SDK Developer Guide—WebAPI(iOS)

Nymi Connected Worker Platform
v1.0
2023-09-21

Contents

Preface.....3

Nymi SDK Overview.....6

Nymi WebAPI Overview..... 9
SDK Package.....10
Sample Application..... 10

Creating NEAs with Nymi WebAPI..... 11

Types of Nymi Band Taps.....11
Authenticated Tap Workflow.....12
Invoking the Nymi Application Scheme from the NEA..... 16
Operations and Notifications for Web App Initialization (web-based iOS application only).....19
Subscribe_endpoint Operation.....19
Bluetooth Notifications..... 20
Subscribe_identity Operation..... 21
Operations and Notifications for Authenticated Tap.....22
Assert_identity Notifications..... 22

Troubleshooting.....24

Server Closes Web Connection..... 24

Preface

Nymi™ provides periodic revisions to the Nymi Connected Worker Platform. Therefore, some functionality that is described in this document might not apply to all currently supported Nymi products. The *Connected Worker Platform Release Notes* provide the most up to date information.

Purpose

This document is part of the Connected Worker Platform (CWP) documentation suite.

This document provides information about how to understand and develop Nymi-enabled Applications (NEA) by utilizing the functionality of the Nymi SDK, over a WebSocket connection that is managed by a web-based or other application. Separate guides are provided for Windows and iOS application development.

Audience

This guide provides information to Developers.

Revision history

The following table outlines the revision history for this document.

Table 1: Revision history

Version	Date	Revision history
1.0	September 21, 2023	First release of this document for the CWP 1.13.0 release.

Related documentation

- **Nymi Connected Worker Platform—Overview Guide**

  This document provides overview information about the Connected Worker Platform (CWP) solution, such as component overview, deployment options, and supporting documentation information.
- **Nymi Connected Worker Platform—Deployment Guide**

  This document provides the steps that are required to deploy the Connected Worker Platform solution.

  Separate guides are provided for authentication on iOS and Windows device.
- **Nymi Connected Worker Platform—Administration Guide**

This document provides information about how to use the NES Administrator Console to manage the Connected Worker Platform (CWP) system. This document describes how to set up, use and manage the Nymi Band™, and how to use the Nymi Band Application. This document also provides instructions on deploying the Nymi Band Application and Nymi Runtime components.

- **Nymi SDK for C Developer's Guide**

  This document provides information about how to develop Nymi-enabled Applications by using the Nymi API(NAPI).
- **Connected Worker Platform with Evidian Installation and Configuration Guide**

  The Nymi Connected Worker Platform with Evidian Guides provides information about installing the Evidian components and configuration options based on your deployment. Separate guides are provided for Wearable, RFID-only, and mixed Wearable and RFID-only deployments.
- **Nymi Connected Worker Platform—Troubleshooting Guide**

  This document provides information about how to troubleshoot issues and the error messages that you might experience with the NES Administrator Console, the Nymi Enterprise Server deployment, the Nymi Band, and the Nymi Band Application.
- **Nymi Connected Worker Platform with Evidian Troubleshooting Guide**

  This document provides overview information about how to troubleshoot issues that you might experience when using the Nymi solution with Evidian.
- **Nymi Connected Worker Platform—FIDO2 Deployment Guide**

  The Nymi Connected Worker Platform—FIDO2 Deployment Guide provides information about how to configure Connected Worker Platform and FIDO2 components to allow authenticated users to use the Nymi Band to perform authentication operations.
- **Connected Worker Platform with POMSnet Installation and Configuration Guide**

  The Nymi Connected Worker Platform—POMSnet Installation and Configuration Guides provides information about how to configure the Connected Worker Platform and POMSnet components to allow authenticated users to use the Nymi Band to perform authentication operations in POMSnet.
- **Nymi Band Regulatory Guide**

  This guide provides regulatory information for the Generation 3 (GEN3) Nymi Band.
- **Third-party Licenses**

  The Nymi Connected Worker Platform—Third Party Licenses Document contains information about open source applications that are used in Nymi product offerings.

How to get product help

If the Nymi software or hardware does not function as described in this document, you can submit a support ticket to Nymi, or email support@nymi.com

How to provide documentation feedback

Feedback helps Nymi to improve the accuracy, organization, and overall quality of the documentation suite. You can submit feedback by using support@nymi.com

Nymi SDK Overview

The Nymi SDK provides Developers with libraries, APIs, sample code and documentation to build a Nymi-enabled Application (NEA).

Overview of an iOS Environment

The following diagram provides a high-level overview of the components in a Connected Worker Platform environment that uses iOS devices to access web-based NEAs.

