI is stored in the SA on the sender's side. It provides the sender with the protocol language resources for addressing the security association (at the recipient's end) on the destination system. The value range from 0 to 255 is reserved.

The SPI is assigned when a security association is created, and the partner is informed. The assignment is made by the system administrator (in manual keying) or dynamically by system instances.

Sequence Number

The sequence number is used to provide optional protection against the replaying of IP segments. The recipient side involved in an SA decides whether the service is not to be used, contrary to the standard procedure, and informs the sender.

When a security association is established, a counter in the SA is initialized with the value zero. The sender increments the counter for each segment and enters the value in the Authentication Header before the checksum is calculated – regardless of whether the recipient has indicated it will not be checking the sequence number. Each packet of an AH SA thus has a sequence number that is protected by the security service.

If the sequence number threatens to overflow to zero, the sender's response depends on whether the service is being used by the recipient. The sender must agree a new SA if the service is not being used. If it is being used, the sequence number zero is sent.

The recipient uses the sequence number to check both whether the IP segment is duplicated and whether the segment is inside its receiving window. The maximum size of the receiving window remains constant. When the segment with the smallest sequence number is received, the receiving window is moved in the direction of higher numbers (sliding window, fixed size).

The sequence number may be checked before the ICV (see below). The receiving window must be adjusted after the validity of the packet is confirmed.

If the recipient has decided not to use the service, the sequence number is ignored.

The sender does not know the size of the receiving window.

When SAs are set up manually, it makes no sense to use the anti-replay service because, if the sequence number overflows, no new SA can be made available.

The anti-replay service cannot be used when multiple senders send to a receiving SA, because in this case the sequence numbers cannot be synchronized.

Integrity Check Value (ICV)

The checksum calculated by the sender is transferred in this field. The length of the checksum depends on the underlying cryptographic algorithm. IPv4 and IPv6 have different length requirements of the AH header, so this field may have to be padded.

The Integrity Check Value calculated covers the data of the original packet, certain information in the Authentication Header and the fields of the IP protocol that are considered to be immutable during transfer.

4.6.2 Encapsulating Security Payload

The following security services can be provided with the help of the Encapsulating Security Payload (ESP) protocol:

- Confidentiality
- Traffic flow confidentiality
- Authentication of the source of the data (i.e. the sender)
- Protection against data corruption
- Protection against replay attacks

The confidentiality of the data is protected by encryption (i.e. a transformation). The encryption is based on a preshared secret key known to the sender and recipient.

The confidentiality and/or concealment of the data flow is a side-effect of this data transformation because the protocols of the higher protocol layers, and thus the addresses contained, cannot be read after transformation. With ESP in tunnel mode, it is also possible to maintain secrecy about which systems are exchanging data with each other.

To authenticate the sender and ensure the integrity of the data, the ESP protocol essentially uses the same algorithms as the AH protocol. A checksum, the Integrity Check Value (ICV), is calculated and entered in the ESP protocol, thus communicating it to the recipient.

Both the encryption service and the authentication service can be disabled. However, one of the two must be provided by an ESP SA.

If the two IPSec instances on the source and destination systems decide to use both services, an ESP SA must be agreed for each with two algorithms and two keys for each direction of transfer.

Replay protection is provided on the basis of a sequence number. It is optional. Since the sequence number in the ESP protocol is only captured by the authentication service, the anti-replay service can only be selected in conjunction with the authentication service. The service is only available when the corresponding SA has not been configured manually.

During the dynamic establishment of the SA, the recipient informs the sender if it is not going to use replay protection. The sender then proceeds accordingly on the basis of this information.

The ESP protocol has the protocol number 50.

IP packet protected by ESP transport mode

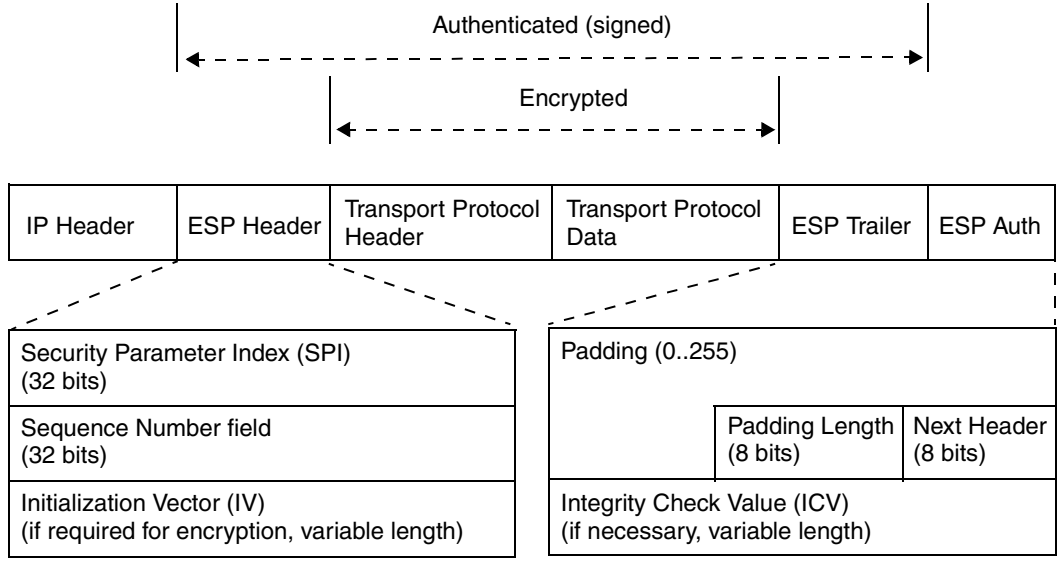


Figure 10: ESP in transport mode

Explanation of the ESP protocol fields

Security Parameter Index (SPI)

The Security Parameter Index identifies the Encapsulating Security Payload SA on the destination system. The SPI is stored in the SA on the sender's side. It provides the sender with the protocol language resources for addressing the security association (on the recipient's side) on the destination system. The value range from 0 to 255 is reserved.

The SPI is assigned when a security association is created, and the partner is informed. The assignment is made by the system administrator (in manual keying) or dynamically by system instances.

Sequence Number

The sequence number is used to provide optional protection against the replaying of IP segments.

The recipient involved in an SA decides whether the service is not to be used, contrary to the standard procedure, and informs the sender.

When a security association is established, a counter in the SA is initialized with the value zero. The sender increments the counter for each segment and enters the value in the ESP header before the checksum is calculated – regardless of whether the recipient has indicated that it will not be checking the sequence number. Each packet of an ESP SA thus has a sequence number that is protected by the security service.

If the sequence number threatens to overflow to zero, the sender's response depends on whether the service is being used by the recipient. The sender must agree a new SA if the service is not being used. If it is being used, the sequence number zero is sent.

The recipient uses the sequence number to check both whether the IP segment is duplicated and whether the segment is inside its receiving window. The maximum size of the receiving window remains constant. When the segment with the smallest sequence number is received, the receiving window is moved in the direction of higher numbers (sliding window, fixed size).

The sequence number may be checked before the ICV. The receiving window must be adjusted after the validity of the packet is confirmed.

If the recipient has decided not to use the service, the sequence number is ignored.

The sender does not know the size of the receiving window.

When SAs are set up manually, it makes no sense to use the anti-replay service because, if the sequence number overflows, no new SA can be made available.

The anti-replay service cannot be used when multiple senders send to a receiving SA, because in this case the sequence numbers cannot be synchronized.

Payload Data

The transformed (encrypted) data of the original packet is transferred in the Payload Data field of the ESP protocol. If the cryptographic algorithm used requires synchronization data, this is transported as well in the Payload Data field. The algorithm determines the length, structure and position of this information (e.g. of an Initialization Vector, IV). It also defines how this data is to be used by the recipient for decryption.

Padding Bytes

The padding bytes in the ESP header have a variety of functions. Up to 255 bytes can be entered in the protocol. The following applies:

– The padding bytes are used to align the subsequent protocol fields correctly.

– The transformation algorithm used works in blocks with multiples of N bytes. The data – consisting of the Initialization Vector, unencrypted data and the two bytes for the Padding Length and Next Header fields – has to be padded up to the next block size.

– Padding bytes are inserted to conceal the length of the original packet. This is part of the service preventing traffic flow analysis.

There are rules that stipulate what the padding bytes used must look like in the event that their appearance is not defined by the transformation algorithm.

Padding Length

Specifies how many padding bytes have been inserted immediately before this field.

Next Header

This protocol field contains the protocol number of the protocol contained in the ESP header.

When the ESP is inserted, the sender takes this value from the corresponding protocol field of the header immediately before the ESP.

The recipient transfers the contents of this field to the protocol header before the ESP when the ESP header is removed.

Integrity Check Value (ICV)

The ICV field is optionally available when the two ESP communication partners have agreed to use the authentication service.

The sender enters the calculated value in this field. The ICV is calculated after the data is encrypted. The length of the checksum depends on the underlying cryptographic algorithm.

The Integrity Check Value calculated covers the ESP protocol and its data. The IP protocol and the ICV field are not included in the calculation.

4.7 Cryptographic methods

IPSec uses symmetric cryptographic methods exclusively in the Authentication Header and Encapsulating Security Payload security protocols. The sender and recipient apply an identical key to the data to be processed. The preshared secret key is a parameter of the security association agreed between the communication partners.

A number of transformation and signature algorithms are used in IPSec. Compliance with IPSec requires that certain algorithms are supported.

Which algorithms are to be supported is described in RFC 2401 and higher for IKEv1 and in RFC 4301 and higher for IKEv2.

4.8 Management of security associations and their keys

4.8.1 Manual management

The manual management of security services is carried out by system administrators, although they can also be assisted by machines. Manual management of security services and of the instances that execute them involves the following actions:

- Agreement on the security services to be provided
- Specification of the cryptographic methods to be used
- Creation of the keys to be used
- Installation of a security association
- Assignment of the Security Parameter Index of an SA
- Transmission of this information to the communication partner, which then has to take the appropriate action

The transmission and storage of secret keys are particularly security-sensitive.

Security associations set up manually do not offer replay protection, since replay protection requires a mechanism for key replacement in good time. This cannot be provided by the system administrator.

Nevertheless, every IPSec-compliant system must permit this kind of management. This requirement ensures that the use of security protection can be made compulsory for selected data transfers.

A security system never gets by entirely without manual intervention, although the management of security associations and the dynamic assignment of (protocol) resources and their distribution is carried out automatically by system instances. The security policies of a system are defined by the system administrator after consultation with colleagues. IPSec therefore requires an administration interface to the Security Policy Database.

4.8.2 Automated management

Automated management – i.e. the establishment and clearance of security associations, the negotiation of the attributes of the security association and key generation and distribution – is defined in three RFCs.

- RFC2408 – Internet Security Association and Key Management Protocol (ISAKMP)
  This provides an abstract framework for such functions. ISAKMP is not limited to IPSec; it can offer its services to any security system. ISAKMP is specified independently of concrete security mechanisms such as encryption methods and authentication methods. Any key exchange protocol can be used within ISAKMP, provided it has the required properties.
- RFC2407 – The Internet IP-Security Domain of Interpretation for ISAKMP
  This specifies the nomenclature, syntax and semantics of the protocol fields and their parameter values for the key management protocols (KMP) that provide the functions and language resources required by IPSec within ISAKMP.
- RFC2409 – The Internet Key Exchange (IKE)
  This is a protocol that meets the requirements of IPSec stipulated within the framework of ISAKMP.

4.8.2.1 Internet Security Association and Key Management Protocol (ISAKMP)

ISAKMP, RFC 2408, does not define a protocol regulating the exchange of key information between the systems involved. ISAKMP cannot be implemented as a protocol. Instead, ISAKMP provides the instances involved with a framework that allows them to agree on a key exchange method and the associated parameters. ISAKMP describes in abstract terms the requirements that have to be met by certain aspects of a concrete protocol in order to be considered a compliant protocol suitable for key exchange.

The abstract definition of a compliant protocol involves the syntactic and semantic description of the protocol elements (payloads), the protocol processes (exchanges), in which the information is exchanged, and the payload processing rules in the context of an exchange. It is defined what the proposals for a security service must look like and how one of these proposals has to be accepted or all of them have to be rejected. In addition to this protocol negotiation, processes are defined by means of which security services can be modified and terminated.

ISAKMP concerns itself with the issues involved in a secure method of exchanging key information, although it is independent of the cryptographic mechanisms to be used, the key generation method used and the key exchange protocol.

The key requirements of ISAKMP to be met by a key exchange protocol are as follows:

- It must be ensured that the instances involved can access the required preshared secret key information.
- It must be possible to check the origin and currency of the key information exchanged. In other words, it must be possible to authenticate the key information.
- The key information of different negotiations must be mutually independent. This prevents more than one communication relationship from being compromised if a secret key is uncovered. This is referred to as Perfect Forward Secrecy (PFS).
- In order to deal with the large number of potential communication partners, a mechanism must be available that enables secret key information to be created and exchanged between systems that have not installed a preshared secret key. This entails the integration of a public key infrastructure (PKI).

Preshared secret key information is only useful to the instances involved if there is a mutual agreement in place about the cryptographic algorithms in which the keys are to be used, their period of validity and how the communication partners reference the keys. The keys themselves and the attributes of the keys described above, together with the identities of the communication partners, can be viewed as a security association.

ISAKMP provides these services in two different negotiation stages known as phases.

Phase 1

In phase 1, the ISAKMP instances agree how the subsequent communication steps are to be secured within ISAKMP. During phase 1, the instances authenticate themselves. This ensures that information to be kept secret is not made available to unauthorized persons. In phase 1, a security association is established between the ISAKMP instances. This is referred to as an ISAKMP SA and managed by the instances themselves. The ISAKMP SA is bidirectional. In other words, the two instances have the same knowledge about the security parameters of the SA, and both instances can use the SA at any time. There can be a number of ISAKMP SAs between two instances. The role of the individual instance is specified implicitly for each ISAKMP SA in phase 1. The active instance plays the role of initiator, and the passive instance the role of responder. The protocol offers language resources for the identification of individual SAs. The roles of the instances have an important function in that they determine the sequence of the identifiers within the protocol elements.

Phase 2

Phase 2 is available for setting up security associations for other services. Under the ISAKMP SA protection mechanisms or the protection of phase 1, many security associations can be set up between the systems and services involved.

In phase 1, compute-intensive methods are used for the mutual authentication of the instances (asymmetric cryptographic methods) and to generate the keys for the ISAKMP SA. This is acceptable, however, because these communication steps do not have to be carried out very often. The much more frequently required negotiations about security services in phase 2, such as the creation of an IPSec SA, can then be conducted considerably more efficiently because they are protected by the ISAKMP SA.

Essentially, the negotiations about the security services to be provided within ISAKMP always revolve around the following:

- The security protocol by means of which the service is provided
- The cryptographic algorithms executed
- The key information together with the key attributes

Exchanges

For both phases, ISAKMP defines exchanges by means of which the services are provided. An exchange describes the number and sequence of the protocol elements and the syntax, semantics and sequence of the language resources (payloads) within the protocol elements. The payloads are combined by an ISAKMP protocol. The ISAKMP protocol contains the identifiers of the SA under which the exchange is to be conducted. ISAKMP defines five standard exchanges:

- Base exchange
  This comprises four messages and permits key exchange and authentication.
- Identity protection exchange
  This comprises six messages. The key exchange takes place before the exchange of the identification and authentication data, which allows this data to be protected.
- Authentication only exchange
  This comprises three messages and provides only one authentication service without encryption.
- Aggressive exchange
  This comprises three messages, offers key exchange, ISAKMP SA establishment without a negotiation option and authentication.
- Informational exchange
  This comprises a single message and transmits SA-specific information to the partner instance.

ISAKMP can use any transport protocol. However, UDP support on port number 500 must be provided.

4.8.2.2 The Internet Key Exchange (IKE)

The Internet Key Exchange (IKE, RFC2409) protocol is a frequently used protocol with the required mechanisms for the automated management of IPSec security associations. IKE makes use of the IPSec DOI and meets the demands placed on key exchange and management protocols by ISAKMP.

IKE connects components of two other security association and management protocols – the Oakley protocol and the SKEME protocol – and maps them to the generically defined ISAKMP. IKE thus uses the concepts of all protocol specifications.

IKE sets up an ISAKMP SA either in a main mode negotiation or an aggressive mode negotiation. These are both phase 1 exchanges. Main mode is an identity protection exchange, while aggressive mode is an aggressive exchange.

The creation of the secret key record is based on the Diffie Hellman method, regardless of the mode selected. Four Diffie-Hellman groups are predefined. Other groups can be negotiated. The algebraic details have to be specified in these negotiations.

IKE supports four different authentication methods:

- Preshared secret key (PSK)
- Public key signature
- Public key encoding
- Revised public key encoding

The authentication data is created in accordance with the authentication mechanism selected. Certain fields of the protocol elements are also interpreted in accordance with the authentication mechanism.

Regardless of that, the same data is taken into account in each method when the authentication data is calculated.

Other algorithms and methods are also permissible and can be negotiated between the IKE instances.

IKE introduces a new exchange for phase 2: quick mode. Quick mode comprises three protocol elements. The proposals in the protocol apply to IPSec SAs that are to be set up. The security protocols are thus ESP and AH. The transformations or signature methods must be supported by the IPSEC protocols. The protocol type and Security Parameter Index (SPI) are included in the calculation of the key for the transformation. The initiator and responder both determine an SPI. Since the SPI is included in the key calculation method, the keys of different IPSec SAs are mutually independent. IKE thus forms the PFS property for IPSec.

4.8.2.3 Internet Key Exchange Protocol Version 2 (IKEv2)

In version 1 of the Internet Key Exchange Protocol (IKEv1) which has been supported to date, weaknesses were noticed with regard to reliability, effectiveness and uniqueness of the protocol definition. "Internet Key Exchange Protocol Version 2" performs the same functions as IKEv1, in other words mutual authentication of the protocol instances involved and setup and administration of security associations.

IKEv2 is defined in RFC4306. RFC4306 incorporates the IKEv1 RFCs 2407, 2408, 2409 and a number of other RFCs. These include RFC3748 "Extensible Authentication Protocol (EAP)", RFC3715 "IPSec Network Address Translation (NAT) Compatibility Requirements" and RFC3948 "UDP Encapsulation of IPSec ESP Packets". Protocol version 2 was created with the intention of eliminating the weaknesses of version 1, while making the protocol simpler and ensuring greater flexibility in its practical application. Although the basic functions of the two protocol versions are the same, IKEv2 is not compatible with IKEv1. This is reflected in the number, content and semantics of the protocol elements of protocol phases 1 and 2 and in the new nomenclature selected. Both protocol versions are processed by protocol instances which are identified by the UDP port number 500. RFC4306 defines the rules specifying how the protocol instances must agree on a version.

4.8.2.4 Changes in IKEv2 compared to IKEv1

IKEv2 includes a number of changes compared to IKEv1 in order to achieve the goals set. Some principal aspects are described below.

Compatibility

The IKE protocol header contains the two fields "major version number" and "minor version number" from which the protocol instances can recognize which protocol version and which functionality of this version the partner concerned supports. The "minor version number" is used solely to display the functionality. The instance which supports a lower version number must ignore the partner's higher version number. The instance which supports a higher version number detects the partner's lesser functionality and takes this into account for the duration of the connection. This, for example, prevents informational exchanges from being initiated which the partner cannot understand.

Versions with different "major version numbers" are incompatible. The "major version number" indicates the protocol version of the current packet. A display field also exists in which the sender can specify that it supports a higher protocol version. If, on the basis of this "I could handle a higher version" flag, the protocol instances recognize that it is possible to agree on a higher protocol version, the current connection setup is aborted and a new connection attempt is started with a higher version. However, if a responder receives a request with a higher version number than the highest it supports, it may not process the message. It should inform the initiator of this, and also provide notification of the highest acceptable version.

The rules for negotiating the protocol version apply from IKEv2 (RFC4306) and are designed to help introduce compatible IKE protocol versions in the future. The negotiation between IKEv1 and IKEv2 instances is not defined. Not only is the flag which enables support of a higher version missing in IKEv1: the highest version supported cannot be specified in the informational exchange with which a request is also rejected because its version number is too high. One possible way of bypassing this migration problem would be to initially send a request with "major version number 2" and, after a monitoring time has elapsed or after a negative response has been received, to repeat the request with "major version number 1" if this is permitted by the system administrator. Otherwise the protocol version which must be used for the various partner instances and which may be accepted has to be configured.

As a further protocol language resource for future enhancements, each payload header contains a "critical flag" which the sender uses to indicate whether the content of the payload may be ignored. If an IKEv2 packet is received which contains an unknown payload with a "critical flag", the entire packet may not be processed. This ensures that protocol enhancements are possible within a version.

Reliability

Information is always transported in request protocol elements by the initiator of a dialog, and the requestor must always answer these. Responses are assigned to the initiating request by means of a sequence number which must be mirrored.

If the initiator does not receive an acknowledgment within a defined period, the request is repeated until the action is terminated (successfully or unsuccessfully).

The protocol sequence number permits multiple actions to be executed simultaneously.

Lifetime of an SA

When the protocol instances must regard their connection partner as unavailable and how such situations can be detected (Dead Peer Detection) is defined in IKEv2. IKEv1 does not provide Dead Peer Detection. However, RFC3706 "A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers" suggests how the availability of a partner can be detected in IKEv1. DPD enables the connection partners to delete the IKE-SA and its subordinate CHILD-SAs at any time. System restarts are therefore possible at any time as it is guaranteed that the connection partner recognizes this and takes defined measures to ensure both sides are synchronized.

Rekeying

Each connection partner is permitted to rekey an SA at any time. The lifetime of an SA can be defined by the protocol instances independently of each other. No provision is made in IKEv2 for negotiating or comparing the SA lifetime. The procedure for updating an obsolete key is described precisely. Rekeying is performed in IKEv2 by replacing an SA with a newly negotiated SA.

Semantic simplification of the protocols

The parts of an IKEv2 message which need to be encrypted are transferred as data of the encrypted payload. The encrypted payload must be the last payload of an IKEv2 protocol. It contains an additional signature field. The signature extends over the entire protocol, including the IKEv2 header. The layout of the encrypted payload header is very similar to that of an ESP trailer.

In IKEv1 the header was used as the basis for deciding whether the data following the header is encrypted. Unencrypted protocol language resources are not possible. The signature for an integrity check is transferred in a special (HASH) payload, which is also encrypted.

The agreement of the partner instances regarding which data streams are to be protected by an IPSec SA is rigid and inflexible in design. IKEv1 does allow IP address ranges to be specified, but when the SA is set up, the responder can only accept or reject the addresses proposed by the initiator. No further selection or restriction is possible. Port number ranges for the TCP and UDP protocols or more precise specification of ICMP messages cannot be presented. This means that successful SA negotiation requires greater configuration effort in the initiator and responder systems. On the other hand, it provokes a higher error rate in the configurations.

In IKEv2 the communication partners are provided with the opportunity of genuinely negotiating the data traffic which is to be protected. The protocol language resource "traffic selector (TS) payload" enables the initiator of a security association to specify a list of protocol/port/IP address descriptions; port and address ranges can also be specified. The responder is allowed to compare the received TS list with the local configuration. Additional language resources in the IKEv2 protocol (such as information that further address ranges are accepted or that only one single address pair is supported) increase the chances of the negotiation succeeding.

Negotiation of a security association is simpler and consequently more efficient in design in IKEv2. The initiator describes each SA which is to be negotiated by means of a list of proposals, the proposal payloads. Proposal payloads are subordinate to the security association payload. Proposal payloads outside an SA payload are not permitted. Each proposal contains a list of required security protocols (e.g. AH, ESP), and a selection of encryption and signature algorithms is offered for each security protocol. These algorithms are

described in transform payloads, which are in turn subordinate to a proposal payload. The responder is permitted to select one of the proposals, for each security protocol defining the transformation algorithm which it considers to be the most suitable. One special feature here is that a list of encryption and signature algorithms from which the responder can select one of each can be specified for the IPSec protocol ESP. If none of the proposals is accepted, the security association must be rejected.

Although the content of an SA payload in IKEv1 is very similar to that in IKEv2, it is interpreted very differently. In IKEv1 the proposal and transform payloads are autonomous structures next to the security association payload. In IKEv1 each encryption algorithm must be combined with each signature algorithm for the IPSec protocol ESP and entered in a separate transform payload. The number of transformations therefore increases exponentially with the number of algorithms.

