NCI AGENCY

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

AGENCY INSTRUCTION

INSTR TECH 06.02.12

SERVICE INTERFACE PROFILE FOR BASIC COLLABORATION 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	
NCI Agency	M. Laukner	michael.laukner@ncia.nato.int
NCI Agency	L. Schenkels	leon.schenkels@ncia.nato.int

Table of Contents

List of Annexes

AGENCY INSTRUCTION 06.02.12

SERVICE INTERFACE PROFILE FOR BASIC COLLABORATION 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 INTRODUCTION

At a fundamental level all collaboration is a human-based activity. The users can communicate and collaborate using a large variety of synchronous (e.g. instant messaging, white-boarding, voice and video conferences) and asynchronous (e.g. email, portals, shared workspaces) collaboration tools.

The initial version of the Collaboration Service Interface Profile (SIP) is focused on instant messaging and is based on the extensible messaging and presence protocol (XMPP). XMPP has been selected as the basis for the Collaboration Service, as it is a mature protocol with widespread adoption and many commercial and open-source implementations. It is also gaining growing acceptance in NATO and among the nations. For that reason it is important to standardize its implementations to maintain interoperability in NATO enterprise and federated scenarios.

This document recognizes that there may be additional collaboration services that are not included in this SIP, which will be addressed in future versions.

---

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

1.1 Purpose of this Document

In order to ensure compatibility between services, both within and without NATO, there is a need to ensure that a standard (and standards-based) profile can be defined which will be mandatory for all service operations in NATO Network Enabled Capability (NNEC).

The NATO Consultation, Command and Control Agency (NC3A) have developed a number of proposals for Service Interface Profiles (SIP) for the services defined in the Core Enterprise Services (CES) Framework [NC3A AC/322(SC/1) 0015 (INV), 2009].

The purpose of this document is to define the SIP that all applications should adhere to when providing or consuming instant messaging services, a component of the Collaboration Service. The Collaboration Service is one of the *Interaction* services defined in the CES Framework. Instant messaging supports instantaneous synchronous communication and includes the capability to discover people/contacts including their real-time presence information, ad hoc collaboration and participation in a number of concurrent sessions.

This document specifies the fundamental features to be implemented by the Collaboration Service, covering addressing, session establishment and communication primitives, whilst providing guidance, clarification and reference to the core XMPP standards (listed in Section 1.8.1). This document also defines the mandatory and recommended security mechanisms to be used by the Collaboration Service.

[NCIA TR/2013/SPW008423/37, 2014] extends on the fundamental features and security mechanisms, specified in this document, and specifies the service interfaces for the instant messaging services that are applicable for a *XMPP Client* and *XMPP Server*. [XSF XEP-0302, 2011] introduces a core and advanced concept, whereby core and advanced are categories for *XMPP Clients* and *XMPP Servers* indicating the listed REQUIRED specifications for compliance purposes for both categories. The Collaboration SIP, for the purpose of XMPP-based instant messaging, reuses this concept for categorizing XMPP *Services* as core and advanced instant messaging services, whereby core relates to a core set of service interfaces that are REQUIRED to be implemented by a compliant Collaboration Service, and advanced represents a suite of service interfaces that MAY be implemented by one or more XMPP entities for providing a compliant Collaboration Service. Appendix A provides a list and a brief explanation of the core and advanced instant messaging services.

In the future, additional forms of Collaboration Services, such as Voice conferencing and video conferencing, will be addressed in (revised) versions of this main Collaboration SIP document with additional Annexes to incorporate the SIPs for each of the additional forms of Collaboration Services.

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 Bi-Strategic Command (Bi-SC) or NNEC related systems.
- The architects and developers of service consumers and providers.
- Coalition partners whose services may need to interact with NNEC services.
- Systems 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, Terminology.
- `Courier font` indicates syntax derived from the different open standards, such as [IETF RFC 6120, 2011], [IETF RFC 6121, 2011] and [IETF RFC 6122, 2011].

1.4 Terminology

Table 1
Terminology

XMPP Server	Provides basic messaging, presence, and XML routing features.
XMPP Client	An application that enables you to connect to an *XMPP Server* for instant messaging with other users over a network.
Service	A feature or function that can be used by any XMPP application. Note: The term *Service* is used throughout this document to refer to XMPP enabling services.
Roster	A user's contact list that is stored by the user's *XMPP Server*.
XML Stream	A container for the exchange of XML elements between any two XMPP entities over a network.
Stream Feature	The set of XMPP protocol interactions that the initiating XMPP entity has to complete before the receiving XMPP entity will accept *XML Stanzas* from the initiating XMPP entity.
XML Stanza	A first-level XML element (at depth=1 of the stream) which typically contains one or more child XML elements (with accompanying attributes, elements, and XML character data) in order to convey the desired information. There are three kinds of *XML Stanzas: Message, Presence*, and *IQ* (short for "Info/Query").
Message	XML structure to support a "fire-and-forget" mechanism (basic "push" method) for getting information from one place to another.
Presence	Describes the status of an XMPP entity, and their availability for communication over a network.
IQ	Provides a structure for request-response interactions that is optimized for a more reliable exchange of data.
JabberID (JID)	A string, structured as an ordered sequence of localpart, domainpart, and resourcepart (where the first two parts are demarcated by the '@' character used as a separator, and the last two parts are similarly demarcated by the '/' character).

Bare JID	A *JabberID* that is of the form localpart@domainpart.
Full JID	A *JabberID* that is of the form localpart@domainpart/resourcepart.

1.5 Namespaces

The following namespaces are used in this document.

Table 2
Namespaces

Namespace	Reference
http://etherx.jabber.org/streams	[IETF RFC 6120, 2011]
http://jabber.org/features/compress	[XSF XEP-0138, 2009]
http://jabber.org/protocol/compress	[XSF XEP-0138, 2009]
jabber:client	[IETF RFC 6121, 2011]
jabber:server	[IETF RFC 6121, 2011]
urn:ietf:params:xml:ns:xmpp-bind	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-sasl	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-stanzas	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-streams	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-tls	[IETF RFC 6120, 2011]
urn:xmpp:bidi	[XSF XEP-0288, 2012]
urn:xmpp:features:bidi	[XSF XEP-0288, 2012]
urn:xmpp:ping	[XSF XEP-0199, 2009]
urn:xmpp:sm:3	[XSF XEP-0198, 2011]

1.6 Goals

The following are the goals of this profile:

- Identifying the instant messaging protocol for the Collaboration Service.
- Identifying the fundamental features that are REQUIRED to be implemented by any XMPP entity.
- Identifying the security mechanisms that are REQUIRED to be implemented by any XMPP entity.

