

# Creazione di dispositivi di rete ISE con l'API ERS

## Sommario

---

[Introduzione](#)

[Prerequisiti](#)

[Requisiti](#)

[Componenti usati](#)

[Configurazione](#)

[Abilita ERS \(porta 9060\)](#)

[Crea amministratore ERS](#)

[Imposta postino](#)

[ISE SDK e autorizzazione Basic Postman](#)

[Creazione e utilizzo di XML](#)

[Crea e utilizza JSON](#)

[Verifica](#)

[Risoluzione dei problemi](#)

---

## Introduzione

Questo documento descrive il processo di creazione di dispositivi di accesso alla rete (NAD) su ISE tramite l'API ERS con PostMan come client REST.

## Prerequisiti

### Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- ISE (Identity Services Engine)
- ERS (Servizi esterni di assistenza a terra)
- I clienti REST come Postman, RESTED, Insonnia, e così via.

### Componenti usati

Le informazioni fornite in questo documento si basano sulle seguenti versioni software:

- Patch 6 per Cisco ISE (Identity Services Engine) 3.1
- Postman REST client v10.17.4



Nota: la procedura è simile o identica per altre versioni ISE e client REST. Se non specificato diversamente, è possibile eseguire la procedura seguente su tutte le versioni software ISE 2.x e 3.x.

---

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

## Configurazione

### Abilita ERS (porta 9060)

Le API ERS sono API REST solo HTTPS che operano sulla porta 443 e sulla porta 9060. La porta 9060 è chiusa per impostazione predefinita, quindi deve essere aperta per prima. Se i client che tentano di accedere a questa porta non attivano prima ERS, viene visualizzato un timeout del

server. Pertanto, il primo requisito è abilitare ERS dall'interfaccia utente di amministrazione di Cisco ISE.

Passare a Amministrazione > Impostazioni > Impostazioni API e abilitare l'interruttore ERS (Read/Write).

The screenshot shows the Cisco ISE Administration System interface. The top navigation bar includes 'Administration - System' and various system management tabs 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' selected. 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 blue bar and a red arrow), and 'Open API (Read/Write)' which is turned off. Below this, under 'CSRF Check ( only for ERS Settings )', there are two radio buttons: 'Enable CSRF Check for Enhanced Security (Not compatible with pre ISE 2.3 Clients)' which is unselected, and 'Disable CSRF For ERS Request (compatible with ERS clients older than ISE 2.3)' which is selected. At the bottom right of the settings area, there are 'Reset' and 'Save' buttons.



Nota: le API ERS supportano TLS 1.1 e TLS 1.2. Le API ERS non supportano TLS 1.0, a prescindere dall'abilitazione di TLS 1.0 nella finestra Security Settings (Impostazioni protezione) dell'interfaccia utente di Cisco ISE (Amministrazione > Sistema > Impostazioni > Impostazioni protezione). L'attivazione di TLS 1.0 nella finestra Impostazioni protezione è correlata solo al protocollo EAP e non influisce sulle API ERS.

---

## Crea amministratore ERS

Creare un Cisco ISE Administrator, assegnare una password e aggiungere l'utente al gruppo admin come Amministratore ERS. È possibile lasciare vuota la parte restante della configurazione.

