Prisms Documentation

Release 1.1

Product

Jun 01, 2020

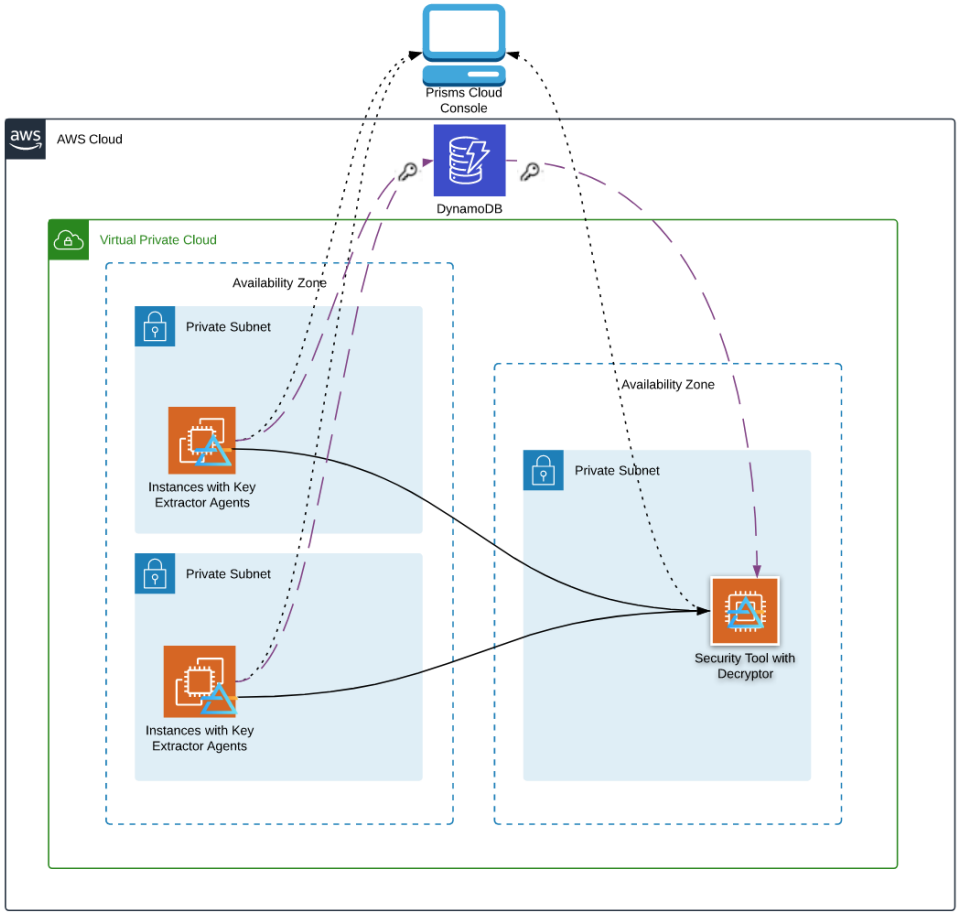
TLS API User Guide

CHAPTER 1

Introduction

Nubeva Prisms TLS edition provides visibility for encrypted cloud traffic. The system has a split SaaS architecture comprised of central control: `Prisms Cloud Console`, `Key Agents` and `Decryptor Agents` (or 'Decryptors'). The control plane is split between the `Prisms Cloud Console` and `Agents`.

An overview of the architecture in an AWS cloud is depicted in the figure below.



Note: The black solid arrows represent decrypted traffic sent using AWS's .

The elements of the control plane can be created using either the or the `Prisms API` . The `Prisms API` is explained in the following sections.

CHAPTER 2

TLS Decrypt API Calls

Prisms API calls provides REST methods for `Projects`, `Accounts`, `Key Agents`, `Decryptors`, `Source Groups`, `Destinations`, `Connections` and `Filters`.

The base URL for making API calls is `https://i.nuos.io/nuapi/`

All calls are authenticated using tokens. The next section describes how to request authentication tokens.

2.1 Headers

Name	Value	Required	Description
Content-type	application/json	Yes	
Authorization	Bearer <your-token>	Yes	For any action to retrieve personal User data, this Authorization header is required. Any action that does not require the Authorization header will specify so.

