NCI AGENCY

NATO Communications and Information Agency
Agence OTAN d'information et de communication

AGENCY INSTRUCTION

INSTR TECH 06.02.02

SERVICE INTERFACE PROFILE FOR REST SECURITY SERVICES

Effective date:

Revision No: Original

Issued by: Chief, Core Enterprise Services

Approved by: Director Service Strategy

Table of Amendments

Amendment No	Date issued	Remarks

Author Details

Organization	Name	Contact Email/Phone
NCI Agency	A. Ross	Alan.Ross@ncia.nato.int

Table of Contents

AGENCY INSTRUCTION 06.02.02

SERVICE INTERFACE PROFILE FOR REST SECURITY SERVICES

0 PRELIMINARY INFORMATION

0.1 References

A. NCIA/GM/2012/235; Directive 1 Revision 1; dated 3 May 2013
B. NCIARECCEN-4-22852 DIRECTIVE 01.01, Agency Policy on Management and Control of Directives, Notices, Processes, Procedures and Instructions, dated 20 May 2014
C. NCIARECCEN-4-23297, Directive 06.00.01, Management and Control of Directives, Processes, Procedures and Instructions on Service Management, dated 03 June 2014

0.2 Purpose

This Technical Instruction (TI) provides detailed information, guidance, instructions, standards and criteria to be used when planning, programming, and designing Agency products and services. In this specific case the TI defines a Service Interface Profile (SIP) for one of NATO's Core Enterprise Services.

TIs are living documents and will be periodically reviewed, updated, and made available to Agency staff as part of the Service Strategy responsibility as Design Authority. Technical content of these instructions is the shared responsibility of SStrat/Service Engineering and Architecture Branch and the Service Line of the discipline involved.

TIs are primarily disseminated electronically[1], and will be announced through Agency Routine Orders. Hard copies or local electronic copies should be checked against the current electronic version prior to use to assure that the latest instructions are used.

0.3 Applicability

This TI applies to all elements of the Agency, in particular to all NCI Agency staff involved in development of IT services or software products. It is the responsibility of all NCI Agency Programme, Service, Product and Project Managers to ensure the implementation of this technical instruction and to incorporate its content into relevant contractual documentation for external suppliers.

1 SIP INTRODUCTION

NATO communication and information systems (CIS) operate in a heterogeneous environment, with service providers and service consumers operating under multiple different frameworks and application contexts. Systems deployed onto NATO networks are subject to an appropriate security approval and/or accreditation process addressing the confidentiality, integrity and availability of security objectives where different available technologies and mechanisms can be used to apply security.

To ensure interoperability between services, both within NATO, and between NATO and its partners, there is a need to define a standard (and standards-based) profile which will be mandatory for all service operations in the NATO Network Enabled Capability (NNEC) messaging environment. This Service Interface Profile (SIP) has been designed to accommodate new and existing security technologies and mechanisms offering a security framework that is implementation-independent.

---

[1] https://servicestrategy.nr.ncia/SitePages/Agency%20Directives%20(Technical).aspx

This specification provides the profile for securing representational state transfer (REST) web services (known as RESTful web services) that are deployed within the NNEC web service infrastructure. It specifies security requirements that need to be accounted for depending on the environment in which the services are being deployed, and the level of assurance required for protecting those services. This profile covers the required security protection profile for a *Client* to access protected resources on a *Resource Server* using REST. It includes:

- The operations for requesting access to protected resources, how the requests are structured and the elements that are contained within the requests.

This profile considers currently available open standards specifications that can be implemented to apply security within the wider context of the web services environment.

1.1 Scope

REST is an architectural style defined as a set of constraints on a distributed hypermedia system and implemented by a set of standard protocols that adhere to these constraints. The REST messaging SIP proposal. [NCIA TR/2012/SPW008423/11, 2013] specifies how web services can be implemented in a REST architectural style honouring the principles of NNEC. The next stage for profiling the use of RESTful web services within NATO is to specify the application of security to RESTful web services.

A detailed literature study was conducted, based on academic research and industry best practices, to determine the best approach for applying security to RESTful web services. The results from this study are documented in [NCIA TR/2012/SPW008423/17, in prep.], which recommends adopting a generic security framework that can interact with different types of identity management systems including interworking with SOA (service-oriented architecture) Platform Security Services as profiled by the NATO Communications and Information (NCI) Agency in [NC3A RD-3140, 2011]. This SIP specifies OAuth 2.0 as the generic security framework, describing the key components that make up the NCI Agency REST security services, the relationships between those key components and the data structures required and used by those key components.

This document also considers that RESTful web services will not only be deployed within a single organization. NATO must provide effective and efficient conduct of modern joint military operations where cross-domain information exchange is required between different security domains under different administrative control. As such, this specification describes appropriate security measures for consideration dependent on the level of assurance that is required to protect those services and the information being accessed.

1.2 Audience

The target audience for this specification is the broad community of NNEC stakeholders, who are delivering capability in an NNEC environment, or anticipate that their services may be used in this environment.

These may include (but are not limited to):

- Project Managers procuring NATO communication and information systems.
- The architects and developers of service consumers and providers that interact with RESTful services as described in REST Messaging SIP Proposal (see [NCIA TR/2012/SPW008423/11, 2013]).
- Coalition partners whose services may need to interact with NNEC services.
- System integrators delivering systems into the NATO environment.

1.3 Notational Conventions

The following notational conventions apply to this document:

- The keywords "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, 1997].
- Words in *italics* indicate terms referenced in Section 1.4.
- `Courier font` indicates syntax derived from the different open standards [OASIS WS-Security, 2006], [W3C WS-Addressing, 2006], [W3C XML-Signature, 2002], [OASIS SAML, 2005], [OASIS SAML Token Profile, 2006], and [WS-I Security, 2010].

1.4 Taxonomy Allocation

This service falls under the following allocation under the C3 Taxonomy [NAC AC/322-N(2012)0092, 2012]:

Technical Services → Core Enterprise Services → SOA Platform Services → SOA Platform IA Services.