Admin User

\* Name **ERS-USER** ←

Status **Enabled** ▾

Email   Include system alerts in emails

Expires

Hard Data

Inactive account never created

---

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 ▾ + ←

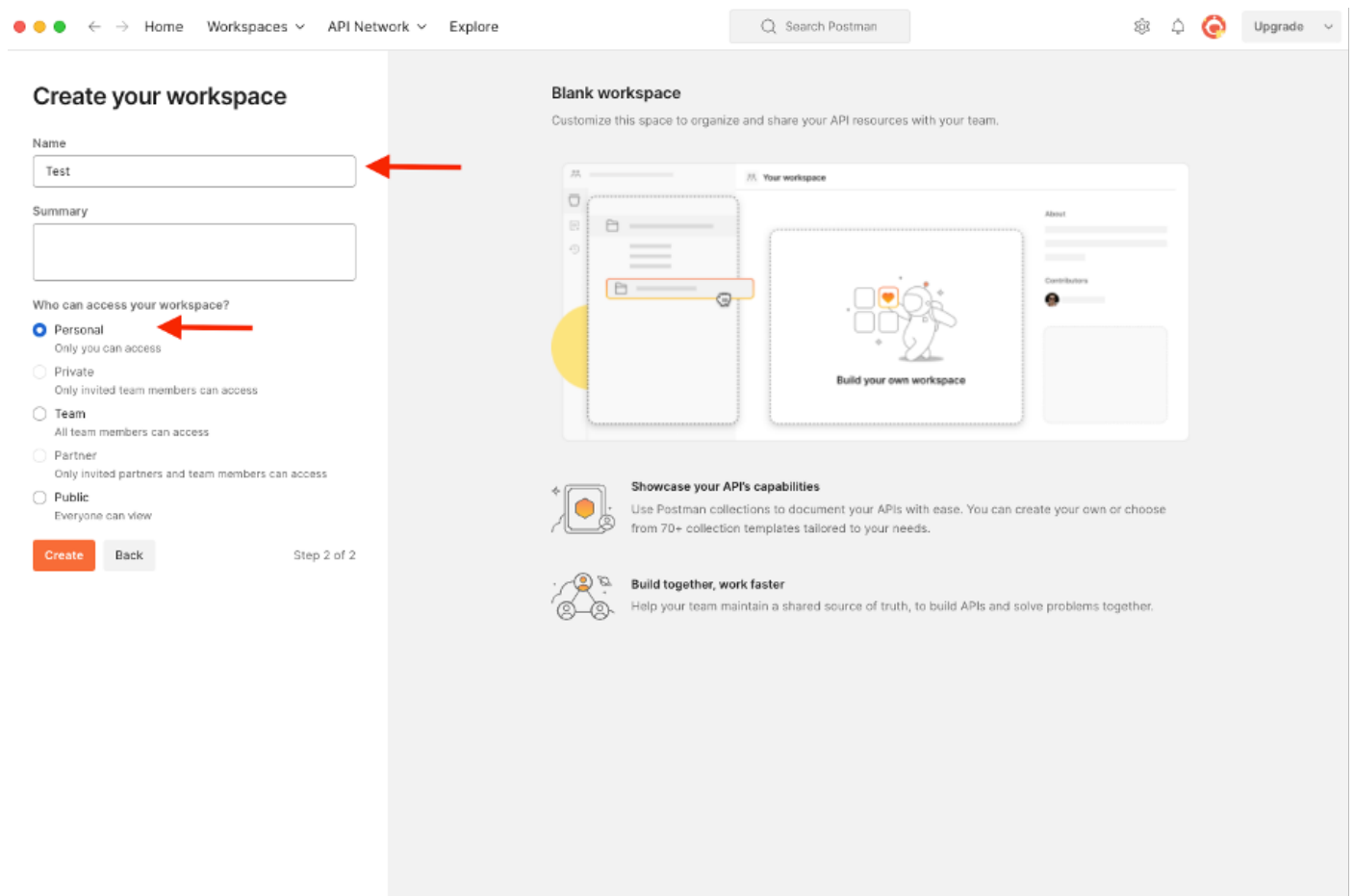
## Imposta postino

Scarica o utilizza la versione online di Postman.

1. Creare un utente e un workspace facendo clic su Crea workspace nella scheda Workspace.

The screenshot shows the Postman application interface. At the top, there are navigation tabs for 'Home', 'Workspaces', 'API Network', and 'Explore'. The 'Workspaces' tab is active, and a dropdown menu is open, showing options like 'Search workspaces' and 'Create Workspace'. A red arrow points to the 'Create Workspace' button. Below the dropdown, there is a list of 'Recently visited' workspaces, currently showing 'Test'. The main content area displays a list of public workspaces, including 'Checkout API (v70)', 'PI (v3)', and 'PI'. The interface also features a sidebar on the left with various navigation options and a top right corner with settings and an 'Upgrade' button.

