NetBrain® Integrated Edition 8.0

Embedded Map Development Guide (Portal User)

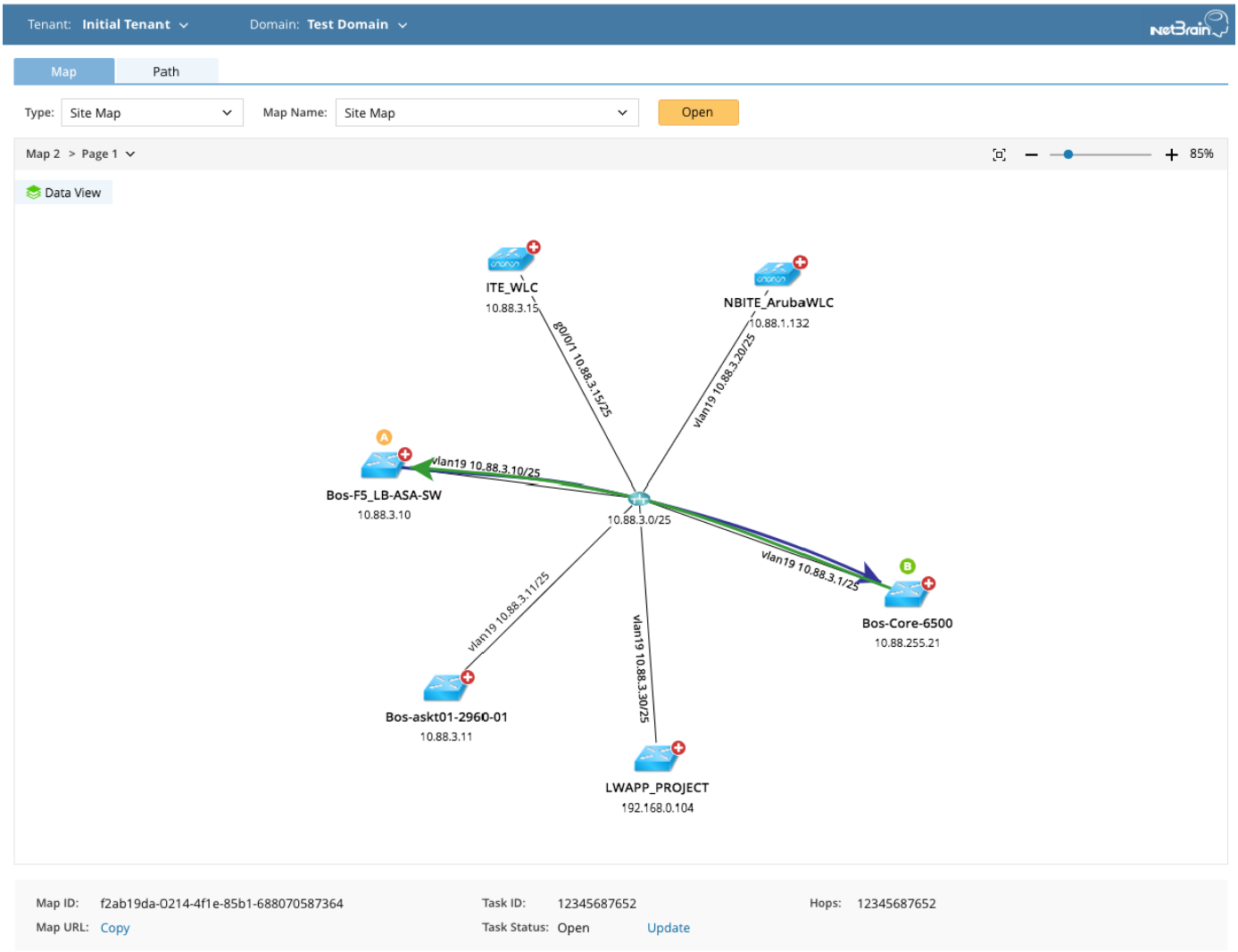
Version 8.01 | Last Updated 2019-11-08

Contents

1. Introduction

NetBrain Embedded Map feature enables you to embed the NetBrain maps within a third-party portal page using an iframe. An embedded map provides a limited view of a regular NetBrain map, which only shows the map itself along with the detailed information within the data view.

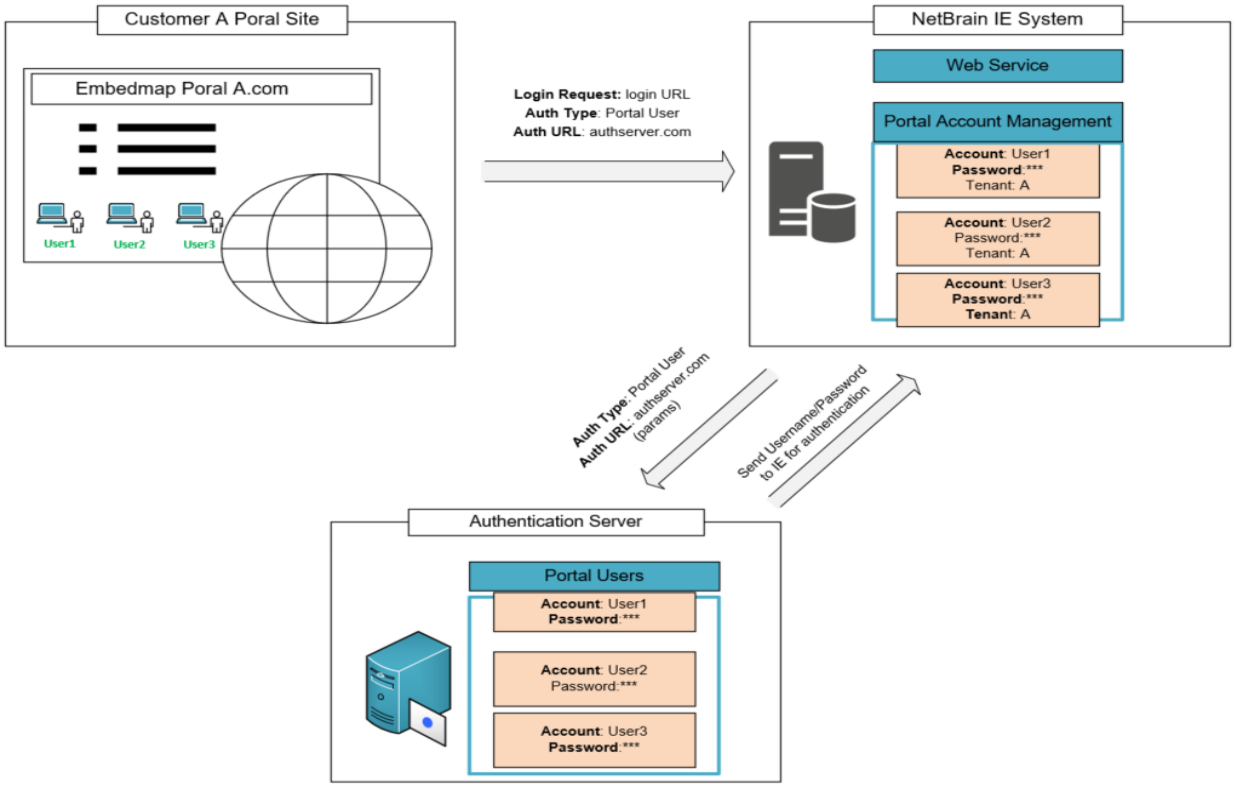
Figure: A site map embedded in a portal page



This guide introduces how to deploy NetBrain embedded-map function on one of your existing portal sites.

2. Embedded-Map Architecture

The architecture diagram below describes a standard embedded-map deployment mode. In this mode, one embedded-map portal is developed with direct requests to the NetBrain IE system.



Three main components are involved in this deployment mode:

Component	Description
Customer Portal (Embedded-Map Portal)	The customer portal is where you will embed NetBrain maps. The embedded-map login is launched at the customer portal with a login URL, authentication type and authentication URL included.
NetBrain IE System	NetBrain system provides web services, APIs and embedded-map accounts to the embedded-map function deployed in a third-party portal.
Authentication Server	Authentication Server receives the authentication request from the NetBrain system and responds to the authentication request with an encrypted username/password. See Creating an authentication server for more details.

To get more deployment modes, see Advanced Deployment Modes for details.

3. Deployment Requirements

The embedded-map feature allows a highly customized development on a web portal based on APIs. And this characteristic of this feature demands that deploying this feature needs the involvement of technical personnel with specific skills.

The table below describes the requirements for technical personnel about deploying an authentication server and embedded-map web portal.

Component	Requirement
Authentication Server	A third-party authentication application is required when you deploy the embedded-map feature. This authentication application should be created via rest APIs, and the involved technical personnel needs to know how to create rest APIs via any of the backend languages, such as Java, NodeJs, Python, and C#. See Creating an authentication server.
Embedded-map Portal	If you want to develop a customized page without using the built-in embedded-map HTML files, the development requires the involvement of web developers. The following capabilities are expected: ▪ Web development ▪ Knowledge of HTML/JavaScript/CSS ▪ Knowledge of asynchronous calls and AJAX calls

4. Creating an Authentication Application

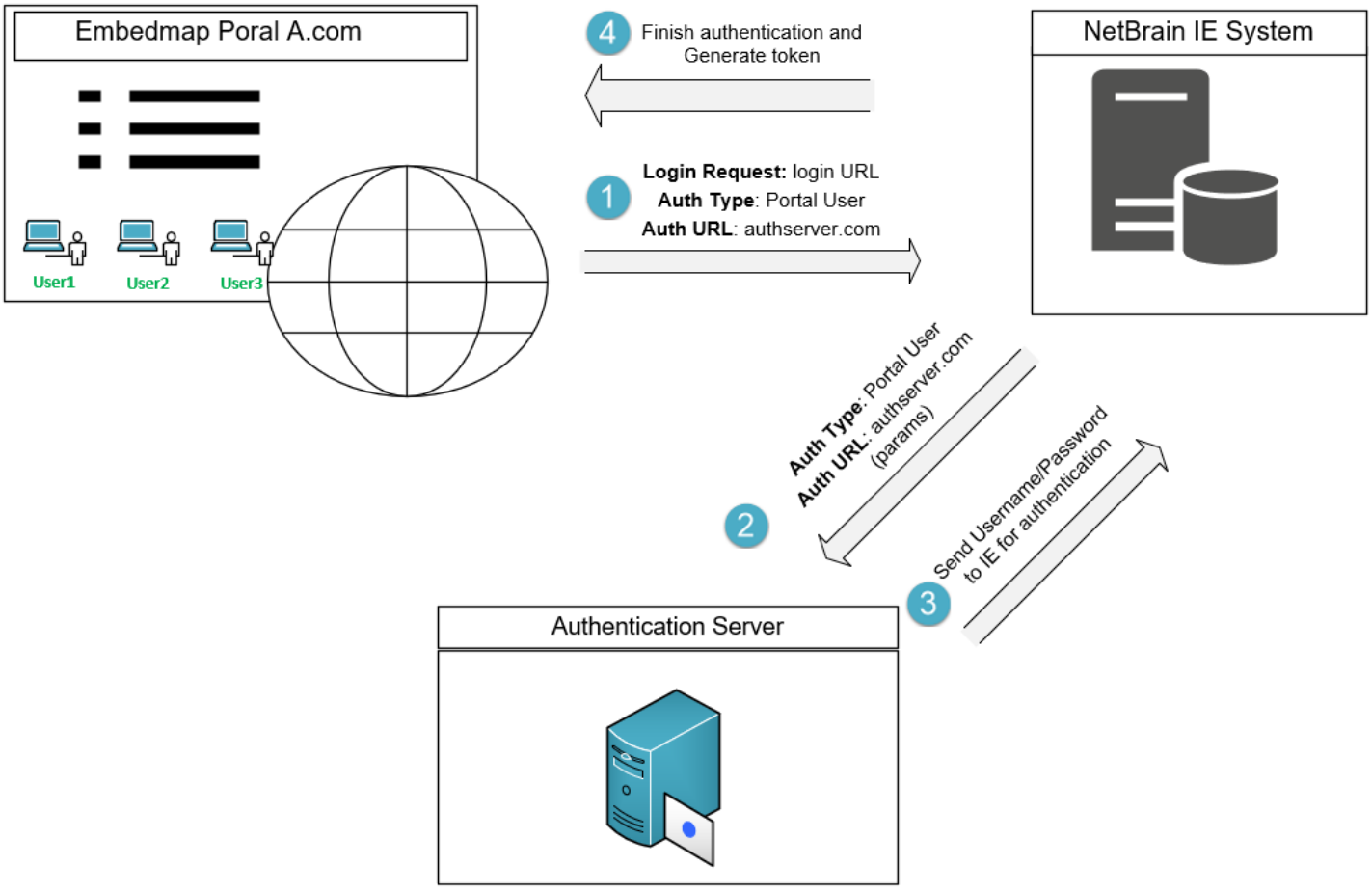
When deploying the embedded-map feature, you need to set up an authentication server to receive and handle the authentication request from the embedded-map portal.