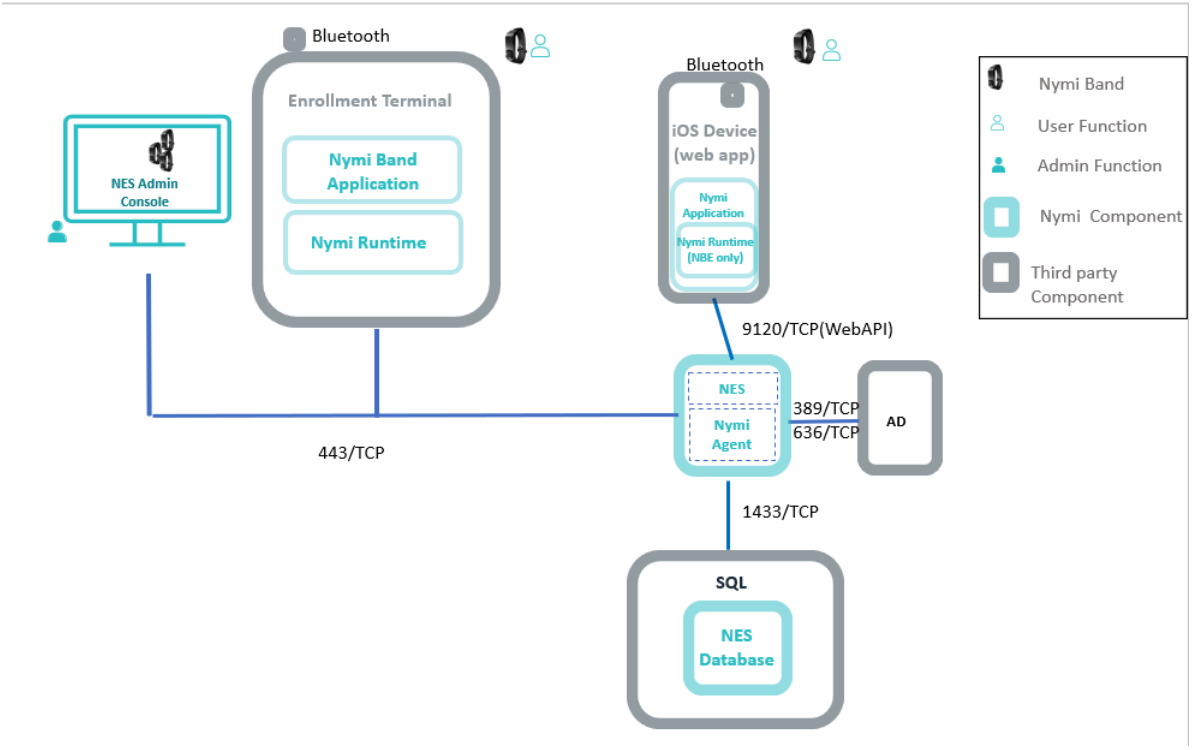

Figure 1: Connected Worker Platform Solution Components and Communication

Table 2: Connected Worker Platform Solution Components

Component	Description
Enrollment Terminal	Windows 10 endpoint that users access to enroll their Nymi Band.

Component	Description
Nymi Band Application (NBA)	A Windows application that you install on the enrollment terminal and is used to enroll a new user and link them to their Nymi Band. The Nymi Band Application requires the Nymi Runtime application, which the Nymi Band Application automatically installs. The Nymi Band Application communicates with the Nymi Band through the Nymi-supplied Bluetooth adapter, which you plug into a USB port on the enrollment terminal.
Nymi Band	A wearable device that is activated by the assigned user's biometrics. An authenticated Nymi Band is Bluetooth Low Energy (BLE) and Near Field Communication (NFC)-enabled. See the Nymi Band section in this guide for more information.
Nymi-enabled Application	Developers can create corporate applications that integrate with Connected Worker Platform by using the WebAPI component in the Nymi API. These applications are called Nymi-enabled Applications (NEAs) and include Manufacturing Execution Systems (MES), Single Sign-On (SSO), and Human Machine Interface (HMI) applications.
iOS Device	An iPad endpoint that users use to: • Perform authentication tasks in a web-based Nymi-enabled Application(NEA). • Perform authentication tasks in a native iOS NEA.
Nymi Application	Required on the iOS devices to perform authentication tasks. Nymi Application is a Nymi-supplied native iOS application that: • Embeds the Nymi Bluetooth Endpoint application, which provides an interface between the native Bluetooth Adapter (BLE) and the Nymi Agent. • Detects an intent to perform an authentication task with a Nymi Band (a tap) and passes the request to the NEA.

Component	Description
Centralized Nymi Agent	Required for environments that use iOS devices. Provides BLE management, manages operations and message routing. Facilitates communication between NEAs and the Nymi Band, and maintains knowledge of the Nymi Band presence and authenticated states. The Nymi Agent is installed in a central location on a single machine or a cluster of two or more machines that is accessible to all user terminals, for example on the NES server. To enable Nymi WebAPI communications between the Nymi Agent and the Nymi Bluetooth Endpoint, you must configure a *nymi_agent.toml* file. The *Nymi Connected Worker Platform—Deployment Guide* provides more information about how to configure the *nymi_agent.toml* file.
Nymi Enterprise Server (NES)	• A management server and collection of services that provides the NES Administrator Console and coordinates communication between the Nymi Band and the customer identity ecosystem (Active Directory) to manage policies and certificates. Includes the following services: • Enrollment Service (ES)—Authenticates, validates, and authorizes certificate requests from requesters, such as the Nymi Band Application and NEAs. • Directory and Policy Services (DPS)—Maintains the NES database, which contains a list of Active Directory (AD) users and the Nymi Bands that are associated with each user. Provides IIS web services, which allows the NES Administrator Console access to the NES database. • Authentication Service (AS)—Provides authentication and authorization support for domain users and computers. AS uses adapters to interface with external directory and database systems, such as an AD adapter to interface with Active Directory.

Component	Description
SQL Server	Database that contains table that store information about NES configuration and Nymi Bands. For Proof of Concept (POC) and pre-production environments, you can use the Nymi-provided SQL Server Express software. For production environments Nymi recommends that you use SQL server.
Domain Controller (DC)	Windows server with Active Directory.

