iPass® Generic Interface Specification

BETWEEN SMART CLIENTS AND ACCESS GATEWAY, VERSION 1.6

Corporate Headquarters
iPass Inc.
3800 Bridge Parkway
Redwood Shores, CA 94065 USA

www.ipass.com
+1 650-232-4100
+1 650-232-0227 fx

TABLE OF CONTENTS

TABLE OF CONTENTS

Scope

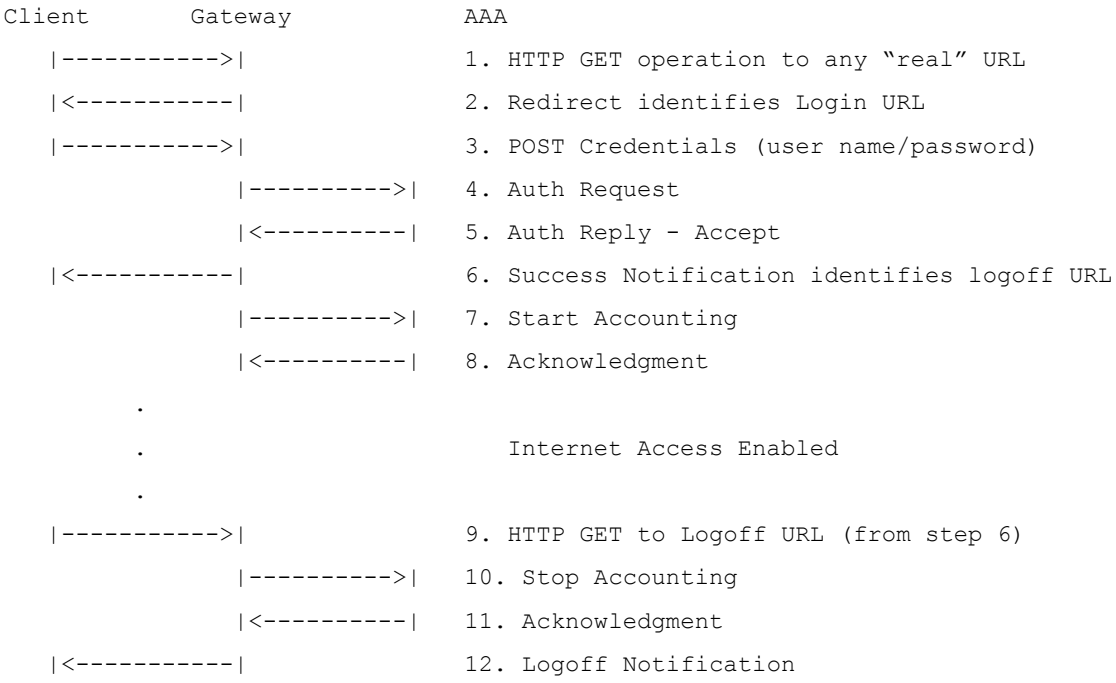
This document contains a Generic Interface Specification (GIS) between a smart client and an access gateway or other hybrid access control system. The GIS is compatible with the existing Web Browser authentication services that are presently deployed at hotspots.

Client Integration

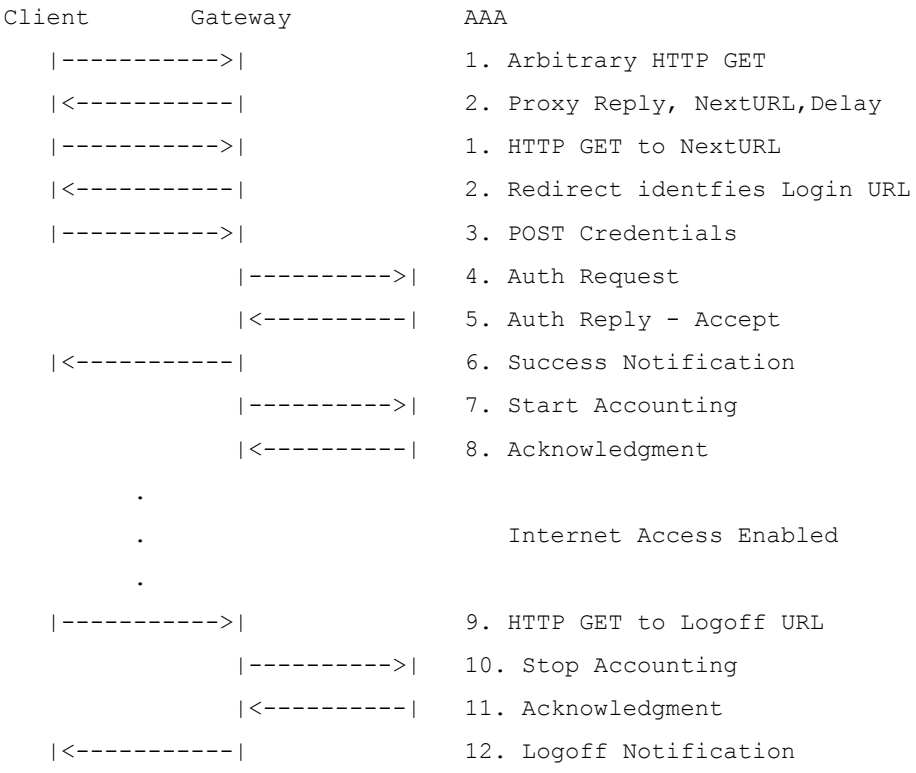
This interface is implemented through the use of client-initiated, HTTP message exchanges. TCP connections are established to ports 80 and 443 unless otherwise indicated. HTTP version 1.0 (http://www.ietf.org/rfc/rfc1945.txt) is specified due to its simplified header formats.

The following interaction diagrams represent the access protocol from the perspective of the smart client.

