NAME

setkey - manually manipulate the IPsec SA/SP database

SYNOPSIS

setkey [**-v**] **-c**
setkey [**-v**] **-f** *filename*
setkey [**-v**] **-e** *script*
setkey [**-Pgltv**] **-D**
setkey [**-Pv**] **-F**
setkey [**-h**] **-x**

DESCRIPTION

The **setkey** utility adds, updates, dumps, or flushes Security Association Database (SAD) entries as well as Security Policy Database (SPD) entries in the kernel.

The **setkey** utility takes a series of operations from the standard input (if invoked with **-c**), from the file named *filename* (if invoked with **-f** *filename*), or from the command line argument following the option (if invoked with **-e** *script*).

-D Dump the SAD entries. If with **-P**, the SPD entries are dumped.

-F Flush the SAD entries. If with **-P**, the SPD entries are flushed.

-g Only SPD entries with global scope are dumped with **-D** and **-P** flags.

-t Only SPD entries with ifnet scope are dumped with **-D** and **-P** flags. Such SPD entries are linked to the corresponding if_ipsec(4) virtual tunneling interface.

-h Add hexadecimal dump on **-x** mode.

-l Loop forever with short output on **-D**.

-v Be verbose. The program will dump messages exchanged on PF_KEY socket, including messages sent from other processes to the kernel.

-x Loop forever and dump all the messages transmitted to PF_KEY socket. **-xx** makes each timestamp unformatted.

Configuration syntax

With **-c** or **-f** on the command line, **setkey** accepts the following configuration syntax. Lines starting

with hash signs ('#') are treated as comment lines.

add [**-46n**] *src dst protocol spi* [*extensions*] *algorithm ...* ;
: Add an SAD entry. add can fail with multiple reasons, including when the key length does not match the specified algorithm.

get [**-46n**] *src dst protocol spi* ;
: Show an SAD entry.

delete [**-46n**] *src dst protocol spi* ;
: Remove an SAD entry.

deleteall [**-46n**] *src dst protocol* ;
: Remove all SAD entries that match the specification.

flush [*protocol*] ;
: Clear all SAD entries matched by the options. **-F** on the command line achieves the same functionality.

dump [*protocol*] ;
: Dumps all SAD entries matched by the options. **-D** on the command line achieves the same functionality.

spdadd [**-46n**] *src_range dst_range upperspec policy* ;
: Add an SPD entry.

spddelete [**-46n**] *src_range dst_range upperspec* **-P** *direction* ;
: Delete an SPD entry.

spdflush ;
: Clear all SPD entries. **-FP** on the command line achieves the same functionality.

spddump ;
: Dumps all SPD entries. **-DP** on the command line achieves the same functionality.

Meta-arguments are as follows:

src
dst Source/destination of the secure communication is specified as IPv4/v6 address. The **setkey** utility can resolve a FQDN into numeric addresses. If the FQDN resolves into multiple

addresses, **setkey** will install multiple SAD/SPD entries into the kernel by trying all possible combinations. **-4**, **-6** and **-n** restricts the address resolution of FQDN in certain ways. **-4** and **-6** restrict results into IPv4/v6 addresses only, respectively. **-n** avoids FQDN resolution and requires addresses to be numeric addresses.

protocol

protocol is one of following:

esp	ESP based on rfc2406
esp-old	ESP based on rfc1827
ah	AH based on rfc2402
ah-old	AH based on rfc1826
ipcomp	IPComp
tcp	TCP-MD5 based on rfc2385

spi Security Parameter Index (SPI) for the SAD and the SPD. *spi* must be a decimal number, or a hexadecimal number with '0x' prefix. SPI values between 0 and 255 are reserved for future use by IANA and they cannot be used.

extensions

take some of the following:

-m *mode* Specify a security protocol mode for use. *mode* is one of following: transport, tunnel or any. The default value is any.

-r *size* Specify the bitmap size in octets of the anti-replay window. *size* is a 32-bit unsigned integer, and its value is one eighth of the anti-replay window size in packets. If *size* is zero or not specified, an anti-replay check does not take place.

-u *id* Specify the identifier of the policy entry in SPD. See *policy*.

-f *pad_option*

defines the content of the ESP padding. *pad_option* is one of following:

zero-pad	All of the padding are zero.
random-pad	A series of randomized values are set.
seq-pad	A series of sequential increasing numbers started from 1 are set.

