3GPP TSG SA WG3 Security — adhoc meeting **S3z030011**
3 - 4 Sept 2003
Antwerp, Belgium

Agenda Item: **GAA**

Source: **Siemens**

Title: **Generic Authentication Architecture evaluation**

Document for: **Discussion and Decision**

1. Introduction

At SA3#29 it was agreed to start work towards an generic authentication architecture(GAA) common to all Rel-6 and future features, cf. also section 2. It may also be useful to apply the GAA to features defined outside 3GPP (e.g. by OMA). It remains to be decided by SA3 to which of these features a GAA should actually apply in which way.

It is assumed that entities involved in the provision of a feature, typically a client implemented on a UE and a server implemented in some network node, need to mutually authenticate. The GAA is to enable these entities to perform this task even if they do not share a previous security association. There are two fundamentally different ways in which a GAA can achieve this:

1) the GAA provides the involved entities with a shared secret key;

2) the GAA provides the involved entities with certificates.

It is a basic decision which still needs to be taken by SA3 whether the scope of the GAA shall include approach 1) or also approach 2). In this paper, we only consider approach 1). But approach 1) includes the support for subscriber certificates, and the question which needs to be decided is, whether the **use** of subscriber certificates for authentication should be considered within the scope of the GAA or not. It is assumed that the shared secret key is bootstrapped from the security association shared between USIM and HSS, as part of a 3G subscription.

For a detailed list of requirements on a GAA, see the companion contribution on GAA requirements.

This contribution analyses alternatives for providing a GAA. Other alternatives may be possible.

2. State of discussion

SA 3 work:

Different security solutions have been proposed for the Release 6 features. We briefly go through them here.

3G-WLAN interworking - network access: the proposed solution is the use of EAP-SIM or EAP-AKA within the IEE 802.1X framework, cf. draft TS 33.234, latest version in S3-030333.

3G-WLAN interworking - UE initiated tunnelling: no contributions have been made in SA3 so far on this issue, but it is clear that security solutions are needed as SA2 have done extensive work on this issue, cf. draft TS 23.234 and contributions to the last SA2 meeting.

Support for subscriber certificates (SSC): during the work on this issue, it was recognised that a much wider scope than just the support for subscriber certificates was required, so the work is now on

"Bootstrapping of application security using AKA and support for subscriber certificates", cf. draft TS SSC, latest version in S3-030317.

Presence (Ut reference point): four solutions have been proposed here:

1) A solution based on IMS registration [1], S3-030413.
2) A solution based on the generic Bootstrapping of application security using AKA (as in TS SSC), cf. S3-030359.
3) A solution which combines key management via http digest akav2 with the use of a reverse http authentication proxy (AP), cf. S3-030359 and S3-030371.
4) The use of subscriber certificates in TLS client authentication, cf. S3-030397.

MBMS: a solution very similar to solution 3) for the Ut reference point has been proposed, cf. S3-030367.

It has to be discussed further by SA3 whether the security needs of all the above features can be addressed by a GAA, or whether particular solutions or optimisation are necessary or considered advantageous. In particular, it is currently unclear whether and how a GAA should be applied to secure network access in 3G-WLAN interworking. The latter is different from the other cases in that it uses EAP messages within the IEEE 802.1X framework, and it is not obvious, how http could be used, whereas in the other cases, http based protocols could be used, but not EAP.

From its scope the work on "Bootstrapping of application security using AKA and support for subscriber certificates" seems the prime candidate for a GAA.

Other 3GPP groups: Both SA2 (in S3z030004, received at the SA3 ad hoc in Sept 2003) and CN4 (in S3-030337, received at SA3#29) have expressed their concern about a proliferation of authentication methods in general and the synchronisation problems of authentication vectors in particular, which may come with it. SA3 replied to CN4 in S3-030473 to address the latter problem as part of a GAA.

It is stressed here that the solution of the bootstrapping problem (provision of a shared secret key) is largely independent of the use of a reverse http authentication proxy. Characteristics of an authentication proxy are briefly mentioned in section 3, open issues relating to an authentication proxy are listed in section 4, whereas the relation of the use of AP to bootstrapping of shared keys is handled in section 5. Section 6 discusses some architectural alternatives for bootstrapping of shared keys, with and without the use of an authentication proxy.

