FAST RE-ESTABILISHMENT OF IKEV2 SECURITY ASSOCIATIONS FOR RECOVERY OF IPSEC GATEWAYS IN MOBILE NETWORK

Peng Yang, Yuanchen Ma and Satoshi Yoshizawa
Hitachi (China) Research and Development Corporation
301, North Wing, Building C, Raycom Infotech Park, Beijing 100190, China

Keywords: IKEv2, Fail-over, IPsec, IPsec gateway, Fast IKEv2 SA re-establishment.

Abstract: IKEv2/IPsec has been widely deployed, such as in VPN and MIPv6, to support mutual authentication, access control and traffic protection in internet. IKEv2/IPsec gateways may maintain huge number of IKEv2/IPsec security associations. If gateway encounters failure or over-load, it will take a long time to re-establish security associations in another IKEv2/IPsec gateway. The major reason is that regular procedure of IKEv2 incurs long delay because of multiple signalling exchanges and complex computation especially in Diffie-Hellman exchange. In this paper, a new IKE SA re-establishment solution is proposed to reduce the overhead of computation and signalling by directly transferring IKE SA from old gateway to new gateway via independent IKE SA storage (stub bank). The most expensive Diffie-Hellman exchange and some of signalling can be avoided. Therefore, a huge amount of IKE/IPsec security associations can be re-established in a short time. The applicability of this solution in mobile network is further analyzed as well.

1 INTRODUCTION

In current Internet world, IP security (IPsec) (Kent, S. and K. Seo, 2005) has been widely deployed to provide confidentiality, data integrity, access control, and data source authentication to IP data packets. These services of security are supported by maintaining shared state between source/destination of an IP datagram. This state, which has cryptographic algorithms and related keys, defines specific treatment on datagram.

The straightforward way to establish the shared state is manual configuration. However, this way is rather static, which may incur serious risk after a long run. Therefore, the protocol called Internet Key Exchange version 2 (IKEv2) (Kaufman, C., 2005) was designed to establish state of IPsec security associations (SA) dynamically. IKEv2 achieves mutual authentication and access control between two parties and establishes an IKE SA. IKE SA includes shared secret that can be used to efficiently establish IPsec CHILD SAs. IPsec has two modes: Encapsulating Security Payload (ESP) and Authentication Header (AH). A set of cryptographic algorithms are used by the IPsec SAs to protect the carried traffic.

Basically, the IKEv2 procedure is comprised of two exchanges as shown in Figure 1: IKE_SA_INIT and IKE_AUTH (Kaufman, C., 2005). In IKE_SA_INIT, the end nodes negotiate cryptographic algorithms, exchange NONCES, and do Diffie-Hellman exchange (Diffie, W., and Hellman M., 1976). The second stage (IKE_AUTH) authenticates previous messages, exchange identities and certificates, and establish first IPsec CHILD_SA.

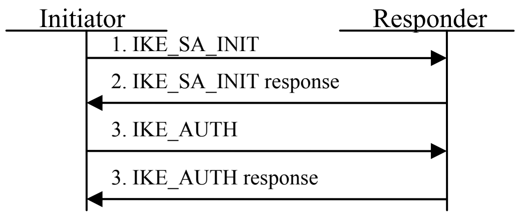

Figure 1: Basic procedure of IKEv2.

After first two exchanges, IKE SA is set up for maintenance of IPsec SAs and protection of IKEv2 messages. With IKE SA, the two nodes can continue to establish other necessary IPsec CHILD_SAs by CREATE_CHILD_SA messages,

In IKE_SA_INIT, Diffie-Hellman (DH) is used to set up a session secret, from which cryptographic

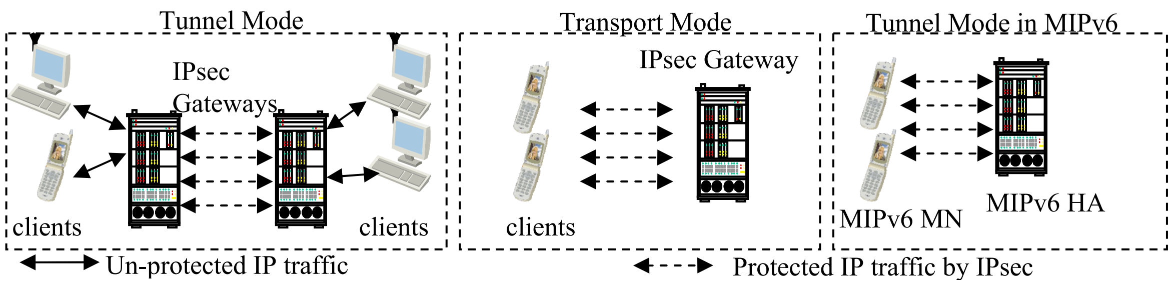

Figure 2: Application scenarios of IKEv2/IPsec.