The IP payload compression protocol (RFC3173) loses its status as an IPSec protocol in IKEv2. In IKEv2, in contrast to IKEv1, IPCOMP is consequently not negotiable as an individual protocol or as a protocol of a protection suite (of an SA bundle). Support of IPCOMP together with the compression algorithms must be indicated in IKEv2 by means of a notify payload. Although IPCOMP is an autonomous protocol, no construct which is comparable with a security association exists in IKEv2. Rather, an "IPCOMP connection" disappears with the CHILD-SA with which it was set up.

Initial exchanges

The initial exchanges (IKE-SA-INIT and IKE-AUTH) correspond to phase 1 in ISAKMP. In IKEv1 the sequence of the protocol elements required is referred to as Main Mode, and its abbreviated version as Aggressive Mode. Phase 1 is used by the IKE protocol instances for mutual authentication and as a rule leads to an IKE SA being set up.

The initiator of an IKEv2 SA sends an IKE-SA-INIT request message to the partner, who answers with a valid IKE-SA-INIT to permit a successful exchange. During this dialog agreements are made regarding the cryptographic method to be used, the initiator and responder nonces to be exchanged and the Diffie-Hellman values to be transferred. This information is employed to generate key material and is used in the following IKE-AUTH exchange, consisting of an IKE-AUTH request and an IKE-AUTH response. The IKE-AUTH dialog authenticates the messages of the IKE-SA-INIT exchange and thus also implicitly authenticates the partner instances. Some of the content of the IKE-AUTH messages is encrypted, as a result of which the identity of the partners for which a CHILD-SA is to be set up already remains secret. A CHILD-SA can already be set up in the IKE-AUTH exchange.

IKE-SA and CHILD-SA in IKEv2 are equivalent to ISAKMP-SA and IPSEC-SA respectively in IKEv1.

CREATE-CHILD-SA exchange

The CREATE-CHILD-SA dialog is used to negotiate a CHILD-SA. Functionally it is comparable to phase 2 in ISAKMP. This dialog permits a CHILD-SA whose lifetime has been exceeded or will shortly be exceeded to be replaced by a new CHILD-SA with the same selectors. The CHILD-SA is made subordinate to the IKE-SA under whose (cryptographic) protection it was negotiated. If the CHILD-SA to be replaced still exists, it must be deleted using an informational exchange.

A CHILD-SA can only exist if an IKE-SA exists. Rekeying of an IKE-SA consequently means that first a new IKE-SA must be set up which will inherit all the CHILD-SAs of the IKE-SA to be replaced, and then the IKE-SA to be replaced must be deleted.

In IKEv1 the corresponding exchange is referred to as Quick Mode.

Informational exchange

Control messages which are used to receive existing SAs or to display an error are exchanged in informational exchanges. In contrast to IKEv1, they also include a request/response dialog. The messages are encrypted, i.e. they are sent under the protection of the IKE-SA for which they provide a service. If the dialog concerns a CHILD-SA, the messages are protected by the IKE-SA to which the CHILD-SA is subordinate.

An SA is deleted by means of an informational exchange containing a DELETE payload.

Informational exchanges enable the IKE instances involved to detect whether particular partners are accessible (Dead Peer Detection). A request with an encrypted "empty" payload is sent for this purpose. If, after several repetitions, the sender does not receive a valid response, it regards the partner as not accessible and deletes the IKE-SA and the associated CHILD-SAs.

4.8.2.5 How IKE works (IKEv1 and IKEv2)

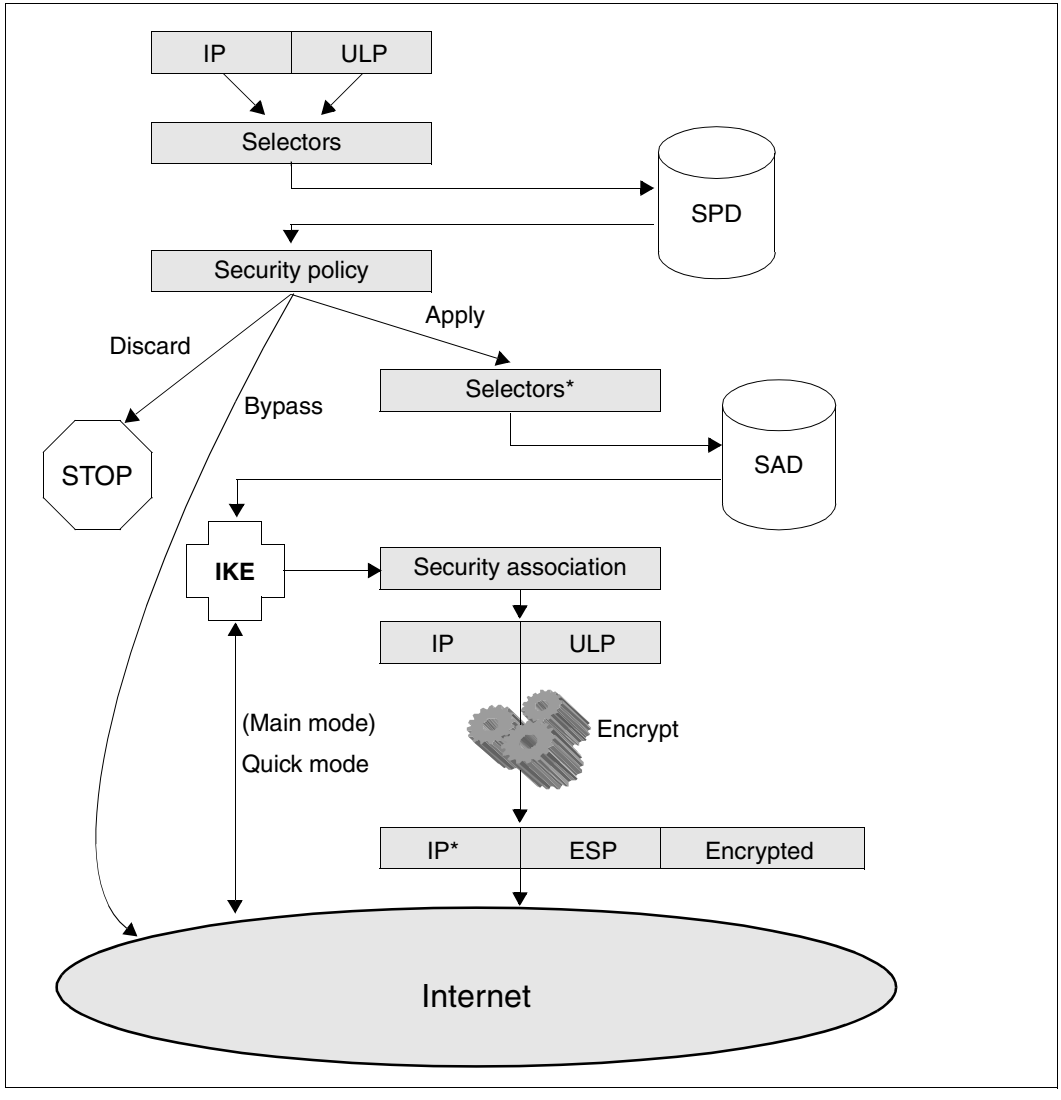


Figure 11: Interaction of the components with IKE

If a security policy for an outbound IP segment stipulates the use of IPSec services but does not stipulate a suitable security association, an appropriate security association is established using the Internet Key Exchange protocol. The IP segment to be sent is buffered in IPSec for the duration of the IKE main/quick mode or Initial-/ Child-SA negotiation. After the successful installation of the outbound SA, the IP segment is processed in accordance with the SA.

5 Implementation of IPSec in BS2000/OSD

The IPSec functionality is implemented by subsystems in BS2000/OSD. The figure below provides an overview of how IPSec is integrated in BS2000/OSD.

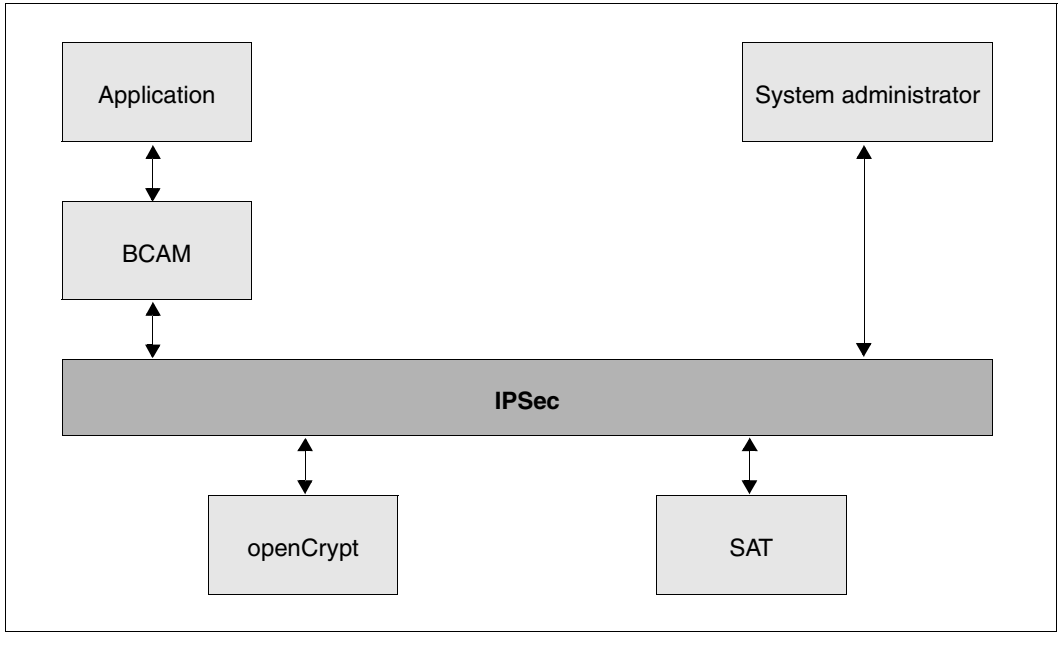


Figure 12: IPSec subsystem in the BS2000/OSD environment

Security Audit Trail (SAT)

The BS2000/OSD functional unit SAT (Security Audit Trail) logs all security-related IPSec events. SAT, which is part of SECOS, is the BS2000/OSD logging component for security-related events. These events are stored in SAT logging files (SATLOG) and can be analyzed using the SATUT utility.

Events that are particularly critical to security can be monitored without delay with the aid of the SAT alarm function. The operator console displays the alarm messages so that appropriate action can be taken. You will find more information on SAT in the SECOS (BS2000/OSD) User Guide.

IPSec logs the following events:

- Successful loading of the IPSec database
- Failure to load the IPSec database
- Infringement of the security policy during data transfer

In the event of an infringement of a security policy, the following data is transferred to SAT:

- Own IPv4 address
- Partner's IPv4 address
- Own IPv6 address
- Partner's IPv6 address
- Error code

The assignment of reported data to events is described in the SECOS (BS2000/OSD) User Guide.

When infringements of a security policy occur during data transfer, besides being logged in SAT they are also reported in messages on the operator console.

openCrypt

No cryptographic algorithms are implemented in the IPSec subsystem. The required cryptographic services are provided by the product openCRYPT (see the openCRYPT V1.2 (BS2000/OSD) manual).

Components of IPSec

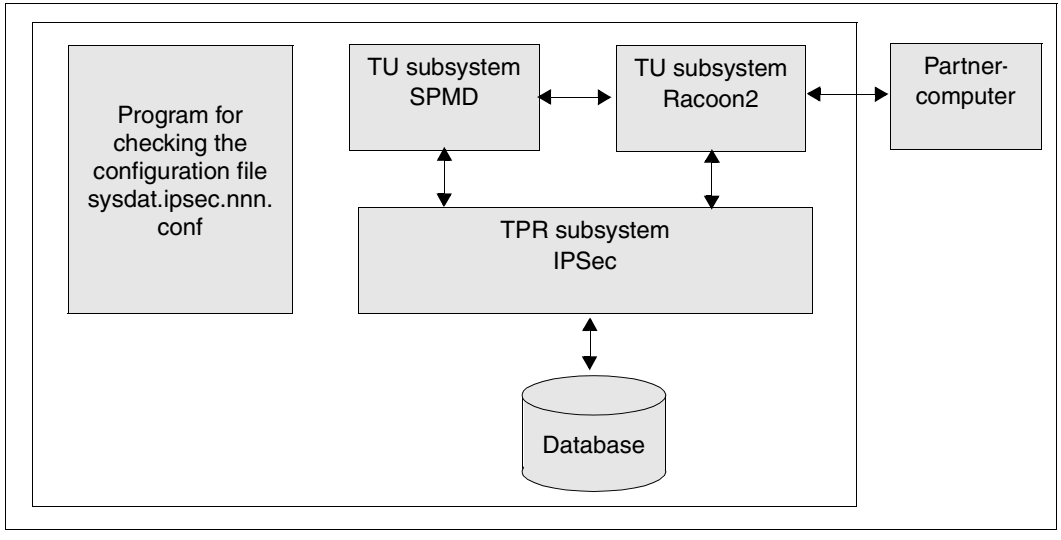


Figure 13: Key components of IPSec

Database

The IPSec database is stored in the virtual address space. It comprises the Security Association Database (SAD) and Security Policy Database (SPD). The SAD and SPD are described in detail in the sections “Security policy” on page 39 and “Security association” on page 41.

When the IPSec subsystem starts up, the SPD and the static (manually configured) part of the SAD are created from the entries in the IPSec configuration files. The IPSec database can be changed at any time during IPSec operation by means of commands which cause the configuration files to loaded again.

The security association database is managed by the IKE daemon. The security policy database is managed by the SPMD program.

IPSec subsystem

All access to the SAD and SPD is provided by the IPSec subsystem. It also handles all security protocols.

Racoon2 subsystem

Racoon2 is a program which implements the Internet Key Exchange Protocol Version 1 and Version 2, see section “Automated management” on page 57. Racoon2 automates the setup of SAs and the exchange of keys between the home and partner computers.

SPMD subsystem

SPMD is a program which manages the local security database. When the IPSec subsystem is started, the SPs defined in the configuration file SYSDAT.IPSEC.nnn.RAC2 are made known to the TPR subsystem IPSEC. Via a communications interface SPMD offers the IKE daemon information on the actions to be executed.

6 IPSec configuration and operation

6.1 IPSec installation

IPSec is installed using IMON (**I**nstallation **Mon**itor). It is advisable to install IPSec in standard installation mode (i.e. on the system standard ID).

IMON installs the following files:

Subsystem catalog:	$TSOS.SYSSSC.IPSEC.*nnn*
Subsystem library:	$TSOS.SKMLNK.IPSEC.*nnn* $TSOS.SPMLNK.IPSEC.*nnn* $TSOS.SYSLNK.IPSEC.*nnn* $TSOS.SYSLNK.IPSEC.*nnn*.RAC2
Program library:	$TSOS.SYSPRG.IPSEC.*nnn*
Command syntax file:	$TSOS.SYSSDF.IPSEC.*nnn*
Message file:	$TSOS.SYSMES.IPSEC.*nnn*
SSINFO file:	$TSOS.SYSSSI.IPSEC.*nnn*
IMON file:	$TSOS.SYSSII.IPSEC.*nnn*
IPSec configuration file:	$TSOS.SYSDAT.IPSEC.*nnn*.CONF
IPSec configuration file:	$TSOS.SYSDAT.IPSEC.*nnn*.RAC2

nnn stands for the version of the BS2000/OSD installation unit IPSEC (e.g. 014 for version 1.4).

6.2 Brief description of startup

Following successful installation, the IPSec subsystem can be started. Here IPSec loads the default configuration file (SYSDAT.IPSEC.nnn.CONF) and starts the IKE daemon Racoon2 and the Security Policy Management Daemon SPMD. The two programs load their joint configuration file (SYSDAT.IPSEC.nnn.RAC2). All communication takes place unprotected, in accordance with the standard rule implemented, so IPSec remains invisible.

The IPSec services do not come into effect until a number of preparatory measures have been taken. First the communication relationships for which IPSec services are to be used must be defined. In other words, policies have to be defined. A policy specifies, among other things, how IPSec security associations are to be created (automatically or manually).

The following example illustrates what has to be done in order to be able to use IPSec services. You will find detailed information on each step in the sections that follow.

1. Definition of a policy
   The $DIALOG connections of a PC have to be secured by encryption and authentication.
   The default policy is entered explicitly in the static configuration file. Data transfer to other partners is permitted by means of this policy.

2. Selection of a method
   The security protocol and encryption algorithms selected depend on which ones are supported by the two IPSec partner systems. The system administrators also have to reach agreement.
   For the purposes of this example, it is assumed that both systems support automatic key exchange, the 3DES-CBC encryption method and the HMAC-SHA-1 authentication method.
   Automatic key exchange is always preferable to manual exchange because it greatly reduces the administration effort required and the key information is transferred securely.

3. Creation of the IPSec configuration files
   The configuration file SYSDAT.IPSEC.*nnn*.CONF in the example is as follows:

```
*
* definition of POLICY records
*
POLICY NAME=THEOTHERS -
,POLICY-RANGE=GLOBAL -
,OWN-ADDRESS=ANY -
,OWN-PORTNUMBER=ANY -
,DIRECTION=ANY -
,PROTOCOL=ANY -
,ICMP-TYPE=ANY -
,ICMP-CODE=ANY -
,MODE = BYPASS
```

The configuration file SYSDAT.IPSEC.*nnn*.RAC2 in the example is as follows:

```
#
# include of the default configuration file with the  default parameter
#
include "SYSDAT.IPSEC.nnn.RAC2.MANUAL-BSP-DEFAULT" ;

# PC-1
# Describes the KMP capabilities of PC-1
# Parameters of the default section are overriden.
#
remote PC-1 {
   acceptable_kmp { ikev1; };
   ikev1 {
      my_id ipaddr 172.25.92.72;
      peers_id ipaddr 172.25.83.42;
      peers_ipaddr 172.25.83.42 port 500;
      kmp_enc_alg { 3des_cbc;};
      kmp_hash_alg { sha1; md5;};
      kmp_dh_group { modp1024; };
      kmp_auth_method { psk; };
      pre_shared_key "PC-1.psk";
   };

};
#
# $DIALOG
# Selector and policy definition for the communication to $DIALOG
#
selector DIAL-in {
   direction inbound;
   src 172.25.83.42;
   dst 172.25.92.72 port 1110;
   upper_layer_protocol "tcp";
   policy_index PC-1;
};
selector DIAL-out {
   direction outbound;
   dst 172.25.83.42 ;
   src 172.25.92.72 port 1110;
   upper_layer_protocol "tcp";
   policy_index PC-1;
};
#
# Requested is IPsec protection with protocol ESP in end-to-end mode.
# Encryption and integrity protection is required.
#
policy PC-1 {
```

```
   action auto_ipsec;
   remote_index PC-1;
   ipsec_mode transport;
   ipsec_index { ipsec_esp; };
   ipsec_level require;
};

ipsec ipsec_esp {
   ipsec_sa_lifetime_time 28800 sec;
   sa_index esp_01;
};

sa esp_01 {
   sa_protocol esp;
#  esp_enc_alg { aes128_cbc; 3des_cbc; };
#  esp_auth_alg { hmac_sha1; hmac_md5; };
   esp_enc_alg {  3des_cbc; };
   esp_auth_alg { hmac_sha1; };
};
```

The configuration file can be saved with the name SYSDAT.IPSEC.nnn.RAC2 .

IPSec can only be configured by means of selectors derived from protocols belonging to the IP suite. For $DIALOG it is necessary to specify the TCP port number 1110. The PC administrator also requires this information. The RFC1006 port number 102 continues to provide unencrypted access to the system.

4. Creation of the PSK file for the partner computer
   Automatic key exchange is implemented by means of the IKE protocol. The two IKE instances (programs) on the partner computers involved have to authenticate themselves. The preshared secret keys (PSKs) method is used for this purpose in BS2000. A separate file in which the PSK is entered should be created for each partner computer. For this example the entry in the PC-1.psk file looks like this:

   ```
   =)(shbwldqU&$RQA
   ```

   The information indicating that PSK is to be used as the authentication method and which key is to be used must be made available to the PC administrator.

5. Configuration of the partner computer
   Analogous configuration steps also have to be performed on the partner computer. Identical configurations must be created.

6. Starting IPSec
   You start the IPSec subsystem by entering the following command:

   ```
   /START-SUBSYSTEM IPSEC
   ```

   The SYSDAT.IPSEC.nnn.CONF file is loaded during the startup. This file contains the static policy and SA definitions. The SPMD program is started. SPMD loads the configuration file SYSDAT.IPSEC.nnn.RAC2 and complements the security policy database by adding the security policies described in the file.
   SPMD then starts the Racoon2 program, which also loads SYSDAT.IPSEC.nnn.RAC2 file. SPMD and Racoon2 consequently have the same SPD.

7. Establishment of a connection
   An encrypted dialog connection is established provided the destination port number 1110 is used in the emulation.

8. Checking of the connection
   To check whether IPSec is being used for the $DIALOG connection to the PC, you can enter the SHOW-CONNECTION command with appropriate IPSEC selection parameters and specify the processor name.

   Let's assume you enter the following command:

   ```
   /SHOW-CONNECTION SELECT=*BY-ATTRIBUTES(IPSEC=*YES)
   ```

   Message `BCA08A2` then has to appear. This describes a connection to the partner computer and contains the information IPSEC=YES.

6.3 IPSec configuration

The functionality of IPSec, the monitoring of data transfer and the use of security services on specific IP segments are checked and controlled using the Security Policy Database (SPD) and Security Association Database (SAD). The databases are created and managed in BS2000/OSD by means of several configuration files. During the installation of IPSec, the blank default file SYSDAT.IPSEC.*nnn*.CONF is created and loaded when the subsystem starts up. The database is also manipulated during IPSec operation by means of configuration files.

Details can be found in section “Changing the IPSec configuration” on page 107.

CAUTION!

The IPSec configuration files contain security-related information and must therefore be protected against unauthorized access. The file access permissions should be set to USER-ACCESS = *OWNER-ONLY.
SPMD and Racoon2 work with the same configuration file. Manipulation of the database during IPSec operation must therefore always take place for both programs.

6.3.1 Processing the files for the static configuration

IPSec configuration files in BS2000/OSD generally contain statements and object definitions. Statements control the sequence in which objects are processed and how they are processed. Object definitions create identifiers, establish relationships between them and thus describe what is manipulated in the database. Objects are identified by names.

The processing of a set of configuration files is subject to a number of **rules**.

- Statements are executed immediately.
- The applicability of a statement is limited to a single configuration file.
- The object names must be unique, since objects outside the defining configuration files must be visible.
- If an object references another object, the referenced object must already be known. The referenced object can be defined in another configuration file loaded previously.
- The sequence in which the object definitions occur is not subject to any further constraints. It does not affect the organization of the database.

The following **language resources** are available for creating and manipulating the IPSec database.

- The following statements are available for processing the contents of a configuration file:
  - FLUSH
  - INCLUDE
  - ADD
  - DELETE
- There are five objects available for defining encryption algorithms, signature methods, security associations and security policies. They correspond to the following configuration records:
  - KEY
  - SIGNATURE
  - SECURITY-ASSOCIATION
  - POLICY
  - PARTNER-SAS

The POLICY object is conceptually the most important language resource for creating the IPSec database in BS2000/OSD. The POLICY determines what is to be done with individual IP segments. For this purpose there are references in the definition record to the corresponding SECURITY-ASSOCIATION records. The POLICY specifies the **selectors** for IP segment filtering. The selectors for the referenced security associations are derived from the selectors of the policy in manual key management. The Security Policy Database obtains the prescribed organization from the defined selectors of the policies.

A number of **other rules** apply to the processing of the defined security policies of a set of configuration files in BS2000/OSD:

- No security labels or name selectors such as DNS or X.500 names are supported.
- The specification of wildcards or ranges is not supported.
- Range overlapping is supported.
- More precisely defined policies are found before less precisely defined policies when the SPD is searched. The number of explicit or implicit ranges specified is used to measure this precision.
- Policies with different names but identical selector definitions are detected and rejected.

Security associations tagged by manual key management are created when the IPSec subsystem starts up and entered in the SAD. Selectors determine how the SAD is organized. The type of the security protocol (AH or ESP) and the Security Parameter Index (SPI) serve as additional criteria for making distinctions. This is necessary for the following reasons:

- A POLICY object can reference more than one SECURITY-ASSOCIATION object to define an SA bundle. If only the selector definitions were taken into account, it would not be possible to distinguish between the security associations adequately.
- In automated key management, security associations are created automatically before the defined lifetime of an SA that is being used expires. This avoids unnecessary data transfer delays. These SAs can be distinguished on the basis of their SPI.
- To search for an SA for an inbound IP segment protected by IPSec, only the IP addresses, the security protocol and the SPI can be evaluated. The selectors that describe the end-to-end relationship of the communication partners cannot be identified at the time of receipt.

