Digest Auth Proxy NodeJS

This NodeJS server demonstrates the Digest Authentication (Symmetric Key Signing) semantics for integration with IdentityX 4.2.

This provides an unauthenticated proxy to the IdentityX SRP REST interface. This server will handle both generating the request digest as well as verifying the response digest. The resource path, query string and body will be proxied to the IdentityX SRP.

The core files surrounding the digest authentication can be found in the model/util directory. The general structure of the code base is as follows:

```
app.js
--routes/all.js
----model/routingDao
------model/util/digestAuth
------model/util/classes/HTTPRequest
------model/util/classes/HTTPResponse
------model/util/classes/HTTPRequestResponse
------model/util/cryptoUtil
```

Dependencies

- Node 8.4.0 (npm 5.3.0)
- Python 2.7

Install instructions

Permanent Token

Dependencies:

- Java JRE 1.8.0_131
- OpenSSL v1.0.2L

Note: This assumes the use of a Java keystore for the initial public and private key generation, however this can be performed through alternative methods.

1) Generate a new public/private key pair in a Java keystore (this can be skipped if reusing an existing keystore & key pair).

```
keytool -genkeypair -alias identityxCert -keyalg RSA -keysize 2048 -dname
"cn=app,ou=Dev,ou=RP,c=US" -storepass secret --keystore IdentityX.jks
```

2) Convert the Java keystore into a PKCS12 keystore, and export the public and private keys into PEM files

```
keytool -importkeystore -srckeystore IdentityX.jks -destkeystore IdentityX.p12
-srcstoretype JKS -deststoretype PKCS12 -srcstorepass secret -deststorepass secret
-srcalias identityxCert -destalias identityxCert

openssl pkcs12 -in IdentityX.p12 -nokeys -passin pass:secret | openssl x509 -out
IdentityX.public.pem

openssl pkcs12 -in IdentityX.p12 -nodes -nocerts  -passin pass:secret | openssl rsa
-out IdentityX.private.pem
```

3) Place the private key file (IdentityX.private.pem) in the 'config' directory

3) Create the permanent token: In the Tenant Admin Console, navigate to Administration > Tokens - Select Create Permanent Token - In the "Subject ID" field enter a name that will differentiate this token - In the "Public Key" field copy and paste the entire contents of the public key file (IdentityX.public.pem) - In the "Key Type" field select "Symmetric" to use the digest scheme based on a shared secret - Click "Add Permission" - For full permissions: - In "Permission Selector", select the tenant name - In "Entity", select "All" - Select the "ALL(*)" checkbox - Click "Create"

Note: Further details can be seen in the "Tenant Admin Console User Guide - IdentityX Deployments", section 2.7

4) Your browser should automatically download a credential.properties file. Place this inside the 'config' directory

NodeJS

1) In the main directory, run `npm install` This will install the relevant packages.

2) Launch the server by running `npm start`

3) While the server is running (in a separate process), run `npm test`. This will run the automated tests to verify that your permanent token works.

Note: properties can be configured in the config/config.js file.

Usage

By default, this server is listening on port `3000`, and can be accessed at `http://127.0.0.1:3000`

You can make requests to this server as though you were sending them to the IdentityX SRP, except

note that only the resource path, query string and body will be proxied to the IdentityX SRP. HTTP Headers are not proxied.

Example 1 - List the first 5 active users, sorted by created date

```
GET
/kiwibank/IdentityXServices/rest/v1/users?status=ACTIVE&amp;limit=5&amp;sortField=c
reated
Host: localhost:3000
Content-Type: application/json
```

Example 2 - Create a new user

```
POST /kiwibank/IdentityXServices/rest/v1/users HTTP/1.1
Host: localhost:3000
Content-Type: application/json

{
    "userId": "user-id-here"
}
```

Troubleshooting

You can test connectivity to the IdentityX server by switching this over to Basic Auth (in the config/config.js file set `config.httpClient.authType = 'BASIC'`.