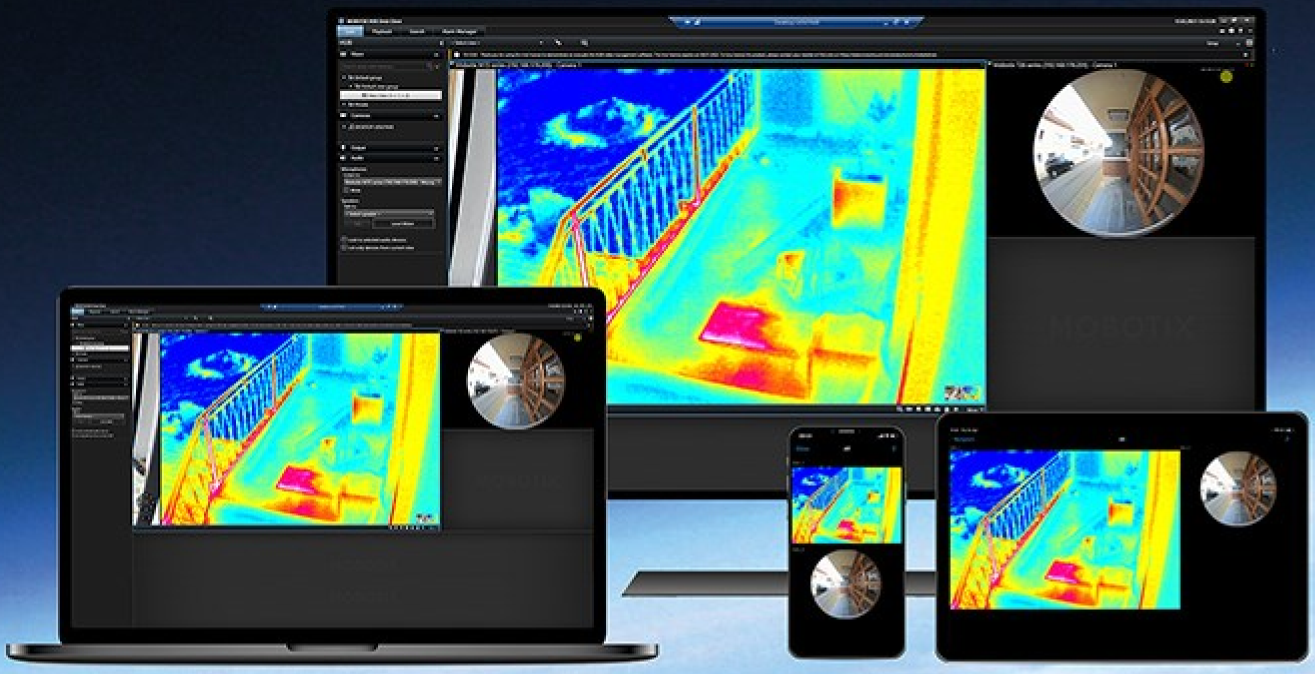
Administrator manual

MOBOTIX HUB API Gateway 2022 R3



Beyond Human Vision

MOBOTIX

Table des matières

Copyright

MOBOTIX AG • Kaiserstrasse • D-67722 Langmeil • Tél. : +49 6302 9816 0 • sales@mobotix.com • www.mobotix.com

Overview

Introduction

The MOBOTIX HUB VMS is planned to include RESTful APIs that expose the functionality currently available through native .NET libraries and various protocols.

The API Gateway is installed on-premise and is intended to serve as a front-end and common entry point for RESTful API services on all the current VMS server components (management server, event server, recording servers, log server, etc). An API Gateway can be installed on the same host as the management server or separately, and more than one can be installed (each on their own host).

New in MOBOTIX HUB VMS 2022 R3

- The API Gateway can be configured to support Cross-Origin Resource Sharing (CORS).
- Pre-release of WebRTC support.

New in MOBOTIX HUB VMS 2022 R2

- A number of syntax issues in the OpenAPI spec file have been fixed.
- More complete coverage of the Configuration API.
- **Breaking change**: In some parts of the RESTful API, booleans were treated as strings, meaning that the values when provided as input would have to be enclosed in quotation marks and also would be returned in this form. As this is not in compliance with the JSON standard, and also not in accordance the OpenAPI spec file we provide, we have decided to change it to use true/false without the quotation marks.

Limitations

- To upgrade from the 2021 R2 pre-release of the API Gateway to later releases, you'll have to uninstall the 2021 R2 pre-release before upgrading.

  Only the Configuration API is exposed as a RESTful API through the API Gateway.

Intended audience

This document is primarily aimed at system integrators and IT administrators. You are assumed to be somewhat familiar with MOBOTIX HUB VMS products.