2. Selezionare Spazio di lavoro vuoto e assegnare un nome al workspace. È possibile aggiungere una descrizione e renderla pubblica. Per questo esempio, Personalis ha selezionato.



Una volta creata l'area di lavoro, è ora possibile configurare le chiamate API.

## ISE SDK e autorizzazione Basic Postman

Per configurare una chiamata, accedere prima a ISE ERS SDK (Software Developer Kit). Questo strumento compila l'intero elenco di chiamate API che ISE può eseguire:

1. Passare a <https://{ise-ip}/ers/sdk>.
2. Accedere usando le credenziali ISE Admin.
3. Espandere la documentazione API.
4. Scorrere verso il basso fino a individuare Periferica di rete e fare clic su di essa.
5. Sotto questa opzione, è ora possibile trovare tutte le operazioni disponibili che è possibile eseguire per i dispositivi di rete su ISE. Selezionare Crea.

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. È ora possibile visualizzare la configurazione richiesta per eseguire la chiamata API utilizzando XML o JSON su qualsiasi client REST, nonché un esempio di risposta prevista.

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>ISE111.il.com</dtlsDnsName>
  <NetworkDeviceIPList>
    <NetworkDeviceIP>
      <ipaddress>1.1.1.1</ipaddress>
      <mask>32</mask>
    </NetworkDeviceIP>
  </NetworkDeviceIPList>
  <NetworkDeviceGroupList>
    <NetworkDeviceGroupLocation#All Locations</NetworkDeviceGroup>
    <NetworkDeviceGroupDevice Type#All Device Types</NetworkDeviceGroup>
  </NetworkDeviceGroupList>
  <profileName>Cisco</profileName>
  <smSettings>