keys are derived. DH exchange is rather intensive in computation, since exponential calculation of large prime factor (at least 768 bits as defined by IETF) is done on both nodes. Some works have been done to accelerate the expensive DH algorithm (Daniel J. Bernstein, 2006). Besides DH exchange, timing overhead of IKEv2 signalling is rather big especially when the Extensible Authentication Protocol (EAP) (Aboba, B., et.al, 2004) is used for third-party authentication. According to our experiment, time consumed by initial two exchanges of IKEv2 could typically be 1 second in PentiumM PC with some other protocol daemons running together. In the meanwhile, the follow-up CREATE_CHILD_SA exchanges are rather less complex and fast.

Typical application scenarios of IKEv2/IPsec are shown in Figure 2. In IPsec tunnel mode, entire IP packet (data and IP header) is encapsulated into a new IP packet and encrypted and/or authenticated by ESP/AH header. Tunnel mode is normally used by VPN for network-to-network communications. In IPsec transport mode, AH/ESP header is added after the IP header for encrypted and/or authenticated, which is used for host-to-host communications. The third scenario (Arkko, J., et. al, 2004) is tunnel mode for host-to-network communication, which is typically used to protect traffic between Mobile IPv6 (MIPv6) Home Agent (HA) and Mobile Node (MN). Sometimes the third one is still treated as tunnel mode. In all of these scenarios, IPsec gateway/client can be either IKEv2 initiator or responder.

In all the three scenarios, IPsec gateways (eg: VPN server, MIPv6 HA) may serve a bunch of IPsec clients simultaneously. If IPsec Gateway fails/over-loads, re-establishment of IKEv2/IPsec SAs during recovery will be quite time-consuming. Gateway must re-run time-consuming exchanges of IKEv2 initiation for a large number of SAs with all end points. A big number of concurrent IKE_INIT could have high possibility of exhausting CPU resources of IPsec gateway. And, the long time of IPsec gateway recovery will also harm service experience of uses. The same problem happens in the security protocols of other layers. Solutions were made to do fast recovery as well, such as in TLS (Salowey, J., et.al., 2008).

In this paper, a new IKEv2 SA re-establishment solution is proposed to reduce the overhead of computation and signalling by directly transferring IKE SAs from old gateway to new gateway via independent IKE SA storage (stub bank). The most expensive Diffie-Hellman calculation and part of exchanges can be avoided. Clients can re-establish IKE SAs without IKE initiation exchanges during gateway recovery. The paper is organized as below: in chapter 2, the solution is discussed in details, wherein the applicability in mobile network will also be discussed. The security and performance are evaluated in chapter 3, which is followed by the conclusion.

2 PROPOSED SOLUTION

2.1 Re-building Key-ring of IKE SA

Upon failure or over-load, IPsec gateway loses all IKE SAs stored locally. A new data structure called "stub" is created, which has part information of all IKE SAs. And gateway can use this data structure to accelerate rebuilding of IKE SAs. The stubs are stored in stub bank, which may be independent equipment or co-locate with some other servers (such as AAA server, etc). One stub must at least include following information:

- IDi, IDr.
- SPIi, SPIr.
- SAr (the accepted proposal).
- SK_d_old.
- life-time

The new keys are derived from SKEYSEED. The way of making SKEYSEED is borrowed from

IKEv2 base protocol (Kaufman, C., 2005) with some simplification as below:

$$SKEYSEED = prf(SK\_d\_old, Ni \mid Nr) \quad (1)$$

"prf" refers to "pseudo-random function", one of the cryptographic algorithms negotiated in IKEv2. SK_d_old is retrieved from corresponding stub by gateway and used for deriving new keys. It definitely exists in IKE SAs of clients. Ni and Nr are nonce generated by both sides. They are conveyed by means defined in chapter 2.3.

