

# ERS APIを使用したISEネットワークデバイスの作成

## 内容

---

[はじめに](#)

[前提条件](#)

[要件](#)

[使用するコンポーネント](#)

[設定](#)

[ERSの有効化 \(ポート9060\)](#)

[ERS管理者の作成](#)

[Postmanの設定](#)

[ISE SDKと基本的なポストマン認可](#)

[XMLを使用したNADの作成](#)

[JSONを使用したNADの作成](#)

[確認](#)

[トラブルシューティング](#)

---

## はじめに

このドキュメントでは、RESTクライアントとしてPostManを使用して、ERS API経由でISE上にネットワークアクセスデバイス(NAD)を作成するプロセスについて説明します。

## 前提条件

### 要件

次の項目に関する知識があることが推奨されます。

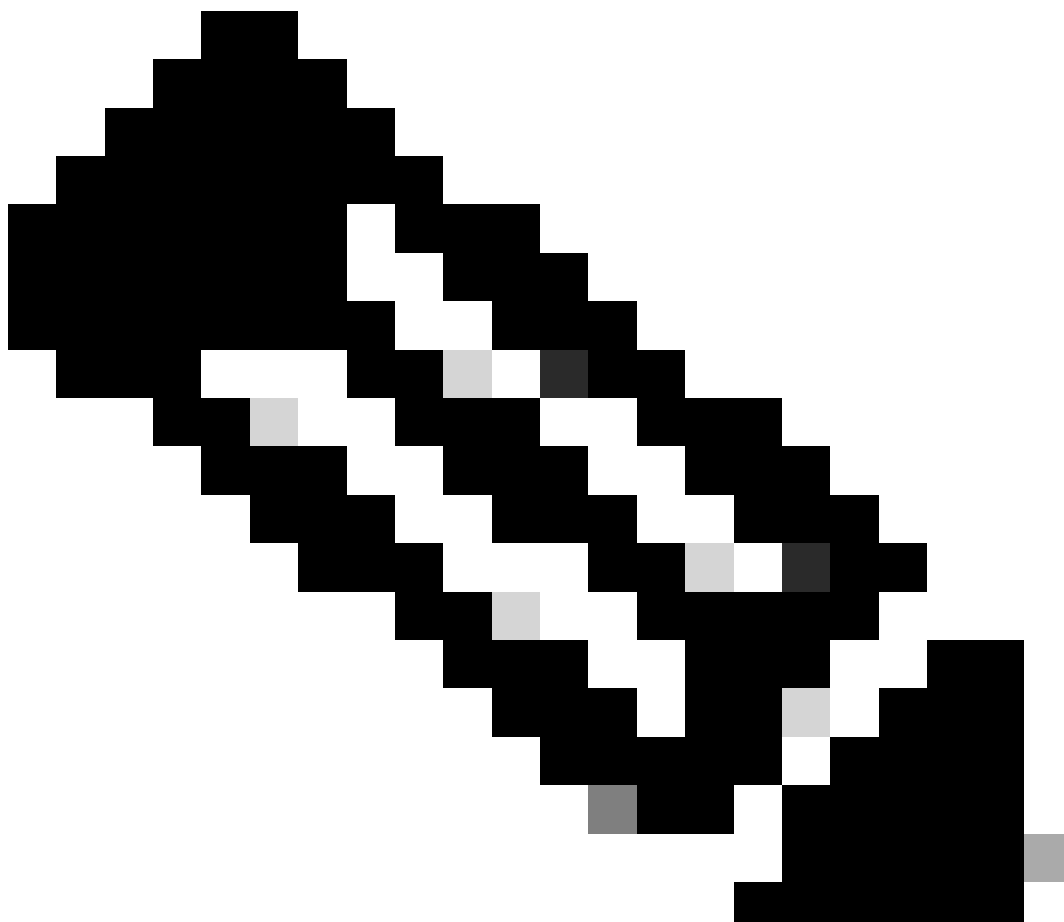
- ISE(Identity Services Engine)
- ERS ( 外部RESTfulサービス )
- RESTクライアントには、Postman、RESTED、Insomniaなどがあります。

### 使用するコンポーネント

このドキュメントの情報は、次のソフトウェアのバージョンに基づいています。

- Cisco ISE(Identity Services Engine)3.1パッチ6
- Postman RESTクライアントv10.17.4

---



注：手順は、他のISEバージョンとRESTクライアントで類似または同一です。これらの手順は、特に記載のない限り、すべての2.xおよび3.x ISEソフトウェアリリースで使用できます。

---

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

## 設定

### ERSの有効化（ポート9060）

ERS APIは、ポート443およびポート9060で動作するHTTPS専用のREST APIです。ポート9060はデフォルトで閉じられているため、最初に関する必要があります。このポートにアクセスしようとしているクライアントが最初にERSをイネーブルにしていなかった場合、サーバからのタイム

アウトが表示されます。したがって、最初の要件は、Cisco ISE管理UIからERSを有効にすることです。

Administration > Settings > API Settingsの順に移動し、ERS (読み取り/書き込み) トグルボタンを有効にします。

The screenshot shows the Cisco ISE Administration UI. The top navigation bar includes 'Administration - System' and various menu items like 'Deployment', 'Licensing', 'Certificates', 'Logging', 'Maintenance', 'Upgrade', 'Health Checks', 'Backup & Restore', 'Admin Access', and 'Settings'. The left sidebar lists various configuration categories, with 'API Settings' highlighted. The main content area is titled 'API Settings' and has three tabs: 'Overview', 'API Service Settings', and 'API Gateway Settings'. Under 'API Service Settings for Administration Node', there are two toggle switches: 'ERS (Read/Write)' which is turned on (indicated by a red arrow), and 'Open API (Read/Write)' which is turned off. Below this, there is a section for 'CSRF Check ( only for ERS Settings )' with two radio button options: 'Enable CSRF Check for Enhanced Security (Not compatible with pre ISE 2.3 Clients)' and 'Disable CSRF For ERS Request (compatible with ERS clients older than ISE 2.3)'. At the bottom right, there are 'Reset' and 'Save' buttons.