-f nocyclic-seq

Do not allow cyclic sequence number.

-lh *time*

-ls *time* Specify hard/soft life time duration of the SA.

-natt *oai [sport] oar [dport]*
Manually configure NAT-T for the SA, by specifying initiator *oai* and requestor *oar* ip addresses and ports. Note that the '[' and ']' symbols are part of the syntax for the ports specification, not indication of the optional components.

-natt_mtu *fragsize*
Configure NAT-T fragment size.

algorithm

-E *ealgo key*
Specify an encryption or Authenticated Encryption with Associated Data (AEAD) algorithm *ealgo* for ESP.

-E *ealgo key* **-A** *aalgo key*
Specify a encryption algorithm *ealgo*, as well as a payload authentication algorithm *aalgo*, for ESP.

-A *aalgo key*
Specify an authentication algorithm for AH.

-C *calgo* [**-R**]
Specify a compression algorithm for IPComp. If **-R** is specified, the *spi* field value will be used as the IPComp CPI (compression parameter index) on wire as is. If **-R** is not specified, the kernel will use well-known CPI on wire, and *spi* field will be used only as an index for kernel internal usage.

key must be double-quoted character string, or a series of hexadecimal digits preceded by '0x'.

Possible values for *ealgo*, *aalgo* and *calgo* are specified in separate section.

src_range
dst_range
These are selections of the secure communication specified as IPv4/v6 address or IPv4/v6 address range, and it may accompany TCP/UDP port specification. This takes the following form:

address
address/prefixlen
address[port]
address/prefixlen[port]

prefixlen and *port* must be a decimal number. The square brackets around *port* are necessary and are not manpage metacharacters. For FQDN resolution, the rules applicable to *src* and *dst* apply here as well.

upperspec

The upper layer protocol to be used. You can use one of the words in */etc/protocols* as *upperspec*, as well as icmp6, ip4, or any. The word any stands for "any protocol". The protocol number may also be used to specify the *upperspec*. A type and code related to ICMPv6 may also be specified as an *upperspec*. The type is specified first, followed by a comma and then the relevant code. The specification must be placed after icmp6. The kernel considers a zero to be a wildcard but cannot distinguish between a wildcard and an ICMPv6 type which is zero. The following example shows a policy where IPSec is not required for inbound Neighbor Solicitations:

spdadd ::/0 ::/0 icmp6 135,0 -P in none;

NOTE: *upperspec* does not work in the forwarding case at this moment, as it requires extra reassembly at forwarding node, which is not implemented at this moment. Although there are many protocols in */etc/protocols*, protocols other than TCP, UDP and ICMP may not be suitable to use with IPsec.

policy *policy* is expressed in one of the following three formats:

-P *direction* discard
-P *direction* none
-P *direction* ipsec *protocol/mode/src-dst/level* [...]

The direction of a policy must be specified as one of: out or in. The direction is followed by one of the following policy levels: discard, none, or ipsec. The discard policylevel means that packets matching the supplied indices will be discarded while none means that IPsec operations will not take place on the packet and ipsec means that IPsec operation will take place onto the packet. The *protocol/mode/src-dst/level* statement gives the rule for how to process the packet. The *protocol* is specified as ah, esp or ipcomp. The *mode* is either transport or tunnel. If *mode* is tunnel, you must specify the end-point addresses of the SA as *src* and *dst* with a dash, '-', between the addresses. If *mode* is transport, both *src* and *dst* can be omitted. The *level* is one of the following: default, use, require or unique. If the SA is not available in every level, the kernel will request the SA from the key exchange daemon. A value of default tells the kernel to use the system wide default protocol e.g., the one from the esp_trans_deflev sysctl variable, when the kernel processes the packet. A value of use means that the kernel will use an SA if it is available, otherwise the kernel will pass the packet as it would normally. A value of require

means that an SA is required whenever the kernel sends a packet matched that matches the policy. The unique level is the same as require but, in addition, it allows the policy to bind with the unique out-bound SA. For example, if you specify the policy level unique, racoon(8) (*ports/security/ipsec-tools*) will configure the SA for the policy. If you configure the SA by manual keying for that policy, you can put the decimal number as the policy identifier after unique separated by colon ':' as in the following example: unique:number. In order to bind this policy to the SA, number must be between 1 and 32767, which corresponds to *extensions* **-u** of manual SA configuration.