- Identifying the service interfaces for core instant messaging services.
- Identifying service interfaces for advanced instant messaging services.

1.7 Non-Goals

The following topics are outside the scope of this profile:

- Recommendations for the use of products or platforms
- Network configuration and parameters required for instant messaging
- Configuration details of a particular server or client implementation
- Describing other collaboration services such as voice and video-based conferencing services.

1.8 Relationships to other Profiles and Specifications

Relationship with other CES SIP is:

- Enterprise Directory Service (see [NC3A RD-3153, 2011]) – Provides identity attributes used for authenticating XMPP entities to the Collaboration Services.

XMPP entities are addressable on the XMPP network. The XMPP-based Collaboration Service is reliant on the domain name system (DNS) services to provide the network-addressing structure for enabling XMPP entities to discover and communicate with each other over the XMPP network.

1.8.1 Normative references

The following documents have fed into this specification, and are incorporated as normative references:

1.8.1.1 XMPP core

- [IETF RFC 6120, 2011] Extensible Messaging and Presence Protocol (XMPP): Core
- [IETF RFC 6121, 2011] Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence
- [IETF RFC 6122, 2011] Extensible Messaging and Presence Protocol (XMPP): Address Format.

1.8.1.2 XMPP extensions

- [XSF XEP-0138, 2009] Stream Compression
- [XSF XEP-0198, 2011] Stream Management
- [XSF XEP-0199, 2009] XMPP Ping
- [XSF XEP-0220, 2011] Server Dial-back
- [XSF XEP-0288, 2012] Bidirectional Server-to-Server Connections.

1.8.1.3 Security

- [IETF RFC 4121, 2005] The Kerberos Version 5 Generic Security Service Application Programming Interface (GSS-API) Mechanism: Version 2
- [IETF RFC 4422, 2006] Simple Authentication and Security Layer (SASL)
- [IETF RFC 4505, 2006] Anonymous Simple Authentication and Security Layer (SASL) Mechanism
- [IETF RFC 4616, 2006] The PLAIN Simple Authentication and Security Layer (SASL) Mechanism
- [IETF RFC 4752, 2006] The Kerberos V5 ("GSSAPI") Simple Authentication and Security Layer (SASL) Mechanism
- [IETF RFC 5246, 2008] The Transport Layer Security (TLS) Protocol Version 1.2.

2 SIP DEFINITION

2.1 Subject

This SIP focuses on the fundamental features and security mechanisms of instant messaging, based on XMPP, as a part of the Collaboration Services, which is part of the *Interaction* services as defined in the Core Enterprise Services Framework [NC3A AC/322(SC/1) 0015 (INV), 2009].

It is impossible to completely guarantee the interoperability of a particular service. However, this SIP aims to increase the level of interoperability based on implementation experience to date.

2.1.1 Collaboration Service standards

Jabber/XMPP technologies were invented by Jeremie Miller in 1998 and the first implementation, jabbered, released in early 1999. This effort was very quickly seized on by a community of developers that collaboratively worked on defining and specifying the core protocols and extensions to the core protocols. This culminated in a non-profit membership organization, The Jabber Software Foundation, being formed in August 2001 (as of 2007 known as the XMPP Standards Foundation). Formalization of the core protocols within the IETF occurred in October 2004 with the publication of the core XMPP specifications ([IETF RFC 3920, 2004] and [IETF RFC 3921, 2004]). The XMPP standards have recently been revised, within the IETF, resulting in the most up-to-date specifications [IETF RFC 6120, 2011], [IETF RFC 6121, 2011] and [IETF RFC 6122, 2011].

2.2 Service Interface

The service interface is not a web service and therefore a web services description language (WSDL) is not appropriate to describe the operations of the service.

The service interface is defined by the IETF XMPP Standards specified in:

- Messaging and Presence Protocol (XMPP): Core , [IETF RFC 6120, 2011]
- Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence , [IETF RFC 6121, 2011]
- Extensible Messaging and Presence Protocol (XMPP): Address Format, [IETF RFC 6122, 2011].

2.3 Fundamental Features

XMPP provides a technology for the asynchronous, end-to-end exchange of text-based and structured data by means of direct, persistent *XML Streams* among a distributed network of globally addressable, presence-aware XMPP entities (*XMPP Clients* and *XMPP Servers)*.

2.3.1 Global addresses

2.3.1.1 JabberID (JID)

As XMPP communications happen on a network, every XMPP entity needs an address, called a *JabberID* (JID).

Each *JabberID* on the network is globally unique and SHOULD be based on the DNS.

A valid *JabberID* is a string of UNICODE ([Unicode Consortium Unicode Version 3.2.0, 2002]) code points, encoded using UTF-8 [IETF RFC 3629, 2003], and structured as an ordered sequence of localpart, domainpart, and resourcepart (where the localpart and domainpart are separated by the '@'character, and the domainpart and the resourcepart are separated by the '/' character).

For example, username@example.com/device

The formal specification of the XMPP address format for a *JabberID* depends on internationalization technologies. Due to the state of flux of these specifications (in relation to the internationalization

technologies), at the time of writing [IETF RFC 6120, 2011], the XMPP address format is defined in [IETF RFC 6122, 2011].

2.3.1.2 Domainpart

The domainpart is the primary identifier and is the only REQUIRED element of a *JabberID*. Typically a domainpart identifies the "local" server to which clients connect for XML routing and data management functionality.

The domain part for every XMPP *Service* SHALL be a fully qualified domain name (FQDN) (see DNS [IETF RFC 1035, 1987]), IPv4 address, IPv6 address, or unqualified hostname (i.e. a text label that is resolvable on a local network).

The domainpart SHALL be case-insensitive.

2.3.1.3 Localpart

The localpart of a *JabberID* is an OPTIONAL identifier placed before the domainpart and separated from it by the '@' character. Typically a localpart uniquely identifies the entity requesting and using network access provided by a server (i.e. a local account), although it can also represent other kinds of XMPP entities (e.g. a chat room associated with a multi-user chat service).

