

# Auxiliary Exchange Authentication

`draft-smyslov-ipsecme-ikev2-aux`

Valery Smyslov  
svan@elvis.ru

IETF 102

# Auxiliary Exchange

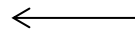
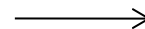
Auxiliary Exchange (**IKE\_AUX**) takes place between IKE\_SA\_INIT and IKE\_AUTH:

Initiator

Responder

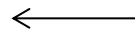
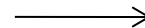
**IKE\_SA\_INIT**

HDR(MID=0), SAi1, KEi, Ni,  
N(AUX\_EXCHANGE\_SUPPORTED)



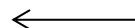
**IKE\_AUX**

HDR(MID=1), SK{...}



**IKE\_AUTH**

HDR(MID=2), SK{IDi, AUTH, SAi2, TSi, TSr}



**IKE\_SA\_INIT**

HDR(MID=0), SAR1, KEr, Nr  
N(AUX\_EXCHANGE\_SUPPORTED)

**IKE\_AUX**

HDR(MID=1), SK{...}

**IKE\_AUTH**

HDR(MID=2), SK{AUTH, SAR2, TSi, TSr}

# Auxiliary Exchange Authentication

- Currently draft defines that IKE\_AUX messages are authenticated by including their ICVs in the signature calculation in IKE\_AUTH:

```
InitiatorSignedOctets = RealMessage1 | AUX_I | NonceRData | MACedIDForI
AUX_I = ICV_I_1 [ | ICV_I_2 [ | ICV_I_3 ... ] ]
ResponderSignedOctets = RealMessage2 | AUX_R | NonceIDData | MACedIDForR
AUX_R = ICV_R_1 [ | ICV_R_2 [ | ICV_R_3 ... ] ]
```

(where `ICV_[I|R]_[N]` are Integrity Check Values from the corresponding IKE\_AUX messages)

- Identified problems (Scott Fluhrer, Daniel Van Geest):
  - (minor) Not all AEAD algorithms produce separate ICV, some may spread authentication information over the ciphertext
    - all AEAD algorithms currently defined for IPsec (CCM, GCM, Chacha20-Poly1305) produce separate ICV
  - (major) Some widely used AEAD algorithms (e.g. GCM) are not second preimage resistant when an attacker knows the key

# Attack Description

- Attacker in the middle equipped with Quantum Computer capable to break DH exchange in IKE\_SA\_INIT in real time (which is presumably not quantum-safe) can learn the keys used to protect subsequent IKE\_AUX messages (SK\_e\*/SK\_a\*)
- If negotiated AEAD algorithm is not resistant to second preimage attack with known key, then the attacker **can change content of these messages** so that peers would not notice this fact
- If these IKE\_AUX messages contain public values for Quantum Safe Key Exchange methods, the attacker **can substitute them with her own**
- If the attacker manages to substitute QSKE public values in such a way, that the peers compute the same SKEYSEED (which she knows), then IKE\_AUTH will succeed and the attacker will **mount a successful MitM attack**

# Possible Solution (1)

- Include whole IKE\_AUX messages into the signature calculation in IKE\_AUTH:

```
InitiatorSignedOctets = RealMessage1 | AUX_I | NonceRData | MACedIDForI  
AUX_I = MSG_I_1 [ | MSG_I_2 [ | MSG_I_3 ... ] ]  
ResponderSignedOctets = RealMessage2 | AUX_R | NonceIDData | MACedIDForR  
AUX_R = MSG_R_1 [ | MSG_R_2 [ | MSG_R_3 ... ] ]
```

(where `MSG_[I|R]_[N]` are corresponding IKE\_AUX messages)

- Properties:
  - completely thwarts the attack
  - peers need to keep IKE\_AUX messages until IKE\_AUTH completes, which opens possibility for DoS attack, since these messages could be large

# Possible Solution (2)

- Include hashes of IKE\_AUX messages into the signature calculation in IKE\_AUTH:

```
InitiatorSignedOctets = RealMessage1 | AUX_I | NonceRData | MACedIDForI  
AUX_I = H(MSG_I_1) [ | H(MSG_I_2) [ | H(MSG_I_3) ... ] ]  
ResponderSignedOctets = RealMessage2 | AUX_R | NonceIData | MACedIDForR  
AUX_R = H(MSG_R_1) [ | H(MSG_R_2) [ | H(MSG_R_3) ... ] ]
```

(where  $H(MSG_{[I|R]}_{[N]})$  are hashes of corresponding IKE\_AUX messages calculated using collision-resistant hash function)

- Properties:
  - completely thwarts the attack
  - IKEv2 doesn't negotiate hash function primitive, so new IANA registry would be needed as well as new negotiation mechanism (or new Transform Type)
    - increases both protocol complexity and size of IKE\_SA\_INIT messages

# Possible Solution (3)

- Similar to solution (2), but uses negotiated PRF with all zero key instead of hash function:

```
InitiatorSignedOctets = RealMessage1 | AUX_I | NonceRData | MACedIDForI  
AUX_I = PRF(0,MSG_I_1) [ | PRF(0,MSG_I_2) [ | PRF(0,MSG_I_3) ... ] ]  
ResponderSignedOctets = RealMessage2 | AUX_R | NonceIData | MACedIDForR  
AUX_R = PRF(0,MSG_R_1) [ | PRF(0,MSG_R_2) [ | PRF(0,MSG_R_3) ... ] ]
```

(where `PRF(0,MSG_[I|R]_[N])` are results of applying PRF with all zero key to corresponding IKE\_AUX messages)

- Properties:
  - thwarts the attack if negotiated PRF is resistant to second preimage attack with known key; among the currently defined PRFS for IKEv2:
    - all HMAC-based PRFs are resistant
    - PRF\_AES128\_XCBC and PRF\_AES128\_CMAC are not
      - these PRFs are not quantum-resistant anyway since they use 128-bit key

# Way Forward

- After some discussion on the list the proposed solution (3) looks like best possible compromise
- Any other ideas?
- Comments? Questions?

Thank you!