1.5 Terminology

In the area of web services and web services security there are a variety of definitions used and a variety of meanings for those definitions. This document uses terminology from the NCI Agency SOA Platform SIPs and OAuth 2.0 specifications. The following terminology is used in this profile and a harmonization between terminologies used throughout the two sets of specifications is provided:

Access token	OAuth 2.0 terminology describing the credentials, issued to a *Client*, to be used to access a protected resource. In the context of this document an *access token* can be classed either as a *Simple Token* or a *Security Token*.
Assertion	In the context of this document an *assertion* is a package of information that contains identity and security information about a *subject*. An *assertion* can be classed as either a *Simple Token* or a *Security Token*.
Authorization Server	OAuth 2.0 terminology describing an entity that authenticates and authorizes a *Client* and issues an *access token* to that *Client* to be used in a request for a protected resource.
Client	OAuth 2.0 terminology describing a service, an end user through a web browser, or an end user through a native application that makes protected resource requests. For the purposes of this SIP a *Client* is capable of maintaining the confidentiality of their security credentials.
Header	The part of the *message* that contains additional information about the *message* beyond the data that is being exchanged.
Message	The structure used for exchanging data between the *Client* and the *Authorization Server* or between the *Client* and the *Resource Server*.
Policy Enforcement Point	SOA Platform Security Services SIP Proposal [NC3A RD-3140, 2011] terminology describing a logical entity or endpoint that enforces security policies. In the context of this document a *Resource Server* can perform the role of a *Policy Enforcement Point*.
Relying Party	SOA Platform Security Services SIP Proposal [NC3A RD-3140, 2011] terminology describing a logical entity or endpoint that relies upon security credentials presented in a *Security Token* or *Simple Token* in order to process a grant access to resources. In the context of this document a *Resource Server* performs the role of a *Relying Party*.

Resource Server	OAuth 2.0 terminology describing an entity that: hosts the protected resource; is capable of accepting and responding to protected resource requests; and, validates *access tokens* ensuring the protected resource request conforms to the access control policy.
Subject	An authenticated entity that can perform actions within a system.
Security Token	Security credentials used to represent a set of privileges issued to a *subject*. A *Security Token* contains cryptographic elements that may bind the *subject* to those privilege attributes or maintain the confidentiality of those privilege attributes. In OAuth 2.0 terminology an *access token* that conforms to this description can be classed as a *Security Token*.
Security Token Service	SOA Platform Security Services SIP Proposal [NC3A RD-3140, 2011] terminology describing a logical entity or endpoint that issues *Security Tokens*. In the context of this document an *Authorization Server* performs the role of a *Security Token Service*.
Simple Token	Security credentials used to represent a set of privileges issued to a *subject*. A *Simple Token* contains no cryptographic elements. In OAuth 2.0 terminology an *access token* that conforms to this description can be classed as a *Simple Token*.
Web Service Provider	REST messaging SIP proposal [NCIA TR/2012/SPW008423/11, 2013] terminology describing a service that produces data for other services. In the context of this document a *Resource Server* performs the role of a *Web Service Provider*.
Web Service Consumer	REST messaging SIP proposal [NCIA TR/2012/SPW008423/11, 2013] terminology describing a service or application that calls other services in order to retrieve data. In the context of this document a *Client* performs the role of a *Web Service Consumer*.

1.6 Goals

The REST security best practices document [NCIA TR/2012/SPW008423/17, in prep.] recommends adopting a generic security framework that can interwork with different types of identity management systems. [NCIA TR/2012/SPW008423/17, in prep.] also considers the different levels of protection mechanisms to mitigate against the risks of identified threats being exposed in different environments where RESTful web services can be deployed within NATO.

The following are the goals of this profile:

- Specify how to apply security for RESTful web services, based on OAuth 2.0, providing consistent authentication and authorization of entities within and beyond the enterprise.
- Provide recommendations on the protection mechanisms that need to be deployed, depending on the different scenarios where resources and RESTful web services need to be securely accessed.

1.7 Non-Goals

The following topics are outside the scope of this profile:

- Defining access control policies that will be enforced for making authorization decisions.
- Defining security credentials that represent the privileges of a *subject* to be used in making authorization decisions.
- Defining the mechanism for which a *Client* discovers the *Authorization Server*.
- Defining security functional and assurance requirements for specific information exchange scenarios.

1.8 Relationships to Other Profiles and Specifications

OAuth 2.0 is becoming the widely adopted standard for applying security to RESTful web services with a large number of implementations and services available. However, a number of specifications for OAuth 2.0 are still going through the Internet Engineering Task Force (IETF) processes prior to being released as Request for Comments (RFCs). This document specifies the use of some of these OAuth 2.0 Internet Drafts (ID), however, it must be noted that the IDs that are being specified within this document are already deemed sufficiently mature, where minimal change is expected. This is supported by the fact that a number of implementations that support the current technical specifications already exist.

1.8.1 Relevant NATO Core Enterprise Services (CES) documents

1.8.1.1 SIP proposal – REST messaging

[NCIA TR/2012/SPW008423/11, 2013].

1.8.1.2 SIP proposal – security services

[NC3A RD-3140, 2011].

1.8.1.3 Best practices in the application of securing RESTful web services

[NCIA TR/2012/SPW008423/17, in prep.].

1.8.1.4 SIP proposal – Enterprise Directory Service

[NC3A RD-3153, 2011] provides identity attributes used for authenticating and authorizing components of the REST security framework.

1.9 Normative References

1.9.1 OAUTH 2.0

1.9.1.1 The OAuth 2.0 authorization framework

http://tools.ietf.org/html/rfc6749.

1.9.1.2 The OAuth 2.0 authorization framework: bearer token usage

http://tools.ietf.org/html/rfc6750.

1.9.2 Transport

1.9.2.1 Hypertext transfer protocol – HTTP/1.1

http://tools.ietf.org/html/rfc2616.

1.9.2.2 The transport layer security (TLS) protocol Version 1.2

http://tools.ietf.org/html/rfc5246.

1.9.3 Authentication mechanisms

1.9.3.1 HTTP basic authentication

http://tools.ietf.org/html/rfc2617.

1.9.3.2 Mutual TLS (X.509 digital certificates)

http://tools.ietf.org/html/rfc5246.

1.9.3.3 Kerberos

http://tools.ietf.org/html/rfc4120

http://tools.ietf.org/html/rfc4559.

1.9.3.4 Assertions

1.9.3.4.1 Security assertion markup language (SAML) 2.0 (OASIS)

http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf.

1.9.3.5 Message security

1.9.3.5.1 XML (extensible markup language) encryption syntax and processing 1.0 (W3C)

http://www.w3.org/TR/xmlenc-core/.

1.9.3.5.2 XML digital signatures 1.0 (W3C)

http://www.w3.org/TR/xmldsig-core.

1.9.3.5.3 Secure/multipurpose internet mail extensions (S/MIME) Version 3.2 message specification

http://tools.ietf.org/html/rfc5751 .

1.10 Non-Normative References

1.10.1 OAuth 2.0

1.10.1.1 Assertion framework for OAuth 2.0

http://tools.ietf.org/html/draft-ietf-oauth-assertions-06.

1.10.1.2 SAML 2.0 bearer assertion profiles for OAuth 2.0

http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer-14.