The localpart SHALL be case-insensitive.

2.3.1.4 Resourcepart

The resourcepart of a *JabberID* is an OPTIONAL identifier placed after the domainpart and separated from it by the '/' character. Typically a resourcepart uniquely identifies a specific connection (e.g. a device or location) or object (e.g. an occupant in a multi-user chat room) belonging to the entity associated with an XMPP localpart at a domain. The resourcepart is used for routing traffic to that connection instead of any other connections that may be open at that moment (associated with that XMPP localpart at a domain).

The resourcepart SHALL be case-sensitive, whereby user@example.com/device is different from user@example.com/Device.

2.3.2 Streaming XML

XMPP is a technology for streaming extensible markup language (XML). An initiating XMPP entity starts by connecting to a receiving XMPP entity and negotiates the session parameters. Once the stream has been negotiated, the initiating XMPP entity can send *Message*, *Presence* and *IQ* stanzas (see Section 2.3.2.6) to other XMPP entities in the network.

2.3.2.1 TCP bindings

The XMPP communication is realized by establishing a transmission control protocol (TCP)/Internet protocol (IP) connection between an initiating entity (*XMPP Client* or *XMPP Server*) and the receiving entity (*XMPP Server*).

An initiating XMPP entity SHOULD find a receiving XMPP entity using the DNS.

When using DNS, the initiating XMPP entity SHALL use the domainpart of the receiving (recipient) *JabberID* to resolve the FQDN of the receiving XMPP entity.

This process SHOULD use DNS service records (DNS-SRV) (see [IETF RFC 2782, 2000]) to obtain the IPv4 or IPv6 address (port and priority) of the receiving XMPP entity.

If the DNS-SRV resolution process is not possible, the process SHOULD be to use normal "A" or "AAAA" address record resolution to determine the IPv4 or IPv6 address of the origin domain of the

receiving entity (using the default ports). The default ports registered with the Internet Assigned Numbers Authority (IANA) are:

- "xmpp-client" port of 5222 for client-to-server connections
- "xmpp-server" port of 5269 for server-to-server connections.

In a particular installation these default ports MAY be changed although this is NOT RECOMMENDED.

2.3.2.2 Open XML Stream

Two fundamental concepts enable the rapid, asynchronous exchange of relatively small payloads of text-based and structured information between presence-aware XMPP entities:

1) *XML Streams*
2) *XML Stanzas (see Section 2.3.2.6).*

An *XML Stream* acts as an envelope for all the *XML Stanzas* sent during a session. The start of an *XML Stream* is denoted unambiguously by an opening "stream header" (i.e. an XML `<stream>` tag with appropriate attributes and namespace declarations).

The XML Namespace for the *XML Streams* SHALL be '`http://etherx.jabber.org/streams`'.

XMPP entities SHALL follow the process as described in Section 4.2 of [IETF RFC 6120, 2011] for opening an *XML Stream*.

The attributes of the *XML Stream* are specified in Section 4.7 of [IETF RFC 6120, 2011].

Table 3 details the relevant attribute value depending on the direction of the *XML Stream* to and from XMPP entities and the specifications that SHALL be followed that are relevant for the *XML Stream* attributes.

Table 3
XML Stream attributes

XML Stream attribute	Initiating XMPP entity to receiving XMPP entity	Receiving XMPP entity to initiating XMPP entity	Specification
`to`	*JabberID* of receiving XMPP entity	*JabberID* of initiating XMPP entity	[IETF RFC 6120, 2011], Section 4.7.2
`from`	*JabberID* of initiating XMPP entity	*JabberID* of receiving XMPP entity	[IETF RFC 6120, 2011], Section 4.7.1
`id`	NOT REQUIRED	Unique *XML Stream* Identifier	[IETF RFC 6120, 2011], Section 4.7.3
`xml:lang`	Default language	Default language	[IETF RFC 6120, 2011], Section 4.7.4
`version`	1.0	1.0	[IETF RFC 6120, 2011], Section 4.7.5

XMPP entities handling invalid XML Namespaces use for the *XML Stream* SHALL follow the specifications described in Section 4.8.1 of [IETF RFC 6120, 2011].

2.3.2.3 Stream negotiation

The receiving XMPP entity imposes certain conditions, on the initiating XMPP entity when connecting as a *XMPP Client* or as a *XMPP Server*.

The receiving XMPP entity informs the initiating XMPP entity about such conditions by communicating the *Stream Features*, whereby all *Stream Features* SHOULD indicate whether they are REQUIRED or OPTIONAL.

At a minimum, the initiating XMPP entity is REQUIRED to authenticate (see Section 2.4.2 for the authentication conformance statements) with the receiving XMPP entity. However, the receiving XMPP entity can consider other conditions to be mandatory-to-negotiate.

Stream negotiation rules for mandatory-to-negotiate SHALL be followed as described in Sections 5.3.1, 6.3.1 and 7.3.1 of [IETF RFC 6120, 2011].

TLS ([IETF RFC 5246, 2008]) SHALL be supported for protecting stream negotiation information and *XML Stanzas* by encrypting *XML Streams* (see Section 2.4.1 for TLS conformance statements), and SHOULD support compression of the encrypted traffic as specified in [IETF RFC 3749, 2004].

In the cases where TLS cannot be implemented, and as such TLS compression cannot be implemented, the XMPP entities SHOULD support the specifications described in [XSF XEP-0138, 2009] for compressing the *XML Streams*.

An XMPP entity SHALL follow the specifications described in Sections 4.3.3 and 4.3.4 of [IETF RFC 6120, 2011] for *XML Stream* restarts.

An XMPP entity SHALL follow the specifications described in Section 4.3.5 of [IETF RFC 6120, 2011] when completing the *XML Stream* negotiation.

2.3.2.4 Resource-binding

Resource-binding, whereby a resource identifier is applied for a particular connection to denote separate points of presence is only applicable for *XMPP Client* to *XMPP Server* communication.

After the *XMPP Client* has successfully authenticated with a *XMPP Server*, it SHALL bind a specific resource to that *XML Stream*.

The XML Namespace for the Resource Binding SHALL be '`urn:ietf:params:xml:ns:xmpp-bind`'.