Nymi WebAPI Overview

Nymi WebAPI is an RFC-6455 compliant WebSocket. NEAs, such as web-based applications use a standard WebSocket client to access Nymi WebAPI.

The Nymi WebAPI:

- Allows developers to utilize the WebSocket functionality of the Nymi SDK in a web-based or native application. The Nymi WebAPI architecture is part of the Nymi SDK.
- Provides bi-directional communication using requests/responses over a persistent connection. All messages sent and received are encoded in JSON format.
- Supports the Microsoft Windows and Apple iOS platforms only
- Provides continuous communication using WebSocket connections between the Nymi Agent and Nymi-enabled Application (NEA) running either as a native application or inside of a web client.
- Communicates with Nymi Bands over a WebSocket client and the integrated Bluetooth device on iOS.

To enable Bluetooth support on the iOS device when you access a web-based Nymi-enabled Application(NEA), you must install the Nymi Application

To secure communication between Nymi Agent and WebAPI client applications, Nymi highly recommends that you enable TLS for the WebAPI interface.

When a user performs a Nymi Band tap, to complete an authentication or e-signature in WebAPI application, the Nymi Bluetooth Endpoint sends an intent event that represents the tap to the application through the interface of the Nymi Agent.

WebSocket Keepalive Message

Nymi implements keepalive messages according to the RFC-6455 WebSocket Protocol standard for bi-directional communication. Nymi sends a ping message every 30 seconds to the NEA and expects to receive a pong message response. The keepalive message indicates that the connection is still responsive.

Nymi-supported web browsers send a pong message in response to the ping control frame message. The pong control frame message ensures that the session is connected to the Nymi Bluetooth Endpoint. NEA supported web browsers do not require any additional configuration to support this functionality.

If you are using a native WebSocket client, additional implementation may be required.

Note: The WebSocket client, which is the NEA, disconnects from the Nymi Agent if there are no messages (including pings and pongs) sent or received for a period of 60 seconds.

SDK Package

The SDK package contains the following folders:

- *nymi-sdk/ios/readme.md*—Readme file.
- *....nymi_sdk/iOS/Apps/NymiApplication*—Installation file and configuration file for the Nymi Application.
- *..nymi_sdk/iOS/NBE_iOS_Framework*—Nymi Bluetooth Endpoint compiled as an iOS framework to integrate with the native iOS application.
- *..nymi_sdk/iOS/sampleApps/browserApp*—Web-based NEA sample application for iOS devices.
- *....nymi_sdk/iOS/sampleApps/NymiSDK_iOSSampleApp*—Contains the sample native iOS Nymi-enabled Applications(NEAs).

Sample Application

The Nymi SDK package includes a sample application that demonstrates some of the key functionality of the Nymi solution.

The sample application is a simple Javascript application that demonstrates all the basic functions that are supported by the API and allows a user to see both JSON request and response examples to help understand how the API works.

Sample Application for Nymi WebAPI

Nymi WebAPI (iOS) includes two sample applications:

- Nymi Test App—A sample application that demonstrates to iOS developers how to integrate the SDK into their native iOS application, which is located within the package at: *nymi-sdk/ios/sampleApps/NymiTestApp*.
- Nymi Sample Browser App—A sample browser application that demonstrates to iOS developers how to integrate the SDK into their web-based applications and interact with the Nymi Application to support the completion of e-signatures on an iOS device, which is located within the package at: *nymi-sdk/iOS/sampleApps/browserApp*.

Creating NEAs with Nymi WebAPI

Customer and partner developers can use the Nymi WebAPI to develop Nymi-enabled Application (NEAs) for iOS. The API is based on JSON messages that are exchanged with the server over a websocket connection. This chapter provides information about the supported operations.

To support communications between web-based NEAs and the Nymi Band, the environment requires an NES server and a centralized Nymi Agent that is accessible from the iOS devices. Additionally, you must install the Nymi Application on each iOS device. *Nymi Connected Worker Platform—Deployment Guide* provides more information.

Note: In this document, the use of device refers to the Nymi Band.

The Nymi Band provides authentication information about a user to applications. An application can use this information on a point-in-time basis (for simple authentication) or continuously (for both authentication and de-authentication).

Types of Nymi Band Taps

To perform an authentication task, a Nymi Band user taps their authenticated Nymi Band on the Bluetooth adapter (BLE tap) that is connected to a user terminal.

The Nymi SDK allows an Nymi-enabled Application (NEA) to authenticate a user. A user provides their authentication intention (intent) when they perform a Nymi Band tap.

When a user performs a BLE tap, NES authenticates the Nymi Band by verifying cryptographic information that the Nymi Band transmits through the BLE tap. NES does not need to establish a Bluetooth connection to perform the cryptographic operation with the Nymi Band.

Table 3: Design Options for Nymi Band Taps

Option	Description	Benefits and Nymi Recommended Use Case
Authenticated Tap	When a user performs a BLE tap, NES authenticates the Nymi Band by verifying cryptographic information that the Nymi Band transmits through the BLE tap. NES does not need to establish a Bluetooth connection to perform the cryptographic operation with the Nymi Band.	Authenticated Tap offers very good security and offers a very fast response time. Nymi recommends this option when the user terminal establishes Bluetooth connections slowly, for example, when the Nymi Band user taps on the Bluetooth reader of an iOS device.