1.10.2 Message security

1.10.2.1 JSON web encryption (JWE)

http://tools.ietf.org/html/draft-ietf-jose-json-web-encryption-06.

1.10.2.2 JSON web signature (JWS)

http://tools.ietf.org/html/draft-ietf-jose-json-web-signature-06.

2 SIP DEFINITION

2.1 Subject

More and more web services are being implemented in the REST architectural style. As such, for new services being implemented on NATO CIS, some will be designed based on the REST architectural style. These NATO-owned RESTful web services require manageable and scalable security mechanisms. The purpose of the REST security services is to ensure that an entity (user or service) requesting access to a protected resource is correctly authenticated and authorized.

NCI Agency conducted a comprehensive literature study based on academic research and industry best practices to determine the best approach for applying security to RESTful web services. The results are documented in [NCIA TR/2012/SPW008423/17, in prep.].

The study highlighted that for RESTful web services there are no natively built-in security features. Originally simple object access protocol (SOAP)-based web services did not provide any security features, however, subsequently a security stack of standards, including WS-Trust and WS-Security, have been defined and implemented to layer security on top of SOAP. REST has no comparable standards to WS-Security. Security standards for SOAP-based web services are well-defined for providing trust between services, authentication of services, authorization of services and end-to-end *message* security. NCI Agency has undertaken extensive practical research in further refining these open standards for applying security to SOAP-based web services within a NATO context, described in [NC3A RD-3140, 2011].

In order to provide a comparable security framework for REST, [NCIA TR/2012/SPW008423/17, in prep.] recommends adopting the OAuth 2.0 open standards specifications. OAuth 2.0 is profiled in this document for providing an authentication and authorization framework that can be implemented for protecting RESTful web services deployed in NATO environments. OAuth 2.0 offers a RESTful WS-Trust/security token service (STS) end-point for obtaining an *access token* (abstracting away the burden of trust and identity management from the *Client*) and WS-Security-like mechanisms for applying the *access token* to the access request and securing the access request.

The study also analysed the protection mechanisms that would be required based on the security measures that must be considered for protecting resources within NATO CIS and other federated CIS. A study into industry best practices illustrated that there were a number of mechanisms being utilized, ranging from TLS with HTTP basic authentication to keyed hash message authentication code (HMAC) canonicalized HTTP *headers*. Many of these protection mechanisms provide adequate security in the domains where they are protecting resources, but the study concluded that such approaches were stove-pipe-centric and not scalable in a heterogeneous environment.

As OAuth 2.0 is a generic security framework, it allows support for the different types of security mechanisms and interworking with currently deployed identity management systems, therefore, promoting the broadest possible range of interoperability for applying security to RESTful web services within and beyond the NATO enterprise.

It is paramount that the security architecture that is required to protect NATO CIS is assessed and understood with all pertinent risks mitigated in order to acquire the full benefits of the REST architectural style in a secure manner. NATO has not adopted any concept of assurance levels. This document references [NIST Special Publication 800-63-1, 2011] to provide recommendations and guidelines based on the levels of assurance[2] required for identified scenarios relevant to NATO (described in REST security best practices document [NCIA TR/2012/SPW008423/17, in prep.]).

2.1.1 OAuth 2.0 high-level overview

OAuth 2.0 is an authentication and authorization framework for securing access to protected resources through RESTful web services. OAuth 2.0 is currently active under the IETF Web Authorization Protocol Charter, where the core framework has just gained RFC status ([IETF RFC 6749, 2012]).

The OAuth 2.0 REST security framework allows for long-term credentials (password or X.509 digital certificate) to be replaced with short-term security credentials (token), providing limited access that can be managed and revoked separately from the long-term security credentials. This approach abstracts away the burden of trust and identity management from the *Client* to the organization identity management system.

OAuth 2.0 consists of three main components (described in Section **Error! Reference source not found.**):

- Client
- Authorization Server
- Resource Server.

Figure 1 represents the logical view for the components of the OAuth 2.0 REST security framework.

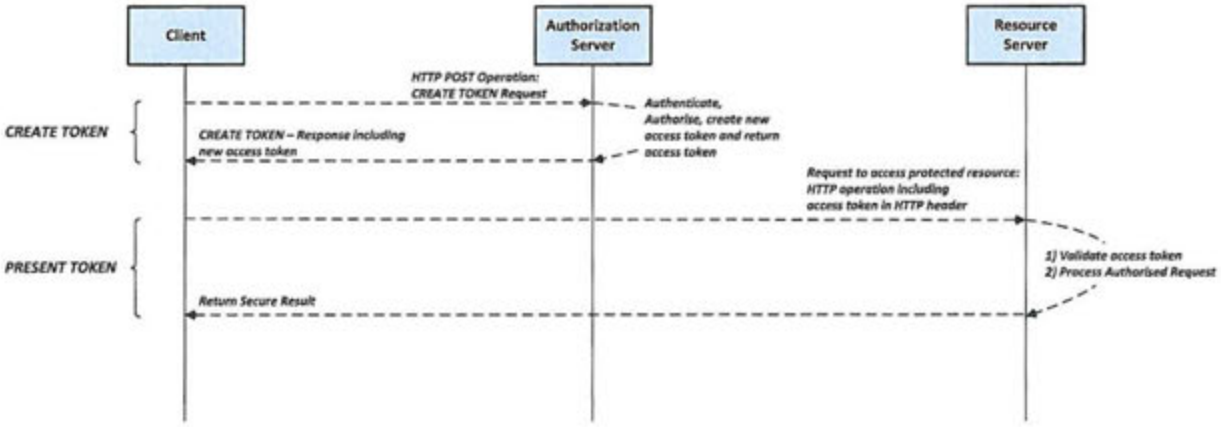

Figure 1 OAuth 2.0 logical components sequence flow diagram

Note that this SIP does not make any recommendations regarding the deployment of these components. An *Authorization Server*, for example, can provide a token endpoint as a stand-alone component or as the same server as the *Resource Server*.

---

[2] [NIST Special Publication 800-63-1, 2011] specifies four levels of assurance, based on the risks and likelihood of threats being exposed as a result of authentication errors; and, the associated consequences as a result of loss of confidentiality and integrity.

Figure 1 also introduces the two fundamental concepts (Create Token and Present Token) supporting protected resource requests.

2.1.1.1 Create token

The *Authorization Server* offers a token end-point as a collection resource. The Create Token concept is based on a *Client* requesting a new set of security credentials (a token resource) from the *Authorization Server* to be used in subsequent requests for a protected resource(s). The *Client* provides security credentials in the *header* of this request to the *Authorization Server*. The *Authorization Server* verifies the security credentials and authenticates the *Client* prior to issuing a new *access token*.

