3GPP TSG SA WG3 Security **S3-030540**

6th – 10th October, 2003

Povoa de Varzim, Portugal

Agenda Item:	Presence, GBA
Source:	Ericsson
Title:	Protocol between authentication proxy and application server
Document for:	Discussion/Decision

1. Introduction

This document further study the issues related to the authentication proxy (AP). In particular, the issue of transferring the identity of the end-user between the AP and the application servers (AS) is investigated. It is proposed that SA3 adopts a working assumption that 3GPP shall develop private extension to solve this problem because there is no standard solution available in the market.

2. AP to AS interface

2.1 Background and current status

One of the alternatives currently discussed for the Presence Ut interface is based on the use of AP. The assumption is that the proxy would act as a gateway for several ASs, and in this way, would minimize the consumption of AKA AVs, and the parallel use of TLS.

This issue is very closely related to General Bootstrapping Architecture (GBA) discussion in SA3. The current working assumption is that GBA would be used to bootstrap also the security for the AP. SA3 is also discussing an optimized version of this architecture where AP and BSF are implemented in the same node.

Ericsson has been studying the existing standards that could be re-used for AP-AS interface. The current understanding is that even though the problem is perceived as relevant in the market, there is no standard that could be directly re-used to solve it. OMA used to have a WI on their Identity Management framework; however, the work has now been re-focused. OMA is currently evaluating the protocols developed by Liberty Alliance related to this problem. On the other hand, the protocols of Liberty Alliance are based on the ideas that are not directly compatible with GBA working assumptions of SA3.

Because there is no protocol available for this problem, it is suggested that 3GPP develops a standard for it.

2.2 Problem statement

HTTP is originally developed as stateless transport mechanism. Each request required new set up of TCP connection, and there was no concept of session that would have created a relationship between the requests. Current way of creating stateful HTTP applications is to authenticate the user in the first request, and then use some session tracking mechanism to know that all subsequent requests are coming form the same client. The client side is typically authenticated using some authentication schema in the HTTP authentication framework [RFC 2617]. There are several ways to implement the session tracking mechanism, e.g:

- Use of TLS or other stateful security mechanism.
- HTTP State Management Mechanism with cookies [RFC 2965].
- URL rewriting mechanism where the session is identified by a parameter in the URL.

There is also a fourth mechanism for session tracking; however, it is limited to HTTP POST method. In this mechanism, the messages include some hidden parameters that are not visible to the user.

When AP is utilized, the AS does not have to authenticate the user in the first request because the proxy already performed the authentication. What is needed is a new mechanism for providing the client identity to the AS in the way that the session management between the AS and the client is still possible.

2.3 Requirements

This document assume the following requirements for the solution:

REQ 1: Authentication proxy shall be able to authenticate the end-user identity using the means of Generic Bootstrapping Architecture.

REQ 2: Authentication proxy shall be able to send the end-user identity to the application server at the beginning of new HTTP session

REQ 3: Application servers shall be able to use appropriate session management mechanisms with the client.

REQ 4: The client shall be able to create several parallel HTTP sessions via the authentication proxy to different application servers.

2.4 Solution

RFC 2965 uses two headers, “Cookie” and “Set-Cookie2”, for managing the HTTP sessions. The general principle is that server may use the “Set-Cookie2” header to send session related information to the client, and the client must return this information in “Cookie” header.

This mechanism can be directly re-used to transport the authenticated identity information between the AP and AS.

This is how the mechanism could work:

1) AP generates a cookie with the user identifier and includes the cookie in every HTTP request. Example of the cookie could be as follows:

```
GET HTTP/1.1

Cookie: $Version="0"; HTTP-Asserted-Identity="username@homenetwork.org"
```

2) AS takes the user identifier from the cookie. AS can assume that the AP has authenticated the client with this identity. Of course, if the AS does not have any service for this particular user identity, it should deny the request.

What happens next is up to the local policy of AS. AS can initiate a new session management mechanism. For example, the AS can generate a new cookie and send it in the HTTP response to the client so in the next request to the same server the client includes this cookie. This use of end-to-end session management is transparent for the proxy. The AS can also use the same cookie that was received from the AP with its own session management with the client.

This solution requires that all ASs that are accessed via the AP must understand the semantics of this cookie. However, the use of AP is not mandatory in current SA3 specifications, and those ASs that do not understand the semantics of the cookie can be accessed using the means of bootstrapping function.

3. Conclusions

This document has discussed the interface between authentication proxy and application servers.

It is proposed that SA3 adopts a working assumption that 3GPP shall develop a solution for this interface because there is no standard available.

Ericsson proposes the use of cookies in the solution because cookies were originally developed for solving the session management problem, and they are currently used to tie the end-user identity to HTTP session. It is rather natural to re-use the same mechanism to solve the problem in hand.

Attached Pseudo-CR documents the proposed working assumption to Presence Security TS.

4. References

[RFC 2617] HTTP Authentication: Basic and Digest Access Authentication, IETF, June 1999.

[RFC 2965] HTTP State Management Mechanism, IETF, October 2000.

