OpenID Connect for MyProxy  
Protocol Specification

Version 0.1 (Sep 2013)

Jim Basney <jbasney@illinois.edu>

Jeff Gaynor <gaynor@illinois.edu>

Wendy Edwards <wedwards@illinois.edu>

# Table of Contents

[Table of Contents](#h.duvdeb4x8pd4)

[Introduction](#h.m4h7z0w9smu2)

[OpenID Connect Endpoints](#h.a02pt7cj6afq)

[OpenID Connect Required Endpoints](#h.npq0vb8f7pjb)

[OpenID Connect Optional Endpoints](#h.z9m05mh17r1w)

[OA4MP Application Specific Endpoints](#h.r38s6eor9470)

[Sequence Diagram](#h.sh0l5veqkeat)

[Message Specification](#h.yyrjwgsxg1c4)

[Client Requests Authorization](#h.cfm05rlw4qy3)

[Authorization Request](#h.dzjw7gmgfel)

[Methods](#h.6l1nu9pigt6g)

[Example](#h.3sk2qbnsdz74)

[Service Provider Grants Authorization](#h.p91em1llm1q3)

[Authorization Server Authenticates End User](#h.6w3wk7x9o6gv)

[Obtaining Consent](#h.5cgftm2ue3xc)

[End User Agrees](#h.4vv1rx37ob6t)

[End User Declines](#h.7lrcd3nixbnt)

[Error Response](#h.az4x7zwa016j)

[Example](#h.v4nia921o0r1)

[Authorization Response](#h.6wef7vutcx9v)

[Example](#h.y4cv5uwhd76d)

[Token Endpoint](#h.7e2wixfw4r0y)

[Getting an Access Token](#h.bf8sabs09jtn)

[Access Token Request](#h.nq5705wuugqx)

[Example](#h.oawpdxmpxhwd)

[Access Token Response](#h.vanmego4hb64)

[Example](#h.nv5hnfu3aac7)

[UserInfo Endpoint](#h.bxijp9m0qm1e)

[UserInfo Request](#h.ndq4whgq53o6)

[Example](#h.8r8x6nmgs6ug)

[UserInfo Response](#h.mtu6qfbuahvq)

[Example](#h.fjbx7x79b68d)

[GetCert Endpoint](#h.3khs91kr9igo)

[Get Certificate Request](#h.ycur897ixgn8)

[Example](#h.7hq3rjt46nhq)

[Get Certificate Response](#h.i4ezrdmtzs9g)

[Example](#h.yyizh4ugv79y)

# Introduction

This document is an initial draft of a protocol specification for OpenID Connect (OAuth 2.0) to be used in OAuth for MyProxy. Using OpenID Connect Authorization Code Flow. The intended audience for this document is developers of OpenID Connect for MyProxy clients and servers.

# OpenID Connect Endpoints

## OpenID Connect Required Endpoints

|  |  |
| --- | --- |
| **Description** | **Default Endpoint URL** |
| Authorization | https://myproxy.example.edu/oauth2/authorize |
| Token | https://myproxy.example.edu/oauth2/token |

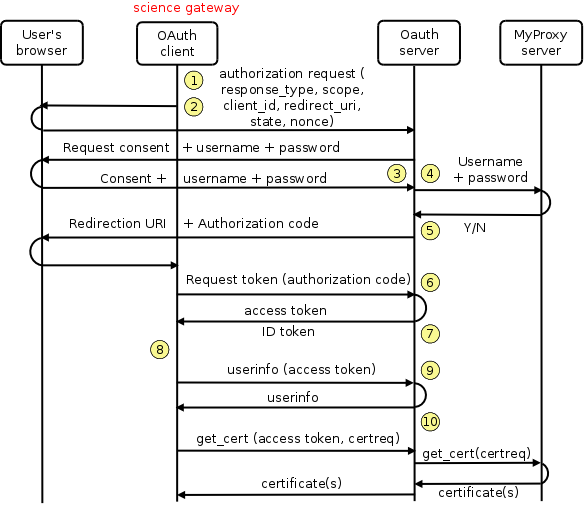
## OpenID Connect Optional Endpoints

|  |  |
| --- | --- |
| **Description** | **Default Endpoint URL** |
| UserInfo | https://myproxy.example.edu/oauth2/userinfo |

## OA4MP Application Specific Endpoints

|  |  |
| --- | --- |
| **Description** | **Default Endpoint URL** |
| Resource Request | http://myproxy.example.edu/oauth2/getcert |
| Client Registration | http://myproxy.example.edu/oauth2/register |

# Sequence Diagram



The numbers correspond to the steps in the authorization code flow (<http://openid.net/specs/openid-connect-standard-1_0.html#code_flow>).

