3GPP TSG SA WG3 Security#33 **S3-040321**
10 – 14 May, 2004
Beijing, China

Agenda Item: **6.9.4 (GAA/HTTPS)**

Source: **Nokia, Siemens**

Title: **Removal of Annex B of TS 33.222 – Pseudo-CR**

Document for: **Discussion and decision**

Abstract

In the current version of the Access to NAF using HTTPS specification (TS 33.222 v100), the informative annex B contains an optimised sequence of events in case of collocated BSF and NAF. It is proposed to eliminate this annex B completely.

1. Reason for proposed change to TS 33.222 v100

The content of annex B is no longer in line with the general trend for GBA during the last SA3 meetings. The following points are not clarified or even in contradiction:

- *Selection of procedure*: it is unclear how the involved entities decide, which procedure to run, the normal GBA procedure, or the optimised sequence.
- *Key derivation:* this is not adequately taken into account, cf. editor's note on the transport of of key derivation information from NAF/BSF to UE.
- *Transaction Identifier*. Is not usable in optimised sequence. This would preclude multiple key derivation.
- *Possibly limited lifetime of TLS connection*. A TLS connection may be terminated e.g. due to cache overflow. It is unclear what is to happen in this case.
- *Protocol variation*: As the proposed optimised sequence deviates from the protocol specified in the normative part of this specification, a normative part would be necessary for this sequence.
- *Status of draft-torvinen-http-digest-aka-v2:* the optimised sequence is based on this Internet draft. However, it seems that this draft has expired.

This CR does not preclude a physical collocation of BSF and NAF. It only removes the optimised sequence of events which would lead to an unnecessary complexity and variety of GBA protocol versions.

The next section contains a pseudo-CR to TS 33.222 v100, implementing the changes proposed in this section.

2. Pseudo-CR

********** begin change **********

5.3 Shared key-based UE authentication with certificate-based NAF authentication

This section explains how the procedures specified in [3] have to be enhanced when HTTPS is used between a UE and a NAF. The only enhancement required is the need to specify how the set up of a TLS tunnel is included in the general procedures specified in [3].

Editor's note: The sequence of events needs to be updated to reflect the initiation of bootstrapping as described in TS 33.220, section 4.3.1.

When the UE accesses a NAF, with which it does not yet share a key, then the sequence of events is as follows:

1. the UE runs http digest aka [11] with the BSF over the Ub interface.

2. If the BSF has no authentication vectors for the UE it fetches authentication vectors from the HSS over the Zh interface.

After the completion of step 1), the UE and the BSF share a secret key. This shared key is identified by a transaction identifier supplied by the BSF to the UE over the Ub interface key, cf. [3, section 4.3.1].

3. The UE establishes a TLS tunnel with the NAF. The NAF is authenticated to the UE by means of a public key certificate.

Editor's note: TLS needs to be profiled in an appropriate section of this specification.

4. The UE sends an http request to the NAF.

5. The NAF invokes http digest [10] with the UE over the Ua interface in order to perform client authentication using the shared key agreed in step 1), as specified in [3, Annex A].

Editor's note: bullet 5 references Annex A in TS 33.220, which is informative.

6. While executing step 5), the NAF fetches the shared key from the BSF over the Zn interface, as specified in [3, Annex A and section 4.3.2].

7. After the completion of step 4), UE and NAF are mutually authenticated as the TLS tunnel endpoints.

The UE may now run an appropriate application protocol with the NAF through the authenticated tunnel.

When the UE accesses a NAF, with which it already shares a key, steps 1), 2), 5) and 6) may be omitted, as specified in [3].

~~Editor's note: the above procedure is generally applicable and conforms to [TS 33.220]. For the case of a co-located BSF and NAF an optimisation is possible which is currently located in the informative Annex Z. SA3 still needs to decide whether the material in the annex should be moved to the main body, or remain in an informative or normative annex, or be deleted.~~

********** end change **********

********** begin change **********

~~Annex B (informative): Optimised Sequence of Events for Access to co-located BSF and NAF via HTTPS~~

~~Editor's note: SA3 still needs to decide whether the material in the annex should be moved to the main body, or remain in an informative or normative annex, or be deleted.~~

~~Editor's note: the material in this annex is based on the information flow in S3-030371, Annex A.~~

Editor's note: The impact on implementation when co-locating BSF and NAF is for further study.