Option	Description	Benefits and Nymi Recommended Use Case
Tap and Lookup	When a user performs a tap, the NEA performs a *lookup* operation to identity the Nymi Band user. The Nymi Band and the Nymi SDK do not exchange cryptographic information, however; the Nymi Band still needs to have authenticated the user by using their fingerprint or corporate credentials authentication.	This design is a legacy option. Nymi recommends that you update your NEA to use Authenticated Tap.

Authenticated Tap Workflow

When a Nymi Band user taps and the Nymi-enabled Application(NEA) handles Nymi Band taps with an Authenticated Tap design, a series of events occur that result in a notification.

The notification indicates to the NEA that:

- A user wearing an authenticated Nymi Band wants to perform an authentication task.
- NES has authenticated the Nymi Band over Bluetooth.

Web-based and native iOS NEAs rely on the Nymi Application to run, monitor, and communicate with the NBE_iOS_Framework. The Nymi Application uses the NBE_iOS_Framework to handle Nymi Band taps and communicate with the centralize Nymi Agent.

The Authenticated Tap workflow and the behaviour of the Nymi Application differs for a Nymi Band tap in a native iOS NEA and a Nymi Band tap web-based NEA, as summarized below and detailed in the following sections.

Table 4: Authenticated tap workflow differences for web-based and native iOS NEAs

Web-based NEA	Native iOS NEA
NEA performs operations to initialize the web application.	Nymi Application performs operations to initialize the web application.
Custom URL scheme (nyminbe) that invokes the Nymi Application does not define a return URL.	Custom URL scheme (nyminbe) that invokes the Nymi Application defines a return URL.
After a Nymi Band tap, the Nymi Application prompts the user to change the focus back to the NEA.	After a Nymi Band tap, the Nymi Application automatically closes and focus returns to the NEA.

Authenticated Tap Workflow - Web-based iOS Application

The following figure summarizes the authentication workflow that the solution follows when a user performs a BLE tap in a web-based iOS Nymi-enabled Application(NEA) to complete an authentications task, and the Nymi Band tap handling design is Authenticated Tap.

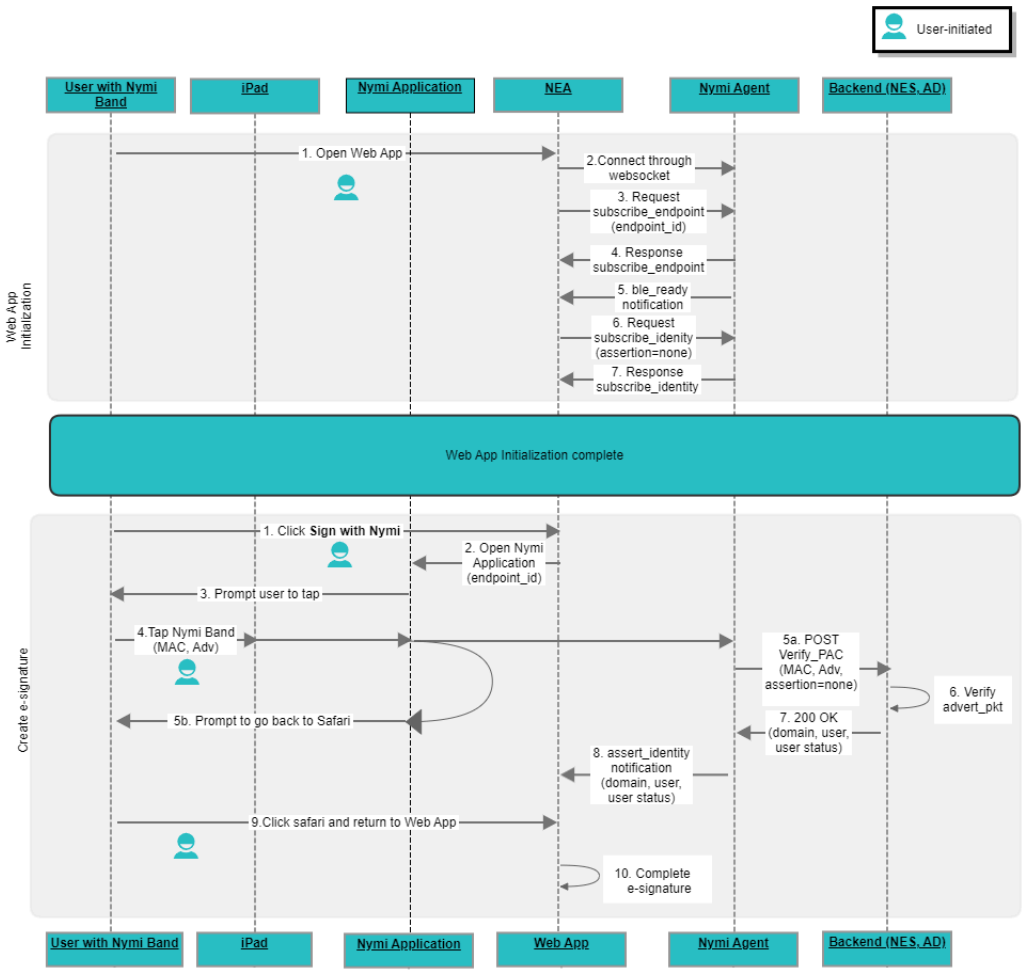

Figure 2: Authentication Workflow for iOS Devices with Web-based iOS Application

The authentication workflow includes two distinct phases. Each phase includes user-initiated and application-initiated actions. For a web-based NEA, the NEA initializes the web application.