The request for an access token is a HTTP POST request. REST best practices documented in [NCIA RoI Best Practices in the use of the REST Architectural Style, 2012] states that the request uniform resource identifier (URI) used in a POST request is a resource which is considered to be a collection resource. The *Authorization Server* token end-point is classed as a collection resource as the URI for the final resource (*access token*) is unknown. That is the new *access token* that is to be issued (created) is dependent on the identity of the *Client* at the time the request is made, the length of time the *access token* is valid and the `scope` of the request.

2.1.1.2 Present token

The methods that can be invoked to manipulate the state of resources published by a RESTful web service are restricted to be only the methods GET, PUT, DELETE, POST etc. as defined in HTTP ([IETF RFC 2616, 1999]). The Present Token concept supports all of the HTTP methods used for accessing a protected resource.

In a request for a protected resource (for example a request to retrieve a resource or a request to delete a resource) the *Client* presents the *access token* to the *Resource Server* hosting the protected resource. The *Client* provides the *access token* in the *header* of the HTTP request. The *Resource Server* validates the *access token*. If the *access token* is successfully validated the *Resource Server* processes the authorized request and the result is returned to the *Client*.

3 REST SECURITY FRAMEWORK SUPPORTING INFRASTRUCTURE

3.1 Message Structure

The overall *message* structure for RESTful web services is defined in a different SIP proposal, i.e. [NCIA TR/2012/SPW008423/11, 2013].

The OAuth 2.0 specifications permit the following elements to be used for passing security credentials:

- [IETF RFC 2617, 1999] HTTP entity-headers `Authorization` and `WWW-Authenticate`
- [W3C HTML 4.01, 1999] form encoded body parameter within the HTTP request/response entity-body
- [IETF RFC 3986, 2005] URI query parameter.

A SIP-compliant *Client* SHALL provide security credentials in the [IETF RFC 2617, 1999] `Authorization` *header*.

A SIP-compliant *Authorization Server* and *Resource Server* SHALL support security credentials in the [IETF RFC 2617, 1999] `Authorization` *header* in requests received from the *Client*.

A SIP-compliant *Authorization Server* and *Resource Server* SHOULD support security credentials in the [W3C HTML 4.01, 1999] form encoded body parameter in requests received from the *Client*.

A SIP-compliant *Authorization Server* SHALL provide security credentials in the [IETF RFC 2617, 1999] `WWW-Authenticate` *header*.

A SIP-compliant *Client* SHALL support security credentials in the [IETF RFC 2617, 1999] `WWW-Authenticate` *header* in responses received from the *Authorization Server*.

A SIP-compliant *Client* SHOULD support security credentials in the [W3C HTML 4.01, 1999] form encoded body parameter in responses received from the *Authorization Server*.

The request URI SHALL NOT contain sensitive information such as identity-related attributes as web servers and intermediaries will normally log the URI.

A REST security framework component compliant with this SIP SHALL NOT support the URI query parameter ([IETF RFC 3986, 2005]) element for carrying security credentials.

A REST security framework component compliant with this SIP SHALL NOT store security credentials in cookies.

3.2 Cryptography

Public key cryptography SHALL be used in accordance with the NATO public key infrastructure (PKI) (NPKI) to establish trust between the REST security framework components and maintain the confidentiality and integrity of NATO CIS.

This profile depends on the following PKI requirements:

- Each *Resource Server* SHALL be issued with a X.509 v.3 digital public/private key pair in accordance with the NCertP ([NAC AC/322-D(2004)0024-REV2-ADD1, 2010]).
- Each *Authorization Server* SHALL be issued with a X.509 v.3 digital public/private key pair in accordance with the NCertP ([NAC AC/322-D(2004)0024-REV2-ADD1, 2010]).
- In the use case where the identity management system is the NPKI all *Client*s SHALL be issued with a X.509 v.3 digital public/private key pair, in accordance with the NCertP ([NAC AC/322-D(2004)0024-REV2-ADD1, 2010]).

- In all other use cases *Resource Servers* and *Authorization Servers* SHALL NOT rely on *Clients* having a X.509 v.3 digital public/private key pair.
- Only NATO-approved algorithms SHALL be used for digest and encryption algorithms.

3.2.1 Protection mechanisms

The REST security best practice document [NCIA TR/2012/SPW008423/17, in prep.] classified the identified scenarios into Level 2 or Level 3 assurance environments according to [NIST Special Publication 800-63-1, 2011].

In a Level 2 environment end-to-end *message* level security is NOT REQUIRED.

In a Level 2 environment point-to-point security, providing confidentiality and integrity between the *Client* and the *Resource Server* or the *Client* and the *Authorization Server*, SHALL be protected with HTTP/TLS ([IETF RFC 5246, 2008]).

In a Level 3 environment where intermediaries are not trusted to not alter the *message*; and, *message* (*message* parts) security needs to be provided outside of the transport layer, *message* level security is REQUIRED.

In a Level 3 environment digital signatures SHALL be used to protect the integrity of the *message* (*message* parts).

In a Level 3 environment digital signatures SHALL be used to provide non-repudiation of *message* origination.

In a Level 3 environment digital encryption MAY be used to provide confidentiality of the *message* (*message* parts).

3.2.2 Validation

In the case where digital signatures are used for providing authentication, integrity and/or non-repudiation, the entity performing the check SHALL be conformant with [IETF RFC 5280, 2008], specifically:

- Verifying proof of possession of the entity's private key
- Validating signed attributes, such as signing time or nonce
- Validating the full certificate path
- Validating the certificate's revocation status.

In the case where digital encryption is used for providing confidentiality, the entity performing the check SHALL be conformant with [IETF RFC5280, 2008], specifically:

- Validating the full certificate path
- Validating the certificate's revocation status.

3.2.3 Signature and encryption open standards specifications

The REST Messaging SIP proposal [NCIA TR/2012/SPW008423/11, 2013] places no constraints on the type of data that can be exchanged between *Clients* and *Resource Servers* or *Clients* and *Authorization Servers*; however, it recommends XML or JSON.

XML data that requires digital signing SHALL be signed in accordance with [W3C XML-Signature, 2002].

XML data that requires digital encryption SHALL be encrypted in accordance with [W3C XML-Encryption, 2002].

Data encapsulated in MIME that requires digital signing SHALL be signed in accordance with [IETF RFC 5751, 2010].

Data encapsulated in MIME that requires digital encryption SHALL be encrypted in accordance with [IETF RFC 5751, 2010].

Digital signatures and digital encryption standards for JSON are currently in ID status. However, it is RECOMMENDED to follow [IETF JOSE JSON Web Signature, 2012], [IETFJOSE JSON Web Encryption, 2012] in cases where JSON data requires digital signatures and/or digital encryption, respectively.