API Gateway features

The MOBOTIX HUB VMS offers a number of APIs to support integrations. The full functionality is currently available through a plug-in environment, through native .NET libraries, and through various SOAP and native protocols. These APIs are used internally by MOBOTIX HUB VMS, and a large number of integrations have been developed using these APIs. But they are not practical for integrations in a cloud environment:

- The SOAP-based protocols relies on Windows Communication Framework (WCF) which is part of .NET Framework, making it difficult to implement non-Windows integrations.
- To use the protocols, your integration must keep track of a number of service endpoints.

The API Gateway simplifies this by providing a single entry point for all services.

The API Gateway relies on an OpenID Connect and OAuth 2.0 Identity Provider (IDP) for authentication and authorization.

To use the API Gateway, a client first authenticates and requests an access token from the Identity Provider. The client receives a bearer token that grants privileges to access services and to perform operations, as determined by the user's roles.

The client now uses the bearer token in the authorization header in subsequent requests. The client renews the bearer token before it expires by posting a new access token request with the same credentials.

The API Gateway acts as broker, routing requests and responses between external clients and the various downstream MOBOTIX HUB VMS services.

The RESTful API is implemented in part by each specific VMS server component, and the API Gateway can simply pass-through these requests and responses, while for other requests, the API Gateway will convert requests and responses as appropriate.

> ⚠ User credentials, bearer tokens, and other sensitive data are transmitted in cleartext if you do not set up certificates and use HTTPS.

Requirements and considerations

Considerations

For development, set up separate development system

You are recommended to use a separate development system.

Use HTTPS

You should consider setting up a server certificate and using HTTPS. While the IDP, API Gateway, and the Management Server all can work with either HTTP or HTTPS, production systems should be set up with server certificates.

> ⚠ User credentials, bearer tokens, and other sensitive data are transmitted in cleartext if you do not set up certificates and use HTTPS.

Requirements

Installation of the HUB API Gateway requires MOBOTIX HUB VMS 2022 R1 or later

Server certificates and host names

If you set up the management server with encryption, you must also set up all API Gateway instances that connect to the management server with encryption. To enable this, the IIS on the host that you install the API Gateway on must be set up with a server certificate.

The server hostname you specify during installation of the API Gateway is used to connect the API Gateway to the identity provider service (IDP) and management server in the MOBOTIX HUB VMS, and should match a DNS name in the management server certificate.

MOBOTIX HUB users

The API Gateway installer must be able to log in to the MOBOTIX HUB VMS during the installation. The Windows user account that you used for installing the MOBOTIX HUB VMS has been added in the MOBOTIX HUB VMS to the Administrators role. You can use the same Windows account when you install the API Gateway.

To authenticate and access the API Gateway, you need an MOBOTIX HUB Basic user account.

- You can create an MOBOTIX HUB Basic user during installation of the MOBOTIX HUB VMS.
- After installation, you can use the **MOBOTIX HUB Management Client** to create MOBOTIX HUB Basic or Windows users.

Installation

Installation overview

This installation overview describes the following tasks:

- Installing the API Gateway, either during or after the initial installation of the MOBOTIX HUB VMS.
- Verifying that an API Gateway is operational.

Install the API Gateway

There are several ways to install an API Gateway:

- During installation of a either a **Single computer** or a **Custom** (distributed) MOBOTIX HUB VMS, using the MOBOTIX HUB VMS product installer.
- After installation of a MOBOTIX HUB VMS, using the API Gateway installer downloaded from the management server's Administrative Installation Page.

Install the API Gateway using the MOBOTIX HUB VMS product installer

The API Gateway is included by default in both **Single computer** and **Custom** installations.

The API Gateway will be installed on the same host as the management server and with the same server certificate if you enable encryption.

Install the API Gateway using the API Gateway installer

The management server has a built-in public installation web page. From this web page, administrators and end-users can download and install additional system components. For example, if you didn't initially install an API Gateway, you can install it later.

1. From the computer where management server is installed, go to the management server's download web page. In Windows' **Start** menu, select **MOBOTIX** > **Administrative Installation Page** and write down or copy the address for later use when installing the system components on the other computers. The address is typically `http://[management server address]/installation/Admin/default-en-US.htm`.
2. Log into the computer where you want to install the API Gateway.
3. Open the management server's download web page in a web browser.
4. Locate the **API Gateway Installer** section, and select **All Languages** to start downloading the installer. Save the installer first, or run it directly from the web page.
5. Choose installation language.
6. Accept license terms.
7. Enter the management server address. Use HTTPS if the management server is configured to be secure.

8. Select the web site on the local IIS to use with the API Gateway, usually the Default Web Site.
9. If the management server is configured to be secure, you must select a server certificate for the API Gateway host. If you are installing the API Gateway on the host of the management server, select the server certificate you used when installing MOBOTIX HUB VMS.
10. Select the service account for the API Gateway, usually the Network Service account.
11. Select file location and product language.
12. Select **Install**.