1. Client prepares an [Authorization Request](#h.cfm05rlw4qy3) containing the desired request parameters.
2. Client [sends a request](#h.cfm05rlw4qy3) to the Authorization Server.
3. Authorization Server [Authenticates](#h.6w3wk7x9o6gv) the End-User.
4. Authorization Server Obtains the [End-User Consent/Authorization](#h.5cgftm2ue3xc).
5. Authorization Server Sends the [End-User back to the Client with an Authorization Code](#h.6wef7vutcx9v).
6. Client [requests a response](#h.nq5705wuugqx) using the Authorization Code at the Token Endpoint, per Section 3.
7. Client [receives a response](#h.vanmego4hb64) that contains an Access Token and ID Token in the response body.
8. Client validates the ID Token and retrieves the End-User's subject identifier.
9. (OPTIONAL) Client [accesses the UserInfo Endpoint](#h.ndq4whgq53o6) with the Access Token, per Section 4.
10. (OPTIONAL) Client [receives UserInfo](#h.mtu6qfbuahvq) Response.

# Message Specification

Per <http://openid.net/specs/openid-connect-standard-1_0.html#TLS_requirements>, implementations must support TLS and TLS is used to communicate with the authorization endpoint. A TLS server certificate check must be performed when using TLS.

All OAuth protocol parameters MUST be transmitted via HTTP request URI query (i.e., URL encoded).

OpenID Connect specifications require that the authorization server support both GET and POST request methods, and the client may use either to send an authorization request to the server. The authorization URL must use HTTPS (<http://openid.net/specs/openid-connect-standard-1_0.html#auth_request>).

# Client Requests Authorization

## Authorization Request

Per <http://openid.net/specs/openid-connect-standard-1_0.html#RequestParameters>, these are the parameters used in OpenID Connect authorization requests. See also <http://openid.net/specs/openid-connect-messages-1_0.html#Messages> for more detail.

|  |  |  |
| --- | --- | --- |
| **Name** | **Required** | **Description** |
| response\_type | REQUIRED | Must be code. (A future version will support id\_token and token for OIDC compliance.) |
| client\_id | REQUIRED | Previously registered OAuth 2.0 Client Identifier. |
| scope | REQUIRED | Space delimited, case sensitive list of ASCII OAuth 2.0 scope values. The following values are supported:   * openid (REQUIRED) * edu.uiuc.ncsa.myproxy.getcert (OPTIONAL) * profile (OPTIONAL) * email (OPTIONAL) * address (OPTIONAL) * phone (OPTIONAL) * offline\_access (OPTIONAL)   For more information about scope, please see <http://openid.net/specs/openid-connect-messages-1_0.html#scopes>. |
| redirect\_uri | REQUIRED | Previously registered Redirection URI. See <http://openid.net/specs/openid-connect-registration-1_0.html> |
| state | RECOMMENDED | Opaque value to maintain state between request and callback. Recommended for CSRF protection. See [RFC 6749 Section 10.12](http://tools.ietf.org/html/rfc6749#section-10.12). |
| nonce | REQUIRED | String used to associate client session with ID token. See OIDC Section 2.2.1.1. |
| prompt | OPTIONAL | Space delimited, case sensitive list of ASCII string values. Defined values are: none, login, consent, and select\_account |

Note: Not using the STATE parameter has led to some vulnerabilities in OAuth.

## Methods

There are three OpenID Connect methods to create and send a request to the authorization endpoint: [Simple Request Method](http://openid.net/specs/openid-connect-standard-1_0.html#SimpleRequestMethod), [Request Parameter Method](http://openid.net/specs/openid-connect-standard-1_0.html#req_param_method), and [Request File Method](http://openid.net/specs/openid-connect-standard-1_0.html#Request_File_Method). Only the Simple Request Method is supported at this time.