There SHALL be an XMPP resource associated with a *Bare JID* of the *XMPP Client*, so that the address, for use over that *XML Stream* (to send and receive *XML Stanzas*) is a *Full JID*.

A compliant XMPP entity SHALL follow the specifications described in Section 7 of [IETF RFC 6120, 2011] for binding of a resource to the *XML Stream*.

2.3.2.5 Address determination

On completion of stream negotiation (see Section 2.3.2.3), SASL negotiation (see Section 2.4.2) and resource binding (see Section 2.3.2.4), the specifications described in Section 4.3.6 of [IETF RFC 6120, 2011] SHALL be followed for determining the XMPP entity *JabberIDs*.

2.3.2.6 XML Stanzas

Once the *XML Stream* has been successfully negotiated, XMPP entities are permitted to exchange *XML Stanzas*.

An *XML Stanza* exists at the direct child level of the root `<stream/>` element and is a concept that represents the basic unit of communication in XMPP.

An *XML Stanza* MAY contain child elements (with accompanying attributes, elements, and XML character data) as necessary in order to convey the desired information. The only *XML Stanzas* defined in the XMPP protocol are the `<message/>`, `<presence/>`, and `<iq/>` elements all of which are qualified by the default namespace for the *XML Stream*.

The default namespace, for the *XML Stream* instantiated between a *XMPP Client* and *XMPP Server*, SHALL be '`jabber:client`'.

The default namespace, for the *XML Stream* instantiated between *XMPP Servers* SHALL be '`jabber:server`'.

The attributes of an *XML Stanza* are specified in Section 8.1 of [IETF RFC 6120, 2011]. Table 4 (for reference purposes only) details the attributes and their relevant specifications.

Table 4
XML *Stanza* attributes

***XML Stanza* attribute**	**Description**	**Specification**
`to`	The recipient *JabberID*	[IETF RFC6120, 2011], Section 8.1.1
`from`	The sending *JabberID*	[IETF RFC6120, 2011], Section 8.1.2
`id`	A unique identifier for the *XML Stanza*	[IETF RFC6120, 2011], Section 8.1.3
`type`	The purpose or context of the *XML Stanza*	[IETF RFC6120, 2011], Section 8.1.4
`xml:lang`	The default language	[IETF RFC6120, 2011], Section 8.1.5

An *XML Stanza, and its semantics,* is determined by several factors:

- The XML Stanza element name (Message, Presence or IQ)
- The value of the '`type`' attribute
- The child elements(s).

2.3.2.7 Close XML Stream

The XMPP entities parties maintain the TCP/IP connection for as long as the *XML Streams* are in use.

The end of the *XML Stream* SHALL be denoted unambiguously by a closing XML `</stream>` tag.

2.3.2.8 Directionality

An *XML Stream* is always unidirectional, whereby *XML Stanzas* can only be sent in one direction over the *XML Stream*, either:

- From the initiating XMPP entity to the receiving XMPP entity

- From the receiving XMPP entity to the initiating XMPP entity.

XMPP entities primarily use two TCP connections (one for each *XML Stream*).

XMPP Servers MAY negotiate the use of a single TCP connection for bidirectional *XML Stanza* exchange.

XMPP Servers that use server-to-server connections for bidirectional *XML Stanza* exchange, such that *XML Stanzas* are sent and received on the same TCP connection, MUST be compliant with the protocol specification in [XSF XEP-0288, 2012].

2.3.2.9 Management of XML Streams

XML Stream handling can be further enhanced by providing active management that allows for:

- Reliable delivery
- Disconnection detection
- Session resumption.

XMPP entities MAY support active management of *XML Streams*.

XMPP entities that support active management of *XML Streams* SHALL be compliant with the protocol specifications in [XSF XEP-0198, 2011] and [XSF XEP-0199, 2009].

2.3.2.10 Error-handling

Section 4.9 of [IETF RFC 6120, 2011] specifies the *XML Stream* errors. When errors occur at the *XML Stream* level they are unrecoverable.

The XML Namespace for the *XML Streams* errors SHALL be '`urn:ietf:params:xml:ns:xmpp-streams`'.

An XMPP entity that detects an error SHALL send an <error/> XML element with the appropriate child XML element specifying the error condition and then close the *XML Stream* as specified in Section 2.3.2.7.

For reference purposes, Table 5 lists the error conditions that SHALL be supported (see Section 4.9.3 of [IETF RFC 6120, 2011] for a description of these errors and Section 4.9.1 of [RFC 6120, 2011] for the rules).

The error conditions related to *XMPP Server* generated Resource Identifiers, when Resource Binding, as specified in Section 7.6.2 of [IETF RFC 6120, 2011] SHALL be supported. The error conditions are listed in Table 6 (for reference purposes only).

The error conditions related to *XMPP Client* submitted Resource Identifiers, when Resource Binding, as specified in Section 7.7.2 of [IETF RFC 6120, 2011] SHALL be supported. The error conditions are listed in Table 7 (for reference purposes only).

2.3.3 Communication primitives

2.3.3.1 Message

The XMPP `<message/>` stanza is the basic "push" method for getting information from one place to another. *Message* Stanzas are differentiated by the '`type`' attribute. The types of *Message* Stanza are listed in Table 8.

[NCIA TR/2013/SPW008423/37, 2014] SHALL further qualify recommendations for handling *Message* Stanzas, based on the instant messaging service that is specifying the use of *Message* Stanza type(s).

The *Message* Stanza MAY contain child elements. [IETF RFC6121, 2011] specifies three OPTIONAL *Message* stanza child elements. These are:

- `Body` – Contains human-readable XML character data specifying the textual contents of the message.
- `Subject` – Contains human-readable XML character data specifying the topic of the message.
- `Thread` – Used to uniquely identify conversations or messaging sessions between two XMPP entities instantiated by *Message* stanzas.

Presence

In XMPP, *Presence* advertises the network availability of other XMPP entities. *Presence* Stanzas are differentiated by their '`type`' attribute. The types of *Presence* Stanzas are listed in Table 9.