CR-Form-v7

CHANGE REQUEST

⌘ 33.cde CR CRNum ⌘rev - ⌘ Current version: 0.1.0 ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: UICC apps⌘ [ ] ME [X] Radio Access Network [ ] Core Network [X]

Title:	⌘	Protocol between authentication proxy and application server
Source:	⌘	Ericsson
Work item code:	⌘	Presence Security TS
Date:	⌘	29/09/2003
Category:	⌘	F
Release:	⌘	Rel-6

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:

2	*(GSM Phase 2)*
R96	*(Release 1996)*
R97	*(Release 1997)*
R98	*(Release 1998)*
R99	*(Release 1999)*
Rel-4	*(Release 4)*
Rel-5	*(Release 5)*
Rel-6	*(Release 6)*

Reason for change:	⌘	Requirements and solution for the protocol to be used between authentication proxy and the application servers are currently missing.
Summary of change:	⌘	Adds the requirements and a solution.
Consequences if not approved:	⌘	Cannot use an authentication proxy between the UE and application servers.

Clauses affected:	⌘	

		Y	N		
Other specs affected:	⌘		N	Other core specifications	⌘
			N	Test specifications	
			N	O&M Specifications	

Other comments:	⌘	

How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at http://www.3gpp.org/specs/CR.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://ftp.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2001-03 contains the specifications resulting from the March 2001 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

***** Begin of Change ****

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document.*

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 22.141: "Presence service; Stage 1".

[3] 3GPP TS 23.141: "Presence service; Stage 2".

[4] 3GPP TS 33.203: "3G security; Access security for IP-based services".

[5] 3GPP TS 23.228: "IP Multimedia Subsystem (IMS); Stage 2".

[6] IETF RFC 2246 (1999) “The TLS Protocol Version 1”

[7] 3GPP TS 23.002: "Network Architecture"

[8] IETF RFC 3268 (2002) “Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)”

[9] IETF RFC 3546 (2003) “Transport Layer Security (TLS) Extensions”

[10] IETF RFC 2965 (2000) “HTTP State Management Mechanism”

***** End of Change ****

***** Begin of Change ****

5 Security features

5.1 Secure Access to the Presence Server/Presence List Server

5.1.1 Authentication of the subscriber and the network

Authentication between the subscriber and the network shall be performed as specified in clause 6.1.

The authentication of the subscriber shall be based on the ISIM as defined in [4]. The authentication of the subscriber shall be HTTP based. The authentication of the subscriber shall not be based on asymmetric mechanisms.

The Server is authenticated by means of asymmetric cryptography using a Server Certificate. The authentication of the Server shall be based on strong security. The use of anonymous Diffie Hellman is not allowed.

Note: The interleaving attack shall not be possible

[Editors Note: The exact details on Server Certificate are FFS cf. X509v3 certificate and PKIX]

[Editors Note: It is FFS how the user is authenticated]

5.1.1.1 Authenticated identity management

[Editors Note: This section is related to a special case when an authenticatin proxy is located between the UE and any application server. Parts of this section may later be moved to another technical specification.]

If there is an authentication proxy between the UE and the Presence Server / Presence List Server, the following requirements shall apply:

Authentication proxy shall be able to authenticate the end-user identity using the means of Generic Bootstrapping Architecture.

> *[Editors Note: the exact details of the security architecture is FFS and dependant on decisions related with the ongoing work on GBA (Generic Bootstrapping Architecture)]*

Authentication proxy shall be able to send the end-user identity to the application server at the beginning of new HTTP session.

Application servers shall be able to use appropriate session management mechanisms with the client.

> Note: The used session management mechanism is out of the scope of 3GPP specifications.

The client shall be able to create several parallel HTTP sessions via the authentication proxy to different application servers.

***** End of Change ****

***** Begin of Change ****

6 Security Mechanisms

[Editors Note: This should be a profiling of [6] and [8]]

6.1 Authentication and key agreement

6.1.1 Authentication of the user

6.1.1.1 Authenticated identity management

[Editors Note: This section is related to a special case when an authenticatin proxy is located between the UE and any application server. Parts of this section may later be moved to another technical specification.]

The authenticated identity management mechanism shall be implemented using an identity management cookie in the “Cookie” header specified in [10].

Procedure:

> 1) Authentication proxy shall authenticate the UE by the means of General Bootstrapping Architecture.

2) Authentication proxy shall generate the identity management cookie with the user identifier and include it in every HTTP request. Since the Cookie header may already include some cookie values, the authentication proxy shall check all cookies named in the same way as the identity management cookie, and remove possible false values. All other cookies are transparent to the proxy.

3) Application server shall take the user identifier from the identity management cookie. The server shall assume that the proxy has authenticated the client with the identity present in the cookie. If the server does not have any service related to the user identity, it shall deny the request.

4) Application server may use any appropriate session management mechanism with the client.

Note: Application server may use cookies for session management as specified in [10], or any other session management mechanism that can be used through a proxy. The application server may also re-use the same cookie that was received from the proxy for managing its own session with the client.

The exact syntax of the identity management cookie is specified in Appendix A.

***** End of Change ****

***** Begin of Change ****

Appendix A: Identity management cookie (normative)

The syntax of the identity management cookie is:

cookie-value = NAME "=" VALUE

NAME = "HTTP-Asserted-Identity"

VALUE = username-value

username-value = quoted-string

[Editors note: The exact content of the username-value dependant on decisions related to the ongoing work on GBA (Generic Bootstrapping Architecture).]

Other parameters are as specified in [10].

Example:

Cookie: $Version="0"; HTTP-Asserted-Identity="username@homenetwork.org"

***** End of Change ****