Phase 1—Initialize NEA

This phase occurs each time a user connects to the NEA and results in the establishment of connectivity between the NEA and the Nymi components.

1. User opens the NEA from a browser on an iOS device.
2. NEA establishes a WebSocket connection to the Nymi Agent.
3. NEA sends a *subscribe_endpoint* request to the Nymi Agent, to establish a connection with the endpoint (iOS device). The request includes an *endpoint_id*, which uniquely identifies the iOS device. An NEA can only subscribe to one endpoint at any given time.

4. Nymi Agent responds to the request. Nymi Agent returns a response and status code.
5. If the *subscribe_endpoint* response indicates success, the Nymi Agent establishes a connection with the endpoint. When the bluetooth adapter on the endpoint is in a ready state, the Nymi Agent sends a *ble_ready* notification to the NEA.
6. NEA sends a *subscribe_identity* request to the Nymi Agent. The *subscribe_identity* operation instructs Nymi WebAPI to use the VerifyPAC API to verify the identity of the user that performs the Nymi Band tap
7. Nymi Agent returns a *subscribe_identity* response and status code to the NEA.

Phase 2—Create E-signature

This phase occurs each time a user performs an e-signature with the Nymi Band and results in the completion of an e-signature with the tap of a Nymi Band.

1. From a window within the NEA that requires an e-signature, the user clicks the button on the screen to use the Nymi Band.
2. The NEA sends a custom URL, which includes the *endpoint_id* to the Nymi Application.
3. The Nymi Application appears on the screen and prompts the user to tap their Nymi Band on the iOS device.
4. User taps the Nymi Band on the iOS device near the Bluetooth antenna. The Nymi Application detects the tap and send the MAC address and advertising packet of the Nymi Band to the Nymi Agent.
5. The following events occur:
   a. Nymi Agent submits a VerifyPAC API request to NES, to verify the authenticity of Nymi Band tap.
   b. The Nymi Application prompts the user to return to Safari.
6. NES verifies the packet and contacts Active Directory to confirm the user credentials, and returns the response to the Nymi Agent.
7. NES returns the response to the Nymi Agent.
8. Nymi Agent sends an *assert_identity* notification to the NEA. If the NES verification succeeded, the notification contains the domain, username, and AD status of the Nymi Band user.
9. User clicks Safari in the navigation bar, the Nymi Application closes and focus changes to the NEA.
10. The NEA reviews the *assert_identity* notification and based on the information, completes the e-signature or does not complete the e-signature.

Authenticated Tap Workflow—Native iOS Application

The following figure summarizes the authentication workflow that the solution follows when a user performs a BLE tap in a native iOS NEA to complete an authentication task, and the Nymi Band tap handling design is Authenticated Tap.

