Abstract

The Session Peering Provisioning Protocol (SPPP) is an XML protocol that exists to enable the provisioning of session establishment data into Session Data Registries or SIP Service Provider data stores. Sending XML data structures over Simple Object Access Protocol (SOAP) and HTTP(s) is a widely used, de-facto standard for messaging between elements of provisioning systems. Therefore the combination of SOAP and HTTP(s) as a transport for SPPP is a natural fit. The obvious benefits include leveraging existing industry expertise, leveraging existing standards, and a higher probability that existing provisioning systems can be more easily integrated with this protocol. This document describes the specification for transporting SPPP XML structures over SOAP and HTTP(s).

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on August 30, 2010.

Copyright Notice
Copyright (c) 2010 IETF Trust and the persons identified as the
document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the BSD License.

Table of Contents

1. Introduction ................................................. 3
2. Terminology .................................................. 4
3. SOAP Features and SPPP ................................. 5
4. HTTP(s) Features and SPPP ................................. 7
5. Authentication and Session Management ................. 8
6. SPPP SOAP WSDL Definition ............................... 9
7. SPPP SOAP Message Examples ............................... 11
8. Security Considerations ..................................... 12
9. IANA Considerations ......................................... 13
10. Acknowledgements ............................................ 14
11. References .................................................. 15
   11.1. Normative References ................................. 15
   11.2. Informational References ............................. 15
Author’s Address ................................................. 16
1. Introduction

SPPP, defined in [I-D. draft-mule-drinks-proto-01], is best supported by a transport and messaging infrastructure that is connection oriented, request-response oriented, easily secured, supports propagation through firewalls in a standard fashion, and that is easily integrated into back-office systems. This is the type of environment that inter-organization provisioning transactions typically take place. Given the current state of industry practice and technologies, SOAP and HTTP(s) are ideal for this type of environment. This document describes the specification for transporting SPPP XML structures over SOAP and HTTP(s).

The specification in this document for transporting SPPP XML structures over SOAP and HTTP(s) is primarily comprised of five subjects: (1) a description of any applicable SOAP features, (2) any applicable HTTP features, (3) authentication and session management, (4) security considerations, and perhaps most importantly, (5) the Web Services Description Language (WSDL) definition.
2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].
3. SOAP Features and SPPP

The list of SOAP features that are explicitly used and required for SPPP are limited. Most SOAP features are not necessary for SPPP. SPPP primarily uses SOAP simply as a standard message envelope technology. The SOAP message envelope is comprised of the SOAP header and body. As described in the SOAP specifications, the SOAP header can contain optional, application specific, information about the message. The SOAP body contains the SPPP message itself, whose structure is defined by the combination of one of the WSDL operations defined in this document and the SPPP XML data structures defined in the SPPP protocol document. SPPP does not rely on any data elements in the SOAP header. All relevant data elements are defined in the SPPP XML schema described in [I-D. draft-mule-drinks-proto-01] and the SPPP WSDL specification described in this document.

WSDL is a widely standardized and adopted technology for defining the top-level structures of the messages that are transported within the body of a SOAP message. The WSDL definition for the SPPP SOAP messages is defined later in this document, which imports by reference the XML data types contained in the SPPP schema. The IANA registry where the SPPP schema resides is described in The IETF XML Registry [RFC3688].

There are multiple structural styles that SOAP WSDL allows. But the best practice for this type of application is what is often referred to as the Document Literal Wrapped style of designing SOAP WSDL. This style is generally regarded as an optimal approach that enhances maintainability, comprehension, portability, and, to a certain extent, performance. Figure 4 illustrates this high level technical structure.
The SOAP operations supported by SPPP are normatively defined later in this document. Each SOAP operation defines a request/input message and a response/output message. Each such request and response message then contains a single object that wraps the SPPP XML data types that comprise the inputs and the outputs, respectively, of the SOAP operation.

SOAP faults are not used by the SPPP SOAP mapping. All SPPP success and error responses are specified within the SPPP protocol specification [I-D.draft-mule-drinks,proto-01].

SOAP 1.1 [SOAP] or higher and WSDL1.1 [WSDL] or higher SHOULD be used.
4. HTTP(s) Features and SPPP

SOAP is not tied to HTTP(s), however, for reasons described in the introduction, HTTP(s) is a good choice as the transport mechanism for the SPPP SOAP messages. HTTP 1.1 includes the "persistent connection" feature, which allows multiple HTTP request/response pairs to be transported across a single HTTP connection. This is an important performance optimization feature, particularly when the connections is an HTTPS connection where the relatively time consuming SSL handshake has occurred. Persistent connections SHOULD be used for the SPPP HTTP connections.