3.3 Authentication mechanisms

The *Authorization Server* SHALL authenticate the credentials provided by the *Client* in order to establish the identity of the entity for which the request is being made.

A SIP-compliant *Client* and *Authorization Server* SHALL support the authentication mechanisms specified in the normative specifications listed in Section 1.9.3.

3.4 System/Enterprise Identity Management Systems

Identity management systems provide the infrastructure, policies, procedures and mechanisms for identification and authentication between the entities (components) within the REST security framework. An identity management system enables entities to present identity information attributes to one another and to authenticate to one another by validating the identity information attributes. There are four main types of identity management systems, differentiated by the technologies deployed for representing and storing identity information. These are:

- Username- and password-based systems

These are widely deployed despite the known security limitations.

- X.509 certificates managed from a PKI

A public-private key pair is used where a certificate authority acts as the identity provider component of the identity management system by certifying the public key.

- Kerberos

Based on shared symmetric keys and is the native authentication mechanism for Windows with the active directory domain services acting as the identity provider component of the identity management system.

- Token-based systems

These comprise a wide variety of systems that use passwords, X.509 or Kerberos to authenticate the entity prior to issuing a *security token* to be used for subsequent use in accessing resources. Tokens may (*security token*) or may not (*simple token*) contain cryptographic elements. An example of a *security token* would be a signed SAML assertion. An example of a *simple token* is an OAuth 2.0 `bearer token`.

The *Authorization Server* SHALL support interworking with all types of identity management systems.

The *Authorization Server* SHALL support the three authentication modes specified in Section 2.3.1 of [NC3A RD-3153, 2011] to authenticate with the Enterprise Directory Service in the cases where the identity attributes, used for authentication and authorization, are stored in the Enterprise Directory Service.

3.5 Assertions

Assertions are used to facilitate interworking with other identity management systems. An assertion is used as an alternative authentication mechanism for the *Authorization Server* to validate and verify in exchange for *access tokens*. The framework for using *assertions* in OAuth 2.0 is specified in [IETF WAP Assertion Framework for OAuth 2.0, 2012]. The following are the format and structure rules for an *assertion*:

- The assertion SHALL contain an Issuer that is the entity that issued the assertion.
- The assertion SHALL contain a Subject which identifies the entity that is requesting an access token.
- The assertion SHALL contain an Audience which is the Authorization Server token end-point URI.
- The assertion SHALL contain an Expires at date and time.
- The assertion SHALL contain an Issued at data and time.
- The assertion SHALL contain a unique identifier.
- The Authorization Server SHALL verify and validate the digital signature of the assertion.

In the case where the security profile requires that *assertions* are to be used for requesting *access tokens*, the *Authorization Server* SHALL support the following two types of *Assertions* as a minimum:

- SAML Assertion ([OASIS SAML v2.0 Core, 2005])

The mechanism for obtaining the SAML assertion SHALL be conformant with the specifications profiled in [NC3A RD-3140, 2011].

The format and structure of a SAML assertion SHALL be conformant with the specifications profiled in [NC3A RD-3140, 2011].

- JSON Web Token (JWT) Assertion ([IETF WAP JSON Web Token, 2012])

The mechanism for obtaining the JWT assertion is out of scope for this SIP proposal.

3.6 Access Token structure

The current definition for an OAuth 2.0 *access token* (as specified in [IETF RFC 6749, 2012]) is '`a string representing an authorization issued to the client`'. As such, there is no defined structure for an *access token* within the core OAuth 2.0 specification [IETF RFC 6749, 2012]. The type of *access token*, security information within that *access token* and the methods used by the *Resource Server* to validate the *access token* is to be defined by accompanying OAuth 2.0 specifications.

Currently, OAuth 2.0 has only specified the use for a `bearer token` as an *access token* ([IETF RFC 6750, 2012]).

Further iterations of this SIP will be developed when other types of *access tokens* have been specified within the IETF.

3.6.1 Simple token

As a minimum, the *Authorization Server* and the *Resource Server* SHALL support the OAuth 2.0 `bearer token`, specified in [IETF RFC 6750, 2012].

An OAuth 2.0 `bearer token` is a type of *simple token*.

In the case where a *Client* is to be issued a `bearer token`, the HTTP transaction between the *Client* and the *Authorization Server* end-points SHALL be protected with TLS.

In the case where a *Client* is to present a `bearer token` to the *Resource Server* in a request to access a protected resource, the *Client* and the *Resource Server* end-points SHALL be protected with TLS.

This SIP does not define the structure of an OAuth 2.0 `bearer token`. However the following constraints on its structure SHALL be honoured:

- The `bearer token` SHALL be self-contained to allow the *Resource Server* to validate the `bearer token`.
- The `bearer token` SHALL have a short lifetime for which the length of time is specified dependent on the level of assurance required to protect the CIS.
- The `bearer token` SHALL contain an audience restriction, scoping their use to the intended *Resource Server* or set of *Resource Servers*.

3.6.2 Security token

In the case where a high level of assurance and robustness is required for protecting access to the requested resources, a security token SHALL be used.

A security token requires the *Client* to bind key material to the *access token* in order for the *Client* to provide proof of possession. There are currently two specifications in ID status:

- Holder-of-the-Key Token Usage ([IETF WAP OAuth 2.0 Holder-of-the-Key Token Usage, 2012])
- MAC (message authentication code) Token Usage ([IETF WAP OAuth 2.0 Message Authentication Code Token Usage, 2012]).

Implementations exist supporting both of these draft specifications; however, there is currently no consensus within the OAuth Working Group Charter as to the preferred draft specification to advance to RFC status. As such, this SIP does NOT RECOMMEND support for either of the two specifications [IETF WAP OAuth 2.0 Holder-of-the-Key Token Usage, 2012] or [IETF WAP OAuth 2.0 Message Authentication Code Token Usage, 2012] for creating, distributing or using a *security token*.

It is RECOMMENDED that the SIP is updated when the structure of a *security token* is specified by the OAuth Working Group Charter.

As an interim approach, a *Client* MAY present a `bearer token` to the *Resource Server* where both end-points are mutually authenticated via TLS.

1.1.1 Refresh tokens

OAuth 2.0 specifies the process where an *access token* can be obtained from the *Authorization Server* in exchange for a `refresh token`. REST Security best practices ([NCIA TR/2012/SPW008423/17, in prep.]) do not recommend the use of refresh tokens.

An *Authorization Server* SHALL NOT issue `refresh tokens`.

An *Authorization Server* SHALL NOT accept a `refresh token` in a request for an *access token*.