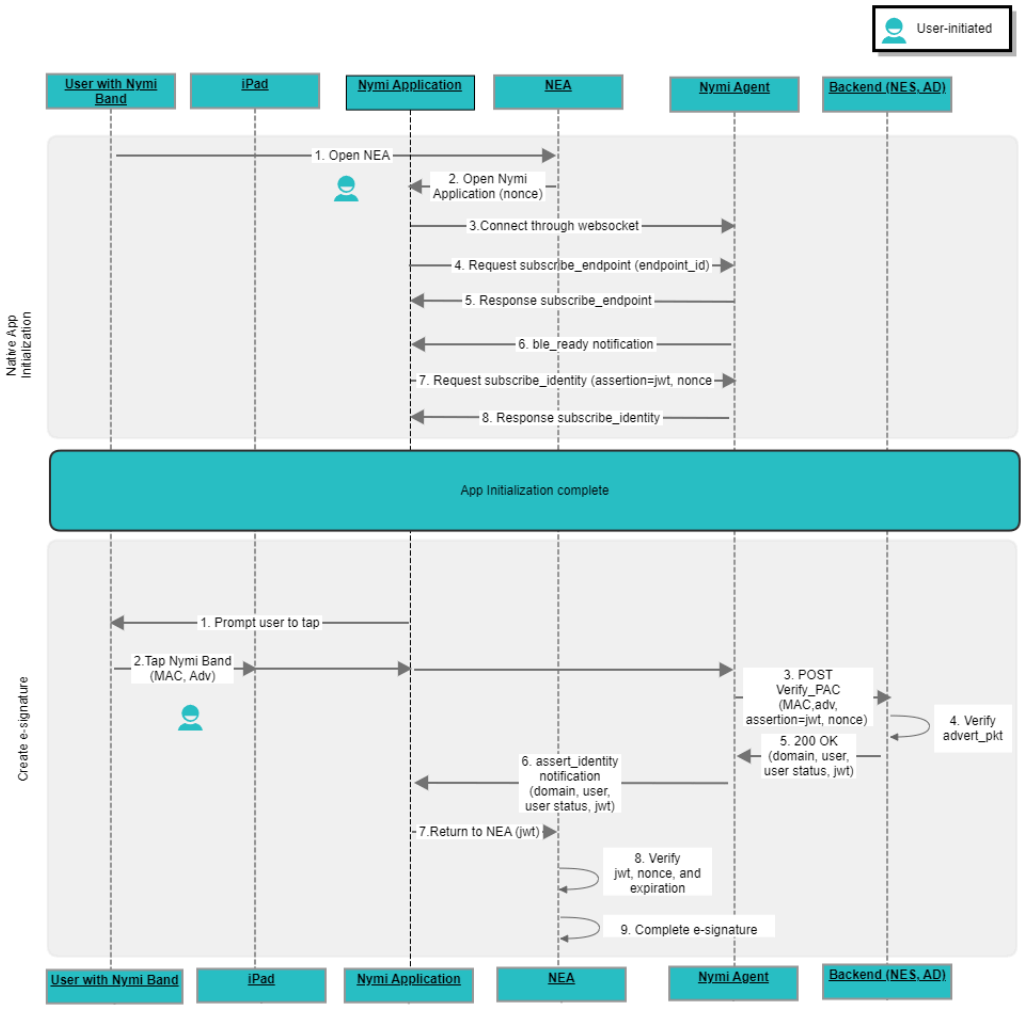

Figure 3: Authenticated Tap Workflow—Native Application

The Authenticated Tap workflow includes two distinct phases. Each phase includes user-initiated and application-initiated actions.

Phase 1—Initialize Web Application

This phase occurs each time a user connects to the NEA and results in the establishment of connectivity between the NEA and the Nymi components.

1. User opens the NEA on an iOS device.
2. The NEA sends a custom URL scheme, with a nonce to the operating system. The operating system starts the Nymi Application.
3. Nymi Application establishes a WebSocket connection to the Nymi Agent.
4. Nymi Application sends a *subscribe_endpoint* request to the Nymi Agent, to establish a connection with the endpoint (iOS device). The request includes an *endpoint_id*, which

uniquely identities the iOS device. Nymi Application can only subscribe an NEA to one endpoint at any given time.

5. Nymi Agent returns a *subscribe_endpoint* response and status code to the Nymi Application.
6. If the *subscribe_endpoint* response indicates success, the Nymi Agent establishes a connection with the endpoint. When the bluetooth adapter on the endpoint is in a ready state, the Nymi Agent sends a *ble_ready* notification to the NEA.
7. Nymi Application sends a *subscribe_identity* request to the Nymi Agent with a JSON Web Token (JWT) and a nonce. The *subscribe_identity* operation instructs Nymi WebAPI to use the VerifyPAC API to verify the identity of the user that performs the Nymi Band tap.
8. Nymi Agent returns a *subscribe_identity* response and status code to the Nymi Application.

Phase 2—Create E-signature

This phase occurs each time a user performs an e-signature with the Nymi Band and results in the completion of an e-signature with the tap of a Nymi Band.

1. From a window within the NEA that requires an e-signature, a screen appears that prompts the user to perform the BLE tap.
2. User taps the Nymi Band on the iOS device near the Bluetooth antenna. The Nymi Application detects the tap and send the MAC address and advertising packet of the Nymi Band to the Nymi Agent.
3. Nymi Agent submits a VerifyPAC API request to NES, to verify the authenticity of the Nymi Band tap.
4. NES verifies the packet and contacts Active Directory to confirm the user credentials.
5. NES returns a response to the Nymi Agent.
6. Nymi Agent sends an *assert_identity* notification to the NEA. If the NES verification succeeded, the notification contains the domain, username, and AD status of the Nymi Band user, as well as the JWT.
7. Nymi Application closes and returns control to the NEA. The iOS screen focus changes back to the NEA.
8. NEA verifies contents of the payload in the *assert_identity* notification.
9. Based on the information in the notification, the NEA completes the e-signature or does not complete the e-signature.

Invoking the Nymi Application Scheme from the NEA

Perform the following high level steps to invoke the Nymi Application from the NEA.

Before you begin

For web-based NEAs only:

- Setup a web server.

- Setup a web page.
- Enable websocket support.
- Implement or use an existing method in your web application that assigns a unique and persistent identifier to each iOS device. Your application will use this identifier as *endpoint_id*. Nymi WebAPI uses *endpoint_id* to create an association between the websocket sessions and the Nymi Application instances that connect to the system. You will pass the *endpoint_id* to the *subscribe* operation, as explained later in this guide.

  Note: This document does not provide implementation details about how to generate, store, and retrieve the unique value.

About this task

The Nymi SDK package contains *Readme.md* files that provides detailed information about how to create an NEA that invokes the Nymi Application scheme and how to modify the Nymi-supplied sample applications.

Procedure

1. If the Nymi Application is running on an iOS device, close the Nymi Application.
2. Include in your NEA a mechanism, such as a button, that invokes the Nymi Application through the use of the custom URL scheme named *nyminbe* with the following parameters:
   - *endpoint_id*—A string value that defines a unique identifier for the iOS device. Do not include spaces.
   - *nymi_agent_host*—A string value that defines the Nymi Agent host name or IP address, which the NEA passes to the Nymi Application. Do not include spaces.
   - *nymi_nes_url*—A string value that defines URL of the Nymi Enterprise Server(NES). Do not include spaces.
   - *nymi_agent_port*—A number value between 1 and 65535 that defines the port number on which the Nymi Application can communicate with the Nymi Agent. The default port is 9120.The value must match the value that is defined in the configuration of the Nymi Agent.
   - *integrator_name*—A string value that defines the name of the web integrator that appears in the Nymi Application.

     Consider the following when you specify the *integrator_name*:

     - Do not include spaces.
     - Do not include any of the following characters: *% < > ? : " ^ @ # { } | \ [ ]*

     The following figure provides an example of the Nymi Application when the integrator name is *nymi*.

