Technical Report: RAKE payload format

Damien Leroy, Gregory Detal

November 12, 2010

1 RAKE payloads

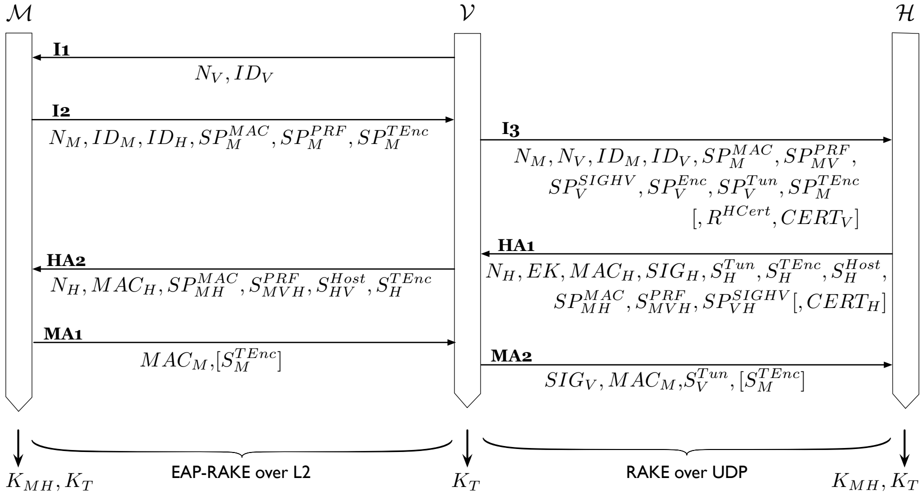

$\mathcal{M}$
$\mathcal{V}$
$\mathcal{H}$
I1
N_V, ID_V
I2
$N_M, ID_M, ID_H, SP_M^{MAC}, SP_M^{PRF}, SP_M^{TEnc}$
I3
$N_M, N_V, ID_M, ID_V, SP_M^{MAC}, SP_{MV}^{PRF},$
$SP_V^{SIGHV}, SP_V^{Enc}, SP_V^{Tun}, SP_M^{TEnc}$
$[, R^{HCert}, CERT_V]$
HA1
$N_H, EK, MAC_H, SIG_H, S_H^{Tun}, S_H^{TEnc}, S_H^{Host},$
$SP_{MH}^{MAC}, S_{MVH}^{PRF}, SP_{VH}^{SIGHV}[, CERT_H]$
HA2
$N_H, MAC_H, SP_{MH}^{MAC}, S_{MVH}^{PRF}, S_{HV}^{Host}, S_H^{TEnc}$
MA1
$MAC_M, [S_M^{TEnc}]$
MA2
$SIG_V, MAC_M, S_V^{Tun}, [S_M^{TEnc}]$
K_{MH}, K_T
EAP-RAKE over L2
K_T
RAKE over UDP
K_{MH}, K_T

Figure 1: Full RAKE protocol including all optional payloads

Figure 1 depicts the full RAKE protocol, i.e., including all settings and practical aspects. Each RAKE message is encoded as a linked list of payloads. Each payload carries one piece of information with some parameters, e.g. the format of this data.

Security Proposal payloads have been added in Figure 1. These are used to negotiate which algorithm should be used, mainly for cryptographic primitives. The first entity sends its own security proposal containing all the algorithms it supports for a specific use, e.g., for the PRF function, for MAC computation, ... The receiver computes the intersection between this list and its own list. If the third party is implied, the updated security proposal is sent to this entity which performs the same operation.

The following describes each message and the payloads it contains.

1.1 I1 message (from $\mathcal{V}$ to $\mathcal{M}$)

N_V A nonce, randomly chosen by $\mathcal{V}$. The nonce size must be 32 bytes.

ID_V The identity of $\mathcal{V}$. The ID can either be an IP address, a fully-qualified name or an ASCII string.

1.2 I2 message (from $\mathcal{M}$ to $\mathcal{V}$)

N_M A nonce, chosen by $\mathcal{M}$. The nonce size must be 32 bytes.

ID_M The identity of $\mathcal{M}$. The ID can either be an IP address, a fully-qualified name or an ASCII string.

ID_H The identity of $\mathcal{H}$. The ID can either be an IP address, a fully-qualified name or an ASCII string.