4 REST SECURITY FRAMEWORK

4.1 Introduction

REST is an architectural style defined by the constrained and consistent use of a number of protocols. For that reason there is no single defined service interface, other than the uniform interfaces of HTTP specified in [IETF RFC 2616, 1999] for GET, PUT, POST, DELETE etc. As such, no new interfaces are provided by this SIP over and above those defined by HTTP.

OAuth 2.0 offers a REST security framework where the following two concepts are supported:

- Create Token – An *Authorization Server* creates a new *access token* to be issued to the requesting *Client*.
- Present Token – A *Resource Server,* hosting the requested protected resource, validates the *access token* that was provided by the *Client* in the request.

The SIP proposal will cover the inputs, outputs and errors for HTTP operations between a *Client*:

- Requesting an access token from the Authorization Server
- Using the access token to make a protected access request to the Resource Server.

The type of input to support the Create Token concept will depend on the `grant type` and the output will depend on the type of *access token* being requested.

The type of input to support the Present Token concept for accessing the protected resource will depend on the type of *access token*.

4.2 Authorization Server

The *Authorization Server* supports the Create Token concept by providing a token end-point (HTTP collection resource identified by the request URI) for a *Client* to make a request for a new *access token* that can be used in subsequent protected resource requests.

4.2.1 POST operation

The *Client* SHALL use HTTP POST operations when requesting an *access token* from the *Authorization Server*.

A new *access token* is issued to the *Client* on a successful validation of the security credentials provided in the request.

4.2.2 Input

The input SHALL be an HTTP POST request to the *Authorization Server* containing the authentication security credentials or *assertion* security credentials for the entity making the request and the type of authorization grant.

The input MAY contain a `scope` HTTP entity-header parameter with values equating to the privileges requested along with the *access token*.

4.2.2.1 Credentials

Table 1 specifies the types of authentication technologies supported within the REST security framework. Table 1 also specifies the REQUIRED location (for each type of authentication technology) within the request for storing security credentials for the *Authorization Server* to use for authenticating the *Client*.

Table 1
Supported authentication technologies and REQUIRED location for Client security credentials in the request to the Authorization Server

Authentication technologies	Security credentials
Username and password	The *Client* SHALL present authentication credentials in the HTTP `Authorization` *header* as specified in [IETF RFC 2616, 1999]. The `auth-scheme` parameter SHALL have a value of `Basic`.
Kerberos	The *Client* SHALL present authentication credentials in the HTTP `Authorization` *header* as specified in [IETF RFC 2617, 1999]. The `auth-scheme` parameter SHALL have a value of `Negotiate`.
X.509 v3.0 public/private key pair	The *Client* SHALL present authentication credentials in the digital certificate and cryptographic information as specified in [IETF RFC 5246, 2008].
Assertions	The *assertion* SHALL be Base64 URL encoded and added to the HTTP request parameter `assertion` within the `Authorization` *Header*.

4.2.2.2 Grant type

REST Security best practices [NCIA TR/2012/SPW008423/17, in prep.] discuss the recommended authorization grant types to be supported by NATO CIS.

The HTTP request SHALL contain a parameter `grant_type` that indicates the type of authorization grant. Table 2 specifies the REQUIRED values.

Table 2 REQUIRED grant types

Grant type	Value
Authorization code grant	`authorization_code`
Client credentials grant	*Client*`_credentials`
Assertion (SAML)	`urn:ietf:params:oauth:client-assertion-type:saml2-bearer`
Assertion (JWT)	`urn:ietf:params:oauth:client-assertion-type:jwt-bearer`

4.2.2.3 Scope

A *Client* can request certain privileges and/or the *Authorization Server* can return the privileges associated with the *access token* request. The `scope` parameter SHALL provide a list of comma-separated values that equate to the requested privileges that are enforced by the *Authorization Server*.

In the case of a SAML assertion the privileges or claims can be supported within the <AttributeStatement/> element.

It is outside the scope of this SIP to define the privileges and the access control policies that are to be enforced.

4.2.3 Output

The output SHALL be an HTTP POST response that contains an *access token*.

The *access token* SHALL be encoded in the `WWW-Authenticate` *header* with the `auth-scheme` parameter value set as the type of *access token*.

As a minimum, the OAuth 2.0 *access token* SHALL be in the format of a `bearer token` as specified in [IETF RFC 6750, 2012].

The `bearer token` SHALL be encoded in the `WWW-Authenticate` *header* with the `auth-scheme` parameter value set as `Bearer`.

4.2.4 Errors

The REST Messaging SIP proposal [NCIA TR/2012/SPW008423/11, 2013] specifies that errors SHALL be conveyed in accordance with [IETF RFC 2616, 1999] status codes.

There are many security related errors that can occur based on a request from a *Client* for an *access token* to be used for accessing resources hosted by a *Resource Server*.

A failure as a result of authentication SHALL result in a status code of 401 being returned to the *Client*.

A failure as a result of authorization SHALL result in a status code of 403 being returned to the *Client*.

A failure as a result of the *Client* using an HTTP verb other than POST SHALL result in a status code of 405 being returned to the *Client*.

Any other failure for processing the request as a result of an error occurring within the *Authorization Server* SHALL result in a 5xx status code being returned to the *Client*.

Additional information that can be used to impact the confidentiality, integrity or availability of the NATO CIS SHALL NOT be provided in the error response.

4.2.5 Additional security considerations

The Create Token concept allows for replacing a long-term security credential, such as a password, with a short-term security credential (*access token*). The Create Token concept represents authentication and authorization of the requesting entity and issuing an *access token* based on the level of privileges for that authenticated requesting entity.

For a Level 2 environment username and password technologies are not permitted. In the case where existing implementations have security applications based on username and password the connection between the *Client* and the *Authorization Server* SHALL be secured with HTTP/TLS using an approved encryption algorithm.

It is RECOMMENDED that validation/retrieval of identity attributes for the purposes of authentication and authorization is performed using the organization/enterprise identity management system.

Use of stove-pipe system provided homogeneous identity management systems SHOULD NOT be used.

In the case when an *access token* is a `bearer token`, confidentiality and integrity of data sent between a *Client* and the *Authorization Server* SHALL be protected by using HTTP/TLS using an approved encryption algorithm.

In the case where an identity provider component of the identity management system is an Active Directory the *Client* SHALL authenticate to the *Authorization Server* using the Kerberos mechanism as defined in HTTP SPNEGO ([IETF RFC 4559, 2006]). The Kerberos delegation mechanism can be used where a web application acting on behalf of a client can request an *access token*.

In the case where an identity provider component of the identity management system is the NPKI, the *Authorization Server* SHALL authenticate the *Client's* X.509 v3.0 Digital Certificate.

