Network Working Group   M. Nakhjiri
Internet-Draft   Huawei USA
Expires: October 7, 2007   April 5, 2007

Keying and signaling for wireless access and handover using EAP (EAP-HR)
draft-nakhjiri-hokey-hierarchy-04

Status of this Memo

Copyright Notice

Abstract

Problems related to AAA-based key management for facilitating optimized secure handovers and re-authentications have been described in several problem statements ([I-D.nakhjiri-aaa-hokey-ps], [I-D.ohba-hokey-3party-keydist-ps] and [I-D.ietf-hokey-reauth-ps]). This document provides description of an EAP initiated key hierarchy as part of the solution for those problems. Additionally a modified version of the 3-party key distribution orocess ([I-D.ohba-hokey-3party-keydist-ps]) is proposed to provide a binding between the generated/distributed keys and the parties using the keys. A new EAP method called EAP handover and re-authentication (EAP_HR) is also described to significantly reduce handover keying and re-authentication latency. AAA attributes and EAP type data extensions are also covered.

Table of Contents

1. Introduction and Problem statement

It is becoming more and more common to use the Extensible Authentication Protocol (EAP) framework ([RFC3748]) for access control authentication and bootstrapping the wireless link security. This is done by performing an EAP authentication method that is capable of generating EAP master sessionkeys, MSK and EMSK ([I-D.ietf-eap-keying]), which are, in turn, used to bootstrap the so called temporary session keys, TSK) for securing the wireless link. The typically deployed model is one where EAP authentication is performed as a peer to peer protocol between the peer and a backend server, without involving much intelligence from the edge of the network. At the edge, the model only uses a logical entity called pass-through authenticator (typically simply called authenticator), which only takes part in changing the form of encapsulation of the EAP, but is otherwise passive until the very end, where the keying material for establishment of TSKs are generated from EAP master session keys and are transported to this authenticator.

Deploying this model creates a number of issues that are listed in the problem statement drafts ([I-D.nakhjiri-aaa-hokey-ps] and [I-D.ietf-hokey-reauth-ps]); for instance, the model does suffers from the inherent lack of support for fast re-authentications in EAP when peer's session is expiring. Another issue is that the way the keying material from the initial EAP authentication is distributed to the authenticators does not readily allow for optimized handovers without breaking security principles ([I-D.housley-aaa-key-mgmt]).

This document provides a description of how the EAP extended master session key (EMSK) can be used according to [I-D.ietf-hokey-emsk-hierarchy]) to arrive at a handover root key (HRK) for the current administrative domain. The root key is in turn used as root for a key hierarchy that provides a solution for fast re-authentication with the HOKEY server and/or quick and secure handover security provisioning by generating and delivering per authenticator keys (MDMSK) to the authenticators. Such per-authenticator (per-MDC) keys (MDMSK) are generated for each serving MDC as the peer attaches to the network or moves its point of attachment. Signaling to provide a channel binding mechanism that also achieves peer consent in delivery of the key to the authenticators is also described. The signaling proposed for bootstrapping the MDMSKs for MDCs in conjunction to initial network setup or handovers is EAP.

Since EAP authentication is a 2 party protocol, additional measures should be taken to properly utilize the EAP generated keys in a 3 party key management (involving peer, authenticator and server) scenarios. This involves channel binding procedures and verification

of key sources. This document provides a modified version of the procedures presented in the previous version of this draft and the solution proposed in [I-D.ohba-hokey-3party-keydist-ps].

A new EAP method, called EAP Handover and Re-authentication (EAP-HR), is proposed here to introduce minimal changes to the existing authenticator deployment base. The proposed signaling adds 1.5 round trip in case of initial network setup and approximately 1 round trip for re-authentication in conjunction with session life time expiry and authenticator (MDC) handovers. It should be noted modifications to the current EAP Identity Type signaling (to carry type data) can achieve the same effect. EAP-HR is proposed as an alternative that does not require any changes to the existing EAP Identity type messaging.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

HOKEY server: This is essentially a AAA server, that can receive a domain specific handover root key (HRK) from the EAP server or a HOKEY server and can act as an authority for authorization HOKEY service, perform re-authentiation and/or generate per-authenticator keys (MDMSK) in conjunction to network setup, handovers and re-authentication.