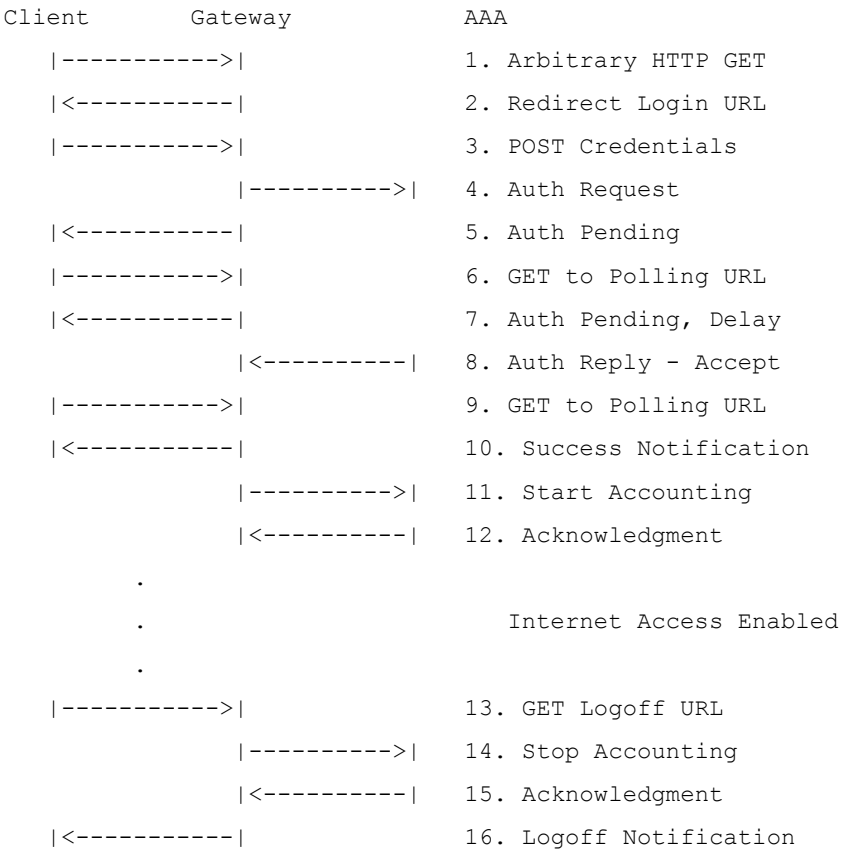
Login Request: Successful Case



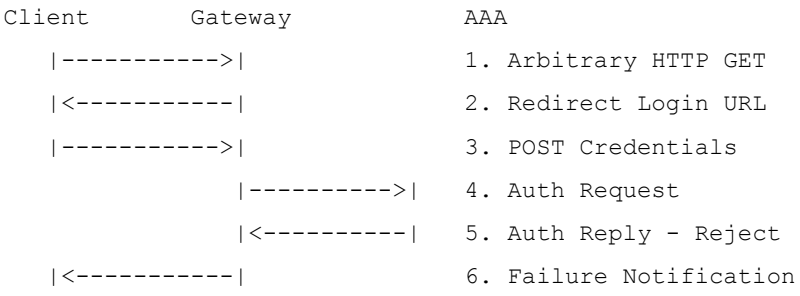
Login Request: Successful Case With Proxy Reply



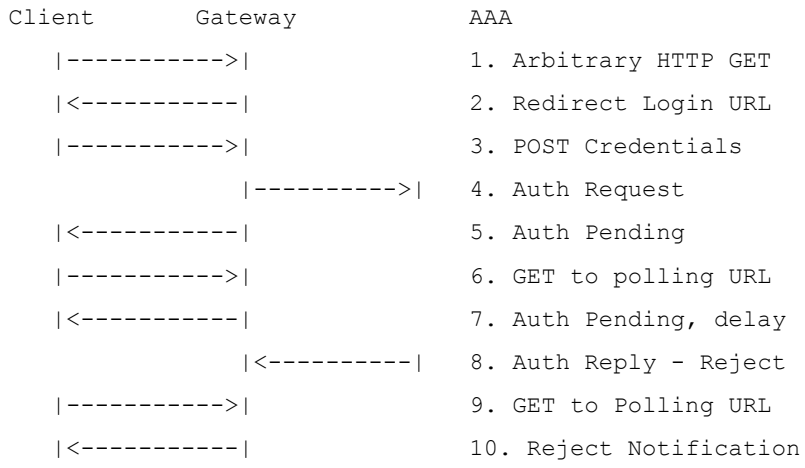
Login Request: Successful Case With Polling



Login Request: Reject



Login Request: Reject With Polling



Protocol Specifics

The smart client to access gateway (GIS) protocol is implemented using protocol messages consisting of well-formed XML documents. Presently, no assumption of standardized URLs is made. Rather, the protocol depends on using an HTTP-hijack/redirect. Most access gateways already provide a redirect mechanism for users attempting to access the network via a web browser. The protocol is designed to provide features that permit the GIS protocol messages to exist within gateway HTML pages which are also served to ordinary web browsers.

To avoid confusing web-browser clients, the gateway should “encapsulate” all GIS messages within an HTML comment tag within each HTML message to prevent interpretation by a web browser. To assure maximum “web browser invisibility” the “encapsulated” XML message should appear outside all inner HTML message segments (e.g., HEAD, BODY, etc) of the HTML message.

The access gateway must ensure HTML compatibility with a wide range of browsers. For this reason, both the HTTP Content-Length and Content-Type (“text/html; charset=UTF-8”) headers should be included in all HTML pages. It is further recommended that gateways include and enforce the Connection header with the “close” keyword. This will help protect the gateway from denial-of-service conditions resulting from connections incidentally left open due to a defective client or malicious user.

Access gateways are widely used to control access to wireless services using IEEE 802.11. Due to the weaknesses in present implementations of the 802.11 air-security protocol (WEP), the GIS protocol requires that SSL be used to protect the subscriber’s authentication credentials. In order to further protect the subscriber from rogue access points, the gateway must utilize a security certificate from a reputable Certificate Authority that can be readily verified by the smart client. Authentication of the access point is outside the scope of this document.

The protocol messages include a proxy notification message as some existing access gateways require it. All messages from the access gateway to the smart client will contain both response codes and message types.

The message types shall be one of the following values:

Message Type	Message Meaning
100	Initial redirect message
110	Proxy notification
120	Authentication notification
130	Logoff notification
140	Response to Authentication Poll
150	Response to Abort Login

The response code shall be one of the following values:

Response Code	Response Meaning
0	No error
50	Login succeeded (Access ACCEPT)
100	Login failed (Access REJECT)
102	Authentication server error/timeout
105	Network Administrator Error: No authentication server enabled
150	Logoff succeeded
151	Login aborted
200	Proxy detection/repeat operation
201	Authentication pending
255	Access gateway internal error

In the following sections, GIS messages and their sub-elements are referred to by their XML root tag name enclosed in angled brackets as it would appear in the opening XML tag:

```
<GIS message name> or <GIS message element name>
```

Message content which is inserted by the gateway is indicated by enclosing its description in braces:

```
{Gateway-generated content}
```

Note: Braces are not a part of the message syntax.

Smart Client HTTP GET to ORIGIN SERVER

The smart client shall perform an HTTP GET to a valid web site to initiate the access sequence.

In situations where the client device is already authorized, the access gateway shall pass the HTTP GET through to the connected public network and return no special response.

If the subscriber should explicitly navigate to the login page within the gateway's walled garden while already authorized for access via the smart client, the access gateway shall respond with a web page indicating that the user is already logged in or other appropriate notification in response to an authorization attempt.

When the client device is not currently authorized for access, the access gateway shall return one of the following in reply to the initial HTTP GET operation:

- An HTTP redirect (301/302/303/307/308) status with an accompanying HTTP Location header and no GIS

content (See section 0).

- an HTTP OK (200) status with an accompanying HTML <META HTTP-EQUIV="Refresh" Content="…"> tag and no GIS content (See section 0).
- An HTTP redirect (301/302/303/307/308) status and HTML page including an XML GIS <Proxy> message (see section 0).
- an HTTP redirect (301/302/303/307/308) status and HTML page including an XML GIS <Redirect> message (see section 0)
- an HTTP OK (200) status and HTML page including an XML GIS <Proxy> message (see section 0)
- an HTTP OK (200) status and HTML page including an XML GIS <Redirect> message (see section 0)

This will be covered in more detail below.