> **Note**: If you deploy the authentication server on the same server with your Netbrain server, make sure that the port of the authentication server is a spare one compared with your NetBrain server ports.

4.1. Authentication Process

When you launch an embedded-map portal, the login authentication starts. The steps below give an overall introduction to this authentication process.

Figure: Authentication flow



1. When you enter the embed-map portal in your browser address bar, the embedded-map portal will send a login request that contains an authentication URL to the NetBrain server.

2. NetBrain system checks the authentication URL and sends the authentication request to the authentication server.
3. The authentication server receives the request and sends a local portal-user username/password back to the NetBrain system.
4. NetBrain system performs authentication based on the credentials received from the authentication server (it will compare the credential stored in the database with that received from the authentication server). Generate the token for the embedded map to use if the credentials are correct.

4.2. Creating an Authentication API

An authentication API needs to be created to receive embedded-map authentication requests. You can create the authentication API based on your business rules. However, the following specifications about the authentication API should be taken into considerations:

No.	Requirement
1	The authentication application must be generated via a rest API.
2	Store the portal usernames/passwords created in NetBrain IE in a place where the API can access or directly hard-code the usernames/passwords in the API script.
3	The API needs to implement RSA encryption to passwords/usernames with `publicKeyExponent` and `publicKeyModulus` public key information received from the body of an authentication request.
4	The API needs to call the GenerateAuthenticationCode API in its script (make sure the connectivity between the authentication server and your NetBrain webserver). The GenerateAuthenticationCode API is provided by NetBrain and used to generate a token for the embedded map after the authentication is passed.

The following is a sample authentication API created with the C# programming language.

```
public class AuthReq
    {
        [JsonProperty("publicKeyExponent", NullValueHandling = NullValueHandling.Ignore)]
        [Required]
        public string PublicKeyExponent { get; set; }

        [Required]
```

```
        [JsonProperty("publicKeyModulus", NullValueHandling = NullValueHandling.Ignore)]
        public string PublicKeyModulus { get; set; }
    }

    public class GenerateAuthenticationCodeRequest
    {
        [JsonProperty("username", NullValueHandling = NullValueHandling.Ignore)]
        public string Username { get; set; }

        [JsonProperty("password", NullValueHandling = NullValueHandling.Ignore)]
        public string Password { get; set; }
    }

    public class GenerateAuthenticationCodeResponse
    {
        [JsonProperty("authenticationCode")]
        public string AuthenticationCode { get; set; }

        [JsonProperty("error")]
        public string Error { get; set; }
    }

    [RoutePrefix("IDP/Auth")]
    public class AuthController : ApiController
    {
        [HttpPost]
        [Route("GenerateAuthenticationCode")]
        [AllowAnonymous]
        public async Task<GenerateAuthenticationCodeResponse>
GenerateAuthenticationCode([FromBody]AuthReq req)
        {
            var res = new GenerateAuthenticationCodeResponse() { AuthenticationCode =
string.Empty, Error = string.Empty };
            try
            {
                // The url http://{ip or domain}/ServicesAPI/EmbedMap/GenerateAuthenticationCode
                var generateAuthorizationCodeUrl = System.Configuration.ConfigurationManager
                    .AppSettings["GenerateAuthenticationCodeUrl"].ToString();
                var postData = BuildPostData(req);
                HttpClientHandler handler = new HttpClientHandler
                {
                    AutomaticDecompression = DecompressionMethods.GZip |
DecompressionMethods.Deflate
                };
                using (var httpClient = new HttpClient(handler))
                {
                    HttpContent content = new StringContent(postData, Encoding.UTF8);
                    content.Headers.ContentType = new MediaTypeHeaderValue("application/json");
                    httpClient.Timeout = TimeSpan.FromSeconds(60);
                    HttpResponseMessage response = await
httpClient.PostAsync(generateAuthorizationCodeUrl, content);

                    response.EnsureSuccessStatusCode();
                    string responseBody = await response.Content.ReadAsStringAsync();
                    var resObj = JObject.Parse(responseBody);
                    if (resObj["operationResult"].Value<int>("ResultCode") > 0)
                    {
                        res.Error = resObj["operationResult"]["ResultDesc"].ToString();
                    }
                    else
                    {
                        res.AuthenticationCode = resObj["data"]["authenticationCode"].ToString();
                    }
                    return res;
                }
            }
```

```
            catch (Exception e)
            {
                res.Error = e.Message;
                return res;
            }
        }

        public string BuildPostData(AuthReq req)
        {
            // just sample code, you can get the user from DB/XML/JSON
            var user = new GenerateAuthenticationCodeRequest
            {
                Username = "portal username"   // username of a portal account created in NetBrain
                Password = "portal password"   // password of a portal account

            //RSA encrypt
            if (!string.IsNullOrEmpty(req.PublicKeyExponent) &&
!string.IsNullOrEmpty(req.PublicKeyModulus))
            {
                user.Password = EncryptPassword(user.Password, req.PublicKeyExponent,
req.PublicKeyModulus);
            }

            return Newtonsoft.Json.JsonConvert.SerializeObject(user,
Newtonsoft.Json.Formatting.Indented);
        }

      public string EncryptPassword(string password, string publicKeyExponent, string
publicKeyModulus)
        {
            if (string.IsNullOrEmpty(password))
                return string.Empty;

            RSAParameters rsaParameters = new RSAParameters()
            {
                Exponent = HexStringToBytes(publicKeyExponent),
                Modulus = HexStringToBytes(publicKeyModulus),
            };
            RSACryptoServiceProvider rsa = new RSACryptoServiceProvider();
            rsa.ImportParameters(rsaParameters);
            byte[] bs = rsa.Encrypt(Encoding.ASCII.GetBytes(password), false);
            return BytesToHexString(bs);
        }

        public byte[] HexStringToBytes(string hex)
        {
            if (hex.Length == 0)
            {
                return new byte[] { 0 };
            }

            if (hex.Length % 2 == 1)
            {
                hex = "0" + hex;
            }

            byte[] result = new byte[hex.Length / 2];

            for (int i = 0; i < hex.Length / 2; i++)
            {
                result[i] = byte.Parse(hex.Substring(2 * i, 2),
System.Globalization.NumberStyles.AllowHexSpecifier);
            }

            return result;
```

```
        }

        public string BytesToHexString(byte[] bs)
        {
            var list = new List<string>();
            foreach (var b in bs)
            {
                list.Add(b.ToString("x2"));
            }

            return string.Join("", list);
        }
}
```

5. Developing a Standard Embedded-Map Portal

You can build a standard embedded-map portal based on your needs by selecting either of the following ways:

- Develop a portal with built-in portal templates. Through this method, you can build a portal quickly by modifying several parameters.
- Build a customized portal. NetBrain provides essential elements, such as APIs and function libraries, and you can highly customize an embedded-map portal using these essential elements.

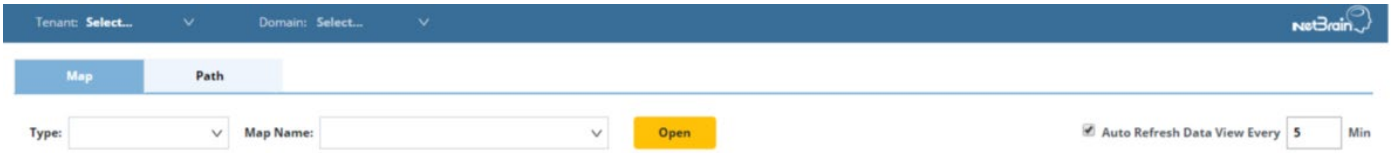
5.1. Developing a Portal with Built-in Templates

NetBrain provides built-in portal templates and you can use the templates to quickly create an embedded-map portal by modifying a few parameters.

The portal created via built-in templates contains two ways to open a map and a function about data view.

- **Open a site/device group/public map**
- **Open a map by calculating a path**
- **Automatically refresh data view applied on an embedded map**

Figure: A sample portal created with built-in templates.

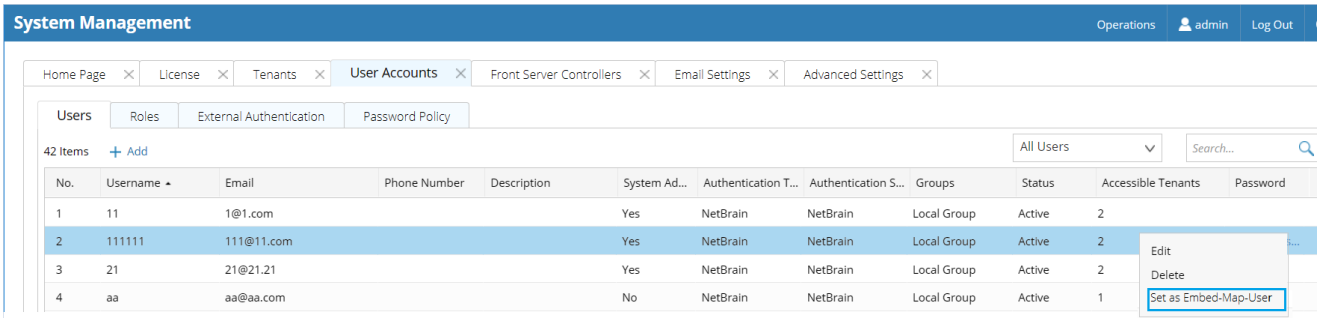


The pseudo-code below describes a high-level workflow to build a portal to embed NetBrain maps with a built-in portal template.

```
1. Create Embed-Map-User accounts in your NetBrain system.

2. Set up an authentication server.

3. Modify web.config file to enable Iframe.

4. Build a portal with the built-in port template.

5. Create a site for the portal.
```

1. Create Embed-Map-User accounts in your NetBrain system. In the System Management page, create and set accounts dedicated to the embedded-map login authentication.

