**3GPP TSG SA WG3 Security — SA3#35** **S3-040729**
**October 5-8, 2004**
**St Paul's Bay, Malta**

**Source:** **Lucent Technologies**
**Document for:** **Discussion**
**Agenda Item:** **6.17**

## 1. Introduction

~~In t~~This document addresses ~~various~~ a number of issues related to GUP security. In the context of GUP, security encompasses two aspects: authentication and encryption.

In the context of GUP, we clearly need both: encryption to make sure that data exchanged between an application and the GUP server remains confidential (e.g. location information, banking details); authentication to make sure that entities trying to access data are authorized to do so (invocation identity) and that the response is authenticated (sender identity). As agreed by the GUP working group, the GUP architecture relies on Liberty Alliance Identity Web Services Framework (LA IS-WSF) specifications (http://www.projectliberty.org/specs/liberty-idwsf-security-mechanisms-v1.1.pdf), i.e., the various parties will communicate with each other using SOAP messages over HTTP. Based on these requirements, there are many ways to implement encryption and authentication.

## 2. Possible solutions, based on LA specifications

The LA specifications do recommend a number of SSL/TLS cipher suites, e.g.,

- TLS_RSA_WITH_RC4_128_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_AES_CBC_SHA
- TLS_DHE_DSS_WITH_AES_CBC_SHA

Other security protocols may also be used: e.g., Kerberos, IPsec “as long as they implement equivalent security measures.”

The LA document also allows a multiplicity of authentication protocols :

| URI | Peer Entity | Message |
|---|---|---|
| *urn:liberty:security:2003-08:null:null* | No | No |
| *urn:liberty:security:2003-08:null:X509* | No | Yes |
| *urn:liberty:security:2003-08:null:SAML* | No | Yes |
| *urn:liberty:security:2004-04:null:Bearer* | No | No |
| *urn:liberty:security:2003-08:TLS:null* | Recipient | No |
| *urn:liberty:security:2003-08:TLS:X509* | Recipient | Yes |
| *urn:liberty:security:2003-08:TLS:SAML* | Recipient | Yes |
| *urn:liberty:security:2004-04:TLS:Bearer* | Recipient | No |
| *urn:liberty:security:2003-08:ClientTLS:null* | Mutual | No |
| *urn:liberty:security:2003-08:ClientTLS:X509* | Mutual | Yes |
| *urn:liberty:security:2003-08:ClientTLS:SAML* | Mutual | Yes |
| *urn:liberty:security:2004-04:ClientTLS:Bearer* | Mutual | No |

Which specific protocol to use is outside the scope of LA specification document.

## 1. For discussion

Those solutions can be split into two categories: transport level and application level. In our case (SOAP over HTTP), the transport level is TCP/IP used by HTTP while the application level is SOAP.

### 1.1. Transport level solution

By using a transport level solution, ~~we create~~ a secure communication channel is created. The application can then directly use the secure channel ~~and does not~~without the need ~~to provide~~for any additional security mechanisms, although this may be implemented, if so desired ~~itself (but it can if it wants to)~~. In our case, this means using SSL/TLS for communication between the parties.

> Note: this method is used in e-commerce: the sites are designed and deployed. ~~and f~~When ~~or~~ security is required, the site is accesses via https:// access, instead of ~~a~~ the unsecured http:// access. In this scenario, the security of the site is totally transparent to the web site developer.

### 1.2. Application level solution

By using an application level security solution, ~~we~~ one does not have to assume~~assume~~ that the communication channel ~~does not~~ provides the ~~proper~~ desired security level, or that the end-points completely trust the intermediary nodes. Therefore ~~or we want additional security, therefore additional security is~~ security is provided ~~added~~ at the application layer. ~~In~~For ~~our~~ GUP implementation~~case~~, this means that the security will have to be provided at the level of the SOAP messages, by encrypting/signing SOAP Body elements.

> Note: this is what is used for secure email using S/Mime. The communication channels are not secure and the application (email in this case) adds an extra layer of security.