Editor's note: The sequence of events needs to be updated to reflect the initiation of bootstrapping as described in TS 33.220, section 4.3.1.

When the UE accesses a NAF, and the NAF is co-located with the BSF, then the optimised sequence of events is as follows:

1. The UE establishes a TLS tunnel with the NAF. The NAF is authenticated to the UE by means of a public key certificate.

Editor's note: TLS needs to be profiled in an appropriate section of this specification.

2. If the UE does not share a key with the NAF, the UE sends an http request to a NAF, containing the UE's identity.

3. If the NAF receives an http request from the UE without an Authorization header, or with an Authorization header it does not accept, the NAF contacts the (co-located) BSF to obtain a challenge and a password, computed from an AKA authentication vector according to [draft-torvinen-http-digest-aka-v2].

4. If the BSF has no authentication vectors for the UE it fetches authentication vectors from the HSS over the Zh interface.

5. The NAF replies to the UE by sending a 401 "unauthorized" message with a WWW-Authenticate header according to [draft-torvinen-http-digest-aka-v2].

6. The UE sends an http request to the NAF with an Authorization header according to [draft-torvinen-http-digest-aka-v2].

7. The NAF verifies the Authorization header.

After the completion of step 7), UE and NAF are mutually authenticated as the TLS tunnel endpoints.

8. The NAF replies to the http request returning the requested information to the UE, if any.

The UE may now run an appropriate application protocol with the NAF through the authenticated tunnel.

Editor's note: the transport of of key derivation information from NAF/BSF to UE needs further study.

Note on co-location of BSF and NAF: a BSF and a NAF may be combined on one machine in such a way that the BSF is accessed through http, not using TLS, and the NAF is accessed through https. From a functional point of view, this case is identical to the general case described in section 4.2. It is even possible to functionally duplicate the BSF on one machine in such a way that the BSF is accessed through http, when TLS is not required, and accessed through https, when access to the NAF requires TLS.

Editor's note on carrying identities: the first http request after TLS set-up needs to contain the identity of the UE. The reason is that for http digest the server can issue a challenge without knowing the client's identity, whereas for http digest aka the challenge is specific to a particular client. There seem to be at least two solutions for this:

a) use a specially formed http GET request, as described for the Ub interface in [TS33.220].

b) use an Authorization header with dummy values (to be defined). The server will not accept the credentials, and will reply with a 401 "unauthorised". For maximum harmonisation, the UE identity, which needs to be included by the UE at the start of the http digest aka protocol run, should be carried in the same way in the general and the optimised case.

Note on tunnelled authentication and the use of http digest aka:

In this annex and in section 4.2 respectively, different versions of http digest aka are used. This prevents man-in-the-middle attacks with tunnelled authentication. Version 1 of http digest aka [11] is used between the UE and the BSF when http digest aka is NOT used to authenticate the client endpoint of a TLS tunnel extending between UE and BSF. Version 1 may be run inside or outside a TLS tunnel, as long as it is not used for client authentication. Version 2 [draft-torvinen-http-digest-aka-v2] is used when http digest aka IS used to authenticate the client endpoint of a TLS tunnel. Version 2 is always run inside a TLS tunnel.

[Editor'Note on tunnelled authentication and the use of http digest aka:

~~Instead of using different versions of http digest aka to distinguish whether http digest aka is used for client authentication of a TLS tunnel or not, this distinction could be provided by different means. Possibilities suggested on the SA3 mailing list include to- extend the specification of http digest akav2 to include a "situation" (or "context") parameter in the computation of the password, then always use http digest akav2, but with different values for the "situation" parameter for the two different uses.]~~

~~Note on transaction identifiers: the general approach, as specified in section 4, which is based on [3], requires the use of a transaction identifier over the interfaces Ua, Ub and Zn. The use of such a transaction identifier is neither possible nor necessary in the optimised case described in this annex~~

**********end change **********