Verify that the API Gateway is operational

> Replace the hostname `test-01.example.com`, username `seamrune`, and password `Rad23Swops#` in the following request samples. In Windows Command Prompt (CMD), replace the line continuation character `\` with `^`.

1. Verify that you can get a list of well-known URIs from the API Gateway:

   cURL

   ```
   curl --insecure --request GET "https://test-01.example.com/api/.well-known/uris"
   ```

   PowerShell

   ```
   $response = Invoke-RestMethod 'https://test-01.example.com/api/.well-known/uris' -
   Method 'GET'
   $response | ConvertTo-Json
   ```

   Response body

   ```
   {
    "ProductVersion": "22.1.5804.1",
    "UnsecureManagementServer": "http://test-01.example.com/",
    "SecureManagementServer": "https://test-01.example.com/",
    "IdentityProvider": "https://test-01.example.com/IDP",
    "ApiGateways": [
      "https://test-01.example.com/API/"
    ]
   }
   ```

   > In case you had installed an API Gateway on another host, you could use the hostname of that host.

2. Verify that you can authenticate and retrieve a bearer token from the built-in IDP.

   cURL

   ```
   curl --insecure --request POST "https://test-01.example.com/idp/connect/token" \
   --header "Content-Type: application/x-www-form-urlencoded" \
   --data-urlencode "grant_type=password" \
   --data-urlencode "username=seamrune" \
   --data-urlencode "password=Rad23Swops#" \
   --data-urlencode "client_id=GrantValidatorClient"
   ```

   PowerShell

   ```
   $headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"
   $headers.Add("Content-Type", "application/x-www-form-urlencoded")
   $body = @{grant_type='password'
       username='seamrune'
       password='Rad23Swops#'
       client_id='GrantValidatorClient'}
   $response = Invoke-RestMethod 'https://test-01.example.com/idp/connect/token' `
       -Method 'POST' -Headers $headers -Body $body
   $response | ConvertTo-Json
   ```

   Response body

   ```
   {
    "access_token": "eyJhbG . . . YTWPjg",
    "expires_in": 3600,
    "token_type": "Bearer",
    "scope": "managementserver"
   }
   ```

   Copy the `access_token` value from the response body; you will use the value as the bearer token value in the following request.

3. Verify that you can submit a request through the API Gateway.

   Replace the hostname `test-01.example.com` and the bearer token value `eyJhbG . . . YTWPjg` in the following request samples.

   cURL

```
curl --insecure --request GET "https://test-01.example.com/api/rest/v1/sites" \
--header "Authorization: Bearer eyJhbG . . . YTWPjg"
```

   PowerShell

```
$headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"
$headers.Add("Authorization", "Bearer eyJhbG . . . YTWPjg")
$response = Invoke-RestMethod 'https://test-01.example.com/api/rest/v1/sites' `
    -Method 'GET' -Headers $headers
$response | ConvertTo-Json
```

   Response body

```
{
 "array": [
  {
   "displayName": "TEST-01",
   "id": "2d12465c-3485-4ca8-a9fb-86a79de1a82f",
   "name": "TEST-01",
   "description": "",
   "lastModified": "2021-11-11T11:11:11.1111111Z",
   "timeZone": "Central Europe Time",
   "computerName": "TEST-01",
   "domainName": "example.com",
   "lastStatusHandshake": "2021-11-11T11:11:11.1111111Z",
   "physicalMemory": 0,
   "platform": "[Not Available]",
   "processors": 0,
   "serviceAccount": "S-1-5-20",
   "synchronizationStatus": 0,
   "masterSiteAddress": "",
   "version": "21.2.0.1",
   "relations": {
    "self": {
     "type": "sites",
     "id": "2d12465c-3485-4ca8-a9fb-86a79de1a82f"
    }
   }
  }
 ]
}
```

Configuration

API Gateway configuration files

API Gateway configuration files are located in the installation location, by default `%ProgramFiles%\MOBOTIX\HUB API Gateway`.

These configuration files are relevant for the API Gateway:

- `GatewayConfig.json`: Reverse proxy (routing), WebRTC
- `appsettings.json`: Log levels
- `nlog.config`: Log layout, log targets, etc.

Reverse proxy

The reverse proxy (routing) functionality of the API Gateway is implemented using **YARP**.

> ⚠ Do not edit the YARP configuration manually. The configuration is created by the product installer and maintained by the Server Configurator.