Mobility Domain/MDC: The terms mobility domain and Mobility Domain Controller (MDC) are introduced to allow for the architectures where an authenticator is responsible for managing a number of edge devices (Access nodes). A cluster of ANs can form a mobility domain, managed by an mobility domain controller (MDC) acting as authenticator and AAA client on behalf of all those ANs when dealing with the AAA and EAP server. The MDC would receive the master key (MDMSK) sent by the home AAA server (deploying HOKEY server) and generate master keys for the ANs it is managing, as it sees fit. More details on this architecture is provided in the appendix.

HRK: Handover root key is a key that will be used as the root key to solve the handover keying and re-authentication problem. The HRK can be derived directly from EMSK as a usage specific and domain independent specific root key for re-authentication and handover, using a Pseudo random function (PRF) that complies with requirements and guidance in [I-D.ietf-hokey-emsk-hierarchy]. For simplicity we refer to this PRF as USRK_PRF.

HHRK: Home handover root key is a key that will be used as the root key to solve the handover keying and re-authentication problem within the home AAA domain. The HHRK can be derived directly from EMSK as a usage specific and domain specific root key for re-authentication and handover, using a Pseudo random function (PRF) that complies with requirements and guidance in [I-D.ietf-hokey-emsk-hierarchy] (USRK_PRF). Alternatively, the HHRK can also be generated from HRK as usage specific but domain independent key from EMSK.

HO_PRF: The PRF that is used by the peer and the HOKEY server to derive any keys from the HRK. The HO-PRF may or may not comply with requirements specified for USRK (since the HRK and not EMSK is being used to provide entropy), and only needs to be supported by the crypto-engines at both peer and the HOKEY server. The HO_PRF can be access technology agnostic and can be pre-configured based on peer and AAA capabilities to avoid cipher suite negotiations, if desired.

IK and CK: Integrity and cipher keys, used to protect the EAP signaling between the peer and the HOKEY server.

MDMSK: Mobility Domain Master Session Key: A key derived specifically for each authenticator/MDC at the HOKEY server and at the peer MDC. This key is then used by the peer and the MDC to establish a secure network attachment link.

MDC Identifier (MDC_ID): The identifier for the MDC serving the peer. This ID must unequivocally and uniquely identify the MDC to both the peer, the EAP server (and the ANs being served by the MDC, when applicable).

3. Key Hierarchy and Generation

Upon successful completion of the EAP authentication method, the EAP server generates the EAP EMSK as defined by the EAP method that was executed. From this EMSK, an USRK designated for handover keying application can be generated, assuming that based on the user profiles, the AAA server can successfully authorize the user (peer) for use of the HOKEY (handover keying and re-authentication) service instead of EAP service. We called this usage specific root key (USRK) the HRK. Specification of generation of USRKs is work under progress ([I-D.ietf-hokey-emsk-hierarchy]), but it is expected that since the EAP layer does not export EMSK, the HOKEY server needs to, following the authorization, request derivation of the HRK from the EAP server.

It should be noted that USRKs are domain indpendent, meaning that an HRK generated for hanodver and re-authentication usage will be the root key for all domains, and a separate root key needs to be generated for each domain, e.g. a home HRK (HHRK) for home domain and a visited HRK (VHRK) for each visited domain. The USRK specification also allows generation of usage and domain specific keys (USDSRK). Thus, it is possible for a single domain operation to simply consider generating a USDSRK for handover keying within home domain (i.e. an HHRK) directly from the EMSK.

Upon request, the EAP layer that holds EMSK generates an HRK and delivers it to the HOKEY server (seen as AAA server in this document) that oversees the operation of home and visited domain HOKEY servers. The HRK is stored at this HOKEY server database, where it is fetched for generation of domain HRKs for each domain and keys for each authenticator within home domain.

Many wireless networks tend to deploy a gateway (e.g. Access Service Node gateway, ASN-GW in WiMAX architecture), that manages a cluster of edge devices (access nodes, ANs) called to form a mobility domain. The gateway tends to include AAA client functionality, DHCP server/ relay function, mobility management within the mobility domain (without dealing with AAA server) and many other functions, such as Mobile IP agent function. For the purpose of keeping our design generic to serve both two-level deployment models and the flat models, we assume the EAP authenticator, AAA client and other mobility domain functionalities required of such access gateways are embedded in one device which we call mobility domain controller (MDC). Thus we refer to the session key delivered to the mobility domain contoller as mobility domain master session key (MDMSK). As the peer moves from one authenticator/MDC to the next, its continuing operation requires delivery of new MDMSKs to the new serving MDC. Following the key hierarchy specification, the MDMSK is derived at the HOKEY/AAA server using the HRK for the serving domain. The MDMSK for each MDC is kept hidden from other MDCs.