## Example

**Client redirects browser with request**

HTTP/1.1 302 Found  
 Location: https://myproxy.example.edu/authorize?  
 response\_type=code  
 &client\_id=s6BhdRkqt3  
 &redirect\_uri=https%3A%2F%2Fmyproxy.example.edu%2Fcb  
 &scope=openid%20edu.uiuc.ncsa.myproxy.getcert  
 &nonce=n-0S6\_WzA2Mj  
 &state=af0ifjsldkj

**Browser issues GET to Authorization URL**

(From <http://openid.net/specs/openid-connect-standard-1_0.html#SimpleRequestMethod>)

GET https://myproxy.example.edu/op/authorize?  
 response\_type=code  
 &client\_id=s6BhdRkqt3  
 &redirect\_uri=https%3A%2F%2Fmyproxy.example.edu%2Fcb  
 &scope=openid%20edu.uiuc.ncsa.myproxy.getcert  
 &nonce=n-0S6\_WzA2Mj  
 &state=af0ifjsldkj

# Service Provider Grants Authorization

## Authorization Server Authenticates End User

The authorization server must validate the request according to Section 5,1 of OpenID Connect Messages 1.0. See <http://openid.net/specs/openid-connect-messages-1_0.html#Validation> . If the end-user is not already logged in or the authorization request contains a prompt parameter with the value "login," the authorization server must attempt to authenticate the user. If the user is not already logged in and the request contains a prompt parameter with the value "none", the authorization server must return an error.

## Obtaining Consent

In OpenID connect, the authorization server must obtain consent and authorization from the end user if the client has not been pre-authorized or the authorization request contains the prompt parameter with the value "consent." If the request has not been pre-authorized and the authorization request contains the prompt parameter with the value "none", the authorization server must return an error.

### End User Agrees

If the end user agrees, the authorization server issues an authorization response. Please see <http://openid.net/specs/openid-connect-standard-1_0.html#art_res_ok> and <http://openid.net/specs/openid-connect-messages-1_0.html#AuthzResponse> .

### End User Declines

If the end user declines, the authorization server must return an error response. See <http://openid.net/specs/openid-connect-standard-1_0.html#authz_error> .

## Error Response

|  |  |  |
| --- | --- | --- |
| **Name** | **Required** | **Description** |
| error | REQUIRED | Error code. |
| error\_description | OPTIONAL | Human-readable ASCII encoded text description of the error. |
| error\_uri | OPTIONAL | Human-readable ASCII encoded text description of the error. |
| state | REQUIRED if the Authorization Request included the state parameter | OAuth 2.0 state value. Set to the value received from the Client. |

### Example

From <http://openid.net/specs/openid-connect-standard-1_0.html#authz_error>

HTTP/1.1 302 Found  
 Location: https://client.example.org/cb?  
 error=invalid\_request  
 &error\_description=  
 the%20request%20is%20not%20valid%20or%20malformed  
 &state=af0ifjsldkj

## Authorization Response

|  |  |  |
| --- | --- | --- |
| **Name** | **Required** | **Description** |
| token\_type | REQUIRED | Must be set to bearer. |
| code | REQUIRED | The authorization code generated by the authorization server (per OAuth 2.0 spec). |
| state | REQUIRED if the Authorization Request included the state parameter | The exact value received from client (if applicable). |

Note: the redirection URL should be part of the authorization response. For more information, see <http://tools.ietf.org/html/rfc6749#section-4.1.2> .

### Example

From <http://openid.net/specs/openid-connect-standard-1_0.html#art_res_ok>

Response type is code

Server redirects browser to client with a URL that has a code parameter.

HTTP/1.1 302 Found  
 Location: https://myproxy.example.edu/cb?  
 code=Qcb0Orv1zh30vL1MPRsbm-diHiMwcLyZvn1arpZv-Jxf\_11jnpEX3Tgfvk  
 &state=af0ifjsldkj

# Token Endpoint

The token endpoint handles requests for retrieving and refreshing access tokens, ID tokens, and other variables. Clients must send requests to the token endpoint via HTTP POST. Clients may also provide authentication parameters in the requests. See <http://openid.net/specs/openid-connect-messages-1_0.html#token_ep> for specifications.

## Getting an Access Token

To retrieve an access token using the authorization code flow, a client must have an authorization code. See <http://openid.net/specs/openid-connect-standard-1_0.html#code_flow> for more information about the authorization code flow.

To obtain an access token, refresh token, or ID token, the client must authenticate to the token endpoint. See <http://openid.net/specs/openid-connect-standard-1_0.html#AccessTokenRequest> .