注:ERS APIはTLS 1.1とTLS 1.2をサポートしています。ERS APIは、Cisco ISE GUIの Security Settingsウィンドウ(Administration > System > Settings > Security Settings)で TLS 1.0を有効にしたかどうかに関係なく、TLS 1.0をサポートしません。[セキュリティの設定]ウィンドウでTLS 1.0を有効にすることは、EAPプロトコルのみに関連し、ERS APIには影響しません。

---

## ERS管理者の作成

Cisco ISE管理者を作成し、パスワードを割り当て、ユーザをERS Adminとして管理グループに追加します。残りの設定は空のままにしておくことができます。

Admin User

\* Name **ERS-USER** ←

Status **Enabled** ▼

Email   Include system alerts in emails

Expires

Hard Date

Inactive account never expires

---

Password

\* Password  ⓘ ←

\* Re-Enter Password  ⓘ

[Generate Password](#)

---

User Information

First Name

Last Name

---

Account Options

Description

Change password on next login

---

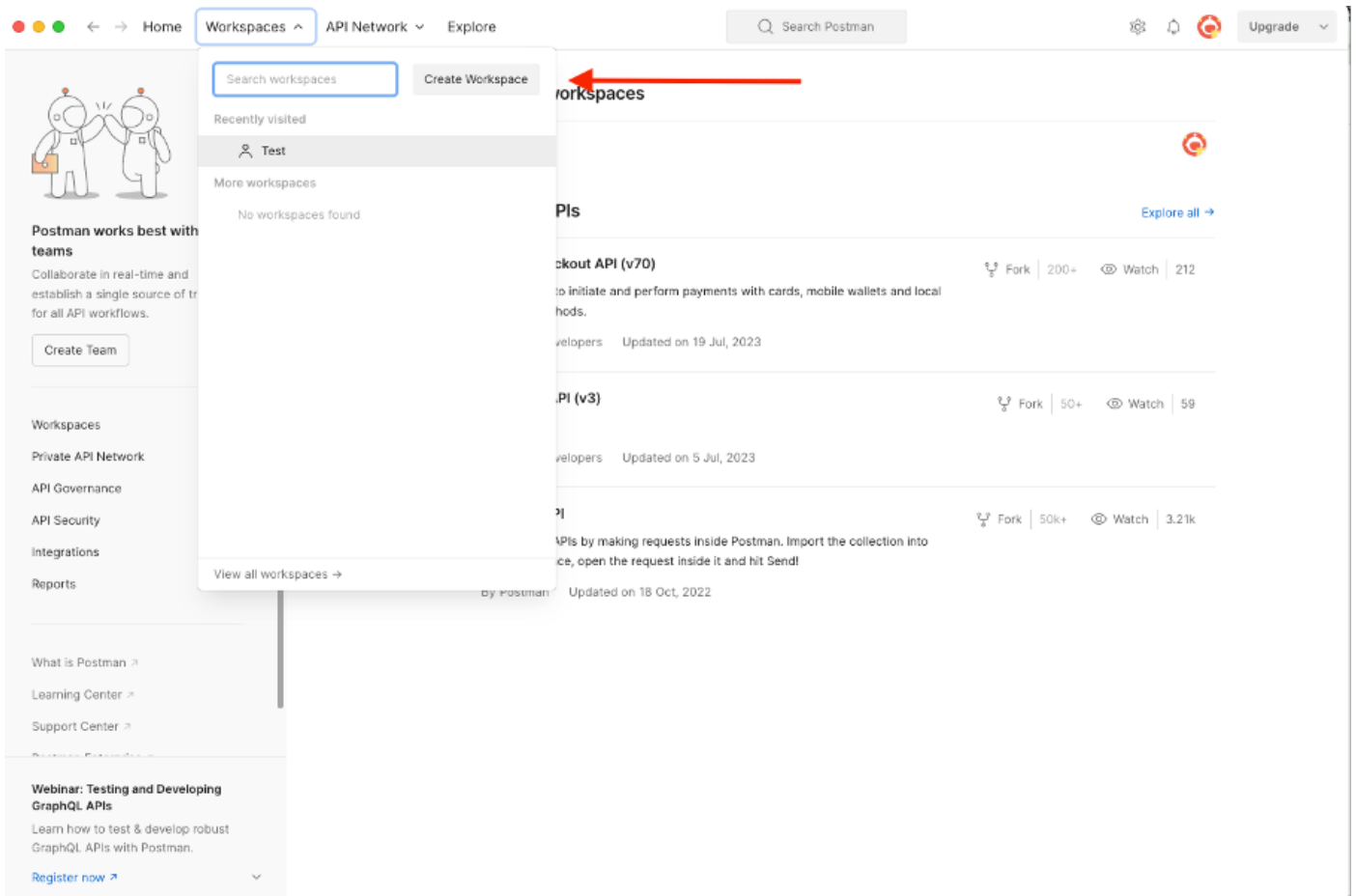
Admin Groups

ERS Admin ←

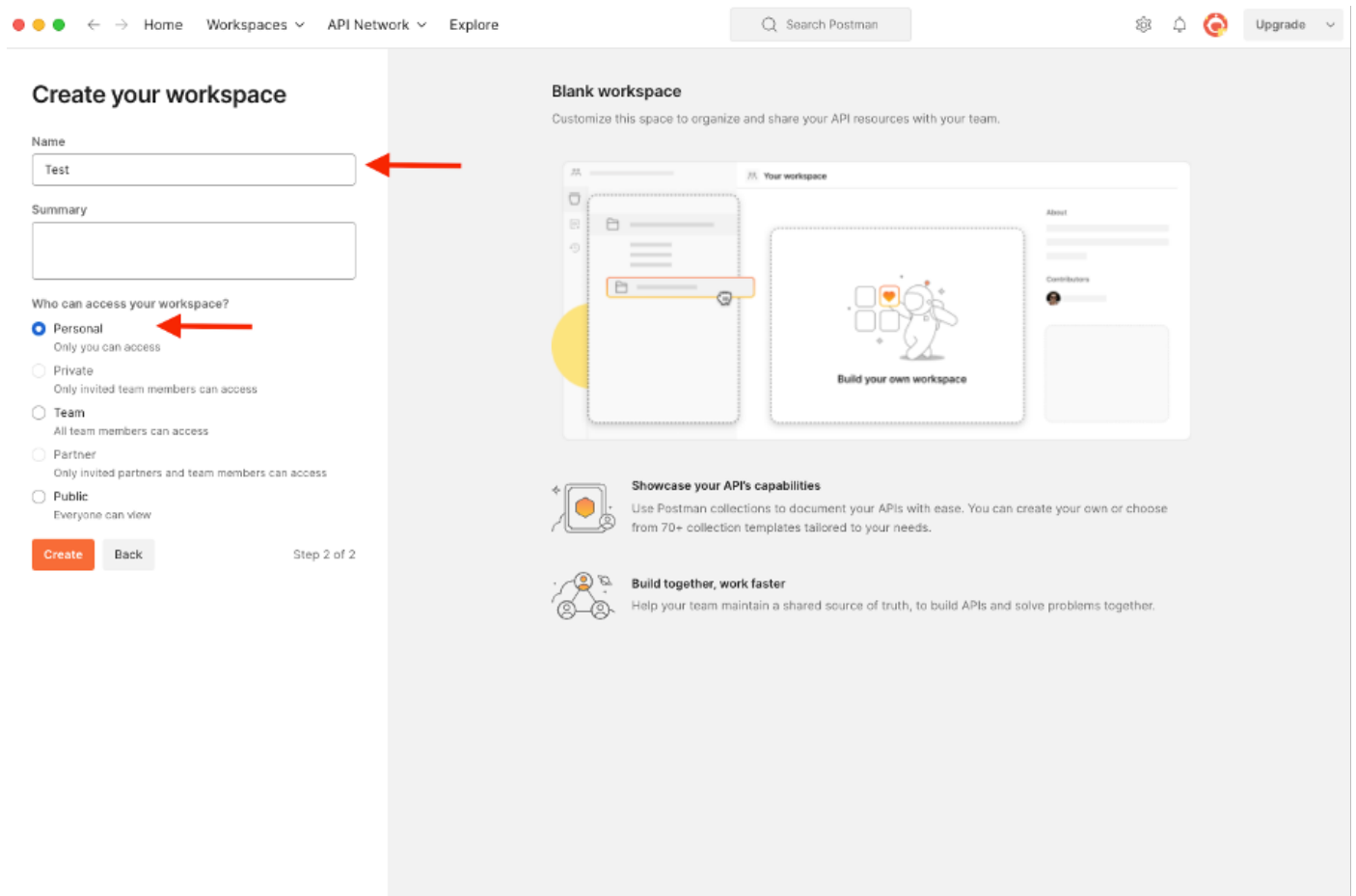
## Postmanの設定

Postmanのオンライン版をダウンロードまたは使用します。

1. ユーザを作成し、ワークスペースを作成します。これを行うには、「ワークスペース」タブにある「ワークスペースを作成」をクリックします。



2. 「ブランク・ワークスペース」を選択し、ワークスペースに名前を割り当てます。説明を追加して公開できます。この例では、Personalisが選択されています。



ワークスペースを作成したら、APIコールを設定できます。

## ISE SDKと基本的なポストマン認可

コールを設定するには、最初にISE ERS SDK (ソフトウェア開発者キット) にアクセスします。このツールは、ISEが実行できるAPIコールのリスト全体をコンパイルします。

1. <https://{{ise-ip}}/ers/sdk>に移動します。
2. ISE管理者クレデンシャルを使用してログインします。
3. API Documentationを展開します。
4. Network Deviceが表示されるまでスクロールダウンして、これをクリックします。
5. このオプションでは、ISEのネットワークデバイスに対して実行できるすべての操作を検索できます。Createを選択します。

External RESTful Services (ERS) Online SDK

Quick Reference

API Documentation

- Filter Policy
- Guest Location
- Guest Smpg Notification Configur
- Guest Ssid
- Guest Type
- Guest User
- Hotspot Portal
