



# **Nokia Mobile VPN Policy Specification**

**NOKIA**

**Nokia for Business**

# Table of Contents

Introduction.....	3
Purpose of this document .....	3
Scope .....	3
References.....	3
Documentation conventions.....	3
Abbreviations and definitions .....	3
Policy files .....	7
Policy information file format .....	9
Policy file format .....	11
General part .....	11
General part overview .....	11
General part grammar .....	11
IPSec part .....	12
IPSec part overview.....	12
IPSec part grammar .....	12
Policy section parameters .....	13
Keys section parameters .....	16
IKE part .....	17
IKE part overview .....	17
IKE section grammar .....	17
IKE parameters .....	19
Policy file examples.....	27
Certificate file format.....	30
Private key file format .....	31
VPN installation file.....	32

# Introduction

## Purpose of this document

This document provides a specification for S60 Mobile VPN policies: which files are needed and which data format is used in these files.

This document can be used as a reference, when generating new VPN policies manually or when you need to understand the details of an existing VPN policy.

## Scope

This document applies to S60 Mobile VPN client version 3.1.

This document specifies the syntax and semantics of each file used in a VPN policy definition and provides some clarifying examples. This document does not describe the creation of the policies or the VPN in general in any way.

## References

[1]	<i>Protocol Numbers</i>	<a href="http://www.iana.org/assignments/protocol-numbers">http://www.iana.org/assignments/protocol-numbers</a>
[2]	<i>ICMP Type Numbers</i>	<a href="http://www.iana.org/assignments/icmp-parameters">http://www.iana.org/assignments/icmp-parameters</a>
[3]	<i>RFC 4301 Security Architecture for IP</i>	<a href="http://www.ietf.org">http://www.ietf.org</a>
[4]	<i>RFC 3748 Extensible Authentication Protocol (EAP)</i>	<a href="http://www.ietf.org">http://www.ietf.org</a>
[5]	<i>PKCS #8: Private-Key Information Syntax Standard, RSA Lab</i>	<a href="http://www.rsa.com">http://www.rsa.com</a>
[6]	<i>PKCS #5: Password-Based Cryptography Standard, RSA Lab</i>	<a href="http://www.rsa.com">http://www.rsa.com</a>

## Documentation conventions

Formal grammar definitions are presented in BNF-like notation. All examples are presented with the following style:

```
Example style.
```

## Abbreviations and definitions

AES	Advanced Encryption Standard, known also as Rijndael; Symmetric cryptography chipper.
AH	Authentication Header: an operating mode of IPSec. May also mean IP packet extension header used by IPSec.
AKA	Authentication and Key Agreement; EAP mechanism for authentication and session key distribution.
ASN.1	Abstract Syntax Notation One (ASN.1)), a standard notation that describes data structures.
BNF	The Backus–Naur form: a notation used to express context-free grammars.
CA	Certificate Authority: an entity that issues digital certificates.
CBC	Chipper Block Chaining: a Block chipper operating mode.
CommDb	Communications database used in Symbian OS prior to 9.0. Contains networking related configuration information.
CommsDat	Communications database in Symbian OS 9.0 and beyond. Has replaced CommDb.
CRACK	Challenge/response authentication of cryptographic keys. An authentication extension

	to IKE version 1.
DES, 3DES	Data encryption standard. Symmetric cryptography chipper. 3DES is a variant of DES, which uses three time larger key than standard DES.
DER	Distinguished Encoding Rules: a data encoding method.
DLL	Dynamic Link Library.
DNS	Domain Name Service.
DPD	Dead Peer Detection.
DSS	Digital Signature Standard.
EAP	Extensible Authentication Protocol: a universal authentication framework.
EM	Event Mediator.
ESL	Enrolment Services List. VPN Extended Manager's way to store ACU enrollment services information into an XML file.
ESN	Extended Sequence Numbers.
ESP	Encapsulated Security Payload: an operating mode of IPSec. May also mean IP packet extension header used by IPSec.
ETL	Enrolment Tasks List. VPN Extended Manager's way to store pending enrollment tasks information into an XML file.
Event Mediator	A Symbian OS server that mediates events between its clients. Clients can both issue event listening requests (asynchronously) and report events (synchronously) to the Event Mediator. On receipt of an event report, the Event Mediator completes all event listening requests that are waiting for the event to occur.
FQDN	Fully Qualified Domain Name: differs from a regular domain name by its absoluteness; a suffix is not added.
GPRS	General Packet Radio Service.
HDR	The IKE header.
HTTP	Hyper Text Transfer Protocol.
IAP	Internet Access Point. A network node at the border of a mobile network and the Internet through which mobile devices can access the Internet.
IAP definition	A piece of information that defines how a certain IAP can be accessed (e.g. defines the access technology used to reach the IAP).
IAP ID	An identifier of an IAP definition in the CommDb. The identifier is local to the device where the IAP definition resides.
ICMP	Internet Control Message Protocol: one of the core protocols of the IP protocol suite.
IETF	Internet Engineering Task Force: a standard body.
IKE	Internet Key Exchange protocol.
IKEv2	The second version of the Internet Key Exchange protocol.
IP	Internet Protocol.
IPC	Inter Process Communication.
IPSec	IP Security Protocol.
ISAKMP	Internet Security Association and Key Management Protocol: a cryptographic protocol which forms the basis of the IKE key exchange protocol.
LAM	Legacy Authentication Method. Used to describe older authentication methods such as static username/password or token-based authentication (e.g. SecurID). Typically LAMs cannot be directly used by IKEv2, but they can be wrapped to (e.g.) EAP.
LAN	Local Area Network.
Library	A library is an executable that is loaded within an existing process, and can dynamically load processes.

Management server	A server that provides VPN software configuration and management support over the OMA DM protocol.
MD5	Message-Digest algorithm 5: a cryptographic hash function.
MODP	More Modular Exponential. Diffie-Hellman groups for Internet Key Exchange (IKE), RFC 4306 Appendix B, RFC 3526.
MsChapv2	A PEAP subtype.
Multihoming	A scenario in which a system can have several network interfaces active, each with its own IP address and each potentially on a different network.
Network ID	An identifier of a network in the CommsDat. The identifier is local to the device where the network definition resides.
NAT	Network Address Translation.
NAT-T	NAT traversal: Standard that describes how IPSec tunnels can be established through NAT gateways, without reconfiguration of the gateways.
NIF	Network interface: an attachment of a device to some network. An interface is uniquely identified by a network ID - IAP ID pair. Several interfaces can have the same network ID. This is the case when there are several ways to access the same network (e.g. WLAN and GPRS interfaces to the same network).
NifMan	Network Interface Manager: a software module that manages network interfaces.
NSSM	Nokia Security Service Manager: a policy server product.
OMA DM	Open Mobile Alliance Device Management: a set of standards for the management of mobile devices.
OS	Operating System.
PCL	Policy Content List: VPN Extended Manager's way of storing policy content information into an XML file.
PEAP	Protected Extensible Authentication Protocol: a method to securely transmit authentication information.
PFS	Perfect Forward Security: an IKE operating mode.
PKCS	Public Key Cryptographic Standards. Refers to a group of public key cryptography standards defined by RSA Security. Single standards are referred to as "PKCS#n" where n is a positive integer starting from 1.
PKCS#5	Password-based Encryption Standard (RFC 2898).
PKCS#10	Certification Request Standard (RFC 2986).
PKCS#12	Defines a file format commonly used to store private keys with related public key certificates, protected with a password-based symmetric key (Personal Information Exchange Syntax Standard", RSA Laboratories, June 24, 1999).
PKI	Public Key Infrastructure.
Plug-in	A separately installed piece of software that increases or enhances the features of a main software module. Can easily be added without changing the main software module.
Policy server	A server that provides content updating and certificate enrollment support for mobile devices over the ACU protocol. A server machine running the NSSM software is an example of a policy server.
Program	An executable that, once loaded, triggers the creation of a process.
ROM	Read-Only Memory.
RSA	An algorithm for public key encryption.
SA	Security Association: a set of parameters that define the properties of an active "connection" between IPSec or IKE peers.
SAD(B)	Security Association Database: a run-time data structure within the IPSec Protocol Module that contains all IPSec SAs definitions.

