Admin API authentication function

Kazuhiro Ito

Overview

The authentication function of Node-RED supports "Username/password based authentication" and "OAuth/OpenID based authentication", but there is a problem that the Admin API cannot be used with "OAuth/OpenID based authentication".

1. Username/password based authentication
   Editor ⇒ Login possible
   Admin API ⇒ API executable

2. OAuth/OpenID based authentication
   Editor ⇒ Login possible
   Admin API ⇒ Unable to execute API (*Unable to get access token)

This document describes the details of the above problems and the countermeasures.

Username/password based authentication

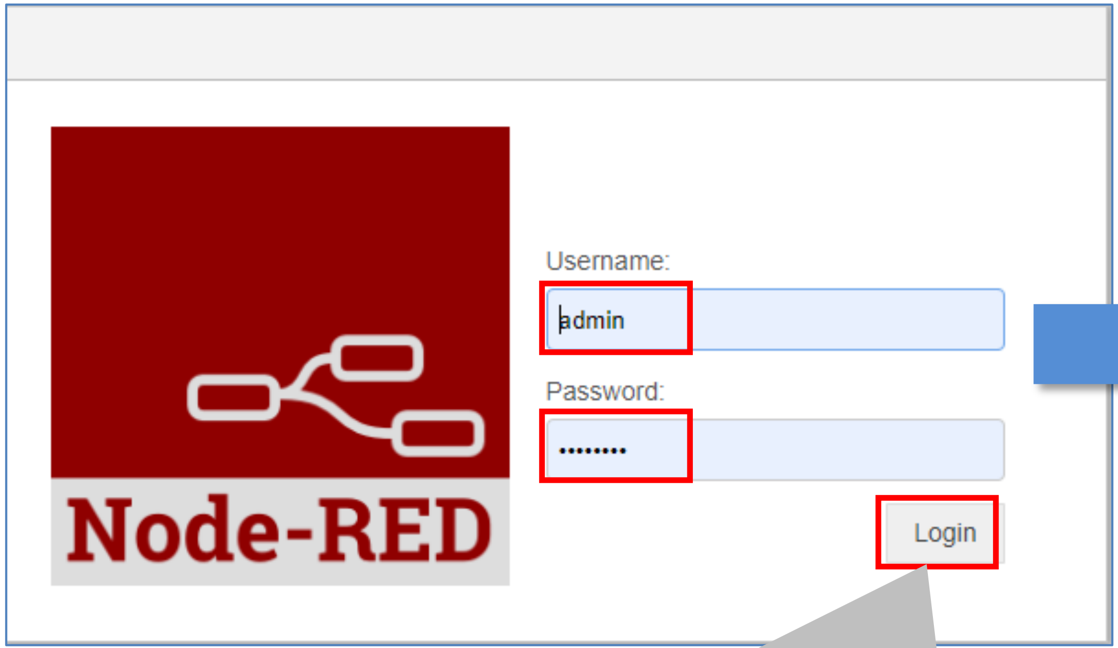
Settings when using Username/password based authentication.