- IP To SCT Mapping
- IP To SCT Mapping Group
- ISE Service Information
- Identity Group
- Identity Sequence
- Internal User
- My Device Portal
- Native Supplicant Profile
- Network Device
- Network Device Group
- Node Details
- PSN Node Details with Radius Set
- Portal
- Portal Theme
- Profiler Profile
- Pull Deployment Info
- Pxgrid Node
- Pxgrid Settings
- Radius Server Sequence
- RestID Store
- SMS Server
- SXP Connections
- SXP Local Bindings
- SXP Vpns
- Security Groups
- Security Groups ACLs
- Security Groups to Virtual Netwo
- Self Registered Portal
- Sponsor Group
- Sponsor Group Member
- Sponsor Portal
- Sponsored Guest Portal
- Support Bundle Download

Network Device

- Overview
- Resource definition
- Revision History
- Update-By-Name
- Delete-By-Name
- Get-By-Name
- Get-By-Id
- Update
- Get-All
- Delete
- Create
- Get Version
- Bulk Request
- Monitor Bulk Status

Overview

Network Device API allows the client to add, delete, update, and search Network Devices. In this documentation, for each available API you will find the request syntax including the required headers and a response example of a successful flow. Please note that each API description shows weather the API is supported in bulk operation. The Bulk section is showing only 'create' bulk operation however, all other operation which are bulk supported can be used in same way.

Please note that these examples are not meant to be used as is because they have references to DB data. You should treat it as a basic template and edit it before sending to server.