After SKEYSEED is derived, new keys of re-established IKE SA can be generated as following:

$$\begin{aligned}&\{SK\_d \mid SK\_ai \mid SK\_ar \mid SK\_er \mid SK\_ei \mid SK\_pi \mid SK\_pr\}\\&= prf+(SKEYSEED, Ni \mid Nr \mid SPIi \mid SPIr\_new)\end{aligned} \quad (2)$$

The usage of keys derived in (2) can be found in (Kaufman, C., 2005). SPIi is provided by client. SPIr_new is newly generated by IPsec gateway to avoid potential SPI conflict.

2.2 Extensions on IKEv2 Protocol

As for protocol, it’s important to make the mutual authentication between clients and new IPsec gateway. Client must provide references of stubs, since new gateway probably doesn’t have stubs beforehand at all. With this reference, new gateway can easily find the stub from stub bank. In this sense, the reference must uniquely index the corresponding stub related to each client. The format of index can be defined by operators of stub bank. In this solution, the quaternion of {IDi, IDr, SPIi, SPIr} is used as stub reference.

In this solution, the regular IKE_SA_INIT exchange is extended to support fast re-establishment of IKE SA. When, client knows failure or over-load of old Gateway and discovers new IPsec gateway, it will initiate the IKE_SA_INIT message with extensions as Figure 3. The content of this message is defined below:

HDR, SAi1, KEi, Ni, [REF], SK{[REF]}. (M1)

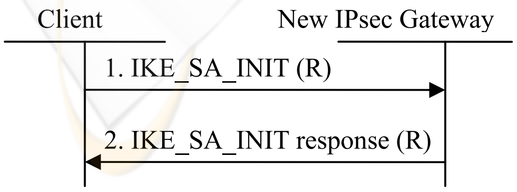

Figure 3: Exchange for re-establishment of IKE SA.

The first four fields are same as the regular IKE_SA_INIT message in IKEv2 protocol. [REF] is reference of stub, which is the quaternion {IDi, IDr, SPIi, SPIr} in this paper. SK{}, as defined in (Kaufman, C., 2005), contains encrypted stub reference. The encryption algorithm is from stub.

Upon receiving this message, gateway will first get the reference from [REF] and retrieve the related stub from stub bank. Then gateway needs to decrypt the SK{} value and compare the decrypted content with [REF]. If they match each other, gateway can successfully authenticate the client and proceed to calculate new key-ring as defined in chapter 2.1. With the new key materials, gateway could build the IKE_SA_INIT response as below:

HDR, Nr, SK{IDr, Nr}. (M2)

HDR is the message header with the type of IKE_SA_INIT. The new SPIr can be found inside as well. Nr is the nonce to refresh key materials. SK{} is encrypted by the new key materials in (2). With the information in this message, the client can re-calculate the key-ring and decrypt the SK{} content. The decrypted Nr value will be compared with the Nr value in the message, in order to authenticate the gateway. If result is OK, the new IKE SA can be set up quickly without DH exchange and some other exchanges (like EAP, etc). Both nodes can proceed to generate the IPsec CHILD SA as defined in (Kaufman, C., 2005).

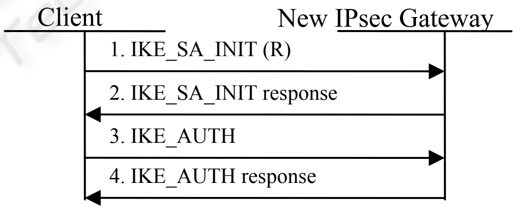

Figure 4: Exchange for re-establishment of IKE SA in unsuccessful case.

The fast IKE SA request from client may be rejected by gateway. For example, authentication of client may failure, or the stub reference is not valid, or stub is no longer valid upon the request, etc. In these cases, gateway can fall back to the regular IKEv2 procedure as shown in Figure 4. If gateway has received the IKE_SA_INIT(R) as defined at message (M1) and decided to reject this request, it will send regular IKE_SA_INIT response to client and follow the regular procedure of IKEv2 as Figure 1. On the other hand, if client fails to authenticate the gateway after message (M2), it can either choose to re-initiate the procedure in Figure 3, or trigger the regular IKEv2 initiation procedure as Figure 1.