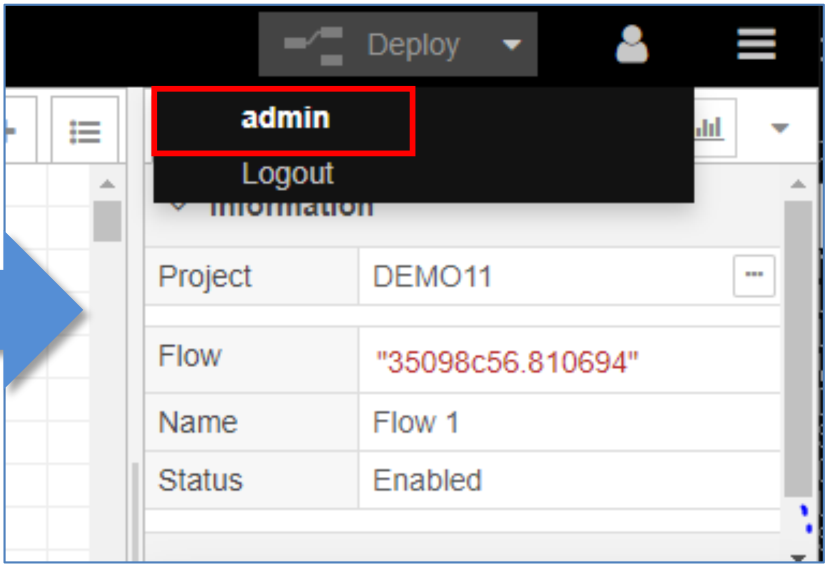
・**settings.js settings**

```
adminAuth: {
    type: "credentials",
    users: [{
        username: “<USER_NAME>",
        password: “<PASSWORD_HASH_VALUE>",
        permissions: "*"
    }]
},
```

Username/password based authentication

- Editor login screen





Project	DEMO11
Flow	"35098c56.810694"
Name	Flow 1
Status	Enabled

POST /auth/token
to authenticate with Node-RED server

Username/password based authentication

HITACHI
Inspire the Next

- **Execute Admin API**

To execute the Admin API, an access token must be specified.

Access token acquisition API:

POST /auth/token

Parameters:

client_id - identifies the client. Currently,
must be either node-red-admin or node-red-editor.
grant_type - must be password
scope - a space-separated list of permissions being requested.
Currently, must be either * or read.
username - the username to authenticate
password - the password to authenticate

Curl example:

```
   curl http://localhost:1880/auth/token --data 'client_id=node-red-
admin&grant_type=password&scope=*&username=admin&password=passwor
d’
```

Username/password based authentication

If successful, the response will contain the access token:

Example response:

```
{
  "access_token": "<ACCESS_TOKEN>",
  "expires_in":604800,
  "token_type": "Bearer"
}
```

In the API call, include the acquired access token in the Authorization header.

Curl example:

```
   curl -H "Authorization: Bearer <ACCESS_TOKEN>"
http://localhost:1880/settings
```

OAuth/OpenID based authentication

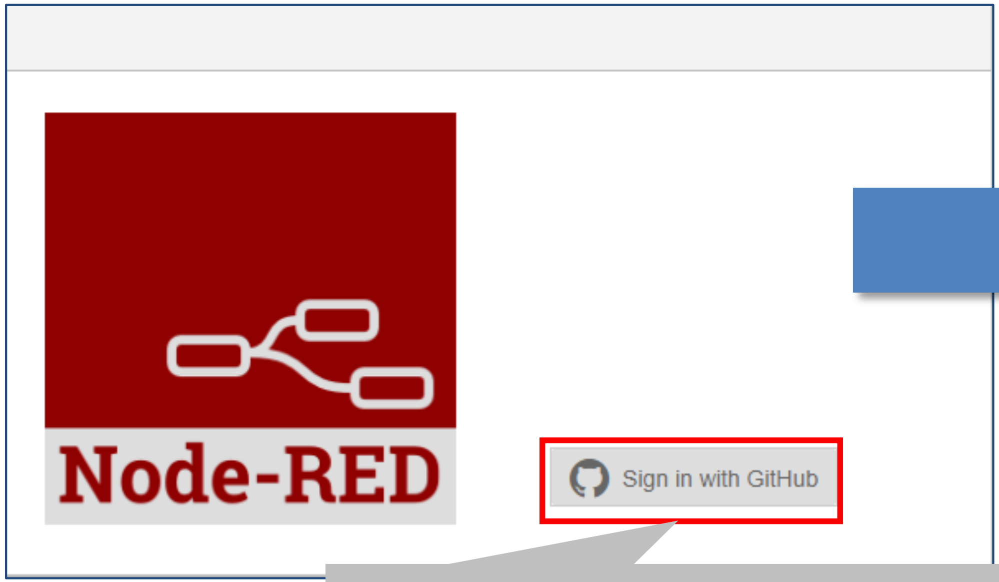
Settings when using OAuth/OpenID based authentication.