BS2000/OSD supports the profiles required by the security architecture (i.e. end-to-end security associations in transport and tunnel mode and security associations between security gateways in tunnel mode). IPSec does not check the addresses specified, so it cannot check whether the endpoint of a security association in tunnel mode is a host system or a security gateway.

- The INCLUDE statement makes it possible to describe the IPSec database in easily manageable blocks. Thus, the secrets required for key management can be stored in separate files. The security policies for individual addresses or address ranges can be kept in separate files.
- The DISCARD policy is suitable for creating a host-system firewall, which can be defined precisely enough to specify individual port numbers and IP addresses and which prevents access to the system or transfers from the system.
- Due to the lack of support for name selectors, no security policies can be defined for applications addressed via TCP port number 102. The BCMAP command in BS2000/OSD offers a way of working around this (see the preceding example).

6.3.2 Control statements for the IPSec configuration file

The IPSec configuration file is a SAM file with the following record format:

- The maximum record length is 255 characters.
- The * character introduces comments.
- The last position in a record is reserved for the continuation character (-).

FLUSH: Delete an existing configuration

The FLUSH statement deletes the existing configuration. The FLUSH statement and a configuration file without an object definition or a statement produce the same result.

Since the FLUSH statement is also executed when it is not at the beginning of an IPSec configuration file, it should be used with care.

The FLUSH statement must have a line to itself.

FLUSH

INCLUDE: Load an additional configuration file

The INCLUDE statement loads a further IPSec configuration file. The defined records are processed synchronously. In other words, records referenced by records to be included must have been loaded already. The INCLUDE statement can be used recursively. It allows the configuration description to be managed in files of a reasonable size. In particular, the INCLUDE statement allows key values and policies to be managed separately.

INCLUDE <filename 1..54>

Operand description

<filename 1..54>
Name of the configuration file to be included

ADD: Change the processing mode to ADD

The ADD statement changes the processing mode. The object definitions that follow the ADD statement create objects in the IPSec database. The ADD statement is limited to the configuration file in which it is located. It applies until the next DELETE statement in the same configuration file or until the end of the file.

The ADD processing mode is the default setting for every configuration file. Configuration files can be loaded one after the other or nested by means of the INCLUDE statement.

The ADD statement must have a line to itself.

ADD

DELETE: Change the processing mode to DELETE

The DELETE statement changes the processing mode. The object definitions that follow the DELETE statement delete objects from the IPSec database. The DELETE statement is limited to the configuration file in which it is located. It applies until the next ADD statement in the same configuration file or until the end of the file.

The object definitions of the objects to be deleted only require the keyword NAME. (See the description of the configuration records KEY, SIGNATURE, SECURITY-ASSOCIATION and POLICY.)

Objects with a valid NAME definition are deleted when other parameter definitions of the object have syntax errors.

The DELETE statement must have a line to itself.

DELETE

KEY: Define encryption algorithm

The KEY record defines a new encryption method within the IPSec database.

KEY
KEY NAME = <alphanum-name 1 .. 32> **,KEY-ALGORITHM = DES-CBC / 3DES-CBC / AES-CBC** **,KEY-VALUE =** <x-string 16 .. 16> / <x-string 32 .. 32> / <x-string 64 .. 64>

KEY
KEY NAME = <alphanum-name 1 .. 32> **,KEY-ALGORITHM = NULL**

Operand description

NAME= <alphanum-name 1 .. 32>
Name of the key. The key name can consist of letters and digits. A distinction is drawn between uppercase and lowercase letters. The first character in the name must be a letter.

KEY-ALGORITHM =
Specifies the encryption algorithm to be used.

KEY-ALGORITHM = DES-CBC
The DES algorithm is used in CBC mode.

KEY-ALGORITHM = 3DES-CBC
The TripleDES algorithm is used in CBC mode.

KEY-ALGORITHM = AES-CBC
The AES algorithm is used in CBC mode.

KEY-ALGORITHM = NULL
No encryption algorithm is used.

KEY-VALUE =
Specifies the key to be used.

KEY-VALUE = <x-string 16 .. 16> / <x-string 32 .. 32> / <x-string 48 .. 48> / <x-string 64 .. 64>
A string of hexadecimal characters enclosed in single quotes (‘) and preceded by x or X (e.g. X’01’ or x’12AB’)

CAUTION!

The permissible key length depends on the encryption algorithm used:

DES-CBC:	16 hexadecimal characters
3DES-CBC:	32/48 hexadecimal characters
AES-CBC:	32/48/64 hexadecimal characters

If one line is not enough for the key, one or more continuation lines can be used. To do this, write to the end of the line to be continued and start the continuation line in column 1 by entering the next character of the key.

SIGNATURE: Define signature method

The SIGNATURE record defines a new signature method in the IPSec database.

SIGNATURE
SIGNATURE NAME = <alphanum-name 1 .. 32> **,SIGNATURE-ALGORITHM = HMAC-MD5** / **HMAC-SHA-1** **,SIGNATURE-VALUE =** <x-string 32 .. 32> / <x-string 40 .. 40>

SIGNATURE
SIGNATURE NAME = <alphanum-name 1 .. 32> **,SIGNATURE-ALGORITHM = NULL**

Operand description

NAME = <alphanum-name 1 .. 32>
Name of the signature method. The name can consist of letters and digits. A distinction is drawn between uppercase and lowercase letters. The first character in the name must be a letter.

SIGNATURE-ALGORITHM =
Specifies the signature algorithm to be used.

SIGNATURE-ALGORITHM = HMAC-MD5
The MD5 algorithm is used.

SIGNATURE-ALGORITHM = HMAC-SHA-1
The SHA-1 algorithm is used.

SIGNATURE-ALGORITHM = NULL
No signature algorithm is used.

SIGNATURE-VALUE =
Specifies the key to be used to create the signature.

SIGNATURE-VALUE = <x-string 32 .. 32> / <x-string 40 .. 40>
A string of hexadecimal characters enclosed in single quotes (‘) and preceded by x or X (e.g. X’01’ or x’12AB’)

CAUTION!

The permissible key length depends on the signature method used:

HMAC-MD5: 32 hexadecimal characters

HMAC-SHA-1: 40 hexadecimal characters

If one line is not enough for the key, one or more continuation lines can be used. To do this, write to the end of the line to be continued and start the continuation line in column 1 by entering the next character of the key.

SECURITY-ASSOCIATION: Define security association

The SECURITY-ASSOCIATION record defines the properties of a security association.

SECURITY-ASSOCIATION

SECURITY-ASSOCIATION NAME = < alphanum-name 1 .. 32>

,TYPE = AUTHENTICATION (...) / **ENCRYPTION (...)**

AUTHENTICATION (...)
SIGNATURE = <alphanum-name 1 .. 32>

ENCRYPTION (...)
SIGNATURE= <alphanum-name 1 .. 32> / ***NONE**
, KEY=<alphnum-name 1 .. 32> / ***NONE**

,**MODE = MANUAL (...)**

MANUAL (...)
INDEX = <integer 256 .. 4294967295>

[**,ECN-TUNNEL = ALLOWED** / **FORBIDDEN**]

[**,ECN-TUNNEL-NEGOTIATION = YES** / **NO**]

Operand description

NAME = <alphanum-name 1 .. 32>
Name of the security association record. The name can consist of letters and digits. A distinction is drawn between uppercase and lowercase letters. The first character in the name must be a letter. The name is used in the SECURITY-POLICY record to reference the SECURITY-ASSOCIATION record to be used.

TYPE = AUTHENTIFICATION (...) / ENCRYPTION (...)
Specifies the type of the security association.

AUTHENTICATION (SIGNATURE = ...)
Authentication is performed with an Authentication Header (AH).

SIGNATURE = <alphanum-name 1 .. 32>
Name of the signature method used. This signature method must already be defined.

ENCRYPTION (SIGNATURE = ... , KEY= ...)
Encryption and/or authentication is performed with an Encapsulated Security Payload (ESP) header.

SIGNATURE = <alphanum-name 1 .. 32> / *NONE
Name of the SIGNATURE record used. This SIGNATURE record must already be defined. *NONE means that no authentication is carried out.

KEY = <alphanum-name 1 .. 32> / *NONE
Name of the KEY record used. This KEY record must already be defined.
*NONE means that no encryption is carried out.

MODE = MANUAL (INDEX = ...)
A static SECURITY-ASSOCIATION is created when the configuration file is loaded.

INDEX = <integer 256 .. 4294967295>
Specifies the Security Parameter Index (SPI), which, in the case of inbound AH and ESP headers, identifies the security association defined by the current SECURITY-ASSOCIATION record. The range <integer 0 .. 255> is reserved for internal use (RFC 2406).

The two subsequent operands must be defined when the security association is used for an IP tunnel.

ECN-TUNNEL = ...
Controls the use of ECN tunnels in accordance with RFC 3168. This parameter is only required when the security association is used for an IPSec tunnel.

ECN-TUNNEL = ALLOWED
The use of ECN tunnels in accordance with RFC 3168 is permitted.

ECN-TUNNEL = FORBIDDEN
The use of ECN tunnels in accordance with RFC 3168 is not permitted.

ECN-TUNNEL-NEGOTIATION = ...
Specifies whether the use of ECN tunnels can be negotiated with the IPSec partner. This parameter is only required when the security association is used for an IPSec tunnel.

ECN-TUNNEL-NEGOTIATION = YES
The use of ECN tunnels in accordance with RFC 3168 can be negotiated with the IPSec partner.

ECN-TUNNEL-NEGOTIATION = NO
The use of ECN tunnels in accordance with RFC 3168 cannot be negotiated with the IPSec partner.

POLICY: Define security policy

The POLICY record defines a new security policy.

POLICY

POLICY NAME = <alphanum-name 1 .. 32>

,POLICY-RANGE = GLOBAL / **PARTNERSYSTEM (...)** / **PARTNER (...)**

PARTNERSYSTEM (IP-ADDRESS = ... / **IPV6-ADDRESS = ...)**

IP-ADDRESS = <ipv4addr> / <ipv4addr/prefix-len> / <ipv4addr-low> - <ipv4addr-high> / **ANY**

IPV6-ADDRESS = <ipv6addr> / <ipv6addr/prefix-len> / <ipv6addr-low> - <ipv6addr-high> / **ANY**

PARTNER (IP-ADDRESS = ... , PORTNUMBER = ... / **IPV6-ADDRESS = ... , PORTNUMBER = ...)**

IP-ADDRESS = <ipv4addr> / <ipv4addr/prefix-len> / <ipv4addr-low> - <ipv4addr-high> / **ANY**

IPV6-ADDRESS = <ipv6addr> / <ipv6addr/prefix-len> / <ipv6addr-low> - <ipv6addr-high> / ANY

,PORTNUMBER = ANY / <port#> / <port#-low> - <port#-high>

,FIRST-SECURITY-ASSOCIATION = <alphanum-name 1 ..32> / ***NONE**

,SECOND-SECURITY-ASSOCIATION = <alphanum-name 1 .. 32> / ***NONE**

[**,COMPRESSION = DEFLATE** / ***NONE**]

,MODE = BYPASS / DISCARD / TRANSPORT, ... / **TUNNEL (...), ...**

TUNNEL (...), ...

INNER-TUNNEL = (END-OF-TUNNEL (...) [, **START-OF-TUNNEL = (...)]**

,SECURITY-ASSOCIATION = ... [,COMPRESSION = ...])

END-OF-TUNNEL (IP-ADDRESS = <ipv4addr > / **IPV6-ADDRESS =** <ipv6addr>**)**

[**,START-OF-TUNNEL (IP-ADDRESS =** <ipv4addr > / **IPV6-ADDRESS =** <ipv6addr>**)**]

,SECURITY-ASSOCIATION = <alphanum-name 1 ..32> / ***NONE**

[**,COMPRESSION = DEFLATE** / ***NONE]**

OUTER-TUNNEL = *NONE / (END-OF-TUNNEL (...) [, **START-OF-TUNNEL = (...)]**

,SECURITY-ASSOCIATION = ... [,COMPRESSION = ...])

END-OF-TUNNEL (IP-ADDRESS = <ipv4addr > / **IPV6-ADDRESS =** <ipv6addr>**)**

[**,START-OF-TUNNEL (IP-ADDRESS =** <ipv4addr > / **IPV6-ADDRESS =** <ipv6addr>**)**]

,SECURITY-ASSOCIATION = <alphanum-name 1 ..32> / ***NONE**

[**,COMPRESSION = DEFLATE** / ***NONE**]

,OWN-ADDRESS = ANY / **IP-ADDRESS =** <ipv4addr> / **IPV6-ADDRESS =** <ipv6addr>

,OWN-PORTNUMBER = ANY / <port#> / <port#> - <port#>

,DIRECTION = IN / **OUT** / **ANY**

,PROTOCOL = TCP / **UDP / ICMP, ...** / **ANY, ...**

,ICMP-CODE = ANY / <code> / <code-low> - <code-high>

,ICMP-TYPE = ANY / <type> / <type-low> - <type-high>

Operand description

NAME = <alphanum-name 1 .. 32>
Name of the security policy. The name can consist of letters and digits. A distinction is drawn between uppercase and lowercase letters. The first character in the name must be a letter.

POLICY-RANGE = ...
Specifies the scope of the security policy.

POLICY-RANGE = GLOBAL
The security policy can be used for data transfer to all partner applications in all partner systems.

POLICY-RANGE = PARTNERSYSTEM (IP-ADDRESS = ... / IPV6-ADDRESS = ...)
The security policy can be used for data transfer to the specified partner systems.

IP-ADDRESS = <ipv4addr> /<ipv4addr/prefix-len> /
<ipv4addr-low> - <ipv4addr-high> / ANY
IPv4 address, IPv4 address with prefix specification or range of IPv4 addresses. This/these specifies/specify the partner system or systems to which the policy can be applied. The IPv4 address(es) must be specified in the usual “decimal dotted” notation. ANY stands for an arbitrary IPv4 address.

IPV6-ADDRESS = <ipv6addr>/ <ipv6addr/prefix-len> /
<ipv6addr-low> - <ipv6addr-high> / ANY
IPv6 address, IPv6 address with prefix specification or range of IPv6 addresses. This/these specifies/specify the partner system or systems to which the policy can be applied. The IPv6 address(es) must be specified in the usual hexadecimal notation with a colon (:). ANY stands for any IPv6 address.

POLICY-RANGE = PARTNER (IP-ADDRESS = ... , PORTNUMBER = ... /
IPV6-ADDRESS = ... , PORTNUMBER = ...)
The security policy can be used for data transfer to the specified partner applications in the specified partner systems.

IP-ADDRESS = <ipv4addr> / <ipv4addr/prefix-len> /
<ipv4addr-low> - <ipv4addr-high> / ANY
IPv4 address, IPv4 address with prefix specification or range of IPv4 addresses. This/these specifies/specify the partner system or systems to which the policy can be applied. The IPv4 address(es) must be specified in the usual “decimal dotted” notation. ANY stands for any IPv4 address.

IPV6-ADDRESS = <ipv6addr>/<ipv6addr/prefix-len> /
<ipv6addr-low> - <ipv6addr-high> / ANY
IPv6 address, IPv6 address with prefix specification or range of IPv6 addresses. This/these specifies/specify the partner system or systems to which the policy can be applied. The IPv6 address(es) must be specified in the usual hexadecimal notation with a colon (:). ANY stands for any IPv6 address.

PORTNUMBER = ANY / <port#> / <port#-low> - <port#-high>
Port number or range of port numbers. This/these specifies/specify the partner application(s) to which the policy can be applied. The valid port numbers are in the range from 0 to 65535. The keyword ANY specifies a value range from 0 to 65535.

FIRST-SECURITY-ASSOCIATION = <alphanum-name 1 .. 32> / *NONE
Name of the security association to be applied to the contents of the IP package. The corresponding SECURITY-ASSOCIATION record must already be defined.

SECOND-SECURITY-ASSOCIATION = <alphanum-name 1 .. 32> / *NONE
Name of the security association to be applied to the contents of the IP packet, which may already be protected by FIRST-SECURITY-ASSOCIATION.
The security policy may only be assigned a second security association if this security policy has already been assigned a first security association by means of FIRST-SECURITY-ASSOCIATION = ...
If two security associations are used, the first one must use the ESP security protocol and the second one the AH security protocol (see the section “Security concepts based on security associations” on page 154).
The keyword *NONE specifies that no second security association is used.

COMPRESSION = DEFLATE / *NONE
The optional operand COMPRESSION is used to request data compression. Compression takes place before an IPSec protocol is used. The parameter value DEFLATE specifies the compression algorithm.
The default value is *NONE.

MODE = BYPASS / DISCARD / TRANSPORT, ... / TUNNEL(...), ...
Specifies what is to be done with the packets of the specified communication relationship.

MODE = BYPASS
No IPSec functions are applied to packets affected by the policy. They are transferred unencrypted and without further checking. A security association does not have to be specified.

MODE = DISCARD
Packets affected by the policy are discarded. This makes it possible to prevent communication relationships. A security association does not have to be specified.

MODE = TRANSPORT, ...
The security policy secures the data transfer in transport mode.

MODE = TUNNEL (...), ...
The security policy secures the data transfer in tunnel mode.

INNER-TUNNEL = (END-OF-TUNNEL (...) [,START-OF-TUNNEL(...)], SECURITY-ASSOCIATION = ...)
Specifies the IPSec tunnel of the security policy. It is defined by the operands END-OF-TUNNEL, START-OF-TUNNEL (optional) and SECURITY-ASSOCIATION. If the START-OF-TUNNEL parameter is not specified, the tunnel's local endpoint is the address specified for OWN-ADDRESS.

END-OF-TUNNEL (IP-ADDRESS = <ipv4addr > / IPV6-ADDRESS = <ipv6addr>)
Specifies the remote address of the inner tunnel endpoint. The tunnel's local endpoint is specified by means of the START-OF-TUNNEL parameter (see below).

IP-ADDRESS = <ipv4addr>
Remote IPv4 address of the tunnel endpoint. The IPv4 address must be specified in the usual "decimal dotted" notation.

IPV6-ADDRESS = <ipv6addr>
Remote IPv6 address of the tunnel endpoint. The IPv6 address must be specified in the usual hexadecimal notation with a colon (:).

START-OF-TUNNEL (IP-ADDRESS = <ipv4addr > / IPV6-ADDRESS = <ipv6addr>)
Specifies the local address of the tunnel endpoint.

IP-ADDRESS = <ipv4addr>
Local IPv4 address of the tunnel endpoint. The IPv4 address must be specified in the usual "decimal dotted" notation.

IPV6-ADDRESS = <ipv6addr>
Local IPv6 address of the tunnel endpoint. The IPv6 address must be specified in the usual hexadecimal notation with a colon (:).

SECURITY-ASSOCIATION = <alphanum-name 1 .. 32> / *NONE
Name of the security association used in the security tunnel. The SECURITY-ASSOCIATION record must already be defined.

COMPRESSION = DEFLATE / *NONE
The optional operand COMPRESSION is used to request data compression. Compression takes place before an IPSec protocol is used. The parameter value DEFLATE specifies the compression algorithm.
The default value is *NONE.

OUTER-TUNNEL = *NONE / (END-OF-TUNNEL (...) [,START-OF-TUNNEL(...)] ,SECURITY-ASSOCIATION = ...)
Specifies the IPSec tunnel of the security policy. It is defined by the operands END-OF-TUNNEL, START-OF-TUNNEL (optional) and SECURITY-ASSOCIATION. If the START-OF-TUNNEL parameter is not specified, the tunnel's local endpoint is the address specified for OWN-ADDRESS. The keyword *NONE specifies that no OUTER-TUNNEL is used.

END-OF-TUNNEL (IP-ADDRESS = <ipv4addr > / IPV6-ADDRESS = <ipv6addr>)
Specifies the remote address of the inner tunnel endpoint. The tunnel's local endpoint is specified by means of the START-OF-TUNNEL parameter (see below).

IP-ADDRESS = <ipv4addr>
Remote IPv4 address of the tunnel endpoint. The IPv4 address must be specified in the usual "decimal dotted" notation.

IPV6-ADDRESS = <ipv6addr>
Remote IPv6 address of the tunnel endpoint. The IPv6 address must be specified in the usual hexadecimal notation with a colon (:).

START-OF-TUNNEL (IP-ADDRESS = <ipv4addr > / IPV6-ADDRESS = <ipv6addr>)
Specifies the local address of the tunnel endpoint.

IP-ADDRESS = <ipv4addr>
Local IPv4 address of the tunnel endpoint. The IPv4 address must be specified in the usual "decimal dotted" notation.

IPV6-ADDRESS = <ipv6addr>
Local IPv6 address of the tunnel endpoint. The IPv6 address must be specified in the usual hexadecimal notation with a colon (:).

SECURITY-ASSOCIATION = <alphanum-name 1 .. 32> / *NONE
Name of the security association used in the security tunnel. The SECURITY-ASSOCIATION record must already be defined.

COMPRESSION = DEFLATE / *NONE
The optional operand COMPRESSION is used to request data compression. Compression takes place before an IPSec protocol is used. The parameter value DEFLATE specifies the compression algorithm.
The default value is *NONE.

OWN-ADDRESS = ANY/ IP-ADDRESS = <ipv4addr> / IPV6-ADDRESS = <ipv6addr>
Specifies which of the local addresses the security policy can be applied to.

OWN-ADDRESS = ANY
Specifies that the security policy can be applied to all local addresses.

OWN-ADDRESS = IP-ADDRESS = <ipv4addr>
Specifies a local IPv4 address to which the security policy can be applied. The IPv4 address must be specified in the usual “decimal dotted” notation.

OWN-ADDRESS = IPV6-ADDRESS = <ipv6addr>
Specifies a local IPv6 address to which the security policy can be applied. The IPv6 address must be specified in the usual hexadecimal notation with a colon (:).

OWN-PORTNUMBER = ANY / <port#> / <port#-low - port#-high>
Specifies a port number or a range of port numbers. The security policy can be applied to your associated application(s). The valid value range for a port number is from 0 to 65535.

OWN-PORTNUMBER = ANY
Specifies the port number range from 0 to 65535.

OWN-PORTNUMBER = <port#>
Specifies a port number.

OWN-PORTNUMBER = <port#-low> - <port#-high>
Specifies a port number range.

DIRECTION = IN / OUT / ANY
Specifies the direction of the data transfer secured by the security policy.

DIRECTION = IN
The policy secures the inbound data transfer.

DIRECTION = OUT
The policy secures the outbound data transfer.

DIRECTION = ANY
The policy secures both the inbound and outbound data transfer.

PROTOCOL = TCP / UDP / ICMP, ... / ANY, ...
Specifies the layer-4 protocol secured by the security policy.

PROTOCOL = TCP
The policy secures the TCP data transfer.

PROTOCOL = UDP
The policy secures the UDP data transfer.

PROTOCOL = ICMP, ...
The policy secures the ICMP data transfer.

PROTOCOL = ANY, ...
The policy secures the TCP, UDP and ICMP data transfer. If PROTOCOL=ANY is specified, the following parameters are set:
PARTNER-PORTNUMBER=ANY

OWN-PORTNUMBER=ANY
ICMP-CODE=ANY
ICMP-TYPE=ANY

ICMP-CODE =
ICMP code or range of ICMP codes to which the policy is applicable.
The valid value range for an ICMP code is from 0 to 255.
This parameter should only be specified if PROTOCOL = ICMP or ANY.

ICMP-CODE = ANY
Any ICMP type is possible.

ICMP-CODE = <code>
Specifies a specific ICMP code.

ICMP-CODE = <code-low> - <code-high>
Specifies a range of ICMP codes.

ICMP-TYPE =
ICMP type or range of ICMP types to which the policy is applicable.
The valid value range for ICMP types is from 0 to 255.
This parameter should only be specified if PROTOCOL = ICMP or ANY.

ICMP-TYPE = ANY
Any ICMP type is possible.

ICMP-TYPE = <type>
Specifies a specific ICMP type.

ICMP-TYPE = <type-low> - <type-high>
Specifies a range of ICMP types.

PARTNER-SAS: Delete SAs created automatically

In DELETE mode, the PARTNER-SAS record specifies which dynamically created SAs (MODE = IKE) are to be deleted. In ADD mode, the record has no effect.