Table 5
List of *XML Stream* error conditions

Error condition
bad-format
bad-namespace-prefix
conflict
connection-timeout
host-gone
host-unknown
improper-addressing
internal-server-error
invalid-from
invalid-namespace
invalid-xml
not-authorized
not-well-formed
policy-violation
remote-connection-failed
reset

resource-constraint
restricted-xml
see-other-host
system-shutdown
undefined-condition
unsupported-encoding
unsupported-feature
unsupported-stanza-type
unsupported-version

Table 6
Resource binding error conditions for *XMPP Server* generated resource identifiers

Error condition
resource-constrained
not-allowed

Table 7
Resource binding error conditions for *XMPP Client* submitted resource identifiers

Error condition
bad-request
conflict

Table 8
Types of *Message* Stanza

***Message* Stanza type**	**Description**
normal	This is the default value. The message is a standalone message for which it is expected that the recipient will reply.

chat	The message is sent in the context of a one-to-one chat session.
groupchat	The message is sent in the context of a multi-user chat environment.
headline	The message provides an alert, notification, or other transient information to which no reply is expected.
error	The message is generated by an XMPP entity that encounters an error when processing a message received from another XMPP entity.

Table 9
Types of *Presence* Stanza

***Presence* Stanza type**	**Description**
subscribe	The sending XMPP entity wishes to subscribe to the receiving XMPP entity's presence.
subscribed	The sending XMPP entity has allowed the receiving XMPP entity to receive its presence.
unavailable	The sending XMPP entity is no longer available for communication.
unsubscribe	The sending XMPP entity is unsubscribing from the receiving XMPP entity's presence.
unsubscribed	The subscription request has been denied or cancelled.
probe	A request for an XMPP entity's current presence.
error	An error has occurred.

An XMPP entity's availability is signaled when it generates a <presence/> stanza with no 'type' attribute. There is no default 'type' value for a *Presence* Stanza.

[NCIA TR/2013/SPW008423/37, 2014] SHALL further qualify recommendations for handling *Presence* Stanzas, based on the instant messaging service that is specifying the use of *Presence* Stanza type(s).

The *Presence* stanza MAY contain child elements. [IETF RFC 6121, 2011] specifies three OPTIONAL *Presence* stanza child elements. These are:

- Show – Specifies the particular availability sub-state of an XMPP entity.
- Status – Contains the human-readable XML character data specifying a natural-language description of an XMPP entity's availability.

- `Priority` – Contains an integer value ranging between -127 to +127 specifying the priority level of the resource.

2.3.3.2 Info/Query (IQ)

The *IQ* Stanza provides a structure for request-response interactions. Unlike the *Message* Stanza, an *IQ* Stanza SHALL include only one payload, which defines the request to be processed or action to be taken by the receiving XMPP entity.

In addition, the requesting XMPP entity SHALL always receive a reply from the responding XMPP entity. Each *IQ* Stanza is differentiated by its 'type' attribute. The types of IQ Stanzas are listed in Table 10.

Table 10
Types of IQ Stanza

IQ Stanza type	Description
get	The requesting XMPP entity requests information.
set	The requesting XMPP entity provides information or makes a request.
result	The responding XMPP entity returns the result of a 'get' or acknowledges a 'set' request.
error	The responding XMPP entity notifies the requesting XMPP entity that it was unable to process the 'get' or 'set' request.

The data content of the request and response SHALL be defined by the namespace declaration of a direct child element of the IQ element, and the interaction SHALL be tracked by the requesting entity through use of the 'id' attribute. Using the values of the *IQ* Stanza's 'type' attribute a structured *IQ* interaction between XMPP entities can be generated, shown in Figure 1.

The specification described in Section 8.2.3 of [IETF RFC 6120, 2011] for *IQ* interactions between XMPP entities SHALL be followed.

2.3.3.3 Extensibility

While the *Message*, *Presence*, and *IQ* stanzas provide basic semantics for messaging, availability, and request-response interactions, XMPP uses XML namespaces to extend the basic *XML Stanza* syntax for the purpose of providing additional functionality.

Thus a *Message* or *Presence* stanza MAY contain one or more optional child elements specifying content that extends the meaning of the message and an *IQ* stanza of type "get" or "set" SHALL contain one such child element.

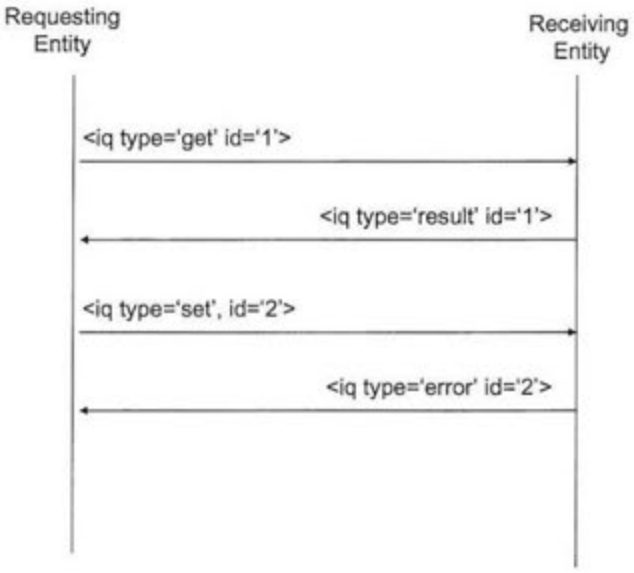

Figure 1 Structured IQ interactions between XMPP entities

A child element MAY have any name and SHALL possess a namespace declaration (other than "jabber:client", "jabber:server", or "http://etherx.jabber.org/streams") that defines all data contained within the child element. Such a child element is said to be EXTENDED CONTENT and its namespace name is said to be an EXTENDED NAMESPACE.

Support for any given extended namespace is OPTIONAL.

In the case where extended namespaces are specified for an advanced instant messaging service, specified in [NCIA TR/2013/SPW008423/37, 2014] all XMPP entities that are conformant with that instant messaging service interface SHALL support that extended namespace.

If an XMPP entity does not understand an extended namespace, the entity's expected behaviour depends on whether the XMPP entity is:

- The *XMPP Client* (see Section 2.3.3.4.1)
- An *XMPP Server* that is routing the *XML Stanza* to the *XMPP Client* (see Section 2.3.3.4.2).