When you want to use an SA bundle, you can define multiple rules. For example, if an IP header was followed by an AH header followed by an ESP header followed by an upper layer protocol header, the rule would be:

esp/transport//require ah/transport//require;

The rule order is very important.

Note that "discard" and "none" are not in the syntax described in ipsec_set_policy(3). There are small, but important, differences in the syntax. See ipsec_set_policy(3) for details.

ALGORITHMS

The following lists show the supported algorithms.

Authentication Algorithms

The following authentication algorithms can be used as *aalgo* in the **-A** *aalgo* of the *protocol* parameter:

algorithm	keylen (bits)	comment
hmac-sha1	160	ah/esp: rfc2404
	160	ah-old/esp-old: 128bit ICV (no document)
null	0 to 2048	for debugging
hmac-sha2-256	256	ah/esp: 128bit ICV (RFC4868)
	256	ah-old/esp-old: 128bit ICV (no document)
hmac-sha2-384	384	ah/esp: 192bit ICV (RFC4868)
	384	ah-old/esp-old: 128bit ICV (no document)
hmac-sha2-512	512	ah/esp: 256bit ICV (RFC4868)
	512	ah-old/esp-old: 128bit ICV (no document)
aes-xcbc-mac	128	ah/esp: 96bit ICV (RFC3566)
	128	ah-old/esp-old: 128bit ICV (no document)
tcp-md5	8 to 640	tcp: rfc2385
chacha20-poly1305	256	ah/esp: 128bit ICV (RFC7634)

Encryption Algorithms

The following encryption algorithms can be used as the *ealgo* in the **-E** *ealgo* of the *protocol* parameter:

algorithm	keylen (bits)	comment
null	0 to 2048	rfc2410
aes-cbc	128/192/256	rfc3602
aes-ctr	160/224/288	rfc3686
aes-gcm-16	160/224/288	AEAD; rfc4106
chacha20-poly1305	256	rfc7634

Note that the first 128/192/256 bits of a key for aes-ctr or aes-gcm-16 will be used as the AES key, and the remaining 32 bits will be used as the nonce.

AEAD encryption algorithms such as aes-gcm-16 include authentication and should not be paired with a separate authentication algorithm via **-A**.

Compression Algorithms

The following compression algorithms can be used as the *calgo* in the **-C** *calgo* of the *protocol* parameter:

algorithm	comment
deflate	rfc2394

EXIT STATUS

The **setkey** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

Add an ESP SA between two IPv6 addresses using the AES-GCM AEAD algorithm.

```
add 3ffe:501:4819::1 3ffe:501:481d::1 esp 123457
        -E aes-gcm-16 0x3ffe050148193ffe050148193ffe050148193ffe ;
```

Add an authentication SA between two FQDN specified hosts:

```
add -6 myhost.example.com yourhost.example.com ah 123456
        -A hmac-sha2-256 "AH SA configuration!" ;
```

Get the SA information associated with first example above:

```
get 3ffe:501:4819::1 3ffe:501:481d::1 ah 123456 ;
```

Flush all entries from the database:

```
flush ;
```

Dump the ESP entries from the database:

```
dump esp ;
```

Add a security policy between two networks that uses ESP in tunnel mode:

```
spdadd 10.0.11.41/32[21] 10.0.11.33/32[any] any
        -P out ipsec esp/tunnel/192.168.0.1-192.168.1.2/require ;
```

Use TCP MD5 between two numerically specified hosts:

```
add 10.1.10.34 10.1.10.36 tcp 0x1000 -A tcp-md5 "TCP-MD5 BGP secret" ;
add 10.1.10.36 10.1.10.34 tcp 0x1001 -A tcp-md5 "TCP-MD5 BGP secret" ;
```

SEE ALSO

ipsec_set_policy(3), if_ipsec(4), racoon(8) (*ports/security/ipsec-tools*), sysctl(8)

Changed manual key configuration for IPsec, https://www.kame.net/newsletter/19991007/, October 1999.

HISTORY

The **setkey** utility first appeared in WIDE Hydrangea IPv6 protocol stack kit. The utility was completely re-designed in June 1998. It first appeared in FreeBSD 4.0.

BUGS

The **setkey** utility should report and handle syntax errors better.

For IPsec gateway configuration, *src_range* and *dst_range* with TCP/UDP port number do not work, as the gateway does not reassemble packets (cannot inspect upper-layer headers).