PARTNER-SAS:
IP-ADDRESS = < Ipv4-addr > / IPV6-ADDRESS = < Ipv6-addr >

Operand description

IP-ADDRESS = <Ipv4-addr>
IPv4 address that specifies the partner system whose SAs are to be deleted

IPV6-ADDRESS = <Ipv6-addr>
IPv6 address that specifies the partner system whose SAs are to be deleted

6.3.3 Checking the IPSec configuration

After an IPSec configuration file is created or modified, it is advisable to check the syntax of a changed file and/or the expected IPSec configuration. IPSec offers a program for this.

START-IPSEC-DB-CHECK

The call is as follows:

► `/START-IPSEC-DB-CHECK`

The following initial message then appears:

```
IPSEC-DB-CHECK Version n.n loaded
```

You are then requested to enter the name of the configuration file to be checked.

```
IPSEC-DB-CHECK Enter filename of configurationfile or *none or *end
```

– When you enter a file name, the syntax check is carried out for this configuration file. If the specified configuration file does not exist, the following error message appears:

  `IPSEC-DB-CHECK could not open configurationfile` *`filename`*

  You are then again requested to specify the name of the configuration file to be checked.

– If you enter `*none`, the configuration records to be checked are read from *SYSDTA.

  ► Conclude the *SYSDTA entry with `END`.

– If you enter `*end`, execution of START-IPSEC-DB-CHECK is terminated without any further action.

The result of the syntax check is written to a logging file.

The next prompt requests that you specify the logging file:

```
IPSEC-DB-CHECK Enter filename of loggingfile or *none or *end
```

– When you enter a file name, the logging records are written to a file with this name:

  – If the file does not yet exist, it is created.

  – If the file already exists, it is overwritten.

– If you enter `*none`, the logging records are written to *SYSOUT.

– If you enter `*end`, the execution of START-IPSEC-DB-CHECK is terminated without any further action.

The following message indicates that checking of the configuration file has begun:

```
IPSEC-DB-CHECK starting
```

Each INCLUDE statement is logged as follows:

```
IPSEC-DB-CHECK including file filename
IPSEC-DB-CHECK including file filename terminated
```

If the configuration file is read without an error, the following messages appear:

IPSEC configurationfile *filename* read without error and without warning.

If warnings are output, the following messages appear:

IPSEC configurationfile *filename* read with n warning(s).
IPSEC-DB-CHECK terminated

If errors are detected when the configuration file is read, the following messages appear:

IPSEC configuration filename read with *x* error(s) and n warning(s)..
See logging file *logging-filename* for more information.
IPSEC-DB-CHECK terminated

Structure of the logging file

The following two information records appear at the beginning of the logging file:

```
IPSEC-DB-CHECK: Logging for test of IPSEC configurationfile file filename
IPSEC-DB-CHECK: Test performed at date time
```

These are followed by the individual lines of the configuration file read. Each line read is preceded by the following message:

```
Read line line number
```

After a configuration record has been successfully read in, the following message appears:

```
New X Record created
```

(*X* stands for key, signature or security association.)

When a POLICY record is read in successfully, the following message appears:

```
New policy created
```

The following message is output for each SA derived from the policy:

```
New i1 i2 SA created
```

(i1 stands for inbound or outbound. i2 stands for first, second, inner tunnel or outer tunnel.)

If an SA derivation results in a duplicate, the following warning is output:

```
Warning: i1 i2 SA according to POLICY record already in database (i3)
```

(i1 and i2 are as above; i3 is the name of the duplicate.)

If an error is detected when the configuration file is checked, the following message is output in the next line:

```
ERROR: explanatory text
```

If the error can be assigned to a particular column (in the current line of the configuration file), this column is marked with the character “^”.

The reported errors must be corrected.

The information as to which SPD and SAD would have been generated from the checked configuration file is then output to the logging file.

6.3.4 IKE configuration

In BS2000/OSD the IKE daemon Racoon2 together with security policy daemon SPMD provides the functions of automated SA and key management. Both programs can be controlled via statements in a configuration file.

When the IPSec subsystem is installed, a default file with the name $TSOS.SYSDAT.IPSEC.*nnn*.RAC2 is created with the relevant statements. There you will find not only default values for protocol parameters, but also specifications defining the port numbers which Racoon2 and SPMD are to work with and the file which contains, for example, the SPMD password. This file, like the PSK files, must be created by the system administrator. The IPSec subsystem and the SPMD and Racoon2 programs can be started with the default file.

This file is referred to in this manual under the name $TSOS.SYSDAT.IPSEC.*nnn*.RAC2.MANUAL-BSP-DEFAULT. The INCLUDE statement can be used to link it into the configuration files which are to be created. It is therefore advisable to back up the installed default file.

Default file SYSDAT.IPSEC.*nnn*.RAC2

```
# The SPMD/Racoon2 default configuration file.
setval
{
   PSKDIR   " ";
};
# interface info
# Which address should the daemons use.
interface
{
# Racoon2 listens with UDP port 500 at every system's IP address.
#
   ike
   {
      MY_IP;
   };
# A specific IP address and/or an alternate port number must be definied
# here.
# Several instructions are possible concurrently.
# The support of NAT-Traversal requires an additional instruction of the
# form  MY_IP port 4500;
#
# SPMD listens  with TCP Portnumber 3333 at the loopback address.
# This is BS2000/OSD special and is needed for communication between
# Racoon2 and SPMD.
    spmd
```

```
   {
      127.0.0.1 port 3333;
   };

# SPMD's logon password is located in the file $(PSKDIR)SPMD.PWD
#
   spmd_password "spmd.pwd";
};
#
# default section
#
default
{
#
# remote describes the KMP peer.
   remote
  {
      ikev1
     {
         proposal_check obey;
         logmode normal;
         kmp_sa_lifetime_time 3600 sec;
         kmp_sa_lifetime_byte 0;
         interval_to_send 10 sec;
         times_per_send 1;
         ipsec_sa_nego_time_limit 40 sec;
         kmp_enc_alg
       {
         3des_cbc;
       };
         kmp_hash_alg
       {
         md5;
       };
         kmp_dh_group
       {
        modp1536;
        modp1024;
        modp768;
       };
         kmp_auth_method
       {
        psk;
       };
         random_pad_content on;
      };
      ikev2
   {
```

```
            logmode normal;
            kmp_sa_lifetime_time 0;
            kmp_sa_lifetime_byte 0;
            max_retry_to_send 3;
            interval_to_send 10 sec;
            times_per_send 1;
            ipsec_sa_nego_time_limit 40 sec;
            kmp_enc_alg
          {
           3des_cbc;
          };
            kmp_prf_alg
          {
           hmac_md5;
          };
            kmp_hash_alg
          {
           hmac_md5;
          };
            kmp_dh_group
          {
           modp1536;
           modp2048;
           modp3072;
          };
            kmp_auth_method
          {
           psk;
          };
            random_pad_content on;
            random_padlen on;
            max_padlen 50 bytes;
         };
    };
# kmp_prf_alg is not needed for IKEv1.
# The notation of the kmp_hash_alg algorithms are distinct in IKEv1 and
# IKEv2.
# The _lifetime value 0 means infinite.
# Authentication method must be preshared secret key. The key must be
# included  in a file. The file name should be specified in a remote
# directive.
#
# The following specifies a minimal default policy directive.
#
    policy
   {
        ipsec_mode transport;
        ipsec_level require;
```

```
    };
 # The following specifies a minimal default ipsec directive.
#
    ipsec
   {
        ipsec_sa_lifetime_time 0;
        ipsec_sa_lifetime_byte 0;
    };

# The following specifies a minimal default sa directive.
#
    sa
    {
        esp_enc_alg
    {
          aes128_cbc;
          3des_cbc;
     };

        ah_auth_alg
    {
          hmac_sha1;
          hmac_md5;
    };
    };
};
```

More detailed documentation is available on the Racoon2 homepage:

http://www.racoon2.wide.ad.jp/w/

6.4 Working with the IPSec subsystem

The files required in BS2000/OSD for IPSec and IKE operation were described in detail in the previous sections. Before starting the IPSec subsystem, you should check whether the files are installed and what they contain. For effective IPSec operation you have to create the following:

- IPSec configuration files
- Preshared Secret Key files (PSK) for the various partners
- SPMD.PWD (file containing the LOGON password for SPMD)

SYSSSI file

When the IPSec subsystem starts up, the statements in the SYSSSI.IPSEC.*nnn* file are executed. When the file is shipped, it contains the following command:

```
LOAD-IPSEC-DB FROM-FILE=$TSOS.SYSDAT.IPSEC.nnn.CONF.
```

IPSec is instructed to create the IPSec database beginning with the default configuration file. The name of the file can be changed at this point in order to start the generation of the IPSec database with a different file.

If it is discovered when the subsystem is started that there is no IPSec configuration file with the name specified in the SSINFO file, the startup of the subsystem is aborted with an error.

If the default port number 3500 for the communication between IPSec and the IKE daemon is already being used by another system program, you can make IPSec use an alternative UDP port number by entering the following statement in the SYSSSI file:

```
SET-PFKEY-PARAMETER PORT=<port number>
```

Starting and stopping the IPSec subsystem

You start the IPSec subsystem by entering the following BS2000/OSD command:

```
/START-SUBSYSTEM IPSEC
```

This starts the program SPMD at the same time. SPMD starts the program Racoon2.

You stop (suspend) the IPSec subsystem by entering the following BS2000/OSD command:

```
/STOP-SUBSYSTEM IPSEC
```

The programs SPMD and Racoon2 are terminated at the same time.

6.5 IPSec subsystem administration

6.5.1 Changing the IPSec configuration

The IPSec configuration can be changed during live operation. Normally, new security policies have to be defined or existing ones modified or deleted. In exceptional cases, such as when an IPSec partner system is temporarily inaccessible or has been restarted, it may be necessary to delete security associations. The only option you have here is to create a configuration file that describes the change to the database and then to load this file.

Example

The statements ADD and DELETE are used in the configuration file.

The MODIFY.DEFAULTPOLICY file changes the default policy from BYPASS to DISCARD.

```
*
DELETE
*
POLICY NAME=THEOTHERS
*
ADD
*
*
* definition of POLICY records
*
POLICY NAME=THEOTHERS -
,POLICY-RANGE=GLOBAL -
,OWN-ADDRESS=ANY -
,OWN-PORTNUMBER=ANY -
,DIRECTION=ANY -
,PROTOCOL=ANY -
,ICMP-TYPE=ANY -
,ICMP-CODE=ANY -
,MODE = DISCARD
```

The following command then modifies the IPSec configuration:

```
/LOAD-IPSEC-DB FROM-FILE=MODIFY.DEFAULTPOLICY
```

LOAD-IPSEC-DB: Load IPSec database

The LOAD-IPSEC-DB command loads an IPSec configuration file and thus changes the IPSec database.

The alias LDIPSDB is the SDF short name for LOAD-IPSEC-DB.

BS2000 console	BS2000 SDF command	Command/SOF file	SNMP management	Parameter service
	x			

List of permissible command sources

LOAD-IPSEC-DB/LDIPSDB Load IPSec database
FROM-FILE = ***UNCHANGED** / ***STD** / **<full-filename 1 .. 54>**

Operand description

FROM-FILE =
Specifies the name of the IPSec configuration file.

FROM-FILE = *UNCHANGED
The currently set file name is used (default).

FROM-FILE = *STD
An IPSec configuration file with the default file name $TSOS.SYSDAT.IPSEC.*nnn*.CONF is used (*nnn* specifies the IPSec version, for example 014 for IPSec V1.4).

FROM-FILE = <full-filename 1 .. 54>
An IPSec configuration file with a name other than the default file name is used.

Command logging

- Correct execution of the LOAD-IPSEC-DB command is acknowledged with message YIS0211. Message YIS0219 may also be output.
- Failed execution of the LOAD-IPSEC-DB command is acknowledged with message YIS0210. Message YIS0218 may also be output.

Command return codes

The following table shows the return codes that are possible for commands.

(SC2)	SC1	Maincode	Bedeutung
0	0	CMD0001	Command successfully processed
0	1	CMD0202	Operand error
0	32	CMD0221	System error
0	64	CMD0216	Privilege not sufficient

Examples

The IPSec subsystem is already generated, and the IPSec database is to be loaded with the data of the IPSec configuration file SYSDAT.IPSEC.TEST.CONF:

```
/LOAD-IPSEC-DB FROM-FILE=SYSDAT.IPSEC.TEST.CONF
```

The IPSec subsystem is already generated, and the IPSec database is to be loaded with the data of an IPSec configuration file with the default file name:

```
/LOAD-IPSEC-DB FROM-FILE=*STD
```

Changing the automated configuration

The security policies of an automated configuration are changed in a similar way, except that the statements for deleting and adding a security policy cannot be written into a configuration file. These statements must be specified as input parameters by utility routines.

```
/START-PROG *M(SYSLNK.IPSEC.nnn.RAC2,CTRLPROG,A,A)

%  BLS0523 ELEMENT 'CTRLPROG', VERSION '300', TYPE 'L' FROM LIBRARY
':BO7A:$TSOS.SYSLNK.IPSEC.nnn.RAC2' IN PROCESS
%  BLS0524 LLM 'CTRLPROG', VERSION '300' OF '2009-02-09 13:07:29' LOADED
%  CCM0001 ENTER OPTIONS:
*policy delete DIAL-in

/START-PROG *M(SYSLNK.IPSEC.nnn.RAC2,CTRLPROG,A,A)
*policy delete DIAL-out
```

This control program enables you to delete the policy which was defined originally and permits access to the DIALOG application from the PC with the IP address 172.25.83.42 (see page 73).

If dialog access is now to be permitted from several PCs, you can, for example, create a configuration file MODIFY.DIALOG with the following content:

```
# PC-1
# Describes the KMP capabilities of PC-1
# Parameters of the default section are overridden.
#
remote PC-1 {
   acceptable_kmp { ikev1; };
   ikev1 {
      peers_ipaddr 172.25.83.0/24 port 500;
      kmp_enc_alg { 3des_cbc;};
      kmp_hash_alg { sha1; md5;};
      kmp_dh_group { modp1024; };
      kmp_auth_method { psk; };
      pre_shared_key "PC-1.psk";
   };
};
#
# $DIALOG
# Selector and policy definition for the communication to $DIALOG from
# a range of partners.
#
selector DIAL-in {
   direction inbound;
   src 172.25.83.0/24;
   dst 172.25.92.72 port 1110;
```

```
   upper_layer_protocol "tcp";
   policy_index PC-1;
};
selector DIAL-out {
   direction outbound;
   dst 172.25.83.0/24;
   src 172.25.92.72 port 1110;
   upper_layer_protocol "tcp";
   policy_index PC-1;
};
```

The configuration file which extends dialog access is made known to the security policy daemon SPMD and the IKE daemon Racoon2 using the following command:

```
/MODIFY-SERVICE-PARAMETER SERVICE-NAME=$YISSPMD,START-PARAMETER= -
/  '-A -f  MODIFY.DIALOG'

/MODIFY-SERVICE-PARAMETER SERVICE-NAME=$YISRAC2,START-PARAMETER= -
/  '-A -f  MODIFY.DIALOG'
```

The -A option causes the content of the file which was specified in the -f option to be inserted in the current configuration.

As an alternative, you can adjust the configuration file SYSDAT.IPSEC.nnn.RAC2 accordingly and reload it using the following commands:

```
/MODIFY-SERVICE-PARAMETER SERVICE-NAME=$YISSPMD,START-PARAMETER= -
/  '-R -f SYSDAT.IPSEC.nnn.RAC2'

/MODIFY-SERVICE-PARAMETER SERVICE-NAME=$YISRAC2,START-PARAMETER= -
/  '-R -f SYSDAT.IPSEC.nnn.RAC2'
```

SPMD then deletes the current security policies from the IPSec kernel and sets up the configuration again in accordance with the new description.

6.5.2 IPSec monitoring

IPSec monitoring provides information on IPSec operation by writing logging information in the current CONSLOG file.

Logging covers the following IPSec messages:

- YIS0310 (data elements)
- YIS0311 (openCRYPT)
- YIS0312 (error in IPSec)

START-IPSEC-MONITORING / SRIPSMN: Start IPSec monitoring

The alias SRIPSM is the SDF short name for START-IPSEC-MONITORING.

BS2000 console	BS2000 SDF command	Command/SOF file	SNMP management	Parameter service
	x			

List of permissible command sources

START-IPSEC-MONITORING / SRIPSMN Start IPSEC monitoring
TIMER = ***STD** / <integer 5..32765>

TIMER = ...
Specifies the timer interval for updating the monitoring output.

TIMER = *STD
The default value (5 seconds) is set as the timer interval.

TIMER = <integer 5 .. 32765>
The specified integer value (unit: seconds) is set as the timer interval.

Command logging

- Correct execution of the START-IPSEC-MONITORING command is acknowledged with message `YIS0223`.
- Failed execution of the command is acknowledged with message `YIS0222`.

Command return codes

The following table shows the return codes that are possible for commands.

(SC2)	SC1	Maincode	Bedeutung
0	0	CMD0001	Command successfully processed
0	1	CMD0202	Operand error
0	32	CMD0221	System error
0	64	CMD0216	Privilege not sufficient

Example

IPSec monitoring is to be started. The display is to be updated at intervals of 10 minutes (600 seconds):

```
/START-IPSEC-MONITORING TIMER=600
```

STOP-IPSEC-MONITORING / SPIPSMN: Stop IPSec monitoring

The alias name SPIPSMN is the SDF short name for STOP-IPSEC-MONITORING.

BS2000 console	BS2000 SDF command	Command/SOF file	SNMP management	Parameter service
	x			

List of permissible command sources

STOP-IPSEC-MONITORING / SPIPSMN Stop IPSEC monitoring

Command logging

- Correct execution of the STOP-IPSEC-MONITORING command is acknowledged with message `YIS0225`.
- Failed execution of the command is acknowledged with message `YIS0224`.

Command return codes

The following table shows the possible return codes for commands:

(SC2)	SC1	Maincode	Bedeutung
0	0	CMD0001	Command successfully processed
0	1	CMD0202	Operand error
0	32	CMD0221	System error
0	64	CMD0216	Privilege not sufficient

Example

IPSec monitoring is to be terminated:

```
/STOP-IPSEC-MONITORING
```

6.5.3 Creating diagnostic documents

If an error situation that you cannot rectify yourself occurs during IPSec operation, please get in touch with your contact person.

To permit efficient troubleshooting, your contact person requires the following information:

- A precise description of the error situation and details of whether the error can be reproduced
- A description of the hardware configuration
- Information on the software configuration with details of the type and scope of the operating system and IPSec software used

  You also have to document the relevant version numbers and any Rep corrections used.
- A complete console log or the $SYSAUDIT.SYS.CONSLOG file for the BS2000/OSD session
- Dumps
- $TSOS.SYS.SERSLOG file
- Any IPSec monitoring documents that are available
- IPSec traces (see below)
- IPSECDIA documents (see below)
- Racoon2-Traces
- SPMD-Traces
- BCAM diagnostic documents. You generate these using the ASTRID utility routine (see the BCAM V21.0A manual).

Creating IPSec traces

You create IPSec traces as follows:

► Start the IPSec trace with the /DCDIAG command.

  – The name for the IPSec user interface trace is IPSEC.COM.

  – You will find more information on the /DCDIAG command in the BCAM V21.0A manual.

Creating Racoon2 traces

You create Racoon2 traces as follows:

- Enter the following command:

```
/MODIFY-SERVICE-PARAMETER SERVICE-NAME=
        $YISRAC2, START-PARAMETER= '-D 7 -l <filename>'
```

  Enter any name you choose for <filename>.

- Once the diagnostic information is written, enter the command again with the following parameters:

```
/MODIFY-SERVICE-PARAMETER SERVICE-NAME=$YISRAC2, START-PARAMETER= '-D 0'
```

Creating SPMD traces

You create SPMD traces as follows:

- Enter the following command:

```
/MODIFY-SERVICE-PARAMETER SERVICE-NAME=$YISSPMD,
        START-PARAMETER= '-D 10 -l <filename>'
```

  Enter any name you choose for <filename>.

- Once the diagnostic information is written, enter the command again with the following parameters:

```
/MODIFY-SERVICE-PARAMETER SERVICE-NAME=$YISSPMD, START-PARAMETER= '-D 0'
```

Creating IPSECDIA documents

You create IPSECDIA information as follows:

- Assign SYSLST to a file.
- Start IPSec diagnostic program: `IPSECDIA`
- Enter the `TOTAL` statement.
- Enter the `END` statement.
- Cancel the redirection of the SYSLST output.

6.6 Configuration examples

The use of the IPSec security mechanisms is illustrated below using sample configurations.

6.6.1 Configuration with manually and automatically defined keys

In this example the following security policies are to be implemented:

1. Communication relationships without IPSec security mechanisms are not permitted.
2. All data traffic within the subnet 172.25.92.0/24 is to be encrypted.
3. Data traffic between the IP addresses 172.25.92.72 and 172.25.92.74 is to be secured with a key.
4. FTP data traffic between the systems with the IP addresses 172.25.92.72 and 172.25.92.74 is to be authenticated and encrypted. To do this, it is necessary to protect FTP server port 21 on the partner system. On the local system, by contrast, FTP data port 20 must be protected. Configuration is thus carried out from the perspective of the FTP client.
5. Transport mode is to be used.
6. The encryption method is AES.
7. The signature method is HMAC-SHA1.
8. The keys are assigned manually. They are to be managed in a separate file.

6.6.1.1 Manually defined keys in an IPSec configuration

The configuration files IPSEC.KEYS and IPSEC.CONF then look like this:

IPSEC.KEYS file

```
KEY NAME=KEYAES1 -
,KEY-ALGORITHM=AES-CBC -
,KEY-VALUE=X'344B69566B3742535466433756457832'
_____________________________________________________________________(7)
*
KEY NAME=KEYAES11 -
,KEY-ALGORITHM=AES-CBC -
,KEY-VALUE=X'5077776952585966476e386934553766'
_____________________________________________________________________(6)
*
KEY NAME=KEYAES2 -
,KEY-ALGORITHM=AES-CBC -
,KEY-VALUE=X'73786472580a72476252537158384e4d'
_____________________________________________________________________(5)
*
KEY NAME=KEYAES3 -
,KEY-ALGORITHM=AES-CBC -
,KEY-VALUE=X'2f6b2f364964567163624b44386e4238'
_____________________________________________________________________(4)
*
SIGNATURE NAME=KEYSHA1 -
,SIGNATURE-ALGORITHM=HMAC-SHA-1 -
,SIGNATURE-VALUE=X'396767586A4435464830473848543647 4B64652F'
_____________________________________________________________________(6,7)
```

IPSEC.CONF file

```
FLUSH
_____________________________________________________________________(1)
*
INCLUDE IPSEC.KEYS
_____________________________________________________________________(2)
*
* definition of SA records
*
SECURITY-ASSOCIATION NAME=SA1 -
,TYPE=ENCRYPTION( -
 SIGNATURE=KEYSHA1 -
,KEY=KEYAES1) -
```

```
,MODE=MANUAL(INDEX=257) -
,ECN-TUNNEL= ALLOWED -
,ECN-TUNNEL-NEGOTIATION= YES
_____________________________________________________________________(7)
*
SECURITY-ASSOCIATION NAME=SA11 -
,TYPE=ENCRYPTION( -
 SIGNATURE=KEYSHA1 -
,KEY=KEYAES11) -
,MODE=MANUAL(INDEX=258) -
,ECN-TUNNEL= ALLOWED -
,ECN-TUNNEL-NEGOTIATION= YES
_____________________________________________________________________(6)
*
SECURITY-ASSOCIATION NAME=SA2 -
,TYPE=ENCRYPTION( -
 SIGNATURE=*NONE -
,KEY=KEYAES2) -
,MODE=MANUAL(INDEX=300) -
,ECN-TUNNEL= ALLOWED -
,ECN-TUNNEL-NEGOTIATION= YES
_____________________________________________________________________(5)
*
SECURITY-ASSOCIATION NAME=SA3 -
,TYPE=ENCRYPTION( -
 SIGNATURE=*NONE -
,KEY=KEYAES3) -
,MODE=MANUAL(INDEX=400) -
,ECN-TUNNEL= ALLOWED -
,ECN-TUNNEL-NEGOTIATION= YES
_____________________________________________________________________(4)
*
* definition of POLICY records
*
POLICY NAME=THEOTHERS -
,POLICY-RANGE=GLOBAL -
,OWN-ADDRESS=ANY -
,OWN-PORTNUMBER=ANY -
,DIRECTION=ANY -
,PROTOCOL=ANY -
,ICMP-TYPE=ANY -
,ICMP-CODE=ANY -
,MODE = DISCARD
_____________________________________________________________________(3)
*
POLICY NAME=MYFRIENDS -
,POLICY-RANGE=PARTNERSYSTEM (IP-ADDRESS=172.25.92.0-172.25.92.255) -
,OWN-ADDRESS=ANY -
```

```
,OWN-PORTNUMBER=ANY -
,DIRECTION=ANY -
,PROTOCOL=ANY -
,ICMP-TYPE=ANY -
,ICMP-CODE=ANY -
,MODE = TRANSPORT -
,FIRST-SECURITY-ASSOCIATION=SA3 -
,SECOND-SECURITY-ASSOCIATION=*NONE
_____________________________________________________________________(4)
*
POLICY NAME=VM4 -
,POLICY-RANGE=PARTNERSYSTEM (IP-ADDRESS=172.25.92.74) -
,OWN-ADDRESS=IP-ADDRESS = 172.25.92.72 -
,OWN-PORTNUMBER=ANY -
,DIRECTION=ANY -
,PROTOCOL=ANY -
,ICMP-TYPE=ANY -
,ICMP-CODE=ANY -
,MODE = TRANSPORT -
,FIRST-SECURITY-ASSOCIATION=SA2 -
,SECOND-SECURITY-ASSOCIATION=*NONE
_____________________________________________________________________(5)
*
POLICY NAME=FTPDATA -
,POLICY-RANGE=PARTNER (IP-ADDRESS=172.25.92.74,PORTNUMBER = 20) -
,OWN-ADDRESS=IP-ADDRESS = 172.25.92.72 -
,OWN-PORTNUMBER=ANY -
,DIRECTION=ANY -
,PROTOCOL=TCP -
,MODE = TRANSPORT -
,FIRST-SECURITY-ASSOCIATION=SA11 -
,SECOND-SECURITY-ASSOCIATION=*NONE
_____________________________________________________________________(6)
*
POLICY NAME=FTPSERVER -
,POLICY-RANGE=PARTNER (IP-ADDRESS=172.25.92.74,PORTNUMBER = 21) -
,OWN-ADDRESS=IP-ADDRESS = 172.25.92.72 -
,OWN-PORTNUMBER=ANY -
,DIRECTION=ANY -
,PROTOCOL=TCP -
,MODE = TRANSPORT -
,FIRST-SECURITY-ASSOCIATION=SA1 -
,SECOND-SECURITY-ASSOCIATION=*NONE
_____________________________________________________________________(7)
```

Explanations

(1) Any existing entries in the SPD and SAD are deleted.