This part of `GatewayConfig.json` is related to reverse proxy functionality:

```
  "ReverseProxy": {
    "Routes": {
      "well-known": {
        "ClusterId": "managementserver",
        "Match": {
          "Path": "/.well-known/{**remainder}"
        },
        "Transforms": [
          {
            "PathPattern": "/ManagementServer/.well-known/{**remainder}"
          }
        ]
      },
      "rest-api": {
        "ClusterId": "managementserver",
        "Match": {
          "Path": "/rest/v1/{**remainder}"
        },
        "Transforms": [
          {
            "PathPattern": "/ManagementServer/Rest/{**remainder}"
          }
        ]
      }
    },
    "Clusters": {
      "managementserver": {
        "Destinations": {
          "hostname": {
            "Address": "https://test-01.example.com/"
          }
        }
      }
    }
  }
```

For more information about YARP, please refer to YARP: Yet Another Reverse Proxy. [1]

WebRTC

WebRTC is a peer-to-peer protocol for streaming data, for example video.

WebRTC support is a pre-release in XProtect 2022 R3, enabled by setting this registry key:

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\VideoOS\Server\Gateway]
"EnableWebRTC"=dword:00000001
```

This part of `GatewayConfig.json` is related to WebRTC:

```
 "WebRTC": {
  "STUN": {
   "Hostname": {
    "Address": "stun:stun1.l.google.com:19302"
   }
  }
 }
```

For more information, please refer to the WebRTC sample documentation at https://github.com/milestonesys/mipsdk-samples-protocol/tree/main/WebRTC_JavaScript.

Logging

The API Gateway uses **NLog** for logging.

Logging is configured in two places:

[1] https://microsoft.github.io/reverse-proxy/index.html

- `appsettings.json`: Log levels
- `nlog.config`: Log layout, log targets, etc.

This part of `appsettings.json` is related to logging:

```
"Logging": {
 "LogLevel": {
  "Default": "Information",
  "Microsoft": "Warning",
  "Microsoft.Hosting.Lifetime": "Information",
  "Yarp": "Warning"
 }
}
```

To include YARP routing log messages, change the log level in `appsettings.json` for `Yarp` to e.g. `Information`.

This is the default NLog configuration file:

```
<?xml version="1.0" encoding="utf-8"?>
<nlog xmlns="http://www.nlog-project.org/schemas/NLog.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  autoReload="true"
  internalLogLevel="Warn"
  internalLogFile="internal-nlog.txt">
  <variable
    name="logDirectory"
     value="C:\ProgramData\MOBOTIX\ApiGateway\Logs" />
  <variable
    name="archiveDirectory"
   value="${var:logDirectory}\Archive" />
  <variable
    name="defaultLayout"
    value="${date:format=yyyy-MM-dd HH\:mm\:ss.fffzzz} [${threadid:padding=6}]
${level:uppercase=true:padding=-10} - ${message} ${exception:format=tostring}" />
  <targets>
    <target
      name="logfile"
      xsi:type="File"
      fileName="${var:logDirectory}\gateway.log"
      archiveFileName="${var:archiveDirectory}\gateway-{####}.log"
      archiveNumbering="Rolling"
      maxArchiveFiles="20"
      archiveEvery="Day"
      archiveAboveSize="1000000"
      archiveOldFileOnStartup="true"
      createDirs="true"
      layout="${var:defaultLayout}" />
  </targets>
  <rules>
    <logger name="*" minlevel="Debug" writeTo="logfile" />
  </rules>
</nlog>
```

For more information about NLog configuration, please refer to NLog Configuration options.[1]

Cross-Origin Resource Sharing (CORS)

The API Gateway can be configured to support Cross-Origin Resource Sharing (CORS). The following response headers are supported:

- `Access-Control-Allow-Origin`[2]
- `Access-Control-Allow-Headers`[3]
- `Access-Control-Allow-Methods`[4]

You enable CORS by setting `Enabled` to `true` and defining response headers and their values in `appsettings.json`:

```
"CORS": {
  "Enabled": true,
  "Access-Control-Allow-Origin": "yourdomain1.com,yourdomain2.com",
  "Access-Control-Allow-Headers": "Content-Type",
  "Access-Control-Allow-Methods": "*"
}
```

Only required response headers should be defined. Each response header can have multiple values, provided as a list of comma-separated values.

> ⚠ For security reasons, we recommend to always specify the `Access-Control-Allow-Origin` value with explicit origins. Never use wildcard (*) or null in your origin as this can put the security of your system at risk.

[1] https://nlog-project.org/config

[2] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Access-Control-Allow-Origin

[3] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Access-Control-Allow-Headers

[4] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Access-Control-Allow-Methods

MOBOTIX

Beyond Human Vision

MOBOTIX AG • Kaiserstrasse • D-67722 Langmeil • Tél. : +49 6302 9816 0 • sales@mobotix.com • www.mobotix.com