Back to top

Resource definition

Attribute	Type	Required	Default value	Description
name	String	Yes		Resource name
id	String	No		Resource UUID, mandatory for update

Developer Resources

6. これで、任意のRestクライアント上でXMLまたはJSONを使用してAPIコールを実行するために必要な設定と、想定される応答例が表示されます。

Quick Reference

API Documentation

- Filter Policy
- Guest Location
- Guest Smpg Notification Configur
- Guest Ssid
- Guest Type
- Guest User
- Hotspot Portal
- IP To SCT Mapping
- IP To SCT Mapping Group
- ISE Service Information
- Identity Group
- Identity Sequence
- Internal User
- My Device Portal
- Native Supplicant Profile
- Network Device
- Network Device Group
- Node Details
- PSN Node Details with Radius Set
- Portal
- Portal Theme
- Profiler Profile
- Pull Deployment Info
- Pxgrid Node
- Pxgrid Settings
- Radius Server Sequence
- RestID Store
- SMS Server
- SXP Connections
- SXP Local Bindings
- SXP Vpns
- Security Groups
- Security Groups ACLs
- Security Groups to Virtual Netwo
- Self Registered Portal
- Sponsor Group
- Sponsor Group Member
- Sponsor Portal
- Sponsored Guest Portal
- Support Bundle Download

Network Device

Create

Request:

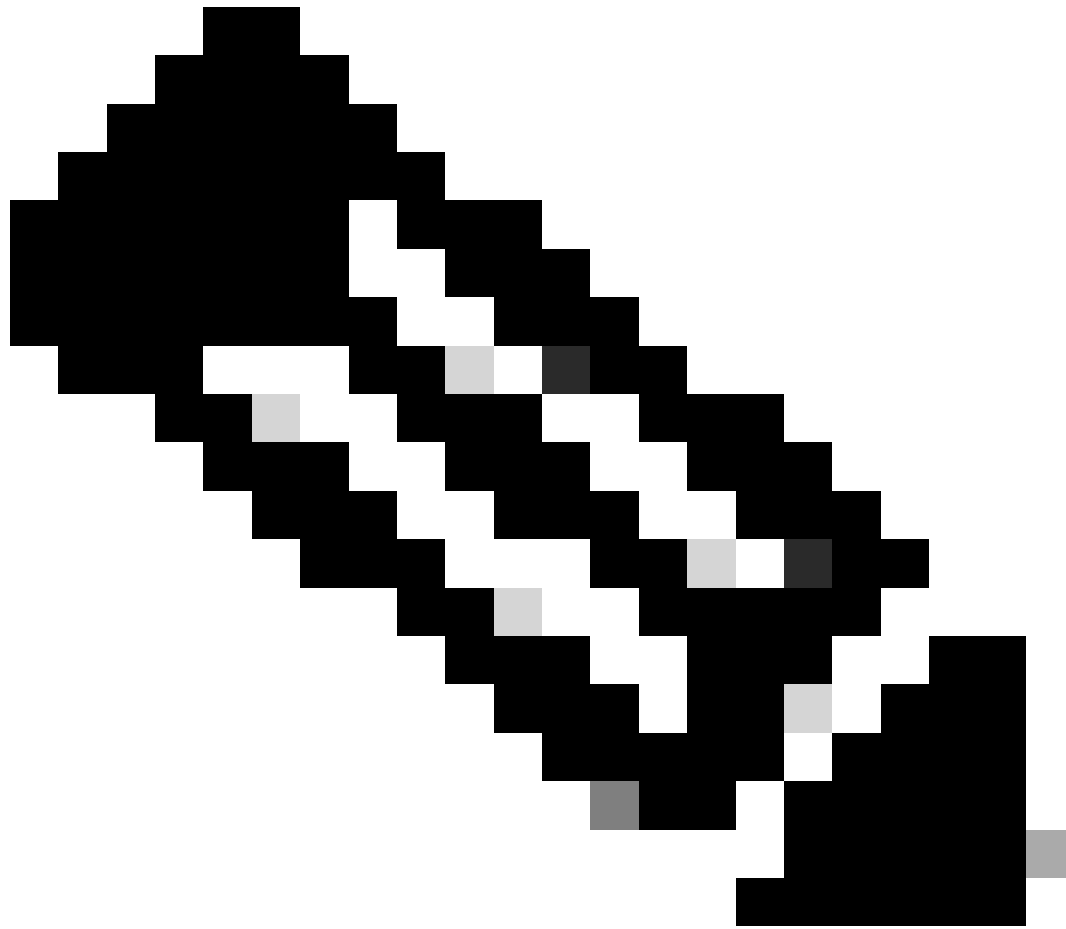
```

Method: POST
URI: https://10.201.230.99/ers/config/networkdevice
HTTP 'Content-Type' Header: application/xml | application/json
HTTP 'Accept' Header: application/xml | application/json
HTTP 'ERS-Media-Type' Header (Not Mandatory): network.networkdevice.1.1
HTTP 'X-CSRF-TOKEN' Header (Required Only if Enabled from GUI): The Token value from the GET X-CSRF-TOKEN fetch request

Request Content:
XML
<?xml version="1.0" encoding="UTF-8">
<ns0:networkdevice xmlns:ns0="network.ers.ise.cisco.com" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ns1="ers.ise.cisco.com" xmlns:ers="ers.ise.cisco.com" description="example nd" ns="">
  <authenticationSettings>
    <dtlsRequired>true</dtlsRequired>
    <enableKeyWrap>true</enableKeyWrap>
    <keyEncryptionKey>1234567890123456</keyEncryptionKey>
    <keyInputFormat>ASCII</keyInputFormat>
    <messageAuthenticatorCodeKey>12345678901234567890</messageAuthenticatorCodeKey>
    <radiusSharedSecret>aaaa</radiusSharedSecret>
  </authenticationSettings>
  <coaPort>1700</coaPort>
  <dtlsDnsName>ISE211.il.com</dtlsDnsName>
  <NetworkDeviceIPList>
    <NetworkDeviceIP>
      <ipaddress>1.1.1</ipaddress>
      <mask>32</mask>
    </NetworkDeviceIP>
  </NetworkDeviceIPList>
  <NetworkDeviceGroupList>
    <NetworkDeviceGroupLocation#All Locations</NetworkDeviceGroup>
    <NetworkDeviceGroupDevice Type#All Device Types</NetworkDeviceGroup>
  </NetworkDeviceGroupList>
  <profileName>Cisco</profileName>
  <smppSettings>
    <linkTrapQuery>true</linkTrapQuery>
    <macTrapQuery>true</macTrapQuery>
    <originatingPolicyServicesNode>autor</originatingPolicyServicesNode>
    <pollingInterval>300</pollingInterval>
    <roCommunity>
  </roCommunity>
  </smppSettings>
</ns0:networkdevice>