SP_M^{MAC} The list of algorithms supported by $\mathcal{M}$ for the `MAC` function.

SP_M^{PRF} The list of algorithms supported by $\mathcal{M}$ for the `PRF` function.

SP_M^{TEnc} The list of algorithms supported by $\mathcal{M}$ for the tunnel encryption, including *none*.

1.3 I3 message (from $\mathcal{V}$ to $\mathcal{H}$)

N_M The nonce chosen by $\mathcal{M}$.

N_V The nonce chosen by $\mathcal{V}$.

ID_M The identity of $\mathcal{M}$.

ID_V The identity of $\mathcal{V}$.

SP_M^{MAC} The list of algorithms supported by $\mathcal{M}$ for the `MAC` function.

SP_{MV}^{PRF} The list of algorithms supported by both $\mathcal{M}$ and $\mathcal{V}$ for the `PRF` function.

SP_V^{SIGHV} The list of algorithms supported by $\mathcal{V}$ for the digital signature between $\mathcal{H}$ and $\mathcal{V}$ (σ_H and σ_V).

SP_V^{Enc} The list of algorithms supported by $\mathcal{V}$ for asymmetric encryption (for χ).

SP_V^{Tun} The list of protocols supported by $\mathcal{V}$ for the tunnel and its authentication.

SP_M^{TEnc} The list of algorithms supported by $\mathcal{M}$ for the tunnel encryption.

R^{HCert} (optional) If certificates are managed with up-to-date CRL (second scenario in **??**), this requests $\mathcal{H}$ to send its certificate.

$CERT_V$ (optional) If certificates are managed with CRL (second scenario in **??**), the certificate of $\mathcal{V}$.

1.4 HA1 message (from $\mathcal{V}$ to $\mathcal{H}$)

N_H A nonce, chosen by $\mathcal{H}$. The nonce size must be 32 bytes.

EK A payload containing χ which is tk encrypted for $\mathcal{V}$.

MAC_H A payload containing μ_H.

SIG_H A payload containing σ_H.

S_H^{Tun} The settings for $\mathcal{H}$-$\mathcal{V}$ tunnel: the protocol, the authentication mechanism and other settings depending on the type of tunnel.

S_H^{TEnc} The settings for $\mathcal{H}$-$\mathcal{M}$ encryption: the protocol, the cryptographic algorithms and other settings depending on the type of encryption. Can be *none* if both $\mathcal{M}$ and $\mathcal{H}$ prefers not enabling it.

S_H^{Host} The connection settings for the mobile, it can contain the IP address, the DNS server to use and the gateway.

SP_{MH}^{MAC} The list of algorithms supported by $\mathcal{M}$ and $\mathcal{H}$ for the `MAC` function.

SP_{MVH}^{PRF} The list of algorithms supported by $\mathcal{M}$, $\mathcal{V}$ and $\mathcal{H}$ for the `PRF` function.

SP_{VH}^{SIGHV} The list of algorithms supported by $\mathcal{V}$ and $\mathcal{H}$ for the digital signature between $\mathcal{H}$ and $\mathcal{V}$ (σ_H and σ_V).

$CERT_H$ (optional) If it has been requested by $\mathcal{V}$, the certificate of $\mathcal{H}$.

1.5 HA2 message (from $\mathcal{V}$ to $\mathcal{M}$)

N_H The nonce chosen by $\mathcal{H}$.

MAC_H A payload containing μ_H.

SP_{MH}^{MAC} The list of algorithms supported by $\mathcal{M}$ and $\mathcal{H}$ for the `MAC` function.

SP_{MVH}^{PRF} One `PRF` function algorithm supported by $\mathcal{M}$, $\mathcal{V}$ and $\mathcal{H}$, and used for all PRF computations.

S_{HV}^{Host} The connection settings for the mobile, it can contain the IP address, the DNS server to use and the gateway.

S_H^{TEnc} The settings for $\mathcal{H}$-$\mathcal{M}$ encryption: the protocol, the cryptographic algorithms and other settings depending on the type of encryption. Can be *none* if both $\mathcal{M}$ and $\mathcal{H}$ prefers not enabling it.

1.6 MA1 message (from $\mathcal{M}$ to $\mathcal{V}$)

MAC_M A payload containing μ_M.

S_M^{TEnc} (optional) The settings for $\mathcal{M}$-$\mathcal{H}$ encryption: the protocol, the cryptographic algorithms and other settings depending on the type of encryption.

