

Using IKEv2 for Key Management for PSP

`draft-smyslov-ipsecme-ikev2-psp`

Valery Smyslov

IPsec Workshop, Vienna, July 2026

PSP Security Protocol (PSP)

- Designed by Google in 2022 for using in large-scale data centers
- Inherits some concepts from IPsec (like SA)
- Optimized for high performance
 - no replay protection
 - mandatory UDP encapsulation
 - decryption key for an SA is computed from a master key stored in NIC
 - time-based IV generation

PSP Key Management

- Each peer has master key that is rotated daily (actually there are two master keys to handle overlap)
- Each side deterministically generates its own decryption key for a PSP SA from the current active master key based on the SA parameters (SPI and cipher suite)
- Peers **somehow** exchanged these keys at the time the PSP SA is being established
 - PSP does not define how this happens

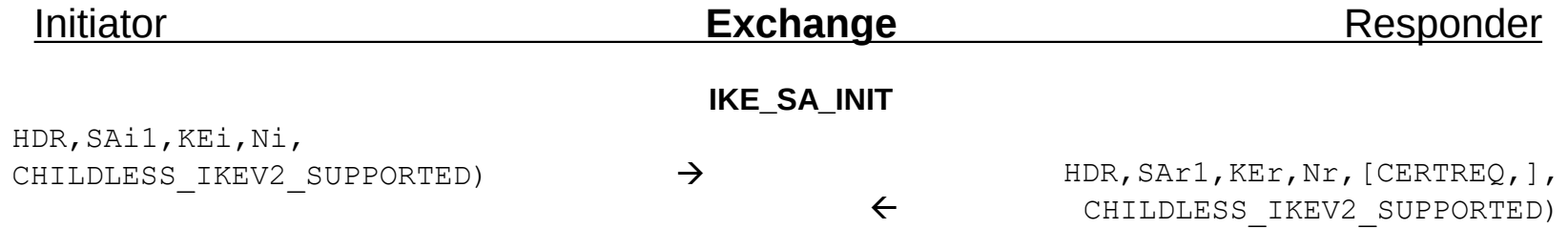
Can IKEv2 be used for PSP Key Management?

- No (in its current form)
 - IKEv2 computes a shared secret using some key exchange method(s) and derives SA keys for both directions from it, so these keys are **related**
 - in PSP each peer independently generates its own decryption key for an SA and these keys are **unrelated**

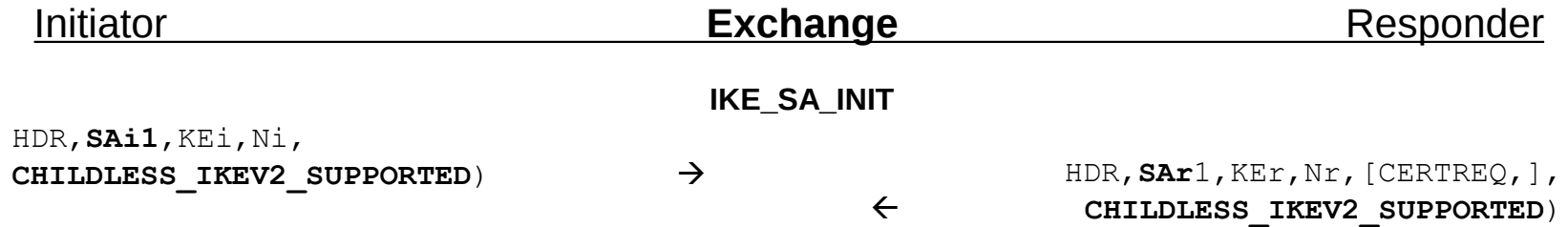
Can IKEv2 be extended for PSP Key Management?

- Yes ([draft-smyslov-ipsecme-ikev2-psp](#))
 - define new Security Protocol (PSP)
 - create a registry for PSP parameters
 - re-use the KD (Key Download) payload from [G-IKEv2](#)
 - re-use the KWA (key Wrap Algorithm) transform from [G-IKEv2](#)
 - re-use an ability to create childless IKE SA ([RFC6023](#))

Protocol Exchanges



Protocol Exchanges



Peers MUST negotiate in IKE_SA_INIT:

- Childless IKE SA (RFC6023)
- KWA transform in SA payload (RFC9838)

Protocol Exchanges

<u>Initiator</u>	<u>Exchange</u>	<u>Responder</u>
	IKE_SA_INIT	
HDR, SAi1, KEi, Ni, CHILDLESS_IKEV2_SUPPORTED)	→	HDR, SAr1, KEr, Nr, [CERTREQ,], CHILDLESS_IKEV2_SUPPORTED)
	←	
	IKE_AUTH	
HDR, SK{IDi, AUTH}	→	
	←	HDR, SK{IDr, AUTH}