- **settings.js settings**

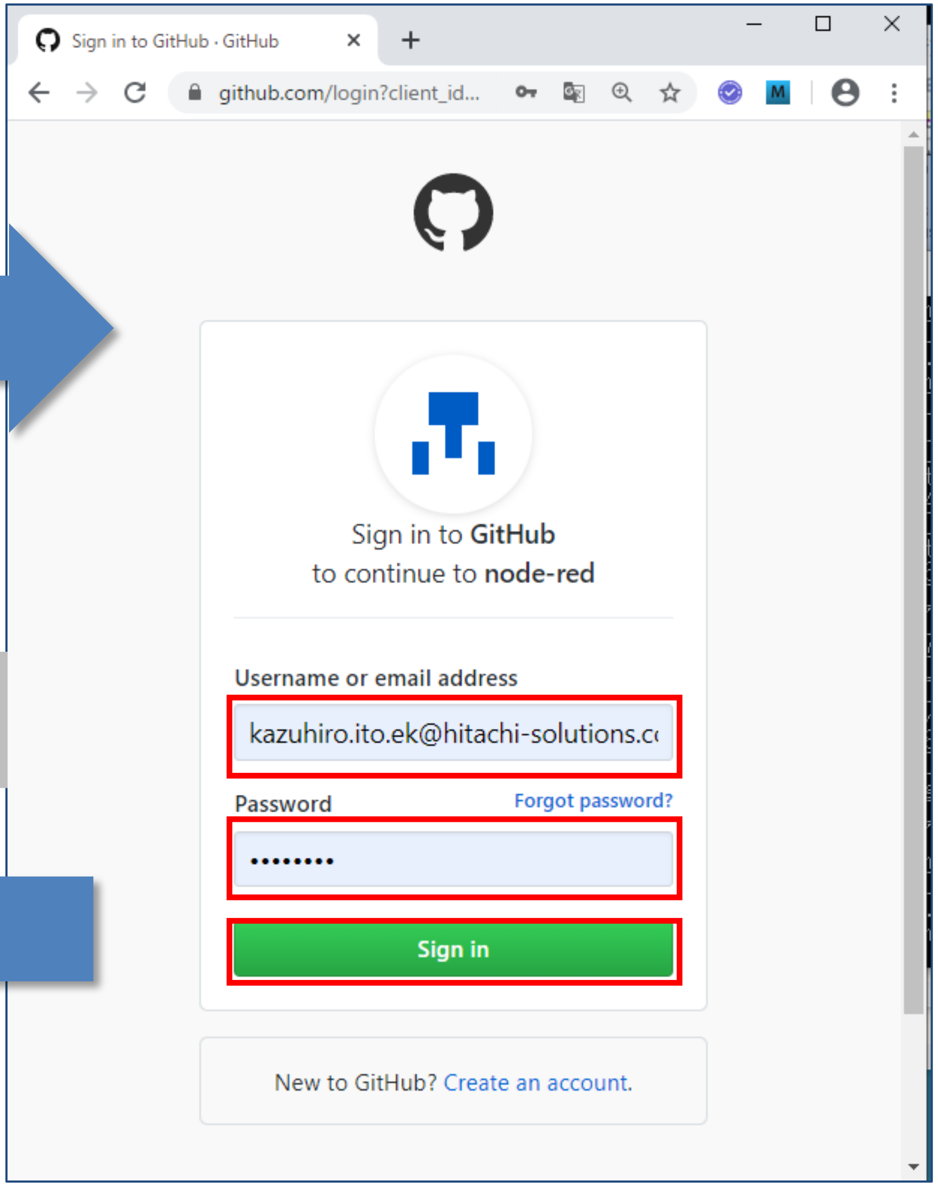
```
adminAuth: require('node-red-auth-github')({
    clientID: “<GITHUB_CLIENT_ID>",
    clientSecret: “<GITHUB_CLIENT_SECRET>",
    baseURL: "http://localhost:1880/",
    users: [
        {
          username: "<USER_NAME>",
          permissions: ["*"]
        }
    ]
}),
```