The entire key hierarchy is shown in the following figure.

```
                                   EMSK
                                    |
                                    HRK
                    ________________|____________________________
                   |        |       |                 |          |
               VHRK1 .. VHRKn      HHRK              CK          IK
               _____________________|___________________......______
               |   |              |                 |               |
            HCK  HIK            MDMSK1            MDMSK2          MDMSKm
```

Figure 1: A keying hierarchy to support handover and re-authentications

HRK | HRK_name_Key=HRK-PRF (EMSK, Usage_label | NULL | Peer_ID | Key_length)

Where, it is assumed that HRK_PRF generates Z=X+Y bits, where the first X bits are used for HRK, while the remaining Y bits are used for HRK_name_key, which is used to create temporal uniqueness in the key name generation (see below). The HRK-PRF may be negotiated, pre-configured or chosen based on a network policy decision in a manner that is compliant with requirements defined in [I-D.ietf-hokey-emsk-hierarchy]

The "Usage_label" value is to be assigned by IANA to the string "Domain Handover Root Key Derivation".

NULL as domain label: It should be noted that for the purpose of supporting roaming the HRK is generated as a usage specific but domain independent root key (USRK) and thus a NULL has been used instead of the domain label.

Peer_ID is the identifier for the peer as known to the server generating HHRK. This identifier is exchanged in the key distribution exchange (KDE) as described shortly.

HRK_name=First (128, HMAC_SHA256(HRK_name_Key, "handover root key derivation"| peer_ID | NULL))

Where, First (N, X) refers to the first N bits of X.

The home HRK (HHRK) and visited domain HRKs (VHRK) are generated as a usage and domain specific root key (USDSRK), specifically for home domain and thus Home_domain_ID or Visited_domain_ID serve as a domain label.

HHRK | HHRK_name_Key=HO_PRF(HRK, Peer_ID | home_doamin_ID |

Key_length)

VHRK | VHRK_name_Key=HO_PRF(HRK, Peer_ID | visited_domain_ID |
Key_length)

Since HHRK and VHRK are no longer derived from the EMSK, the PRF used for generating these keys may or may need to comply with the requirements in [I-D.ietf-hokey-emsk-hierarchy] and thus we have used the notion of HO_PRF to indicate this flexibility.

HHRK_name=First (128, HMAC_SHA256(HHRK_name_Key, "domain handover
root key derivation"| peer_ID))

VHRK_name=First (128, HMAC_SHA256(VHRK_name_Key, "domain handover
root key derivation"| peer_ID))

Home_domain_ID or the Visited_domain_ID is the identifier for the home or visited domain as known to both peer and the server generating HHRK or VHRK. When roaming from one domain to the next, the peer needs to request for the domain handover root key to be generated from the HRK and exchange its own identity as well as the domain identity with the server. To protect the key distribution signaling, the peer and main HOKEY server can use a pair of domain independent integrity key (IK) and cipher key (CK), which are generated as follows.

IK | IK_name_key=HO-PRF (HHRK, "Integrity Key" | peer_ID | NULL |
Key_length)

IK_name=First(128, HMAC_SHA256(IK_name_key "Integrity Key"| peer_ID)

CK | CK_name_key=HO-PRF (HHRK, "Cipher Key" | peer_ID | NULL |
Key_length)

CK_name=First(128, HMAC_SHA256(CK_name_key, "Cipher Key"| peer_ID)

It is important that the IK and CK used for protecting the signaling in a roaming case can stay domain indepdent and thus the use of NULL instead of the domain identifier.

It is assumed that HHRK is delivered to the HOKEY server within the home domain. It should be noted that HHRK is only accessible to the peer and the HOKEY server within home domain, but not to HOKEY servers within other domains. HHRK is not accessible to any authenticators. The HHRK is then used to generate MDMSKs for the MDCs within the home domain and to generate integrity key and cipher key (IK and CK) to protect the EAP signaling between the HOKEY server and the peer. Thus, the HHRK is used to also generate the IK and CK

to protect the EAP-HR signaling to perform re-authentication with the HOKEY server or to perform the exchanges required to arrive at an MDMSK for any authenticator the peer attaches to.

```
HIK | HIK_name_key=HO-PRF (HHRK, "domain integrity Key" | peer_ID
| Home_domain_ID | Key_length )

HIK_name=First(128, HMAC_SHA256(HIK_name_key "domain integrity
Key"| peer_ID)

HCK | HCK_name_key=HO-PRF (HHRK, "domain cipher Key" | peer_ID |
Home_domain_ID | Key_length )

HCK_name=First(128, HMAC_SHA256(HCK_name_key, "domain cipher Key"|
peer_ID)
```