   Figure: Set an account as Embed-Map-User.



> **Important Notes:**
>
> - An embedded-map account cannot be used to log into NetBrain IE.
> - An embedded-map account can be assigned with different accessible tenants/domains and roles and privileges. Before creating the accounts, it is better to plan them based on your scenarios.
> - All users logging to the embedded-map portal will have the same access privileges inherited from the configured portal users.

2. Set up an authentication server. See Creating an Authentication Server for details.

3. Modify the **web.config** file of your NetBrain Web Server to enable iframe. An iframe is used to embed another document within the current HTML page.

   1) Locate the **web.config** file under ***../Program Files\NetBrain\Web Server\nb_publish_client/web.config***.

   2) Edit the file, append the **Content-Security-Policy configuration** to the bottom of the <httpProtocol> <customHeaders> section, and replace the value highlighted below with the actual address of the portal that will embed your NetBrain maps.

   Sample Web.config File:

```
<httpProtocol>
  <customHeaders>
    <remove name="X-Frame-Options"/>
    <remove name="X-Content-Type-Options"/>
    <remove name="X-XSS-Protection"/>
    <remove name="Strict-Transport-Security"/>

    <add name="X-Frame-Options" value="SAMEORIGIN"/>
    <add name="X-Content-Type-Options" value="nosniff"/>
    <add name="X-XSS-Protection" value="1; mode=block"/>
    <add name="Strict-Transport-Security" value="max-age=31536000; includeSubDomains;
```

```
preload"/>
    <add name="Content-Security-Policy" value="frame-ancestors http://10.10.10.1:801"/>
  </customHeaders>
```

> **Tip:** To learn more about the HTTP Content-Security-Policy response header, click here.

3) Restart the Windows WWW service.

4. Modify the built-in port templates to create an embedded-map portal.

1) Copy the built-in embedded-map template folder to the server where you will build the embedded-map portal. On the server where your NetBrain system is installed, locate the built-in template folder named **embedmapdemo** under the directory ***<../Program Files\NetBrain\Web Server\nb_publish_client >***, and copy the folder to the server where your portal will be built.

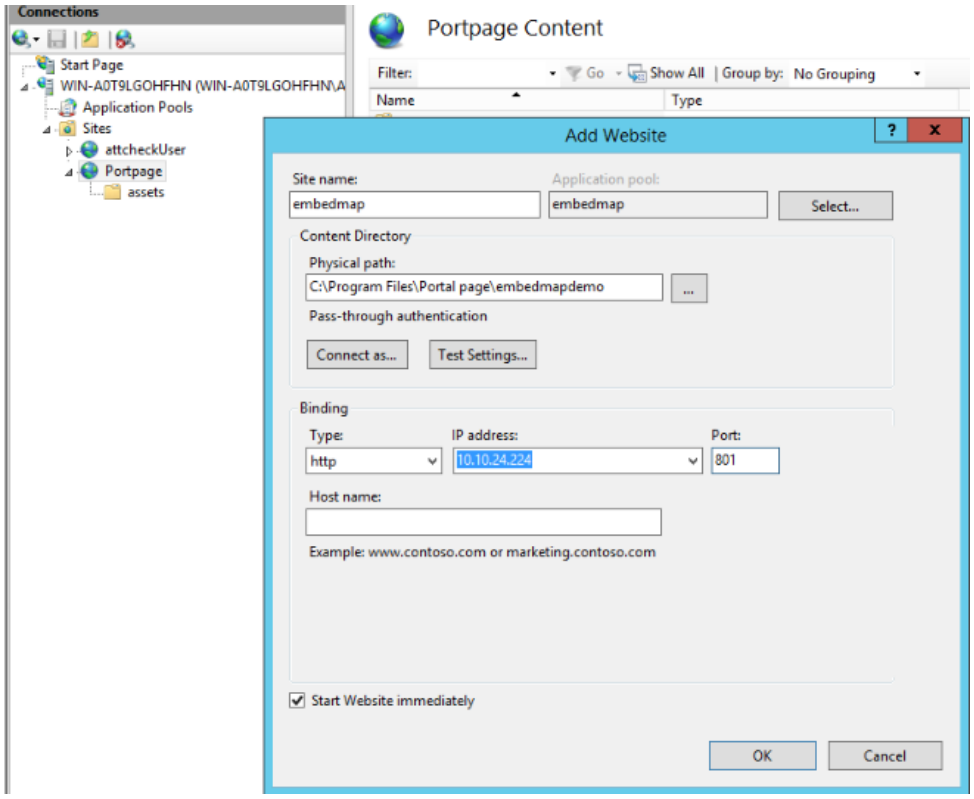
2) Edit any demo file in the **embedmapdemo** folder (for example, **demo2.html**) and update the parameters highlighted below with the actual values:

```
<script>

    // init the param for netbrainTools, virtualDir is option, virtualDir: '/' or
virtualDir: '/NG', and default is '/' if no this field.

    // 'https://www.netbrain-az.com'

    netbrainInstanceOption = { netbrainUrl: 'http://10.10.3.18', virtualDir: '/',
authentication: { type: 'portalUser', authUrl:
'http://10.10.3.51:89/IDP/Auth/GenerateAuthenticationCode' },
```

Parameter	Description
netbrainUrl	The login address of your NetBrain IE.
type	The authentication type for embedded map login. There are two types: **sso** and **portalUser**. In this case, fill in this filed with **portalUser**.
authUrl	The address of the authentication server that you have created for the authentication of embedded-map user accounts.

5. Create a site for the modified embedded-map file and bind the site with the IP of the portal server.

   Example: Create a portal site with IIS.



6. Enter **http(s)://<binding address:Port>/<specific demo filename>.html** in your web address bar. For example, http://10.10.24.224:801/demo2.html.

7. Select a Tenant/Domain and open a map.

5.2. Developing a Customized Portal

Combining with the APIs and functions provided by NetBrain, you can build an embedded-map portal with high customization.

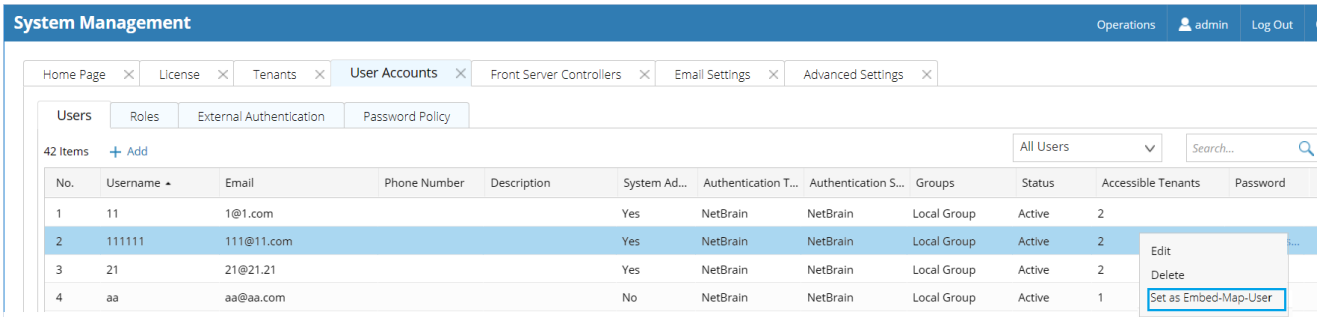
The pseudo-code below describes a high-level workflow to deploy a customized embedded-map portal.

```
1. Set up Embed-Map-User accounts in your NetBrain system.

2. Create an authentication server.

3. Modify the web.config file to enable Iframe.

4. Create an HTML file for the Embedded-Map portal.

5. Reference NetBrain script library in the portal HTML file.
```

```
6. Configure the basic parameters of embedded-map login in the portal HTML file.

7. Initialize NetBrain instance in the portal HTML file.

8. Construct the drop-down menu for Tenant selection in the portal HTML file.

9. Construct the drop-down menu for Domain selection in the portal HTML file.

10. Initialize map instances in the portal HTML file.

11. Open Site/Device Group/Public Map in the portal HTML file.

12. Open a path map in the portal HTML file.

13. Open a map created via Qapp in the portal HTML file.

14. Add the Data View Auto Refresh function in the portal HTML file.
```

1. Create Embed-Map-User accounts in your NetBrain system. In the System Management page, create and set accounts dedicated to the Embed-map login authentication.

   Figure: Set an account as Embed-Map-User.



> **Important Notes:**
>
> - An embedded-map account cannot be used to log into NetBrain IE.
> - An embedded-map account can be assigned with different accessible tenants/domains and roles and privileges. Before creating the accounts, it is better to plan them based on your scenarios.
> - All users logging to the embedded-map portal will have the same access privileges inherited from the configured portal users.

2. Set up an authentication server. See Creating an Authentication Server for details.

3. Modify the **web.config** file of your NetBrain Web Server to enable iframe. An iframe is used to embed another document within the current HTML page.

   1) Locate the **web.config** file under ***../Program Files\NetBrain\Web Server\nb_publish_client/web.config***.

2) Edit the file, append the **Content-Security-Policy configuration** to the bottom of the <httpProtocol> <customHeaders> section, and replace the value highlighted below with the actual address of the portal that will embed your NetBrain maps.

Sample Web.config File:

```
<httpProtocol>
  <customHeaders>
    <remove name="X-Frame-Options"/>
    <remove name="X-Content-Type-Options"/>
    <remove name="X-XSS-Protection"/>
    <remove name="Strict-Transport-Security"/>

    <add name="X-Frame-Options" value="SAMEORIGIN"/>
    <add name="X-Content-Type-Options" value="nosniff"/>
    <add name="X-XSS-Protection" value="1; mode=block"/>
    <add name="Strict-Transport-Security" value="max-age=31536000; includeSubDomains;
preload"/>
    <add name="Content-Security-Policy" value="frame-ancestors http://10.10.10.1:801"/>
  </customHeaders>
```

> **Tip:** To learn more about the HTTP Content-Security-Policy response header, click here.