HTTP 1.1 [HTTP] or higher SHOULD be used.
5. Authentication and Session Management

All SOAP and HTTP SPPP Clients and Servers MUST support Transport Layer Security (TLS) as defined in [RFC5246] as the secure transport mechanism. All SOAP ESPP Clients and Servers MUST use HTTP Digest Authentication as defined in [RFC2617] as the secure authentication mechanism. As a result, the communication session is established as a result of the initial HTTP connection setup, the digest authentication, handshake, and the TLS handshake. When the HTTP connection is broken down, the communication session ends.
6. SPPP SOAP WSDL Definition

The SPPP WSDL is defined below. The SPPP WSDL is what is commonly referred to as _Generic WSDL_. It is generic in the sense that there is not a specific WSDL operation defined for each business action that is supported by the SPPP protocol. There is a single WSDL operation called submitRequest. It takes as input an spppRequestMsg object and returns as output an spppResponseMsg object. These objects _wrap_ the spppRequest and spppResponse objects respectively. These two object data structures are described in the SPPP protocol specification [REF NUM]. And finally, the spppSOAPBinding in the WSDL defines the binding style as _document_ and the encoding as _literal_. It is this combination of _wrapped_ input and output data structures, _document_ binding style, and _literal_ encoding that characterized the Document Literal Wrapped style of WSDL specificaitons.

The advantage of generic WSDL is that the WSDL is more succinct, much simpler, and therfore more easily maintained. As operations are added into or removed from the SPPP protocol, the WSDL does not need to change. This approach is made possible by the fact that the SPP XML data types and supported actions are defined in the SPPP XML schema, not in the WSDL. As a result the supported actions do not need to be re-defined here inside the SPPP SOAP WSDL.

TBD Add information about versioning and namespaces for the WSDL.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl=http://schemas.xmlsoap.org/wsdl/
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:sppps="urn:ietf:params:xml:ns:sppp:soap:1"
 targetNamespace="urn:ietf:params:xml:ns:sppp:soap:1"
 xsi:schemaLocation="spppbase.xsd">
 <wsdl:types>
 <xsd:schema targetNamespace="urn:ietf:params:xml:ns:sppp:soap:1"
 xmlns:sppps="urn:ietf:params:xml:ns:sppp:soap:1"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 </xsd:schema>
</wsdl:types>
```
<wsdl:message name="spppRequestMsg">
  <wsdl:part name="rqst" element="spppb:spppRequest"/>
</wsdl:message>

<wsdl:message name="spppResponseMsg">
  <wsdl:part name="rspns" element="spppb:spppResponse"/>
</wsdl:message>

<wsdl:portType name="spppPortType">
  <wsdl:operation name="submitRequest">
    <wsdl:input message="spppRequestMsg"/>
    <wsdl:output message="spppResponseMsg"/>
  </wsdl:operation>
</wsdl:portType>

<wsdl:binding name="spppSoapBinding" type="spppPortType">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="submitRequest">
    <soap:operation soapAction="submitRequest" style="document"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>

<wsdl:service name="spppService">
  <wsdl:port name="spppPort" binding="spppSoapBinding">
    <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
  </wsdl:port>
</wsdl:service>

</wsdl:definitions>

Figure 2: WSDL
7. SPPP SOAP Message Examples

TBD
8. Security Considerations

TBD
9. IANA Considerations

This document uses URNs to describe XML namespaces and XML schemas conforming to a registry mechanism described in [RFC3688].

URI assignments are requested: TBD
10. Acknowledgements

TBD
11. References

11.1. Normative References

[I-D.draft-mule-drinks,proto-01]
Mule, J-F., "Session Peering Provisioning Protocol",
draft-ietf-mule-drinksproto-01 (work in progress),
March 2009.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S.,
Leach, P., Luotonen, A., and L. Stewart, "HTTP
Authentication: Basic and Digest Access Authentication",

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,


11.2. Informational References

[I-D.drinks-usecases-requirements-00]
Channabasappa, S., "DRINKS Use cases and Protocol
Requirements", draft-ietf-drinks-usecases-requirements-00
(work in progress), March 2009.

Interconnect (SPEERMINT) Terminology", RFC 5486,
March 2009.
Author’s Address

Kenneth Cartwright
TNS
1939 Roland Clarke Place
Reston, VA  20191
USA

Email: kcartwright@tnsi.com