(2) The file containing the keys is loaded. This implements point 8.

(3) All data packets from all local addresses to partner addresses are discarded (firewall outbound, point 1).

(4) The data traffic in the local subnet is secured in transport mode. The AES encryption method is used (points 2, 5 and 6).

(5) The data traffic from the local IP address 172.25.92.72 to the computer with the IP address 172.25.92.74 is secured in transport mode. The AES encryption method is used (points 3, 5 and 6).

(6) The data traffic from any port of the local IP address 172.25.92.72 to the FTP data port on the system with the IP address 172.25.92.74 is secured in transport mode. The AES encryption method and the HMAC-SHA1 authentication method are used (points 4, 5, 6 and 7).

(7) The data traffic from any port of the local IP address 172.25.92.72 to the FTP server port on the system with the IP address 172.25.92.74 is secured in transport mode. The AES encryption method and the HMAC-SHA1 authentication method are used (points 4, 5, 6 and 7).

A check with START-IPSEC-DB-CHECK outputs the SPD and SAD in the logging file:

```
      S E C U R I T Y   P O L I C Y   D A T A B A S E


outbound policies in ascending order


FTPDATA
  nr of ranges:    1
  ranges set:      own-port
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        TCP
  partner port:    20
  own port:        0 - 65535
  mode of use:     transport
FTPSERVER
  nr of ranges:    1
  ranges set:      own-port
```

```
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        TCP
  partner port:    21
  own port:        0 - 65535
  mode of use:     transport

VM4
  nr of ranges:    3
  ranges set:      prot part-port own-port type code
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  mode of use:     transport

MYFRIENDS
  nr of ranges:    5
  ranges set:      part-addr own-addr prot part-port own-port type code
  partner address: IPv4   172.25.92.0 - 172.25.92.255
  own address:     ANY
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  mode of use:     transport

THEOTHERS
  nr of ranges:    5
  ranges set:      part-addr own-addr prot part-port own-port type code
  partner address: ANY
  own address:     ANY
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535


 inbound policies in ascending order


 FTPDATA
  nr of ranges:    1
  ranges set:      own-port
```

```
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        TCP
  partner port:    20
  own port:        0 - 65535
  mode of use:     transport

FTPSERVER
  nr of ranges:    1
  ranges set:      own-port
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        TCP
  partner port:    21
  own port:        0 - 65535
  mode of use:     transport

VM4
  nr of ranges:    3
  ranges set:      prot part-port own-port type code
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  mode of use:     transport

MYFRIENDS
  nr of ranges:    5
  ranges set:      part-addr own-addr prot part-port own-port type code
  partner address: IPv4   172.25.92.0 - 172.25.92.255
  own address:     ANY
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  mode of use:     transport

THEOTHERS
  nr of ranges:    5
  ranges set:      part-addr own-addr prot part-port own-port type code
  partner address: ANY
  own address:     ANY
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
```

```
  partner port:    0 - 65535
  own port:        0 - 65535
  mode of use:     transport


outbound ESP security associations in ascending order

SA11
  nr of ranges:    1
  ranges set:      own-port
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        TCP
  partner port:    20
  own port:        0 - 65535
  SPI:             258 00000102
  mode of use:     transport

SA1
  nr of ranges:    1
  ranges set:      own-port
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        TCP
  partner port:    21
  own port:        0 - 65535
  SPI:             257 00000101
  mode of use:     transport

SA2
  nr of ranges:    2
  ranges set:      part-port own-port type code
  partner address: IPv4   172.25.92.74
  own address:     IPv4   172.25.92.72
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  SPI:             300 0000012C
  mode of use:     transport

SA3
  nr of ranges:    4
  ranges set:      part-addr own-addr part-port own-port type code
  partner address: IPv4   172.25.92.0 - 172.25.92.255
  own address:     ANY
  protocol:        ANY
    ICMP type:       0 - 255
```

```
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  SPI:             400 00000190
  mode of use:     transport


inbound ESP security associations in ascending order

SA1
  nr of ranges:    0
  own address:     IPv4   172.25.92.72
  SPI:             257 00000101
  partner address: IPv4   172.25.92.74
  protocol:        TCP
  partner port:    21
  own port:        0 - 65535
  mode of use:     transport

SA11
  nr of ranges:    0
  own address:     IPv4   172.25.92.72
  SPI:             258 00000102
  partner address: IPv4   172.25.92.74
  protocol:        TCP
  partner port:    20
  own port:        0 - 65535
  mode of use:     transport

SA2
  nr of ranges:    0
  own address:     IPv4   172.25.92.72
  SPI:             300 0000012C
  partner address: IPv4   172.25.92.74
  protocol:        ANY
    ICMP type:       0 - 255
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  mode of use:     transport

SA3
  nr of ranges:    1
  ranges set:      own-addr
  own address:     ANY
  SPI:             400 00000190
  partner address: IPv4   172.25.92.0 - 172.25.92.255
  protocol:        ANY
    ICMP type:       0 - 255
```

```
    ICMP code:       0 - 255
  partner port:    0 - 65535
  own port:        0 - 65535
  mode of use:     transport


No outbound AH security associations!
No inbound AH security associations!
No outbound IPCOMP security associations!
No inbound IPCOMP security associations!
```

6.6.1.2 Automatic key exchange in an IPSec configuration

The equivalent of the example with manually assigned keys and Security Policy Indexes (SPIs) shown in section “Manually defined keys in an IPSec configuration” on page 118 can be presented by the configuration file below.

The security associations are set up by IKE as soon as data communication takes place which is covered by the security policies defined. Not only the configuration file "SYSDAT.IPSEC.*nnn*.RAC2.MANUAL-BSP-DEFAULT" must exist, but also the SPMD password file and the PSK files for the IKE partner instances. The comments for the example above apply analogously. Here it must be borne in mind that the SPI of individual security associations can be assigned "randomly" by the IPSec kernel.

```
include "SYSDAT.IPSEC.nnn.RAC2.MANUAL-BSP-DEFAULT" ;
#
# VM4
# The system with IP address 172.25.92.74 (VM4)
# prefers IKEv1 and requires a different policy.
# Parameters of the default section are overriden.
#
remote VM4 {
   acceptable_kmp { ikev1; };
   ikev1 {
      my_id ipaddr 172.25.92.72;
      peers_id ipaddr 172.25.92.74;
      peers_ipaddr 172.25.92.74 port 500;
      kmp_enc_alg { 3des_cbc;};
      kmp_hash_alg { sha1; md5;};
      kmp_dh_group { modp1536; modp2048; };
      kmp_auth_method { psk; };
      pre_shared_key "VM4.psk";
   };
#
# Selector and policy definition for the peer VM4.
#
selector VM4-in {
   direction inbound;
```

```
   src 172.25.92.74;
   dst 172.25.92.72;
   upper_layer_protocol "any";
   policy_index VM4;
};
selector VM4-out {
   direction outbound;
   dst 172.25.92.74;
   src 172.25.92.72;
   upper_layer_protocol "any";
   policy_index VM4;
};
#
# All IP pakets must be ESP secured.
#
policy VM4 {
   action auto_ipsec;
   remote_index VM4;
   ipsec_mode transport;
   ipsec_index { ipsec_esp-2; };
   ipsec_level require;
};

ipsec ipsec_esp-2 {
   ipsec_sa_lifetime_time 86400 sec;
   ipsec_sa_lifetime_byte 10 MB;
   sa_index esp_01;
};

sa esp_01 {
   sa_protocol esp;
   esp_enc_alg { aes128_cbc; };
   esp_auth_alg { non_auth; };
};
# Selector and policy definition for FTP from and to VM4.
# This is an example for a application/port based policy.
#
selector FTPSERVER-in {
   direction inbound;
   src 172.25.92.74;
   dst 172.25.92.72 port 21;
   upper_layer_protocol "tcp";
   policy_index FTPSERV;
};
selector FTPSERVER-out {
   direction outbound;
   dst 172.25.92.74 port 21;
   src 172.25.92.72;
```

```
   upper_layer_protocol "tcp";
   policy_index FTPSERV;
};
#
# All TCP pakets must be AH+ESP secured.
#
policy FTPSERV {
   action auto_ipsec;
   remote_index VM4;
   ipsec_mode transport;
   ipsec_index { ftp_server; };
   ipsec_level require;
};

ipsec ftp_server {
   ipsec_sa_lifetime_time 3600 sec;
   ipsec_sa_lifetime_byte 1 MB;
   sa_index{ esp_01; ah_01; };
};

sa ah_01 {
   sa_protocol ah;
   ah_auth_alg { hmac_sha1; };
};
# Selector and policy definition for the FTP data of VM4.
#
selector FTPDATA-in {
   direction inbound;
   src 172.25.92.74;
   dst 172.25.92.72 port 20;
   upper_layer_protocol "tcp";
   policy_index FTPDATA;
};
selector FTPDATA-out {
   direction outbound;
   dst 172.25.92.74 port 20;
   src 172.25.92.72;
   upper_layer_protocol "tcp";
   policy_index FTPDATA;
};
#
# All IP pakets must be AH+ESP secured.
#
policy FTPDATA {
   action auto_ipsec;
   remote_index VM4;
   ipsec_mode transport;
   ipsec_index { ftp_data; };
```

```
   ipsec_level require;
};

ipsec ftp_data {
   ipsec_sa_lifetime_time 28800 sec;
   ipsec_sa_lifetime_byte 1 GB;
   sa_index{ esp_01; ah_01; };
};
#
# Selector and policy definition for the default policy DISCARD.
#
selector THEOTHERS-in {
   direction inbound;
   src IP_ANY;
   dst 172.25.92.72/0;
   upper_layer_protocol "any";
   policy_index THEOTHERS;
};
selector THEOTHERS-out {
   direction outbound;
   dst IP_ANY;
   src 172.25.92.72/0;
   upper_layer_protocol "any";
   policy_index THEOTHERS;
};
#
# All IP pakets are discarded unless there is a policy.
#
policy THEOTHERS {
   action discard;
};
```

6.6.1.3 Comments about the examples

SPs and SAs are stored in the database in ascending order on the basis of their selectors. They are also searched in this order.

The security policies specified with most precision appear before those specified with less precision. The measure used to gauge precision is the number of ranges specified.

In our sample configuration, ANY was specified for the local port number for the FTPSERVER SP and individual values for all other selectors. One range was therefore specified. In the case of the VM4 SP, only the local IP address and the partner IP address were specified. Three ranges were therefore specified, and consequently the VM4 SP comes after the FTPDATA and FTPSERVER SPs. If the same number of ranges are

specified, sorting is alphabetical within a selector. The partner port number 20 of the FTPDATA SP is lower than the partner port number 21 of the FTPSERVER SP. FTPDATA therefore comes before FTPSERVER.

The same entries were made for inbound and outbound data traffic, so the SPD also looks the same.

SAs are derived from the SPs.

There are neither inbound nor outbound packets in the SAD for the THEOTHERS SP. This makes sense because there are no SAs required for packets that are to be discarded.

Processing of data packets

The SPD is searched for a suitable security policy for each outbound data packet. If none is found, no IPSec processing takes place. If one is found, the procedure is as specified for the MODE operand:

- BYPASS: There is no IPSec processing.
- DISCARD: The packet is discarded.
- TRANSPORT/TUNNEL: The associated SAs are searched for, and the data packet is encrypted and/or signed accordingly.

The procedure for inbound data packets is analogous. First, a search is carried out for suitable SAs, and the data packet is encrypted and/or verified. Then a search is carried out for a security policy. If one is found, the security protocols used are compared with the requested ones. If they match, the packet is further processed. Otherwise, it is discarded.

The processing of data packets is illustrated in examples based on the configuration described above. A corresponding IPSec configuration must, of course, exist on the partner computers addressed.

1. A TCP packet is to be sent by an application with the port number 3069 on the computer with the IP address 172.25.92.72 to an application with the port number 3069 on the computer with the IP address 172.25.92.74:

   IPSec forms the following search selectors:

   ```
   partner address: IPv4   172.25.92.74
   own address:     IPv4   172.25.92.72
   protocol:        TCP
   partner port:    3069
   own port:        3069
   ```

   It then searches the SPD for outbound packets. The SP with the name VM4 is found. The SP applies to all protocols. The port number 3069 searched for is in the range from 0 to 65535. This requires an ESP SA in transport mode. The selectors formed for the

SP search are used to search the SAD for outbound data packets. The SA with the name SA2 is found. The data packet is encrypted using the specifications in SA2, assigned the SPI 300 and then sent.

If the reply from the computer 172.25.92.74 comes back as a packet secured by ESP, the selectors are used in the SAD to search for inbound ESP packets:

```
own address:     IPv4   172.25.92.74
SPI:             300
```

SA22 is found. The data packet is decrypted with the specifications in SA2. The selectors are then formed for searching for the SP:

```
partner address: IPv4   172.25.92.74
own address:     IPv4   172.25.92.74
protocol:        TCP
partner port:    3069
own port:        3069
```

The VM4 SP is found, and an ESP SA is requested there. The packet is therefore accepted.

2. A TCP packet is to be sent by an application with the port number 3069 on the computer with the IP address 172.25.92.72 to an application with the port number 3069 on the computer with the IP address 172.25.92.74:

   IPSec forms the following search selectors:

   ```
   partner address: IPv4   172.25.92.75
   own address:     IPv4   172.25.92.72
   protocol:        TCP
   partner port:    3069
   own port:        3069
   ```

   It then searches the SPD for outbound packets. The SP with the name MYFRIENDS is found. The subsequent procedure is as in the first example.

3. A TCP packet is to be sent by an application with the port number 3069 on the computer with the IP address 172.25.92.72 to an application with the port number 3069 on the computer with the IP address 172.25.123.137:

   IPSec forms the following search selectors:

   ```
   partner address: IPv4   172.25.123.137
   own address:     IPv4   172.25.92.72
   protocol:        TCP
   partner port:    3069
   own port:        3069
   ```

   It then searches the SPD for outbound packets. The SP with the name THEOTHERS is found. This results in the packet being discarded.

4. An ESP-secured packet is received on the computer 172.25.92.72. The following selectors are used to search for inbound ESP packets in the SAD:

```
own address:     IPv4   172.25.92.72
SPI:             2771
```

   No SA is found. The packet is therefore discarded.

6.6.2 VPN tunnel with IPSec

The example below illustrates how an IPSec tunnel is used.

The BS2000 system (192.168.11.87) and a router (192.168.11.13) are the tunnel's endpoints. The subnet 172.25.226.128 - 172.25.226.255 (in the notation 172.25.226.128/25) is reached via the router which itself has a board in this subnet.

The following security policies are to apply:

1. All data transfer between the home IP address 192.168.11.87 and the subnet 172.25.226.128/25 is to be encrypted between the IP addresses 192.168.11.87 and 192.168.11.13.
2. Tunnel mode is to be used.
3. The encryption methods AES256, AES192 and 3DES are offered/accepted via a selection list.
4. The keys are exchanged automatically. IKEv1 is to be used for authenticating the key management instances, setting up the security associations and key exchange.

The configuration files ROUTER8.PSK, the SPMD password file and the file for the default section SYSDAT.IPSEC.*nnn*.RAC2.MANUAL-BSP-DEFAULT must exist.

The configuration described can then be displayed using the following configuration file:

SYSDAT.IPSEC.014.RAC2

```
include "SYSDAT.IPSEC.014.RAC2.MANUAL-BSP-DEFAULT";

#
# IPSec tunnel tunr8: tunnel to router8
# router8 speaks IKEv1 and the preshared key is stored
# in the file "ROUTER8.psk".
# Parameters of the default section are overridden.
#

remote tunr8 {
   acceptable_kmp { ikev1; };
   ikev1 {
```

```
      my_id ipaddr 192.168.11.87;
      peers_ipaddr 192.168.11.13 port 500;
      kmp_enc_alg { 3des_cbc;};
      kmp_hash_alg { sha1; };
      kmp_dh_group { modp1024; };
      kmp_auth_method { psk; };
      pre_shared_key "ROUTER8.psk";
   };
};
_____________________________________________________________________(1)


#
# Selector and policy definition for the subnet 172.25.226.128/25
#

selector tunr8-in {
   direction inbound;
   src 172.25.226.128/25;
   dst 192.168.11.87;
   upper_layer_protocol "any";
   policy_index tunr8;
};
selector tunr8-out {
   direction outbound;
   dst 172.25.226.128/25;
   src 192.168.11.87;
   upper_layer_protocol "any";
   policy_index tunr8;
};
_____________________________________________________________________(2)

policy tunr8 {
   action auto_ipsec;
   remote_index tunr8;
   ipsec_mode tunnel;
   ipsec_index { ipsec_esp_multi; };
   ipsec_level require;
   peers_sa_ipaddr 192.168.11.13;
   my_sa_ipaddr 192.168.11.87;
};
_____________________________________________________________________(3)

ipsec ipsec_esp_multi {
   ipsec_sa_lifetime_time 28800 sec;
   sa_index esp_multi;
};
```

```
sa esp_multi {
   sa_protocol esp;
   esp_enc_alg { aes256_cbc; aes192_cbc; 3des_cbc; };
   esp_auth_alg { non_auth; };
};
_____________________________________________________________________(4)
```

Explanations:

(1) The remote statement contains the kmp parameters for mutual authentication of the IKE instances. IKEv1 is used (point 4.).

(2) Any data transfer (i.e. using any protocol) between the system 192.168.11.87 and a system in the subnet 172.25.226.128/25 is to use the tunnel between 192.168.11.87 and 192.168.11.13 (point 1.).

(3) The policy between the systems 192.168.11.87 and 192.168.11.13 uses tunnel mode (point 2.).

(4) AES256, AES192 and 3DES are offered/accepted as encryption methods (point 3.).

6.6.3 IP Payload Compression Protocol in IPSec

The following security policies apply for the example below:

1. The data transfer between the systems with the IP addresses 172.25.226.220 and 172.25.226.221 is secured by an ESP SA in transport mode.
2. The encryption methods AES and 3DES and authentication methods HMAC-SHA1 and HMAC-MD5 are proposed/accepted.
3. Compression of the IP payload using the compression algorithm DEFLATE is permitted.
4. Mutual authentication of the key management instances, setup of the ESP SA and the key exchange must take place using the Internet Key Exchange Protocol (IKEv1).

The default values are contained in the SYSDAT.IPSEC.*nnn*.RAC2.MANUAL-BSP-DEFAULT file, which is loaded using an INCLUDE statement.

The configuration files VM1.PSK and SYSDAT.IPSEC.*nnn*.RAC2 look like this:

VM1.PSK file

```
SECRET
```

SYSDAT.IPSEC.*nnn*.RAC2 file

```
include "SYSDAT.IPSEC.nnn.RAC2.MANUAL-BSP-DEFAULT";

#
# my own configuration
#
remote VM1_IKE1 {
   acceptable_kmp { ikev1; };
   ikev1 {
      my_id ipaddr 172.25.226.220;
      peers_ipaddr 172.25.226.221 port 500;
      kmp_enc_alg { 3des_cbc;};
      kmp_hash_alg { sha1; md5;};
      kmp_dh_group { modp1024; };
      kmp_auth_method { psk; };
      pre_shared_key "VM1.psk";
   };
};
_____________________________________________________________________(1)

selector VM1_1 {
   direction inbound;
   src 172.25.226.221;
   dst 172.25.226.220;
   upper_layer_protocol "any";
   policy_index VM1;
};

selector VM1_2 {
   direction outbound;
   dst 172.25.226.221;
   src 172.25.226.220;
   upper_layer_protocol "any";
   policy_index VM1;
};
_____________________________________________________________________(2)

policy VM1 {
   action auto_ipsec;
   remote_index VM1_IKE1;
   ipsec_mode transport;
   ipsec_index { ipsec_IPCOMP_def_ESP_md5-sha1.aes-3des; };
   ipsec_level require;
};
```

```
_____________________________________________________________________(3)
ipsec ipsec_IPCOMP_def_ESP_md5-sha1.aes-3des {
   ipsec_sa_lifetime_time 28800 sec;
   sa_index { IPCOMP_def; ESP_md5-sha1.aes-3des; };
};

sa ESP_md5-sha1.aes-3des {
   sa_protocol esp;
   esp_auth_alg { hmac_md5; hmac_sha1; };
   esp_enc_alg { aes128_cbc; 3des_cbc; };
};
_____________________________________________________________________(4)
sa IPCOMP_def {
   sa_protocol ipcomp;
   ipcomp_alg { deflate; };
};
_____________________________________________________________________(5)
```

Explanations:

(1) The remote statement contains the kmp parameters for mutual authentication of the IKE instances. IKEv1 is used (point 4.).

(2) Data transfer between the systems 172.25.226.220 and 172.25.226.221 is secured by an IPSec ESP SA in transport mode (point 1.).

(3) The policy uses IKEv1 (point 4.).