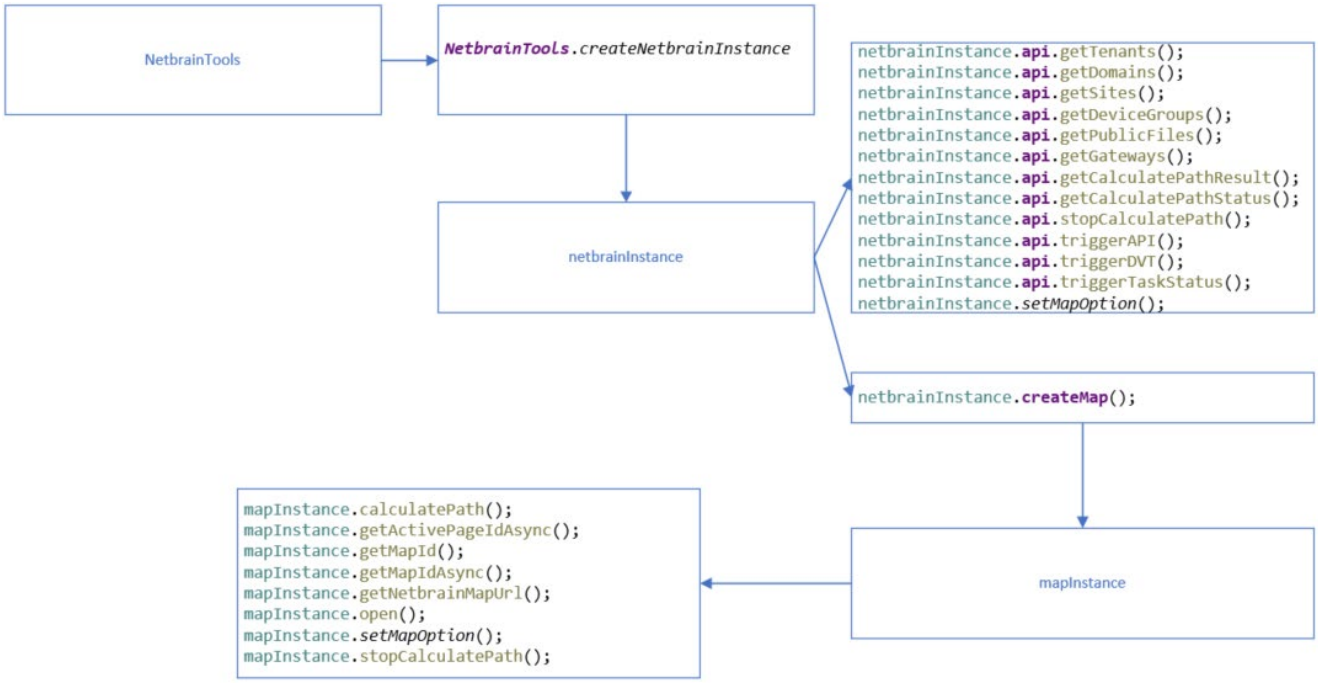
3) Restart the Windows WWW service.

4. Create an HTML file for the embedded-map portal.

5. Reference the following script in the HTML file. The script is to call the script library (named after `NetbrainTools`) of your NetBrain web server, which provides an interface to NetBrain APIs encapsulated via JS functions.

```
<html>
<head>
    <script src="http(s)://<your NetBrain Web Server
address>/js/nb.embedmap.netbraintools.min.js"></script>
</head>
<body>
</body>
</html>
```

Figure: APIs in NetbrainTools Structure



6. Configure the basic paramters of embedded-map login.

```
 var netbrainOptions = {

    netbrainUrl: "http://10.10.3.18",  //required, the address of you NetBrain IE.

    authentication: {

        type: 'portalUser',           // required

        authUrl: "http://10.10.3.51:89/IDP/Auth/GenerateAuthenticationCode" // required. The
URL of the authentication server.

    },

    virtualDir: '/',     //optional. The path of the virtual directory where NetBrain is
deployed.

};
```

7. Initialize NetBrain instance.

```
var netbrainInstance, mapInstance;   // define and introduce global variables netbrain and
map.

NetbrainTools.createNetbrainInstance(netbrainOptions)

    .then(function(_netbrainInstance) {

        netbrainInstance = _netbrainInstance;

    })
```

```
    .then(initBinds)

    .catch(logError);
```

`netbrainInstance` contains NetBrain APIs encapsulated via JS functions for Embedded Map deployment. After initializing NetBrain instance, These APIs encapsulated via JS functions are available to call to construct embedded-map elements.

Sample <netbrainInstance>

```
{

    api: object,

    createMap: function,

    relogin: function

}
```

> **Note:** Before proceeding to initialize NetBrain instance, please ensure `netbrainInstance` is accessible. Failing to fetch netbrainInstance will cause error prompt indicating 'create netbrain instance failed'.

8. Use the `netbrainInstance.api.getTenants()` function to get all tenants accessible to a user. This function can be used to construct the drop-down list for the **Tenant Section** field.

```
netbrainInstance.api
    .getTenants({})
    .then(function(result) {
        var tenants = result.tenants;
        console.log(tenants);
    });

//output example:

{
  "tenants": [
    {
      "tenantId": "6b2ac0ce-5817-831f-4d79-acfc62aa2920",
      "tenantName": "xxf_tenant"
    },
    {
      "tenantId": "e58f795c-8262-126f-fa04-90210bfcbe6c",
      "tenantName": "xxf_sdn_tenant"
    },
    {
      "tenantId": "fc29f127-35af-20cc-797e-0caec793cc26",
      "tenantName": "rwang_tenant"
    },
    {
      "tenantId": "06b63b6c-698e-f5fb-c249-082036df95e1",
      "tenantName": "MIMIC"
    },
```

```
    {
      "tenantId": "42a296d4-d8ca-5a6c-5e0c-ecdc393f0ff7",
      "tenantName": "gcui_test"
    }
  ],
  "statusCode": 790200,
  "statusDescription": "Success."
}
```

> **Tip:** For more details about how to leverage NetBrain APIs to retrieve the tenant list, see Get all accessible tenants.

9. Use the `netbrainInstance.api.getDomains()` function to get all domains accessible to a user. This function can be used to construct the drop-down list for the **Domain Section** field.

> **Note:** Make sure to replace the highlighted values with the desired `tenantId` value.

```
netbrainInstance.api
    .getDomains({
        tenantId: '6b2ac0ce-5817-831f-4d79-acfc62aa2920'
    })
    .then(function(result) {
        var Domains = result.domains;
        console.log(Domains);
    });

// output example:

{
  "domains": [
    {
      "domainId": "debfbef3-fa56-45f0-82b1-a90e20154454",
      "domainName": "xxf_domain"
    },
    {
      "domainId": "10c05674-0b3b-4b4e-b672-87ae32eace98",
      "domainName": "normaluser"
    },
    {
      "domainId": "b9e6e637-eab0-4024-905f-3906756f2fb9",
      "domainName": "ertret"
    }
  ],
  "statusCode": 790200,
  "statusDescription": "Success."
}
```

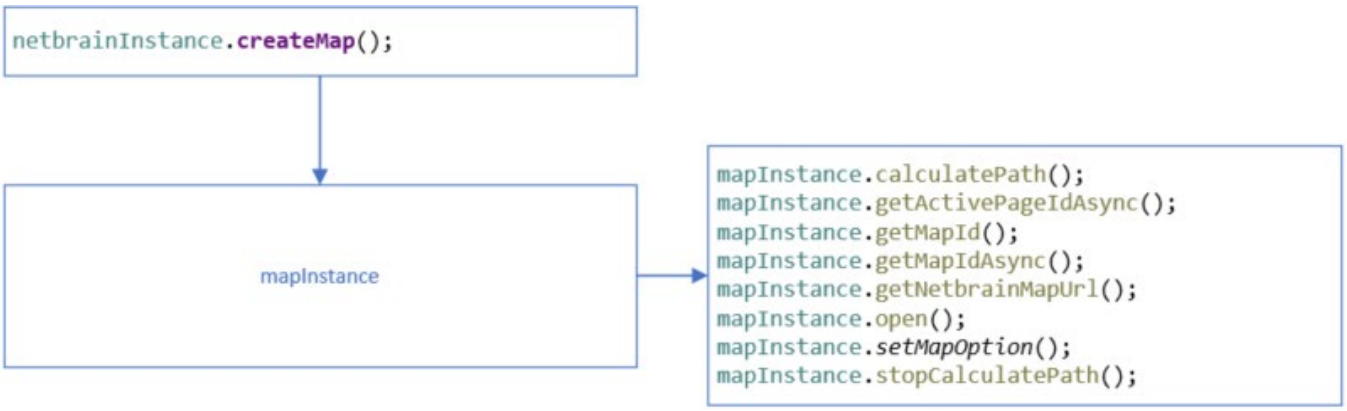
> **Tip:** For more details about how to leverage NetBrain API to retrieve the domain list from one of your NetBrain tenants, see Get all accessible domains of a tenant.

10. Initialize a map instance.

> **Note:** `mapContainer` is a container div used to render returned map instance.

```
netbrainInstance.createMap($('#mapContainer').get(0) ).then(function(Instance) {
    mapInstance = Instance;
});

//Verify mapInstance has been generated:

if (mapInstance) {
    openMap(param);
} else {
    netbrainInstance.createMap(mapOption.container).then(function (instance) {
        mapInstance = instance;
        openMap(param);
    });
}
```

`mapInstance` contains NetBrain APIs (encapsulated via JS functions) to open a map. After initiating the map instance, These APIs encapsulated via JS functions are available to call to build map-related elements.



11. Open a Site Map/Device Group Map/Public Map**.**

    1) Construct a map type drop-down list for the **Site Map/Device Group Map/Public Map**.

> **Note:** Site Map/Device Group Map/Public Map are hard-coded map types. The following APIs are designed for the purposes of rendering a map list when the corresponding map type is selected.

Site Map Type

> **Note:** Make sure to replace the highlighted values with your specific request parameters.

```
netbrainInstance.api
    .getSites({
        tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c',
        domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480',
```

```
        sitePath: 'My Network'
    })
    .then(function(result) {
        var sites = result.sites;
        console.log(sites);
    });

//result example:

{
  "sites": [
    {
      "siteId": "13e91c22-ffc8-4a31-82ad-786ae976c31e",
      "sitePath": "My Network/USXACanada",
      "isContainer": true,
      "children": [
        "e2eaad3f-b829-4715-87c0-72eb1f038274",
        "718a69b0-448b-4645-b0de-466adf88423d"
      ],
      "siteType": 1
    },
    {
      "siteId": "97a22e99-a042-4366-8ecc-d69bd32adc42",
      "sitePath": "My Network/USXACanada/United States of
America/MASSACHUSETTS/CANTON/Canton_NB1",
      "isContainer": false,
      "siteType": 2
    },
    {
      "siteId": "8ea30269-b348-4ca4-86fc-79557881a6f3",
      "sitePath": "My Network/USXACanada/United States of
America/MASSACHUSETTS/CANTON/Canton_NB2",
      "isContainer": false,
      "siteType": 2
    },
    // ...
    {
      "siteId": "732e8ab6-6b69-417d-ad03-2cc447100166",
      "sitePath": "My Network",
      "isContainer": true,
      "children": [
        "13e91c22-ffc8-4a31-82ad-786ae976c31e",
        "a3a9d329-fcdd-49d9-9669-9ec205d587c2",
        "74a72061-0412-428d-8448-6cbd3f9c70b1",
        "d2628098-be31-49b5-befc-167e9f1e9d88"
      ],
      "siteType": 0
    }
  ],
  "statusCode": 790200,
  "statusDescription": "Success."
}
```

> **Tip:** For more details about Site API parameters, refer to Get Site.

Device Group Type

> **Note:** Make sure to replace the highlighted values with your specific request parameters.

```
netbrainInstance.api
    .getDeviceGroups({
        tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c',
        domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480'
    })
    .then(function(result) {
        var deviceGroups = result.deviceGroups;
        console.log(deviceGroups);
    });

//result example:

{
  "deviceGroups": [
    {
    "id": "8196020b-b223-4bc8-8178-ac4da95b1695",
    "name": "#BGP 64512",
    "type": 2
  },
  {
    "id": "fbd027f2-24c9-4616-a6bd-62b7613b07e1",
    "name": "#BGP 65000",
    "type": 2
  },
  {
    "id": "f84008dd-a406-4b30-b71a-18e66b4a7d68",
    "name": "#BGP 65012",
    "type": 2
  }
  ],
  "statusCode": 790200,
  "statusDescription": "Success."
}
```

> **Tip:** For more details about Device Group API parameters, refer to Get Device Group.