SDK	Software Developer's Kit.
Scope (level)	In a multihoming environment, each IP address has a scope (level) that can be determined from the address format. Examples of IPv6 address scope levels include node-local, site-local and global scope.
SecurID	A two-factor authentication mechanism. Based on something the user knows (a password or PIN) and something that the user has (an authenticator).
SHA-1	Secure Hash Algorithm: a cryptographic hash function.
SIM	Subscriber Identity Module: a removable smart card used for identification.
SIS	Symbian OS Installation System.
SIT	Socket Interaction Thread.
Socket Interaction Thread	A thread that is launched by the Event Mediator to fulfill event listening requests that require communication with the Socket Server (either directly or through some other servers).
SPD	Security Policy Database: a run-time data structure within the IPSec Protocol Module that defines which IP traffic is to be protected and how.
SPI	Security Parameter Index: an identification tag used by IPSec to identify which encryption rule and algorithm is used.
TCP	Transmission Control Protocol.
UDP	User Datagram Protocol, one of the core protocols of the IP protocol suite.
X.500	A series of computer networking standards.
X.509	ITU-T standard for Public Key Infrastructure. X.509 specifies standard formats for public key certificates.
XAUTH	Extended authentication. An authentication extension to IKE version 1.
VPN	Virtual Private Network.
VPN policy	A configuration that specifies the characteristics of a VPN connection to some protected network.
Zone ID	Within the scope level defined by an IP address, the address may have an associated ID. This ID is an additional piece of information that is used to disambiguate the routing of IP packets when the address alone is not sufficient. For example, in Symbian OS an IPv4 address can have scope level network. The zone ID associated with such an address is called the network ID. This ID is used to specify the exact network that the address belongs to. For example, in the case of several interfaces with overlapping 10.* address spaces, the network ID can be used to direct IP packets to their correct destination.

# Policy files

---

One VPN policy is made up of several files. Every valid VPN policy needs at least two different files: a policy file and a policy information file. Optional files are the user's private key file, user's certificate file, CA certificate file and peer certificate file.

The files belonging to the same policy are recognized by the beginning of the file name. The name format of each file type is defined in the table below.

File	Presence	Name format
Policy information file	Required	<code>&lt;policy_id&gt;.pin</code>
Policy file	Required	<code>&lt;policy_id&gt;.pol</code>
User's private key file	Optional	<code>&lt;policy_id&gt;-user.key</code>   <code>&lt;policy_id&gt;-user-&lt;number&gt;.key</code>
User's certificate	Optional	<code>&lt;policy_id&gt;-user.cer</code>   <code>&lt;policy_id&gt;-user-&lt;number&gt;.cer</code>
CA certificate	Optional	<code>&lt;policy_id&gt;-ca.cer</code>   <code>&lt;policy_id&gt;-ca-&lt;number&gt;.cer</code>
VPN installation file	Optional	<i>Bundle of zipped policy files for manual installation.</i>
P12 file	Optional	<i>PKCS#12 package file. Used only within VPN package.</i>
VPC	Optional	<i>Plain text file used only within .p12 package to configure included certificates and keys.</i>

The `<policy_id>` part of the file name can be chosen freely, but it must not contain the following reserved strings:

- `.pin.`
- `.pol.`
- `.key.`
- `.cer.`
- `-ca.`
- `.p12.`
- `.vpc.`

Whether the presence of the different PKI files (CER, .key, etc.) is required or not depends on the VPN authentication method(s) required by the VPN policy file (.pol). In some cases, only the policy and policy information files are needed.

Here is an example list of files that constitute a VPN policy. In this case, the policy comes with a single set of private key and certificate files and the policy ID is "test\_policy":

```
Policy information file: test_policy.pin
Policy file:           test_policy.pol
User's private key:    user.key
User's certificate:    user.cer
CA certificate:        ca.cer
```

Here is another example list of files that constitute a single policy. In this case, the policy contains two sets of private key and certificate files and the policy ID is "1234567890".

```
Policy information file: 1234567890.pin
Policy file:           1234567890.pol
User's private key (1): user-1.key
User's certificate (1): user-1.cer
CA certificate (1):     ca-1.cer
User's private key (2): user-2.key
```

User's certificate (2):	user-2.cer
CA certificate (2):	ca-2.cer

# Policy information file format

---

A policy information file (*\*.pin*) contains general information about the VPN policy. The information is used, for example, by the VPN setting UI to show policy details.

The policy information file is a simple text file, which must use 8-bit ISO 8859 character set (similar to the Latin 1 character set, or code page 1252, used in Windows environments).

The grammar of the policy information file is specified as follows:

```
pin_file ::= policy_name_section newline
           policy_version_section newline
           policy_description_section newline
           issuer_name_section newline
           contact_info_section

policy_name_section ::= "[POLICYNAME]" newline
                    policy_name

policy_version_section ::= "[POLICYVERSION]" newline
                       policy_version

policy_description_section ::= "[POLICYDESCRIPTION]" newline
                             policy_description

issuer_name_section ::= "[ISSUENAME]" newline
                     issuer_name

contact_info_section ::= "[CONTACTINFO]" newline
                      contact_info

policy_name ::= a string of characters from the allowed character set, max 64
              characters

policy_version ::= a string of characters from the allowed character set, max 16
                characters

policy_description ::= a string of characters from the allowed character set, max
                    256 characters

issuer_name ::= a string of characters from the allowed character set, max 64
             characters

contact_info ::= a string of characters from the allowed character set, max 64
              characters

new_line ::= '\n' | '\r' | "\r\n"
```

An example of the policy information file:

```
[POLICYNAME]
Nokia Intranet
[POLICYVERSION]
1.0
[POLICYDESCRIPTION]
Secure remote Intranet access
[ISSUENAME]
Nokia
```

[CONTACTINFO]  
vpnsupport@company.com, +358 9 1234 123

The [ISSUENAME] and [CONTACTINFO] fields are not shown to the user. If you want the issuer name and contact info to be visible to the users in the VPN policy details option, present the information in the [POLICYDESCRIPTION] field.

# Policy file format

A VPN policy file (*\*.pol*) is a text file that uses the 8-bit ISO 8859 character set (similar to the Latin 1 character set, or code page 1252, used in Windows environments).

The policy file is divided into three logical parts:

1. General part
2. IPSec part
3. IKE part

Each of the three parts constitutes from several sections.

Each section has its own syntax and content. The sections are separated from each other by section names enclosed in square brackets [ ].

The following generic rules apply to the syntax of all sections:

- Each section begins with a section separator (['']) at the beginning of a line.
- The configuration information within a certain section ends at the next section separator or at the end of the policy.
- Within section content, no line is allowed to begin with a start bracket (because that would terminate the section).
- The relative order of the sections in the policy is not important (any order can be used).
- Any characters following the # character to the end of the same line are treated as white space (this rule can be used to comment the specification).

## General part

### General part overview

The general part contains policy format version information, purely normative information about the policy, which is intended for the administrator of the policy.

The sections of the general part are described in more detail in the table below.

Section	Presence	Description
SECURITY_FILE_VERSION	Required	Specifies the version of the policy format used in the policy file. This field can be used to ensure compatibility with future policy and software versions. Currently used version is 1.
[INFO]	Optional	Optional field. Contains general information about the policy (e.g. using application and policy name).

### General part grammar

The grammar of the general part is:

```
general_part ::= "SECURITY_FILE_VERSION: 1" newline
               policy_info_section newline

policy_info_section ::= "[INFO]" newline
                    info newline

info ::= a string of characters from the allowed character set, max 1024 characters

newline ::= '\n' | '\r' | "\r\n"
```

## IPSec part

### IPSec part overview

The IPSec part of a VPN policy consists of the [POLICY] section and rarely needed [KEYS] section. The [KEYS] section is meant only for testing and evaluation purposes.

Section	Presence	Description
[POLICY]	Required	Specifies what type communications require the IPSec protection (selectors), and the kind of protection that is applied (encryption and authentication algorithms, requirements for the key lengths, etc.).
[KEYS]	Optional	Contains explicit security associations (SA) that are fully specified, including the session keys. This section is only needed when the service does not support automatic key management negotiation (ISAKMP/IKE), although both can co-exist.