3. Advantages of a reverse http proxy

The purpose of this chapter is to collect all non-security issues involved in using a reverse http proxy. This type of proxy is transparent for the client, in contrast to using outgoing proxies whose addresses need to be configured at the browser.

The authentication proxy described in S3-030367 and 371 is used as a reverse proxy, which means that from the Application Server (AS) point of view the proxy filters all incoming http traffic. Firewalls are typically co-located to the proxy and take a policing function to prohibit direct http addressing to destinations (urls) that should not be directly addressed.

While the proxy has to view all http traffic, its availability and performance is crucial. Any malfunction due to unavailability will impact all services connected through it. Typically the high availability may be guaranteed by using a load balancer between multiple servers. This, however, need not mean multiple interfaces towards the HSS as the multiple server can be seen as one machine from the HSS. In fact, multiple interfaces need to be avoided according to the HSS guidelines in S3-030460 .

---

[1] Siemens would be prepared to pursue this variant not any further PROVIDED a more generally applicable security solution for the Ut reference point can be found in the context of a GAA.

SA3#29 agreed that the amount of TLS connections could be limited by using a reverse http proxy. A TLS connection will be established between the UE and the reverse proxy (and terminated at the reverse proxy), instead of having many TLS connections starting at the UE towards multiple AS. This therefore provides an advantage from the terminal point of view.

The use of reverse proxies is common practice for access to AS via Internet as it goes together with firewalls and processing enhancements by caching. Such proxies may already be deployed by operators for these reasons for Pre-Rel6 applications that are accessible via the PS domain.

The aforementioned advantages suggest that a GAA should be designed in such a way that it is compatible with the use of reverse http proxies.

4. Open issues related to authentication proxies

However, there are still a number of open issues to be solved:

- how to prevent application layer identity spoofing if the security protocol is e.g. TLS. A protocol signalling the authenticated identity from the AP to the AS needs to be defined (already mentioned in S3-030371);
- whether the same AP can or should be used for different http-based applications, e.g. Ut interface and MBMS (e.g. different frequencies of update of key used for client authentication);
- how and how often a refresh of the TLS security context can be done, and who can/should initiate it? (a refresh is needed as keys in the UE are vulnerable to some extent);
- how network certificates are handled;
- requirements regarding E2E security between AS to UE need to be clarified. E2E security has to be provided on top of http (in addition to TLS) if required;
- the enhanced Cx-like interface between AP and HSS needs to be specified;
- http digest akav2 needs to be finalised at the IETF;
- replication of APs (e.g. due to load balancing or the use of different APs for different applications) should not lead to multiple interfaces towards the HSS;
- use of APs for http-based services and BSF for non-http-based services increases number of interfaces towards the HSS.

5. Relation of authentication proxies and bootstrapping of shared keys

A TLS tunnel is established between a UE and an AP. TLS is used to provide server authentication through server certificates as well as confidentiality and integrity for application traffic, but client authentication is provided through a variant of http digest. Hence, client authentication requires the establishment of a shared secret between UE and AP. How this shared secret is provided to UE and AP is the subject of bootstrapping and may be performed in a variety of ways. The problem of bootstrapping a shared secret between a UE and an AP is identical to the bootstrapping of a shared secret between a UE and an application server (AS). Some of these alternatives for bootstrapping may have the advantage that they easily fit in a generic architecture, others may be optimised for a particular situation, cf. section 6.

It can be seen from the above that the use of http authentication proxies (AP) and the method to bootstrap shared keys are orthogonal problems. An AP can be used (or not), irrespective of the choice of the

bootstrapping method. In the next section, we therefore concentrate on alternatives for bootstrapping shared keys, both with and without the use of an authentication proxy.