Public Map Type

> **Note:** Make sure to replace the highlighted values with your specific request parameters.

```
netbrainInstance.api
    .getPublicFiles({
```

```
        tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c',
        domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480',
        folderId: '',
        fileTypes: [ 0, 11 ]
    })
    .then(function(result) {
        var files = result.Items;
        console.log(files);
    });

//result example:

{
    "items":[

        {
            "originalId":"267752c5-c01b-8ee3-d5aa-8fec88da8927",
            "id":"28337b50-25a0-40df-a0c8-20b79dce3f42",
            "name":"Public/test",
            "type":0
        },
        {
            "originalId":"267752c5-c01b-8ee3-d5aa-8fec88da8927",
            "id":"ff4e2847-2692-4a55-932c-b58978baaf2a",
            "name":"Public/Map3",
            "type":11
        },
        {
            "originalId":"7a249ae9-8dab-435d-88fa-5c698356c069",
            "id":"4a6cd1a3-558f-4288-acf3-e13825ea28af",
            "name":"Public/test/Map1cgc",
            "type":11
        }
    ],
    "statusCode":790200,
    "statusDescription":"Success."
}
```

> **Tip:** For more details about Public Map API parameters, refer to Get File.

2) Open a specific map. This step defines how to open a selected map.

> **Note:** Make sure to replace the highlighted values with your specific request parameters.

Open a site map

```
mapInstance.open({
    siteId: siteId,
    tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c',
    domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480'
}).then(function () {
```

```
    // ...
});
```

Open a device group map

```
mapInstance.open({
    deviceGroupId: deviceGroupId,
    tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c',
    domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480'
}).then(function () {
    // ...
});
```

Open a map in the Public file

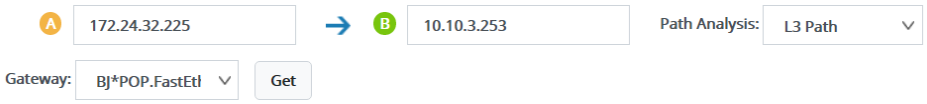
```
mapInstance.open({
    mapId: mapId,
    tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c',
    domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480'
}).then(function () {
    // ...
});
```

3) Refresh a map.

```
mapInstance.refresh();
```

12. Open a path map. In the portal, you can define options related to path calculation and open the path map by calculating a path.

Figure: example options of path calculation in an embedded-map portal



1) Calculate a path. Replace the highlighted values with your specific request parameters.

```
mapInstance.calculatePath({
    tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c', //required
    domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480', //required
    sourceIP: '10.10.3.253', //required
    sourcePort: 0,
    sourceGwIP: '10.10.3.253',
    sourceGwDev: 'GW2Lab',
    sourceGwIntf: 'GigabitEthernet0/0.10',
    destIP: '172.24.32.225', //required
    destPort: 0,
    pathAnalysisSet: 1,
    protocol: 4,
    isLive: 1
}).then(function (result) {
    var taskId =result.taskId
    // ...
});
```

2) View the path result.

```
netbrainInstance.api.getCalculatePathResult ({
    tenantId: 'e58f795c-8262-126f-fa04-90210bfcbe6c',
    domainId: 'b3752773-f37f-4a3e-b547-062d2c0a0480',
    taskId: 'b69c0af1-73ce-49e7-8cc7-59ce8ce5eb37'
}).then(function (result) {
    var hoplist =result.hopList
    // ...
});

// Result Example:

{
  "hopList": [
    {
      "hopId": "b1612e9f-491f-4edf-9e40-156052b0f708",
      "srcDeviceName": "GW2Lab",
      "inboundInterface": "GigabitEthernet0/1",
      "mediaName": "172.24.30.0/30",
      "dstDeviceName": "NY_Router",
      "outboundInterface": "FastEthernet0/0",
      "nextHopIdList": [
        "97f40ee6-dee0-4b1f-a723-fec130292aa0"
      ]
    },
    {
      "hopId": "97f40ee6-dee0-4b1f-a723-fec130292aa0",
      "srcDeviceName": "NY_Router",
      "inboundInterface": "Vlan100",
      "mediaName": "172.24.30.4/30",
      "dstDeviceName": "NY_POPP",
      "outboundInterface": "Ethernet0/1",
      "nextHopIdList": [
        "6846b938-a666-412c-99b8-9e400a80b6b7"
      ]
    },
    {
      "hopId": "6846b938-a666-412c-99b8-9e400a80b6b7",
      "srcDeviceName": "NY_POPP",
      "inboundInterface": "Ethernet0/0",
      "mediaName": "172.24.31.64/26",
      "dstDeviceName": "NY-core-bak",
      "outboundInterface": "FastEthernet0/0",
      "nextHopIdList": [
        "63d88b57-201e-41e3-85f3-a5965c450d52"
      ]
    },
    {
      "hopId": "63d88b57-201e-41e3-85f3-a5965c450d52",
      "srcDeviceName": "NY-core-bak",
      "inboundInterface": "FastEthernet0/1.1",
      "mediaName": "172.24.31.192/26",
      "dstDeviceName": "BJ*POP",
      "outboundInterface": "FastEthernet0/1",
```

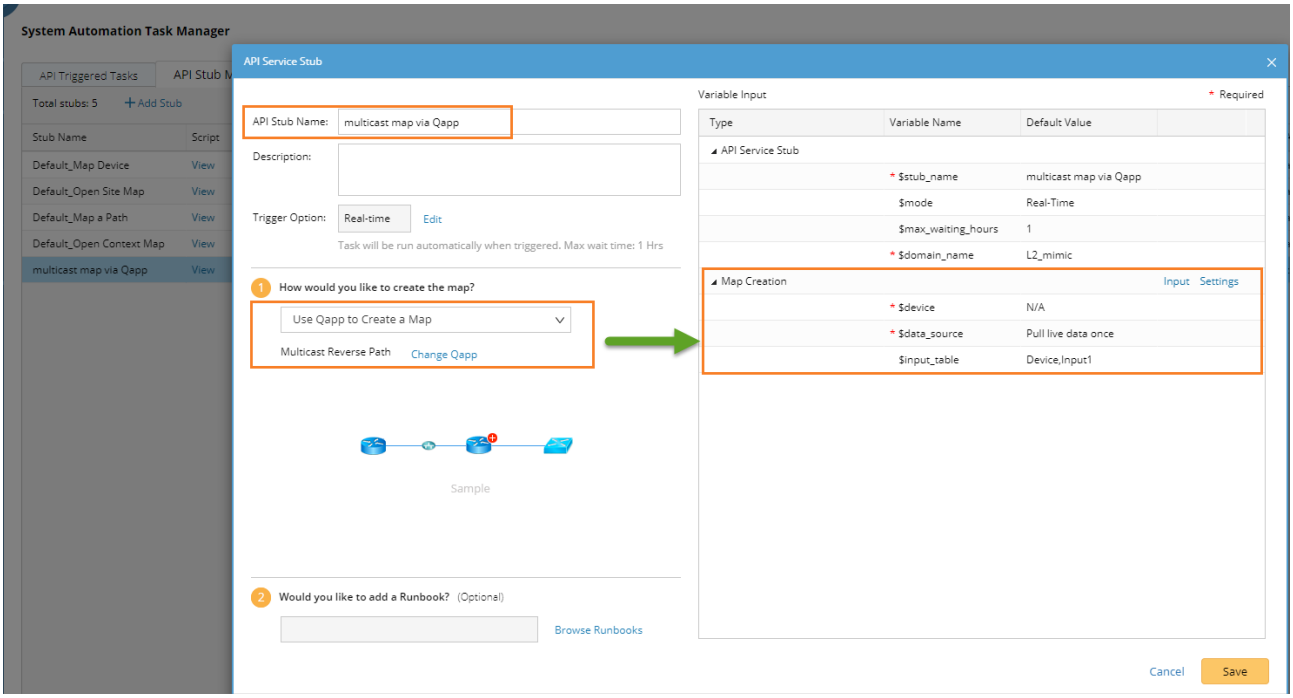
```
      "nextHopIdList": []
    }
  ],
  "statusCode": 790200,
  "statusDescription": "Success."
}
```

> **Tip:** For more details about the path APIs, refer to the following API documentation:
>
> - Calculate a path
> - Get the gateway of a device
> - Get the status of a path

13. Open a map created via Qapp. In the portal, you can call NetBrain triggered APIs to create a map using the Qapp added in an API-triggered task.

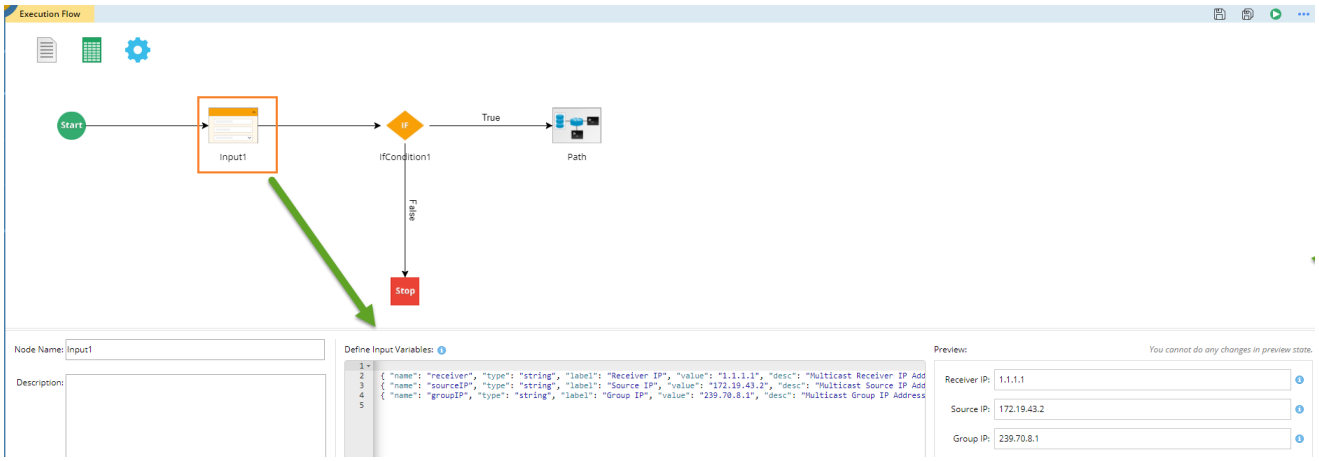
    1) Define an API-triggered task in your NetBrain IE system. In the task, select to create a map via Qapp.

    Figure: Define an API-triggered task to create a map via Qapp



A Qapp to create a map can contains input variables and you can create different maps via a Qapp by entering variable values.