The Integrity and cipher keys (IK and CK) are used to protect the EAP-HR signaling between the peer and the HOKEY servers and are cached by the peer and HOKEY server within the current domain and are not exposed to any outside parties.

```
MDMSK_i= HO-PRF (HHRK, "MDMSK generation" | peer_ID |
Home_domain_ID | MDC_ID | Nonce | Key_length)
```

MDMSK_i is the key sent to ith authenticator/ MDC within the domain (in this case the home domain) by the HOKEY server within the domain. This key is then used by the peer and the MDC to establish a secure network attachment link. It should be noted that the distribution of MDMSK to the MDC needs to happen in a manner that provides proper binding between the key and the identity of the peer and the MDC, and must be delivered in a fashion that no other MDC can gain access to this key. The Nonce is added to create temporal uniqueness to avoid generation of the same key during multiple visits of the peer to the same authenticator during the same EAP session.

The HOKEY server may delete the MDMSK_i cache after transport, if required for compliance with principle of least privilege.

In cases where access nodes are involved, further key hierarchy levels may be required. This is explained in more details in the appendix.

4. 3 party Key distribution exchange (KDE)

In this section we attempt to describe how the earlier proposed key hierarchy can be used for improving the latency involved in handover and re-authentication. Using the key hierarchy would mean that an MDMSK (generated from an EMSK-based hierarchy) would have to be installed in the authenticator dealing with the peer both at the time of network setup (bootstrapping) and at the time of authenticator handover. However, as explained in [I-D.ohba-hokey-3party-keydist-ps], the EMSK is generated as a result of an EAP method, which is a two party exchange, while the resulting MDMSK must be transported to an authenticator, which is considered a third party for the initial EAP exchange. Thus, both the initial 2 parties in the EAP process, i.e. The peer and the server need to make sure that they are dealing with the same third party (authenticator) before allowing the authenticator access to the MDMSK. [I-D.ohba-hokey-3party-keydist-ps] proposes a modified version of the Otway-Rees protocol that meets the requirements for such 3-party lay-out. This document provides an slight adaptation of that proposal to carry the MDMSK from the HOKEY server to the authenticator. The description below can be carried over a generic transport and thus is independent of the exact type of protocol that is used. However for the purpose of this document the assumption is that the 3 party mechanism parameters are carried for EAP messages that are themselves encapsulated over an access technology suited transport between the peer and the authenticator and over AAA protocols between the authenticator and the HOKEY server.

The exchange proposed below is to perform a channel binding and avoid the lying NAS scenario, where the authenticator announces a down link ID to the peer (DAID) and a different uplink ID to the server (UAID). The peer uses DAID in its token towards the server, while the authenticator uses its UAID in its token to the server. Server must use the UAID from peer token to calculate the MIC in the authenticator ([PID, UAID]Kas) and if there is a match, then the server can verify that DAID and UAID are the same as the AID and proceed with generating and provisioning of MDMSK, otherwise the server MUST return a failure code instead of generating an MDMSK.

The 3 party key distribution basically consists of 1 exchange, i.e. 2 messages between the peer and the HOKEY server. However, each message traverses over two logical hops (peer-authentcator) and (authenticator-HOKEY server) and thus the exchange can be seen as 4 logical messages. It should be noted that message 0 below is adding to comply with EAP request/response state machine requirements and can be eliminated otherwise as the information in message 0 can be advertised through other means.

0 Authenticator to peer: EAP(DAID, DID)

1 Peer to Authenticator: EAP((PID, DAID, DID, Np,KNps), [PID, DAID, Np,KNps]Kps)

2 Authenticator to Server: AAA(PID, UAID, [PID, UAID]Kas), AAA(EAP((PID, DAID, DID, Np,KNps), [PID, DAID, DID, Np,KNps]Kps))

3 Server to Authenticator: AAA({PID,AID,KNpa, KLpa, Kpa}KEas), AAA(EAP(KNpa, KLpa, KNps, [PID, AID, DID, Np+1,KNpa,KLpa, KNps]Kps))

4 Authenticator to Peer: EAP(KNpa, KLpa, KNps, [PID, AID, DID, Np+1, KNpa,KLpa, KNps]Kps])

the notation is as follows:

PID: peer ID. The information is expected by carried by an existing attribute within EAP and/or AAA protocols.