    <linkTrapQuery>true</linkTrapQuery>
    <macTrapQuery>true</macTrapQuery>
    <originatingPolicyServicesNode>autor</originatingPolicyServicesNode>
    <pollingInterval>300</pollingInterval>
    <roCommunity>roCommunity
  </smSettings>
</ns0:networkdevice>

```

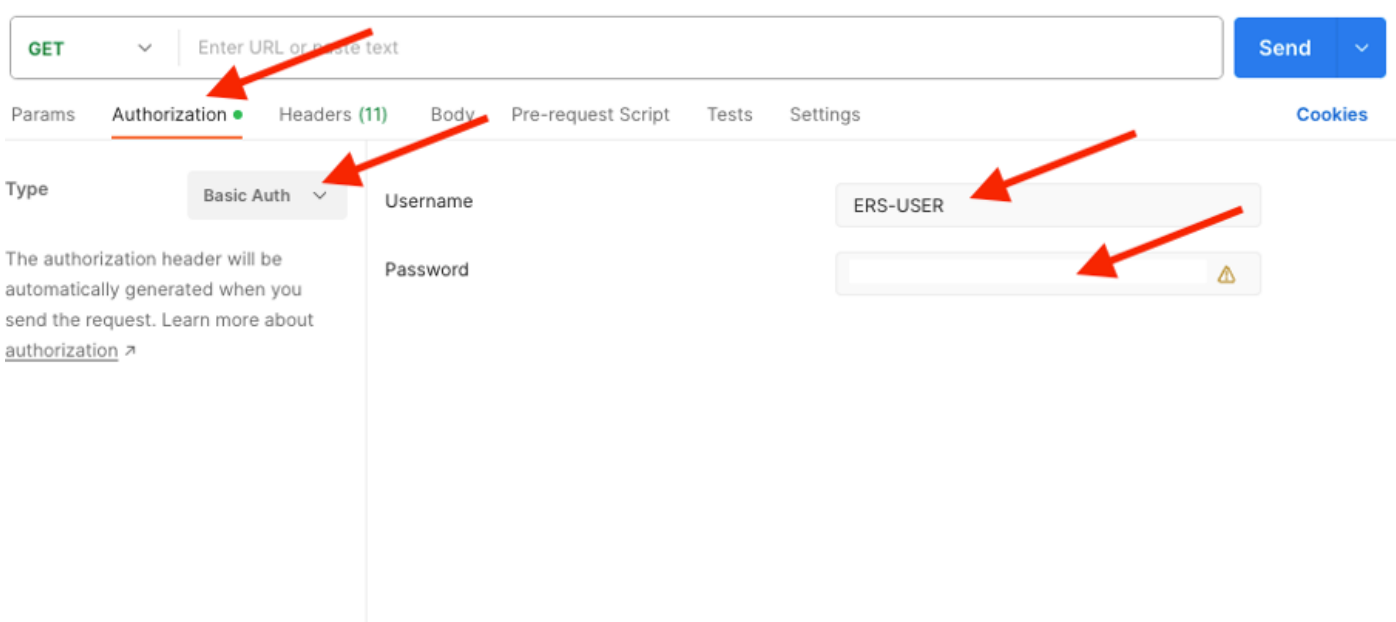
7. Back to Postman configurare l'autenticazione di base per ISE. Nella scheda Authorization (Autorizzazione), selezionare Basic Auth (Autenticazione di base) come tipo di autenticazione, quindi aggiungere le credenziali utente ISE ERS precedentemente create su ISE.



Nota: la password viene visualizzata come testo non crittografato a meno che non siano configurate variabili in Postman.

---



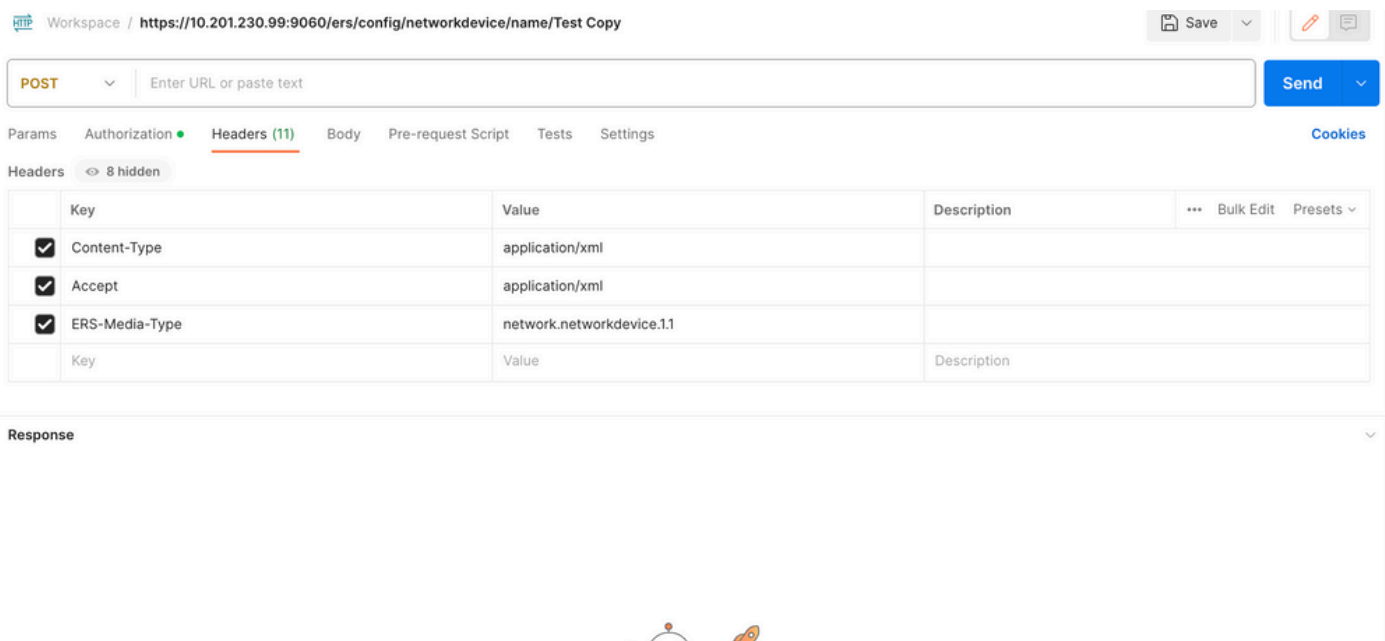


## Creazione e utilizzo di XML

Creare TESTNAD1 con le impostazioni RADIUS TACACS, SNMP e TrustSec utilizzando XML.

1. Nell'SDK, in Crea, sono presenti le intestazioni e i modelli necessari per eseguire la chiamata, nonché la risposta prevista.

2. Passare alla scheda Intestazioni e configurare le intestazioni necessarie per la chiamata API come mostrato nell'SDK. La configurazione dell'intestazione deve essere simile alla seguente:



3. Spostarsi sull'intestazione Body e selezionare raw. In questo modo è possibile incollare il modello XML necessario per la creazione di NAD.

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

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

1

Response



#### 4. Il modello XML ha il seguente aspetto (modificare i valori in base alle esigenze):

