Kerberos SAML Profiles

Working Draft 01, 9 January 2004

Document identifier:
draft-sstc-solution-profile-kerberos-01

Location:
http://www.oasis-open.org/committees/documents.php?wg_abbrev=security

Editors:
John Hughes, Entegrity Solutions(john.hughes@entegrity.com)
Tim Alsop, CyberSafe Ltd(Tim.Alsop@CyberSafe.Ltd.UK)

Contributors:
TBD

Abstract:
This document describes a number of use cases and profiles pertaining to the application of SAML with Kerberos, DCE and Windows.

Status:
Committee members should send comments on this specification to the securityservices@lists.oasis-open.org list. Others should subscribe to and send comments to the security-services-comment@lists.oasis-open.org list. To subscribe, send an email message to security-services-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Security Services TC web page (http://www.oasisopen.org/committees/security/).

Table of Contents

1 Introduction

This document proposes a number of use cases and profile where SAML can be used in conjunction with Kerberos based technology. This includes:

- Kerberos v5 (as defined in RFC 1510)
- Kerberos GSS-API mechanism (as defined in RFC 1964)
- DCE (Distributed Computing Environment)
- Windows 2000/2003

In particular of interest are the last two technologies. Both of these leverage the ability of Kerberos to transport authorization data.

Note that whilst the use cases contained within this document build upon the Browser profiles defined else where in SAML 2.0, new components are introduced.

1.1 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 IETF RFC 2119 [RFC2119].

2 Use Case

The high-level Kerberos use case has a user of a client workstation logging on to the local Key Distribution Centre (KDC), the Kerberos client software being present on the workstation. Following successful authentication the client credentials reside on the workstation.

The workstation user then wishes to gain access to resources on a remote site in another management domain, so that:

- No further authentication is required
- Session attributes are transferred seamlessly over to the remote application so that it makes appropriate authorization decisions

Note that the local site may, or may not have a web server

Figure 1 illustrates the high level use case.

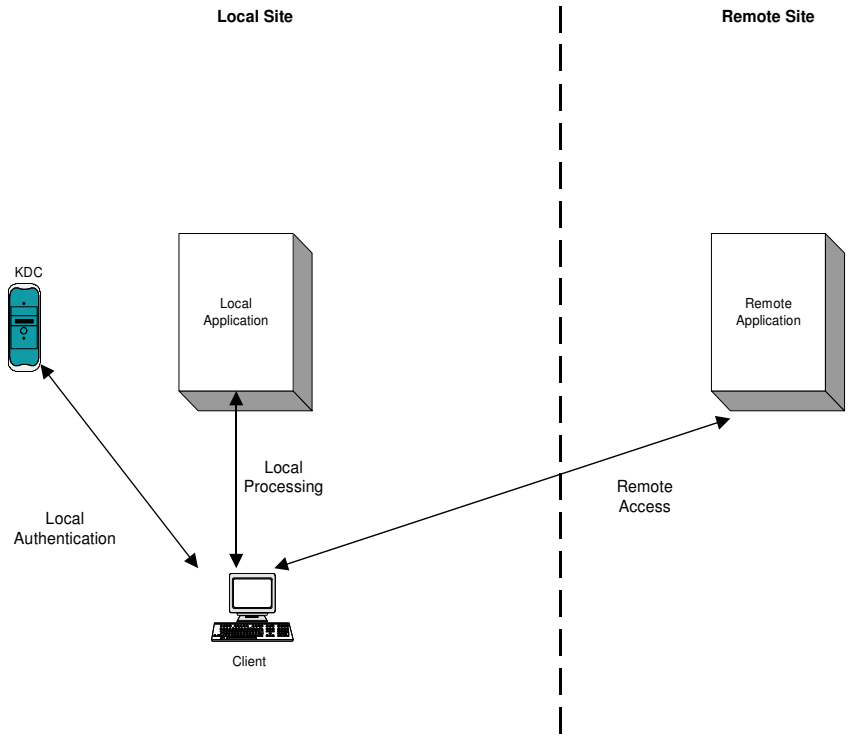

Figure 1– Overall Kerberos Use case

This use case can be further broken down into a number of use case scenarios, each involving a different mix of SAML components and protocols. The following sections articulates the use case scenarios, with further sections describing the components being used.