2.3.3.3.1 XMPP Client

If an *XMPP Client* receives a *XML Stanza* that contains a child element it does not understand, it SHALL silently ignore that particular XML data, i.e. it SHALL NOT process it or present it to a user or associated application (if any). In particular:

- If an *XMPP Client* receives a *Message* or *Presence* stanza that contains XML data qualified by a namespace it does not understand, the portion of the stanza that qualified by the unknown namespace SHALL be ignored.
- If an *XMPP Client* receives a message stanza whose only child element is qualified by a namespace it does not understand, it SHALL ignore the entire stanza.
- If an *XMPP Client* receives an *IQ* stanza of type "get" or "set" containing a child element qualified by a namespace it does not understand, the entity SHALL return an IQ stanza of type "error" with an error condition of <service-unavailable/>.

2.3.3.3.2 XMPP Server

If a *XMPP Server* handles a stanza that contains a child element it does not understand, it SHALL ignore the associated XML data by routing or delivering it untouched to the *XMPP Client*.

2.3.3.4 Error-handling

Section 8.3 of [IETF RFC 6120, 2011] specifies the *XML Stanza* errors. When errors occur at the *XML Stanza* level they are recoverable.

The XML Namespace for the *XML Stanza* errors SHALL be '`urn:ietf:params:xml:ns:xmpp-stanzas`'.

An XMPP entity that detects an error SHALL send an *XML Stanza* (of the same kind as the generated *XML Stanza* that triggered the error) with the appropriate child XML element, identified by a '`error-type`' and specifying the error condition.

For reference purposes Table 11 lists the '`error-types`' that SHALL be supported (see Section 8.3.2 of [IETF RFC 6120, 2011]]).

Table 11
Types of *XML Stanza* errors

***XML Stanza* error type**	**Description**
auth	Retry after providing credentials
cancel	Do not retry
continue	Proceed
modify	Retry after changing the data set
wait	Retry after waiting

For reference purposes Table 12 lists the error conditions that SHALL be supported (see Section 8.9.3 of [IETF RFC 6120, 2011] for a description of these errors).

An XMPP entity SHALL support the handling of *XML Stanza* errors as specified in Section 8.9.3 of [IETF RFC 6120, 2011] and SHALL apply the rules specified in Section 8.3.1 of [IETF RFC 6120, 2011].

Table 12
List of *XML Stanza* error conditions

Error conditions
bad-request
conflict

feature-not-implemented
forbidden
gone
internal-server-error
item-not-found
jid-malformed
not-acceptable
not-allowed
not-authorized
policy-violation
recipient-unavailable
redirect
registration-required
remote-server-not-found
remote-server-timeout
resource-constraint
service-unavailable
subscription-required
undefined-condition
unexpected-request

2.4 Security

2.4.1 Connection

TLS ([IETF RFC 5246, 2008]) SHALL be supported to protect any information being transferred between XMPP entities, specifically the "STARTTLS" extension to encrypt the communication between XMPP entities, dependent on the authentication mechanism negotiated (see Section 2.4.2).

The XML Namespace for the STARTTLS extension SHALL be '`urn:ietf:params:xml:ns:xmpp-tls`'.

XMPP entities that are negotiating STARTTLS SHALL follow the specification as described in Section 5 of [IETF RFC 6120, 2011].

On successful TLS negotiation the receiving XMPP entity SHALL send stream features to the initiating XMPP entity which SHOULD include the SASL stream feature qualified by the XML namespace '`urn:ietf:params:xml:ns:xmpp-sasl`'.

The cipher suites and hashing algorithms to be supported by TLS and SASL SHALL be compliant with the NATO public key infrastructure (PKI) cipher suites and algorithms.

2.4.1.1 Error-handling

If the XMPP entities fail to negotiate TLS, the receiving XMPP entity SHALL terminate the TCP connection and SHALL close the *XML Stream* as defined in Section 2.3.2.7.

2.4.2 Authentication

A compliant XMPP entity is REQUIRED to support SASL ([IETF RFC 4422, 2006]) as the method for authentication.

The XML Namespace for the SASL extension SHALL be '`urn:ietf:params:xml:ns:xmpp-sasl`'.

Support for SASL negotiation is REQUIRED for *XMPP Client* to *XMPP Server* authentication.

XMPP Server to *XMPP Server* authentication MAY support SASL negotiation.

If a receiving *XMPP Server* cannot offer SASL negotiation, it SHALL attempt to authenticate using the Server Dialback protocol [XSF XEP-0220, 2011] (see Section 2.4.3). XMPP entities that are negotiating SASL SHALL follow the specification as described in Section 6 of [IETF RFC 6120, 2011].

The following SASL mechanisms SHALL be supported:

- ANONYMOUS [IETF RFC 4505, 2006]
- PLAIN [IETF RFC 4616, 2006]
- GSSAPI [IETF RFC 4121, 2005] with KERBEROS_V5 [IETF RFC 4752, 2006]
- EXTERNAL [IETF RFC 4422, 2006].

If PLAIN is used then the TLS "STARTTLS" extension SHALL be used to protect the credentials. If EXTERNAL is used then the use of TLS "STARTTLS" is REQUIRED.

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

2.4.2.1 Error-handling

Failure of SASL authentication is defined in Section 6.4.5 of [IETF RFC 6120, 2011]. The list of error conditions that SHALL be supported are listed in Table 13 (for reference purposes) and specified in [IETF RFC 6120, 2011].

Table 13
List of SASL error conditions

Error conditions
aborted
account-disabled
credentials-expired
encryption-required
incorrect-encoding
invalid-authzid
invalid-mechanism
malformed-request
mechanism-too-weak
not-authorized
temporary-auth-failure

2.4.3 Federation

The XMPP Architecture is a federated architecture which allows for XMPP entities to host *Services* that are accessible across the XMPP network, whereby providing cross-domain collaboration. XMPP entities discover XMPP *Services* distributed in the XMPP network using DNS services (specified in Section 2.3.2.1).

Inter-domain federation of XMPP entities is dynamic and it is RECOMMENDED that *XMPP Servers* federate and authenticate using TLS and SASL EXTERNAL, whereby the X.509 v3.0 digital certificates for each *XMPP Server* are exchanged during the TLS negotiation and verified to provide mutual authentication.

As specified in Section 2.4.2, *XMPP Servers* that cannot support TLS and SASL EXTERNAL SHALL failback to Server Dialback.

Server Dialback is a weak authentication protocol that is specified in [XSF XEP-0220, 2011]. Server Dialback provides identity verification, through DNS and the use of keys based on a shared secret, for an *XML Stream*.