2.2 Authentication

In order to keep your account safe and secure, we provide you an API call to fetch a Authorization Bearer token. Please make sure to fetch the token for your project and include the following header:

2.3 Get Started with Postman

If you want to quickly get started, here is a Postman collection with the available API endpoints. Create environment variables and replace values to your Account specific information.

`NuAPI Collection`

CHAPTER 3

Examples

3.1 Creating Tokens

Use the `/tokens` endpoint to fetch the token required to authenticate all your future requests. When you call upon the `/tokens` endpoint, you will need to provide your Email and AccountID as payload parameters.

Your Email and AccountID can be found at https://i.nuos.io/account. If you do not have an Email and AccountID, please sign up at https://i.nuos.io and you can learn more about Nubeva at https://www.nubeva.com.

First, call upon the `/tokens` endpoint:

Bash

```
curl -X POST https://i.nuos.io/nuapi/tokens -H "Content-type: application/json"
--data '{ "Email": "YOUR EMAIL", "AccountID": "YOUR ACCOUNTID" }'
```

Python

```
import requests

payload = {'Email': 'YOUR EMAIL', 'AccountID': 'YOUR ACCOUNTID'}
r = requests.post('https://i.nuos.io/nuapi/tokens', json=payload)
print(r.json())
```

Save the `token` key value from the response. The value will be used for all future requests as the Bearer Token.

An example of authenticated calls using the Bearer Token is below:

Bash

```
curl https://i.nuos.io/nuapi/accounts -H "Authorization: Bearer <YOUR-TOKEN>"
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
r = requests.get('https://i.nuos.io/nuapi/accounts', headers=headers)
print(r.json())
```

3.2 Launch Commands

If you would like to use the NuAPI to fetch the Key Agent and Decryptor launch commands, you may do the following:

Bash

```
echo -e "\nGet Key Agent Launch Instructions"
curl https://i.nuos.io/nuapi/keyagents/commands?projectid="YOUR PROJECTID" -H
↪"Authorization: Bearer <YOUR-TOKEN>"

echo -e "\nGet Decryptor Launch Instructions"
curl https://i.nuos.io/nuapi/decryptors/commands?projectid="YOUR PROJECTID" -H
↪"Authorization: Bearer <YOUR-TOKEN>"
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
project_id = "YOUR PROJECTID"

# Both /keyagents/commands and /decryptors/commands use the same GET query parameters
params = {
    "projectid": project_id
}

print("\nGet Key Agent Launch Instructions")
r = requests.get('https://i.nuos.io/nuapi/keyagents/commands', headers=headers,␣
↪params=payload)
response = r.json()
print(response['item']['linux'])

print("\nGet Decryptor Launch Instructions")
r = requests.get('https://i.nuos.io/nuapi/decryptors/commands', headers=headers,␣
↪params=payload)
response = r.json()
print(response['item']['linux'])
```

3.3 Setting Up an Existing Project

3.3.1 Description

This section will cover setting up a simple project using the NuAPI. The following will be covered:

0. Identify your project

1. Setting up Source Groups
2. Setting up Filters to include your Key Agents into the Source Groups
3. Setting up Destination Groups
4. Creating a Connection to tap traffic

By the end of this tutorial, you will have one Key Agent extracting out session keys and a Decryptor decrypting any traffic that it picks up from its interface.

For this example, AWS Cloud services will be used for the source and destination instances.

3.3.2 Requirements

- **An account** setup at https://i.nuos.io. Once you have an account, you will also have a project setup with us.
- Retrieve your **AccountID** from the website at https://i.nuos.io/account.
- Please have an **authentication token** to access your data through the NuAPI. Please check out Creating Tokens if you need assistance in creating the tokens.
- Have a **Key Store Database**. When you create Project, a default Key Store Database is created for you within Nubeva's account. You may create your own Key Store Database within your AWS account by following the *Private KeyDB Tip* here. You may check if you have a key store database and the name of it by fetching your Project through the `GET /projects` endpoint and validating the `CredObj` key value on your NuAPI project result.
- **2 EC2 instances** for `Source` and `Destination`. For simplicity, please make sure that both instances are in the same VPC and that both instances have public access (ie both have public IPs).
- **Docker** needs to be installed on both EC2 instances.
  - Directions on installation of the Key Agent can be found at Installing Key Agents.
  - Directions on installation of the Decryptor can be found at Installing Decryptors.