3 Use Case Scenarios

3.1 Browser client – Browser requesting SAML assertion

In this scenario the user on the client workstation has authenticated itself to the local KDC and obtained a TGT. The TGT is stored locally in the workstations credentials cache. The user may or may not interact with a local kerberosised application. At some point the user wishes to use a web browser on the client workstation to gain access on a remote web server in a different management domain.

Figure 3 provides an overview of the components.

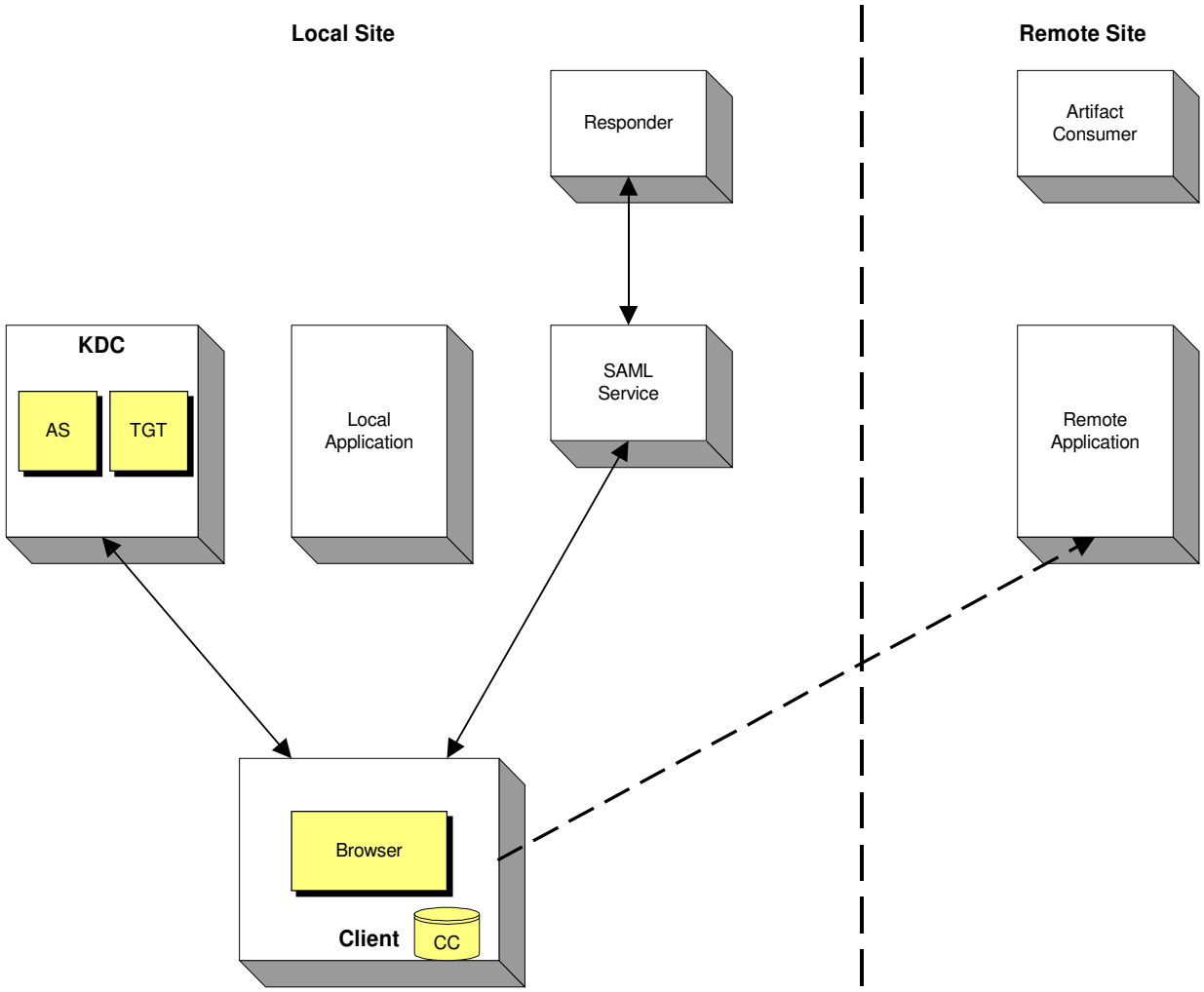

Figure 2– Browser Client – Browser Requesting SAML Assertion

The proposed processing is as follows:

1. The user on the workstation authenticates into the local domain/cell/realm, using client software resident on the workstation. Successful authentication results in an appropriate TGT being provided back to the workstation. The user can then access resources in the local site
2. The user then wishes to gain access to a remote site. In this scenario the remote site is a web server (or Portal). The first step in the process is to obtain a Service Ticket (ST) for a SAML Service within the local site. The workstation could initiate this either from a signed applet or an application

running on the workstation.

3. Having obtained the SAML Service ST, the application (or applet) then sends a request to the SAML service to generate a SAML Assertion. The SAML Service front ends the local site's SAML Responder. Using the SAML Service ST to gain access to the SAML Service means that the workstation has authenticated itself to the SAML Service and the user identity has also been passed to it. The request defines whether the response back to the workstation contains either an artifact (which references a SAML assertion) or a SAML assertion. Refer to Section 4.1.1 for the definition of the protocol.
4. The SAML Service obtains the user identification from the ST used to authenticate the connection between the workstation and the SAML Service. However if any PAC attributes exists within the ST then these are used to create AttributeStatements within the SAML assertion. Refer to Section 5.
5. The SAML Service responds back to the application/applet with either
   - an artifact that references the generated SAML assertion
   - the generated SAML Assertion within a SAML Response message
6. The Browser on the workstation then attempts to gain access to a resource on the remote web site. Either the Browser/POST or Browser/Artifact profile processing can then be used to process the request.

As for the Browser/POST and Browser/Artifact profiles confidentiality and message integrity MUST be maintained. If the artifact is passed to the remote site then it is RECOMMENDED that either SSL 3.0 or TLS 1.0 is used to protect the connection. If a SAML assertion is passed to the remote site then the SAML Response MUST be digitally signed following the guidelines given in **[SAMLCore]**.

3.2 Browser client – Application requesting SAML assertion

In this use case scenario it's the local application (for instance a web server or Portal) that requests and issues the SAML Assertion. This scenario would be typical of an environment where Kerberos is used to perform authentication within the local site and a Portal is used as an intermediary to gain access to remote resources.

Figure 4 provides an overview of the architecture.