```

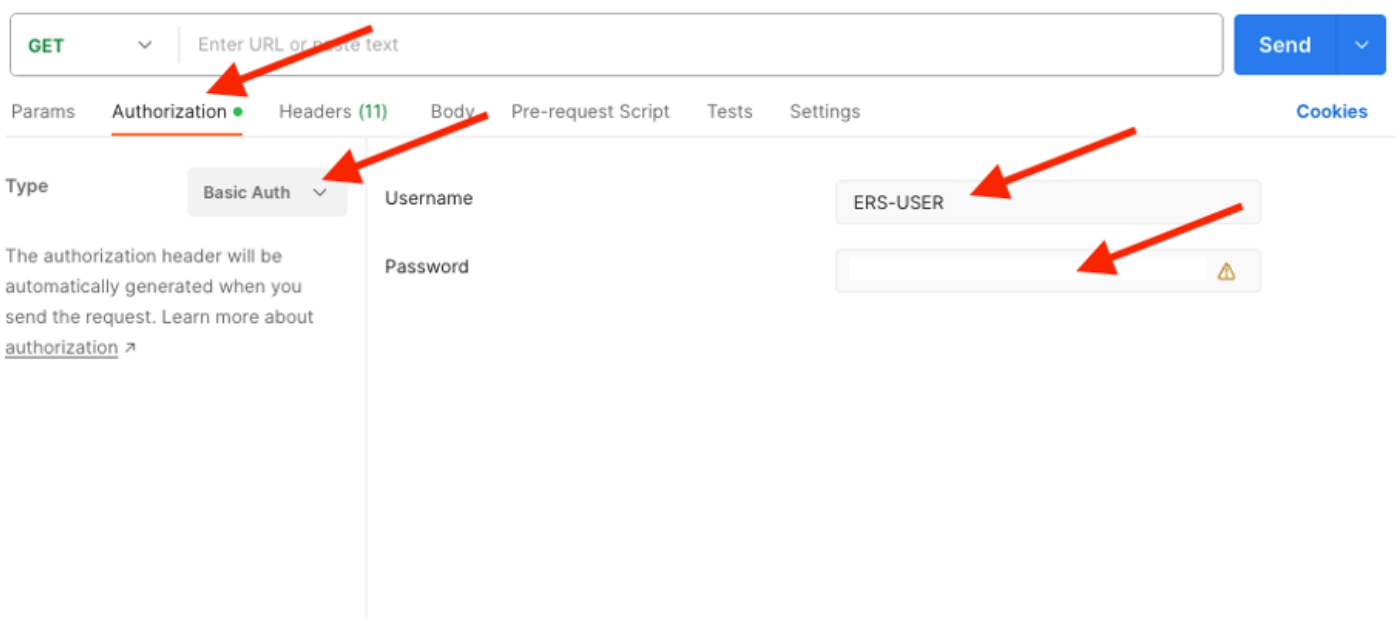
7. Postmanに戻るISEへの基本認証を設定します。Authorization タブで、認証タイプとしてBasic Authを選択し、ISEで作成済みのISE ERSユーザクレデンシャルを追加します。