DID: Domain ID, used for generation of domain specific keys, such as HHRK (see key generation).

AID: Authenticator ID (obtained by the peer through beacon announcements or as part of EAP Identity Request)

DAID: Authenticator ID as perceived by the peer (down link ID)

UAID: Authenticator ID as perceived by the server (uplink ID)

{X}K: X encrypted with key K

[X]K: Message Authentication Code over X with key K.

X(Y): Y carried with X protocol

Kps: A symmetric key shared between peer and Server for signing (IK, provisioned by EAP/HOKEY hierarchy) and identified by KNps.

KEps: A symmetric key shared between peer and Server for encryption (CK, provisioning by EAP/HOKEY hierarchy) and identified by KEpsid

KEas: A symmetric key shared between authenticator and Server for encryption (provisioned out of band).

Kas: A symmetric key between authentication and server for MDCs for signing (provisioned out of band).

Kpa: A symmetric key to be shared between peer and authenticator (key to be distributed to authenticator, the MDMSK in this document). The key is named as KNpa.

KLx : Key lifetime for key x

KNx: Key name for Key x, e.g. KENas: key name for KEas

Nx : Nonce provided by the party X

(PID,DAID, DID, Np), [PID, DAID, DID, Np]Kps is called the peer request token (PRT), which carries the identities of both peer and authenticator along with a signature. The signature is called the peer request authenticator (PRA).

(PID, UAID, [PID, UAID]Kas) is called Authenticator_ID_token (AIT), which carries a signature, called Authenticator ID authenticator.

{PID, AID, KNpa, KLpa, Kpa}KEas is called Authenticator_key_token (AKT), which carries the Kpa wrapped with KEas (encryption key shared between authenticator and server).

KNpa, KLpa, KNps, [PID, AID, DID, Np+1,KNpa,KLpa, KNps]Kps is called Server_authorization_token (SAT).

5. Signaling using EAP-HR and AAA

The exchange for key distribution to a 3th party after an initial two-party authentication was explained above. Our intention is to show that is possible to perform the 3-party key distribution exchange (KDE) with maximum 1 1/2 round trip and in most cases with 1 round trip by using an EAP method, called EAP-HR. The method simply consists of an EAP-HR request/response and completes with an EAP success. In cases where the EAP-HR request starts from the authenticator, the number of round trips is almost 1 (EAP-HR response + EAP Success), since the authenticator-peer trip time is negligible compared to server-peer trip time. in cases where EAP-HR request has to start from the server, the number of round trips would be 1 1/2.

5.1. signaling scenarios

5.1.1. Network setup scenario

The network setup is the scenario when the peer is attempting to attach to the network for the first time. The assumption is that the peer performs an EAP mutual authentication with the EAP server at the home domain and establishes MSK and EMSK. A HOKEY compliant peer and

server peer will now follow the HOKEY key hierarchy, which provides per-authenticator keys (MDMSK) to the authenticator the peer is attaching through. A HOKEY compliant server, may be able indicate its compliance to the peer by setting the value session-life-time attribute to zero, when this attribute is communicated to the authenticator. In a typical deployment, authenticator keys and session life time are carried by the same AAA messages that carry the EAP success ([RFC3579] and [RFC4072]). This will trigger the authenticator to start the 3-party key distribution process (KDE) by sending an EAP-HR request. In cases, where sending trigger to the authenticator is not possible, the EAP-HR request needs to be initiated from the server, adding a 1/2 round trip to the exchange. However, this additional delay would not be critical since no peer applications has started yet.

As one can see below, an EAP-HR request can be used to carry authenticator ID (DAID) and the domain identifier and can trigger the 3-party key distribution exchange (KDE) at the peer.

The peer will create the PRT token and starts the KDE by sending the content of KDE message 1 as type data within EAP-HR response.

The authenticator forwards the EAP-HR response within a AAA request message, while the contents of message 1 within the EAP attribute and the rest within AAA AVPs.