1.7 MA2 message (from $\mathcal{V}$ to $\mathcal{H}$)

SIG_V A payload containing σ_V.

MAC_M A payload containing μ_M.

S_V^{Tun} The settings for $\mathcal{V}$-$\mathcal{H}$ tunnel: the protocol, the authentication mechanism and other settings depending on the type of tunnel.

S_M^{TEnc} (optional) The settings for $\mathcal{M}$-$\mathcal{H}$ encryption: the protocol, the cryptographic algorithms and other settings depending on the type of encryption.

2 RAKE payload format

Each RAKE message is encoded as a linked list of payloads. This chapter describes each of these payloads and fields in them.

2.1 RAKE header

0 8 16 24 31

Next Payload	Rdv	Ver	Seq No

2.1.1 Next Payload

The next payload identifier, can either be:

0x00	No payload
0x01	Nonce payload
0x02	Identity payload
0x03	Security Proposal payload
0x04	EKT payload
0x05	MAC payload
0x06	Signature payload
0x07	Certificate payload
0x08	Request payload
0x09	Setting payload

2.1.2 Reserved

This field should be left blank.

2.1.3 Version

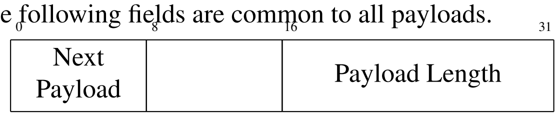
The version of the protocol, currently 0.

2.1.4 Sequence No

The sequence number of the message. Used for retransmission when the RAKE protocol is used directly over UDP.

2.2 Generic payload

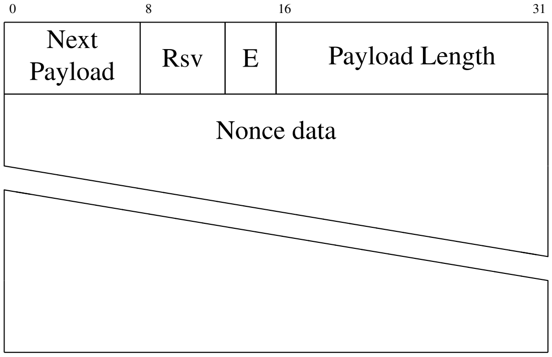
The following fields are common to all payloads.



2.2.1 Payload Length

Length of the payload ("header" of the payload included) in bytes.

2.3 Nonce payload



2.3.1 Entity (E)

The entity which have generated this nonce, can either be:

0x01 The mobile ($\mathcal{M}$)
0x02 The visited network ($\mathcal{V}$)
0x04 The home network ($\mathcal{H}$)

The entity values can be *xored* if the payload is about several entities, which is not applicable for nonce.

2.4 Identity payload

0	8		16 31
Next Payload	Type	E	Payload Length
Identity data			

2.4.1 ID Type

The identity type, among:

0x00 Undefined
0x01 IPv4 address
0x02 Fully-qualified address name (FQDN)
0x03 RFC822 [**?**] address
0x04 IPv6 address
0x05 ASCII

2.5 Security Proposal payload

0	8		16 31
Next Payload	Type	E	Payload Length
# Proposals	Proposal 1	···	Proposal N

2.5.1 Security Proposal (SP) Type

The SP type, among:

0x00 PRF algorithms
0x01 MAC algorithms
0x02 Digital signature algorithms
0x03 Encryption algorithms
0x04 Digest algorithms
0x05 Tunnel encryption algorithms
0x06 Algorithms for the $\mathcal{H}$-$\mathcal{V}$ tunnel

2.5.2 # proposals

The number of 8-bit proposals in the next field.