```
<?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>
```



**Nota:** è importante notare che le righe successive sono obbligatorie solo se `<enableKeyWrap>{false|true}</enableKeyWrap>` è impostato su **true**. In caso contrario, è possibile eliminare lo stesso dal modello XML:

---

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

È possibile rimuovere la configurazione non necessaria dal modello e lasciare semplicemente i dati effettivamente necessari da aggiungere durante la creazione del NAD. Ad esempio, questo è lo stesso modello, ma solo con la configurazione TACACS. Independentemente dalla configurazione richiesta, assicurarsi che il modello termini con `</ns0:networkdevice>`.

```
<?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. Incollare il modello XML per **raw** sotto l'intestazione **Body**.

6. Selezionare **POST** come metodo, incollare [https://\(ISE-ip\)/ers/config/networkdevice](https://(ISE-ip)/ers/config/networkdevice) e fare clic su **Send**. Se tutti gli elementi sono stati configurati correttamente, è necessario visualizzare un messaggio **201 Created** (Creato) e il risultato è vuoto.

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

- URL: `https://10.201.230.99/ers/config/networkdevice`
- Method: **POST**
- Body type: **raw** (selected)
- Body content (XML):

```
50 <downloadEnvironmentDataEveryXSeconds>86400</downloadEnvironmentDataEveryXSeconds>
51 <downloadPeerAuthorizationPolicyEveryXSeconds>86400</downloadPeerAuthorizationPolicyEveryXSeconds>
52 <downloadSGACLListsEveryXSeconds>86400</downloadSGACLListsEveryXSeconds>
53 <otherSGADevicesToTrustThisDevice>>false</otherSGADevicesToTrustThisDevice>
54 <reAuthenticationEveryXSeconds>86400</reAuthenticationEveryXSeconds>
55 <sendConfigurationToDevice>>false</sendConfigurationToDevice>
56 <sendConfigurationToDeviceUsing>ENABLE_USING_COA</sendConfigurationToDeviceUsing>
57 </sgaNotificationAndUpdates>
58 </trustsecsettings>
59 </ns0:networkdevice>
```
- Status: **201 Created** (indicated by a green checkmark icon)
- Time: 791 ms, Size: 1.22 KB
- Response body is empty.

7. Confermare se il NAD è stato creato eseguendo una chiamata **GET** per il NAD o controllando l'elenco ISE NAD.

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

Headers 10 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	

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"

```

Cisco ISE 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

<input type="checkbox"/>	Name	IP/Mask	Profile Name	Location	Type	Description
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>	TESTNAD1	1.1.1.1/32	Cisco	LAB	All Locations Access-Layer	This NAD was added via ERS API
<input type="checkbox"/>						

## Crea e utilizza JSON

Creare TESTAND2con le impostazioni RADIUS TACACS, SNMP e TrustSec utilizzando JSON.

1. Nell'SDK, in **Crea**, sono presenti le intestazioni e i modelli necessari per eseguire la chiamata, nonché la risposta prevista.
2. Passare alla scheda **Intestazioni** e configurare le intestazioni necessarie per la chiamata API come mostrato nell'SDK. La configurazione dell'intestazione deve essere simile alla seguente:

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

POST  Send

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. Spostarsi sull'intestazione **Body** e selezionare **raw**. In questo modo è possibile incollare il modello JSON necessario per la creazione di NAD.

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


POST  Send

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

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

1

Response



4. Il modello JSON deve avere il seguente aspetto (modificare i valori in base alle esigenze):