### IPSec part grammar

The grammar of the IPSec is:

```
Ipsec_part ::= Policy_section newline
              Keys_section newline

Policy_section ::= "[POLICY]" newline
                (SecurityPolicy newline)*

SecurityPolicy ::= ("sa" Association | Selector)*

Association ::= AssociationName " = " "{" AssociationParameter* "}"

AssociationName ::= TokenString

TokenString ::= a sequence of characters not containing white space

AssociationParameter ::= "ah" |
                        "esp" |
                        "auth_alg" Integer |
                        "encrypt_alg" Integer |
                        "identity_local" TokenString |
                        "identity_remote" TokenString |
                        "pfs" |
                        "protocol_specific" |
                        "local_port_specific" |
                        "remote_port_specific" |
                        "proxy_specific" |
                        "src_specific" |
                        "replay_win_len" Integer |
                        "min_auth_bits" Integer |
                        "max_auth_bits" Integer |
                        "min_encrypt_bits" Integer |
                        "max_encrypt_bits" Integer |
                        "hard_lifetime_allocations" Integer |
                        "hard_lifetime_bytes" Integer |
                        "hard_lifetime_addtime" Integer |
                        "hard_lifetime_usetime" Integer |
                        "soft_lifetime_allocations" Integer |
                        "soft_lifetime_bytes" Integer |
                        "soft_lifetime_addtime" Integer |
                        "soft_lifetime_usetime" Integer

Selector ::= SelectorItem " = " ( "{" AssociationReference* "}" | "drop" )

SelectorItem ::= "inbound" |
                "outbound" |
                "remote" AddressMaskPair |
                "local" AddressMaskPair |
```

```

        "protocol" Integer |
        "local_port" Integer |
        "remote_port" Integer |
        "icmp_type" Integer |
        "icmp_code" Integer |
        "if" InterfaceName

AddressMaskPair ::= Ip4AddressMaskPair | Ip6AddressMaskPair

Ip4AddressMaskPair ::= Ip4Address Ip4Address

Ip6AddressMaskPair ::= Ip6Address Ip6Address

AssociationReference ::= [ "?" ]AssociationName "(" Tunnel ")"

Tunnel ::= Address | Domain name | .

InterfaceName ::= TokenString

Keys_section ::= "[KEYS]" newline
                (Key_line newline)*

Key_line ::= "pfkey_add" Integer Integer Integer Integer Integer Integer Integer
Address
            Address Integer Integer Integer Key Key

Key ::= "0" | Hexadecimal_Integer

new_line ::= '\n' | '\r' | "\r\n"

```

## Policy section parameters

Parameter	Description
esp ah	Indicates the type of the SA. One and only one of these parameters must be present in each AssociationParameter set.
encrypt_alg (Integer)	<p>Defines the encryption algorithm that is used during an SA. Only one instance of this parameter can appear on the list:</p> <ul style="list-style-type: none"> <li>• 2 DES</li> <li>• 3 3DES</li> <li>• 11 NULL encryption</li> <li>• 12 AES</li> </ul> <p>The encrypt_alg parameter is required for ESP, but it is not accepted for AH (the integer values are defined in reference document [3]).</p>
auth_alg (Integer)	<p>Defines the authentication algorithm that is used during an SA. Only one instance of this parameter can appear on the list:</p> <ul style="list-style-type: none"> <li>• 2 MD5</li> <li>• 3 SHA-1</li> </ul> <p>The auth_alg parameter is required for AH, but it is optional for ESP (the integer values are defined in reference document [3]).</p>
min_auth_bits max_auth_bits min_encrypt_bits max_encrypt_bits	<p>Specify limits for the acceptable key lengths to be used in the selected algorithm if the key length of the algorithm can be adjusted. For example, AES.</p> <p>Do not specify these parameters if you use an algorithm with a fixed key length, such as DES, 3DES, MD5, and SHA-1.</p> <p>To use AES, set the value of encrypt_alg to 12 and the value of max_encrypt_bits to 128, 192, or 256, depending on the AES variant that the VPN gateway supports.</p>

<p>hard_lifetime_bytes hard_lifetime_addtime hard_lifetime_usetime</p>	<p>Specifies the hard lifetimes of the IPsec SA. After the hard life time expires, the original IPsec SA is deleted. New IPsec SA is created automatically next time, when traffic is sent through the tunnel.</p> <p>The parameter hard_lifetime_bytes specifies the life time in bytes. This is not supported yet and now should be set to value 0.</p> <p>The parameter hard_lifetime_addtime specifies the life time of the SA counted from the creation of the SA.</p> <p>The parameter hard_lifetime_usetime specifies the life time of the SA counted from the moment SA was first used to transmit data.</p> <p>When a hard lifetime expires, the SA is unconditionally deleted. Use soft lifetimes to restart rekeying negotiations before the old SA is really deleted and thus avoid unnecessary delays in communication. Soft lifetimes do not have any effect if they expire after a hard lifetime expires.</p>
<p>soft_lifetime_bytes soft_lifetime_addtime soft_lifetime_usetime</p>	<p>Specifies the soft lifetimes for the IPsec SA. After the soft life time expires the IPsec SA is rekeyed, but the SA is not deleted.</p> <p>The parameter soft_lifetime_bytes specifies the life time in bytes. This is not supported yet and now should be set to value 0.</p> <p>The parameter soft_lifetime_addtime specifies the life time of the SA counted from the creation of the SA.</p> <p>The parameter soft_lifetime_usetime specifies the life time of the SA counted from the moment SA was first used to transmit data.</p> <p>When a hard lifetime expires, the SA is unconditionally deleted. Use soft lifetimes to restart rekeying negotiations before the old SA is really deleted and thus avoid unnecessary delays in communication. Soft lifetimes do not have any effect if they expire after a hard lifetime expires.</p>
<p>pfs</p>	<p>Perfect Forward Secrecy. This parameter is passed to the key management process (IKE) only.</p> <p>If you specify this parameter, IKE initiates a new Diffie-Hellman exchange to obtain new master key keying material for each new session key that IPsec SAs require. Thus, each quick mode negotiation includes a Diffie-Hellman exchange. If you do not specify pfs, the IPsec SA session keys are based on keying material that is created in IKE Phase 1 negotiations.</p>
<p>identity_local (TokenString)</p>	<p>Passed to the key-management process as a source identity in the ACQUIRE message (for an outbound SA). When the same policy specification is applied to an inbound SA, identity_local must match the destination identity.</p> <p>In IKE, identity_local is the source identity of the Phase 2 proposal, when this host is the initiator of the negotiations. If identity_local is not present, the IKE default is used.</p> <p>In typical IPsec VPN configurations, the IKE default is the IP address of the VPN client. As the IP address is practically always dynamic, it must not be configured statically here.</p>
<p>identity_remote (TokenString)</p>	<p>Passed to the key-management process as destination identity in the ACQUIRE message. When the same specification is applied to an inbound SA, identity_remote must match the source identity.</p> <p>In IKE, identity_remote is the destination identity of the Phase 2 proposal when this host is the initiator of the negotiations. If identity_remote is not present, the IKE default is used.</p> <p>However, in IPsec VPN configurations, you must specify this value. In association definitions that are associated with address selectors, the value is the same as the corresponding selector value (for example, a subnet definition). In association definitions that are associated with plain port or protocol selectors, the value should indicate the full IP address space (zero address and mask).</p> <p>Currently, IKEv2 does not support other identity_remote values than 0.0.0.0/0. This also reflects to remote selectors ( "all hosts" 0.0.0.0 selector must be used).</p>
<p>replay_win_len (Integer)</p>	<p>If non-zero, enables an antireplay service for the IPsec SA. The maximum supported window length value is 32.</p>

local_port_specific	The SAs that are created using this template contain connection information (protocol, source, and destination ports) from the packets that they were negotiated to protect. The SA applies only to packets that contain the defined connection information.
remote_port_specific	
protocol_specific	
proxy_specific	Reserved for future use.
src_specific	<p>The SAs created by using this template contain the source address from the packets that they were negotiated to protect. The SA applies only to packets that contain the defined source address. Usually, src_specific must always be set.</p> <p>With multicast destinations this parameter has a special meaning: If the src_specific parameter is not set (it is missing), the incoming multicast messages are accepted from any source address. With this configuration it is possible to define one SA for incoming multicast packets coming from multiple sources.</p>

Table 1 IPsec Selection parameters

Parameter	Description
inbound	Specifies that the selector applies to all incoming (received) IP packets that match the rest of the selector parameters. A selector definition that contains the inbound selector parameter can contain all selector parameter types.
outbound	Specifies that the selector applies to all outgoing (sent) IP packets that match the rest of the selector parameters. A selector definition that contains the outbound selector parameter can contain all selector parameter types.
local (Address Mask)	<p>Specifies the source and destination of the connection:</p> <ul style="list-style-type: none"> <li>• <b>remote</b>—the address and mask pair defines the remote end of the connection.</li> <li>• <b>local</b>—the address and mask pair defines the local end of the connection.</li> </ul> <p>The local and remote parameters always require both IP address and the network mask.</p> <p>The IP address and the address from the IP packet are separately combined with the mask by using bitwise AND. If the results are equal, this selector item matches. To match a single specific IP address, specify an all-one-mask (e.g. for IPv6 FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF).</p> <p>Only one instance of local and remote can appear in a selector. You can specify both local and remote selectors, but you cannot specify two remote addresses for a single selector. If more than one destination is needed, use multiple selectors.</p> <p>You can include a scope ID in the selector IP address to limit the scope of the selector. Use the percent (%) character to separate the scope ID from the IP address. For example in IPv4:</p> <pre>remote 10.253.55.0%2 255.255.255.0 = { test_55(10.253.16.20) }</pre> <p>The interpretation of the scope ID depends on the corresponding IP address. For private addresses (10.*), the scope ID denotes a network ID that is defined in the network configuration in the mobile device.</p> <p>You can accommodate selectors with protocol and remote_port (or local_port) parameters if needed. For example in IPV4:</p> <pre>remote 10.253.55.0%2 255.255.255.0 protocol 6 remote_port 80 = { test_55(10.253.16.20) }</pre> <p>Tunnel endpoint can be either an IP address (as per above example, "10.253.16.20") or alternatively a valid domain name (e.g. "sgw.company.com") that can be looked up by performing a standard DNS query.</p>
remote (Address Mask)	