The server after calculating the data required for KDE message 3 will include the data as explained earlier, party within the EAP-Success embedded within the AAA response messages and partly as AAA AVPs within the same AAA messsage. The AVPs will include an encrypted copy of the MDMSK calculated for the authenticator

The authenticator extracts the AVPs and the EAP-Success message and forwards the EAP success message as described in KDE message 4 to the peer.

```
      peer           Authenticator               server
      -----             --------                 -------
        |             EAP-Authentication            |
        | <------------------------------------->   |
        |       EAP-Success (lifetime=0)            |
        | <-----------------------------------------|
        |EAP-HR/Req (KDE 0) |                       |
        |   <---------------|                       |
        |    EAP-HR/Res (KDE 1 and 2)               |
        |  ------------------------------------->   |
        |              EAP-Success (KDE 3 and 4)    |
        | <-----------------------------------------|
```

Figure 2: Network Setup using EAP-HR

5.1.2. Authenticator handover and/or re-authentication scenarios

Both authenticator handover and re-authentication scenarios can follow the same signaling process for the following reasons. In case of handover, the peer as a result of scanning process, detects the new point of attachment and the new authenticator and sends a request for attachment/ association to the new authenticator through the lower layer. This can trigger the EAP-HR process at the new authenticator (assuming reactive handover keying). In case of re-authentication, the authenticator based on its configuration state, or through lower layer request from the peer (if peer is aware that its keys are expiring) knows that the peer needs to extend its existing session and keys and thus can act based on internal triggers.

```
              peer           Authenticator               server
              -----             --------                 -------
              |EAP-HR/Req (KDE 0) |                        |
              |   <---------------|                        |
              |    EAP-HR/Res (KDE 1 and 2)                |
              |  ------------------------------------->    |
              |              EAP-Success (KDE 3 and 4)     |
              | <------------------------------------------|
```

Figure 3: Handover using EAP-HR

It should be noted that proactive signaling is desired and may be possible in some cases. Proactive signaling would mean the peer and/or authenticator start the acquisition of the MDMSK prior to completion of the link handover to the new authenticator. The 3 party process could only happen if the peer has a link with the new

authenticator (direct or indirect through the new or previous point of attachment). This is since the signaling needs to go through the new authenticator so that it can provide its own identity to the server. We do not discuss proactive keying in the current version of this document.

5.2. AAA and EAP extensions

The intent is to use EAP signaling between the peer and server. However, as it is customary, the authenticator-server portion of EAP signaling is carried over a AAA protocol between the authenticator and the server. We do not go into the details of the choice the use of the AAA protocol and simply provide a generic description of the attribute value pairs (AVP) that are needed to accomplish the exchanges required for key derivation and distribution.

5.2.1. AAA AVPs

This section provides a list of the new AVPs that may be required for the KDE.

Authenticator_DID_AVP: Format to follow RADIUS attribute or Diameter AVP syntax. In Diameter, this may be part of a grouped attribute carrying the rest of peer request token information.

This AVP is to carry the MDC ID (DAID) reported to the key generating server (HOKEY server) through the peer.

Authenticator_UID_AVP: Format to follow RADIUS attribute or Diameter AVP syntax

This AVP is to carry the MDC ID (UAID) reported to key generating server directly. A new Authenticator_UID_AVP may not be required since, the MDC may simply act as a NAS and thus the uplink identifier of the MDC is the same as the NAS_ID, which is a well-known RADIUS/ DIameter attribute.

Domain_ID_AVP: Format to follow RADIUS attribute or Diameter AVP syntax

This AVP is to carry the domain identifier used in generatio of domain specific keys.

MDC_ID_AVP: Format to follow RADIUS attribute or Diameter AVP syntax

This AVP is to carry the MDC ID from the server to the peer in return AAA message. This is to indicate to the peer what ID was used by the server to generate the MDMSK. When MDC is the same as NAS, the well known NAS_ID can be used.

Authenticator_ID_token_AVP: Format to follow RADIUS attribute or Diameter AVP syntax.

This AVP is to carry the Authenticator_ID_token, which includes peer and authenticator uplink identities and a signature, signed by the authenticator (i.e. PID, UAID, [PID, UAID]Ka). In case of RADIUS, where grouped attributes may not be supported, the signature portion of the token needs to be carried as a separate attribute.

Authenticator_KEY_AVP: Format to follow RADIUS attribute or Diameter AVP syntax.

This AVP is to carry the Authenticator_key_token (AKT), including MDMSK from the server to the authenticator. In Diameter a grouped attribute can carry the entire Authenticator_key_token (i.e. Including the key name). When attribute grouping is not allowed (e.g. Current state of RADIUS attributes), information such as KLpa and KNpa need to be carried in separate attributes.