注:Postmanで変数が設定されていない限り、パスワードはクリアテキストで表示されます。

---

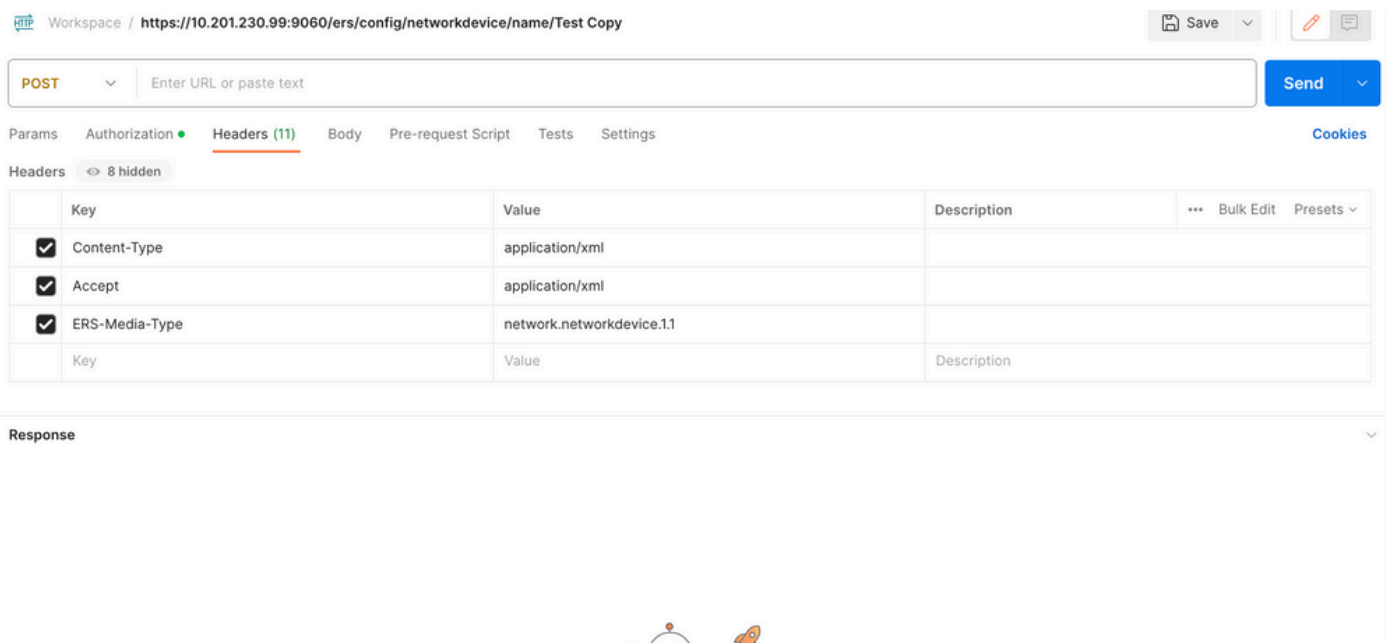




## XMLを使用したNADの作成

XMLを使用して、RADIUS TACACS、SNMP、およびTrustSec設定でTESTNAD1を作成します。

1. SDKのCreateの下に、コールの実行に必要なヘッダーとテンプレート、および予期される応答が表示されます。
2. Headersタブに移動し、SDKに表示されるAPIコールに必要なヘッダーを設定します。ヘッダー設定は次のようになります。



3. Bodyヘッダーに移動し、rawを選択します。これにより、NADの作成に必要なXMLテンプレートを貼り付けることができます。

Workspace / <https://10.201.230.99:9060/ers/config/networkdevice/name/Test Copy> Save


POST Enter URL or paste text Send

Params Authorization Headers (11) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL XML Beautify

1

Response



#### 4. XMLテンプレートは次のようになります ( 必要に応じて値を変更します )。

```
<?xml version="1.0" encoding="UTF-8"?> <ns0:networkdevice xmlns:ns0="network.ers.ise.cisco.com" xmlns:xs="Schema XML File"
xmlns:ns1="ers.ise.cisco.com" xmlns:ers="ers.ise.cisco.com" description="This NAD was added via ERS API" name="TESTNAD1">
<authenticationSettings> <dtlsRequired>true</dtlsRequired> <enableKeyWrap>true</enableKeyWrap>
<keyEncryptionKey>1234567890123456</keyEncryptionKey> <keyInputFormat>ASCII</keyInputFormat>
<messageAuthenticatorCodeKey>12345678901234567890</messageAuthenticatorCodeKey>
<radiusSharedSecret>cisco123</radiusSharedSecret> </authenticationSettings> <coaPort>1700</coaPort>
<dtlsDnsName>Domain</dtlsDnsName> <NetworkDeviceIPList> <NetworkDeviceIP> <ipaddress>NAD IP Address</ipaddress>
<mask>32</mask> </NetworkDeviceIP> </NetworkDeviceIPList> <NetworkDeviceGroupList> <NetworkDeviceGroup>Location#All
Locations#LAB</NetworkDeviceGroup> <NetworkDeviceGroup>Device Type#All Device Types#Access-Layer</NetworkDeviceGroup>
</NetworkDeviceGroupList> <profileName>Cisco</profileName> <snmpsettings> <linkTrapQuery>true</linkTrapQuery>
<macTrapQuery>true</macTrapQuery> <originatingPolicyServicesNode>Auto</originatingPolicyServicesNode>
<pollingInterval>3600</pollingInterval> <roCommunity>aaa</roCommunity> <version>ONE</version> </snmpsettings> <tacacsSettings>
<connectModeOptions>ON_LEGACY</connectModeOptions> <sharedSecret>cisco123</sharedSecret> </tacacsSettings> <trustsecsettings>
<deviceAuthenticationSettings> <sgaDeviceId>TESTNAD1</sgaDeviceId> <sgaDevicePassword>cisco123</sgaDevicePassword>
</deviceAuthenticationSettings> <deviceConfigurationDeployment> <enableModePassword>cisco123</enableModePassword>
<execModePassword>cisco123</execModePassword> <execModeUsername>Admin</execModeUsername>
<includeWhenDeployingSGTUpdates>true</includeWhenDeployingSGTUpdates> </deviceConfigurationDeployment>
<pushIdSupport>false</pushIdSupport> <sgaNotificationAndUpdates> <coaSourceHost>ise3-1test</coaSourceHost>
<downloadEnvironmentDataEveryXSeconds>86400</downloadEnvironmentDataEveryXSeconds>
<downloadPeerAuthorizationPolicyEveryXSeconds>86400</downloadPeerAuthorizationPolicyEveryXSeconds>
<downloadSGACLListsEveryXSeconds>86400</downloadSGACLListsEveryXSeconds>
<otherSGADevicesToTrustThisDevice>false</otherSGADevicesToTrustThisDevice>
<reAuthenticationEveryXSeconds>86400</reAuthenticationEveryXSeconds>
<sendConfigurationToDevice>false</sendConfigurationToDevice>
<sendConfigurationToDeviceUsing>ENABLE_USING_COA</sendConfigurationToDeviceUsing> </sgaNotificationAndUpdates>
</trustsecsettings> </ns0:networkdevice>
```



注：次の行は、`<enableKeyWrap>{false|true}</enableKeyWrap>`が`true`に設定されている場合にのみ必要であることに注意してください。それ以外の場合は、XMLテンプレートから同じものを削除できます。

```
<keyEncryptionKey>1234567890123456</keyEncryptionKey> <keyInputFormat>ASCII</keyInputFormat>  
<messageAuthenticatorCodeKey>12345678901234567890</messageAuthenticatorCodeKey>
```