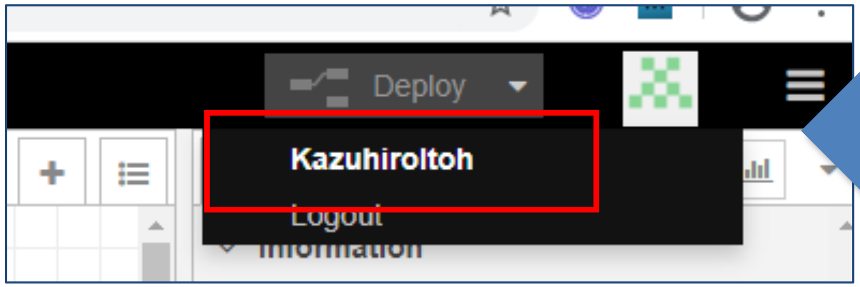
OAuth/OpenID based authentication

・Editor login screen



GET /auth/strategy
to redirect to authentication service


Sign in to GitHub · GitHub
github.com/login?client_id...
Sign in to GitHub
to continue to node-red
Username or email address
kazuhiro.ito.ek@hitachi-solutions.c
Password
Forgot password?
••••••••
Sign in
New to GitHub? Create an account.



OAuth/OpenID based authentication

・Execute Admin API

To execute the Admin API, you need to specify the access token, but there is no way to get the access token only with REST-API.

API for authentication:

GET /auth/strategy

Curl example:

```
#curl -v http://localhost:1880/auth/strategy
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 1880 (#0)
> GET /auth/strategy HTTP/1.1
> Host: localhost:1880
> User-Agent: curl/7.58.0
> Accept: */*
>
< HTTP/1.1 302 Found
< X-Powered-By: Express
< Access-Control-Allow-Origin: *
< Location:
https://github.com/login/oauth/authorize?response_type=code&redirect_uri=http%3A%2F%2Flocalhost%3A1880%2
Fauth%2Fstrategy%2Fcallback&client_id=xxxxxxxx
< Content-Length: 0
:
```

Countermeasure

As a countermeasure, I am considering adding the following two functions to Node-RED to get an access token for the Admin API.

Function 1.
　Add the function to enable Username/password based authentication by POST /auth/token.

Function 2.
　Make other authentication (custom Strategy) settings applicable in the form of plugins.

Countermeasure (Function 1)

Add settings to enable authentication by POST /auth/token for Admin API. Support authentication function for API without affecting editor login.

・**settings.js settings**

```
adminAuth: require('node-red-auth-github')({
    clientID: "<GITHUB_CLIENT_ID>",
    clientSecret: "<GITHUB_CLIENT_SECRET>",
    baseURL: "http://localhost:1880/",
    users: [
        {
          username: "<GITHUB_USER_NAME>",
          permissions: ["*"]
        },{
          username: “<ADMIN_API_USER_NAME>",
          password: "< PASSWORD_HASH_VALUE>",
          permissions: ["*"]
        }
    ],
    apiAuth: {
        credentials: true
    }
}),
```

Countermeasure (Function 1)