The authorization server must respond to token requests with success or error messages. See <http://openid.net/specs/openid-connect-standard-1_0.html#AccessTokenResponse> and <http://openid.net/specs/openid-connect-standard-1_0.html#AccessTokenErrorResponse> .

## Access Token Request

|  |  |  |
| --- | --- | --- |
| **Name** | **Required** | **Description** |
| grant\_type | REQUIRED | Value MUST be set to "authorization\_code". |
| code | REQUIRED | The authorization code received from the authorization server. |
| redirect\_uri | REQUIRED | Must be identical to value in request |
| client\_id | REQUIRED | Only authenticated clients are supported. |

See <http://tools.ietf.org/html/rfc6749#section-4.1.3>

### Example

From <http://openid.net/specs/openid-connect-standard-1_0.html#AccessTokenRequest>

POST /token HTTP/1.1  
 Host: myproxy.example.edu  
 Content-Type: application/x-www-form-urlencoded  
 Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW  
 grant\_type=authorization\_code&code=SplxlOBeZQQYbYS6WxSbIA&redirect\_uri=https%3A%2F%2Fclient.example.org%2Fcb&client\_id=id12345

## Access Token Response

|  |  |  |
| --- | --- | --- |
| **Name** | **Required** | **Description** |
| id\_token | REQUIRED if response\_type includes id\_token. | ID Token value associated with the authenticated session. (Not included in Oauth 2.0.) |
| access\_token | REQUIRED | The access token issued by the authorization server. |
| token\_type | REQUIRED | The type of the token issued. Currently only "Bearer" is supported. |
| expires\_in | RECOMMENDED | The lifetime in seconds of the access token. |
| refresh\_token | OPTIONAL | The refresh token, which can be used to obtain new access tokens using the same authorization grant |
| scope | REQUIRED (unless identical to scope requested by client) | The scope of the access token. Included in cases when the client requests something (like email) that the particular server doesn’t support. |

See <http://tools.ietf.org/html/rfc6749#section-5.1>

### Example

From <http://openid.net/specs/openid-connect-standard-1_0.html#AccessTokenResponse>

HTTP/1.1 200 OK  
 Content-Type: application/json  
 Cache-Control: no-store  
 Pragma: no-cache  
  
 {  
 "access\_token": "SlAV32hkKG",  
 "token\_type": "Bearer",  
 "refresh\_token": "8xLOxBtZp8",  
 "expires\_in": 3600,  
 "id\_token": "eyJhbGciOiJSUzI1NiJ9.ew0KICAgICJpc3MiOiAiaHR0cDovL  
 3NlcnZlci5leGFtcGxlLmNvbSIsDQogICAgInVzZXJfaWQiOiAiMjQ4Mjg5NzYxM  
 DAxIiwNCiAgICAiYXVkIjogInM2QmhkUmtxdDMiLA0KICAgICJub25jZSI6ICJuL  
 TBTNl9XekEyTWoiLA0KICAgICJleHAiOiAxMzExMjgxOTcwLA0KICAgICJpYXQiO  
 iAxMzExMjgwOTcwDQp9.lsQI\_KNHpl58YY24G9tUHXr3Yp7OKYnEaVpRL0KI4szT  
 D6GXpZcgxIpkOCcajyDiIv62R9rBWASV191Akk1BM36gUMm8H5s8xyxNdRfBViCa  
 xTqHA7X\_vV3U-tSWl6McR5qaSJaNQBpg1oGPjZdPG7zWCG-yEJC4-Fbx2FPOS7-h  
 5V0k33O5Okd-OoDUKoFPMd6ur5cIwsNyBazcsHdFHqWlCby5nl\_HZdW-PHq0gjzy  
 JydB5eYIvOfOHYBRVML9fKwdOLM2xVxJsPwvy3BqlVKc593p2WwItIg52ILWrc6A  
 tqkqHxKsAXLVyAoVInYkl\_NDBkCqYe2KgNJFzfEC8g"  
 }

# UserInfo Endpoint

The UserInfo endpoint is an OAuth 2.0 protected resource that returns claims about the authenticated end user. To obtain these claims, the client sends a request via either the GET or POST method to the UserInfo Endpoint. See <http://openid.net/specs/openid-connect-messages-1_0.html#userinfo> . The UserInfo endpoint complies with the OAuth 2.0 Bearer Token Usage (RFC6750) specification. See <http://tools.ietf.org/html/rfc6750> . The UserInfo request must include an access token obtained from a successful authorization request.

