

Kerberized Internet Negotiation of Keys (kink)
INTERNET-DRAFT
[draft-ietf-kink-ike-over-kkmp-00.txt](http://www.ietf.org/drafts/ietf-kink-ike-over-kkmp-00.txt)
Expires: 16 May 2001

T. Kivinen
SSH Communications Security
16 November 2000

Running IKE Phase 2 over Artificial Kerberos IKE SA

Status of This memo

This document is a submission to the IETF Kerberized Internet Negotiation of Keys (KINK) Working Group. Comments are solicited and should be addressed to the working group mailing list (ietf-kink@vpnc.org) or to the editor.

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/1id-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

Abstract

This document defines how to create artificial IKE SA using kerberos [[RFC-1510](#)]. It defines how to calculate SKEYID, cookies and IV needed by the IKE SA from the Kerberos session key. After the artificial IKE SA is created, it can be used to run normal IKE [[RFC-2409](#)] phase 2 negotiations. Those negotiations include quick Mode (creating IPsec SA), new group mode, delete notifications, and error notifications.

INTERNET-DRAFT

16 November 2000

Table of Contents

1.	Introduction	2
2.	Specification of Requirements	3
3.	SKEYID material Calculation	3
4.	IV and Cookie Calculation	3
5.	Transmitting the KRB_AP_* Messages Inside the IKE Payload . . .	4
6.	Example Quick Mode Negotiation	5
7.	Security Considerations	5
8.	References	6
9.	Authors' Addresses	6

[1.](#) Introduction

After the IKE [[RFC-2409](#)] phase 1 finishes it produces following information that is used to create the IKE SA:

CKY-I and CKY-R
cookies used to identify the IKE SA.

SKEYID
a string derived from secret material known only to the active players in the exchange.

SKEYID_e
keying material used by the IKE SA to protect its own messages.

SKEYID_a
keying material used by the IKE SA to authenticate its own messages.

SKEYID_d
keying material used to derive keys for IPsec SAs.

IV
initialization vector used by the IKE SA for the phase 2 messages.

IKE SA algorithms
encryption, hash, and authentication algorithms.

The SKEYID material can easily be created from the kerberos session key by just hashing it in the similar way they are created in the IKE [[RFC-2409](#)]. Cookies are just random numbers, but as they should remain constant over the negotiation between two parties, it would be desirable to derive them from the session key so they remain constant as long as the kerberos ticket is valid. Encryption, hash and authentication algorithms can be fixed to be 3DES, SHA-1 and HMAC-SHA1.

The IKE SA identities are not used after the phase 1, thus we do not need them here.

[I.](#) Kivinen

[page 2]

INTERNET-DRAFT

16 November 2000

The IKE SA is authenticated using the normal KRB_AP_REQ added to first packets in each phase 2 negotiations. The mutual authentication is done only if the phase 2 negotiation includes multiple packets, in which case the second packet encrypted using session key authenticates the responder.

Before this exchange can happen the initiator must first do normal kerberos authentication to KDC and receive valid kerberos ticket for the responder.

[2.](#) Specification of Requirements

This document shall use the keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" to describe requirements. They are to be interpreted as described in [[RFC-2119](#)] document.

[3.](#) SKEYID material Calculation

>From the kerberos ticket we have the session key, which is just a shared secret with the other host. We use that session key instead of the Diffie-Hellman shared secret in the SKEYID calculation. Also there is no point of including the cookies in the SKEYID calculation, as they are generated from the session key. The SKEYID is calculated as follows:

```
SKEYID = kerberos_session_key
SKEYID_d = prf(SKEYID, kerberos_session_key | 0)
SKEYID_a = prf(SKEYID, SKEYID_d | kerberos_session_key | 1)
```

$\text{SKEYID}_e = \text{prf}(\text{SKEYID}, \text{SKEYID}_a \mid \text{kerberos_session_key} \mid 2)$

4. IV and Cookie Calculation

To encrypt the Phase 2 messages the IKE SA needs a "last phase 1 CBC output block". This is used to calculate the IV for the first Phase 2 message. Because we do not have last block of the phase 1, we calculate the IV from the the KRB_AP_REQ message. The IV used for the first encrypted block in the first phase 2 message is derived from a hash of a concatenation of the KRB_AP_REQ and the phase 2 message id using the fixed SHA-1 hash algorithm. Because the IV length for 3DES is 8 bytes, we take first 8 bytes of the calculated hash. The IV is calculated like this:

$\text{IV} = \text{SHA-1}(\text{KRB_AP_REQ} \mid \text{MESSAGE-ID})[0..7]$

Cookies for the IKE SA are generated from the kerberos session key, so it will remain constant as long as the kerberos session key is valid. Both the initiator and responder cookies are the first 16 bytes of the SHA-1 hash of the kerberos_session_key concatenated with the number 42 encoded as one byte. The number 42 is selected arbitrarily, so that it will not match anything we already use. Initiator CKY-I and Responder CKY-R are calculated like this:

$\text{CKY-I} = \text{SHA-1}(\text{kerberos_session_key} \mid 42)[0..15]$

$\text{CKY-R} = \text{SHA-1}(\text{kerberos_session_key} \mid 42)[0..15]$

5. Transmitting the KRB_AP_* Messages Inside the IKE Payload

The KRB_AP_REQ and KRB_AP_REP must be delivered from one machine to another and back inside the IKE payloads, and they cannot be encrypted, as the receiver of the packet does not yet know how to create the encryption key because it has not yet seen the KRB_AP_REQ message. Because of this we need to add a new payload to the beginning of first phase 2 payload in both direction and that must be transmitted without encryption. Because the rest of the payload is still encrypted the encryption bit in the ISAKMP header is set.

This means that we need to define new payload type for that kerberos message and if the next payload field in the generic header is that then the first payload is in clear, and encryption starts only after that payload. The whole packet is still authenticated as defined in [REVISED-HASH].

The phase 2 payload will look like this:

```

          1                2                3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
!                               Initiator                               !
!                               Cookie                                 !
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
!                               Responder                               !
!                               Cookie                                 !
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
!  NP = XXX      ! MjVer ! MnVer ! Exchange Type !      Flags      !
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
!                               Message ID                             !
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
!                               Length                                 !
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
! Next Payload  !  RESERVED  !      Payload Length      !
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
!                               Kerberos AP message                   !
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
| Encrypted phase 2 payload starts here                               |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

```

The KRB_AP_REQ payload is only added to first phase 2 payload sent out in one negotiation, i.e it is always in the first quick mode packet going out, and it is always in all notifications and delete notifications.

The KRB_AP_REP payload can only added to the first phase 2 payload sent by the responder, i.e it is always in the second quick mode packet. It is only added there if requested so. In case of kerberos error the responder sends back an message containing the KRB_ERROR message in the kerberos message payload, and optional IKE notification payload and

authentication hash payload in the body.

Note, that if the responder was not able to get the kerberos session key from the KRB_AP_REQ, it cannot decrypt and authenticate the original first message. In that case it also cannot encrypt and authenticate the error message. In that case the notification and hash payloads are not sent, and the encryption bit in the ISAKMP header is not set.

The initiator can authenticate the responder by verifying the hash payload in the second quick mode packet. Only the responder can create it because it knows the session key used to create authentication MAC. This means that KRB_AP_REP message is not usually needed.

Also because this quick mode is normally used without PFS the responder can immediately after receiving first packet instantiate inbound SA and then send reply back to initiator. This means that when the initiator receives the reply from responder it can also immediately start using the SA without need to wait for the third message to reach the responder.

The third message only gives protection against replay attacks and because IPsec keys are derived also from the nonces inside the first and second packet, any valid IPsec packet will also give proof of liveness. Thus responder can instantiate outbound SA immediately when it receives the third quick mode message, or when it receives valid authenticated IPsec packet for the inbound SA. This removes need for commit bit and reduces number of round trips from 1.5 to 1 (the last half round trip is already interleaved with the normal IPsec traffic). This optimization should NOT be used if PFS is used, because in that case instantiating the IPsec SA requires costly Diffie-Hellman operation, which should be postponed to the point where replay attacks are not possible.

6. Example Quick Mode Negotiation

Here is an example of quick mode negotiation between host A and host B.

Host A	Host B
-----	-----
HDR, KRB_AP_REQ, *HASH(1), SA, Ni, ... --->	
<--- HDR, [KRB_AP_REQ], *HASH(2), SA, Nr, ...	
HDR*, HASH(3) --->	

Where the '*' marks where the encryption begins.

7. Security Considerations

This document assumes that the session key generated by the kerberos is safe and the whole security of the IKE SA is derived from that. The IPsec SAs created using the Quick Mode negotiation over that IKE SA derive entropy also from the nonces passed in the negotiation, and also from the Diffie-Hellman if PFS is used in the quick mode.

8. References

[REVISED-HASH] Kivinen T., "Fixing IKE Phase 1 Authentication HASH",
[draft-ietf-ipsec-ike-hash-revised-01.txt](#) (WORK IN PROGRESS)

[RFC-2409] Harkins D., Carrel D., "The Internet Key Exchange (IKE)",
November 1998

[RFC-2119] Bradner, S., "Key words for use in RFCs to indicate
Requirement Levels", March 1997

[RFC-1510] Kohl, J., "The Kerberos Network Authentication Service (V5)",
September 1993

9. Authors' Addresses

Tero Kivinen
SSH Communications Security Corp
Fredrikinkatu 42
FIN-00100 HELSINKI
Finland
E-mail: kivinen@ssh.fi