Figure: Qapp with input variables



2) Call the API-triggered task to create a map via the Qapp in the API-triggered task.

   a) Call the API-triggered function and construct elements of creating a map via the Qapp**.** Replace the highlighted parameters with the actual values.

```
var param = {
    "tenantId": tenantId,
    "domainId": domainId,
    "basic_setting": {
        "user": $triggerAPIConfig.userName, //required, input a portal user name
        "device": $triggerAPIConfig.deviceName,//required, any device in your NetBrain system
        "stub_name": $triggerAPIConfig.stubName//required, must be same with the API stub name in your NetBrain system
    },
    "domain_setting": {
        "tenant_id": tenantId,
        "domain_id": domainId
    },
    "map_setting": {
        "map_create_mode": 7,
        "map_qapp_para": {
            "dataSource": {
                "type": 1 //required 1:Pull live data once, 2:Current baseline
            },
            "frequency": {
                "type": 1,
```

```
                "times" : 10,
                "interval": {
                    "unit": 2,
                    "duration": 2
                }
            },
            "input_variable_parameters": [{  // The parameters must be the same with those you have defined in the input variables of a Qapp. If no input variables,set it as "input_variable_parameters": []
                "desc": "Multicast Receiver IP Address",
                "label": "Receiver IP",
                "name": "receiver",
                "value": "172.24.30.2"
            }, {
                "desc": "Multicast Source IP Address",
                "label": "Source IP",
                "name": "sourceIP",
                "value": "172.19.43.2"
                }]
        }
     }
};
netbrainInstance.api.triggerAPI(param).then(function(data) {
    if (data.error) {
        alert(data.error);
        return;
    }
    netbrainInstance.createMap(mapOption.container).then(function (instance) {
        mapInstance = instance;
        triggerTaskStatus(data, instance);
    });
});
```

b) Call the `netbrainInstance.api.triggerTaskStatus` function to get the status of the API-triggered task. The call ends when the task fails or is finished.

```
function triggerTaskStatus(data, instance) {
    netbrainInstance.api.triggerTaskStatus({
        "taskId": data.taskId,
        "tenantId": tenantId,
```

```
        "domainId": domainId //
    }).then(function(res) {
        if(res == 2 || res == 3) {
            setTimeout(function() {
                triggerTaskStatus(data, instance)
            }, 5000);
        }
        else {
            triggerDVT(data, instance);
        }
    });
}
```

c) Call the `netbrainInstance.api.triggerDVT` function to apply the data view generated via running the Qapp to the map.

```
function triggerDVT(data, instance) {
    netbrainInstance.api.triggerDVT({
        "taskId": data.taskId,
        "tenantId": tenantId,
        "domainId": domainId
    }).then(function(res) {
        var param = {
            tenantId: tenantId,
            domainId: domainId,
            mapId: data.mapId
        };
        if (res && res.length > 0) {
            param.dataViewId = res[0].dataViewGroupId;
        }
        // open map
        mapInstance.open(param).then(function () {
              /*
                 ...
                 success code
                 ...
              */
        }).catch(function(err) {
```

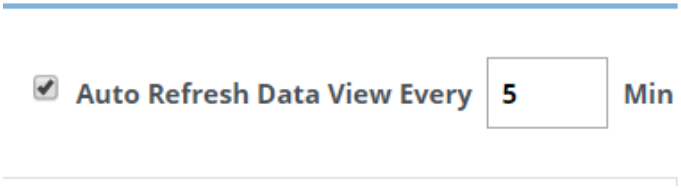
```
            console.log(err);
        });
    });
}
```

Figure: Sample UI of map creation via a Qapp with input variables

14. Refresh data views applied to an embedded map automatically using the `mapInstance.setMapOption()` function.

```
netbrainInstance.setMapOption({
    dataViewRefresh: {
        enable: true,
        frequency: 5
    }
});
```

Figure: Sample UI of Auto Refresh Data View.



6. Advanced Deployment Modes

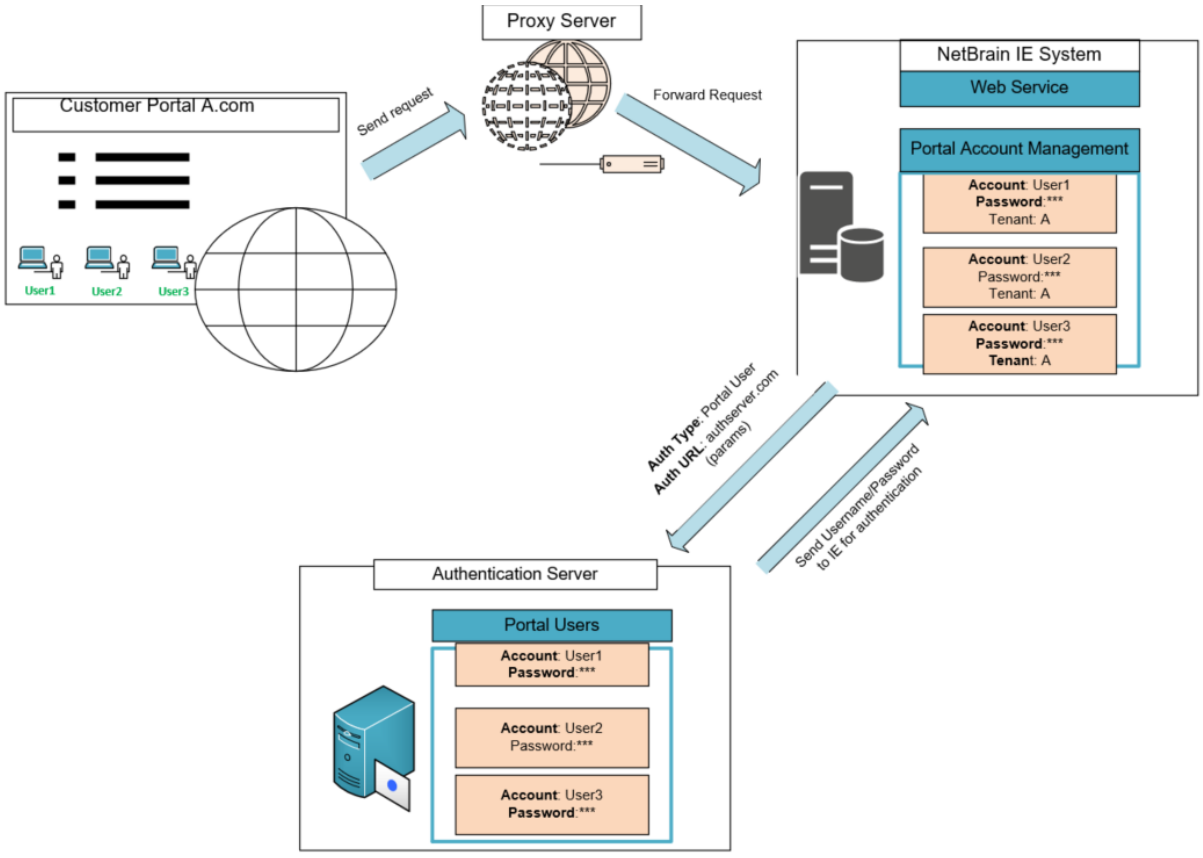
Besides the standard deployment mode, the embedded-map feature also supports the following deployment modes:

- Proxy Deployment
- Multi-tenant Deployment

6.1. Proxy Deployment

In the **Proxy Deployment** mode, a proxy server is deployed to forward requests and responses between the embedded-map portal and the NetBrain IE system.

Figure: Sample architecture of proxy development mode



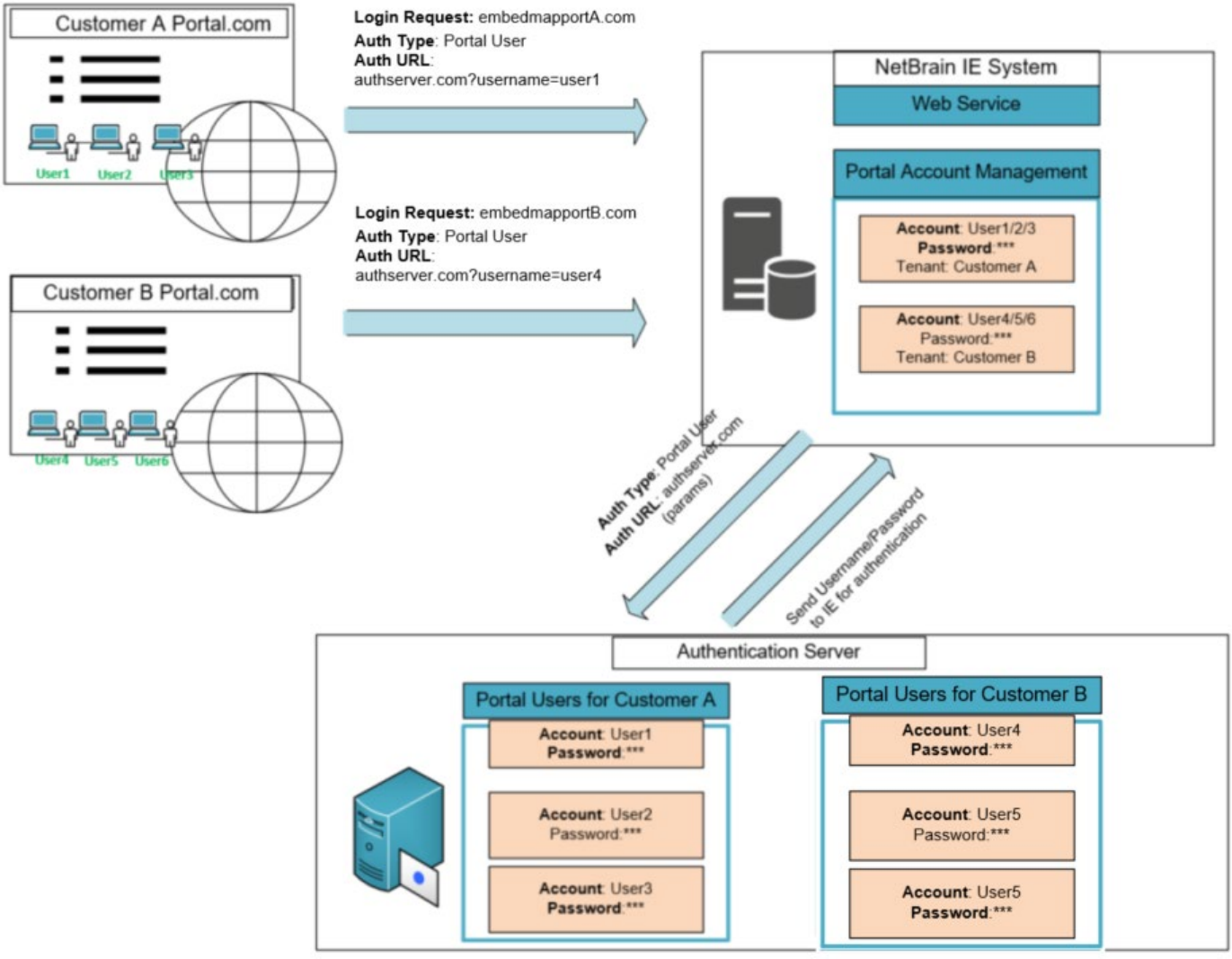
The detailed solution varies from your existing network deployment architecture and requires knowledge of NetBrain protocols, embedded-map requests, and other information. And if you consider this deployment mode, contact NetBrain Support Team to get technical support.