In the case where the identity provider component of an identity management system is a SAML token issuing STS, the *Client* SHALL present a digitally signed SAML assertion for authenticating and authorizing to the *Authorization Server*. This mechanism supports federated identities between mutually trusting security domains (Level 3 environment).

For a Level 3 environment where one or more intermediaries are deployed between the *Client* and the *Authorization Server* it is RECOMMENDED to use digital encryption to maintain the confidentiality of the *access token*.

The lifetime of an *access token* SHALL be configurable dependent on the NATO environment.

According to [NIST Special Publication 800-63-1, 2011] an *access token* lifetime of no greater than 12 hours for a Level 2 environment is RECOMMENDED.

According to [NIST Special Publication 800-63-1, 2011] an *access token* lifetime of no greater than 2 hours for a Level 3 environment is RECOMMENDED.

4.3 Resource Server

The Present Token concept supports consistent and compliant use of the uniform interface offered by HTTP operations for accessing a resource as specified in the REST Messaging SIP proposal [NCIA TR/2012/SPW008423/11, 2013]. The *Client* makes a protected access request for the resource to the *Resource Server* (`authority` part referred to within the request URI) presenting the *access token* in the *header* of the HTTP request. The *access token* is self-contained which allows the *Resource Server* to validate the *access token* without reference to another service (or previous state). If the *access token* is successfully validated, the *Resource Server* processes the authorized request and the result is returned to the *Client*.

4.3.1 Operations

The *Client* SHALL support the HTTP operations GET, DELETE, PUT, POST, HEAD and OPTIONS as defined in [IETF RFC 2616, 1999] for requesting access to protected resources.

4.3.2 Input

The input SHALL be an HTTP request to the *Resource Server*, hosting the protected resources, that contains the *access token* obtained from the *Authorization Server* (as specified in Section 4.2.3).

The *access token* SHALL be encoded in the HTTP `Authorization` entity-header.

The `auth-scheme` parameter for the HTTP `Authorization` entity-header SHALL be that specified to indicate the type of *access token*.

As a minimum, the `bearer token` SHALL be presented to the *Resource Server* as specified in [IETF RFC 6750, 2012].

The `auth-scheme` parameter for the HTTP `Authorization` *header* SHALL be `bearer` when presenting the `bearer token`.

4.3.3 Output

The output of the request SHALL be an HTTP response containing an HTTP return code based on the operation of the HTTP request.

4.3.4 Errors

The REST Messaging SIP proposal [NCIA TR/2012/SPW008423/11, 2013] specifies that errors SHALL be conveyed in accordance with [IETF RFC 2616, 1999] status codes.

There are many security-related errors that can occur based on a request from a *Client* for accessing resources hosted by a *Resource Server*.

If a request for a protected resource does not contain an HTTP `Authorization` *header*, the *Resource Server* SHALL return a status code of 401 to the *Client*.

A failure as a result of validating an *access token* due to the lifetime of that *access token* being exceeded SHALL result in a status code of 401 being returned to the *Client*.

In the cases where a *Client* receives a 401 status error code, that *Client* SHALL request an *access token* from the *Authorization Server* as specified by the Create Token concept in Section 4.2.

Any other failure as a result of validating an *access token* SHALL result in a status code of 403 being returned to the *Client*.

A failure as a result of the *Client* using an HTTP verb that is not allowed SHALL result in a status code of 405 being returned to the *Client*.

Any other failure as a result of processing the request as a result of an error occurring within the *Resource Server* SHALL result in a 5xx status code being returned to the *Client*.

No additional information that can be used to impact the confidentiality, integrity or availability of the NATO CIS SHALL be provided in the error response.

4.3.5 Additional security considerations

The Present Token concept represents the validation of an *access token*, presented by a *Client* when requesting access to a protected resource hosted by the *Resource Server*. The validation of the *access token* is carried out by the *Resource Server* and negates the need for the *Resource Server* to understand all the authentication and authorization mechanisms that may be deployed within the CIS and potentially federated CIS.

In the case when an *access token* is a `bearer token`, the confidentiality and integrity of data sent between a *Client* and the *Resource Server* SHALL be protected by using HTTP/TLS using an approved encryption algorithm.

For a Level 2 environment the *Client* MAY provide proof of possession of the *access token* by mutually authenticating with the *Resource Server* via TLS using an approved encryption algorithm.

For a Level 3 environment it is RECOMMENDED for the *Client* to provide proof of possession of the *access token* by digitally signing the message.

For a Level 3 environment where integrity of the *message* is maintained outside of the HTTP/TLS session, digital signatures SHALL be used.

For a Level 3 environment use of signed timestamp, nonce or other unique verifiable identifier attributes within the digital signature is REQUIRED to mitigate against the risk of *access token* replay attacks.

For a Level 3 environment where one or more intermediaries are deployed between the *Client* and the *Resource Server* it is RECOMMENDED to use digital encryption to maintain the confidentiality of the *message*.

5 REFERENCES

[IETF JOSE JSON Web Encryption, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, "JSON Web Encryption (JWE)", Internet Draft, M. Jones, E. Rescorla, J. Hildebrand, at http://datatracker.ietf.org/doc/draft-jones-json-web-encryption/, 6 November 2012, viewed 12 December 2012.

[IETF JOSE JSON Web Signature, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, "JSON Web Signature (JWS)", Internet Draft, M. Jones, J. Bradley, N. Sakimura, at http://datatracker.ietf.org/doc/draft-jones-json-web-signature/, 6 November 2012, viewed 12 December 2012.

[IETF RFC 2119, 1997]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 2119, "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, at http://tools.ietf.org/html/rfc2119, March 1997, viewed 12 December 2012.

[IETF RFC 2616, 1999]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 2616, "Hypertext Transfer Protocol -- HTTP/1.1", R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, at http://tools.ietf.org/html/rfc2616, June 1999, viewed 12 December 2012.

[IETF RFC 2617, 1999]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 2617, "HTTP Authentication: Basic and Digest Access Authentication", J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart, at http://tools.ietf.org/html/rfc2617, June 1999, viewed 12 December 2012.

[IETF RFC 3986, 2005]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 3986, "Uniform Resource Identifier (URI): Generic Syntax", T. Berners-Lee, R. Fielding, L. Masinter, at http://tools.ietf.org/html/rfc3986, January 2005, viewed 12 December 2012.

[IETF RFC 4559, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4559, "SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows", K. Jaganathan, L. Zhu, J. Brezak, at http://tools.ietf.org/html/rfc4559, June 2006, viewed 12 December 2012.

[IETF RFC 5246, 2008]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 5246, "The Transport Layer Security (TLS) Protocol Version 1.2", T. Dierks, E. Rescorla, at http://tools.ietf.org/html/rfc5246, August 2008, viewed 12 December 2012.