2.5.3 Proposal

The proposal value is one of the following values (non-exhaustive list). PRF algorithm proposals:

0x00 PRF function of TLS (defined in [**?**])
0x01 SHA-1 function

MAC algorithm proposals:

0x02 HMAC-MD5
0x03 HMAC-SHA1

Digital signature proposals:

0x04 RSA PKCS #1

Encryption proposals:

0x05 RSA PKCS #1 v1.5

Digest function proposals:

0x06 MD5 function
0x07 SHA-1 function

Tunnel encryption:

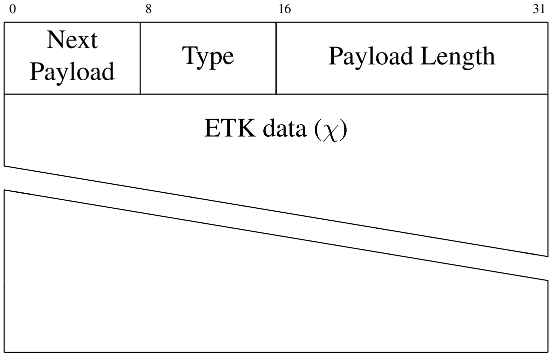
0x08 None (disabled encryption)
0x09 IPsec-ESP tunnel with DES CBC encryption and HMAC-MD5 authentication
0x0A IPsec-ESP tunnel with DES CBC encryption and HMAC-SHA1 authentication
0x0B IPsec-ESP tunnel with DES CBC encryption and HMAC-SHA256 authentication
0x0C IPsec-ESP tunnel with DES CBC encryption and HMAC-SHA512 authentication
0x0D IPsec-ESP tunnel with 3DES CBC encryption and HMAC-MD5 authentication
0x0E IPsec-ESP tunnel with 3DES CBC encryption and HMAC-SHA1 authentication
0x0F IPsec-ESP tunnel with 3DES CBC encryption and HMAC-SHA256 authentication
0x10 IPsec-ESP tunnel with 3DES CBC encryption and HMAC-SHA512 authentication
0x0D IPsec-ESP tunnel with AES-CBC-128 encryption and HMAC-MD5 authentication
0x0E IPsec-ESP tunnel with AES-CBC-128 encryption and HMAC-SHA1 authentication

0x0F IPsec-ESP tunnel with AES-CBC-128 encryption and HMAC-SHA256 authentication
0x10 IPsec-ESP tunnel with AES-CBC-128 encryption and HMAC-SHA512 authentication
0x11 IPsec-ESP tunnel with AES-CBC-192 encryption and HMAC-MD5 authentication
0x12 IPsec-ESP tunnel with AES-CBC-192 encryption and HMAC-SHA1 authentication
0x13 IPsec-ESP tunnel with AES-CBC-192 encryption and HMAC-SHA256 authentication
0x14 IPsec-ESP tunnel with AES-CBC-192 encryption and HMAC-SHA512 authentication
0x16 IPsec-ESP tunnel with AES-CBC-256 encryption and HMAC-MD5 authentication
0x17 IPsec-ESP tunnel with AES-CBC-256 encryption and HMAC-SHA1 authentication
0x18 IPsec-ESP tunnel with AES-CBC-256 encryption and HMAC-SHA256 authentication
0x19 IPsec-ESP tunnel with AES-CBC-256 encryption and HMAC-SHA512 authentication

$\mathcal{V}$-$\mathcal{H}$ tunnel:

0x1A IPsec-AH tunnel with HMAC-MD5 for authentication
0x1B IPsec-AH tunnel with HMAC-SHA1 for authentication
0x1C IPsec-AH tunnel with HMAC-SHA256 for authentication
0x1D IPsec-AH tunnel with HMAC-SHA512 for authentication

2.6 Encrypted Temporal Key payload



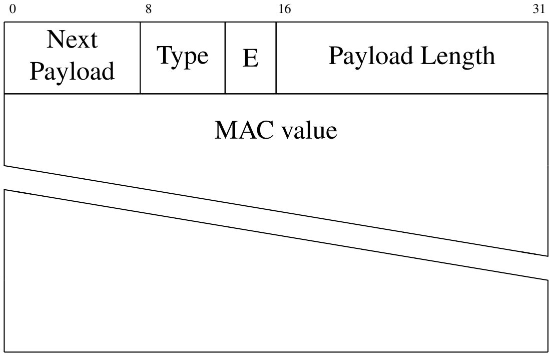
0	8	16	31
Next Payload	Type	Payload Length	
ETK data (χ)			

2.6.1 ETK Type

The ETK type, among:

0x05 RSA PKCS #1 v1.5

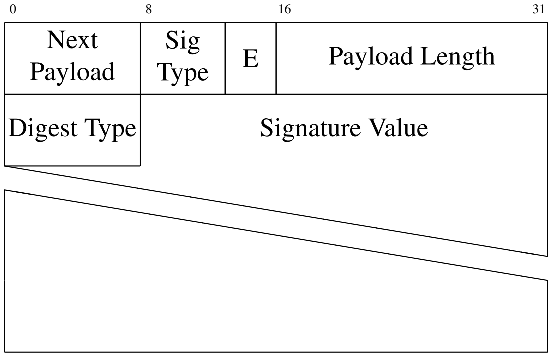
2.7 MAC payload



2.7.1 ETK Type

Values are the same as the proposals related to encryption.