6.2. Multi-tenant Deployment

When you manage different customer networks based on tenants in your NetBrain system, it will be a good practice to deploy the embedded-map feature in the multi-tenant mode.

In this mode, different portals are created for customer networks with requests sent to the same one IE system and authentication done based on different portals and tenants.

Figure: Sample architecture of multi-tenant development mode



The solution to this deployment mode is described as follows:

1. In your NetBrain IE system, create portal users and assign tenants to each user based on portals.
2. Append usernames that belong to a portal as parameters to the authentication server URL when setting up authentication server URL in the portal.

```
<script>

// init the param for netbrainTools, virtualDir is option, virtualDir: '/' or virtualDir:
'/NG', and default is '/' if no this field.

// 'https://www.netbrain-az.com'

    netbrainInstanceOption = { netbrainUrl: 'NetBrain IE address', virtualDir: '/',
authentication: { type: 'portalUser', authUrl: 'authUrl?username=user1' },
```

3. On Authentication Server side, extract the username parameter from the request and then select the corresponding username/password for authentication.

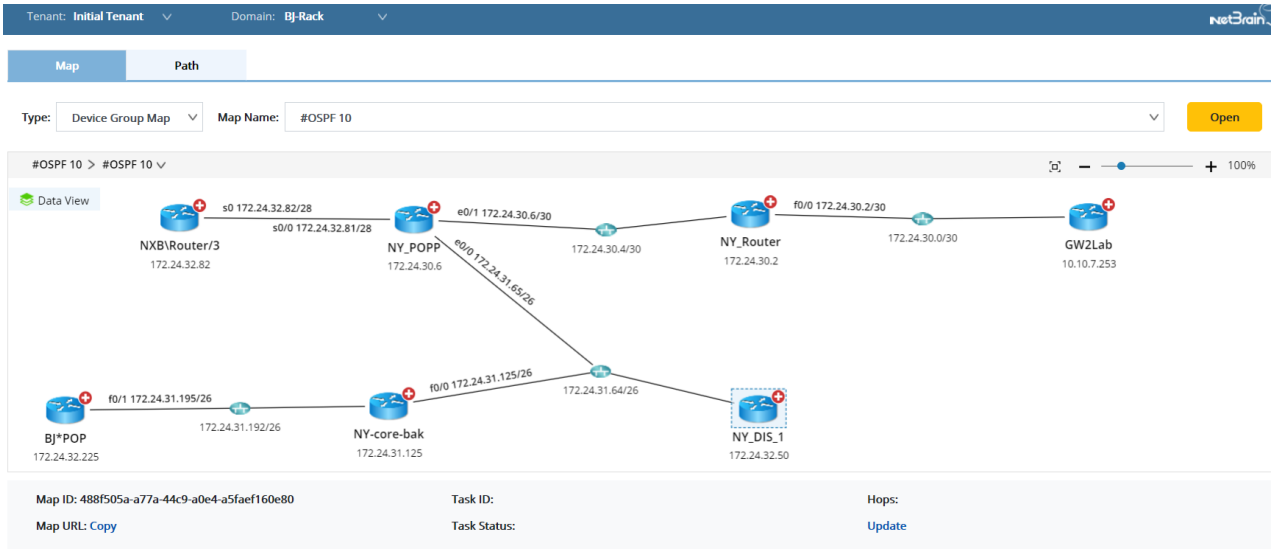
```
string username = HttpUtility.ParseQueryString(authUrl).Get("username ")
```

7. Working with Embedded Map Portal

Follow the steps below to view an embedded map in your portal.

7.1. Open a Device Group Map

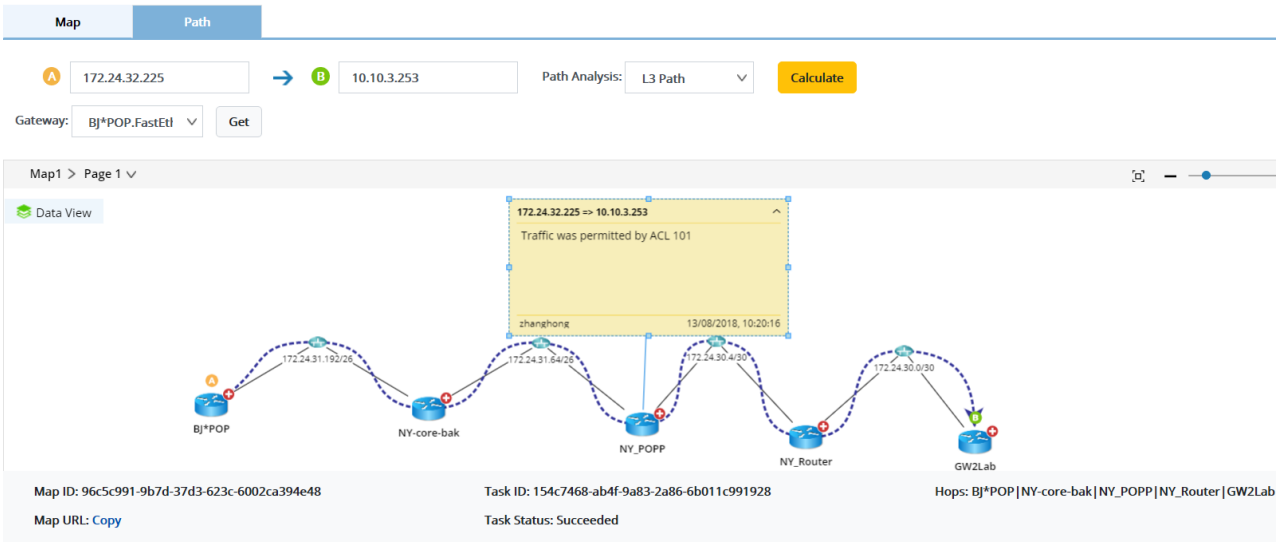
1. Log into your portal and navigate to the NetBrain page.
2. Select the desired tenant and domain.
3. Select **Device Group Map** from the **Type** drop-down menu.
4. Select a map from the **Map Name** drop-down menu.
5. Click **Open**.



7.2. Calculate a Path

1. Log into your portal and navigate to the NetBrain page.
2. Select the desired tenant and domain.
3. Create a map by calculating a path.
    1) Enter the source IP and click **Get** to obtain the gateway list of the source device.
    2) Enter the destination IP and select a path analysis method.

3) Click **Calculate** to start the path calculation.



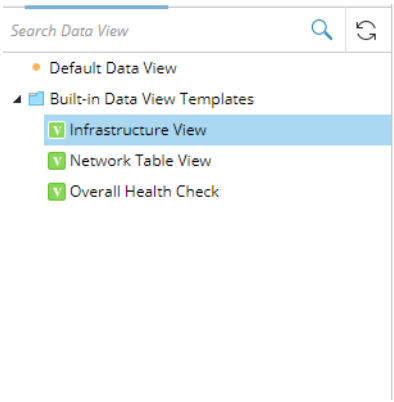
> **Note:** The calculation uses live data as a data source by default.

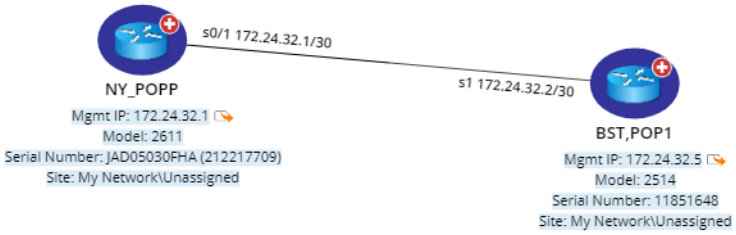
7.3. Embedded Map Operation

You can perform the following operations on an embedded map.

Applying data views

1. On the map page, click the **Data View** tab. It lists all the data views applicable to the current map page.
2. On the **Data View** tab, select a data view to apply. Data units in this data view are highlighted with different background colors.





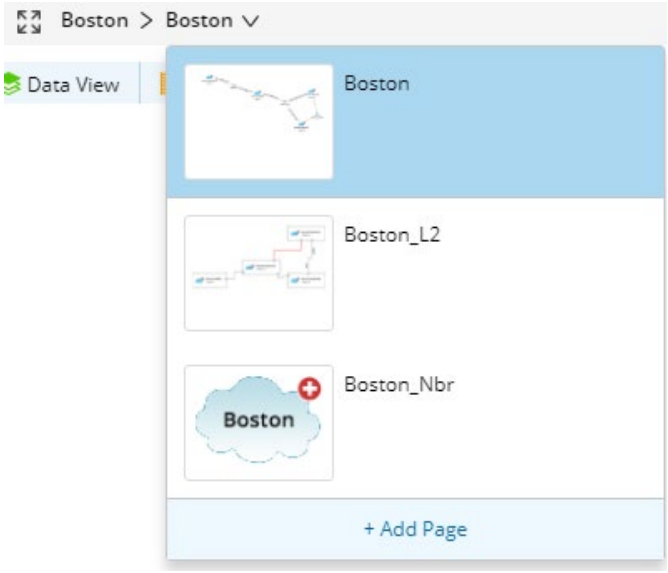
> **Tip:** You can switch between different data views in the Data View pane to suit your specific project needs.

Fitting to Screen

On the upper-right corner of a map, you can click the icon to adjust the display ratio of your map to fit your screen.

Browsing more map pages

Click the map name and select more maps from the drop-down menu.



Zooming to view more data details

In an embedded map, you can zoom in the map to show more data details over an interface link.

> **Note:** The following functionalities are currently not supported by Embedded Map.
>
> - Executing a Runbook
> - Running a Qapp
> - Extending Neighbor
> - Operations in the right-click menu
> - Pointing to a link or data to show the corresponding configuration in a tip window

Appendix: NetBrain APIs for Embedded Map Deployment

The table below lists the NetBrain APIs you may use to deploy the NetBrain Embedded Map.

Category	APIs
Tenant and Domain List	▪ Get all accessible tenants ▪ Get all accessible domains of a tenant
Site	▪ Get child sites of a specific site
Device Group	▪ Get device group list
File	▪ Get file list
Path	▪ Calculate a path ▪ Get path calculation result ▪ Get path calculation status ▪ Get the gateway information of a device ▪ Stop a path
Authentication	▪ Generate authentication node

Get all accessible tenants

This function returns a list of accessible tenants (including tenant ID and names). The returned tenant list varies by the user privileges you use to log in. To retrieve a full list of all available tenants, you must log in with admin permissions.

Resource Information

Method	URL	Required authentication
GET	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Tenants	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials. **Example:** `{ "token": "301ebdd8-0045-429d-8807-c51b8db7f5f0" }`

Response

Parameter	Type	Description	Example
tenants	array	A list of all accessible tenants.	`{ "tenants": [ { "tenantId": "3e75247a-309c-4231-96a5-823b6cb1e78d", "tenantName": "bos_office" }, { "tenantId": "5a75247a-309c-4231-96a5-823b6cb1e78d", "tenantName": "ny_office" } ] }`
tenantId	string	The tenant ID.	
tenantName	string	The tenant name.	
statusCode	integer	Code issued by NetBrain server indicating the execution result.	
statusDescription	string	The explanation of the status code.	