Curl example:

```
$ curl http://localhost:1880/auth/login
{"type":"strategy","prompts":[{"type":"button","label":"Sign in with
GitHub","url":"auth/strategy","icon":"fa-github"}]}

$ curl http://localhost:1880/settings
Unauthorized

$ curl http://localhost:1880/auth/token --data 'client_id=node-red-
admin&grant_type=password&username=admin&password=password'
{"access_token":"BwhDDrF7QZFJym6sHOLh2m5QlmBCG6J7LnAzIYvszcc03F3NbrK1bXTeu
CdrykeYK1pge803X2tASCR6v7uFxwwGv2Q8vMChazdwmDdY6AfT6vtiYID8bCbi2eLZ5Fy0",
"expires_in":604800,"token_type":"Bearer"}

$ curl -H "Authorization: Bearer BwhDDrF7QZFJym6..." http://localhost:1880/settings
{"httpNodeRoot":"/","version":"1.0.2","user":{"username":"admin","permissions":["*"]},"cont
ext":{"default":"memory","stores":["memory"]},"project":{"path":"/home/it…
```

Countermeasure (Function 1)

- **@node-red/editor-api/lib/index.js**

```
diff -r -C 6 org/@node-red/editor-api/lib/index.js new/@node-red/editor-api/lib/index.js
*** org/@node-red/editor-api/lib/index.js    2020-01-15 10:12:10.428143100 +0900
--- new/@node-red/editor-api/lib/index.js   2020-01-15 10:28:21.664092400 +0900
***************
*** 66,77 ****
--- 66,87 ----
          adminApp.use(bodyParser.urlencoded({limit:maxApiRequestSize,extended:true}));

          adminApp.get("/auth/login",auth.login,apiUtil.errorHandler);
          if (settings.adminAuth) {
              if (settings.adminAuth.type === "strategy") {
                  auth.genericStrategy(adminApp,settings.adminAuth.strategy);
+                 // Credentials authentication for REST-API
+                 if (settings.adminAuth.apiAuth && settings.adminAuth.apiAuth.credentials) {
+                     adminApp.use(passport.initialize());
+                     adminApp.post("/auth/token",
+                         auth.ensureClientSecret,
+                         auth.authenticateClient,
+                         auth.getToken,
+                         auth.errorHandler
+                     );
+                 }
              } else if (settings.adminAuth.type === "credentials") {
                  adminApp.use(passport.initialize());
                  adminApp.post("/auth/token",
                      auth.ensureClientSecret,
                      auth.authenticateClient,
                      auth.getToken,
```

Countermeasure (Function 2)

Make other authentication (custom Strategy) settings applicable in the form of plugins. This is the method used in Hitachi's environment.*1

This feature also works for "Username/password based authentication".

* 1 This requires patching the Node-RED source, so we want to make it a plugin.

- **settings.js settings**

```
adminAuth: require('node-red-auth-github')({
   clientID: "<GITHUB_CLIENT_ID>",
   clientSecret: "<GITHUB_CLIENT_SECRET>",
   baseURL: "http://localhost:1880/",
   users: [
      {
        username: "<USER_NAME>",
        permissions: ["*"]
      }
   ],
   apiAuth: {
      strategy: require('node-red-api-auth-test') ({
         tokenHeader: 'x-nodered-token',
         failedOnMissing: true
      }),
   }
}),
```

By specifying an access token in the header x-nodered-token, it is possible to apply a plug-in that permits execution of the API.

Countermeasure (Function 2)

Curl example:

```
$ curl http://localhost:1880/settings
Unauthorized

$ curl –H ‘x-nodered-token:<UNIQUE_TOKEN>’ http://localhost:1880/settings
{"httpNodeRoot":"/","version":"1.0.2","user":{"username":"admin","permissions":[
"*"]},"context":{"default":"memory","stores":["memory"]},"project":{"path":"/hom
e/it…
```

Countermeasure (Function 2)