[IETF RFC 5280, 2008]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 5280, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, at http://www.ietf.org/rfc/rfc5280.txt, May 2008, viewed 12 December 2012.

[IETF RFC 5751, 2010]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 5751, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", B. Ramsdell, S. Turner, at http://www.ietf.org/rfc/rfc5751.txt, January 2010, viewed 12 December 2012.

[IETF RFC 6749, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 6749, "The OAuth 2.0 Authorization Framework", D. Hardt, at http://tools.ietf.org/html/rfc6749, October 2012, viewed 12 December 2012.

[IETF RFC 6750, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, Request for Comments 6750, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", M. Jones, D. Hardt, at http://tools.ietf.org/html/rfc6750, October 2012, viewed 12 December 2012.

[IETF WAP Assertion Framework for OAuth 2.0, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, "Assertion Framework for OAuth 2.0", Internet Draft, B. Campbell, C. Mortimore, M. Jones, Y. Goland, at http://datatracker.ietf.org/doc/draft-ietf-oauth-assertions/, 26 November 2012, viewed 12 December 2012.

[IETF WAP JSON Web Token, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, "JSON Web Token (JWT)", Internet Draft, M. Jones, J. Bradley, M. Jones, at http://datatracker.ietf.org/doc/draft-ietf-oauth-json-web-token/, 6 November 2012, viewed 12 December 2012.

[IETF WAP OAuth 2.0 Holder-of-the-Key Token Usage, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, "The OAuth 2.0 Authorization Framework: Holder-of-the-Key Token Usage", Internet Draft, J. Bradley, P. Hunt, T. Nadalin, H. Tschofenig, at http://datatracker.ietf.org/doc/draft-tschofenig-oauth-hotk/, 16 July 2012, viewed 12 December 2012.

[IETF WAP OAuth 2.0 Message Authentication Code Token Usage, 2012]:
Internet Engineering Task Force (on-line), http://www.ietf.org, "OAuth 2.0 Message Authentication Code (MAC) Tokens", Internet Draft, J. Richer, P. Hunt, H. Tschofenig, at http://datatracker.ietf.org/doc/draft-ietf-oauth-v2-mac/, 28 November 2012, viewed 12 December 2012.

[NAC AC/322-D(2004)0024-REV2-ADD1, 2010]:
North Atlantic Council Document AC/322-D(2004)0024-REV2-ADD1, "NATO Public Key Infrastructure (NPKI) Certificate Policy", NAC, Brussels, Belgium, 30 March 2010 (NATO Unclassified).

[NAC AC/322-N(2012)0092, 2012]:
North Atlantic Council Note AC/322-N(2012)0092, "C3 Classification Taxonomy", NAC, Brussels, Belgium, 24 May 2012 (NATO Unclassified).

[NC3A RD-3140, 2011]:
NATO Consultation, Command and Control Agency Reference Document 3140, "Security Services Service Interface Profile Proposal, Version 1.1." *(provisional title)*, NC3A Core Enterprise Services Team, NC3A, The Hague, Netherlands, unpublished draft dated November 2011 (NATO Unclassified).

[NC3A RD-3153, 2011]:
NATO Consultation, Command and Control Agency Reference Document 3153, "Enterprise Directory Services Interface Profile Proposal. Version 1.0" *(provisional title)*, NC3A Core Enterprise Services Team, NC3A, The Hague, Netherlands, unpublished draft dated March 2011 (NATO Unclassified).

[NCIA RoI Best Practices in the Use of the REST Architectural Style, 2012]:
NATO Communications and Information Agency, "Best Practices in the Use of the REST Architectural Style", Record of Investigation, A. Ross, A. Tucker, NCI Agency, The Hague, Netherlands, November 2012 (NATO Unclassified).

[NCIA TR/2012/SPW008423/11, 2013]:
NATO Communications and Information Agency Technical Report TR/2012/SPW008423/11, "REST Messaging Service Interface Profile Proposal", A. Ross, A. Tucker, NCI Agency, The Hague, Netherlands, April 2013 (NATO Unclassified).

[NCIA TR/2012/SPW008423/17, in prep.]:
NATO Communications and Information Agency Technical Report TR/2012/SPW008423/17, "Best Practices for RESTful Web Services Security" *(provisional title)*, A. Ross, D. Gujral, D. Marco-Mompel, A. Tucker, NCI Agency, The Hague, Netherlands, in preparation (NATO Unclassified).

[NIST Special Publication 800-63-1, 2011]:
National Institute of Standards and Technology (on-line), http://csrc.nist.gov, NIST Special Publication 800-63-1, "Electronic Authentication Guideline", W.E. Burr, Donna F. Dodson, E.M. Newton, R.A. Perlner, T. Polk, S. Gupta, E.A. Nabbus, at http://csrc.nist.gov/publications/nistpubs/800-63-1/SP-800-63-1.pdf, December 2011, viewed 12 December 2012.

[OASIS SAML v2.0 Core, 2005]:
Organization for the Advancement of Structured Information Standards (on-line), http://www.oasis-open.org, OASIS Standard saml-core-2.0-os, "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0", at http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf, 15 March 2005, viewed 12 December 2012.

[W3C HTML 4.01, 1999]:
World Wide Web Consortium (on-line), http://www.w3.org, "HTML 4.01 Specification", W3C Recommendation, D. Raggett, I. Jacobs, A. Le Hors, at http://www.w3.org/TR/1999/REC-html401-19991224, 24 December 1999, viewed 12 December 2012.

[W3C XML-Signature, 2002]:
World Wide Web Consortium (on-line), http://www.w3.org, "XML-Signature Syntax and Processing", W3C Recommendation, D. Eastlake, J. Reagle, D. Solo, at http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/, 12 February 2002, viewed 12 December 2012.

6 ABBREVIATIONS

CES	Core Enterprise Services
CIS	Communications and information system
HMAC	Hash message authentication code
HTTP	Hypertext transfer protocol
ID	Internet Draft
IETF	Internet Engineering Task Force
JSON	JavaScript object notation
JWE	JSON encryption
JWS	JSON web signature
JWT	JSON web token
MAC	Message authentication code
MIME	Multipurpose Internet mail extension
NCI	NATO Communications and Information
NNEC	NATO Network Enabled Capability
NPKI	NATO public key infrastructure
OASIS	Organization for the Advancement of Structured Information Standards
PKI	Public key infrastructure
REST	Representational state transfer
RFC	Request for comments
S/MIME	Secure MIME
SAML	Security assertion markup language
SIP	Service Interface Profile
SOA	Service-oriented architecture
SOAP	Simple object access protocol
STS	Security token service
TLS	Transport layer security
URI	Uniform resource identifier
XML	Extensible markup language