HTTP Redirect with no GIS Content

When an HTTP REDIRECT (301/302/303/307/308) status is returned, the response MAY NOT contain an HTML body. IF present, the body of the page MAY contain an XML document containing either the <Redirect> message or the <Proxy> message defined in the following sections.

When the HTML message is NOT PRESENT –or- does not contain an XML document containing a GIS message, the GIS smart client shall perform another HTTP GET operation to the URL present in the HTTP Location header accompanying the HTTP 301/302/303/307/308 status. The page retrieved by the GIS client after implementing the HTTP redirect MUST contain an XML document containing the GIS <Redirect> elements as defined in the table below or a <Proxy> message as defined in the next two sections.

META "Refresh" with no GIS Content

When an HTTP OK (200) status is returned, the body of the page MAY contain an XML document containing either the <Redirect> message or the <Proxy> message defined in the following sections.

When the HTML message does not contain an XML document containing a GIS message, the GIS smart client shall perform another HTTP GET operation to the URL present in the "Content" attribute of the META tag accompanying the HTTP 200 status.

> Note: The "refresh" string in the META tag must be present as exactly one of the following two strings:
> • Refresh
> • refresh
> No other representations with alternative capitalization shall be used to assure compatibility with older HTML parsers.

The page retrieved by the GIS client MUST contain an XML document containing the GIS <Redirect> message or a <Proxy> message as defined in the next two sections.

Redirect

When the returned HTTP message contains an HTML message containing the <Redirect> message, any HTTP redirect or META "Refresh" function indicated by such HTTP message shall be ignored. Instead the GIS smart client shall process the <Redirect> message elements described in the following table.

The <Redirect> information shall be contained within a valid HTML message, delimited appropriately with the <HTML> and </HTML> tags. The HTML message may contain other valid HTML message elements (e.g., HEAD, BODY, etc.).

Redirect Message Elements

Information name	Field format/value	Required/ Optional
Access procedure	<AccessProcedure> 1.0 </AccessProcedure>	Required
Location Identifier	<AccessLocation> {Location ID} </AccessLocation>	Required
Location Name	<LocationName> {User readable location name} </LocationName>	Required
Login URL	<LoginURL> https://{site specific login URL} </LoginURL>	Required- Must be a secure URL
Abort Login URL	<AbortLoginURL> http[s]://{abort login URL} </AbortLoginURL>	Optional* (0 or 1) see note below
Message Type	<MessageType> 100 </MessageType>	Required
Response Code	<ResponseCode> {Response Code} </ResponseCode>	Required

Note: The AbortLoginURL element is only required if the gateway is implementing the **Authentication Results Polling** option (see section 0). The AbortLoginURL may specify either a secure (https:) or unsecure (http:) site**. If the gateway does not support this functionality, this tag should NOT be present in the <Redirect> message.**

The *Access Procedure* must be exactly the string: 1.0 (numeral one, period, numeral zero) for gateways whose features comply with just this version of the GIS specification. Future versions of the specification will assure upward client compatibility for clients which ignore message elements which are not a part of this specification version. **For this reason, a schema-validating XML parser SHOULD NOT be used by GIS clients**. All GIS message versions shall be well-formed.

The *Location ID* specified uniquely identifies the device or subnet through which the access will occur. If this ID is a characteristic of the physical device, replacement of the device may modify the ID received from the access location.

The *location name* can be presented by the smart client to the user to identify the access location.

When all required parameters are not present, an internal malfunction of the access gateway shall be assumed, and the smart client will behave as though it received a response code 255: access gateway internal error.

The *AbortLoginURL* is used by the smart client to inform the access gateway that some error has occurred during the login process. When this is received by the access gateway, every attempt should be made to abort the session cleanly without generating an accounting record. This message element is required only when the gateway implements the authentication results polling procedures defined in a later section.

`{response code}` shall be one of the values listed in the following table:

Response Code	Response Message
0	No error
105	Network Administrator Error: No authentication server enabled
255	Access gateway internal error

Proxy

When the returned HTTP message contains an HTML message containing the <Proxy> message, any HTTP redirect or META Refresh function indicated by such message shall be ignored. Instead the GIS smart client shall process the <Proxy> message elements described in the following table.

The GIS <Proxy> message MAY contain an optional <Delay> element. The proxy message SHALL only be returned in response to the initial HTTP GET at login. The information SHOULD be contained within a valid HTML message, delimited appropriately with the <HTML> and </HTML> tags. The HTML message may contain other valid HTML message elements (e.g., HEAD, BODY, etc.).

Proxy message Elements

Information name	Field format/value	Required/ Optional
Message Type	<MessageType> 110 </MessageType>	Required
Response Code	<ResponseCode> {Response Code} </ResponseCode>	Required
Next URL	<NextURL> http[s]://{<site specific URL>} </NextURL>	Optional
Delay in seconds	<Delay> {Number of seconds data} </Delay>	Optional

When all *required* parameters are not present, the smart client will assume an internal malfunction of the access gateway, and the smart client shall behave as though an access gateway internal error response code was received.

If the <Proxy> message includes a <Delay> element, the smart client SHALL suspend execution for the number of seconds before it resends the HTTP GET to the gateway.

> **Note:** The <Proxy> message may be returned by the gateway multiple times (but only once in response to each HTTP GET operation). Some reasonable limit to the number of recurrences SHOULD be implemented by the GIS smart client to avoid an undesirable user experience in the event of gateway malfunction. A 6-cycle limit is recommended.

Gateways SHOULD specify a <Delay> interval large enough to reflect actual processing delay to prevent large numbers of GET recurrences and potential authentication failure due to GIS client limits described in the preceding paragraph. Gateways SHOULD specify a non-zero <Delay> value whenever there will be an actual processing delay on the gateway to avoid being overwhelmed by rapidly repeating HTTP GET operations by smart clients.

If the optional <NextURL> element is present, the GIS smart client SHALL perform another GET operation to the specified URL. This effectively implements a URL "redirect" specific to the GIS smart client. This use of the <Proxy> message is available to redirect GIS clients to a different gateway pageset without changing the normal, browser-oriented gateway page sequence.

The URL must be different in each <Proxy> message. If no <NextURL> element is present in a given proxy message, the last-used URL value WILL be used by the smart client.

`{response code}` shall be one of the values listed in the following table:

Response Code	Response Message
200	Proxy detection/repeat operation
255	Access gateway internal error

Authentication

The authentication phase of the protocol shall be started by an authentication request POST operation by the smart client. This will be followed by an Authentication Reply from the gateway. This phase may include an *optional* "authentication results polling" behavior by the gateway as defined in this section.

Authentication Request

The smart client shall perform a secure HTTP POST operation to the login URL returned in the redirect message. Since the post will be using HTTPS, it should be assumed that port 443 would be used unless otherwise specified as part of the LoginURL.

The POST parameters shall be as follows:

- **UserName:** the full user id (NAI) including appropriate clearinghouse routing prefixes
- **Password:** the user's password
- **Button:** form button identifier
- **OriginatingServer:** the URL of the server to which the activation GET operation was directed

Field name	Field naming/format specification	Required/ Optional
User name input field	name="UserName" max size="253"	Required
Password input	name="Password" max size="128"	Required
Button Identifier	name="button" content="Login"	Required
Form Name	Name="FNAME" content="0" (numeral zero)	Required
Origin Server	Name="OriginatingServer" content={original server GET URL}	Required

Authentication Reply

The access gateway shall return an <AuthenticationReply> message in response to the Authentication Request operation performed by the smart client.