- **@node-red/editor-api/lib/auth/index.js**

```
diff -r -C 6 org/@node-red/editor-api/lib/auth/index.js new/@node-red/editor-api/lib/auth/index.js
*** org/@node-red/editor-api/lib/auth/index.js          2020-01-15 10:13:07.450582300 +0900
--- new/@node-red/editor-api/lib/auth/index.js          2020-01-15 10:25:09.667887200 +0900
***************
*** 44,55 ****
--- 44,61 ----
  function init(_settings,storage) {
      settings = _settings;
      if (settings.adminAuth) {
          var mergedAdminAuth = Object.assign({}, settings.adminAuth, settings.adminAuth.module);
          Users.init(mergedAdminAuth);
          Tokens.init(mergedAdminAuth,storage);
+         // Strategy authentication for REST-API
+         if (settings.adminAuth.apiAuth && settings.adminAuth.apiAuth.strategy) {
+             var strategy = settings.adminAuth.apiAuth.strategy;
+             var options = strategy.options;
+             passport.use('api', new strategy.strategy(options, options.verify));
+         }
      }
  }
  /**
   * Returns an Express middleware function that ensures the user making a request
   * has the necessary permission.
   *
```

Countermeasure (Function 2)

- @node-red/editor-api/lib/auth/index.js

```
***************
*** 57,69 ****
   * @return {Function} - an Express middleware
   * @memberof @node-red/editor-api_auth
   */
  function needsPermission(permission) {
      return function(req,res,next) {
          if (settings && settings.adminAuth) {
!             return passport.authenticate(['bearer','anon'],{ session: false })(req,res,function() {
                  if (!req.user) {
                      return next();
                  }
                  if (permissions.hasPermission(req.authInfo.scope,permission)) {
                      return next();
                  }
```

Countermeasure (Function 2)

- **@node-red/editor-api/lib/auth/index.js**

```
--- 63,80 ----
   * @return {Function} - an Express middleware
   * @memberof @node-red/editor-api_auth
   */
  function needsPermission(permission) {
      return function(req,res,next) {
          if (settings && settings.adminAuth) {
!             var auth_seqs = ['bearer','anon'];
!             if (settings.adminAuth.apiAuth && settings.adminAuth.apiAuth.strategy) {
!                 // Strategy authentication for REST-API
!                 auth_seqs.push('api');
!             }
!             return passport.authenticate(auth_seqs,{ session: false })(req,res,function() {
                  if (!req.user) {
                      return next();
                  }
                  if (permissions.hasPermission(req.authInfo.sc        mission)) {
                      return next();
                  }
```

"bearer": Login access token
"anon": Default user
and
"api": For API execution(Support for additional checks.)

Countermeasure (Function 2)

- **Reference: Plug-in code example**(‘node-red-api-auth-test’)

```
var UniqueTokenStrategy = require("passport-unique-token");

var requiredOptions = [
    'tokenHeader',
    'failedOnMissing'
];

module.exports = function(opts) {
  for (var i=0;i<requiredOptions.length;i++) {
    if (!opts.hasOwnProperty(requiredOptions[i])) {
      throw new Error("Missing auth option: "+requiredOptions[i]);
    }
  }
  var apiAuth = {
    strategy: UniqueTokenStrategy.Strategy,
    options: {
        tokenHeader: opts.tokenHeader,
        failedOnMissing: opts.failedOnMissing,
        verify: function (token, done) {
          var adminToken = process.env.NODERED_ADMIN_TOKEN;
          if (adminToken && token === adminToken) {
            var retUserObj = {
              username: process.env.NODERED_ADMIN_USERNAME || 'admin',
              permissions: [ '*' ],
            };
            return done(null, retUserObj, {scope: ['*']});
          } else {
            return done(null, false);
          }
        }
    }
  }
  return apiAuth;
}
```

Appendix

Reference URL

・Securing Node-RED
https://nodered.org/docs/user-guide/runtime/securing-node-red

・Admin API/Authentication
https://nodered.org/docs/api/admin/oauth

・Node-RED Authentication with GitHub
https://github.com/node-red/node-red-auth-github

・Passport Unique Token Strategy
https://www.npmjs.com/package/passport-unique-token

HITACHI

Inspire the Next