<code>protocol (Integer)</code>	<p>The selector matches if the protocol field of the packet matches this integer. For example, the following selector matches all TCP packets (inbound and outbound):</p> <pre>protocol 6 = { ... }</pre> <p>Only one instance of the protocol parameter can appear in a selector. The <i>Internet Assigned Numbers Authority</i> (IANA) specifies the protocol numbers. For more information, see reference document [1].</p>
<code>remote_port (Integer)</code> <code>local_port (Integer)</code>	<p>The port information is known to be present only in UDP (17) and TCP (6) protocols. This selector matches when the port field of the protocol matches the integer.</p> <p>For example, the following selector implicitly matches TCP or UDP packets whose destination port is set to 80 (which is the default HTTP port):</p> <pre>remote_port 80 = { ... }</pre> <p>The preceding example does not specify the direction (inbound or outbound) of the packet. You can use the following two selectors, instead:</p> <pre>outbound remote_port 80 = { ... } inbound remote_port 80 = { ... }</pre> <p>Only one instance of <code>local_port</code> and <code>remote_port</code> can be defined in a selector.</p>
<code>icmp_type (Integer)</code> <code>icmp_code (Integer)</code>	<p>This selector item matches only if the IP packet is an <i>Internet control message protocol</i> (ICMP) packet with matching type and/or code. Only one instance of <code>icmp_type</code> and <code>icmp_code</code> can appear in a selector.</p> <p>IANA specifies the ICMP types and codes. For more information, see reference document [2].</p>
<code>if (InterfaceName, i.e. TokenString)</code>	<p>This selector item matches only if the IP packet is associated with the specified interface.</p>

### Keys section parameters

Each key (or actually SA) is defined on a single line of text that starts with the keyword `pfkey_add` followed by parameters separated with spaces. The position of a parameter defines its meaning. There are no optional parameters. Note also, that not all SA parameters can be set in this way (specifically, the lifetimes cannot be fully controlled).

*Table 2 Keys section parameters*

Field 1	Type: AH = 1, ESP = 2.
Field 2	SPI value as an integer.
Field 3	Encryption algorithm (use 0, if the parameter does not apply, for example with AH).
Field 4	Authentication algorithm (use 0, if the parameter does not apply).
Field 5	SA direction: inbound=4, outbound=8 (must be one of these).
Field 6	Hard lifetime in bytes (use 0 for not enabled).
Field 7	Hard lifetime in seconds (use 0 for not enabled).
Field 8	Source IP address using the dotted IP address notation for IPv4 (e.g., 123.123.123.123) or 16-bit hex numbers separated by ':' for IPv6 (e.g. 2001:618:400:6a::abc).
Field 9	Destinations IP address using the dotted IP address notation for IPv4 or 16-bit hex numbers separated by ':' for IPv6.
Field 10	Protocol (use 0, if not defined).
Field 11	Source port (use 0, if not defined).
Field 12	Destination port (use 0, if not defined).
Field 13	Authentication key as a hexadecimal string (use 0, if not defined).
Field 14	Encryption key as a hexadecimal string (use 0, if not defined).

An example [KEYS] is presented below:

```
[KEYS]
pfkey_add 2 88001 2 0 8 0 0 0.0.0.0 194.100.123.131 0 0 0
blb2b3b4b5b6b7b8b9b0c1c2c3c4c5c6c7c8c9c0
c1c2c3c4c5c6c7c8c9cacbccdcecf0d1d2d3d4d5d6d7d8
pfkey_add 1 1 0 2 4 0 0 0.0.0.0 ::1 0 0 0 1234567890 0
pfkey_add 2 88001 2 2 4 0 0 194.100.123.131 0.0.0.0 0 0 0
blb2b3b4b5b6b7b8b9ba1c2c3c4c5c6c7c8c9cb
c1c2c3c4c5c6c7c8c9c0cbcccdcecf0d1d2d3d4d5d6d7d8
pfkey_add 2 1365 11 1 8 0 0 0.0.0.0 129.6.51.42 0 0 0
0123456789abcdef0123456789abcdef 0123456789abcdef0123456789abcdef
pfkey_add 2 819 11 1 4 0 0 129.6.51.42 0.0.0.0 0 0 0
0123456789abcdef0123456789abcdef 0123456789abcdef0123456789abcdef
```

## IKE part

### IKE part overview

An IKE policy consists of one or more [IKE] sections. The VPN software supports both IKE version 1 and 2. When it comes to the IKE policy format, most of the IKE policy parameters are common to both IKE versions. However, some parameters have been added to the policy format to support some special IKE version 2 features and certain existing parameters are interpreted in a "suitable" manner in the case of IKE version 2.

For backward compatibility reasons, all existing policies are interpreted as IKE version 1 policies. Policies are interpreted as IKE version 2 policies only if they explicitly specify their version as 2 with the new IKE\_VERSION parameter.

### IKE section grammar

The IKE section syntax can be described by the following BNF-like notation:

```
IKE_part ::= "[IKE]" newline
           IKE_section

IKE_section ::= "IKE_VERSION: " Integer new_line &
               "ADDR: " Address new_line new_line &
               "MODE: " Mode new_line &
               "SEND_NOTIFICATION: " Bool new_line &
               "ID_TYPE: " Integer new_line &
               "FQDN: " String new_line
               "REMOTE_ID_TYPE: " Integer new_line &
               "REMOTE_IDENTITY: " String new_line &
               "SKIP_REMOTE_ID_CHECK: " Bool new_line &
               "GROUP_DESCRIPTION_II: " new_line
               "USE_COMMIT: " Bool new_line &
               "INITIAL_CONTACT: " Bool new_line &
               "REPLAY_STATUS: " Bool new_line &
               "SEND_CERT: " Bool new_line &
               "RESPONDER_LIFETIME: " Bool new_line &
               "IPSEC_EXPIRE: " Bool new_line &
               "CRACK_LAM_TYPE: " Crack_lam_type_attr new_line &
               "CRACK_LAM_USERNAME: " String new_line &
               "CRACK_LAM_PASSWORD: " String new_line &
               "USE_INTERNAL_ADDR: " Bool new_line &
               "USE_NAT_PROBE: " Bool new_line &
               "ESP_UDP_PORT: " Integer new_line &
               "USE_XAUTH: " Bool new_line &
               "USE_MODE_CFG: " Bool new_line &
               "DPD_HEARTBEAT: " Integer new_line &
               "NAT_KEEPALIVE: " Integer new_line &
               "REKEYING_THRESHOLD: " Integer new_line &
               "EAP_PROTOCOL: " Integer new_line &
               "EAP_REALM_PREFIX: " String new_line &
               "EAP_MANUAL_REALM: " String new_line &
               "EAP_MANUAL_USERNAME: " String new_line &
               "EAP_HIDE_IDENTITY: " Bool new_line &
               "PROPOSALS: " Integer Proposal_attr new_line &
               "PRESHARED_KEYS: " Integer Presharedkey_attr new_line &
               "CAS: " Integer Ca_cert_attr new_line &
```

```

        "OWN_CERT_TYPE: " Own_cert_type new_line &
        "OWN_CERTS: " ((Cert_attr Key_attr) | Own_cert_attr) & new_line
Address ::= Ip4Address | Ip6Address | FQDN
Mode ::= "Main" | "Aggressive"
Bool ::= "TRUE" | "FALSE"
Crack_lam_type_attr ::= "PASSWORD"
Own_cert_type ::= "USER" | "DEVICE"
Proposal_attr ::= "ENC_ALG:" ("DES-CBC" | "3DES-CBC" | "AES128-CBC" | "AES192-CBC"
| "AES256-CBC") &
        "AUTH_METHOD:" Auth_method_type &
        "HASH_ALG: " ("MD5" | "SHA1") &
        "GROUP_DESCRIPTION: " ("MODP_768" | "MODP_1024" | "MODP_1536") &
        "LIFETIME_KBYTES:" Integer &
        "LIFETIME_SECONDS: " Integer &
        "PRF: " ("3DES-CBC-MAC" | "NONE")
Auth_method_type ::= "RSA_SIGNATURES" |
        "DSS_SIGNATURES" |
        "RSA_ENCRYPT" |
        "RSA_REV_ENCRYPT" |
        "PRE-SHARED" |
        "IKE-CRACK"
Presharedkey_attr ::= "FORMAT: " ("HEX_FORMAT" | "STRING_FORMAT")
        "KEY: " (<key length> <space> (<key in hexadecimal format> |
<key in string format>))
Ca_cert_attr ::= "FORMAT: " Ca_cert_format_type &
        "DATA: " Ca_cert_data
Cert_attr ::= "FORMAT: " Format_type &
        "DATA: " Cert_data
Own_cert_attr ::= "SUBJECT_DN_SUFFIX: " <full or partial subject distinguished
name> &
        (" | "IDENTITY_AS_RFC822NAME: " ("1" | "0")) &
        (" | "RFC822NAME_FQDN: " <RFC822 name>)
Key_attr ::= "PRIVATE_KEY_FORMAT: " Pkey_type &
        "PRIVATE_KEY_DATA: Cert_data
Ca_cert_format_type ::= "BIN" | "NAME" | "KEYID" | "APPLUID"
Ca_cert_data ::= <filename> | <subject distinguished name> |
        <subject key identifier as hexadecimal string> |
Applicability_uid_list
Applicability_uid_list ::= Uid | Uid "," Applicability_uid_list
Uid ::= <UID as hexadecimal string>
Format_type ::= "BIN"
Cert_data ::= <filename>
Key_data ::= <filename>
Pkey_type ::= "BIN"
new_line ::= '\n' | '\r' | "\r\n"