When the gateway response contains the <AuthenticationReply> message it shall further contain the elements described in the table below. The information SHOULD be contained within a valid HTML message, delimited appropriately with the <HTML> and </HTML> tags. The HTML message may contain other valid HTML message elements (e.g., HEAD, BODY, etc.).

AuthenticationReply Message Elements

Information name	Field format/value	Required/ Optional
Message Type	<MessageType> 120 </MessageType>	Required
Response Code	<ResponseCode> {Response Code} </ResponseCode>	Required
Reply Message	<ReplyMessage> {Reply Message Text} </ReplyMessage>	Optional *(0 or more)
Login results URL	<LoginResultsURL> http[s]://{site specific login URL} </LoginResultsURL>	Optional**
Logoff URL	<LogoffURL> http[s]://{site specific logoff URL} </LogoffURL>	Optional***
Redirection URL	<RedirectionURL> http[s]://{redirection URL} </RedirectionURL>	Optional

* The <ReplyMessage> element MUST be returned when a RADIUS authentication system is used by the gateway and a returned RADIUS Access-Accept or Access-Reject message contains RADIUS attribute 18, *Reply-Message*. When multiple RADIUS *Reply-Message* attribute instances are present in the RADIUS message, each instance should be returned in a separate <Reply Message> element. The order of the individual <Reply Message> instances MUST reflect the order of the RADIUS *Reply-Message* attribute instances in the RADIUS message.

The <LoginResultsURL> element must be present in the authentication reply if the response code is “Authentication Pending” (code 201). This element should NOT be present under any other circumstances. It may contain session specific information if required by the access gateway. The Login Results URL may specify either a secure (https:) or unsecure (http:) site. **No component of either the user name or password shall be passed as URL parameters EXCEPT when such communication is directed to a secure URL. Client programs shall filter URL parameters to prevent insecure communication of user credentials by misconfigured gateways.

***The <LogoffURL> element must be present in the authentication reply if the response code is “Login succeeded”. It may contain session specific information if required by the access gateway. The Logoff URL may specify either a secure (https:) or unsecure (http:) site. **No component of either the user name or password shall be passed as URL parameters EXCEPT when such communication is directed to a secure URL. Client programs shall filter URL parameters to prevent insecure communication of user credentials by misconfigured gateways.**

`{response code}` shall be one of the values listed in the following table:

Response Code	Response Meaning
50	Login succeeded (Access ACCEPT)
100	Login failed (Access REJECT)
102	Authentication server error/timeout
201	Authentication pending
255	Access gateway internal error

As described above, the <ReplyMessage> element(s) return text to the smart client that is taken from the RADIUS Reply-Message attribute, when RADIUS authentication is used. This allows the AAA server to provide a human readable reason for rejecting an authentication request or other administrative guidance (e.g., "Your password will expire tomorrow"). Multiple Reply-Messages may be included in the RADIUS message and they must be returned to the smart client in the same order as they appear in the RADIUS message.

The access gateway may choose to block the smart client's execution during execution of the authentication request and then return the result to the user if the time-to-authenticate is expected to be low. This is the typical execution behavior of most access gateways.

Alternately, if there are many concurrent authentication requests and/or the time-to-authenticate is very high or the access gateway has limited memory resources, the access gateway may choose, instead, to immediately return an <AuthenticationPending> message causing the smart client to poll for the result of its authentication request. This allows the access gateway to immediately reuse memory and operating system resources that would otherwise be held idle while waiting for completion of the authentication operation.

If the reply to the authentication POST operation includes the *Authentication Pending* response code (201), the smart client shall begin polling the access gateway for the authentication results. This requires the inclusion of the optional <LoginResultsURL> element in the <AuthenticationReply> message from the gateway. The login results URL may specify either a secure (https:) or unsecure (http:) site.

The <RedirectionURL> can be used by the smart client as the start page for a launched browser session following authentication completion.

Authentication Results Polling

If the <AuthenticationReply> message contains the *Authentication Pending* response code, the smart client SHALL begin the authentication results polling procedure. The polling procedure shall consist of a series of one or more HTTP GET operations by the smart client to a secure URL, each followed by an HTTP 200 or HTTP 301/302/303/307/308 status message from the access gateway.

> **Note**: There is no requirement that a gateway implement this polling functionality. All GIS-compliant smart clients, however, must implement support for the authentication results polling procedure.

Authentication Poll

The smart client shall send a HTTP GET to the <LoginResultsURL> that was returned in the <AuthenticationReply> message. When the polling GET operation accesses a secure (HTTPS) URL, it is assumed that port 443 will be used unless otherwise specified as part of the URL. Whenever possible, the <LoginResultsURL> SHOULD specify a secure URL since the <AuthenticationPollReply> returned by the gateway may contain <ReplyMessage> text that is confidential to the user. The use of an insecure URL also exposes the content of the <LogoffURL> creating an opportunity for

nuisance denial-of-service attacks on the user's session in some gateway architectures. The <LoginResultsURL> may, of course, include URL parameters. **However, such parameters MUST NOT include any element of the user credentials (username or password) UNLESS the URL is secure. Client software developers should verify compliance with this requirement to prevent accidental or intentional exposure of the user's credentials.**

Response to Authentication Poll

The access gateway shall return one of the following in reply to the authentication results poll.

- an HTTP redirect (301/302/303/307/308) status with an accompanying HTTP Location header
- an HTTP OK (200) status with an accompanying HTML <META HTTP-EQUIV="Refresh" Content="…"> tag.

The reply shall contain an XML document containing the <AuthenticationPollReply> elements as described in the table below. The information SHOULD be contained within a valid HTML message, delimited appropriately with the <HTML> and </HTML> tags. The HTML message may contain other valid HTML message elements (e.g., HEAD, BODY, etc.).

AuthenticationPollReply Message Elements

Information name	Field format/value	Required/ Optional
Message Type	<MessageType> 140 </MessageType>	Required
Response Code	<ResponseCode> {Response Code} </ResponseCode>	Required
Reply Message	<ReplyMessage> {Reply Message Text} </ReplyMessage>	Optional*
Delay in seconds	<Delay> {Number of seconds data} </Delay>	Optional
Logoff URL	<LogoffURL> http[s]://{site specific logoff URL} </LogoffURL>	Optional**
Redirection URL	<RedirectionURL> http[s]://{redirection URL} </RedirectionURL>	Optional

* The <ReplyMessage> element MUST be returned when a RADIUS authentication system is used by the gateway and a returned RADIUS Access-Accept or Access-Reject message contains RADIUS attribute 18, *Reply-Message*. When multiple RADIUS *Reply-Message* attribute instances are present in the RADIUS message, each instance should be returned in a separate <Reply Message> element. The order of the individual <Reply Message> instances MUST reflect the order of the RADIUS *Reply-Message* attribute instances in the RADIUS message.