The UserInfo claims are returned as members of a JSON object unless a signed or encryped response was requested during client registration. See <http://openid.net/specs/openid-connect-messages-1_0.html#UserInfoResponse> . This is a link to standard claims, e.g., name, profile, picture: <http://openid.net/specs/openid-connect-messages-1_0.html#StandardClaims> .

## UserInfo Request

|  |  |  |
| --- | --- | --- |
| **Name** | **Required** | **Description** |
| access\_token | REQUIRED | Access Token obtained from an OpenID Connect Authorization Request |

See <http://openid.net/specs/openid-connect-messages-1_0.html#UserInfoRequest>

### Example

From <http://openid.net/specs/openid-connect-standard-1_0.html#UserInfoRequest>

GET /userinfo HTTP/1.1  
 Host: myproxy.example.edu  
 Authorization: Bearer SlAV32hkKG

## UserInfo Response

The UserInfo claims are returned in the form of a JSON object. More information about the claim fields can be found at <http://openid.net/specs/openid-connect-messages-1_0.html#StandardClaims>

### Example

From <http://openid.net/specs/openid-connect-standard-1_0.html#UserInfoResponse>

HTTP/1.1 200 OK  
 Content-Type: application/json  
  
 {  
 "sub": "jdoe",  
 "name": "Jane Doe"  
 "given\_name": "Jane",  
 "family\_name": "Doe",  
 "email": "janedoe@example.com",  
 "picture": "http://example.com/janedoe/me.jpg"  
 }

The Subject (sub) member contains the MyProxy username.

# GetCert Endpoint

The GetCert endpoint is an OAuth 2.0 protected resource that returns an X.509 certificate chain (one or more certificates). To obtain the certificate, the client sends a GET request including an access token obtained from a successful authorization request. The certificate chain is returned as in OAuth for MyProxy 1.0 (except without the username).

## Get Certificate Request

|  |  |  |
| --- | --- | --- |
| Name | Required | Description |
| access\_token | REQUIRED | OIDC (OAuth 2.0) access token obtained with token request |
| certreq | REQUIRED | Base64 encoding of a DER format PKCS#10 certificate request. It MUST contain a 2048 bit RSA key. |
| certlifetime | OPTIONAL | Requested certificate lifetime (in seconds) |

### Example

GET /getcert?certlifetime=950400&certreq=MIICVDCCATwCAQAwETEPMA0GA1UEAxMGaWdub3JlMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB%0ACgKCAQEAhUKrBs9%2B1GLUmfWjluNZVjud7%2Fnin0sdmOYQHB2a2pBqdSQ3hG2wG0x9%2FNsst9AGiQ%2Fh%0AQ1LoR4uOARyBd8d5cJ7UOwN%2BtktkMovLgM4GcnCkVPsUEcA7ZrbbCLWzfTgU25PuCdrlFF8xaahX%0AvR%2Bivcs%2FKKjAV4UCTNI9ft%2Bwxg%2FE5JbQGpQZrIi8o%2B79MwzgdxvJfFVg0ZBDzNEB%2F7n10TYSW0Ez%0AxmJABK6EouuaVmZCmLxoVfpwrn1%2BgfJPAPCWb27CXMCKt5zHmBKG7LbKD3hJhyZ25MzZEu4R67eT%0AAEVuy5MaydhcW8rP%2FszGwZ5r%2B%2BtCFSCmNet9bwmaZkdlsQIDAQABMA0GCSqGSIb3DQEBBQUAA4IB%0AAQBTu7r3D%2BUMHnb6JPyTdADmep%2BxBFI21wS6Z5rckvCAzAZSlcRxfGGZhrdgoOgfbE80FP1lhn1%2B%0Agvm13ku%2B4kCc1I8r7FwIOs7vCd2g%2B6gus%2BvgBM0hCxfuyNvRzbLtMbudj%2BPOReQMf0Y6%2Bng89DmW%0AXtm2J6ZmpaQx3fNmJ8KFtuZdzdIjQhMg6772fKTDNOvThhtrXnch%2FWt%2BTg4jES0vWzFLL4OgFbdl%0ADQX1HZXIoCjk%2BnVwTsPwm8E55p3qHKQ6fIMn0%2BCPBjz%2F6PiqxpquR9kOkqLNAXnbCOb5XwMTM26P%0AC5WtKkEnFKAdCIDAS0Uv34fwZ%2BO7fHy2eestubpf HTTP/1.1  
 Host: myproxy.example.edu  
 Authorization: Bearer SlAV32hkKG

## Get Certificate Response

A standard HTTP response. Any return code other than 200 indicates an error. The body of the response contains the PEM encoded certificate chain with Content-Type: text/plain.