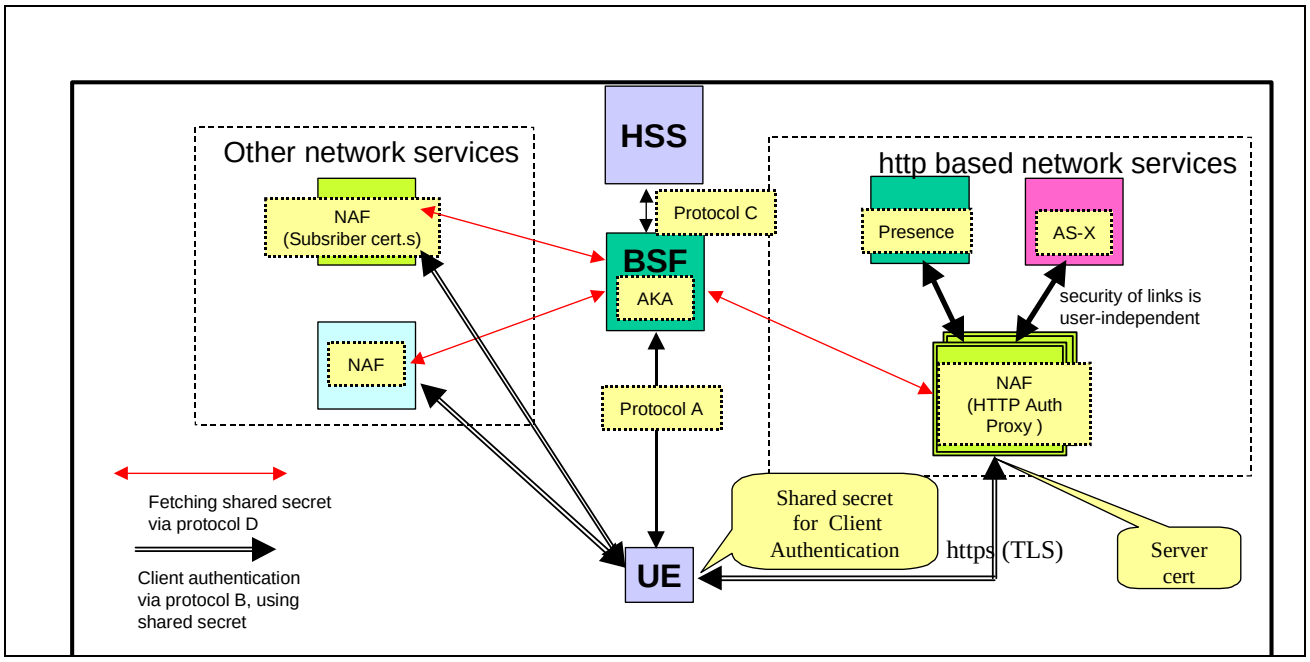
It is important to note here that client authentication has to be performed after the TLS tunnel has been set up, so that the server can be assured that whatever comes through the TLS tunnel later was sent by the authenticated client. A number of other precautions have to be satisfied as well, in particular to prevent MitM attacks.

The use of tunnelled authentication protocols has so far not been taken into account in the work on draft TS SSC. So, if the latter is to be taken as a basis for a GAA, it is to be extended to take tunnelled authentication into account.

6. Architectural alternatives

6.1 General approach using bootstrapping of application security (as in TS SSC) and http authentication proxies (APs).

The bootstrapping of application security presented in this subsection conforms to the draft TS on support for subscriber certificates (TS SSC). In this section, the AP plays the role of a NAF as defined in TS SSC.



When the UE wants to access one of the application servers, which are attached to the AP, on the right hand side of the figure, then the sequence of events is as follows (overview):

1) the UE starts http digest aka (rfc3310, protocol A) with the BSF. The BSF may contact the HSS to fetch authentication vectors (protocol C). After step 1), the UE and the BSF share a secret key, cf. TS SSC, section 4.3.1.

2) The UE sends an http request towards an application server. The http request is intercepted by the http authentication proxy (AP). The UE establishes a TLS tunnel with the AP. The AP is authenticated to the UE by means of a server certificate.

3) The UE runs http digest (rfc 2617, protocol B) with the AP = NAF to perform client authentication using the key agreed in step 1), as described in S3-030357 (a pseudo-CR to TS SSC presented at SA3#29). In the process, the AP fetches the agreed key from the BSF (protocol D), as described in TS SSC, section 4.3.2.

4) The UE runs the application protocol with the NAF.

This sequence of events conforms to TS SSC. But the set up of the TLS tunnel, which is not part of TS SSC, had to be added here. As long as the BSF and the AP = NAF are distinct entities, this sequence of events is fine. But when they are co-located then this sequence is no longer optimally efficient (cf. subsection 6.2).

When the UE wants to access one of the application servers (NAFs) on the left hand side of the figure, which is not attached to the AP, then the sequence of events is as follows (overview):

1) the UE starts http digest aka (rfc3310, protocol A) with the BSF. The BSF may contact the HSS to fetch authentication vectors (protocol C). After step 1), the UE and the BSF share a secret key, cf. TS SSC, section 4.3.1.