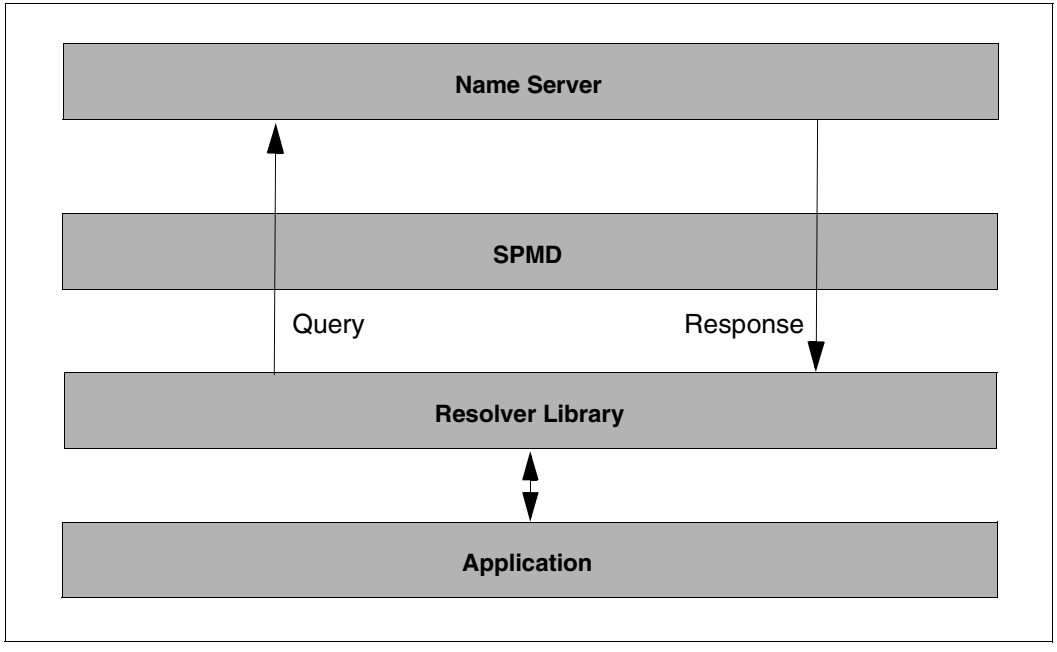
(4) The encryption methods AES and 3DES and authentication methods HMAC-SHA1 and HMAC-MD5 are proposed/accepted (point 2.).

(5) DEFLATE is permitted as the compression algorithm for IP payload (point 3.).

6.6.4 Supporting the Domain Name System (DNS)

DNS support means that FullyQualified Domain Names (FQDNs) are used instead of IPv4 or IPv6 addresses.

To support the Domain Name System (DNS), SPMD is located as a proxy between the application and the name server:



SPMD sends queries for FQDNs which are specified in the configuration file. SPMD recognizes incoming queries and forwards them to the name server. SPMD analyzes the name server's response and uses it to resolve the names specified in the configuration file. The security policies with IPv4 or IPv6 addresses are made known to the TPR subsystem.

The example below shows what you must do to be able to use IPSec with FQDNs. A client/server configuration is used here. The client is a computer which has no permanent IP address. It is consequently specified in the configuration file by means of its name. The server's permanent IP address, on the other hand, is known to the clients.

- Change the name server entry in the resolver configuration file to:

```
nameserver 127.0.0.1
```

► Then stop and restart LWRESD so that the change becomes effective:

/STOP-LWRESD
/START-LWRESD

Sample configuration file SYSDAT.IPSEC.014.RAC2

```
#
# include of the default configuration file with the default parameter
#
include SYSDAT.IPSEC.014.RAC2.MANUAL-BSP-DEFAULT ;
# resolver info

resolver
{
# spmd acts as a resolver
   resolver on;
# this is the DNS server
   nameserver {172.25.92.122  port 53;};
# spmd listens to dns queries on loopback
   dns_query {127.0.0.1 port 53;};
};

#
# VM2
# The client system VM2 must use IKEv2
# Parameters of the default section are overriden.
#

remote VM2 {
   acceptable_kmp { ikev2; };
   ikev2 {
      my_id ipaddr 172.25.92.74;
      peers_id fqdn "BCABLZ02.SO13.MCH.FTS.NET";
      kmp_enc_alg { 3des_cbc;};
      kmp_hash_alg { hmac_md5;};
      kmp_dh_group { modp1536; };
      kmp_prf_alg { hmac_md5;  };
      kmp_auth_method { psk; };
      pre_shared_key "VM2.psk";
   };
};

#
# Selector and policy definition for the peer VM4.
# All IP pakets must be ESP secured.
#

selector VM2-out {
   direction outbound;
```

```
   src 172.25.92.74;
   dst "BCABLZO2.SO13.MCH.FTS.NET";
   upper_layer_protocol "tcp";
   policy_index VM2;
};

selector VM2-in {
   direction inbound;
   src "BCABLZO2.SO13.MCH.FTS.NET";
   dst 172.25.92.74;
   upper_layer_protocol "tcp";
   policy_index VM2;
};

policy VM2 {
   action auto_ipsec;
   remote_index VM2;
   ipsec_mode transport;
   ipsec_index { ipsec_esp; };
   ipsec_level require;
   peers_sa_ipaddr IP_RW;
   my_sa_ipaddr 172.25.92.74;
};

ipsec ipsec_esp {
   ipsec_sa_lifetime_time 28800 sec;
   sa_index esp_01;
};

sa esp_01 {
   sa_protocol esp;
   esp_enc_alg { aes128_cbc;  };
   esp_auth_alg { hmac_sha1;  };
};
```

6.6.5 Supporting Network Address Translation (NAT)

NAT traversal is required to permit data protected by IPSec to pass NAT devices.

The example below shows how IPSec supports NAT traversal. A client/server configuration is used here. The client is a computer which has no permanent IP address. It is consequently specified in the configuration file by means of its name. The server's permanent IP address, on the other hand, is known to the clients.

- Change the name server entry in the resolver configuration file to:

  ```
  nameserver 127.0.0.1
  ```

- Then stop and restart LWRESD so that the change becomes effective:

  /STOP-LWRESD
  /START-LWRESD

Sample configuration file SYSDAT.IPSEC.014.RAC2

```
#
# include of the default configuration file with the default parameter
#

include SYSDAT.IPSEC.014.RAC2.MANUAL-BSP-DEFAULT ;

# interface info

interface
{
   ike {
      MY_IP port 500;
# The support of NAT-Traversal requires an additional instruction
# with port 4500;
      MY_IP port 4500;
   };
};

# resolver info
resolver
{
# spmd acts as a resolver
   resolver on;
# this is the DNS server
   nameserver {172.25.92.122  port 53;};
# spmd listens to dns queries on loopback
   dns_query {127.0.0.1 port 53;};
};

# VM2
# The client system VM2 must use IKEv2
```

```
remote VM2 {
   acceptable_kmp { ikev2; };
   ikev2 {
      my_id ipaddr 10.0.0.4;
      peers_id fqdn "BCABLZO2.SO13.MCH.FTS.NET";
      kmp_enc_alg { 3des_cbc;};
      kmp_hash_alg { hmac_md5;};
      kmp_dh_group { modp1536; };
      kmp_prf_alg { hmac_md5;  };
      kmp_auth_method { psk; };
      pre_shared_key "VM2.psk";
   };
};

#
# Selector and policy definition for the peer VM2.
# All IP pakets must be ESP secured.
#

selector VM2-out {
   direction outbound;
   src 10.0.0.4;
   dst "BCABLZO2.SO13.MCH.FTS.NET";
   upper_layer_protocol "tcp";
   policy_index VM2;
};

selector VM2-in {
   direction inbound;
   src "BCABLZO2.SO13.MCH.FTS.NET";
   dst 10.0.0.4;
   upper_layer_protocol "tcp";
   policy_index VM2;
};

policy VM2 {
   action auto_ipsec;
   remote_index VM2;
   ipsec_mode transport;
   ipsec_index { ipsec_esp; };
   ipsec_level require;
   peers_sa_ipaddr IP_RW;
   my_sa_ipaddr 10.0.0.4;
};

ipsec ipsec_esp {
   ipsec_sa_lifetime_time 28800 sec;
   sa_index esp_01;
};
```

```
sa esp_01 {
   sa_protocol esp;
   esp_enc_alg { aes128_cbc;  };
   esp_auth_alg { hmac_sha1;  };
};
```

7 IPSec messages

The IPSec messages have the message class YIS. All IPSec messages thus begin with YIS and are followed by a four-digit decimal number with leading zeros. It has been decided to dispense with a full list of messages here.

You can display the text of a message whose number you know by entering the following command:

```
HELP-MSG-INFORMATION MSG-IDENTIFICATION=YIS<nnnn>
```

You can create a full list of all the messages in the message class YIS by using the program MSGMAKER. This displays the current contents of the IPSec message file:

```
SYSMES.IPSEC.<nnn>
```

To start MSGMAKER, enter the following command:

```
/START-MSGMAKER
```

The user guidance provided in MSGMAKER makes it easy to create the list of IPSec messages.

8 Appendix

8.1 Position of the IP Payload Compression Protocol (IPCOMP)

The IP Payload Compression Protocol is always positioned immediately ahead of the transport protocol, in other words before the original payload of an IP segment. IPCOMP enables data compression which may be wished for on Layer 2 to be shifted to Layer 3. If this measure were not taken, no compression would take place owing to encryption by IPSec.

Since IPCOMP does not provide any transformation which is used to secure data communication, it is not described in the sections below.

8.2 Position of the security protocols

Position of the AH relative to the IPv4 header

The Authentication Header is positioned as follows in the IPv4 packet:

- Immediately after the original IP header, and
- Before every protocol header (TCP, UDP, ICMP, etc.) of a higher layer of the IP stack, and
- Before every other IPSec header

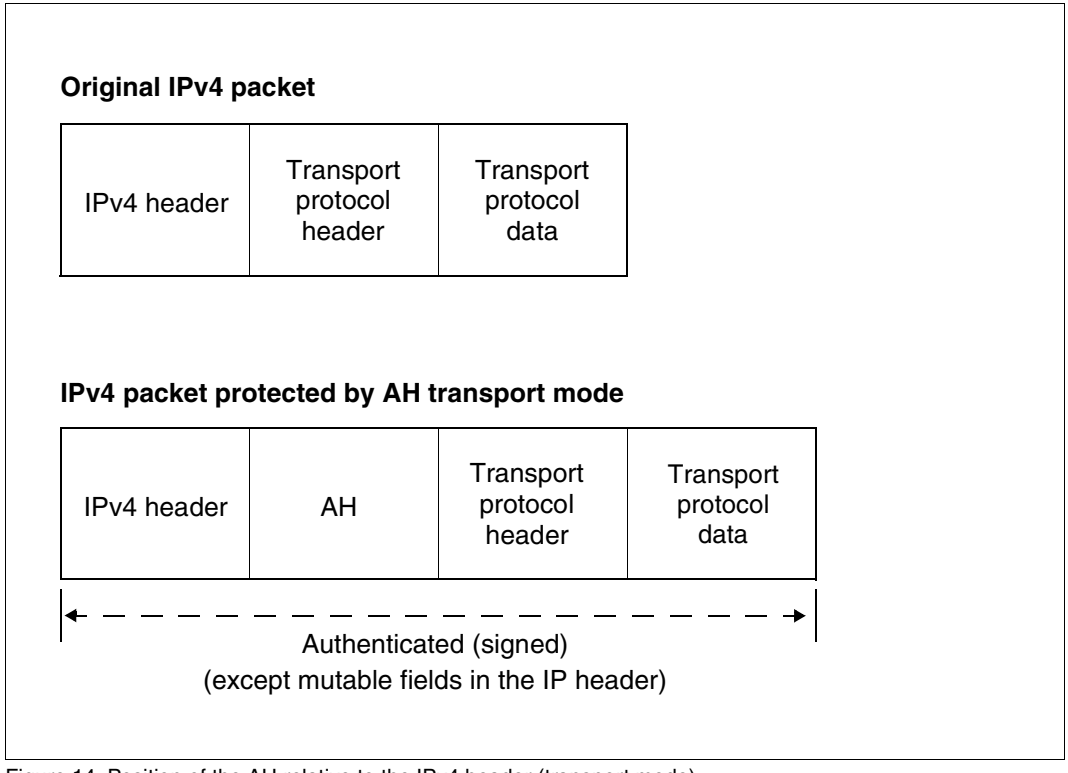


Figure 14: Position of the AH relative to the IPv4 header (transport mode)

Position of the AH relative to the IPv6 header

In the IPv6 packet, the AH is inserted after the extension headers for hop-by-hop, routing and fragmentation.

The following applies to the position of the AH relative to the destination option header(s):

- If the destination option header(s) is/are to be processed by security gateways (routers) specified in the destination field of the IPv6 header, the AH must be positioned after the destination option extension header(s).
- If the destination option header is only to be processed by the destination host, the AH must be inserted in front of the destination option header(s).

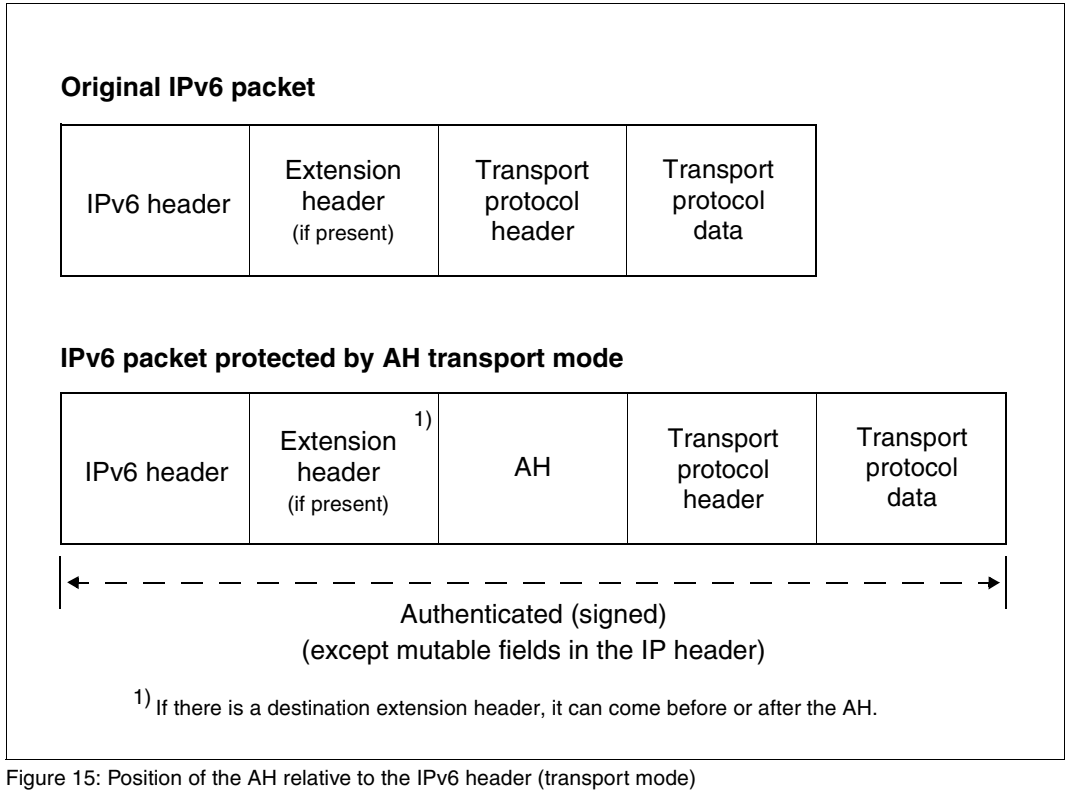


Figure 15: Position of the AH relative to the IPv6 header (transport mode)

Position of the AH relative to the IPv4 header

In tunnel mode, the AH is positioned immediately before the original IPv4 header, and a new IPv4 header is positioned before the AH.

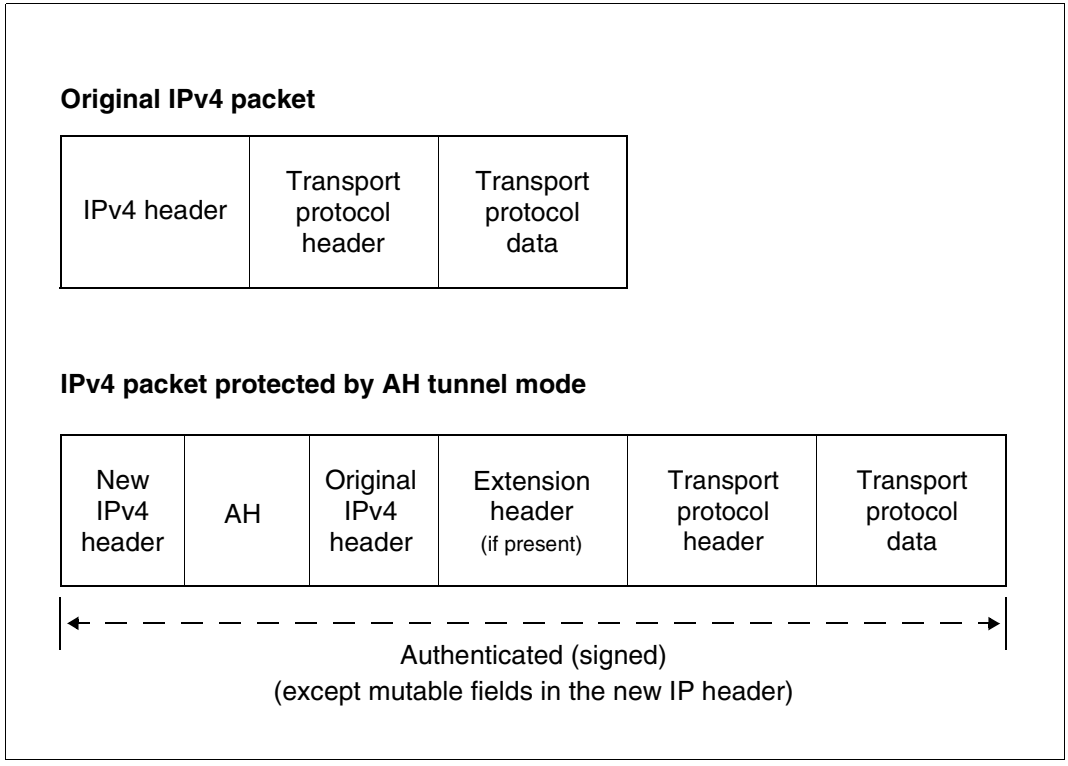


Figure 16: Position of the AH relative to the IPv4 header (tunnel mode)

Position of the AH relative to the IPv6 header

In tunnel mode, the AH is positioned immediately before the original IPv6 header, and the AH is preceded by a new IPv6 header plus any extension headers that may exist.

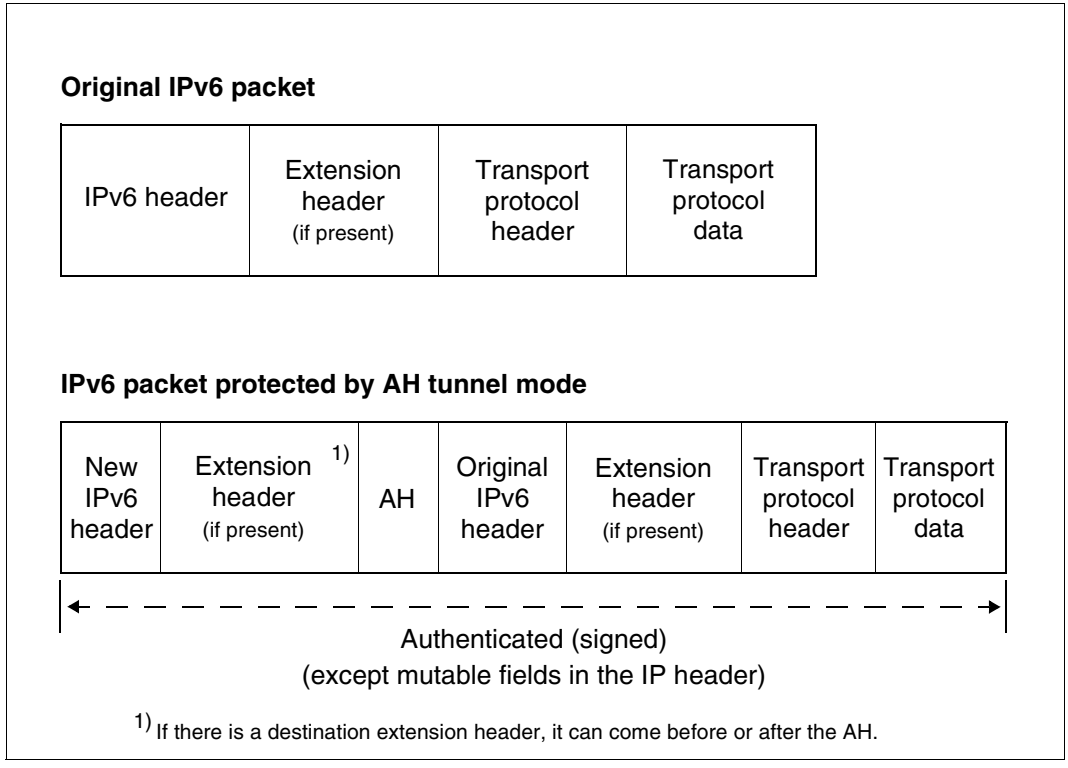


Figure 17: Position of the AH relative to the IPv6 header (tunnel mode)

Position of the ESP header relative to the IPv4 header

In the IPv4 header, the ESP header is inserted immediately after the IPv4 header (including the associated options) and before the protocol header of a higher layer:

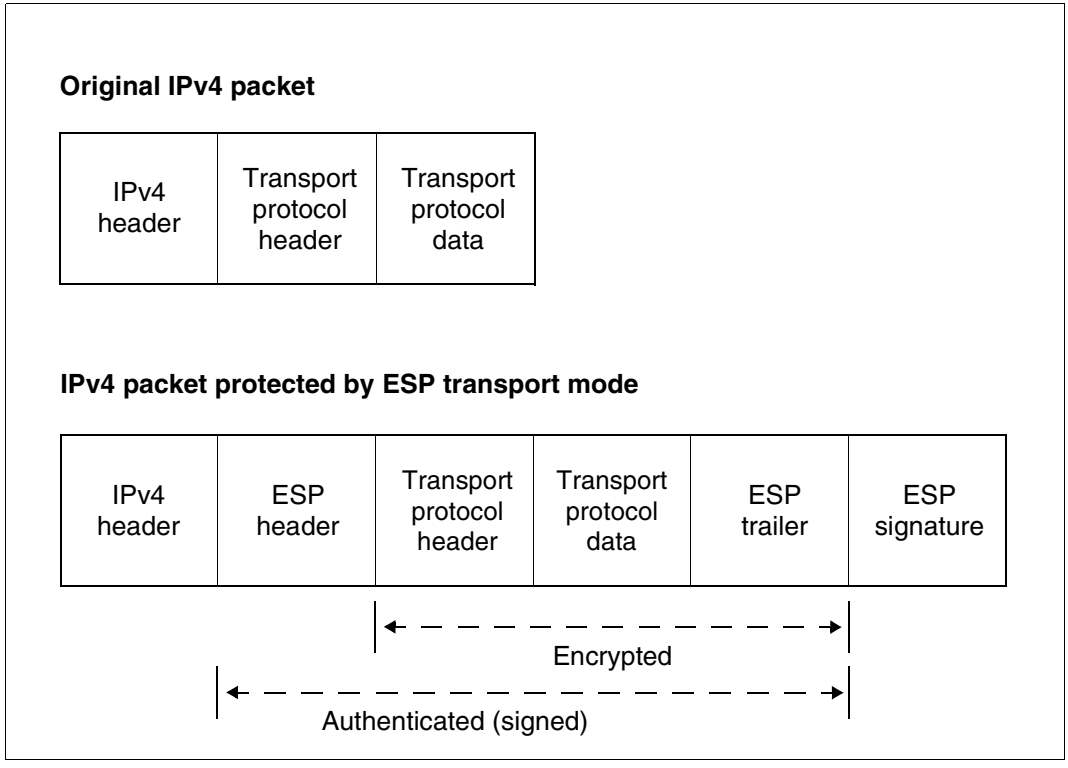


Figure 18: Position of the ESP header relative to the IPv4 header (transport mode)

Position of the ESP header relative to the IPv6 header

In the IPv6 protocol, the ESP header is inserted after the extension headers for hop-by-hop, routing and fragmentation.

The following applies to the position of the ESP header relative to the destination option header(s):

- If the destination option header(s) is/are to be processed by systems (routers) specified in the destination field of the IPv6 header, the ESP header must be positioned after the destination option extension header(s).
- If the destination option header is only to be processed by the destination host, the ESP header can also be inserted after the destination option header(s).