不要な設定をテンプレートから削除し、NADの作成中に実際に追加する必要があるデータを残すことができます。例として、同じテンプレートがTACACS設定だけに含まれている場合を示します。必要な構成に関係なく、テンプレートの末尾が`</ns0:networkdevice>`であることを確認してください。

```
<?xml version="1.0" encoding="UTF-8"?> <ns0:networkdevice xmlns:ns0="network.ers.ise.cisco.com" xmlns:xs="Schema XML File"  
xmlns:ns1="ers.ise.cisco.com" xmlns:ers="ers.ise.cisco.com" description="This NAD was added via ERS API" name="TESTNAD1">
```

```
<NetworkDeviceIPList> <NetworkDeviceIP> <ipaddress>NAD IP Address</ipaddress> <mask>32</mask> </NetworkDeviceIP>
</NetworkDeviceIPList> <NetworkDeviceGroupList> <NetworkDeviceGroup>Location#All Locations#LAB</NetworkDeviceGroup>
<NetworkDeviceGroup>Device Type#All Device Types#Access-Layer</NetworkDeviceGroup> </NetworkDeviceGroupList>
<profileName>Cisco</profileName> <tacacsSettings> <connectModeOptions>ON_LEGACY</connectModeOptions>
<sharedSecret>cisco123</sharedSecret> </tacacsSettings> </ns0:networkdevice>
```

5. rawのXMLテンプレートをBodyヘッダーの下に貼り付けます。

6. 方法としてPOSTを選択し、<https://{ISE-ip}/ers/config/networkdevice>をペーストして、Sendをクリックします。すべてが正しく設定されていれば、「201 Created」というメッセージが表示され、結果は空白になります。

The screenshot shows a REST client interface with the following details:

- Method: POST
- URL: <https://10.201.230.99/ers/config/networkdevice>
- Body: XML content (raw view) with lines 50-59. The XML includes configuration for network device IP, group, profile, and shared secret.
- Status: 201 Created
- Time: 791 ms
- Size: 1.22 KB

7. NADのGETコールを実行するか、ISE NADリストを確認して、NADが作成されているかどうかを確認します。

Workspace / https://10.201.230.99:9060/ers/config/networkdevice/name/Test Copy

GET https://10.201.230.99/ers/config/networkdevice Send

Params Authorization Headers (13) Body Pre-request Script Tests Settings Cookies

Headers 10 hidden

Key	Value	Description
<input checked="" type="checkbox"/> Content-Type	application/json	
<input checked="" type="checkbox"/> Accept	application/json	
<input checked="" type="checkbox"/> ERS-Media-Type	network.networkdevice.1.1	
Key	Value	Description

Body Cookies (2) Headers (15) Test Results Status: 200 OK Time: 237 ms Size: 3.13 KB Save as Example

Pretty Raw Preview Visualize JSON

```

52   "type": "application/json"
53   }
54 }
55 {
56   "id": "afe572d0-5bcc-11ee-9ab7-9a446445bd4f",
57   "name": "TESTNAD1",
58   "description": "This NAD was added via ERS API",
59   "link": {
60     "rel": "self",
61     "href": "https://10.201.230.99/ers/config/networkdevice/afe572d0-5bcc-11ee-9ab7-9a446445bd4f",
62     "type": "application/json"
63   }
64 },
65 {
66   "id": "63efbc20-4f5a-11ed-b560-6e7768fe732e",
67   "name": "Wireless-9800",
68   "description": "Wireless Controller C9800",
69   "link": {
70     "rel": "self"

```

Administration - Network Resources

Network Devices Network Device Groups Network Device Profiles External RADIUS Servers RADIUS Server Sequences NAC Managers External MDM Location Services

Network Devices

Default Device Device Security Settings

Network Devices

Selected 0 Total 6

Edit + Add Duplicate Import Export Generate PAC Delete

Name	IP/Mask	Profile Name	Location	Type	Description
TESTNAD1	1.1.1.1/32	Cisco	LAB	All Locations Access-Layer	This NAD was added via ERS API

## JSONを使用したNADの作成

JSONを使用して、RADIUS TACACS、SNMP、およびTrustSec設定でTESTNAD2を作成します。

1. SDKのCreateの下に、コールの実行に必要なヘッダーとテンプレート、および予期される応答が表示されます。
2. Headersタブに移動し、SDKに表示されるAPIコールに必要なヘッダーを設定します。ヘッダー設定は次のようになります。

Workspace / <https://10.201.230.99:9060/ers/config/networkdevice/name/Test> Save Send

POST Enter URL or paste text

Params Authorization Headers (12) Body Pre-request Script Tests Settings Cookies

Headers 9 hidden

Key	Value	Description	Bulk Edit	Presets
<input checked="" type="checkbox"/> Content-Type	application/json			
<input checked="" type="checkbox"/> Accept	application/json			
<input checked="" type="checkbox"/> ERS-Media-Type	network.networkdevice.1.1			
Key	Value	Description		

3. Bodyヘッダーに移動し、rawを選択します。これにより、NADの作成に必要なJSONテンプレートを貼り付けることができます。

Workspace / <https://10.201.230.99:9060/ers/config/networkdevice/name/Test Copy> Save Send

POST Enter URL or paste text

Params Authorization Headers (11) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL XML

1

Response

4. JSONテンプレートは次のようになります (必要に応じて値を変更します)。

```
{ "NetworkDevice": { "name": "TESTNAD2", "description": "This NAD was added via ERS API", "authenticationSettings": {
"radiusSharedSecret": "cisco123", "enableKeyWrap": true, "dtlsRequired": true, "keyEncryptionKey": "1234567890123456",
"messageAuthenticatorCodeKey": "12345678901234567890", "keyInputFormat": "ASCII" }, "snmpsettings": { "version": "ONE",
"roCommunity": "aaa", "pollingInterval": 3600, "linkTrapQuery": true, "macTrapQuery": true, "originatingPolicyServicesNode": "Auto" },
"trustsecsettings": { "deviceAuthenticationSettings": { "sgaDeviceId": "TESTNAD2", "sgaDevicePassword": "cisco123" },
"sgaNotificationAndUpdates": { "downloadEnvironmentDataEveryXSeconds": 86400, "downloadPeerAuthorizationPolicyEveryXSeconds":
86400, "reAuthenticationEveryXSeconds": 86400, "downloadSGACLListsEveryXSeconds": 86400, "otherSGADevicesToTrustThisDevice":
false, "sendConfigurationToDevice": false, "sendConfigurationToDeviceUsing": "ENABLE_USING_COA", "coaSourceHost": "ise3-1test" },
"deviceConfigurationDeployment": { "includeWhenDeployingSGTUpdates": true, "enableModePassword": "cisco123", "execModePassword":
"cisco123", "execModeUsername": "Admin" }, "pushIdSupport": "false" }, "tacacsSettings": { "sharedSecret": "cisco123",
"connectModeOptions": "ON_LEGACY" }, "profileName": "Cisco", "coaPort": 1700, "dtlsDnsName": "Domain", "NetworkDeviceIPList": [ {
"ipaddress": "NAD IP Adress", "mask": 32 } ], "NetworkDeviceGroupList": [ "Location#All Locations", "Device Type#All Device Types" ] } }
```