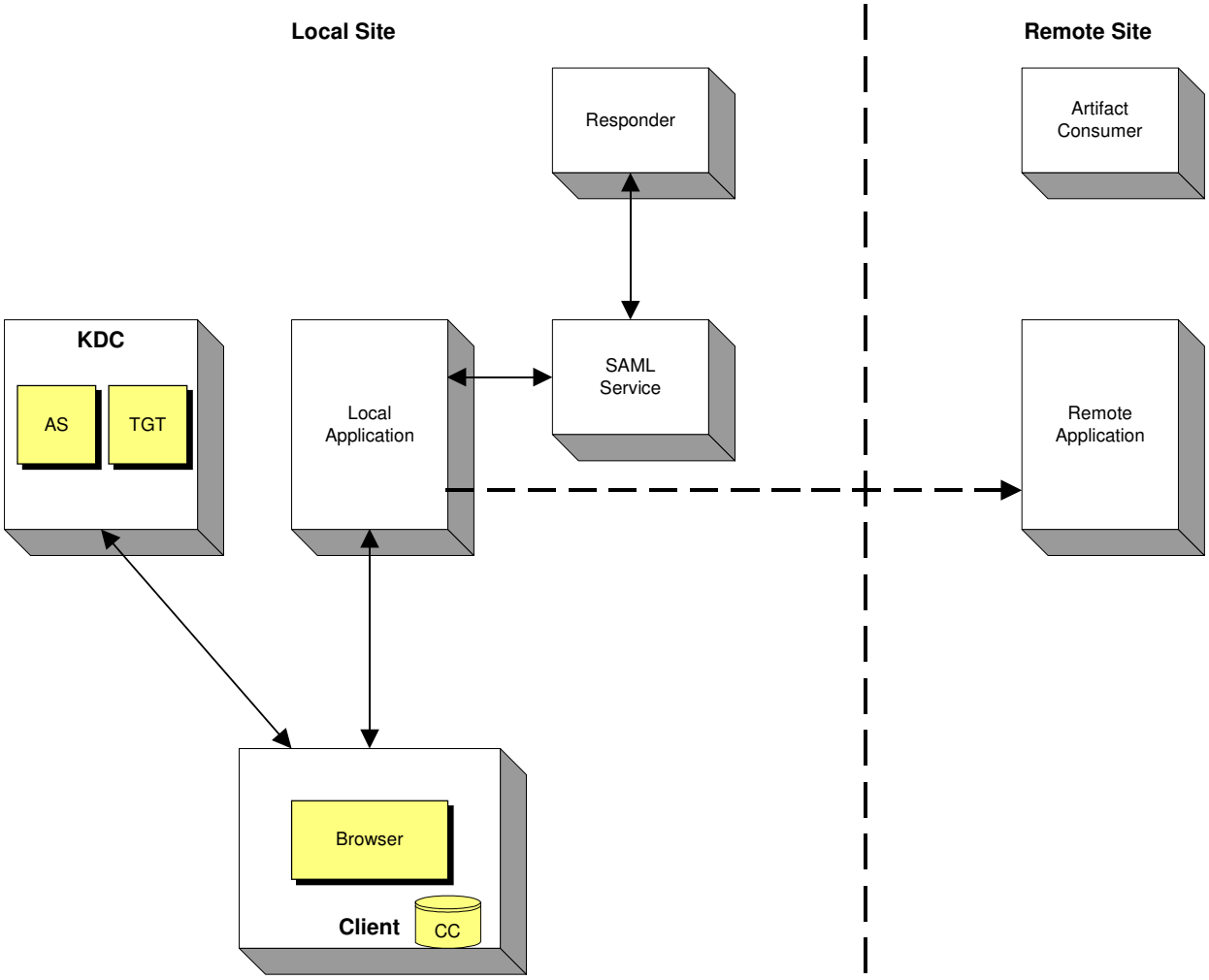

Figure 4– Browser Client – Application Requesting SAML Assertion

The proposed processing is as follows:

1. The user on the workstation authenticates into the local domain/cell/realm, using client software resident on the workstation. Successful authentication results in an appropriate TGT being provided back to the workstation. The user can then access resources in the local site
2. The user then wishes to gain access to remote resources (probably unbeknown to them) via the local Portal. The user's credentials on the workstation need to be passed to the Portal, this MUST be performed in a secure manner. It is RECOMMENDED that either SASL, TLS or SPNEGO are used to secure transport the user's credentials to the Portal.. The local Portal obtains a Service Ticket (ST) for the SAML Service using the user's credentials.
3. Having obtained the SAML Service ST, the portal sends a request to the SAML service to generate a SAML Assertion. The request defines whether the response back to the workstation contains either an artifact or an assertion. Refer to Section 4.1.1 for the definition of the protocol.
4. The SAML Service obtains the user identification from the ST used to authenticate the connection between the Portal and the SAML Service. However if any PAC attributes exists within the ST then these are used to create AttributeStatements within the SAML assertion. Refer to Section 5.
5. The SAML Service responds back to the Portal either
   - an artifact that references the generated SAML assertion
   - the generated SAML Assertion within a SAML Response

6. The Portal then gains access to the remote resource either by:
   - Passing the artifact as a query variable in the HTTP URL
   - Passing the artifact as a named custom Header variable
   - The generated SAML Response as POST Data

For this last step confidentiality and message integrity MUST be maintained. If the artifact is passed to the remote site then it is RECOMMENDED that either SSL 3.0 ot TLS 1.0 is used to protect the connection. If a SAML Response is passed to the remote site then the SAML Response MUST be digitally signed following the guidelines given in **[SAMLCore]**.