Get all accessible domains of a tenant

This function returns a list of accessible domains in a specific tenant. The returned accessible domains vary by the user privileges you use to log in. To retrieve a full list of domains in a specified tenant, you must log in with system admin or tenant admin permissions.

Resource Information

Method	URL	Required authentication
GET	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API//V1/CMDB/Domains	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.
tenantId *(query)*	Unique identifier for the tenant from which you desire to retrieve the domain information. tenantId can be retrieved from get all accessible tenants.

Response

Parameter	Type	Description	Example
domains	array	A list of all accessible domains.	{ "domains": [{ "domainId": "4e75247a-309c-4231-96a5-823b6cb1e78d", "domainName": "domain1" }, { "domainId": "5e75247a-309c-4231-96a5-823b6cb1e78d", "domainName": "domain2" }] }
domainId	string	The domain ID.	
domainName	string	The domain name.	
statusCode	integer	Code issued by NetBrain server indicating the execution result.	
statusDescription	string	The explanation of the status code.	

Get child sites of a specific site

This function returns a list of child sites of a specified site.

Resource Information

Method	URL	Required authentication
GET	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Sites/ChildSites	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.
sitePath*required *(query)*	The full path of a site, for example, My Network/NA/US.

Response

Parameter	Type	Description	Example
sites	array	A list of all child sites.	{ "sites": [{ "siteId": "47e5d1c1-5ddc-4e5f-b37c-21616193dd36", "sitePath": "My Network/C0/L4", "isContainer": false, "siteType": 2 }, { "siteId": "020cb2a2-d192-4c29-a9bd-53787d866f85", "sitePath": "My Network/C0", "isContainer": true, "children": ["47e5d1c1-5ddc-4e5f-b37c-21616193dd36", "16d1cd8e-eb8e-42ca-a19d-54d7a7fbd2a2", "c36eb043-a30f-4b58-b05f-957f845c40e3", "688bc6b2-3b34-42ff-96a2-c06687d2c03a", "6348e733-1c6b-4d76-8926-2d20622cf164", "3ed2ccba-9a00-48d7-9af0-
siteId	string	The site ID.	
sitePath	string	The full path of a site.	
siteType	integer	The type of a site. ▪ 0: root site ▪ 1: container site ▪ 2: leaf site	
isContainer	bool	Whether it is a container site.	
children	array	List of child sites of the specified container site.	
statusCode	integer	Code issued by NetBrain server indicating the execution result.	
statusDescription	string	The explanation of the status code.	

Parameter	Type	Description	Example
			a17e9aa8ccfb"], "siteType": 1 }], "statusCode": 790200, "statusDescription": "Success." }

Calculate a Path

This function is used to calculate a path between two endpoints. The result will be returned in the form of a path ID, and you can use the path ID in the Get Path Calculation Result as the request parameter to get each hop information of the path.

Resource Information

Method	URL	Required authentication
POST	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Path/Calculation	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.
body*required *(body)*	The request body contains the following parameters: ▪ sourceIP*(string): IP address of the source device. ▪ sourcePort (integer): Source protocol port, for example, 23 for telnet. This parameter can be null. ▪ sourceGwIP* (string): Gateway for path calculation. ▪ sourceGwDevice* (string): Hostname of the gateway device. ▪ sourceGwIntf* (string): Name of the gateway interface. ▪ destIP*(string): IP address of the destination device. ▪ destPort (integer): Destination protocol port, for example, 23 for telnet. This parameter can be null. ▪ pathAnalysisSet* (integer): Path type used to calculate. ○ 1: L3 Path ○ 2: L2 Path ○ 3: L3 Active Path ▪ protocol* (integer): Application protocol, for example, 4 for IPv4, and 6 for TCP. ▪ isLive (integer): Data source used to calculate the path. ○ 0: use data from the current baseline. ○ 1: use data via live access. **Example**: { "sourceIP": "10.10.3.253", "sourcePort": 0, "sourceGwIP": "10.10.3.253", "sourceGwDev": "GW2Lab", "sourceGwIntf": "GigabitEthernet0/0.10", "destIP": "172.24.32.225", "destPort": 0,

Name	Description

	<pre> "pathAnalysisSet": 1,
	  "protocol": 4,
	  "isLive": 1
	}</pre>

Response

Parameter	Type	Description	Example

taskId	string	The ID of the task.	<pre>{
			  "taskId": "string",
			  "statusCode": 790200,
			  "statusDescription": "success"
			}</pre>

statusCode	integer	Code issued by NetBrain server indicating the execution result.	
statusDescription	string	The explanation of the status code.	

Get the gateway information of a device

This function returns the gateway information of a device based on its IP or hostname.

Resource Information

Method	URL	Required authentication
GET	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Path/Gateways	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.
ipOrHost*required *(query)*	The IP or hostname of a device.

Response

Parameter	Type	Description	Example

gatewayList	array	A list of returned gateway devices. The list contains the following parameters:
		▪ ip (string): the IP address of a gateway.
		▪ devName (string): the hostname of a gateway.
		▪ intfName (string): the name of the gateway interface.
			<pre>{
			  "statusCode": 790200,
			  "statusDescription": "success",
			  "gatewayList": [
			  {
			  "ip": "string",
			  "devName": "string",
			  "intfName": "string"
			  }
			  ]
			}</pre>

statusCode	integer	Code issued by NetBrain server indicating the execution result.	
statusDescription	string	The explanation of the status code.	

Get path calculation status

This function returns the path status.

Resource Information

Method	URL	Required authentication
GET	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Path/Calculation/{taskID}/Status	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.
taskId*required *(query)*	The task ID retrieved from Calculate a Path.

Response

Parameter	Type	Description	Example

statusCode	integer	Code issued by NetBrain server indicating the execution result.	{
			"statusCode":790200,
			"statusDescription":"success",
			"result":{
			"resultCode":1,
			"resultDescription":"Running"
			}
			}

statusDescription	string	The explanation of the status code.	
resultCode	integer	The status code of the specified path. ▪ 0: Initialized ▪ 1: Running ▪ 2: Succeeded ▪ 3: Failed ▪ 4: Canceled	
resultDescription	string	The explanation of the result code.	

Get path calculation result

This function returns the hop information of a path calculated through the Calculate a Path API.

Resource Information

Method	URL	Required authentication
GET	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Path/Calculation/{taskID}/Result	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.
taskID*required *(query)*	The task ID retrieved from Calculate a Path.

Response

Parameter	Type	Description	Example
statusCode	integer	Code issued by NetBrain server indicating the execution result.	
statusDescription	string	The explanation of the status code.	
hopList	array	A list of hops along a path.	
hopId	string	The ID of a hop.	
srcDeviceName	string	The hostname of the source device.	
inboundInterface	string	The name of the inbound interface.	
mediaName	string	The media name.	
dstDeviceName	string	The hostname of the destination device.	
outboundInterface	string	The name of the outbound interface.	
nextHopIdList	array	A list of the IDs of next hops.	


```
{
  "statusCode": 790200,
  "statusDescription": "success",
  "hopList": [
    {
      "hopId": "string",
      "srcDeviceName": "string",
      "inboundInterface": "string",
      "mediaName": "string",
      "dstDeviceName": "string",
      "outboundInterface": "string",
      "nextHopIdList": [
        "7a09854d-1d87-4656-8556-a7b142fccb75"
      ]
    }
  ]
}
```

Get device group list

This function returns a list of device groups.

Resource Information

Method	URL	Required authentication
GET	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/DeviceGroups	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.

Response

Parameter	Type	Description	Example
deviceGroups	array	A list of all accessible domains.	{ "deviceGroups": [{ "id":"8196020b-b223-4bc8-8178-ac4da95b1695", "name":"#BGP 64512", "type":2 }, { "id":"fbd027f2-24c9-4616-a6bd-62b7613b07e1", "name":"#BGP 65000", "type":2 }, { "id":"fcdb1b8b-bffd-478f-a914-e867f6a87f86", "name":"retest", "type":0 }], "statusCode":790200, "statusDescription":"Success." }
id	string	The ID of a device group	
name	string	The name of a device group	
type	integer	The type of a device group. ▪ 0: Public ▪ 1: Private ▪ 2: System	

Get file list

This function returns a list of files contained in a specified folder.

Resource Information

Method	URL	Required authentication
POST	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Files/	Yes

Parameters

Name	Description
Token*required (header)	The token can be obtained by sending a POST request to the log in session endpoint and provide valid credentials.

Body*required (body)	The request body contains the following parameters:
	▪ folderId (string): The ID of the folder from which you want to get the files. Root folder (public folder) will be returned if folderId is null.
	▪ fileTypes*(array): the file types you want to retrieve. There are three file types:
	▪ 0: Folder
	▪ 11: Map
	▪ 21: Dashboard
	Example:
	{
	"folderId": "",
	"fileTypes": [0, 11 ,21]
	}

Response

Parameter	Type	Description	Example
items	array	A list of folders and files.	{ "items": [{ "originalId":"75ff3cdf-dff4-48c6-a736-7a86e4374a29", "id":"ad09aa07-b31d-4f42-a0aa-319697825b09", "name":"Public/Site Maps", "type":0 }, {
id	string	The ID of a folder in the file tree.	
name	string	The name of a file.	
originalId	string	The ID of a specific dashboard or file. (Used for Map or Dashboard type only.)	
type	integer	The type of a file. ▪ 0: Folder	

Parameter	Type	Description	Example

		▪ 11: Map
		▪ 21: Dashboard
			"originalId":"75ff3cdf-dff4-48c6-a736-7a86e4374a29",
			"id":"2a19165f-a4a5-4488-ac5d-acdf9e287ed6",
			"name":"Public/New Folder/New Folder/New Map",
			"type":11
			},
			{
			"originalId":"d2650deb-5276-44cb-be21-43e2b129380a",
			"id":"a84cdca3-3710-47b1-b037-665e38fd6d08",
			"name":"Public/New Folder(1)/New Map",
			"type":11
			}
			],
			"statusCode":790200,
			"statusDescription":"Success."
			}

Stop a path

This API is used to force stop a path calculation process.

Resource Information

Method	URL	Required authentication
Post	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/API/V1/CMDB/Path/Calculation/Stop	Yes

Parameters

Name	Description
Token*required *(header)*	The token can be obtained by sending a POST request to the login session endpoint and provide valid credentials.
taskId*required *(query)*	The task ID retrieved from Calculate a Path.

Response

Parameter	Type	Description	Example
statusCode	integer	Code issued by NetBrain server indicating the execution result.	{ "result": true, "statusCode": 790200, "statusDescription": "Success." }
statusDescription	string	The explanation of the status code.	
result	bool	The execution (force stop) result.	

Generate Authentication Code

This function returns an authentication code based on username and password.

Resource Information

Method	URL
Post	http(s)://<IP address of your NetBrain Web API Server>/ServicesAPI/EmbedMap/GenerateAuthenticationCode

Parameters

Name	Description
userName*required *(body)*	The name of an Embed-Map user received from the authentication server.
password*required *(body)*	The password that received from the authentication server.

Response

Parameter	Type	Description
ResultCode	integer	Code issued by NetBrain server indicating the execution result.
ResultDesc	String	The result description. ▪ 0: Operation Successful ▪ 480301: Invalid user. ▪ 480302: Incorrect username or password ▪ 480303: Empty username or password ▪ 480304: Exception
authenticationCode	string	The returned authentication code.