INBOX.LV OPEN ID - Technical Specification

To provide Open ID login on third party page it is necessary to install the Open ID library, which is available at the following address https://github.com/openid

Next is described how to configure a test app for interaction with Inbox OpenID. After installing the Open ID client, create or edit index.php in project folder.

Example of index.php file (This example is available at https://github.com/openid/php-openid/tree/master/examples/consumer).

```
<?php
require_once "common.php";
global $pape_policy_uris;
?>
<html>
<head><title>PHP OpenID Authentication Example</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8"/>
</head>
<style type="text/css">
    * {
        font-family: verdana, sans-serif;
    }
    body {
        width: 60em;
        margin: 1em;
    }
    div {
        padding: .5em;
    }
    table {
        margin: none;
        padding: none;
    }
    .alert {
        border: 1px solid #e7dc2b;
        background: #fff888;
    }
    .success {
        border: 1px solid #669966;
        background: #88ff88;
    }
    .error {
        border: 1px solid #ff0000;
        background: #ffaaaa;
    }
    #verify-form {
        border: 1px solid #777777;
        background: #dddddd;
        margin-top: 1em;
        padding-bottom: 0em;
    }</style>
<body><h1>PHP OpenID Authentication Example</h1>
<p>This example consumer uses the
    <ahref
    ="http://github.com/openid/php-openid">PHPOpenID</a> library. It just verifies that
the URL that you enteris your
    identity URL.
</p>


<?php if (isset($msg)) {
    print "<div class=\"alert\">$msg</div>";
} ?><?php if (isset($error)) {
    print "<div class=\"error\">$error</div>";
} ?><?php if (isset($success)) {
    print "<div class=\"success\">$success</div>";
} ?>
<div id="verify-form">
    <form method="get" action="try_auth.php" id="auth_form">
        Autorizacion button: <a href="#"
onclick="document.getElementById('auth_form').submit();return false;">
            <img src="INBOX_OPEN_ID_IMG" style="border:solid 1px #aaa;"
align="absmiddle"/>
        </a>
        <!--Identity&nbsp;URL:--><input type="hidden" name="action" value="verify"/>
        <input type="hidden" name="openid_identifier"
value="http://login.inbox.lv/openid/INBOX_OPEN_ID/"/>
        <!--<p>Optionally, request these PAPE policies:</p><p>
        <?php
        foreach ($pape_policy_uris as $i => $uri) {
            print "<input type=\"checkbox\" name=\"policies[]\" value=\"$uri\" />";
            print "$uri<br/>";
        }
        ?></p><input type="submit" value="Verify" />--></form>
</div>
</body>
</html>
```

In this file replace INBOX_OPEN_ID to public key, which will be provided to you after registration by the administration of portal inbox.lv. Use icons from the library of graphic materials OPEN ID as the parameter INBOX_OPEN_ID_IMG. Change the settings and update the file on the server.
The following is an example how to initiate an authorization try_auth.php (common.php file delivers client-side for OPEN ID).

```
<?php

require_once "common.php";
session_start();

function getOpenIDURL() {
    // Render a default page if we got a submission without an openid
    if (empty($_GET['openid_identifier'])) {
        $error = "Expected an OpenID URL.";
        include 'index.php';
        exit(0);
    }

    return $_GET['openid_identifier'];
}

function run() {
    $openid = getOpenIDURL();
    $consumer = getConsumer();
```

Start of the OpenID authentication process:

```
$auth_request = $consumer->begin($openid);
```

Request means that there is no authentication possible, and we cannot launch the Open ID

```
    if (!$auth_request) {
        displayError("Authentication error; not a valid OpenID.");
    }

    $sreg_request = Auth_OpenID_SRegRequest::build(
                                     // Required
                                     array('nickname'),
                                     // Optional
                                     array('fullname', 'email'));

    if ($sreg_request) {
        $auth_request->addExtension($sreg_request);
    }

  $policy_uris = null;
  if (isset($_GET['policies'])) {
       $policy_uris = $_GET['policies'];
  }

    $pape_request = new Auth_OpenID_PAPE_Request($policy_uris);
    if ($pape_request) {
        $auth_request->addExtension($pape_request);
    }
```

Redirecting users to the OpenID server for authentication.
Store the token for this authentication so we can verify the response.
For OpenID 1, send a redirect. For OpenID 2, use a Javascript form to send a POST request to the server.

```
    if ($auth_request->shouldSendRedirect()) {
        $redirect_url = $auth_request->redirectURL(getTrustRoot(),
                                                   getReturnTo());
```