```
{ "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" ] }
```



**Nota:** è importante notare che le righe successive sono obbligatorie solo se `enableKeyWrap":{false|true}`, è impostato su `true`. In caso contrario, è possibile eliminare lo stesso dal modello JSON:

---

`"keyEncryptionKey": "1234567890123456", "messageAuthenticatorCodeKey": "12345678901234567890", "keyInputFormat": "ASCII"` È inoltre possibile rimuovere la configurazione non necessaria dal modello e lasciare semplicemente i dati effettivamente necessari da aggiungere durante la creazione del NAD.

5. Incollare il modello JSON per **raw** sotto l'intestazione **Body**.

6. Selezionare **POST** come metodo, incollare [https://\(ISE-ip\)/ers/config/networkdevice](https://(ISE-ip)/ers/config/networkdevice) e fare clic su **Send**. Se tutti gli elementi sono stati configurati correttamente, è necessario visualizzare un messaggio **201 Created** e il risultato è vuoto.

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      "keyFormat": "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. Confermare se il NAD è stato creato eseguendo una chiamata GET per il NAD o controllando l'elenco ISE NAD.

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      "keyFormat": "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   "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": "9dd45a60-5bd7-11ee-9ab7-9a446445bd4f",
67   "name": "TESTNAD2",
68   "description": "This NAD was added via ERS API",
69   "link": {
70     "rel": "self",
71     "href": "https://10.201.230.99/ers/config/networkdevice/9dd45a60-5bd7-11ee-9ab7-9a446445bd4f",
72     "type": "application/json"
73   }
74 },
75 }
```



Administration - Network Resources

Network Devices

Network Devices

Default Device

Device Security Settings

Network Devices

Selected 0 Total 7

Edit + Add Duplicate Import Export Generate PAC Delete

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

## Verifica

Se è possibile accedere alla pagina dell'interfaccia utente del servizio API, ad esempio <https://{iseip}:{port}/api/swagger-ui/index.html> o <https://{iseip}:9060/ers/sdk>, significa che il servizio API funziona come previsto.

## Risoluzione dei problemi

- Tutte le operazioni REST vengono controllate e i log vengono registrati nei log di sistema.
- Per risolvere i problemi relativi alle API aperte, impostare il **livello di log** per il componente **apiservice** su **DEBUG** nella finestra **Configurazione log di debug**.
- Per risolvere i problemi relativi alle API ERS, impostare il **livello di log** per il componente **ers** su **DEBUG** nella finestra **Configurazione log di debug**. Per visualizzare questa finestra, passare all'interfaccia grafica di Cisco ISE, fare clic sull'icona Menu e scegliere **Operazioni > Risoluzione dei problemi > Debug guidato > Debug Log Configuration**.
- È possibile scaricare i log dalla finestra **Scarica log**. Per visualizzare questa finestra, passare all'interfaccia utente di Cisco ISE, fare clic sull'icona **Menu** e scegliere **Operazioni > Risoluzione dei problemi > Log di download**.
- È possibile scegliere di scaricare un bundle di supporto dalla scheda Support Bundle facendo clic sul pulsante **Download** nella scheda, oppure scaricare i log di debug **api-service** dalla scheda **Debug Logs** facendo clic sul **valore Log File** per il log di debug api-service.

## Informazioni su questa traduzione

Cisco ha tradotto questo documento utilizzando una combinazione di tecnologie automatiche e umane per offrire ai nostri utenti in tutto il mondo contenuti di supporto nella propria lingua. Si noti che anche la migliore traduzione automatica non sarà mai accurata come quella fornita da un traduttore professionista. Cisco Systems, Inc. non si assume alcuna responsabilità per l'accuratezza di queste traduzioni e consiglia di consultare sempre il documento originale in inglese (disponibile al link fornito).