```

## IKE parameters

The supported IKE parameters are described in the table below. It is noted if a parameter is specific for IKE version or could be dropped out from the policy file.

Parameter	Description
IKE_VERSION	Specifies the version of the IKE protocol to be used. Possible values: <ul style="list-style-type: none"> <li>• 2 (IKE protocol version 2 is used).</li> <li>• Any other (IKE protocol version 1 is used).</li> </ul> If the parameter is missing, IKE protocol version 1 is used.
ADDR	IP address or FQDN and network mask of the IKE host (typically VPN gateway) that the client should communicate with. In tunnel mode IPSec, the address is the address of the tunnel end-point that the corresponding IPSec selector refers to with its association reference parameter and the mask is a full address mask. In transport-mode IPSec, the address and mask are the same as the address and mask of the corresponding IPSec selector.
DNS_SERVER	DNS server IP address: used in association with the VPN tunnel. If the secure gateway provides a DNS server IP address, it overrides this value.
MODE	IKE mode to use in the communication. Possible values: <ul style="list-style-type: none"> <li>• MAIN.</li> <li>• AGGRESSIVE (This choice is only for IKEv1).</li> </ul>
SEND_NOTIFICATION	Specifies whether the client sends notification messages when errors occur. Possible values: <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul> The value TRUE is preferred, as it may make troubleshooting easier.
ID_TYPE	Determines how the client identifies itself to the VPN gateway during IKE phase 1 negotiations. RFC 2407 defines the possible ID types as: <ul style="list-style-type: none"> <li>• 0 = Reserved.</li> <li>• 1 = Single IPv4 address.</li> <li>• 2 = Fully qualified domain name string.</li> <li>• 3 = Fully qualified user name string.</li> <li>• 5 = Single IPv6 address.</li> <li>• 9 = Binary DER encoding of an ASN.1 X.500 Distinguished Name of the owner of the certificates that are being exchanged to establish the SA.</li> <li>• 11 = Opaque byte stream that might be used to pass vendor-specific information necessary to identify which preshared key should be used to authenticate Aggressive mode negotiations.</li> </ul> When using certificate-based VPN client authentication, the value should be set to 9. When communicating with Nokia IP VPN gateways and using the CRACK authentication method, this value should be set to the special value 129. When communicating with CheckPoint gateways and using the CRACK authentication method, this value must be set to 3. When communicating with Cisco VPN 3000 Series Concentrator gateways, this value should be set to 11. In addition, the value of the FQDN field must be set appropriately (see below). If no IKE ID type is specified, the IP address of VPN Client is used as the ID.
FQDN	Specifies an opaque string that is used to pass the identity of initiator (ID <sub>i</sub> ) in IKE negotiations. If this optional parameter is not used, the identity of initiator is derived from own cert's subject alternative name extension field. In addition, this parameter is used when communicating with Cisco VPN 3000 Series Concentrator gateways. There, this parameter specifies the name of a user group in

	the VPN gateway. In addition, the password of the user group must be specified as the value of the PRESHARED_KEYS parameter (see below) and the ID_TYPE parameter must be set to value 11 (see above).
REMOTE_ID_TYPE	Specifies the type of the value specified for the REMOTE_IDENTITY parameter (see below). This parameter can be used together with IKE version 2 to support an additional remote host identity check from the remote host's device certificate's subjectAltName extension. Possible values: <ul style="list-style-type: none"> <li>• 1 (IPv4 ADDR).</li> <li>• 2 (FQDN).</li> <li>• 3 (RFC822_NAME).</li> <li>• 5 (IPv6 ADDR).</li> </ul>
REMOTE_IDENTITY	Specifies the expected identity of the remote host. This parameter can be used together with IKE version 2 to support an additional remote host identity check from the remote host's device certificate's subjectAltName extension. The type this parameter's value is specified in the REMOTE_ID_TYPE parameter (see above).
SKIP_REMOTE_ID_CHECK	If set to TRUE, remote identity (id,) of SGW not verified against the subjectAltName in the remote host's device certificate. Default value is FALSE. This parameter can be used together only with IKE version 2.
GROUP_DESCRIPTION_ID	The Oakley group to be used during perfect forward secrecy (PFS) Quick Mode negotiations. Possible values: <ul style="list-style-type: none"> <li>• MODP_768.</li> <li>• MODP_1024.</li> <li>• MODP_1536.</li> </ul> <b>Note:</b> This parameter is used for different purposes than parameter GROUP_DESCRIPTION.
USE_COMMIT	Specifies whether or not to use the Quick Mode Commit Bit to synchronize key exchange, protect against loss of transmissions over unreliable networks, and guard against the need for multiple retransmissions. Possible values: <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul> This parameter applies only to the case where the client acts as the responder in IKE negotiations. As this is rarely the case, USE_COMMIT is usually set to FALSE.
INITIAL_CONTACT	Specifies whether the client should send a status message to inform a remote host that this is the first SA to establish with the remote host. The recipient of this notification message might delete any existing SAs it has for the sender under the assumption that the sender has rebooted and no longer has access to the original SAs and the associated keying material. Possible values: <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul> It is advisable to set this parameter to TRUE.
REPLAY_STATUS	Specifies whether the client should send a message to the remote host to confirm the remote host's choice as to whether it is going to perform anti-replay protection or not. This is used only with IKEv1. Possible values: <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul> The status value that the responder sends to the initiator depends on the value of the "replay_win_len" IPsec policy parameter. This parameter is typically set to FALSE.
RESPONDER_LIFETIME	Specifies whether the client should send a status message to the remote host to communicate the IPsec SA lifetime that the client has chosen. This parameter applies only to cases where the client is acting as the responder in IKE negotiations. Possible values: <ul style="list-style-type: none"> <li>• TRUE</li> </ul>

	<ul style="list-style-type: none"> <li>• FALSE</li> </ul> <p>This parameter can be used together only with IKE version 1 and it is advisable to set this parameter to TRUE.</p>
SEND_CERT	<p>Specifies whether the client should transmit its client certificate to the remote host even if the remote host does not send a certificate-signing request. Possible values:</p> <ul style="list-style-type: none"> <li>• TRUE</li> <li>• FALSE</li> </ul> <p>This parameter can be used together only with IKE version 1.</p>
IPSEC_EXPIRE	<p>Specifies whether IPsec SAs expire when the IKE SA that was used to negotiate them expires or is deleted. Possible values:</p> <ul style="list-style-type: none"> <li>• TRUE (IPsec SAs expire when the IKE SA that was used to negotiate them expires or is deleted)</li> <li>• FALSE (IPsec SAs expire according to their lifetime)</li> </ul> <p>This parameter can be used together only with IKE version 1.</p>
CRACK_LAM_TYPE	<p>Specifies the type of the CRACK (Challenge/Response Authentication of Cryptographic Keys) authentication method. Possible value:</p> <ul style="list-style-type: none"> <li>• PASSWORD.</li> </ul>
CRACK_LAM_USERNAME	<p>Specifies username for silent CRACK authentication.</p>
CRACK_LAM_PASSWORD	<p>Specifies password for silent CRACK authentication.</p>
USE_INTERNAL_ADDR	<p>Specifies whether to use the Nokia VPN internal addressing feature or not. Internal addressing is a technique where the VPN gateway grants to the client a corporate LAN internal address and access to DNS services. With internal addressing, mobile users are virtually part of the corporate network. Possible values:</p> <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul> <p>In IKEv2 the internal addressing uses standard IKEv2 configuration payloads.</p>
USE_NAT_PROBE	<p>Specifies whether to use the Nokia-specific IPsec over NAT or the IETF NAT-T NAT traversal mechanism. With the NAT traversal feature, the existence of public network NAT services can be automatically detected and IP packets can be encapsulated in UDP packets to bypass the NAT service. Possible values:</p> <ul style="list-style-type: none"> <li>• TRUE (use Nokia's IPsec over NAT mechanism).</li> <li>• FALSE (use IETF NAT-T).</li> </ul> <p><b>Note:</b> In addition to this field, also the ESP_UDP_PORT MUST be set appropriately (see below).</p> <p>To enable NAT traversal in IKE version 2, this parameter must be set to FALSE.</p>
ESP_UDP_PORT	<p>Specifies the UDP destination port that is used for IPsec ESP UDP encapsulation when NAT traversal is in use.</p> <p>With Nokia's IPsec over NAT mechanism (see above), if this parameter is missing or set to 0, the default port 9872 is used.</p> <p>With IETF NAT-T (see above), this parameter MUST be set to 0.</p> <p>With IKE version 2 NAT traversal, this parameter should be set to 0.</p>
USE_XAUTH	<p>Specifies whether to use the extended authentication protocol (XAUTH) within IKE (specified in &lt;draft-beaulieu-ike-xauth-02.txt&gt;). Possible values:</p> <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul>
USE_MODE_CFG	<p>Specifies whether to use the IKE configuration method (MODE-CFG) within IKEv1 (specified in &lt;draft-dukes-ike-mode-cfg-01.txt&gt;). Possible values:</p> <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul>