2.8 Signature payload



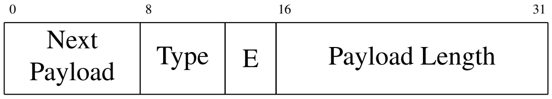
2.8.1 Signature Type

Values are the same as the proposals related to signature.

2.8.2 Digest Type

Values are the same as the proposals related to digest.

2.9 Request payload

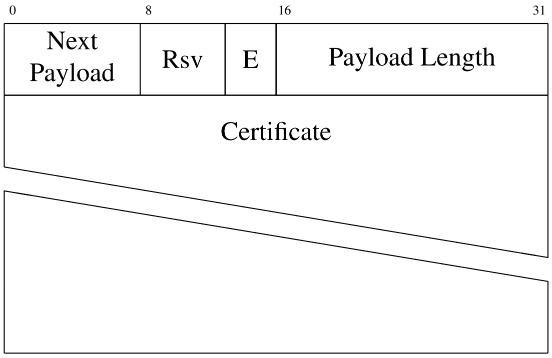


2.9.1 Request Type

The request type can either be:

0x00 Signature request
0x01 Accounting request
0x02 Certificate request

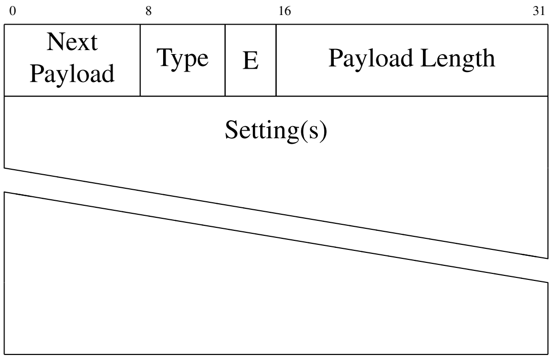
2.10 Certificate payload



2.10.1 Certificate

The certificate in DER (Distinguished Encoding Rules), i.e. binary, encoding

2.11 Setting payload



2.11.1 Setting Type

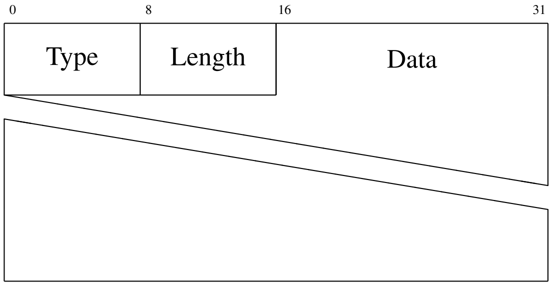
The request type can either be:

0x01 $\mathcal{M}$-$\mathcal{H}$ Encryption
0x02 $\mathcal{V}$-$\mathcal{H}$ Tunnel
0x03 Host configuration

2.11.2 Setting(s)

A list of settings. Each setting is encoded as a *setting subpayload*.

2.12 Setting subpayload



2.12.1 Setting Subpayload Type

The request type can either be:

0x01 Host IP address
0x02 Gateway IP address
0x03 DNS primary server IP address
0x04 DNS secondary server IP address
0x05 IPsec SPI for $\mathcal{H}$-$\mathcal{V}$
0x06 IPsec SPI for $\mathcal{V}$-$\mathcal{H}$
0x07 IPsec SPI for $\mathcal{M}$-$\mathcal{H}$
0x08 IPsec SPI for $\mathcal{H}$-$\mathcal{M}$
0x09 Tunnel start-point IP address or FQDN
0x10 Tunnel end-point IP address or FQDN
0x11 A proposal

2.12.2 Setting Subpayload Length

The length of this subpayload (in bytes).

2.12.3 Setting Subpayload Data

The data in itself. If it is an address or a Fully-Qualified Domain Name (FQDN), the data field begins with a 8-bit field defining its type (IPv4, IPv6 or FQDN).