3.3 Non-Browser client – Client requesting SAML assertion

In this use case scenario it's the workstation that requests and issues the SAML Assertion, however the workstation does not have any HTTP-based components present.

Figure 5 provides an overview of the architecture.

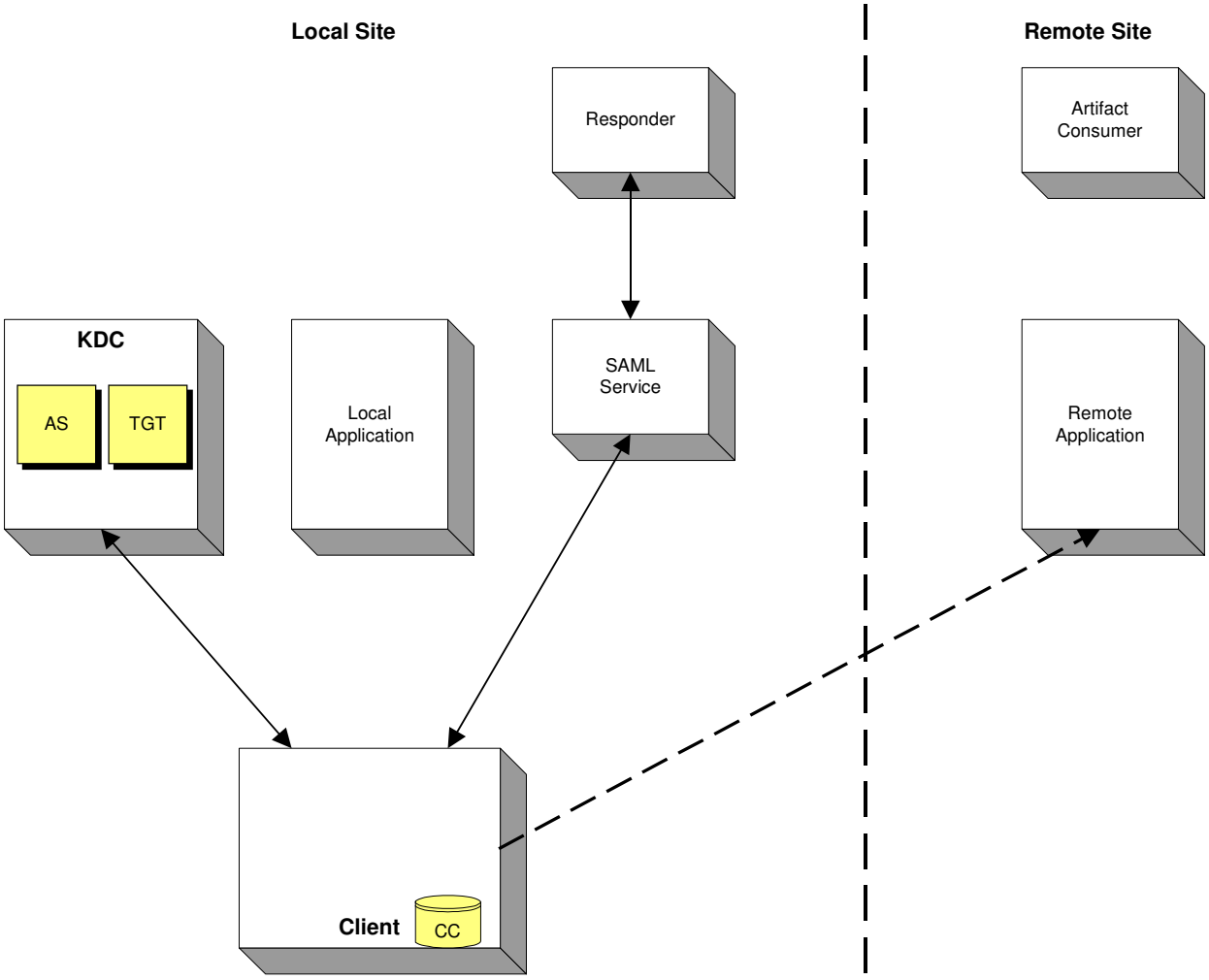

Figure 5– Non-Browser Client – Client Requesting SAML Assertion

The proposed processing is as follows:

1. The user on the workstation authenticates into the local domain/cell/realm, using client software resident on the workstation. Successful authentication results in an appropriate TGT being provided back to the workstation. The user can then access resources in the local site

2. The user then wishes to gain access to a remote site. In this scenario the remote site is an application that in non-HTTP based. The first step in the process is to obtain a Service Ticket (ST) for a SAML Service with the local site. The workstation would initiate an application running on the workstation to perform this.
3. Having obtained the SAML Service ST, the workstation then sends a request to the SAML service to generate a SAML Assertion. The SAML Service front ends the SAML Responder. Using the SAML Service ST to gain access to the SAML Service means that the workstation has authenticated itself to the SAML Service and the user identity has also been passed to it. The request defines whether the response back to the workstation contains either an artifact or an assertion. Refer to Section 4.1.2 for the definition of the protocol.
4. The SAML Service obtains the user identification from the ST used to authenticate the connection between the workstation and the SAML Service. However if any PAC attributes exists within the ST then these are used to create AttributeStatements within the SAML assertion. Refer to Section 5.
5. The SAML Service responds back to the application either
    - an artifact that references the generated SAML assertion
    - the generated SAML Assertion within a SAML Response
6. The workstation then uses the either the Artifact or generation SAML Assertion to communicate with the remote application. The application protocol used to perform this is not defined.

For this last step confidentiality and message integrity MUST be maintained. If the artifact is passed to the remote site then it is RECOMMENDED that either SSL 3.0 ot TLS 1.0 is used to protect the connection. If a SAML Response is passed to the remote site then the SAML Response MUST be digitally signed following the guidelines given in **[SAMLCore]**.

3.4 Non-Browser client – Application requesting SAML assertion

In this use case scenario it's the application that requests and issues the SAML Assertion, however the workstation does not have any HTTP-based components present.

Figure 6 provides an overview of the architecture.

The proposed processing is as follows:

1. The user on the workstation authenticates into the local domain/cell/realm, using client software resident on the workstation. Successful authentication results in an appropriate TGT being provided back to the workstation. The user can then access resources in the local site
2. The application on the local web site then wishes to gain access to resources on the remote site. The user's credentials on the workstation need to be passed to the local application, this MUST be performed in a secure manner. It is RECOMMENDED that either SASL, TLS or SPNEGO are used to secure transport the user's credentials to the Portal.. The local Portal obtains a Service Ticket (ST) for the SAML Service using the user's credentials.
3. Having obtained the SAML Service ST, the local application sends a request to the SAML service to generate a SAML Assertion. The request defines whether the response back to the workstation contains either an artifact or an assertion. Refer to Section 4.1.2 for the definition of the protocol.
4. The SAML Service obtains the user identification from the ST used to authenticate the connection between the applicationl and the SAML Service. However if any PAC attributes exists within the ST then these are used to create AttributeStatements within the SAML assertion. Refer to Section 5.
5. The SAML Service responds back to the local application either
    - an artifact that references the generated SAML assertion
    - the generated SAML Assertion
6. The local application then uses the either the Artifact or generated SAML Assertion to communicate with the remote application. The application protocol used to perform this is not defined.

For this last step confidentiality and message integrity MUST be maintained. If the artifact is passed to the remote site then it is RECOMMENDED that either SSL 3.0 ot TLS 1.0 is used to protect the connection. If a SAML Response is passed to the remote site then the SAML Response MUST be digitally signed following the guidelines given in **[SAMLCore]**.