2) The UE sends a request (e.g. an http request) towards the application server (NAF).

3) The UE runs protocol B with the NAF using the key agreed in step 1) (e.g. http digest = rfc 2617 to perform client authentication, as described in S3-030357). In the process, the NAF fetches the agreed key from the BSF (protocol D), as described in TS SSC, section 4.3.2.

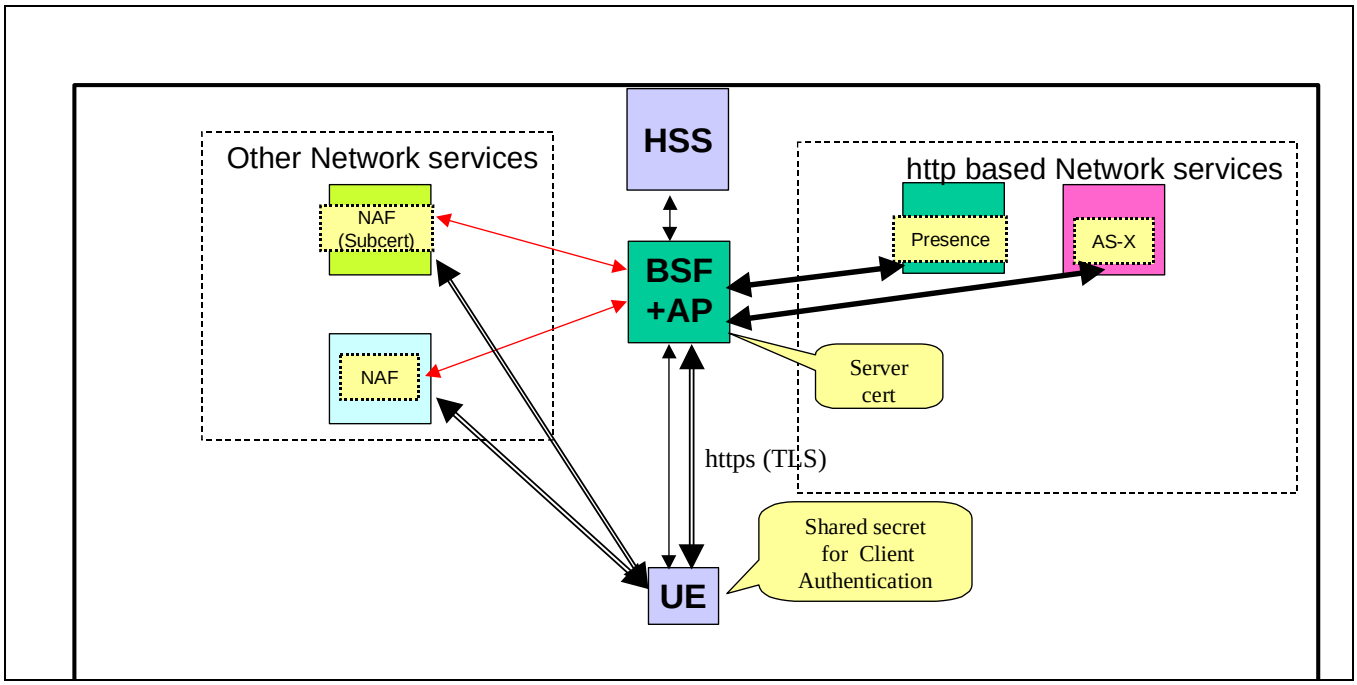
4) The UE runs the application protocol with the NAF.

This sequence of events conforms to TS SSC. No additions to TS SSC are required as no AP or TLS tunnel are involved here.

6.2 Approach using bootstrapping of application security and http authentication proxies with co-located BSF and AP.

The co-location of BSF and AP is an obvious design alternative, which allows to reduce the number of different boxes and does not require the standardisation of protocol D between BSF and NAF = AP. However, it needs to be investigated further whether a physical separation of the authentication and key agreement functionality (BSF) from the application traffic (routed through the AP) would provide enhanced security.

The sequence of events as described for the general case in the previous subsection can apply also here when BSF and AP are co-located. In should not be generally assumed that BSF and AP are co-located. However, the general solution should also make sense for this special case. But, as we see in section 6.2.2, the message flow resulting from the one in section 6.1 is sub-optimal. On the other hand, the optimised flow in section 6.2.2 does not conform to TS SSC.