**The <LogoffURL> element must be present in the authentication poll reply if the response code is "Login succeeded". It may contain session specific information if required by the access gateway. The Logoff URL may specify either a secure (https:) or unsecure (http:) site.

`{response code}` shall be one of the values listed in the following table:

Response Code	Response Meaning
50	Login succeeded (Access ACCEPT)
100	Login failed (Access REJECT)
102	Authentication server error/timeout
201	Authentication pending
255	Access gateway internal error

If the authentication is complete, the response to the authentication poll will contain the authentication result. If not (when response code 201 is returned), the smart client will delay for the number of seconds specified in the <Delay> element before resending the HTTP GET to the <LoginResultsURL> specified in the <AuthenticationReply> message from the gateway.

The optional <ReplyMessage> returns text to the smart client that is taken from the RADIUS Reply-Message attribute, if RADIUS authentication is used. This allows the AAA server to provide a human readable reason for rejecting an authentication request. Multiple Reply-Messages may be included in a RADIUS Access-Reply message and they must be returned to smart client in the same order as they appear in the RADIUS Access-Reply message.

The <RedirectionURL> can be used by the smart client as the start page for a launched browser session following authentication completion.

Abort Login

When the gateway implements authentication results polling and a protocol error occurs during the polling process, the smart client SHALL perform a GET operation to the <AbortLoginURL> specified in the initial <Redirect> message from the gateway. The access gateway should respond with an HTTP 200 or HTTP 302.

Abort Login Request

To abort a login, the smart client shall send a HTTP GET operation to the <AbortLoginURL> returned in the initial <Redirect> message.

Abort Login Reply

The access gateway shall return one of the following responses in reply to the Abort Login Request operation performed by the smart client:

- HTTP 200 and HTML page including the <AbortLoginReply> message
- HTTP 200 with <META HTTP-EQUIV=”Refresh” …> tag and HTML page including the <AbortLoginReply> message
- HTTP 302 redirect and HTML page including the <AbortLoginReply> message

The reply shall contain an XML document containing the <AbortLoginReply> elements described in the table below. The smart client shall NOT act on the redirection information when an HTTP 302 status is returned or the Content attribute when a META Refresh tag is present. The Abort Login Reply information SHOULD be contained within a valid HTML message, delimited appropriately with the <HTML> and </HTML> tags. The HTML message may contain other valid HTML message elements (e.g., HEAD, BODY, etc.).

When the Abort Login request is received by the gateway AFTER the corresponding authentication operation has completed, the gateway shall respond in one of the following ways:

- If the Login operation failed, the gateway shall return an < AbortLoginReply > message specifying the Login Aborted response code (151).
- If the Login operation succeeded, the gateway shall return an < AbortLoginReply > message specifying the Login succeeded response code (50) and includes the <LogoffURL> element.

AbortLoginReply Message Elements

Information name	Field format/value	Required/ Optional
Message Type	<MessageType> 150 </MessageType>	Required
Response Code	<ResponseCode> {Response Code} </ResponseCode>	Required
Logoff URL	<LogoffURL> http[s]://{site specific logoff URL} </LogoffURL>	Optional*

* The <LogoffURL> must be present in the < AbortLoginReply > message if the response code is "Login Succeeded" (50). It may contain session-specific information if required by the access gateway. If the login operation has already succeeded, the user's access session has not been terminated. The smart client may terminate the session by sending a logoff request to the <LogoffURL>. The Logoff URL may specify either a secure (https:) or unsecure (http:) site.

`{response code}` shall be one of the values listed in the following table:

Response Code	Response Meaning
50	Login succeeded (Access ACCEPT)
151	Login aborted
255	Access gateway internal error

Logoff

The logoff phase of the protocol is triggered by a GET operation to the <LogoffURL> by the smart client. This operation is followed by a HTTP 200 or HTTP 302 response by the access gateway.

Logoff Request

To initiate a logoff, the smart client SHALL perform a HTTP GET operation to the <LogoffURL> returned in either the <AuthenticationReply> or <AuthenticationPollReply> message.

Logoff Reply

The access gateway shall return one the following in response to the logoff request operation by the smart client:

- HTTP 200 status and HTML page with <LogoffReply> message
- HTTP 302 status and HTML page with <LogoffReply> message

The reply shall contain an XML document with the <LogoffReply> message elements described in the table below. The client shall not act on the redirection information when an HTTP 302 status is returned. The Logoff Reply information SHOULD be contained within a valid HTML message, delimited appropriately with the <HTML> and </HTML> tags. The HTML message may contain other valid HTML message elements (e.g., HEAD, BODY, etc.).

LogoffReply Message Elements

Information name	Field format/value	Required/ Optional
Message Type	<MessageType> 130 </MessageType>	Required
Response Code	<ResponseCode> {Response Code} </ResponseCode>	Required

`{response code}` shall be one of the values listed in the following table:

Response Code	Response Meaning
150	Logoff succeeded*
255	Access gateway internal error

*Gateways shall return the “Logoff succeeded” response code when the logoff request refers to a session which has been previously terminated due to a prior logoff request or gateway session timeout. When the gateway is unable to determine the prior state of a session, the gateway shall return the “Logoff succeeded” response code.

Appendix A: ACCESS GATEWAY INTERFACE SPECIFICATION LICENSE AGREEMENT

ACCESS GATEWAY INTERFACE SPECIFICATION LICENSE AGREEMENT

This License Agreement (the "Agreement") is a legal agreement between you, either an individual or a single legal entity ("You"), and iPass Inc. ("iPass") that governs Your acquisition and use of the version of the iPass specification identified above and any accompanying documentation (the "Specification"). You must accept the terms of this Agreement before viewing, downloading, copying, or otherwise using (collectively, "Acquiring") the Specification.

By clicking "ACCEPT" at the end of this Agreement, or by Acquiring the Specification, You are indicating that You have read and understood, and assent to be bound by, the terms of this Agreement. If you are an individual working for a company, you represent and warrant that you have all necessary authority to bind your company to the terms and conditions of this Agreement.

If You do not agree to the terms of the Agreement, You are not granted any rights whatsoever in the Specification.

The Specification is owned by iPass or its licensors and protected by copyright and other intellectual property rights. **The Specification is licensed, not sold or given away, strictly under the terms of this Agreement.**

1. **IPASS LICENSE GRANT.** Subject to the terms of this Agreement, iPass grants to You a royalty-free, worldwide, non-exclusive, non-transferable license, without the right to sublicense, under the Specification Intellectual Property, (a) to make, have made, use, sell, offer to sell, and import Implementations of the Specification and (b) to make a reasonable number of verbatim copies of the Specification solely for Your internal use in connection with exercising the license granted in clause (a). The "Specification Intellectual Property" is composed of iPass copyrights in the Specification and the claims of patents (if any) owned by iPass that are necessarily infringed by making, having made, using, selling or importing an Implementation, but specifically excludes any trademark, service mark, or trade name of iPass or its affiliates. An "Implementation" is any implementation of the Specification developed by You that (y) is a complete and fully-compliant implementation of the mandatory requirements set forth in the Specification, without sub-setting or super-setting and (z) if such implementation includes any optional components of the Specification, includes the complete and fully-compliant implementation of the requirements of such optional components, without sub-setting or super-setting. If You are a legal entity, You may distribute copies made under clause (b) only to Your employees for their use solely in their work for You, on the condition that such persons are provided with a copy of this Agreement and made aware of its terms prior to or concurrent with such distribution; otherwise, You may not distribute copies of the Specification.