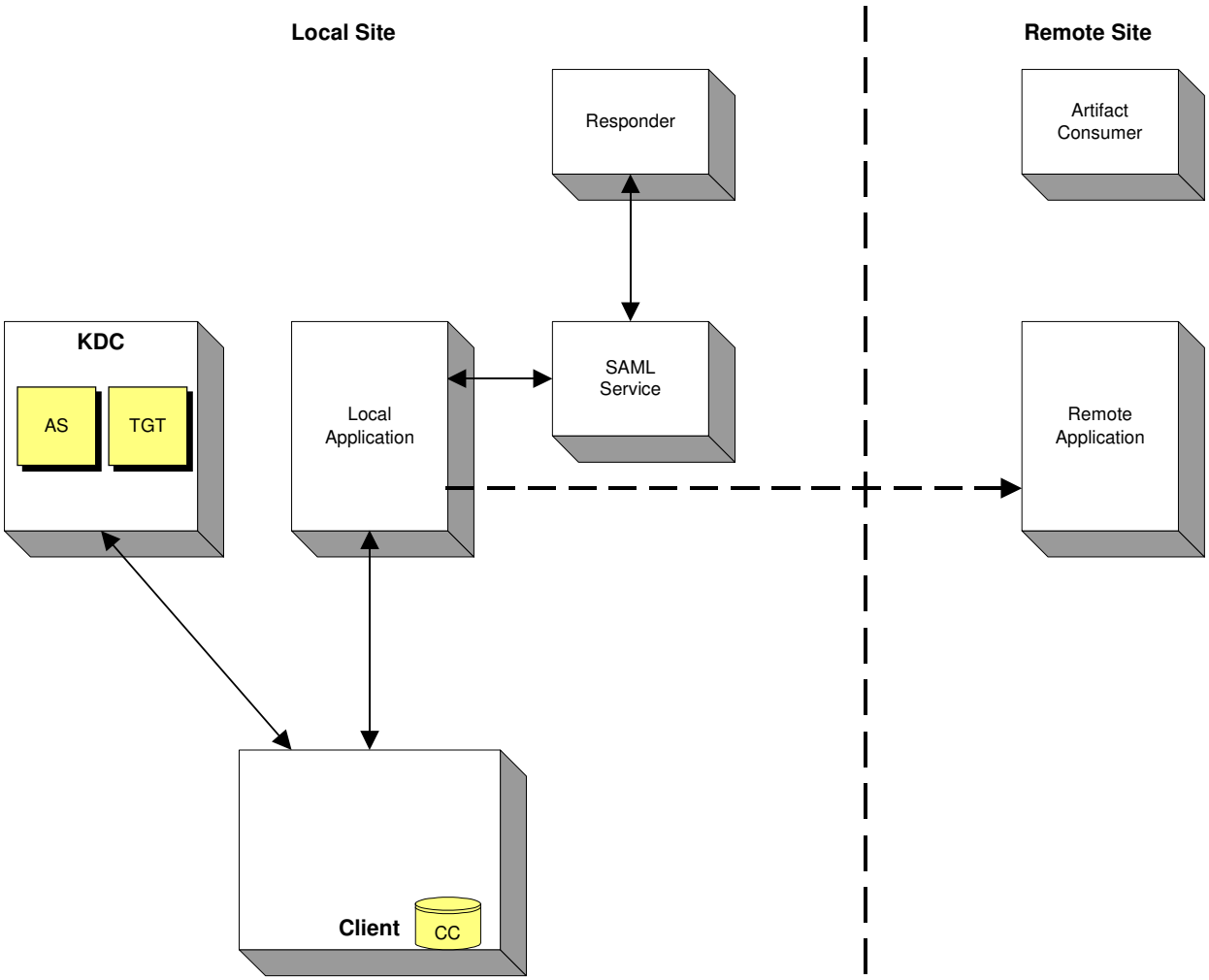

Figure 6– Non-Browser Client – Application Requesting SAML Assertion

4 Solution Components

4.1 SAML Service

The SAML Service is a front end to a SAML Responder and is a Kerberos-based service. The SAML Service can be co-located with the SAML Responder or a simple wrapper. In all cases the connection between the SAML Service and SAML Responder MUST be secure.