XMPP servers that implement Server Dialback SHALL perform Server Dialback for both *XML Streams* that are established between *XMPP Servers*, whereby providing bi-directional communication. Figure 2 illustrates the process for a unidirectional *XML Stream*.

An XMPP entity, when implementing Server Dialback, SHALL be conformant with the specifications detailed in Section 2 of [XSF XEP-0220, 2011].

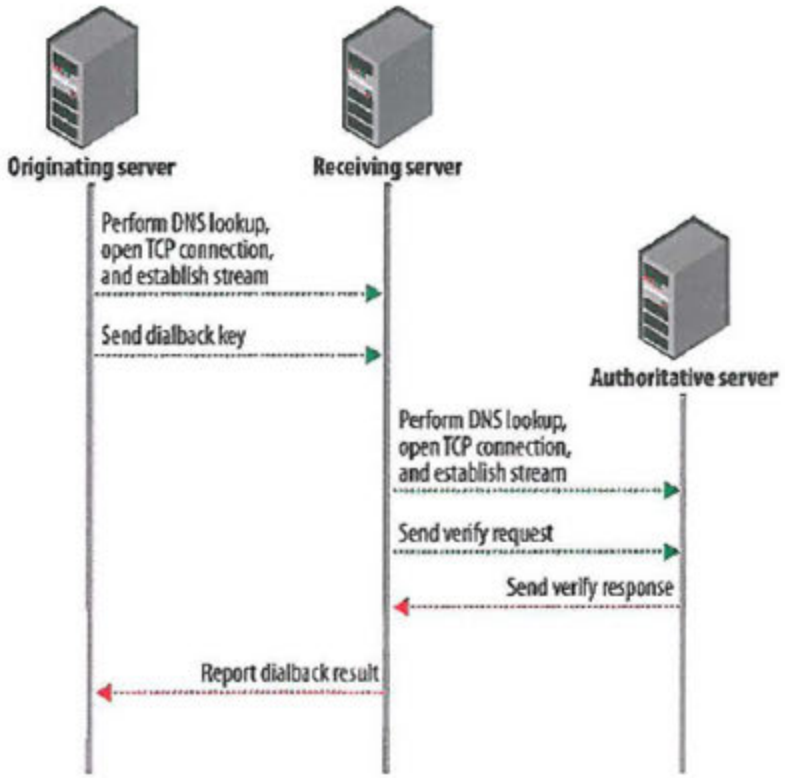

Figure 2 Server dial-back process for providing unidirectional *XMPP Server* identification

3 REFERENCES

[IETF RFC 1035, 1987]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 1035, "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION", P. Mockapetris, at http://tools.ietf.org/html/rfc1035, November 1987, viewed 5 March 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 5 March 2012.

[IETF RFC 2634, 1999]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 2634, "Enhanced Security Services for S/MIME", P. Hoffman, Editor, at http://www.ietf.org/rfc/rfc2634.txt, June 1999, viewed 5 March 2012.

[IETF RFC 2782, 2000]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 2782, "A DNS RR for specifying the location of services (DNS SRV)", A. Gulbrandsen et al., at http://tools.ietf.org/html/rfc2782, February 2000, viewed 5 March 2012.

[IETF RFC 3629, 2003]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3629, "UTF-8, a transformation format of ISO 10646", F. Yergeau, at http://www.ietf.org/rfc/rfc3629, November 2003, viewed 5 March 2012.

[IETF RFC 3749, 2004]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3749, "Transport Layer Security Protocol Compression Methods", T. Dierks, at http://www.ietf.org/rfc/rfc3749, May 2004, viewed 5 March 2012.

[IETF RFC 3920, 2004]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3920, "Extensible Messaging and Presence Protocol (XMPP): Core", P. Saint-Andre, at http://www.ietf.org/rfc/rfc3920, October 2004, viewed 5 March 2012.

[IETF RFC 3921, 2004]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3921, "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence, P. Saint-Andre, at http://www.ietf.org/rfc/rfc3921, October 2004, viewed 5 March 2012.

[IETF RFC 4121, 2005]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4121, "The Kerberos Version 5 Generic Security Service Application Programming Interface (GSS-API) Mechanism: Version 2", L. Zhu et al., at http://www.ietf.org/rfc/rfc4121, July 2005, viewed 5 March 2012.

[IETF RFC 4422, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4422, "Simple Authentication and Security Layer (SASL)", A. Melnikov et al., at http://www.ietf.org/rfc/rfc4422, June 2006, viewed 5 March 2012.

[IETF RFC 4505, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4505, "Anonymous Simple Authentication and Security Layer (SASL) Mechanism", K. Zeilenga, at http://www.ietf.org/rfc/rfc4505, June 2006, viewed 5 March 2012.

[IETF RFC 4616, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4616, "The PLAIN Simple Authentication and Security Layer (SASL) Mechanism", K. Zeilenga, at http://www.ietf.org/rfc/rfc4616, June 2006, viewed 5 March 2012.

[IETF RFC 4752, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4752, "The Kerberos V5 ("GSSAPI") Simple Authentication and Security Layer (SASL) Mechanism", A. Melinokv, at http://www.ietf.org/rfc/rfc4752, November 2006, viewed 5 March 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, at http://www.ietf.org/rfc/rfc5246, August 2008, viewed 5 March 2012.

[IETF RFC 6120, 2011]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 6120, "Extensible Messaging and Presence Protocol (XMPP): Core", P. Saint-Andre, at http://www.ietf.org/rfc/rfc6120, March 2011, viewed 5 March 2012.

[IETF RFC 6121, 2011]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 6121, "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", P. Saint-Andre, at http://www.ietf.org/rfc/rfc6121, March 2011, viewed 5 March 2012.

[IETF RFC 6122, 2011]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 6122, "Extensible Messaging and Presence Protocol (XMPP): Address Format", P. Saint-Andre, at http://www.ietf.org/rfc/rfc6122, March 2011, viewed 5 March 2012.

[NAC AC/322(SC/1) 0015 (INV), 2009]:
North Atlantic Council document AC/322(SC/1) 0015 (INV), "NATO Core Enterprise Services Framework v1.2", April 2009.