If the redirect URL can't be built, display an error message.

```
      if (Auth_OpenID::isFailure($redirect_url)) {
          displayError("Could not redirect to server: " . $redirect_url->message);
        } else {
```

Send redirect.

```
            header("Location: ".$redirect_url);
        }
    } else {
```

Generate form markup and render it.

```
            $form_id = 'openid_message';
            $form_html = $auth_request->htmlMarkup(getTrustRoot(), getReturnTo(),
                                                       false, array('id' => $form_id));
```

Display an error if the form markup couldn't be generated otherwise, render the HTML.

```
         if (Auth_OpenID::isFailure($form_html)) {
                displayError("Could not redirect to server: " . $form_html->message);
            } else {
                print $form_html;
            }
        }
    }

    run();

    ?>
```

Next is a sample file finish_auth.php, responsible for processing the data after authorization (common.php file delivers client-side for OPEN ID).

```
    <?php

    require_once "common.php";
    session_start();

    function escape($thing) {
        return htmlspecialchars($thing);
    }

    function run() {
        $consumer = getConsumer();
```

Complete the authentication process using the server's response.

```
        $return_to = getReturnTo();
        $response = $consumer->complete($return_to);
    //    print_r($response);
    //    exit;
```

Check the response status.

```
        if ($response->status == Auth_OpenID_CANCEL) {
```

This means the authentication was cancelled.

```
          $msg = 'Verification cancelled.';
        } else if ($response->status == Auth_OpenID_FAILURE) {
```

Authentication failed; display the error message.

```
            $msg = "OpenID authentication failed: " . $response->message;
        } else if ($response->status == Auth_OpenID_SUCCESS) {
```

This means the authentication succeeded; extract the identity URL and Simple Registration data (if it was returned).

```
            $openid = $response->getDisplayIdentifier();
            $esc_identity = escape($openid);

            $success = sprintf('You have successfully verified ' .
                               '<a href="%s">%s</a> as your identity.',
                               $esc_identity, $esc_identity);

            if ($response->endpoint->canonicalID) {
                $escaped_canonicalID = escape($response->endpoint->canonicalID);
                $success .= ' (XRI CanonicalID: '.$escaped_canonicalID.') ';
            }

            $sreg_resp = Auth_OpenID_SRegResponse::fromSuccessResponse($response);

            $sreg = $sreg_resp->contents();

            if (@$sreg['email']) {
```

You also returned email, nickname and full name

```
               $success .= " You also returned '".escape($sreg['email']).
                    "' as your email.";
            }

            if (@$sreg['nickname']) {
                $success .= " Your nickname is '".escape($sreg['nickname']).
                    "'.";
            }

            if (@$sreg['fullname']) {
                $success .= " Your fullname is '".escape($sreg['fullname']).
                    "'.";
            }

    $pape_resp = Auth_OpenID_PAPE_Response::fromSuccessResponse($response);

    if ($pape_resp) {
                if ($pape_resp->auth_policies) {
                    $success .= "<p>The following PAPE policies affected the
    authentication:</p><ul>";

                    foreach ($pape_resp->auth_policies as $uri) {
                        $escaped_uri = escape($uri);
                        $success .= "<li><tt>$escaped_uri</tt></li>";
                    }

                    $success .= "</ul>";
                } else {
                    $success .= "<p>No PAPE policies affected the authentication.</p>";
                }

                if ($pape_resp->auth_age) {
                    $age = escape($pape_resp->auth_age);
                    $success .= "<p>The authentication age returned by the " .
```

```
                        "server is: <tt>".$age."</tt></p>";
                }

                if ($pape_resp->nist_auth_level) {
                    $auth_level = escape($pape_resp->nist_auth_level);
                    $success .= "<p>The NIST auth level returned by the " .
                        "server is: <tt>".$auth_level."</tt></p>";
                }

} else {
                $success .= "<p>No PAPE response was sent by the provider.</p>";
}
    }

    include 'index.php';
}

header('Content-type: text/html; charset=UTF-8');
run();

?>
```

Copy or update the index.php file on your web server. Open the index.php page in the browser and check how the OPEN ID client works with Inbox servers.

Important! Your web server should be allowed an HTTP / HTTPS connection to the Open ID inbox.lv server LOGIN.INBOX.LV.

Technical contact:
mailto: support@inbox.lv
tel.+371 67359626, 82 000
https://company.inbox.lv/inboxid/