	This parameter is not used in IKEv2.
DPD_HEARTBEAT	Specifies how often the VPN client sends the R_U_THERE notification to the VPN gateway. The value is specified as an integer, in seconds. The R_U_THERE notification is a part of the Dead Peer Detection (DPD) protocol. If the parameter is missing or set to 0, the DPD protocol is not used. This parameter applies also to DPD in IKE version 2.
NAT_KEEPAALIVE	<p>Specifies the NAT keepalive timeout for both IETF and Nokia NAT traversal mechanisms. The value is specified as an integer, in seconds.</p> <p>The parameter takes effect only after the VPN client is found to reside behind NAT during the NAT traversal negotiations. If this is not the case, the client does not send any keepalive messages. This applies to both Nokia and IETF NAT traversal mechanisms.</p> <p>With IETF NAT-T in IKE version 1 and IKE version 2 NAT traversal, this parameter specifies how often the client sends an empty UDP packet to port 4500 at the VPN gateway. If the parameter is missing or set to 0, the client uses a default value of 120 seconds.</p> <p>With IKE version 2, the parameter specifies how often the client sends an empty UDP packet to port 4500 at the VPN gateway. If the parameter is missing or set to 0, the client uses a default value of 120 seconds.</p> <p>With Nokia's IPsec over NAT, the parameter specifies how often the client sends an empty UDP packet to port 500 at the VPN gateway. Also with Nokia's IPsec over NAT, the VPN client also sends a dummy ICMP Echo request to the port that the ESP_UDP_PORT parameter defines or, if this value is not set, to the 9872 default port. Finally, in the case of Nokia IPsec over NAT, if the NAT keepalive parameter is missing or set to 0, the client does not send any keepalive packets to the gateway.</p>
REKEYING_THRESHOLD	<p>Specifies if and when IKE SA rekeying is used. If the parameter has a valid value, the client starts IKE SA rekeying when the specified percentage of the IKE SA expiration timeout is reached. Thus, the new IKE SA negotiation is started before the existing IKE SA expires. The percentage value is specified as an integer with accepted values ranging from 70 to 95. If the value is lower than 70, the threshold value 70 is used and if the value is larger than 95, the threshold value 95 is used. If the parameter is missing or set to 0, IKE SA rekeying is not used.</p> <p>The IKA SA rekeying feature is needed with some VPN gateways (for example, Cisco VPN 3000 Series Concentrator) to prevent the VPN internal IP address from changing and thus causing transport level connections to be lost. With these gateways, when using the challenge-response authentication method and the Nokia internal-addressing feature, if the IKE SA is not rekeyed but allowed to expire before a new SA is negotiated, the internal IP address value received from the gateway within a new IKE SA negotiation differs from the internal IP address value received within the expired IKE SA negotiation. When this address change happens, the client's transport level connections get lost.</p> <p><b>Note:</b> This parameter applies also to the IKE SA rekeying feature defined in IKE version 2. The values of the parameter are as described above also in the case of IKE version 2.</p>
EAP_PROTOCOL	<p>VPN legacy authentication methods used only with IKE version 2. This is based on the use of the Extended Authentication Protocol (EAP). This parameter specifies whether EAP is in use, and if it is, which EAP protocol (or type) is used. If this parameter is missing, EAP is not in use. Possible supported values are:</p> <ul style="list-style-type: none"> <li>• 18 (SIM).</li> <li>• 23 (AKA).</li> </ul> <p>The following values are unsupported:</p> <ul style="list-style-type: none"> <li>• 4 (MD5-challenge).</li> <li>• 5 (One Time Password).</li> <li>• 6 (Generic Token Card).</li> <li>• 13 (Transport Layer Security).</li> <li>• 17 (Cisco's LEAP).</li> <li>• 21 (Tunneled TLS).</li> </ul>

	<ul style="list-style-type: none"> <li>• 25 (PEAP).</li> <li>• 26 (MsChapv2).</li> <li>• 32 (SecurID).</li> </ul> <p>(The integer values are defined in reference document [4]).</p> <p><b>Note:</b> The exact EAP types that can be used in real-world VPN environments depend on the VPN gateway configuration.</p>
EAP_REALM_PREFIX	If EAP is used, specifies a realm prefix string. Used only with EAP_PROTOCOL parameter and IKEv2.
EAP_MANUAL_REALM	If EAP is used, specifies a manual realm string. Used only with EAP_PROTOCOL parameter and IKEv2.
EAP_MANUAL_USERNAME	If EAP is used, specifies a manual user name string. Used only with EAP_PROTOCOL parameter and IKEv2.
EAP_HIDE_IDENTITY	If EAP is used, specifies whether identity hiding is used or not. Possible values: <ul style="list-style-type: none"> <li>• TRUE.</li> <li>• FALSE.</li> </ul> <p>This parameter is currently not used.</p>
PROPOSALS	Begins a “proposal section” and specifies the number of proposal definitions in the section. Each proposal definition consists of a set of parameters that are used when initiating an IKE negotiation. Parameters in this section MUST be in order presented below.
ENC_ALG (proposal parameter)	Specifies the encryption algorithm to be used in IKE communication. Possible values: <ul style="list-style-type: none"> <li>• DES-CBC.</li> <li>• 3DES-CBC.</li> <li>• AES128-CBC.</li> <li>• AES192-CBC.</li> <li>• AES256-CBC.</li> </ul> <p>The encryption algorithms are defined in RFCs 2407 and 3602.</p>
AUTH_METHOD (proposal parameter)	Specifies the VPN authentication method. Possible values: <ul style="list-style-type: none"> <li>• RSA_SIGNATURES.</li> <li>• DSS_SIGNATURES.</li> <li>• PRE-SHARED.</li> <li>• IKE-CRACK.</li> </ul> <p><b>Note:</b> This parameter is ignored and could be discarded, if EAP authentication is used in IKEv2.</p>
HASH_ALG (proposal parameter)	Specifies the hash algorithm to compute initialization vectors (IVs) or digests in IKE communication. Possible values: <ul style="list-style-type: none"> <li>• SHA1.</li> <li>• MD5.</li> </ul> <p>The hash algorithms are defined in RFC 2407.</p>
GROUP_DESCRIPTION (proposal parameter)	Specifies the Oakley group to be used during the Diffie-Hellman (DH) exchange in IKE negotiations. Possible values: <ul style="list-style-type: none"> <li>• MODP_768.</li> <li>• MODP_1024.</li> <li>• MODP_1536.</li> </ul> <p><b>Note:</b> This parameter is used for different purposes than parameter GROUP_DESCRIPTION_II.</p>
GROUP_TYPE	Specifies the Oakley group type presented above. Possible values:

(proposal parameter)	<ul style="list-style-type: none"> <li>• DEFAULT.</li> <li>• MODP.</li> </ul> <p>Normally value DEFAULT should be used.</p>
LIFETIME_SECONDS (proposal parameter)	Specifies the lifetime of the IKE SA in terms of the duration of the SA.
PRF (proposal parameter)	<p>Specifies the pseudo-random function negotiated in IKE policy. Possible values are:</p> <ul style="list-style-type: none"> <li>• 3DES-CBC-MAC.</li> <li>• NONE.</li> </ul> <p>If parameter value is NONE then HASH_ALG value is used.</p>
PRESHARED_KEYS	<p>If pre-shared keys are used for authentication, this parameter specifies the key in use, its format, and key length. The parameter has the following sub-fields:</p> <ul style="list-style-type: none"> <li>• FORMAT Either HEX_FORMAT or STRING_FORMAT.</li> <li>• KEY number of characters in the key and key value separated by space.</li> </ul> <p>For example:</p> <pre>PRESHARED_KEYS: FORMAT: STRING_FORMAT KEY: 8 password</pre> <p>Or for hex format keys:</p> <pre>PRESHARED_KEYS: FORMAT: HEX_FORMAT KEY: 8 abcd1234</pre> <p>The maximum number of characters allowed for the key is 256.</p>
CAS	<p>Specifies the number of CA certificate identifiers that specify the CA certificates that the client trusts, followed by the identifiers of these certificates. Each certificate identifier is specified as a pair of subfields FORMAT and DATA. A single certificate identifier can identify one or more certificates depending on its format and data values.</p> <p>An example with two certificates:</p> <pre>CAs: 2 FORMAT: BIN DATA: test_policy-ca-1.cer FORMAT: BIN DATA: test_policy-ca-2.cer</pre> <p>The possible values for the FORMAT and DATA subfields are described in Table 3.</p>
OWN_CERT_TYPE	<p>Specifies store location for client certificate. Possible values:</p> <ul style="list-style-type: none"> <li>• USER.</li> <li>• DEVICE.</li> </ul> <p>If silent certificate authentication is needed, the value MUST be set to DEVICE. The default value is set to value USER.</p>
OWN_CERTS	<p>Specifies a reference to a user certificate and private key that are used for user or client authentication during IKE negotiations. The previously presented parameter OWN_CERT_TYPE specifies a location, where this certificate is stored.</p> <p>The parameter has the following subfields:</p> <ul style="list-style-type: none"> <li>• FORMAT (must be set to BIN).</li> <li>• DATA (certificate file name, without path).</li> <li>• PRIVATE_KEY_FORMAT (must be set to BIN).</li> <li>• PRIVATE_KEY_DATA (private key file name, without path).</li> <li>• SUBJECT_DN_SUFFIX (The subject DN suffix to be used in the user certificates used for accessing this VPN gateway. See Table 4 for the supported attribute types.</li> </ul> <p>For example:</p>

- OU=NIC, O=Company, L=Santa Cruz, ST=California, C=US. This could be a fully or partially constructed name without wild characters. As an example, partial name could be like: 'O=SomeO,C=SomeC'. This would match with user certificate with subject DN: 'CN=Some User, O=SomeO, C=SomeC'.
- IDENTITY\_AS\_RFC822NAME (Integer, a flag that indicates whether the user certificates used for accessing this VPN gateway should include the user identity as an rfc822Name in the certificates' subject alternative name extension field.  
Possible values:  
0 = no  
1 = yes
- RFC822NAME\_FQDN (The fully qualified domain name (FQDN) to be used in the rfc822Name value if the user identity is included as an rfc822Name in the certificates' subject alternative name extension field. The field can define the whole rfc822Name or only the domain part of the name.).
- PRIVATE\_KEY\_LENGTH (Integer, The expected length of the private key whose corresponding public key is included in the user certificate used with this VPN gateway).

**Example 1:**

When policy is imported using .vpn file and the certificates and the private key are included in the .vpn bundle as a separate files the policy has to define the names for included certificate and private key file:

```
OWN_CERTS:
FORMAT: BIN
DATA: test_policy-user.cer
PRIVATE_KEY_FORMAT: BIN
PRIVATE_KEY_DATA: test_policy-user.key
```

**Example 2:**

When the policy is imported using .vpn file and the certificates and the private key are included inside a PKCS#12 file the user certificate and the private key has to be defined in policy as follows. (It is important to use file names user-1.cer and user-1.key.):

```
OWN_CERTS:
FORMAT: BIN
DATA: user-1.cer
PRIVATE_KEY_FORMAT: BIN
PRIVATE_KEY_DATA: user-1.key
```

**Example 3:**

If the policy is delivered using OMA DM the user certificate can be defined using either certificate SubjectName or RFC822 name defined in certificate SubjectAltName. If the SubjectName is used the user certificate has to be defined as follows:

```
OWN_CERTS:
PRIVATE_KEY_LENGTH: 2048
SUBJECT_DN_SUFFIX: OU=SomeOU, O=SomeO, C=SomeC
```

**Example 4:**

If the policy is delivered using OMA DM the user certificate can be defined using either certificate SubjectName or RFC822 name defined in certificate SubjectAltName. If the RFC822 is used the user certificate has to be defined as follows:

```
OWN_CERTS:
```

	<pre>PRIVATE_KEY_LENGTH: 1024 IDENTITY_AS_RFC822NAME: 1 RFC822NAME_FQDN: some.com</pre> <p>The IKE protocol implementation uses these parameters, if they are present, when it searches the S60 cert store for the client certificate to use when authenticating to a certain VPN gateway. For example:</p> <p>If some of the fields in <i>SUBJECT_DN_SUFFIX</i> parameter contain a dot character, the value of the field has to be enclosed with double quotation marks. Example:</p> <pre>SUBJECT_DN_SUFFIX: OU=SomeOU, O=SomeO, C="SomeC1, SomeC2"</pre>
--	--

Table 3 Possible values for CAs parameter

Format	Data
<p><b>BIN</b></p> <p>This identifier identifies a single certificate.</p> <p><b>Note:</b> This format is used when VPN policy is imported using .vpn file.</p>	<p>Certificate file name (without path).</p> <p>Example 1:</p> <pre>CAs: 1 FORMAT: BIN DATA: test_policy-ca.cer</pre> <p>Example 2:</p> <p>If CA certificates are inside a PKCS#12 file, which is packaged inside the .vpn file the CA certificates have to be referenced as follows:</p> <pre>CAs: 2 FORMAT: BIN DATA: ca-1.cer FORMAT: BIN DATA: ca-2.cer</pre>
<p><b>NAME</b></p> <p>This identifier identifies one or more certificates.</p>	<p>Certificate subject (distinguished) name. See Table for the supported attribute types.</p> <p>Example:</p> <pre>CAs: 1 FORMAT: NAME DATA: cn=name, ou=unit, o=org</pre> <p>If some of the fields in the <i>DATA</i> parameter contain a dot character the value of the field must be enclosed with double quotation marks.</p> <p>Example:</p> <pre>DATA: cn="name1,name2", ou=unit, o=org</pre>
<p><b>KEYID</b></p> <p>This identifier identifies a single certificate.</p>	<p>A hexadecimal string representation of a 160-bit SHA-1 hash of the value of the BIT STRING <i>subjectPublicKey</i> (excluding the tag, length, and number of unused bits) field of the CA certificate.</p> <p>Example:</p> <pre>CAs: 1 FORMAT: KEYID DATA: 406aec085279ba6e16022d9e0629c0229687dd48.</pre>
<p><b>APPLUID</b></p> <p>This identifier identifies one or more certificates.</p>	<p>One or more certificate applicability/application UIDs, separated by commas, represented as hexadecimal strings.</p> <p>Example 1:</p> <pre>CAs: 1 FORMAT: APPLUID DATA: 101F7993</pre> <p>Example 2:</p>

CAs: 1  
FORMAT: APPLUID  
DATA: 100042AB,1000183D

The latter example would match all certificates that have both of the listed applicability UIDs set.

Table 4 Supported attribute types in issuer and subject names

Attribute type	Short name	OID
commonName	CN	2.5.4.3
surname	SN	2.5.4.4
serialNumber	-	2.5.4.5
countryName	C	2.5.4.6
localityName	L	2.5.4.7
stateOrProvinceName	ST	2.5.4.8
organizationName	O	2.5.4.10
organizationalUnitName	OU	2.5.4.11
title	-	2.5.4.12
givenName	-	2.5.4.42
initials	-	2.5.4.43
generationQualifier	-	2.5.4.44
dnQualifier	-	2.5.4.46
domainComponent	DC	0.9.2342.19200300.100.1.25