### 1.3. Transport vs. Application Solutions: Pros and cons

A transport level solution performs well for point-to-point security, i.e. when party A and party B want to communicate securely with each other. Problems arise if a third party is ~~we want to bring a third party in the picture~~involved (e.g. a non-trusted proxy). An application level security solution does not have this limitation.

Transport level solution are also very appropriate for synchronous communications where the client and the server create a session where a query is sent and a response is sent back. In the case of asynchronous communications, it is not feasible to let the connection open. Therefore, when the response is ready to be sent, a new session may need~~s~~ to be established. An application level solution does not have this limitation.

The biggest advantage of using "transport" security over "application" security is the fact that the transport security ~~is~~ can be considered as "~~universal~~always available" and ~~and~~ transparent to the application (e.g. as in the e-commerce example mentioned above). While ~~A~~application level security is more flexible, the ~~but~~ ~~the~~ overall security of the implementation does depend on the specific implementation(s) and the implementer's skills.

### 1.4. Encryption and Authentication Solutions

There are various solutions for both encryption and authentication. We also detail the various requirements. For the various solutions, we will use Client (C) and Server (S).

#### 1.4.1. SSL/TLS

C owns PKI certificate $Cert_{C|}$~~C~~ from certification authority $CA_{C|}$~~C~~.
S owns PKI certificate $Cert_{S|}$~~S~~ from certification authority $CA_{S|}$~~S~~.
$CA_{S|}$~~S~~ is in the list of C's trusted CAs.
$CA_{C|}$~~C~~ is in the list of S's trusted CAs.

~~transport layer security:~~
TCP with SSL/TLS using client and server certificates application layer: **HTTP with SOAP:**

**Pros:**

totally transparent to the GUP components, ~~-~~ trusted solution based on PKI and SSL certificates

**Cons**:

computation intensive (especially for devices with limited computing power) expensive for asynchronous communications (the SSL/TLS channel needs to be re-established, which ~~is~~ may be expensive) ~~rigid architecture because of point-to-point securit~~.~~y~~

### 1.4.2. Application layer solution WS-Security

(see http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)

C and S communicate over an unsecure channel.
The SOAP messages they exchange contain:
encrypted data in their soap:Body (body)
signature data in the soap:Env (envelope)

For encryption, a key has to be negotiated.
For authentication, each party needs to check the other party's signature via a CA, using an identity provider.

(Note that with a transport level solution, we get all of this for free).

Pros:
flexible
Cons:
there are ~~a lot of~~many algorithms to choose from,
~~———~~less mature technology requires some ~~extra~~ additional development~~implementation~~.

### 1.4.3. WS-Encryption + HTTP authentication

The previous solution can be changed as follows. The SOAP message is not signed using the envelope only, but ~~. R~~rathe~~r,~~r the entire message (envelope and body) ~~are~~ is signed and the signature is ~~put~~ included in the HTTP header.

### 1.4.4. WS-Encryption + SSL/TLS

Encryption is done using SSL/TLS (server-side certificates). Authentication is done using WS-Security.

### 1.4.5. SAML

SAML is an XML dialect to exchange "security assertions". Per se, it does not provide any security. In ~~our~~ GUP context, SAML can be used by either party to check that the signatures are OK, using a trusted third party (TTP). Upon receiving the message, the party sends a SAML request to the TTP to check that the signature is valid.

## 2. Conclusion

Currently, the LA specifications do allow implementations to negotiated among many different security mechanisms. Although the final decision on which specific security mechanism to be used for GUP is CN4's, 3GPP SA3 should use its security expertise and evaluate the different available solutions, present the pros and cons of each, and potentially propose~~d~~ a "default" protocol (or a small set of

options to chose from) to the ~~the 3GPP~~ CN4, via a LS ~~GUP~~. For example, the use of SSL/TLS with both client and server side certificates can be defined as the “default” mechanism, since it has been successfully implemented in many applications and is widely supported in the field. Other mechanisms can be made optional.