注：次の行が必要になるのは、`enableKeyWrap`:{false|true}がtrueに設定されている場合だけであることに注意してください。それ以外の場合は、JSONテンプレートから同じものを削除できます。

---

`"keyEncryptionKey": "1234567890123456", "messageAuthenticatorCodeKey": "12345678901234567890", "keyInputFormat": "ASCII"` また、不要な設定をテンプレートから削除し、NADの作成中に実際に追加する必要があるデータを残すこともできます。

5. rawのJSONテンプレートをBodyヘッダーの下に貼り付けます。

6. 方法としてPOSTを選択し、`https://{ISE-ip}/ers/config/networkdevice`をペーストして、Sendをクリックします。すべてが正しく設定されていれば、**201 Created**メッセージが表示され、結果が空になります。

Workspace / <https://10.201.230.99:9060/ers/config/networkdevice/name/Test Copy> Save

POST <https://10.201.230.99/ers/config/networkdevice> Send

Params Authorization Headers (13) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "NetworkDevice": {
3     "name": "TESTNAD2",
4     "description": "This NAD was added via ERS API",
5     "authenticationSettings": {
6       "radiusSharedSecret": "cisco123",
7       "enableKeyWrap": true,
8       "dtlsRequired": true,
9       "keyEncryptionKey": "1234567890123456",
10      "messageAuthenticatorCodeKey": "12345678901234567890",
11      "keyInputFormat": "ASCII"
12    }
13  }
14 }
```

Body Cookies (2) Headers (17) Test Results Status: 201 Created Time: 678 ms Size: 1.03 KB Save as Example

Pretty Raw Preview Visualize JSON

1

7. NADのGETコールを実行するか、ISE NADリストを確認して、NADが作成されているかどうかを確認します。

Workspace / <https://10.201.230.99:9060/ers/config/networkdevice/name/Test Copy> Save

GET <https://10.201.230.99/ers/config/networkdevice> Send

Params Authorization Headers (13) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "NetworkDevice": {
3     "name": "TESTNAD2",
4     "description": "This NAD was added via ERS API",
5     "authenticationSettings": {
6       "radiusSharedSecret": "cisco123",
7       "enableKeyWrap": true,
8       "dtlsRequired": true,
9       "keyEncryptionKey": "1234567890123456",
10      "messageAuthenticatorCodeKey": "12345678901234567890",
11      "keyInputFormat": "ASCII"
12    }
13  }
14 }
```

Body Cookies (2) Headers (18) Test Results Status: 200 OK Time: 659 ms Size: 3.74 KB Save as Example

Pretty Raw Preview Visualize JSON

```
57 {
58   "name": "TESTNAD1",
59   "description": "This NAD was added via ERS API",
60   "link": {
61     "rel": "self",
62     "href": "https://10.201.230.99/ers/config/networkdevice/afe572d0-5bcc-11ee-9ab7-9a446445bd4f",
63     "type": "application/json"
64   }
65 },
66 {
67   "id": "9dd45a60-5bd7-11ee-9ab7-9a446445bd4f",
68   "name": "TESTNAD2",
69   "description": "This NAD was added via ERS API",
70   "link": {
71     "rel": "self",
72     "href": "https://10.201.230.99/ers/config/networkdevice/9dd45a60-5bd7-11ee-9ab7-9a446445bd4f",
73     "type": "application/json"
74   }
75 }
```



Network Devices

Name	IP/Mask	Profile Name	Location	Type	Description
TESTNAD1	1.1.1.1/32	Cisco	LAB	Access-Layer	This NAD was added via ERS API
TESTNAD2	2.2.2.2/32	Cisco	All Locations	All Device Types	This NAD was added via ERS API

## 確認

<https://{iseip}:{port}/api/swagger-ui/index.html>や<https://{iseip}:9060/ers/sdk>などのAPIサービスのGUIページにアクセスできる場合は、APIサービスが期待どおりに動作していることを意味します。

## トラブルシューティング

- すべてのREST操作が監査され、ログがシステムログに記録されます。
- Open APIに関連する問題をトラブルシューティングするには、**Debug Log Configuration**ウィンドウでapiserviceコンポーネントのログレベルを**DEBUG**に設定します。
- ERS APIに関する問題をトラブルシューティングするには、**Debug Log Configuration**ウィンドウでersコンポーネントの**Log Level**を**DEBUG**に設定します。このウィンドウを表示するには、Cisco ISE GUIに移動し、メニューアイコンをクリックして、**Operations > Troubleshoot > Debug Wizard > Debug Log Configuration**の順に選択します。
- ログは、**Download Logs**ウィンドウからダウンロードできます。このウィンドウを表示するには、Cisco ISE GUIに移動し、メニューアイコンをクリックして、**Operations > Troubleshoot > Download Logs**の順に選択します。
- Support Bundleタブの下の**Download**ボタンをクリックして、このタブからサポートバンドルをダウンロードするか、api-serviceデバッグログの**Log File**の値をクリックして、このapi-serviceデバッグログを**Debug Logs**タブからダウンロードするかを選択できます。

## 翻訳について

シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。