5.2.2. EAP-HR and EAP-HR Type data fields

The following data chunks are to be carried by EAP-HR type signaling as EAP type data. They may be carried as part of type data for the new type EAP-HR, or for an EAP-Identity if the IETF decides to allow addition of type data to EAP Identity messaging. In the following we are going to follow the design based on the assumption of use of EAP-HR.

```
     0                   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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Code      |  Identifier   |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |     Flags     |    Subtype    |KDE Data payload
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            KDE Data payload
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: EAP-HR Request and Response packet formats

Code:

1 and 2 for EAP-HR Request and Response

3 for EAP Success

Identifier: in case of EAP-HR Requests and Responses is implemented as typical for EAP Request and response pairs.

Flags: TBD

Type X for EAP-HR (X to be assigned by IANA)

Subtype: KDE data SubtypeTo indicate the type of KDE data

0 DID (used in EAP-HR Request and Response)

1 DAID (used in EAP-HR Request and Response)

2 PRT (Peer_Request_Token) used in EAP-HR Response

Notes:

KDE data stands for 3 party key distribution exchange data and is implemented as EAP-HR Type data (as is conventional for EAP methods) is defined as follows

DID (Domain_ID): carrying the domain identifier required for key derivation.

DAID (down link Authenticator ID) carrying the authenticator ID seen by the peer.

PRT is to include (PID, DAID, Np,KNps), [PID, DAID, Np,KNps]Kps) from the peer to the authenticator and server.

The proposed EAP-HR method terminates by an EAP-Success sent from the server to the peer through the authenticator. The proposal includes modification of the EAP Success message to carry additional data.

```
   0                   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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Code      |  Identifier   |            Length             |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Type      |    Flags      |    Type payload
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |            Type payload
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Modified EAP-Success

Code: 3 for EAP Success

Identifier As typical for EAP success

Type Type of data the EAP Success is carrying (it is possible to map the Type space to cover the EAP method types as well, i.e.

Type 0-X as assigned for EAP methods by IANA

Type X+1-254 for additional type of data to be carried by EAP Success. Here we assume type Y belongs to the allowed type space

Y SAT (Sever_Authorization_token)

SAT to include KNpa, KLpa, KNps, [PID, AID, Np+1,KNpa,KLpa, KNps]Kps carried from the server to the peer through the authenticator..

5.2.3. backward compatibility with EAP

For simplicity we call an entity (server or peer) that implements HOKEY, hokey-compliant. The same entity is "hokey incompliant" if it only supports EAP keying. A AAA server that implements HOKEY a HOKEY server.

Hokey-compliant peer and server can use handover root key (HRK) to generate per-MDC keys (MDMSK). A "hokey compliant" server transports the MDMSK to the MDC/ authenticator. On the other hand, hokey-incompliant (legacy) peers and servers use MSK instead of both HRK and MDMSK. Authenticators may not require much change to comply with the new key hierarchy, except to be able to send EAP-HR if needed. It is also possible to use EAP identity Request/Response instead of EAP-HR to achieve the same effect, but assuming EAP identity messages can modified to carry type data.

The other impact is that EAP Success will have to carry additional data, but the legacy authenticator should be oblivious to this, since authenticator will normally carry EAP Success to the peer and receives its own key material through AAA attributes (outside EAP Success) anyhow.

When "hokey compliant" MDC and AAA server are dealing with a "hokey non-compliant" peer, the peer will simply not understand the EAP-HR request and responds accordingly, to which the HOKEY server responds by sending the MSK as done in EAP keying framework. However, the HOKEY server should through accessing user profile be able to tell whether the peer is hokey compliant or not.

6. Security report Card

This section of the document provides a test of the provided key hierarchy against the security goals stated in the handover keying problem statement draft [I-D.nakhjiri-aaa-hokey-ps]

Key Context and scope, prevention of domino effect: The context and scope for each key is clearly defined by including the identities of the parties that are to share the key and by including the purpose of the key in the key generation.

Key Naming and freshness: All keying material starting from MDMSK and the derivatives are uniquely named, using the identity of the parties sharing the key. EAP session ID, when available can be used to provide freshness and key name uniqueness. However, since generation of EAP session ID is optional for many methods, we have opted for use of cryptographically generated key names using the freshness properties of the parent key.

Authentication of all the parties: The key distribution mechanism, described provides authentication of all parties to each other.

Channel binding: The key distribution mechanism proposed provides proper binding of the key to the parties that will use it.

EAP method independence: The key hierarchy in this document stems from the EAP method generated keys (MSK and EMSK). As long as the method is capable of creating EMSKs, this hierarchy is method independent.