2. **YOUR LICENSE GRANT.** You hereby grant to iPass and its affiliates a royalty-free, worldwide, non-exclusive, perpetual and irrevocable license under all of Your present and future copyrights, trade secret rights, patent rights, and other intellectual property rights in any Feedback You provide to iPass, to copy, modify, perform, display, create derivate works of, and otherwise use such Feedback, and to make, have made, use, sell, offer to sell, import and otherwise exploit any implementation of such Feedback, including without limitation the right to sublicense such rights through multiple tiers of distribution. iPass may assign its rights under such license in conjunction with all or any part of its rights in the Specification. "Feedback" means any communication pertaining to the Specification, including without limitation changes, fixes, improvements, enhancements, applications, suggestions, ideas, concepts, know-how, techniques, data, translations, reformatting, and the like.

3. **RESTRICTIONS ON USE.** Except as expressly permitted in Section 1 (if at all), You may not (a) copy, translate, modify, create derivative works of, or otherwise use the Specification or any part thereof, (b) distribute, sell, assign, pledge, sublicense, lease, loan, rent, or otherwise transfer the Specification or any part thereof in any form to another person, (c) remove from the Specification, or alter, any of the trademarks, trade names, logos, patent or copyright notices or other proprietary notices or markings, or add any other notices or markings to the Specification, or (d) permit any other party to do any of the foregoing under clauses (a) through (c). iPass does not grant to You any express or implied licenses or rights to any enabling technologies or systems that may be necessary to develop, demonstrate, make, use or sell an Implementation.

4. **OWNERSHIP.** You agree that the Specification and all intellectual property rights therein are owned by iPass. iPass reserves title and all rights to and interests in the Specification and the Implementations not expressly granted to You in Section 1, including without limitation all patent rights, copyrights, trademarks, trade names, trade secrets and other proprietary rights.

5. **MAINTENANCE AND UPDATES.** You understand that iPass may update the Specification at any time but is under no obligation to inform You of or furnish to You such updates pursuant to this Agreement. This Agreement does not grant You any right, license, or interest in or to any direct support, maintenance, improvements, modifications, enhancements, or updates to the Specification or supporting documentation.

6. **NO WARRANTIES.** THE SPECIFICATION IS PROVIDED AND LICENSED TO YOU "AS IS". The Specification could include technical inaccuracies or typographical errors, and changes may be periodically added to the information therein. (Such changes will generally be incorporated into new versions of the Specification, if any.) You assume the entire risk as to, and acknowledge that You rely solely at Your own risk on, results and performance arising out of the use of the Specification or any implementation thereof. Should the Specification prove to have defects in any way, You assume the entire cost of all servicing, repair or correction arising in connection with such defects.

IPASS DISCLAIMS ALL CONDITIONS, REPRESENTATIONS AND WARRANTIES, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING WITHOUT LIMITATION ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, ACCURACY OF INFORMATIONAL CONTENT, SYSTEM INTEGRATION, AND NON-INFRINGEMENT OF THIRD PARTY RIGHTS, AND ANY WARRANTIES AGAINST INTERFERENCE WITH YOUR ENJOYMENT OF THE SPECIFICATION. WITHOUT LIMITING THE FOREGOING, IPASS SPECIFICALLY DISCLAIMS ALL WARRANTIES AND REPRESENTATIONS THAT THE SPECIFICATION, IPASS'S EFFORTS, OR ANY SYSTEM WITH WHICH YOU WILL USE THE SPECIFICATION OR AN IMPLEMENTATION THEREOF WILL MEET YOUR REQUIREMENTS, FULFILL ANY OF YOUR PARTICULAR PURPOSES OR NEEDS, OR THAT THE OPERATION OR IMPLEMENTATION OF THE SPECIFICATION OR ANY IMPLEMENTATION THEREOF WILL BE UNINTERRUPTED OR ERROR FREE. YOU ASSUME THE RESPONSIBILITY FOR THE SELECTION OF YOUR REQUIREMENTS, SOFTWARE, AND HARDWARE TO ACHIEVE YOUR INTENDED RESULTS. Some jurisdictions do not allow the disclaimer of implied warranties, so the above disclaimer may not apply to You, in which case the duration of any such implied warranties is limited to thirty (30) days from the date the Specification is first Acquired by You. In case of breach of such implied warranties, iPass's sole and exclusive obligation and liability and Your sole and exclusive remedy will be, at iPass's sole discretion, to (i) repair, correct, or work around any defect or (ii) provide a replacement copy of the Specification.

7. **LIMITATION OF LIABILITY.** IN NO EVENT WILL IPASS BE LIABLE FOR ANY LOST PROFITS, LOST BUSINESS, OR LOST DATA, OR FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, PUNITIVE, SPECIAL, OR INCIDENTAL DAMAGES ARISING FROM OR RELATING TO THIS AGREEMENT OR THE SPECIFICATION, INCLUDING WITHOUT LIMITATION DAMAGES THAT ARISE OUT OF YOUR USE OR INABILITY TO USE THE SPECIFICATION (AND ANY INTELLECTUAL PROPERTY SUBSISTING THEREIN), EVEN IF IPASS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

THE TOTAL CUMULATIVE LIABILITY OF IPASS IN CONNECTION WITH THIS AGREEMENT AND THE SPECIFICATION, WHETHER IN CONTRACT, TORT OR OTHERWISE, TO YOU OR ANY THIRD PARTY, WILL NOT EXCEED ONE HUNDRED DOLLARS ($100).

8. **INDEMNIFICATION.** You agree to defend, indemnify and hold iPass harmless from and against all loss, cost, liability, damage, and expense arising from or relating to Your use or misuse of the Specification or Your breach of this Agreement, including without limitation all fines, penalties, liabilities, damages, costs, and expenses incurred by iPass as a result of Your failure to comply with export control laws and regulations in accordance with Section 11.

9. **TERMINATION.** This Agreement is effective until terminated. You may terminate this Agreement any time by permanently destroying all copies of the Specification and by permanently discontinuing all use of the Specification. Unauthorized copying of the Specification, or failure to comply with the terms of this Agreement including without limitation the public distribution of any implementation of the Specification that is not an Implementation will result in automatic termination of this Agreement, without limiting any other rights or remedies of iPass. iPass may in its sole discretion terminate this Agreement upon notice to You if an Implementation by You gives rise to a lawsuit against iPass or its officers, directors, employees or agents (i) for which the indemnification in Section 8 does not apply, (ii) for which You assert that such indemnification does not apply, or (iii) for which, in iPass's reasonable opinion, You do not have the resources to reasonably

fulfill Your obligations under Section 8. Upon termination of this Agreement, the license granted in Section 1 will terminate and You must immediately destroy all copies of the Specification in Your possession and/or control. The remaining provisions of this Agreement will survive termination.