## Policy file examples

The following is an example of a simple VPN policy using IKEv1. It is presented here just to illustrate the above policy content overview. The policy is not guaranteed to be fully accurate in all details.

```
SECURITY_FILE_VERSION: 3
[INFO]
Intranet VPN access
[POLICY]
sa ipsec_generic*_1 = {
  esp
  encrypt_alg 3
  auth_alg 3
  identity_remote 10.113.1.0/24
  src_specific
  hard_lifetime_bytes 0
  hard_lifetime_addtime 3600
  hard_lifetime_usetime 3600
  soft_lifetime_bytes 0
  soft_lifetime_addtime 3600
  soft_lifetime_usetime 3600
}
sa ipsec_generic*_2 = {
  esp
  encrypt_alg 3
  auth_alg 3
  identity_remote 10.113.4.0/24
```

```

src_specific
hard_lifetime_bytes 0
hard_lifetime_addtime 3600
hard_lifetime_usetime 3600
soft_lifetime_bytes 0
soft_lifetime_addtime 3600
soft_lifetime_usetime 3600
}

remote 10.113.1.0 255.255.255.0 = { ipsec_generic*_1(217.152.33.82) }
remote 10.113.4.0 255.255.255.0 = { ipsec_generic*_2(217.152.33.82) }
inbound = { }
outbound = { }

[IKE]
ADDR: 217.152.33.82 255.255.255.255
MODE: Main
SEND_NOTIFICATION: TRUE
ID_TYPE: 3
USE_COMMIT: FALSE
INITIAL_CONTACT: FALSE
REPLAY_STATUS: FALSE
CRACK_LAM_TYPE: PASSWORD
USE_INTERNAL_ADDR: TRUE
USE_NAT_PROBE: TRUE
NAT_KEEPALIVE: 60
USE_XAUTH: FALSE
USE_MODE_CFG: FALSE
REKEYING_THRESHOLD: 0
PROPOSALS: 1
ENC_ALG: 3DES-CBC
AUTH_METHOD: IKE-CRACK
HASH_ALG: SHA1
GROUP_DESCRIPTION: MODP_1024
LIFETIME_KBYTES: 0
LIFETIME_SECONDS: 7200
PRF: NONE
CAs: 1
  FORMAT: BIN
  DATA: test_policy-ca.cer

```

The following is an example of a simple VPN policy using IKEv2 with EAP authentication. It is presented here just to illustrate the above policy content overview. The policy is not guaranteed to be fully accurate in all details.

```

SECURITY_FILE_VERSION: 1
[INFO]
IPVPNEAP
[POLICY]

sa ipsec_generic*_1 = {
  esp
  encrypt_alg 12
  auth_alg 3
  max_encrypt_bits 128
  identity_remote 0.0.0.0/0
  src_specific
  hard_lifetime_bytes 0
  hard_lifetime_addtime 3600
  hard_lifetime_usetime 3600
  soft_lifetime_bytes 0
  soft_lifetime_addtime 3600
  soft_lifetime_usetime 3600
}

remote 0.0.0.0 0.0.0.0 = { ipsec_generic*_1 (10.20.70.111) }

```

```
inbound = {}
outbound = {}

[KEYS]
[IKE]
IKE_VERSION: 2
ADDR: 10.20.70.111 255.255.255.255
MODE: Main
SEND_NOTIFICATION: TRUE
GROUP_DESCRIPTION_II: MODP_1024
ID_TYPE: 3
REMOTE_IDENTITY: APN1
FQDN: APN1
USE_COMMIT: FALSE
SEND_CERT: FALSE
INITIAL_CONTACT: TRUE
REPLAY_STATUS: TRUE
USE_XAUTH: FALSE
USE_MODE_CFG: TRUE
USE_INTERNAL_ADDR: TRUE
USE_NAT_PROBE: FALSE
CRACK_LAM_TYPE: PASSWORD
ESP_UDP_PORT: 0
EAP_PROTOCOL: 18
EAP_HIDE_IDENTITY: FALSE
PROPOSALS: 1
ENC_ALG: AES128-CBC
HASH_ALG: SHA1
GROUP_DESCRIPTION: MODP_1024
GROUP_TYPE: DEFAULT
LIFETIME_KBYTES: 0
LIFETIME_SECONDS: 86400
PRF: NONE
CAs: 1
  FORMAT: NAME
  DATA: CN=CompanyVPNCA,O=VPN,C=COM
```

# Certificate file format

Certificate files must be in the DER-encoded X509.3 ASN.1 format.

# Private key file format

The private key files must contain the key in the PKCS#8 `EncryptedPrivateKeyInfo` format. Details are presented in reference document [\[5\]](#). The private key information must be encrypted according to the PKCS#5 v2.0 standard (see reference document [\[6\]](#)). The only supported algorithm for encryption operation (PBES2) is 3DES (des-ede3-cbc). The key generation function (PBKDF2) is default PRF `hmacWithSHA1`.

# VPN installation file

You can make ZIP file from policy files of your configuration and replace ZIP extension with VPN. Then you can install that policy by sending VPN file to a phone and running it.

If you are using a PKCS #12 package file using P12 extension for installation of certificates and associated keys, you can include that package in the VPN installation file for automated certificate and key installation. The VPN installation file may include only one P12 file. The VPN file may also include one and only one configuration file (VPC) using XML syntax, where you can set a parameter *lockPolicy* for your device lock policy and password for unzipping PKCS #12 package without any user invention. The parameter *lockPolicy* may contain any values between 0 and 3, where value 0 is the current policy and value 1 is weakest policy (and most unsecure) while value 3 is the most restrictive. However if current restrictions are more restrictive than the new ones, old restrictions are remaining in force.

The values are declared as presented in table below.

*Table 5 Possible values for CAs parameter*

Value	Description
0	Installing this policy has no impact on the current device lock settings.
1	No special requirements on lock code. <ul style="list-style-type: none"><li>• Autolock timeout 30 minutes.</li></ul>
2	<ul style="list-style-type: none"><li>• Lock code minimum length 5 characters.</li><li>• Autolock timeout 10 minutes.</li><li>• Wipe after 10 unsuccessful unlocking attempts.</li></ul>
3	<ul style="list-style-type: none"><li>• Lock code minimum length 5 characters.</li><li>• Both characters and numbers required.</li><li>• Both upper and lower case letters required.</li><li>• Passcode expires every 30 days and must be changed.</li><li>• New passcode must not match any of the 5 previous passcodes.</li><li>• Autolock timeout 5 minutes.</li><li>• Wipe after 10 unsuccessful unlocking attempts.</li></ul>

It is noticeable that you have to write names of certificates and associated keys to policy file (POL) as presented in chapter 2.

The PKCS #12 and VPC files are identified by their extensions; thus there is no need for any naming issues for those files. The VPC file format example is presented below:

```
<vpncommands version="3.1">
  <versionInfo>
    <version>1.0</version>
    <description>Cmd file, devlock loose policy, p12 pwd</description>
    <created>[vvvv-mm-dd]T[hh:mm:ss]</created>
  </versionInfo>
  <device>
    <devicelock>
      <lockPolicy>0</lockPolicy>
    </devicelock>
  </device>
  <pkcs12>
    <p12pwd>password</p12pwd>
  </pkcs12>
</vpncommands>
```

None of the sections in this file are mandatory, and there can be at most one instance of each.

## Legal Notice

© Nokia 2008. Reproduction, transfer, distribution or storage of part or all of the contents in this document in any form without the prior written permission of Nokia is prohibited.

Nokia and Nokia Connecting People are registered trademarks of Nokia Corporation. Other product and company names mentioned herein may be trademarks or tradenames of their respective owners.

Nokia operates a policy of continuous development. Nokia reserves the right to make changes and improvements to any of the products described in this document without prior notice.

THE CONTENTS OF THIS DOCUMENT ARE PROVIDED "AS IS". EXCEPT AS REQUIRED BY APPLICABLE LAW, NO WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE MADE IN RELATION TO THE ACCURACY, RELIABILITY OR CONTENTS OF THIS DOCUMENT. UNDER NO CIRCUMSTANCES SHALL NOKIA BE RESPONSIBLE FOR ANY LOSS OF DATA OR INCOME OR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES HOWSOEVER CAUSED.

Nokia reserves the right to revise this document or withdraw it at any time without prior notice.

## Work together. Smarter.

**Nokia Inc.** 102 Corporate Park Drive, White Plains, NY 10604 USA

**Americas** Tel: 1 877 997 9199 • Email: [usa@nokiaforbusiness.com](mailto:usa@nokiaforbusiness.com)

**Asia Pacific** Tel: +65 6588 33 64 • Email: [asia@nokiaforbusiness.com](mailto:asia@nokiaforbusiness.com)

**Europe** France +33 170 708 166 • UK +44 161 601 8908 • Email: [europe@nokiaforbusiness.com](mailto:europe@nokiaforbusiness.com)

**Middle East and Africa** Dubai +971 4 3697600 • Email: [mea@nokiaforbusiness.com](mailto:mea@nokiaforbusiness.com)

[www.nokiaforbusiness.com](http://www.nokiaforbusiness.com)

© 2008 Nokia. All rights reserved. Nokia and Nokia Connecting People are registered trademarks of Nokia Corporation. Other trademarks mentioned are the property of their respective owners. Nokia operates a policy of continuous development, therefore, reserves the right to make changes and improvements to any of the products described in this document without prior notice.

# NOKIA

**Nokia for Business**