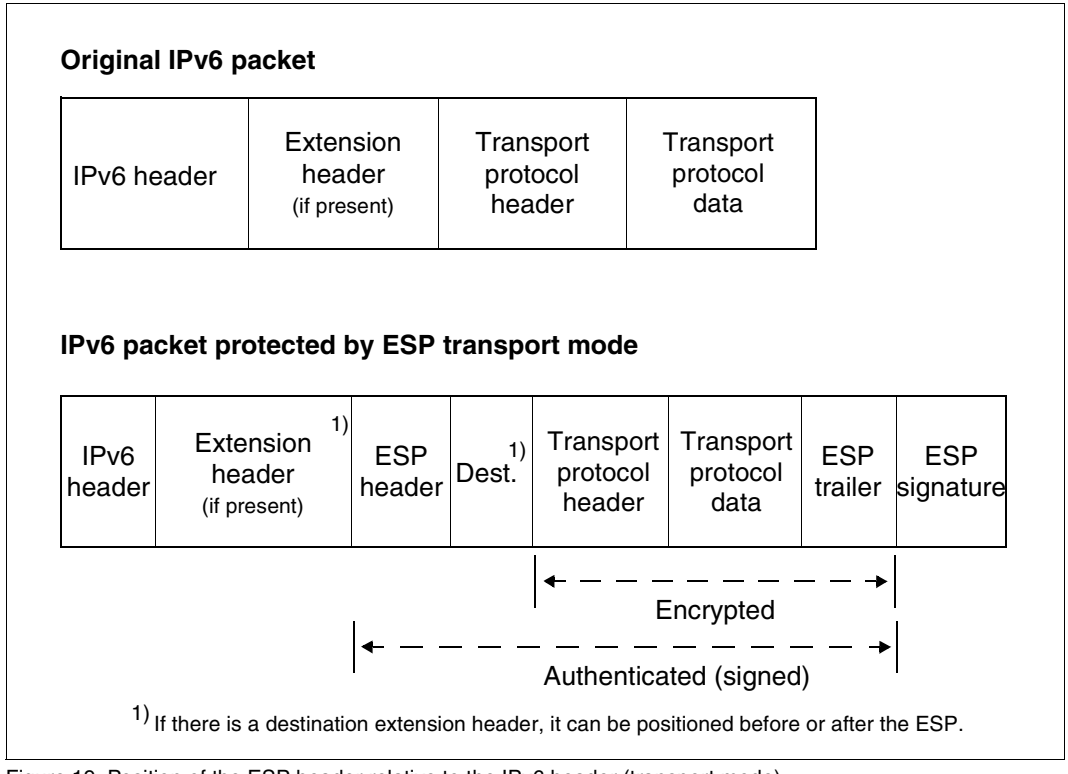


Figure 19: Position of the ESP header relative to the IPv6 header (transport mode)

Position of the ESP header relative to the IPv4 header

In tunnel mode, the ESP is positioned before the original IPv4 header, and the new IPv4 header precedes the ESP header.

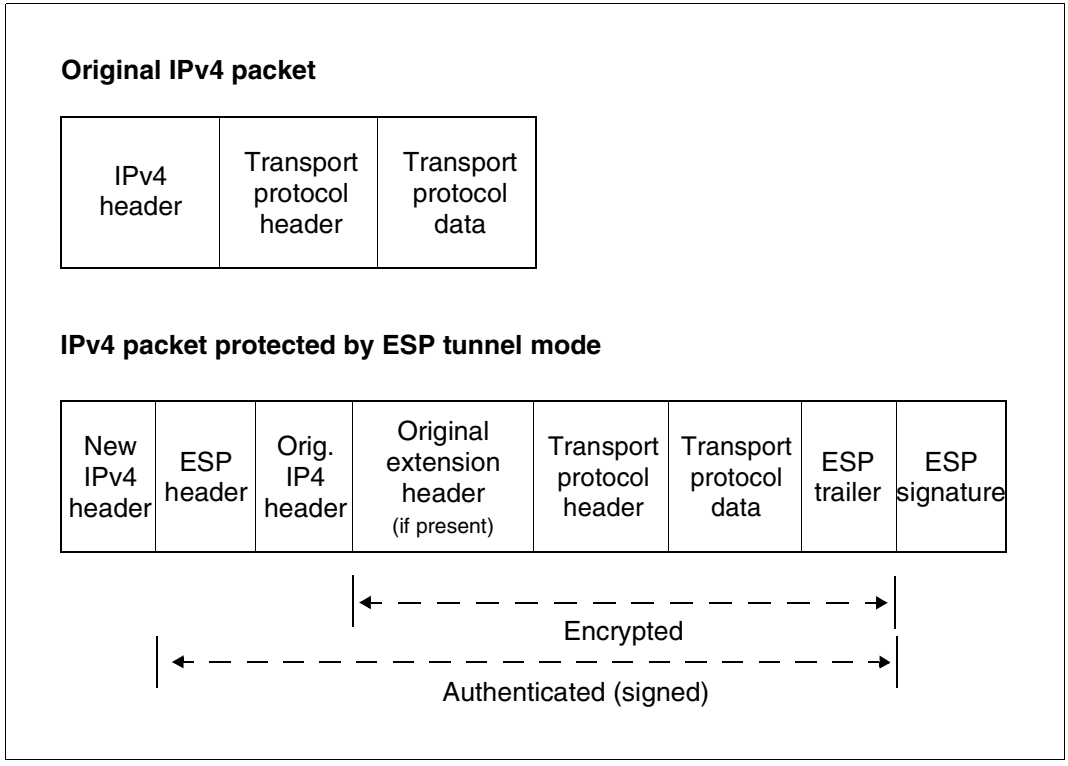


Figure 20: Position of the ESP header relative to the IPv4 header (tunnel mode)

Position of the ESP header relative to the IPv6 header

In tunnel mode, the ESP is positioned before the original IPv6 header, and the new IPv6 header together with any extension headers that may exist precede the ESP.

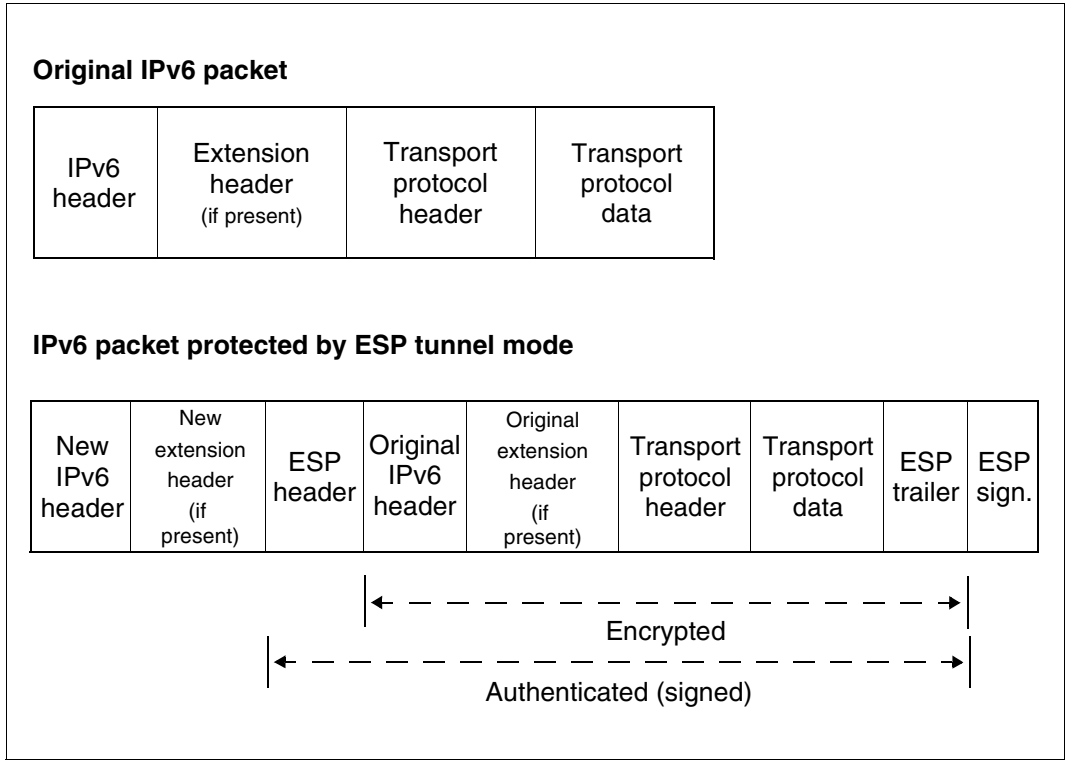


Figure 21: Position of the ESP header relative to the IPv6 header (tunnel mode)

8.3 Security concepts based on security associations

This section describes:

- Security concepts supported by IPSec when ESP and AH headers are used in transport mode and tunnel mode (transfer modes)
- Ways in which AH and ESP headers can be combined in IP packets
- The range of security policies that can be implemented on the basis of the AH and ESP protocols and transport mode and tunnel mode
- Examples of user scenarios

8.3.1 Security concepts based on transport-mode and tunnel-mode SAs

Security concepts based on transport-mode SAs

This section outlines the security concepts that can be implemented by using AH and ESP headers in transport mode.

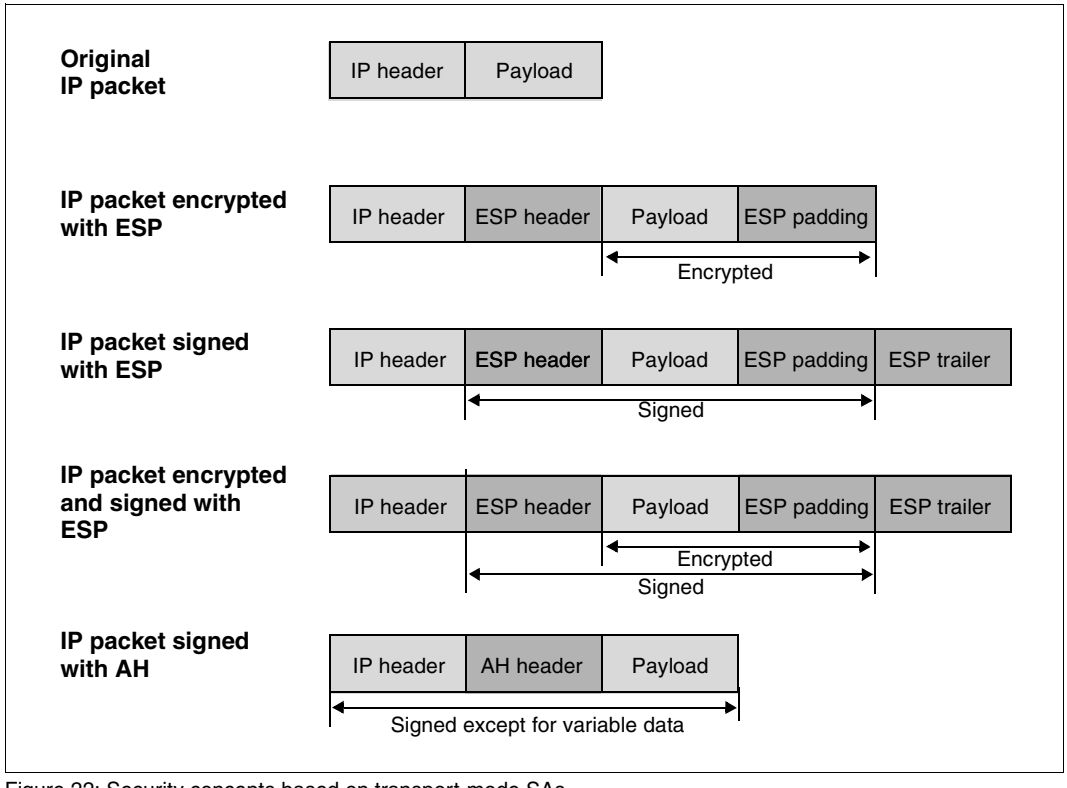


Figure 22: Security concepts based on transport-mode SAs

Security concepts based on tunnel-mode SAs

This section outlines the security concepts that can be implemented by using AH and ESP headers in tunnel mode.

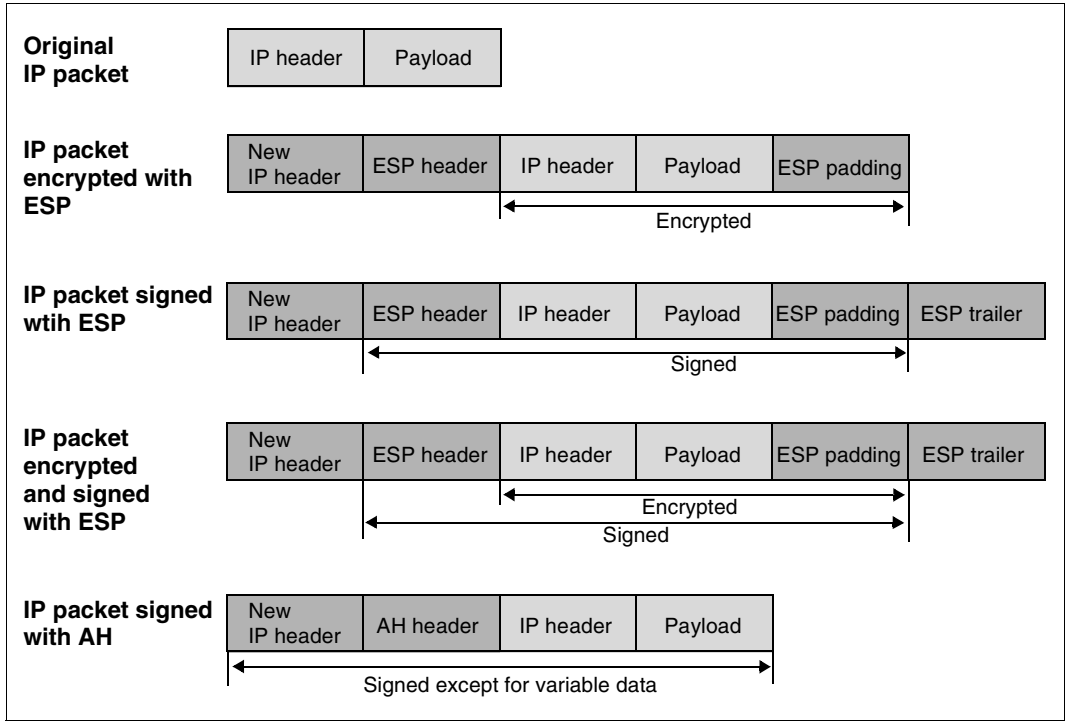


Figure 23: Security concepts based on tunnel-mode SAs

8.3.2 Combining the AH and ESP

This section describes the ways in which IPSec allows the Authentication Header and Encapsulating Security Payload to be combined for inbound and outbound IP traffic.

AH and ESP combinations for inbound IP traffic

IPSec supports the following AH and ESP combinations for inbound IP traffic.

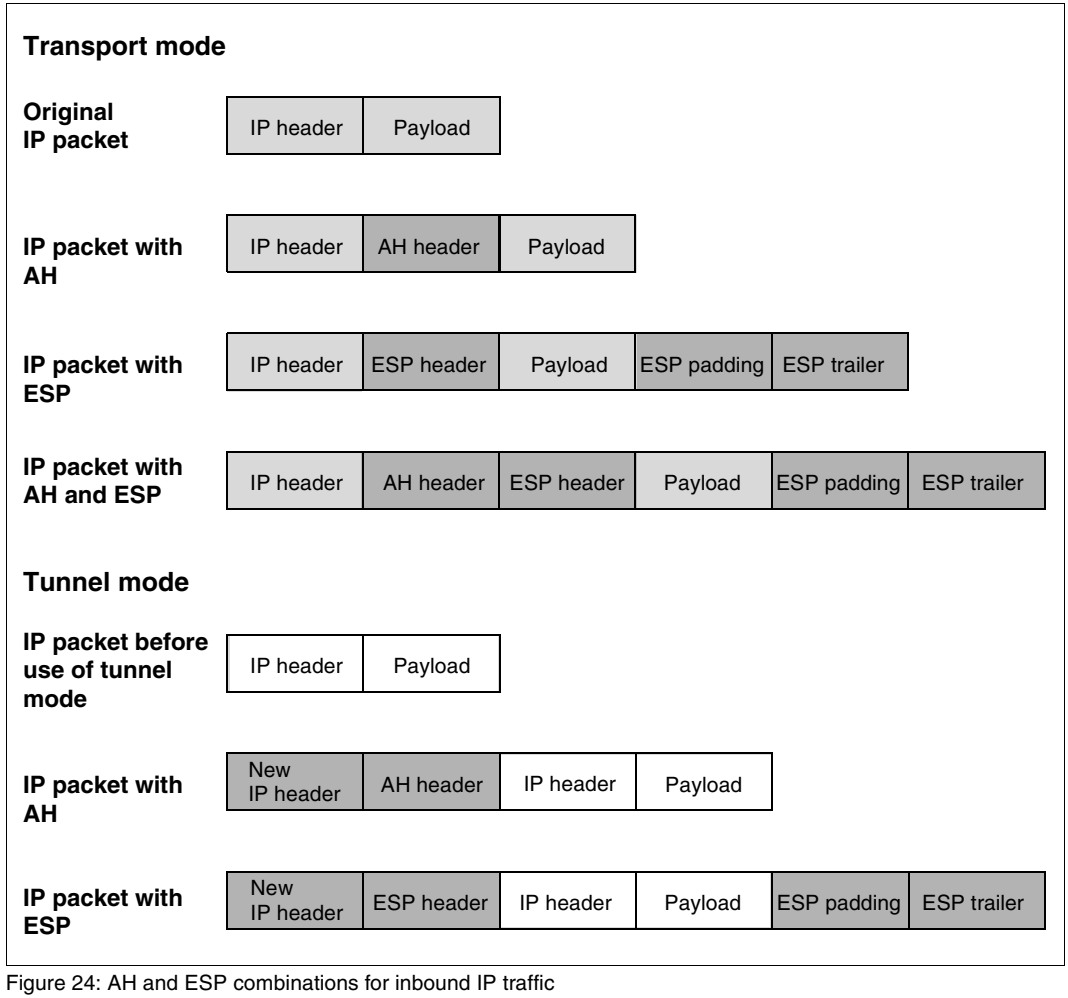


Figure 24: AH and ESP combinations for inbound IP traffic

AH and ESP combinations for outbound IP traffic

For outbound IP traffic, IPSec also supports other sequences of Authentication Headers and Encapsulating Security Payload headers. However, the prerequisite is that an associated security policy must exist for outbound data.

8.3.3 Range of security policies supported by IPSec

With its Authentication Header and Encapsulating Security Payload headers, IPSec permits a wide range of different security policies.

The header layouts described on the following pages reflect the different security philosophies.

Header layout in transport mode

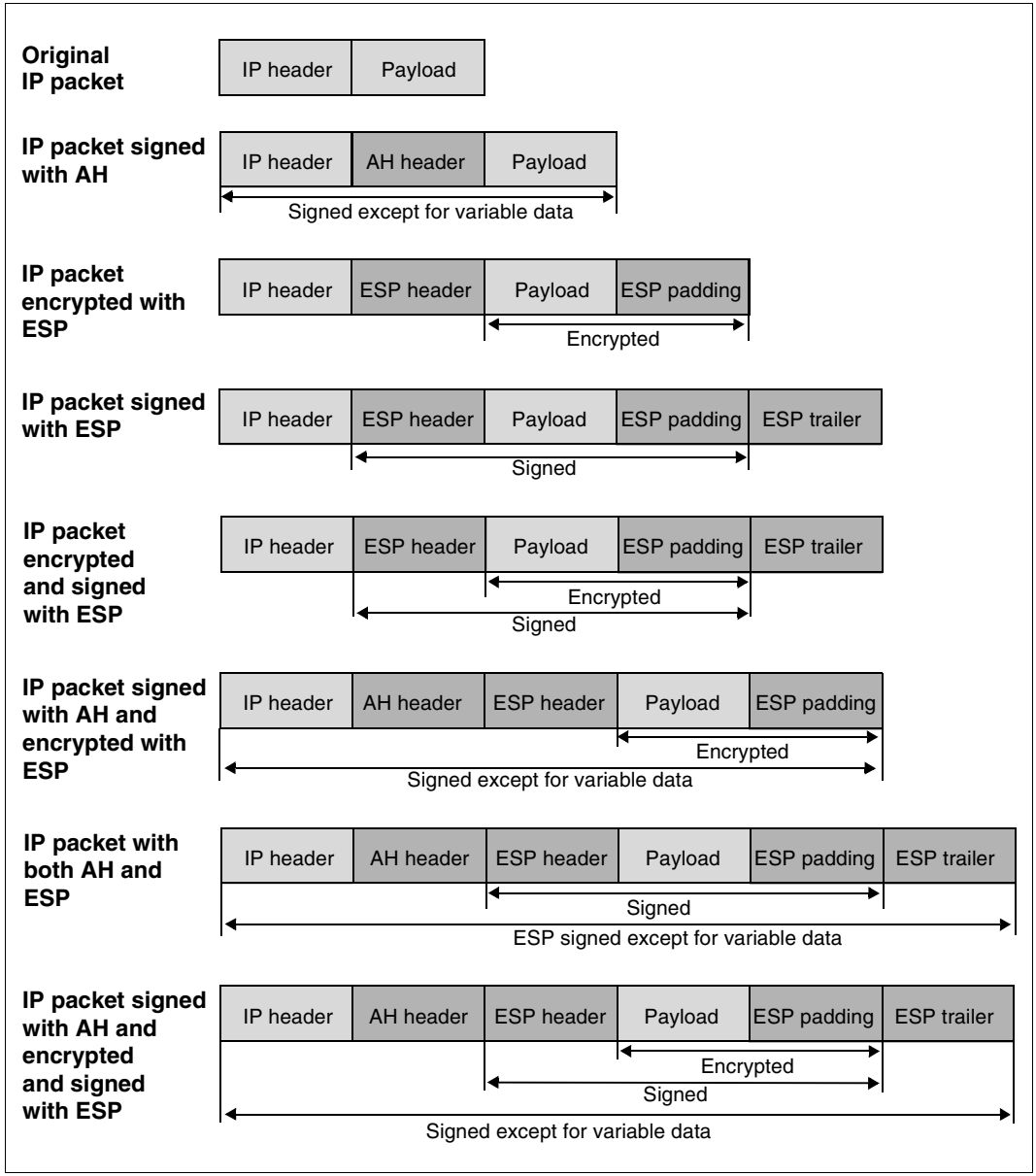


Figure 25: Header layout in transport mode

Header layout in tunnel mode with the Authentication Header in the tunnel

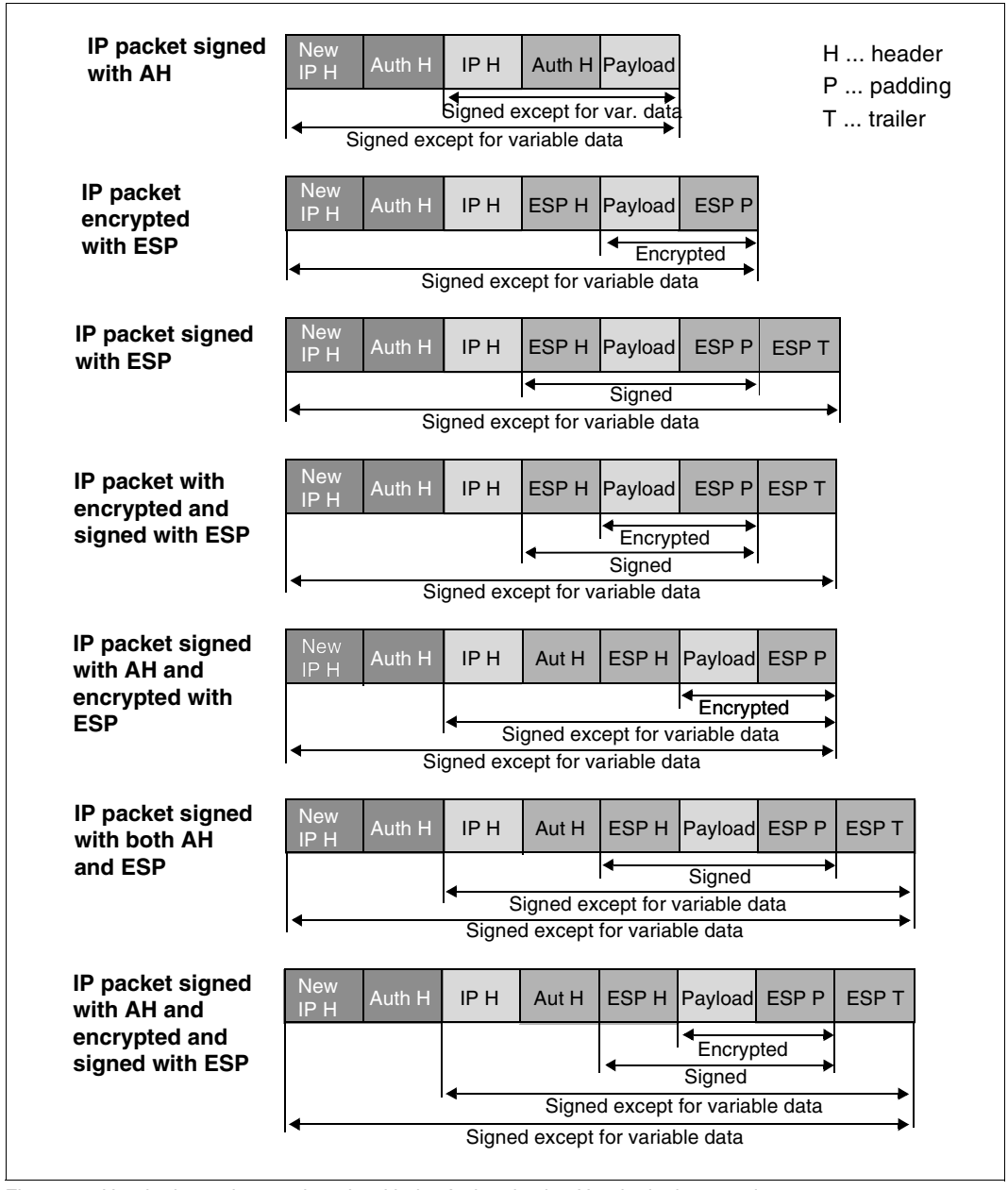


Figure 26: Header layout in tunnel mode with the Authentication Header in the tunnel