6.2.1 Generic message flow

When the UE wants to access one of the application servers on the right hand side of the figure, which are connected to the AP, the sequence of events as described for the general case in the previous subsection can apply here as well, i.e.

1) the UE starts http digest aka (rfc3310, protocol A) with the BSF=AP. The BSF may contact the HSS to fetch authentication vectors (protocol C). After step 1), the UE and the BSF share a secret key, cf. TS SSC, section 4.3.1.

2) The UE sends an http request towards an application server. The http request is intercepted by the http authentication proxy (AP). The UE establishes a TLS tunnel with the AP. The AP is authenticated to the UE by means of a server certificate.

3) The UE runs http digest (rfc 2617, protocol B) with the AP = NAF to perform client authentication using the key agreed in step 1), as described in S3-030357. There is no need now for the AP to invoke protocol D in order to fetch the agreed key from the BSF , as AP and BSF are co-located in this example.

4) The UE runs the application protocol with the NAF.

Please note, that, in spite of the use of http digest aka, there is no risk of a MITM attack, as the client authentication is based on http digest in step 3).

We skip the case where an application server is not connected to the AP (left hand side of the figure) here, as there is no change with respect to section 6.1.

6.2.2 Optimised message flow

The message flow described in section 6.2.1 is generally applicable and conforms to TS SSC. However, the flow can be optimised for http-based application servers, as can be seen from the following.

When the UE wants to access one of the http-based application servers connected to the AP the sequence of events can be as follows:

1) The UE sends an http request towards an application server. The http request is intercepted by the http authentication proxy (AP). The UE establishes a TLS tunnel with the AP. The AP is authenticated to the UE by means of a server certificate.

2) the UE starts http digest akav2 (Internet draft) with the AP. The BSF may contact the HSS to fetch authentication vectors (Cx-like protocol). After step 2), the UE and the BSF share a secret key.

3) The UE runs the application protocol with application server through the AP.

This optimised flow corresponds to what has been proposed for the Ut reference point in Ericsson's contribution S3-030371. The run of http digest for client authentication (step 3) in section 6.2.1 can be omitted, as http digest akav2 performs both, key establishment and client authentication. Now, it would be clearly desirable that the general approach described in section 6.1 was specified in such a way that the specialisation to the case described in section 6.2.1 (BSF and AP co-located) would result in an optimised procedure, as in section 6.2.2.

However, there are the following obstacles:

a) TS SSC uses http digest aka, whereas the optimised flow uses http digest akav2. http digest akav2 is needed in the optimised flow to prevent MitM attacks,. On the other hand, http digest aka cannot be simply replaced by http digest akav2 in TS SSC either, because the use of http digest akav2 inside and outside a TLS tunnel would again make MitM attacks possible.

b) Rules would need to be added to TS SSC regarding tunnelled authentication. In particular, the order, in which TLS tunnel set-up with server authentication, and client authentication are performed, needs to be specified.

c) Conditions would have to be given in TS SSC when a separate run of http digest for client authentication can be omitted.

d) It would have to be specified in TS SSC how a TLS security context could be refreshed.

It is proposed here as a way ahead towards a GAA whether TS SSC can be modified or extended in this way. It is considered too early to make a decision on the alternatives examined in this section.

A future evaluation of alternatives should take into account the requirements on a GAA defined in S3z030003, as well as the HSS-related guidelines in S3-030460.

7. Conclusions and open issues

It can be conluded from the discussion on GAA requirements collected in S3z030003 and from the LSs from SA2 and CN4 that a GAA is desirable. It is proposed that SA3 endorses this view and agrees that future work on the GAA should be based on the following statements:

1. A GAA should be based on TS SSC.
2. TS SSC should be modified and extended (if possible) to cover the case of a co-located AP in an optimal way, if feasible.
3. A GAA should respect the requirements on a GAA defined in S3z030003.
4. A GAA should respect the HSS-related guidelines in S3-030460.
5. SA3 must decide to which Rel 6 GAA should apply in which way.
6. Concrete security solution for a particular feature shall be specified in the pertinent TS, but as much reference as possible should be made to an update TS SSC.

7. A GAA should be designed in such a way that it is compatible with the use of reverse http authentication proxies.
8. The open issues relating to authentication proxies need to be solved.
9. Security aspects of the network configuration need to be studied further. In particular, it needs to be investigated whether a physical separation of the authentication and key agreement functionality from the application traffic enhances security.