7. Security Considerations

The key distribution mechanism described in this document relies on the a pre-established trust infrastructure that wraps and delivers the MDMSK from the server to the authenticator through the AAA architecture. Any vulnerabilities arising from AAA infrastructure insecurity, e.g. existence of transitive trust relationships will directly impact the security of this key delivery mechanism and thus the privacy of the peer, the network and their link. Furthermore, since the entire hierarchy is generated from the EMSK, using PRFs that are possibly dictated by network policy, the cryptographic strength of all such child keys will depend on the strength of the EMSK and the PRFs that are used. Furthermore, life time and caching of these keys are determined by the network policy and implementation and can introduce additional vulnerabilities and targets for attacks.

8. IANA consideration

This document defines a new EAP method type, along with associated type data as well as number of new AAA AVPs. The values for these types and AVPs need to be assigned by IANA. It may also require a number of new Diameter command codes, if Diameter is used. In that case, allocation of new command codes needs to be done through IANA as well.

9. Acknowledgements

The author would like to thank Jari Arkko for useful suggestions on generation of the handover key hierarchy and Mohan Parthasarathy and Hao Zhou for discussions and feedback.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[I-D.ietf-eap-keying]
Aboba, B., "Extensible Authentication Protocol (EAP) Key Management Framework", draft-ietf-eap-keying-18 (work in progress), February 2007.

[I-D.nakhjiri-aaa-hokey-ps]
Nakhjiri, M., "AAA based Keying for Wireless Handovers: Problem Statement", draft-nakhjiri-aaa-hokey-ps-03 (work in progress), June 2006.

[I-D.ietf-hokey-reauth-ps]
Clancy, C., "Handover Key Management and Re-authentication Problem Statement", draft-ietf-hokey-reauth-ps-01 (work in progress), January 2007.

[I-D.ietf-hokey-emsk-hierarchy]
Salowey, J., "Specification for the Derivation of Usage Specific Root Keys (USRK) from an Extended Master Session Key (EMSK)", draft-ietf-hokey-emsk-hierarchy-00 (work in progress), January 2007.

[I-D.ohba-hokey-3party-keydist-ps]
Ohba, Y., "Problem Statement and Requirements on a 3-Party Key Distribution Protocol for Handover Keying", draft-ohba-hokey-3party-keydist-ps-01 (work in progress), March 2007.

[RFC3579] Aboba, B. and P. Calhoun, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)", RFC 3579, September 2003.

[RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.

10.2. Informative references

[I-D.housley-aaa-key-mgmt]
Housley, R. and B. Aboba, "Guidance for AAA Key Management", draft-housley-aaa-key-mgmt-09 (work in progress), February 2007.

Appendix A. Appendix A: Support of handovers within a mobility domain (intra-authenticator handovers)

This appendix intends to provide some suggestions on how to deploy the handover keying mechanisms within a typical wireless network architecture, where the authenticator is implemented inside an access gateway node that manages a mobility domain, consisting of a number of access nodes. The access nodes terminate the wireless link with the peer, but the mobility across many access nodes is managed by the same mobility domain controller (MDC). In such cases the MDMSK transported to the authenticator/MDC is used to create master keys (LSAP_MK) for the link security association protocol (LSAP) that is run between the peer and the AN to produce keys that secure the wireless link between the peer and AN. Again the goal is to provide each AN with its own LSAP_MK that is cryptographically separate from the LSAP_MK that may be provided to the neighbor ANs for dealing with the same peer during the same session. To reduce the handover latency, it is desired that the inter-AN handovers within the same mobility domain are handled at MDC level without interaction with the AAA or HOKEY server. However, since in many architectures, the MDC and ANs are physically disjoint, distribution of LSAP_MK to the AN from the MDC will suffer from the same 3-party key distribution issues as those discussed for MDMSK distribution. A similar approach can be applied to transport keys from the MDC to the AN. However the AAA protocols cannot be used in this case, since the MDC is a AAA client.

```
                                  EMSK
                                      |
                                  HRK
            ________________|_________________________________
            |        |             |                 |      |
            VHRK1 .. VHRKn      HHRK                 CK     IK
            _____________________|___________________......______
            |   |              |                 |               |
           HCK HIK            MDMSK1            MDMSK2          MDMSKm
            ______________________|__________________________
            |         |                       |             |
            MDC_CK    MDC_IK               LSAP_MK1     LSAP_MK2
```

Figure 6: A keying hierarchy to intra-authenticator handovers

Author's Address

Madjid Nakhjiri
Huawei USA

Email: mnakhjiri@huawei.com

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).