Header layout in tunnel mode with ESP encryption in the tunnel

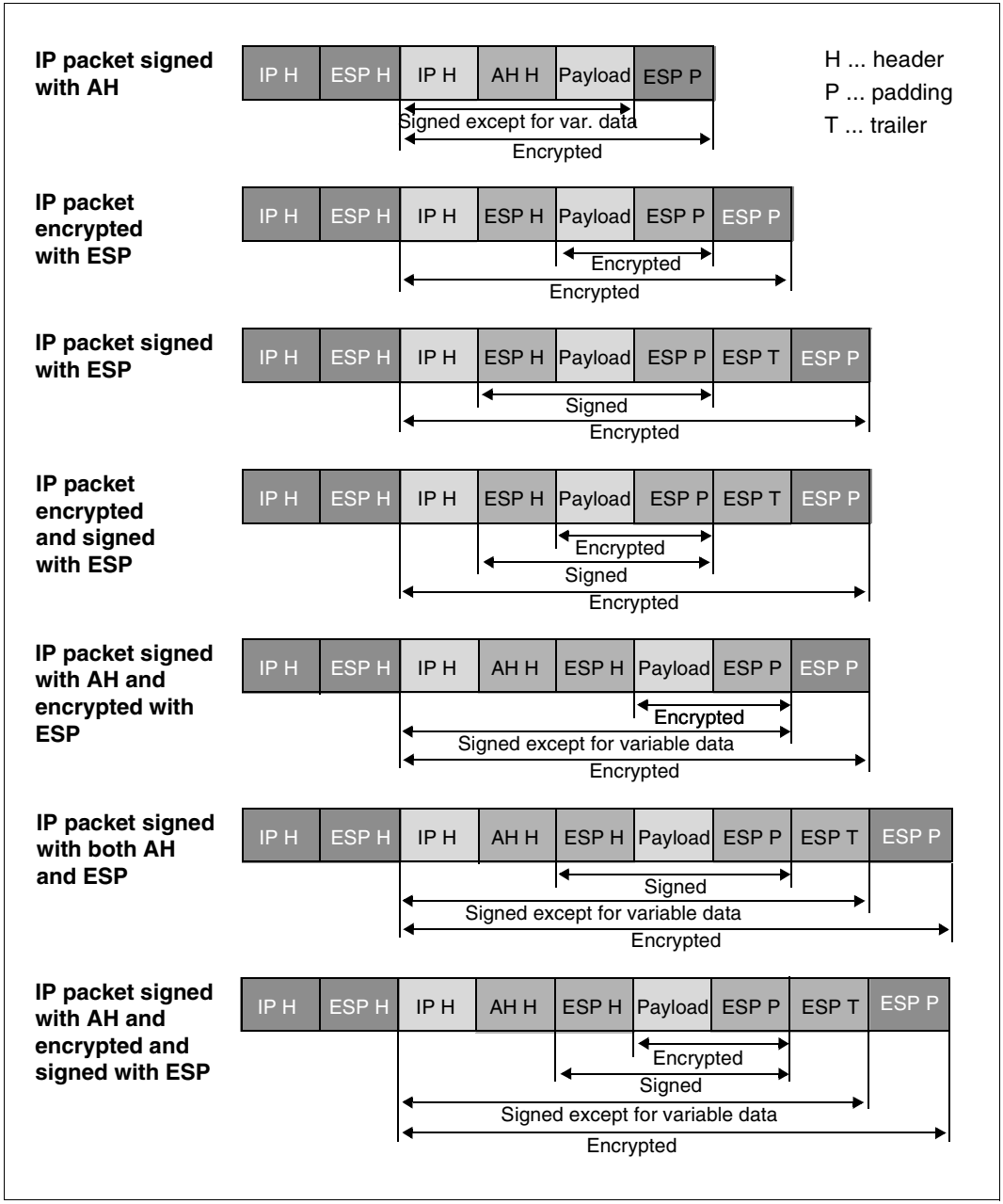


Figure 27: Header layout in tunnel mode with ESP encryption in the tunnel

Header layout in tunnel mode with ESP signature in the tunnel

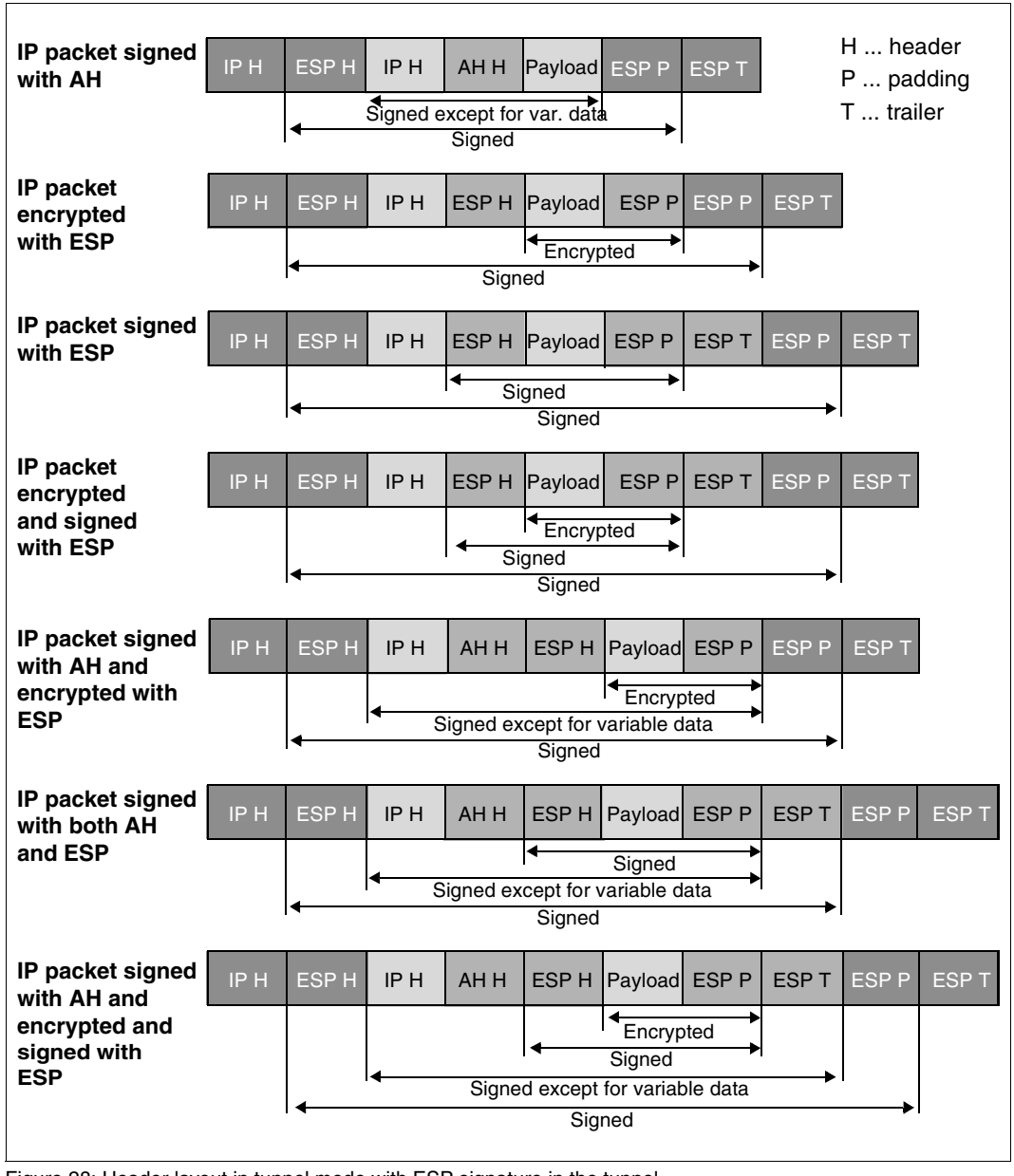


Figure 28: Header layout in tunnel mode with ESP signature in the tunnel

Header layout in tunnel mode with ESP encryption and signature in the tunnel

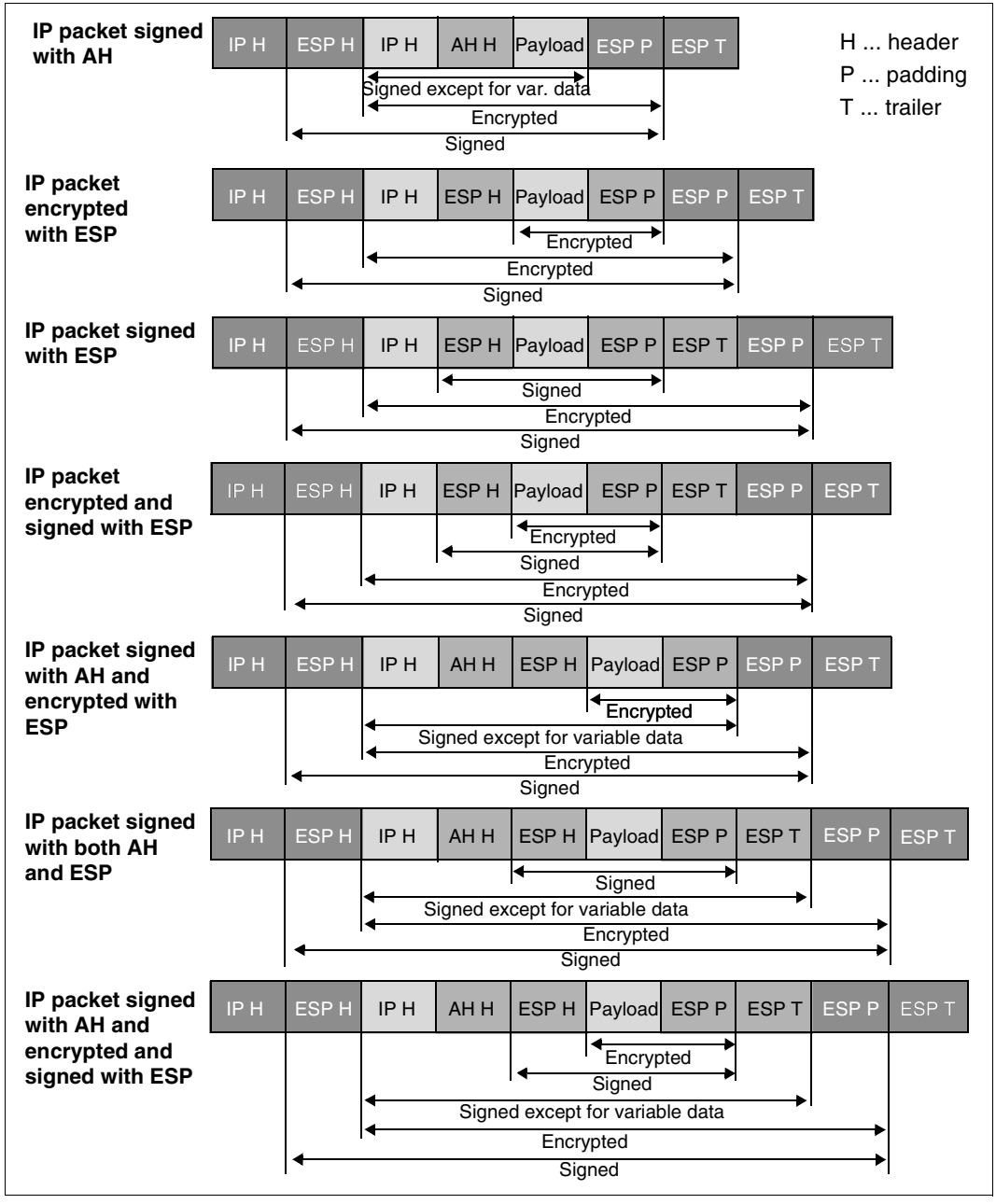


Figure 29: Header layout in tunnel mode with ESP encryption in the tunnel

Header layout in tunnel mode without inner IPSec header

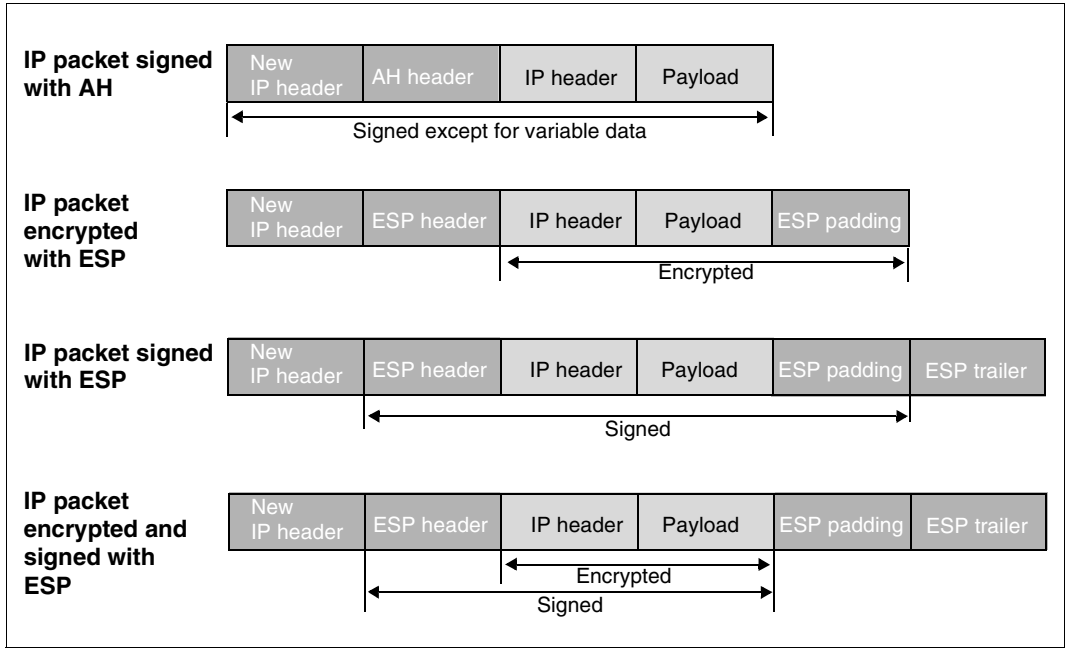


Figure 30: Header layout in tunnel mode without inner IPSec header

Abbreviations

AH	Authentication Header
DNS	Domain Name System
ESP	Encapsulating Security Payload
FQDN	Fully Qualified Domain Name
HBCI	Home Banking Computer Interface
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
IPCOMP	IP Payload Compression Protocol
ICV	Integrity Check Value
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange Protocol
IP	Internet Protocol
IPSec	IP Security
ISAKMP	Internet Security Association and Key Management Protocol
IV	Initialization Vector
KMP	Key Management Protocol
MAC	Message Authentication Code
NAT	Network Address Translation
NAT-T	Network Address Translation - Traversal
NEA	(Name of the Transdata architecture)
PFS	Perfect Forward Secrecy
PKI	Public Key Infrastructure
PSK	Preshared Secret Key
RFC	Request for Comment
SA	Security Association
SAD	Security Association Database

SAT	Security Audit Trail
SET	Secure Electronic Transaction
SP	Security Policy
SPD	Security Policy Database
SPI	Security Parameter Index
SPMD	Security Policy Management Daemon
SSL	Secure Sockets Layer
TLS	Transport Layer Security
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
ULP	Upper Layer Protocol

Glossary

AES (Advanced Encryption Standard)

The Advanced Encryption Standard (AES) is a symmetric cryptographic method issued as a standard by the National Institute of Standards and Technology (NIST) in October 2000 as the successor of DES and 3DES. It is also known as the Rijndael algorithm (pronounced “Raindahl”) after its Belgian developers Joan Daemen and Vincent Rijmen.

ANSI (American National Standards Institute)

This institute develops standards for various accredited standards committees (ASC). The X9 committee is mainly concerned with security standards for financial services.

Asymmetric encryption

In asymmetric encryption, each communication partner has two keys: a private (secret) key and a public key. These two keys are mathematically related.

Block cipher

A block cipher is an algorithm that encrypts a data block by means of a key value. The encrypted block has the same length.

CBC (cipher block chaining)

Cipher block chaining mode is a cryptographic mode in which block ciphers can be used. Before a plaintext block is encrypted, it is XORed with the secret text block created in the previous step.

Cipher text

This is the result of a change made to letters or bits by replacing them, exchanging them or replacing and exchanging them.

Cryptoki

A program interface to devices that save cryptographic information and execute cryptographic functions. Specified by the PKSC#11 standard.

Decryption

The process of converting ciphered (or encrypted) text back to plaintext.

DES (Data Encryption Standard)
A 64-bit block cipher or symmetric algorithm that is also referred to as the Data Encryption Algorithm (DEA) (ANSI) or DEA-1 (ISO).

Diffie-Hellman
Diffie-Hellman key exchange is a protocol derived from cryptography. Two communication partners create a secret key known only to them.

FIPS (Federal Information Processing Standard)
A US government standard published by NIST.

Hash function
A one-way hash function is a function that converts a large volume of data to a generally smaller volume of data. The process cannot be reversed to recreate the original.

HMAC (keyed-hash message authentication code)
A key-dependent one-way hash function specifically designed for use with MAC (Message Authentication Code) and based on IETF RFC 2104.

IETF (Internet Engineering Task Force)
The Internet Engineering Task Force (IETF) is an organization that concerns itself with the technical development of the Internet.

IKE (Internet Key Exchange)
A protocol for setting up and managing SAs

Initialization vector or IV
A block of random data that uses “chaining feedback mode” (see “CBC (cipher block chaining)”) and serves as the starting point for a block cipher.

Integrity
Proof that data has not been changed (by unauthorized persons) during saving or transfer.

ISAKMP (Internet Security Association and Key Management Protocol)
A framework for key and SA management

ISO (International Organization for Standardization)
This organization is responsible for a wide range of standards (the OSI model, for example) and also for international relations with ANSI for X.509.

Key

This is a method of granting and refusing access, ownership rights or control rights. It is represented by any number of values.

Key exchange

A procedure that uses two or more nodes to transfer a secret session key via an insecure channel.

Key length

The number of bits used to represent the key size. The longer the key, the more secure it is.

Key management

A method of saving and distributing cryptographic keys securely. The entire process of securely creating and distributing cryptographic keys to authorized recipients.

MAC (Message Authentication Code)

A key-dependent one-way hash function that requires an identical key to verify the hash.

MD5 (Message Digest 5)

MD5 (Message Digest Algorithm 5) is a widespread cryptographic hash function that generates a 128-bit hash value. MD5 was developed by Ronald L. Rivest in 1991.

Mechanism

A process used to implement cryptographic operations

Message digest

A checksum that is calculated from a message. If you change just a single character in a message, the message will have a different message digest.

NIST (National Institute for Standards and Technology)

An institute of the U.S. Department of Commerce. Publishes standards on compatibility (FIPS).

NSA (National Security Agency)

An agency of the U.S. Department of Defense. Concerns itself primarily with information security tasks.

Plaintext or cleartext

Data or messages before encryption, in a format that can be easily read by humans – also known as unencrypted text

Public key

The publicly available component of an integrated asymmetric key pair, often referred to as the encryption key.

PKCS (Public Key Crypto Standards)

A range of de-facto standards for encryption using public keys, developed by an informal consortium (Apple, DEC, Lotus, Microsoft, MIT, RSA and Sun). It includes algorithm-specific and algorithm-independent implementation standards and specifications for the definition of message syntax and other protocols that are controlled by RSA Data Security, Inc.

Private key

The “secret” component of an integrated asymmetric key pair, the component that is in private possession. It is often referred to as the decryption key.

Pseudo-random number

A number that is calculated by applying algorithms, it creates random values that are derived from the computer environment (e.g. mouse coordinates). See “Random number”.

Random number

An important aspect for many encryption systems and a necessary element in the creation of unique keys that cannot be calculated by a potential hacker. Real random numbers are usually derived from analog sources and generally require the use of special hardware.

RFC (Request for Comment)

An IETF document from the subgroup FYI RFC, which provides an overview and introduction, or from the subgroup STD RFC, which specifies Internet standards. The abbreviation FYI stands for “for your information”. Each RFC has an RFC number that is used to identify it or search for it (www.ietf.org).

RSA

Short for RSA Data Security, Inc. The abbreviation RSA is derived from the names of the company founders, Ron Rivest, Adi Shamir and Len Adleman, who developed an algorithm of the same name. The RSA algorithm is used in cryptography with public keys. It is based on the fact that, while two large prime numbers are easy to multiply together, it is very difficult to reduce the product back to the original two numbers.

Secret key

The “session key” in symmetric algorithms

Session key

The secret (symmetric) key used to encrypt all data records on a transaction basis. A new session key is used for each communication session.

SET (Secure Electronic Transaction)

Used to transfer credit card details securely over the Internet.

SHA-1 (Secure Hash Algorithm)

The SHA (FIPS 180-1) developed by NIST was revised in 1994. SHA-1 was the result. SHA-1 generates a 160-bit hash.

SSL (Secure Socket Layer)

This was developed by Netscape to ensure the security and non-disclosure of sensitive information on the Internet. Supports server/client authentication and ensures the security and integrity of the transmission channel. This works at the transport layer and serves as a “socket library”, which permits an application-independent mode of operation. Encrypts the entire communication channel.

Stream cipher

A class of symmetric key encryption in which the conversion can be altered for each character of plaintext to be encrypted. It is recommended for environments in which there is limited storage capacity available for buffering data.

Symmetric algorithm

Also referred to as a conventional secret-key algorithm or single-key algorithm. The encryption key is either identical to the decryption key or one key can be derived from the other. There are two subcategories: block and stream.

TLS (Transport Layer Security)

An IETF draft. Version 1 is based on version 3.0 of the SSL protocol and is used to maintain privacy when communicating over the Internet.

Triple-DES, 3DES

An encryption configuration in which the DES algorithm is used three times with two or three different keys.

Unencrypted text

See “Plaintext or cleartext”.

Related publications

The manuals are available as online manuals, see *http://manuals.ts.fujitsu.com*, or in printed form which must be paid and ordered separately at *http://manualshop.ts.fujitsu.com*.

openNet Server V3.4 (BS2000/OSD)
BCAM V21.0A
User Guide

openCRYPT V1.2 (BS2000/OSD)
Security with Cryptography
User Guide

openNet Server (BS2000/OSD)
IPv6 Introduction and Conversion Guide, Stage 1
User Guide

SOCKETS(BS2000) V2.4
SOCKETS for BS2000/OSD
User Guide

openNet Server V2.0, interNet Services V2.0
SNMP Management for openNet Server and interNet Services
User Guide

interNet Services (BS2000/OSD)
Administrator Guide

interNet Services (BS2000/OSD)
User Guide

interNet Value Edition V1.0B (BS2000/OSD)
User Guide

IMON (BS2000/OSD)
Installation Monitor
User Guide

BS2000/OSD-BC
Introductory Guide to Systems Support
User Guide

SECOS (BS2000/OSD)
Security Control System
User Guide

SECOS (BS2000/OSD)
Security Control System
Ready Reference

RFCs

Comprehensive information on the Requests for Comments (RFCs) can be found on the home page of the Internet Engineering Task Force (IETF):

www.ietf.org

Index