### Example

HTTP/1.1 200 OK

Content-Type: text/plain

-----BEGIN CERTIFICATE-----

MIIDzzCCAregAwIBAgIEFzQEITANBgkqhkiG9w0BAQUFADCBhTELMAkGA1UEBhMC

VVMxODA2BgNVBAoTL05hdGlvbmFsIENlbnRlciBmb3IgU3VwZXJjb21wdXRpbmcg

QXBwbGljYXRpb25zMRMwEQYDVQQDEwpKaW0gQmFzbmV5MRIwEAYDVQQDEwkxMjA1

MzU4NDgxEzARBgNVBAMTCjE4NzAwMzExMTAwHhcNMTMwOTA5MTYwNDIwWhcNMTMw

OTEwMDQwNzU5WjCBmTELMAkGA1UEBhMCVVMxODA2BgNVBAoTL05hdGlvbmFsIENl

bnRlciBmb3IgU3VwZXJjb21wdXRpbmcgQXBwbGljYXRpb25zMRMwEQYDVQQDEwpK

aW0gQmFzbmV5MRIwEAYDVQQDEwkxMjA1MzU4NDgxEzARBgNVBAMTCjE4NzAwMzEx

MTAxEjAQBgNVBAMTCTM4OTI4NDg5NzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC

AQoCggEBAIVCqwbPftRi1Jn1o5bjWVY7ne/54p9LHZjmEBwdmtqQanUkN4RtsBtM

ffzbLLfQBokP4UNS6EeLjgEcgXfHeXCe1DsDfrZLZDKLy4DOBnJwpFT7FBHAO2a2

2wi1s304FNuT7gna5RRfMWmoV70for3LPyiowFeFAkzSPX7fsMYPxOSW0BqUGayI

vKPu/TMM4HcbyXxVYNGQQ8zRAf+59dE2EltBM8ZiQASuhKLrmlZmQpi8aFX6cK59

foHyTwDwlm9uwlzAirecx5gShuy2yg94SYcmduTM2RLuEeu3kwBFbsuTGsnYXFvK

z/7MxsGea/vrQhUgpjXrfW8JmmZHZbECAwEAAaMxMC8wDgYDVR0PAQH/BAQDAgSw

MB0GCCsGAQUFBwEOAQH/BA4wDDAKBggrBgEFBQcVATANBgkqhkiG9w0BAQUFAAOC

AQEALcInsL20lw3eG4T48Qu1nCkdiyzJ0n7pBC1lhUO7DNBUfqi84YZX61yJjN/a

cLHS9cbBRNHkXD/irI1+4EGafeaNIhxVJ2qOpb5wdKefnBujF7Op0F+ZTt5BlZ8m

kwZ10R028ODQ8sMMKMdIRplZS4ONrpEjuElPiO8F3i5MyWU8KS+0hrYHliWJ5ZYq

MeiPfsmczIeUjoQMolrAlT1vVkwPsp8T3NVMk1qF1Q3brOO/VY41Lsl15RnNt49M

Uf2bQLUb/vjF+9W7WxJZJY16FqthPQWSdTByWSaZcEj1iiCdjev7YzmS3ng87LoE

0ZsnC8+ECNmU8WzuPS7hS8VcUA==

-----END CERTIFICATE-----

-----BEGIN CERTIFICATE-----

MIIDpTCCAo2gAwIBAgIEb3ZpBjANBgkqhkiG9w0BAQUFADBwMQswCQYDVQQGEwJV

UzE4MDYGA1UEChMvTmF0aW9uYWwgQ2VudGVyIGZvciBTdXBlcmNvbXB1dGluZyBB

cHBsaWNhdGlvbnMxEzARBgNVBAMTCkppbSBCYXNuZXkxEjAQBgNVBAMTCTEyMDUz

NTg0ODAeFw0xMzA5MDkxNjAzMTdaFw0xMzA5MTAwNDA3NTlaMIGFMQswCQYDVQQG

EwJVUzE4MDYGA1UEChMvTmF0aW9uYWwgQ2VudGVyIGZvciBTdXBlcmNvbXB1dGlu

ZyBBcHBsaWNhdGlvbnMxEzARBgNVBAMTCkppbSBCYXNuZXkxEjAQBgNVBAMTCTEy

MDUzNTg0ODETMBEGA1UEAxMKMTg3MDAzMTExMDCCASIwDQYJKoZIhvcNAQEBBQAD

ggEPADCCAQoCggEBAKbbQNYjp+zIP1FEmiquSrfBvdKhpWnZpp7jUKvtLsYPM7nV

G3WeNABgKzGWdtEFc3p4kZB2jv0PsEPqeoFa0Moo3SInwmp56Kad77R1mTUH0rzb

2pTrtsHC/QZGvu/NTadl/ewuRssKyKg90WoJtYEkGWhmgITwfXKp9V9oZJlqHtuz

BvF8daOrbaUd/fej3YkM8SnN67/YcgYi5VlOrHLsnAiIQmQL1Vc+sdBzL1ZDjFYR

X399x3ZDSrMkmTySrwIAQwnhAOpPqSKwUico0mu06vRDrjbxZIXLE3fuck6SYVfe

KRxIwgYJyQNpDAbTBuDY8GS0emv9O6Y/Z69ql7kCAwEAAaMxMC8wDgYDVR0PAQH/

BAQDAgSwMB0GCCsGAQUFBwEOAQH/BA4wDDAKBggrBgEFBQcVATANBgkqhkiG9w0B

AQUFAAOCAQEAVWNdzh+t5H1fmz0kvSsFCA/pif7LGQXzoDyICF2WOT+mcI7wnKWH

B9TMquMh2fBs18EM3SAh/LTgQuSDPyWEAvk40GxwUYQTvjr9V+inAyxM/7zOWrh5

6Wa4y8L+TM+9xgyc8Tl1Bg3VDG6UEnrt1iCly2r/o7DDufY7bKKXjAhrAU2EY/As

VSoqddcg45/Z6Rk0AGqDM2dUmHyKoGiWi+IqyYFpM4Bo49BDuw6QoNr2CVbOLl/u

sYwB+tTZiPrk+V0UucPifx4i0FyZX0nM1JGY7pQ1ZLqU82Ser/pJ2gAZDGgBHF+F

dYMM1JhUFHqWO3p1+aDQjrGK8XkkspTMPA==

-----END CERTIFICATE-----

-----BEGIN CERTIFICATE-----

MIIDezCCAmOgAwIBAgIEBy87KDANBgkqhkiG9w0BAQUFADBcMQswCQYDVQQGEwJV

UzE4MDYGA1UEChMvTmF0aW9uYWwgQ2VudGVyIGZvciBTdXBlcmNvbXB1dGluZyBB

cHBsaWNhdGlvbnMxEzARBgNVBAMTCkppbSBCYXNuZXkwHhcNMTMwOTA5MTYwMzEx

WhcNMTMwOTEwMDQwNzU5WjBwMQswCQYDVQQGEwJVUzE4MDYGA1UEChMvTmF0aW9u