2.3 Operations of Stub Bank

There are many ways to build stub bank. It can be independent equipment, or be together with some servers like AAA server, OMC, etc. Whatever embodiment of stub bank is, there must be some way to synchronize stubs with IKE SAs on IPsec gateway. The protocols to synchronize stubs could be extension of other protocols (such RADUIS, etc), or new protocols. In this paper, we implemented a simple protocol for maintenance of stub. The stubs will be expired after the corresponding IKE SA is expired. Another problem is when the stubs should be synchronized. Basically, when IKE SA is set-up, re-keyed, expired, or closed, the related stubs must be synchronized between stub bank and IPsec gateway. Sometimes, the IKE session may exist for a long time without re-key, operators can define a cycle for synchronization. In this paper, the timer of stub synchronization cycle is set to be half cycle of the re-key timer. Stub is only valid for one time use. After successful transaction, the stub will be deleted. And, new IPsec gateway must update the stub while IKE SA is re-built.

2.4 Analysis on Applicability in Mobile Network

As specified in (Devarapalli V., et.al, 2007) and (Arkko, J., et. al, 2004), IPsec is mandatory to protect mobile IPv6 (MIPv6) signalling by transport mode. It's recommended to use protect traffic by tunnel mode. IKE SA is set up between mobile node (MN) and home agent (HA). CHILD SAs are between home address (HoA) of MN and HA, while IKE SA is between Care-of Address (CoA) and HA.

When HA, which is also IPsec gateway, fails or overloads The route of CoA in MN is still available. It can use CoA to initiate the re-establishment of IKE SA quickly. After IKE SA is set up, MN can use build the CHILD SA between HoA and HA by regular IKEv2 protocol. The solution in this paper does not rely on HoA of MN. So, unavailability of HoA route does not hurt its applicability in MIPv6.

Another consideration is handover. MN gets new CoA in another network after handover. If HA fails before MIPv6 re-registration and IKE/IPsec SA update, the new CoA will not be updated in the network side. However, the solution of this paper does IKE SAs' rebuilding and mutual authentication without the help from CoA as well.

Lastly, mobile network has limited radio resources, especially in reverse link. When, a big number of clients want to initiate a procedure at the same time, the intensive signalling may congest the reverse link. But, in the solution of this paper, the extension part of IKE_SA_INIT is quite short: a few bytes for reference and encrypted reference. It's has much low possibility of congestion.

3 EVALUATIONS

3.1 Security Analysis

3.1.1 Mutual Authentication

One of the most important functions of DH exchange is to authenticate the IKEv2 client and gateway with each other.

SK_d is keying material of IKE SA for deriving new keys for CHILD SAs (Kaufman, C., 2005). It is updated when IKE SA re-key is done. Mutual authentication in this paper is done based on SK_d. While client initiates IKE_SA_INIT with extensions, it must encrypt the stub reference in SK{} by SK_d. Gateway can authenticate the client by decrypting SK{} content using SK_d in the stub and comparing it to [REF] provided by client. In order to be authenticated by clients, gateway needs to enclose encrypted IDr and Nr in SK{} by the newly generated keying materials as defined in (2). Then, client will derive new keys and decrypt SK{} from new gateway. Finally, it can authenticate the gateway by comparing decrypted information to the corresponding part in the message.

Operator may want to refresh key material before IKE SA re-establishment. In this case, client can generate a temporary key (say SK_dt) for mutual authentication as (3). The information of SK_d_old, SPIi and SPIr can be found in stub. Ni is generated by client for refreshment of key materials.

$$\begin{aligned}&\{SK\_dt \mid SK\_ai \mid SK\_ar \mid SK\_ei \mid SK\_er\}\\&= prf+(SK\_d\_old, Ni \mid SPIi \mid SPIr)\end{aligned} \qquad (3)$$

3.1.2 Deny-of-Service (DoS) Attack

While new IPsec gateway fails/overloads, there may have a big number of clients try to initiate the IKE SA re-establishment at the same time. The gateway may treat this situation as Deny-of-Service (DoS) attack. This paper borrows the cookie mechanism of IKEv2 protocol with some extensions.