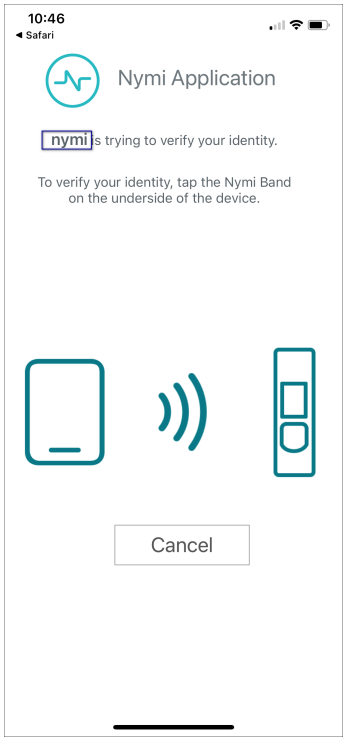

Figure 4: Integrator Name

- *webapi_url*—Specify with Native iOS applications only. A string value that defines the URL of the Nymi Agent websocket.
- *nonce*—Specify with Native iOS applications only. A nonce is a random string value that signs an assertion, to prevent replay attacks.
- *return_url*—Specify with Native iOS applications only. A string value that defines the URL of the partner application, which the Nymi Application uses to return the user back to the NEA. Do not include spaces.

Sample format of the call to invoke the schema:

- Native iOS application:

```
var target_url = nyminbe://example_integrator_name?
nymi_agent_host=agent.mydomain.com&nymi_agent_port=9120
&endpoint_id=example_unique_id&webapi_url=example_webapi_url&nonce=example_nonce_string
&return_url=example_return_url
```

- Web-based application:

```
var target_url = nyminbe://example_integrator_name?
nymi_agent_host=agent.mydomain.com&nymi_agent_port=9120&endpoint_id=example_unique_id
```

Operations and Notifications for Web App Initialization (web-based iOS application only)

This section summarizes the operations and notifications that initialize the Web App and allow the Nymi-enabled Application(NEA) to handle Nymi Band taps.

Note: This section does not apply to native iOS applications because the Nymi Application performs the operations that initialize the web application.

Subscribe_endpoint Operation

The `subscribe_endpoint` operation allows an NEA to change the Nymi Bluetooth Endpoint to which it is subscribed.

subscribe_endpoint request operations appear in the following format:

```
{
  "operation": "subscribe_endpoint",
  "exchange":"exchange_value",
  "payload": {
     "endpoint_id": "mobile_endpoint_id"
  }
}
```

where:

- *operation* is *subscribe_endpoint*.
- *exchange* is any value and is used to match the response to the request.
- *endpoint_id* is a unique identifier that an NEA assigns to every iOS device. The NEA passes the same value to the Nymi Application, when the NEA invokes the Nymi Application.

The *subscribe_endpoint* operation returns a status code only, no errors are returned.

```
{
  “operation”: “subscribe_endpoint”,
  "exchange":"exchange_value",
  “payload”: {}
  “status”: 0,
  “error”: {}
}
```

You can only subscribe an NEA to one endpoint at any given time. When you request the *subscribe_endpoint* operation, the NEA is automatically unsubscribed from the previously

subscribed endpoint. Any Nymi Bands that were present on the previously subscribed endpoint become absent, and the NEA receives corresponding presence update notifications. The NEA will then receive a Bluetooth status notification. If the requested Nymi Bluetooth Endpoint has connected successfully and is in a ready state, the NEA will receive a *ble_ready* notification, followed by presence update notifications for any present bands on that endpoint. Otherwise, the NEA will receive an error message. See *Bluetooth Notifications* for more information about possible error messages.

Note: The NEA remains subscribed to the requested *endpoint_id* even if it is not able to connect to that Nymi Bluetooth Endpoint. If the Nymi Bluetooth Endpoint becomes ready at a later time (for example, when a user turns on the user terminal), then NEA receives a ble_ready*endpoint_id* message at that time.

Bluetooth Notifications

Nymi Bluetooth Endpoint is a client service that communicates with the Bluetooth Adapter. Bluetooth notifications for Bluetooth Adapter status are non-transactional.

The Bluetooth Adapter communicates to the Nymi Band. Each time that a Bluetooth Adapter becomes available, the *update* function retrieves a notification in the following format.

```
{
  "operation": "ble_ready",
  "exchange": null,
  "status": 0,
  "payload": {},
  "error ": {}
}
```

If a Bluetooth Adapter becomes unavailable, the *update* function retrieves an error notification in the following format.

```
{
  "operation": "error",
  "exchange": null,
  "payload": {},
  "status": "error_code",
  "error": {
      "error_description":"error_description>",
      "error_specifics":"error_specifics"
  }
}
```

where *error_code* is one of the following values: 5000, 5010, 5100.

For more information about error codes, see *Error Handling*.

Subscribe_identity Operation

The `subscribe_identity` operation enables an NEA to process Bluetooth Tap notifications through the VerifyPAC API on NES to confirm the identity of the Nymi Band user.

subscribe_identity request operations appear in the following format:

```
{
  "operation": "subscribe_identity",
  "exchange":"exchange_value",
  "payload": {
    "assertion": "none"


  }
}
```

where:

- *operation* is *subscribe_identity*.
- *exchange* is any value and is used to match the response to the request.
- payload contains an *assertion* parameter that is set to none.

The *subscribe_identity* operation returns response.