4.1.1 SOAP binding

This uses the standard SOAP binding for the SAML protocol as defined in TBD. Two types of requests can be made on the SAML Service, to either request an assertion or an artifact (which refers to a SAML assertion). In both cases the SAML protocol `<SubjectQuery>` element is extended

4.1.1.1 Element <SubjectRequestArtifact>

This query requests that an artifact is returned for the given subject. The following schema fragment defines the `<SubjectRequestArtifact>`

TBD

The SAML Service MUST validate that the identity supplied in the Service Tick matches that in the `<Subject>` element.

4.1.1.2 Element <SubjectRequestAssertion>r

This query requests that an assertion is returned for the given subject. The following schema fragment defines the `<SubjectRequestAssertion>`

TBD

The SAML Service MUST validate that the identity supplied in the Service Tick matches that in the `<Subject>` element.

4.1.1.3 Element <ArtifactResponse>

When an Artifact is requested using the query SubjectRequestArtifact, the SAML response contains a <ArtifactResponse> element. The following schema fragment defines the `<ArtifactResponse>` element

TBD

4.1.2 Non-HTTP binding

TBD

4.2 Authorization Data

TBD – refer to following section

5 Normalization

5.1 Introduction

TBD

5.2 Kerberos

TBD

Example of how details of a Kerberos principal are carried within a SAML Assertion.

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
       MajorVersion="1"
       MinorVersion="1"
       AssertionID="P1YaAztP6UfswxAjax5TPxQ"
       Issuer="www.entegrity.com"
       IssueInstant="2002-06-19T17:05:37.795Z">
       <saml:Conditions NotBefore="2002-06-19T17:00:37.795Z"
                        NotOnOrAfter="2002-06-19T17:10:37.795Z"/>
       <saml:AuthenticationStatement
         AuthenticationMethod="urn:ietf:rfc:1510"
         AuthenticationInstant="2002-06-19T17:05:17.706Z">
         <saml:Subject>
              <saml:NameIdentifier
                   NameQualifier="http://www.entegrity.com/"
                   Format="urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos">
                   talsop@CYBERSAFE.LTD.UK
              </saml:NameIdentifier>
              <saml:SubjectConfirmation>
                   <saml:ConfirmationMethod>
                        urn:oasis:names:tc:SAML:1.0:cm:artifact
                   </saml:ConfirmationMethod>
                   <saml:SubjectConfirmationData>
                        AAGZE1RNQJEFzYNCGAGPjWvtDIRSZ41WDqBphqA
                   </saml:SubjectConfirmationData>
              </saml:SubjectConfirmation>
         </saml:Subject>
       </saml:AuthenticationStatement>
</saml:Assertion>
```

5.3 Distributed Computing Environment (DCE)

TBD. How DCE attributes are mapped into SAML Attribute Statements.

5.4 Windows

TBD. How Windows PAC attributes are mapped into SAML Attribute Statements.

6 SAML-Defined Identifiers

6.1 Authentication Method Identifiers

6.1.1 Kerberos

URI: urn:ietf:rfc:1510

The authentication was performed by means of the Kerberos protocol **[RFC 1510],** an instantiation of the Needham-Schroeder symmetric key authentication mechanism **[Needham78]**

6.2 NameIdentifier Format Identifiers

6.2.1 Kerberos Principal Name

URI: urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos

Indicates that the content of the `<NameIdentifier>` element is in the form of a Kerberos principal name.

6.2.2 DCE Principal Name

URI: urn:oasis:names:tc:SAML:2.0:nameid-format:DCE

Indicates that the content of the `<NameIdentifier>` element is in the form of a DCE principal name.

7 References

7.1 Normative References

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, IETF RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt.

TBD

A. Acknowledgments

The editors would like to acknowledge the contributions of the OASIS Security Services Technical Committee, whose voting members at the time of publication were:

- TBD

B. Revision History

Rev	Date	By Whom	What
01	8 Jan 2004	John Hughes	Initial Version

C. Notices

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.