As shown in Figure 5, if new gateway finds potential risk of DoS attack, it will respond the message with cookie.

The format of message is defined below (M3).

HDR, N(Cookie). (M3)

While receiving cookie, client will re-initiate the IKE_SA_INIT with reference extensions and cookie. The format of this message is defined below (M4):

HDR, N(Cookie), SAi1, KEi, Ni, [REF], SK{[REF]}. (M4)

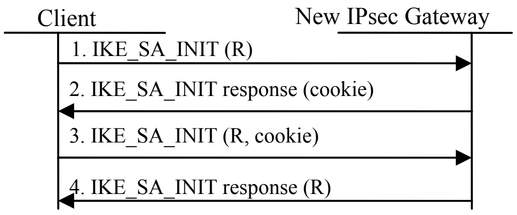
Client
New IPsec Gateway
1. IKE_SA_INIT (R)
2. IKE_SA_INIT response (cookie)
3. IKE_SA_INIT (R, cookie)
4. IKE_SA_INIT response (R)

Figure 5: Cookie mechanism for DoS attack.

At gateway, it will compare received cookie with the one stored locally to validate client. This attack can be less effective, if an implementation of a gateway uses minimal CPU and commits no state until it knows the client can receive packets at the address from which it claims to be sending them.

3.1.3 Life-time of Stub

It's important to set a life-time of stub. The main reason is IKE SA of client may be expired during unavailability of gateway. In this case, the fast IKE SA can not be finished, since there is no information for derivation of key materials on client. So, the life time of stub is recommended to be the same as IKE SA. Once key-ring is updated, life time of stub must be reset as well.

3.1.4 Detection of Failure or Overload

It's not the main target of this solution to define the way to detect the gateway failure or overload. One possible method is that client can know failure of IPsec gateway when multiple packets of ICMP unreachable are sent from network side. The operators who deployed this solution can define the related policies.

But, there is the possibility that client misjudge gateway failure/overload, while the IKE SA is still valid on the gateway. In this case, gateway may rebuild IKE SA and IPsec CHILD SA in vain, which is obviously unacceptable. Furthermore, there is the case that client keeps on sending IKE_SA_INIT, for example it is cheated continuously. So, gateway must reject IKE_SA_INIT with extension, while the corresponding IKE SA is still alive on gateway.

3.1.5 Integrity Protection and Replay Protection

The integrity checksum at the end of SK{} payload is used to protect the integrity of the entire messages as defined in (Kaufman, C., 2005).

Third mid-man may intercept the IKE_SA_INIT message and replay it to new gateway. The Ni in IKE_SA_INIT can be used to do replay protection. In this sense, the new gateway is needed to cache the nonce for a while. And, new gateway can extend the regular cookie mechanism of IKE_SA_INIT to check return routability of client as chapter 3.1.3.

3.2 Performance Analysis

The performance improvement can be separated into two aspects: time of IKE SA re-establishment and computational load on new IPsec gateway.

As for timing, considering single IKEv2 session, regular initiation procedure will at least need 2 round-trips. The solution in this paper only needs 1 round-trip of extended IKE_SA_INIT, which also avoids DH exchange. Since authentication server may be far away from gateway, the saved time in case of IKEv2 with EAP is very significant in real deployment. Furthermore, if certificate is used for authentication in regular IKEv2 protocol, more time will be saved as well.

As for computational load of IPsec gateway, it's obvious that DH exchange can be avoided, which is much more complex than symmetrical transforms used for SK{}.

3.3 Experimental Result

The solution is tested on openikev2 (openikev2). A simple stub bank is implemented with a simple protocol for maintenance of stubs. The gateway and client are supported by PC (Pentium M 1.2GHz, 512MB RAM, 1Gbps Ethernet, Fedora Core 4). The size of key is 128 bit. In the experiment, we just chose DH group 1 (768 bit modulo) and DH group 2 (1024 bit modulo). Certificate is not used in the regular IKEv2 procedure. To evaluate timing performance, some codes are added in software of client to code read the clock information (in micro-second) directly from CPU registers.