[NC3A RD-3153, 2011]:
NATO Consultation, Command and Control Agency Reference Document 3153,"Enterprise Directory Services Service Interface Profile Proposal" (*Provisional Title*), NC3A Core Enterprise Services Team, NC3A, The Hague, Netherlands, unpublished document dated September 2011 (NATO Unclassified).

[NCIA TR/2013/SPW008423/37, 2014]:
NATO Communications and Information Agency Technical Report 2013/SPW008423/36, "Core and Advanced Instant Messaging Collaboration Service Interface Profile Proposal", A. Ross, M. Laukner, L. Schenkels, NCI Agency, The Hague, Netherlands, January 2014 (NATO Unclassified).

[Unicode Consortium Unicode Version 3.2.0, 2002]:
The Unicode Consortium (on-line) http://www.unicode.org Unicode Version 3.2.0, "Components of the Unicode Standard Version 3.2.0", at http://www.unicode.org/versions/components-3.2.0.html, March 2002, viewed 5 March 2012.

[XSF XEP-0138, 2009]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0138: Stream Compression", XMPP Extensions, Joe Hildebrand, Peter Saint-Andre, at http://xmpp.org/extensions/xep-0138.html, 27 May 2009, viewed 5 March 2012.

[XSF XEP-0198, 2011]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0198: Stream Management", XMPP Extensions, Justin Karneges et al., at http://xmpp.org/extensions/xep-0198.html, 29 June 2011, viewed 12 July 2013.

[XSF XEP-0199, 2009]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0199: XMPP Ping", XMPP Extensions, Peter Saint-Andre, at http://xmpp.org/extensions/xep-0199.html, 03 June 2009, viewed 12 July 2013.

[XSF XEP-0220, 2011]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0220: Server Dialback", XMPP Extensions, Jeremie Miller et al., at http://xmpp.org/extensions/xep-0220.html, 19 September 2011, viewed 5 March 2012.

[XSF XEP-0288, 2012]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0288: Bidirectional Server-to-Server Connections", XMPP Extensions, Philipp Hancke et al., at http://xmpp.org/extensions/xep-0288.html, 21 August 2012, viewed 10 July 2013.

[XSF XEP-0302, 2011]:
XMPP Standards Foundation (on-line), http://xmpp.org, "XEP-0302: Compliance Suites 2012", XMPP Extensions, Peter Saint-Andre, at http://xmpp.org/extensions/xep-0302.html, 21 July 2011, viewed 5 March 2012.

4 ABBREVIATIONS

API	Application programming interface
Bi-SC	Bi-Strategic Command
CES	Core Enterprise Service
DNS	Domain name system
FQDN	Fully qualified domain name
GSS	Generic security service
IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
IP	Internet protocol
NC3A	NATO Consultation, Command and Control Agency
NISP	NATO Interoperability Standards and Profile
NNEC	NATO Network Enabled Capability
PKI	Public key infrastructure
RFC	Request for comments
SASL	Simple authentication and security layer
SIP	Service Interface Profile
TCP	Transmission control protocol
TLS	Transport Layer Security
WSDL	Web services description language
XML	Extensible markup language
XMPP	Extensible messaging and presence protocol

ANNEX 1 – CORE INSTANT MESSAGING SERVICES

This annex provides a brief summary of the capabilities offered by the core and advanced instant messaging services provided by the Collaboration Service. Detailed service interfaces for these *Services* are provided in [NCIA TR/2013/SPW008423/37, 2014].

1.1 Core Instant Messaging Services

1.1.1 Presence service

This *Service* advertises the network availability of other extensible messaging and presence protocol (XMPP) entities hence providing the knowledge of whether those XMPP entities are online and available for communication. The core Presence Service manages a subscription model, in effect a simple publish-subscribe method, whereby XMPP entities that have subscribed to an XMPP entity's presence receive updated presence information when that XMPP entity comes online and goes offline.

1.1.2 Roster service

This *Service* provides the capability for an XMPP entity to maintain a list of known and trusted XMPP entities for communicating with. This list is stored on the *XMPP Server* that manages the account for that XMPP entity. This *Service* also provides the capability to request and store additional information pertaining to XMPP entities on the XMPP network.

1.1.3 One-to-one messaging service

This *Service* provides the capability for any two XMPP entities on a network to exchange XML *Messages*.

1.2 Advanced Instant Messaging Services

1.2.1 XMPP service discovery service

This *Service* enables an XMPP entity to discover and search which XMPP entities are available on the network and what features or capabilities (*Services*) an available XMPP entity offers.

1.2.2 Multi-party messaging service

This *Service* provides the capability for an XMPP entity to join a chat room, as a participant of that chat room, and exchange messages with multiple participants.

1.2.3 Notification service

This *Service* provides a publish-subscribe capability optimized for one-to-many delivery. Specific topics (known as nodes) are created and published by an XMPP entity, whereby information published to a node triggers a notification broadcasted to all XMPP entities that have explicitly subscribed.

1.2.4 Structured data form service

This *Service* provides an XMPP operator with the ability to select, fill in and submit structured data forms. The *Service* offers a dynamic approach to enabling an XMPP community with the ability to discover, subscribe and be notified of up-to-date structured data forms within the XMPP network. A submitted structured data form is then distributed to subscribed XMPP entities for processing of the form data.

This *Service* enables a structured and flexible forms based messaging exchange between XMPP entities.

1.2.5 Whiteboarding service

This *Service* provides a virtual whiteboard for shares images or files and lets multiple participants work and annotate on these images or files concurrently, with real-time updates being shared between all participants.

1.2.6 Time-sensitive messaging service

This *Service* provides the capability to ensure delivery of a *Message* Stanza before an absolute point in time.

1.2.7 Labelling service

This *Service* provides a lightweight approach to Security Labelling of XMPP stanzas by offering three techniques:

3) Extensible support for different formats of security labels, i.e. NATO XML Label ([IETF RFC 2634, 1999]) ESS Security Label.
4) A user-friendly display marking, representative of the encoded security label (in whatever format), transmitted alongside each other whereby a foreground and background colour for the display marking is provided. This means that an *XMPP Client* does not need to understand the format of the security label when displaying the security label.
5) The *XMPP Client* can obtain a catalogue of security labels (subset of available security labels within a security policy) to apply to an XMPP stanza for any intended receiving XMPP entity.