Protocol Exchanges

<u>Initiator</u>	<u>Exchange</u>	<u>Responder</u>
	IKE_SA_INIT	
HDR, S <i>A</i> _{i1} , K <i>E</i> _i , N <i>i</i> , CHILDLESS_IKEV2_SUPPORTED)	→	HDR, S <i>A</i> _{r1} , K <i>E</i> _r , N <i>r</i> , [CERTREQ,], CHILDLESS_IKEV2_SUPPORTED)
	←	
	IKE_AUTH	
HDR, SK{ID <i>i</i> , AUTH}	→	
	←	HDR, SK{ID <i>r</i> , AUTH}

IKE SA is created in childless mode

Protocol Exchanges

Initiator	Exchange	Responder
	IKE_SA_INIT	
HDR, S <i>A</i> _{i1} , K <i>E</i> _i , N <i>i, CHILDLESS_IKEV2_SUPPORTED)</i>	→	HDR, S <i>A</i> _{r1} , K <i>E</i> _r , N <i>r</i> , [CERTREQ,], CHILDLESS_IKEV2_SUPPORTED)
	←	
	IKE_AUTH	
HDR, SK{ID <i>i</i> , AUTH}	→	
	←	HDR, SK{ID <i>r</i> , AUTH}
	CREATE_CHILD_SA	
HDR, SK{ S <i>A</i> _i , K <i>D</i> _i , T <i>S</i> _i , T <i>S</i> _r }	→	
	←	HDR, SK{ S <i>A</i> _r , K <i>D</i> _r , T <i>S</i> _i , T <i>S</i> _r }

Protocol Exchanges

Initiator	Exchange	Responder
	IKE_SA_INIT	
HDR, S <i>A</i> _{i1} , K <i>E</i> _i , N <i>i</i> , CHILDLESS_IKEV2_SUPPORTED)	→	HDR, S <i>A</i> _{r1} , K <i>E</i> _r , N <i>r</i> , [CERTREQ,], CHILDLESS_IKEV2_SUPPORTED)
	←	
	IKE_AUTH	
HDR, SK{ID <i>i</i> , AUTH}	→	
	←	HDR, SK{ID <i>r</i> , AUTH}
	CREATE_CHILD_SA	
HDR, SK{S <i>A</i> _i , K <i>D</i> _i , T <i>S</i> _i , T <i>S</i> _r }	→	
	←	HDR, SK{S <i>A</i> _r , K <i>D</i> _r , T <i>S</i> _i , T <i>S</i> _r }

MUST NOT be ----- changes in CREATE_CHILD_SA ----- **MUST be**
 Nonce payloads KD payloads (RFC9838)
 KE payloads PSP protocol in SA payload

Modified CREATE_CHILD_SA: existing entities

- **SA** payload **MUST** be present and **MUST** contain proposal(s) with **PSP** security protocol
 - this proposal **MUST** only contain **PSP Parameters** transform(s)
- **Traffic Selector** payloads **MUST** be present
- **USE_TRANSPORT_MODE** notification **MAY** be present
- **REKEY_SA** notification **MAY** be present

Modified CREATE_CHILD_SA: new entities

- **Key Download** payloads (RFC9838) **MUST** be present; the sender of a KD payload provides keying material that it will use as a receiver
 - each KD payload **MUST** contain **Group Key Bag** substructure(s) (RFC9838) for PSP protocol with SPI(s) corresponding to proposal(s) in SA payload
 - each substructure **MUST** contain exactly one **SA_KEY** attribute (RFC9838) with **Key ID** and **KWK ID** fields set to zero, containing PSP key wrapped using negotiated KWA

Modified CREATE_CHILD_SA: prohibited entities

- **Nonce** payloads **MUST NOT** be present
- **Key Exchange** payloads **MUST NOT** be present

PSP SA Negotiation

- Initiator's SA payload MAY contain several proposals for PSP (with different SPIs)
 - each proposal MAY contain several PSP Parameters transforms
- Responder MUST select and return single PSP proposal with single PSP parameters transform in it
- The number of KD payloads MUST correspond to the number of proposals in SA payload

Key Wrapping

- PSP keys are wrapped in SA_KEY attribute using the Key Wrap Algorithm (KWA) negotiated in IKE_SA_INIT
- Wrapping key is computed as
`SK_w = prf+(SK_d, "Key Wrap for PSP")`

Thanks!