10. **U.S. GOVERNMENT USE.** The Specification may include commercial technical data as defined in 48 C.F.R. 12.211 (Sep 1995). Consistent with 48 C.F.R. 12.211 through 12.212, 48 C.F.R. 227.7202-1 through 227.7202-4 (Jun 1995), and 48 C.F.R. 252.227-7015 (Nov 1995), all U.S. Government end users acquire the Specification with only those rights set forth herein.

11. **EXPORT CONTROL.** You will comply with all applicable export and import control laws and regulations in Your use of the Specification and, in particular, You will not export or re-export the Specification or any Implementation without the required United States and foreign government licenses.

12. **MISCELLANEOUS.** This Agreement is the final, complete and exclusive agreement between the parties relating to the subject matter hereof, and supersedes all prior or contemporaneous understandings and agreements relating to such subject matter, whether oral or written. No waiver or modification of the Agreement will be valid unless signed by each party. The waiver of a breach of any term hereof will in no way be construed as a waiver of any other term or breach hereof. The headings in this Agreement do not affect its interpretation. You may not assign or transfer any of Your rights or obligations under this Agreement to a third party without the prior written consent of iPass. Any attempted assignment or transfer in violation of the foregoing will be void from the beginning. iPass may assign this Agreement without consent to any affiliate or to a successor in interest to all or a substantial part of iPass's business. If any provision of this Agreement is held by a court of competent jurisdiction to be unenforceable, the remaining provisions of this Agreement will remain in full force and effect. This Agreement is governed by the laws of the State of California without reference to conflict of laws principles that would require the application of the laws of any other jurisdiction. The United Nations Convention on Contracts for the International Sale of Goods is specifically disclaimed. All disputes arising out of this Agreement will be subject to the exclusive jurisdiction of the state and federal courts located in Santa Clara County, California, and the parties agree and submit to the personal and exclusive jurisdiction and venue of these courts. Should You have any question about this Agreement, or if You desire to contact iPass Inc., please contact us by mail at 3800 Bridge Parkway, Redwood Shores, CA 94065, by phone at 650-232-4100, or by email at legal@iPass.com.

Appendix B: Example HTTP and XML Message exchange between Smart Client and A SIMPLE ACCESS Gateway

The messages documented in this section are associated with their [*originator*], either the smart **client** or the access **gateway**.

Authentication Procedure Initiation [client]

```
GET / HTTP/1.0
```

Activation – Authentication Redirect [gateway]

HTTP 302 Found

{Other HTTP headers}

```
<HTML> <!--
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
  xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”

xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayParam.xsd">
  <Redirect>
      <AccessProcedure>1.0</AccessProcedure>
      <AccessLocation>12</AccessLocation>
      <LocationName>
ACMEWISP,Gate_14_Terminal_C_of_Newark_Airport
</LocationName>
 <LoginURL>https://www.acmewisp.com/login/?sid=4a4&try=1</LoginURL>
      <MessageType>100</MessageType>
      <ResponseCode>0</ResponseCode>
  </Redirect>
</WISPAccessGatewayParam>
--> </HTML>
```

Authentication Request [client] via SSL

```
POST /login/?sid=4a4&try=1 HTTP/1.0
```

{Other HTTP headers}

```
button=Login&UserName=WISP1/joseph@company.com&Password=xxxxx&FNAME=0&OriginatingServer=http://xxx.yyy.zzz.eee/
```

Authentication Reply [gateway] (Login Successful)

HTTP 200 OK

{Other HTTP headers}

```
<HTML> <!--
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayPara
m.xsd">
 <AuthenticationReply>
       <MessageType>120</MessageType>
       <ResponseCode>50</ResponseCode>
       <ReplyMessage>”Message of the Day”</ReplyMessage>
    <LogoffURL>http://www.acmewisp.com/logoff?ses=A134f3</LogoffURL>
 </AuthenticationReply>
</WISPAccessGatewayParam>
--> </HTML>
```

Authentication Reply [gateway] (Login rejected)

HTTP 200 OK

{Other HTTP headers}

```
<HTML> <!--
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayPara
m.xsd">
 <AuthenticationReply>
       <MessageType>120</MessageType>
       <ResponseCode>100</ResponseCode>
       <ReplyMessage>Invalid Password</ReplyMessage>
 </AuthenticationReply>
</WISPAccessGatewayParam>
--> </HTML>
```

Client-initiated Connection Termination (logoff) of Authenticated User [client]

```
GET /logoff?ses=A134f3
```

Logoff Reply [gateway]

HTTP 200 OK

{Other HTTP headers}

```
<HTML> <!--
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
   xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayPara
 m.xsd">
 <LogoffReply>
       <MessageType>130</MessageType>
       <ResponseCode>150</ResponseCode>
 </LogoffReply>
</WISPAccessGatewayParam>
--> </HTML>
```

Appendix C: Example Detailed XML Message exchange between Smart Client and a Gateway Implementing Authentication Result Polling

The messages documented in this section are associated with their [*originator*], either the smart **client** or the access **gateway**.

Authentication Procedure Initiation [client]

```
GET / HTTP/1.0
```

Activation - Proxy Reply [gateway]

```
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
 xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”
   xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayP
 aram.xsd">
 <Proxy>
       <MessageType>110</MessageType>
       <NextURL>http://www.acmewisp.com/proxypoll</NextURL>
       <ResponseCode>200</ResponseCode>
    <Delay>5</Delay>
 </Proxy>
</WISPAccessGatewayParam>
```

Activation - Redirect Reply [gateway]

```
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
 xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”
   xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayP
 aram.xsd">
 <Redirect>
       <AccessProcedure>1.0</AccessProcedure>
       <AccessLocation>12</AccessLocation>
       <LocationName>
       ACMEWISP,Gate_14_Terminal_C_of_Newark_Airport
    </LocationName>
       <LoginURL>http://www.acmewisp.com/login</LoginURL>
       <AbortLoginURL>http://www.acmewisp.com/abortlogin</AbortLoginURL>
```

```
        <MessageType>100</MessageType>
        <ResponseCode>0</ResponseCode>
    </Redirect>
</WISPAccessGatewayParam>
```

Authentication Request [client] via SSL

```
POST /login HTTP/1.0

button=Login&UserName=WISP1/joseph@company.com&Password=xxxxx&FNAME=0&Originating
Server=http://xxx.yyy.zzz.eee/
```

(a)Authentication Reply [gateway] (Login Result Pending-begin result polling)

```
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayParam
.xsd">
    <AuthenticationReply>
        <MessageType>120</MessageType>
        <ResponseCode>201</ResponseCode>
        <ReplyMessage>”Message of the Day”</ReplyMessage>
        <LoginResultsURL>http://www.acmewisp.com/loginpoll</LoginResultsURL>
    </AuthenticationReply>
</WISPAccessGatewayParam>
```

Client-initiated Authentication Result Poll [client]