The network structure in the experiment is shown in Figure 6. All the equipments are connected by a CISCO 2950 switch in same VLAN. An attacker is added to test performance of replay protection and anti-forge. A simple rule of fail/overload detection is made: if 3 consecutive IPsec packets (with continuous sequence numbers) are echoed by the switch with ICMP unreachable, the client will start the procedure of fast IKE SA re-establishment with the other IPsec gateway. So, it can simulate fail/overload of gateway by shutting down the connection of original gateway. The addresses of gateways are pre-configured in clients.

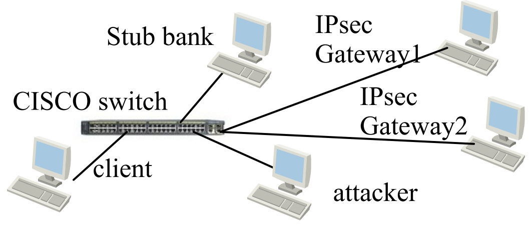

Figure 6: Experiment platform.

Table 1: Time of single connection.

	Regular IKEv2	Fast solution
DH group1	954 ms	1.73ms
DH group 2	1026 ms	1.82ms
DH group1 with EAP MD5	1722 ms	1.79ms
DH group2 with EAP MD5	2035 ms	1.81 ms

The time of single IKEv2 initiation is shown in Table 1. It can be seen that the time of fast solution in this paper can be saved by over 98% percent. If high order DH group is chosen, this improvement can be much bigger.

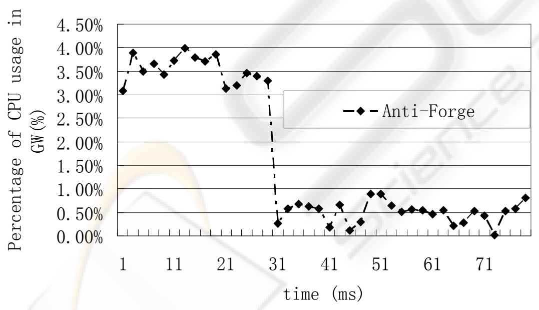

Figure 7: Evaluation of anti-forge.

In regular IKEv2 initiation procedure, the time is mainly put on DH exchange. And, much CPU resources are consumed by exponential calculation (50% percent on the PC). While the fast solution is applied, it only adds 4% CPU usage by avoiding DH exchange. So, when IPsec gateway serves a big number of clients this solution is much more applicable for.

In the experiment, a simple attacking test is designed by continuously sending a number of forged IKE_SA_INIT messages with different IP addresses from attacker to gateway2. The CPU usage is only increased by 3.5% averagely in a very short time because of stub retrieval and decryption. Then, gateway treats this case as DoS attack. The small amount of CPU usage (0.5%) is incurred by TCP/IP processing and cookie maintenance.

4 CONCLUSIONS

In this paper, a new solution is proposed to do stateless fast IKE SA re-establishment, while the IPsec gateway encounters failure or overload. By extending regular IKE_SA_INIT messages, it can avoid the most time-consuming IKEv2 exchanges and computationally intensive DH exchange to re-build IKE SAs. It can be much faster and light-weighted to transfer millions of IKE sessions from old gateway to target gateway. On the other hand, the derivation of key materials and mutual authentication can be done securely without compromising base IKEv2/IPsec framework. No sensitive information is sent though network. Lastly, the modification of base IKEv2 protocol is quite small.

REFERENCES

Kaufman, C. “Internet Key Exchange (IKEv2) Protocol”. RFC 4306, IETF, 2005

Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, IETF, December 2005.

Diffie, W., and Hellman M., "New Directions in Cryptography", IEEE Transactions on Information Theory, V. IT-22, n. 6, June 1976.

Arkko, J., et. al, “Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents”, RFC 3776, IETF, June 2004.

Salowey, J., et.al., "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, IETF, Jan. 2008.

Daniel J. Bernstein,”Curve25519: New Diffie-Hellman Speed Records”, Lecture Notes in Computer Science, Volume 3958/2006, pp 207-228, April, 2006

Kivinen, T. and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key xchange (IKE)", RFC 3526, IETF,May 2003.

openikev2, http://openikev2.sourceforge.net/

Aboba, B., et.al, “Extensible Authentication Protocol (EAP)”, RFC 3748, June 2004.

Devarapalli V., et.al, “Mobile IPv6 Operation with IKEv2 and the Revised IPsec Architecture”, RFC 4877, Apr. 2007.