The following table summarizes the status codes can appear:

Table 5: Subscribe_identity status codes

Status Code	Description
0	Connection was successfully subscribed in the WebAPI to the *subscribe_identity* workflow.
1000	*subscribe_identity* contains an invalid JSON string, or the payload contains an unacceptable assertion value.

The response appears in the following format:

```
{
  “operation”: “subscribe_identity”,
  "exchange":"exchange_value",
  “payload”: {}
  “status”: 0|1000,
  “error”: {}
}
```

When the verification succeeds, *subscribe_identity* enables the *assert_identity* notification, as described in the following section.

Operations and Notifications for Authenticated Tap

Use the `subscribe_identity` operation, which provides the `assert_identity` notification to verify the Nymi Band user that taps on a Bluetooth Adapter.

Assert_identity Notifications

When the user taps their Nymi Band on the Bluetooth adapter and VerifyPAC successfully verifies the user, WebAPI sends an *assert_identity* notification to the subscribed client connection with a status of 0.

The Nymi-enabled Application(NEA) retrieves a notification in the following format.

```
{
  "operation": "assert_identity",
  "exchange": "exchange",
  "status": 0,
  "payload": {
     "User": "username",
     "Domain": "domain_name",
     "UserStatus": "user_status",
     "Jwt": "jwt_token"
  },
  "error ": {}
}
```

where:

- *operation* is *assert_identity*.
- *exchange* is any value and is used to match the response to the request.
- payload displays the username and domain for the Nymi Band user, and optionally the status of the user in Active Directory (AD) and the NES-issued JWT token.

The AD status for a user appears in the response when user status check is enabled in NES. The following table summarizes the possible user statuses.

- **Table 6: AD user statuses**

User Status	Definition
Active	User account is enabled.
NotExist	User account was deleted from AD.
Inactive	User account is disabled.
Active | Locked	User account is locked. This status can appear with Active and Password Expired.

User Status	Definition
Active | PasswordExpired	User account has an expired password. This status can appear with Active and Locked.

By default, NES is not configured to perform user status checks in AD. Contact the NES Administrator to enable AD user status checking, and optionally the checking interval in the NES Administrator Console.

If the VerifyPAC fails, the *update* function retrieves an error notification in the following format.

```
{
  "operation": "error",
  "exchange": null,
  "payload": {},
  "status": "error_code",
  "error": {
      "error_description":"error_description",
      "error_specifics":"error_specifics"
   }
 }
```

The following table summarizes the error code and error description that VerifyPAC might return to the NEA.

Table 7: VerifyPAC Errors

Error Code	Error Description
7001	*Authentication Error. Log in to the Nymi Band Application to update Nymi Band settings.*. This error appears when battery level became very low before the user charged the Nymi Band, and the real-time clock cannot keep time. To resolve this issue, the Nymi Band user must log into the Nymi Band Application while they wear their authenticated Nymi Band to reset the real time clock.
7002	*Authentication Error. Cannot find the Nymi Band in the Nymi Enterprise Server. Contact your administrator.* This error appears when the user enrolled their Nymi Band to a different NES.
7003	*Authentication Error. Key cannot be found on the Nymi Enterprise Server. Contact your administrator.* This error appears when the advertising key does not exist for the Nymi Band.
7004	*Authentication Error. Please retry.*. This error appears when the VerifyPAC operation cannot verify the authenticity of the presence authentication code (PAC). Provide the user with an message similar to the following: *Authentication Error. Retry.*
7005	*Communication error. Contact your administrator.* This error appears when there is an issue with the VerifyPAC request payload.

Note: The subscription exists until the WebSocket connection/session ends for the client.

Troubleshooting

Nymi API writes information to log files that allow you to monitor and troubleshoot the NEA.

For additional assistance, visit the Support page on the Nymi website, or contact your Nymi Solution Consultant.

The following table summarizes the log files that are available for troubleshooting.

Table 8: Log file locations

Component	Log location	Files
Nymi Agent	*C:\Nymi\NymiAgent*	*nymi_agent.log*
Nymi Application	Open the Nymi Application, and then select **Logs**.	n/a

Server Closes Web Connection

When a Nymi Band user performs an authentication task, such as an e-signoff, the web server does not receive the `intent` message.

The following message appears in the developer tools window of the web browser:

> WebSocket connection to '*nymi_agent_server* failed. The operation couldn't be completed. (kNWErrorDomainPOSIX error 53 - Software caused a connection abort)

The *nymi_agent.log* file displays the following error:

```
DISCONNECT "nymi_agent_server:63839": {:error, :closed}
```

Cause

By default Safari closes the websocket connection after 2 seconds of inactivity. If the user does not complete the Nymi Band tap within the two seconds time period in the Nymi Application, Safari closes the web connection.

Resolution

Extend the time that the connection remains opened to 1 minute.

From the **Settings**, navigate to **Safari > Advanced > Experimental Features** and disable the **NSURL Session WebSocket** option.

Nymi Inc. (Nymi) believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

The information in this document is provided as-is and Nymi makes no representations or warranties of any kind. This document does not provide you with any legal rights to any intellectual property in any Nymi product. You may copy and use this document for your referential purposes.

This software or hardware is developed for general use in a variety of industries and Nymi assumes no liability as a result of their use or application. Nymi, Nymi Band, and other trademarks are the property of Nymi Inc. Other trademarks may be the property of their respective owners.

Published in Canada.
Nymi Inc.
Toronto, Ontario
www.nymi.com