GET /loginpoll

(a) Authentication Reply [gateway] (Result Still Pending-repeat polling operation again in 5 seconds)

```
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
    xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayPara
m.xsd">
<AuthenticationPollReply>
        <MessageType>140</MessageType>
        <ResponseCode>201</ResponseCode>
        <ReplyMessage>Authentication Pending</ReplyMessage>
        <Delay>5</Delay>
```

```
</AuthenticationPollReply>
</WISPAccessGatewayParam>
```

(b)Authentication Reply [gateway] (Login rejected)

```
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
   xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayP
 aram.xsd">
 <AuthenticationPollReply>
       <MessageType>140</MessageType>
       <ResponseCode>100</ResponseCode>
       <ReplyMessage>Invalid Password</ReplyMessage>
 </AuthenticationPollReply>
</WISPAccessGatewayParam>
```

(c)Authentication Reply [gateway] (Login Successful)

```
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
   xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayP
 aram.xsd">
 <AuthenticationPollReply>
       <LogoffURL>http://www.acmewisp.com/logoff</LogoffURL>
       <MessageType>140</MessageType>
       <ResponseCode>50</ResponseCode>
       <ReplyMessage>”Message of the Day”</ReplyMessage>
 </AuthenticationPollReply>
</WISPAccessGatewayParam>
```

Client-initiated Connection Termination (logoff) of Authenticated User [client]

GET `/logoff`

Logoff Reply [gateway]

```
<?xml version="1.0" encoding="UTF-8"?>
<WISPAccessGatewayParam
   xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xsi:noNamespaceSchemaLocation="http://www.acmewisp.com/WISPAccessGatewayPara
 m.xsd">
 <LogoffReply>
```

```
            <MessageType>130</MessageType>
            <ResponseCode>150</ResponseCode>
        </LogoffReply>
    </WISPAccessGatewayParam>
```

Appendix D: XML Schema

The following XML schema is strictly invalid. Its invalidity is due to a limitation of the W3C XMLSchema definition v1.0 which does not permit specification of 'maxOccurence="unbounded"' as an attribute of an xs:all type. While the schema is strictly invalid, it reflects the intention of its designers. This issue only applies to schema elements which contain a <ReplyMessage> element definition. Where present, the <ReplyMessage> element may be present zero or more times. The schema will become valid with the release of the v1.1 of the W3C XMLSchema definition.

```
<?xml version="1.0" encoding="UTF-8"?>
   <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <xs:element name="WISPAccessGatewayParam">
           <xs:complexType>
                  <xs:choice>
                  <xs:element name="Proxy" type="ProxyType"/>
                  <xs:element name="Redirect" type="RedirectType"/>
                  <xs:element name="AuthenticationReply" type="AuthenticationReplyType"/>
                  <xs:element name="AuthenticationPollReply"
type="AuthenticationPollReplyType"/>
                  <xs:element name="LogoffReply" type="LogoffReplyType"/>
                  <xs:element name="AbortLoginReply" type="AbortLoginReplyType"/>
           </xs:choice>
           </xs:complexType>
    </xs:element>
    <xs:simpleType name="AbortLoginURLType">
           <xs:restriction base="xs:anyURI"/>
    </xs:simpleType>
    <xs:simpleType name="NextURLType">
           <xs:restriction base="xs:anyURI"/>
    </xs:simpleType>
    <xs:simpleType name="AccessProcedureType">
           <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <xs:simpleType name="AccessLocationType">
           <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <xs:simpleType name="LocationNameType">
           <xs:restriction base="xs:string"/>
    </xs:simpleType>
```

```
<xs:simpleType name="LoginURLType">
	<xs:restriction base="xs:anyURI"/>
</xs:simpleType>
<xs:simpleType name="RedirectionURLType">
	<xs:restriction base="xs:anyURI"/>
</xs:simpleType>
<xs:simpleType name="MessageTypeType">
	<xs:restriction base="xs:integer"/>
</xs:simpleType>
<xs:simpleType name="ResponseCodeType">
	<xs:restriction base="xs:integer"/>
</xs:simpleType>
<xs:simpleType name="ReplyMessageType">
	<xs:restriction base="xs:string"/>
</xs:simpleType>
<xs:simpleType name="LoginResultsURLType">
	<xs:restriction base="xs:anyURI"/>
</xs:simpleType>
<xs:simpleType name="LogoffURLType">
	<xs:restriction base="xs:anyURI"/>
</xs:simpleType>
<xs:simpleType name="DelayType">
	<xs:restriction base="xs:integer"/>
</xs:simpleType>
<xs:complexType name="ProxyType">
	<xs:all>
		<xs:element name="MessageType" type="MessageTypeType"/>
		<xs:element name="ResponseCode" type="ResponseCodeType"/>
		<xs:element name="NextURL" type="NextURLType" minOccurs="0" maxOccurs="1"/>
		<xs:element name="Delay" type="DelayType" minOccurs="0" maxOccurs="1"/>
	</xs:all>
</xs:complexType>
<xs:complexType name="RedirectType">
	<xs:all>
		<xs:element name="AccessProcedure" type="AccessProcedureType"/>
		<xs:element name="AccessLocation" type="AccessLocationType"/>
		<xs:element name="LocationName" type="LocationNameType"/>
		<xs:element name="LoginURL" type="LoginURLType"/>
```

```
                <xs:element name="AbortLoginURL" type="AbortLoginURLType" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="MessageType" type="MessageTypeType"/>
                <xs:element name="ResponseCode" type="ResponseCodeType"/>
          </xs:all>
   </xs:complexType>
   <xs:complexType name="AuthenticationReplyType">
          <xs:all>
                <xs:element name="MessageType" type="MessageTypeType"/>
                <xs:element name="ResponseCode" type="ResponseCodeType"/>
                <xs:element name="ReplyMessage" type="ReplyMessageType" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element name="LoginResultsURL" type="LoginResultsURLType"
minOccurs="0" maxOccurs="1"/>
                <xs:element name="LogoffURL" type="LogoffURLType" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="RedirectionURL" type="RedirectionURLType"
minOccurs="0" maxOccurs="1"/>
          </xs:all>
   </xs:complexType>
   <xs:complexType name="AuthenticationPollReplyType">
          <xs:all>
                <xs:element name="MessageType" type="MessageTypeType"/>
                <xs:element name="ResponseCode" type="ResponseCodeType"/>
                <xs:element name="ReplyMessage" type="ReplyMessageType" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element name="Delay" type="DelayType" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="LogoffURL" type="LogoffURLType" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="RedirectionURL" type="RedirectionURLType"
minOccurs="0" maxOccurs="1"/>
          </xs:all>
   </xs:complexType>
   <xs:complexType name="LogoffReplyType">
          <xs:sequence>
                <xs:element name="MessageType" type="MessageTypeType"/>
                <xs:element name="ResponseCode" type="ResponseCodeType"/>
          </xs:sequence>
   </xs:complexType>
   <xs:complexType name="AbortLoginReplyType">
          <xs:sequence>
```

```
                <xs:element name="MessageType" type="MessageTypeType"/>
                <xs:element name="ResponseCode" type="ResponseCodeType"/>
                <xs:element name="LogoffURL" type="LogoffURLType" minOccurs="0"
maxOccurs="1"/>
          </xs:sequence>
    </xs:complexType>
```