YWwgQ2VudGVyIGZvciBTdXBlcmNvbXB1dGluZyBBcHBsaWNhdGlvbnMxEzARBgNV

BAMTCkppbSBCYXNuZXkxEjAQBgNVBAMTCTEyMDUzNTg0ODCCASIwDQYJKoZIhvcN

AQEBBQADggEPADCCAQoCggEBALJgJe5KQXd65Or32r02+C7GmaqhYqSMsHPCVtcu

R6qrU1tuWxBjDf9z40LGFq6YRqIqbFyrISQlVoAlBzQhd9QQLvdBh4zerbf1vwXG

KOpNlIa1lBH6PYGiMPoRc8NUk/vhoPSQukHrUteb3gwnfz0aKe7JuhPCLlzOXeau

t8766GUwkwHx9Dbzx+NAVYDXrMF6ZUmtVLCkXfonHZYqdp49yMHLC8J99bbcJhso

wAkFuwWS0VeeXWNmOPZBL9n9cdsH8GeogaTqVDXlRzeLT0CmzE3NpXwTSYiyTl9Z

PL0GDs8drollKSvlmswduyHvAYGGgBSpQQQXThHVVLJpuWECAwEAAaMxMC8wDgYD

VR0PAQH/BAQDAgSwMB0GCCsGAQUFBwEOAQH/BA4wDDAKBggrBgEFBQcVATANBgkq

hkiG9w0BAQUFAAOCAQEAY5dP0WBLDtNeV6avCC6o7oPCxTyIHKa73eMMPeXd5pSd

ARnD7KpWc+34pWvthWp6iCNmuF1EoGulnTAlY5Lt/9f/of4+N9hNFNG3Brt23CFr

VA1pHhKJStHMW3Eu/bp+I6jKHTE5nuFeKue9ySCM71G+lBQjmOlHpGeStLgVIuzK

F/v20yroYpd/tPLlIMaugF+vZFzZ7oJQe9CogiDbEG6GyLXU8GjRO3S3hnwoV8LM

+l45bNI6hYFhs1FYs5YEdGzzL8gqi0k8TnpbKldi3oZ8iAWREDiE9gcrBZwHcVSK

DP49DC7fLiWSbg5rBMAHeaAJOHXAK3vPbmO1HviPAQ==

-----END CERTIFICATE-----

-----BEGIN CERTIFICATE-----

MIIEIzCCAwugAwIBAgIDJLKlMA0GCSqGSIb3DQEBBQUAMHsxCzAJBgNVBAYTAlVT

MTgwNgYDVQQKEy9OYXRpb25hbCBDZW50ZXIgZm9yIFN1cGVyY29tcHV0aW5nIEFw

cGxpY2F0aW9uczEgMB4GA1UECxMXQ2VydGlmaWNhdGUgQXV0aG9yaXRpZXMxEDAO

BgNVBAMTB015UHJveHkwHhcNMTMwOTA5MTYwMjU5WhcNMTMwOTEwMDQwNzU5WjBc

MQswCQYDVQQGEwJVUzE4MDYGA1UEChMvTmF0aW9uYWwgQ2VudGVyIGZvciBTdXBl

cmNvbXB1dGluZyBBcHBsaWNhdGlvbnMxEzARBgNVBAMTCkppbSBCYXNuZXkwggEi

MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCkARzrj/ZbinjdCJ3UhLzExsKB

/ezpVR0ibR0ebms4Nqw5nUlrPW65KVB/yYk4j860RG9FZVRaEzW8qehdhpHqlS/u

dlcgM3dE7raFbY5TS10xJUxR79rEuVMWLhpjqGStiksPwTSVoUxuGPWdOyG6woOA

wdnKzHYTupZnD8C9z9AE4J5jyrtGM4SiI/n9OZR4DuL9JEQkHhegbqWkjsVSg1Es

aqjIFW7kt1XmigRbaZt9P0+27NMZY0zIO+aJPf+q5mopOk6bu3835gb01rl/ab8H

bskfMzPF4ZlF1SYqWHtPXGD1PFNtNNDdxFSc/yjFZXA/IjeQRZpnkqJUSDRFAgMB

AAGjgc4wgcswDgYDVR0PAQH/BAQDAgSwMB0GA1UdDgQWBBTwBDOewbkKwLS4J4iJ

GaXe7FYaCjAfBgNVHSMEGDAWgBTX/KUCdjr2E/oroeDmUDXHI8d7UTAMBgNVHRMB

Af8EAjAAMDQGA1UdIAQtMCswDAYKKwYBBAGkPmQCBTAMBgoqhkiG90wFAgIDMA0G

CyqGSIb3TAUCAwIBMDUGA1UdHwQuMCwwKqAooCaGJGh0dHA6Ly9jYS5uY3NhLnVp

dWMuZWR1L2YyZTg5ZmUzLmNybDANBgkqhkiG9w0BAQUFAAOCAQEAa4AZfbmP568J

xHucCUWeJ7AXh5kKKcCAhqL9TWvHmQyKXfqqT7cOmcs9ln/eqNSK7/UEfJzz4+oM

2lVx7aDxa4ADZcdFzwDUXSnyb33/1OvKTinEVrc9PgIbp8thqcqmXY2TvdZiATla

pTjKK+/nGyArO1rFrvllI6mnQAO1RCALMQiOpyHp1m8As5p77TcI4Z/jcM/YJSiY

R61Ge6eiHSifroHWQRJyAOlNDTTTtfUqpwexi05atZve34wHA0ZTBlWnuVMegNnh

tlU8iqxdQke1KJ4o2ln6MZWaybqhV6EtM8YsNzrVKfYhfUA1VXyLnA7XQAM7lu7O

8ocVs/SNPw==

-----END CERTIFICATE-----