3.3.3 Identify Your Project

You have to first identify the Project that you want to set up your environment. By default, when you first sign up for an account with Nubeva, a project is created for you.

To identify which Project you would like to set up your environment is, first list out what projects are currently available to you within your account.

Bash

```
curl https://i.nuos.io/nuapi/projects -H "Authorization: Bearer <YOUR-TOKEN>"
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
r = requests.get('https://i.nuos.io/nuapi/projects', headers=headers)
print(r.json())
```

From the list of projects that you have retrieved through the `/projects` endpoint, identify the desired project and take note of the UniqueID. The UniqueID will be the ProjectID for all future calls.

3.3.4 Setting Up Source Groups

A Source Group needs to be created so that it can group Key Agents into a single unit.

Once the Source Group has been created, please keep note of the UniqueID that is returned from the result.

Bash

```
curl -X POST https://i.nuos.io/nuapi/srcgroups -H "Authorization: Bearer <YOUR-TOKEN>
↪" -H "Content-type: application/json"
--data '{"ProjectID": "YOUR PROJECTID", "Name": "AWS", "Description": "Only include␣
↪key agents in the AWS Cloud"}'
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
project_id = 'YOUR PROJECTID'
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
payload = {
    "ProjectID": project_id,
    "Name": "AWS",
    "Description": "Only include key agents in the AWS Cloud"
}
r = requests.post('https://i.nuos.io/nuapi/srcgroups', json=payload, headers=headers)
response = r.json()
print("UniqueID: %s" % response['item']['UniqueID'])
```

3.3.5 Setting Up Filters

Note: Make sure to have your Key Agent running. Directions on installation of the Key Agent can be found at Installing Key Agents.

Filters need to be created and attached to Source Groups so that the active Key Agents know to which Source Group they belong to.

First, we need to understand what type of Filters we can create. To check what the available options are, we will run `POST /filters/options`.

Bash

```
curl -X POST https://i.nuos.io/nuapi/filters/options -H "Authorization: Bearer <YOUR-
↪TOKEN>" -H "Content-type: application/json"
--data '{"ProjectID": "YOUR PROJECTID"}'
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
project_id = 'YOUR PROJECTID'
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
payload = {
    "ProjectID": project_id
```

(continues on next page)

(continued from previous page)

```
}
r = requests.post('https://i.nuos.io/nuapi/filters/options', json=payload,␣
↪headers=headers)
print(r.json())
```

Since this tutorial launches a Key Agent in AWS EC2 instances, one of the metadata options is type Cloud and value AWS.

```
{
    "Type": "Cloud",
    "Value": [
        "AWS"
    ]
}
```

We will create a filter that includes Key Agents that have the metadata type Cloud as the value AWS with `POST /filters`. Please make sure to note down the filter response's UniqueID.

Bash

```
curl -X POST https://i.nuos.io/nuapi/filters -H "Authorization: Bearer <YOUR-TOKEN>" -
↪H "Content-type: application/json"
--data '{"ProjectID": "YOUR PROJECTID", "Operator": "Equals", "FilterType": "Metadata
↪", "Value": "AWS", "Type": "Cloud"}'
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
project_id = 'YOUR PROJECTID'
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
payload = {
    "ProjectID": project_id,
    "Operator": "Equals",
    "FilterType": "Metadata",
    "Value": "AWS",
    "Type": "Cloud"
}
r = requests.post('https://i.nuos.io/nuapi/filters', json=payload, headers=headers)
response = r.json()
print("UniqueID: %s" % response['item']['UniqueID'])
```

Once the filter has been created, we need to attach the filter to the desired Source Group. We will attach the filter by calling `POST /filters/attach` and indicating that our newly created filter UniqueID needs to be linked to the recently created Source Group UniqueID.

Bash

```
curl -X POST https://i.nuos.io/nuapi/filters/attach -H "Authorization: Bearer <YOUR-
↪TOKEN>" -H "Content-type: application/json"
--data '{"SourceGroupID": "YOUR SOURCE GROUP UNIQUEID", "UniqueID": "YOUR FILTER␣
↪UNIQUEID"}'
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
sg_id = 'YOUR SOURCE GROUP UNIQUEID'
filter_id = 'YOUR FILTER UNIQUEID'
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
payload = {
    "SourceGroupID": sg_id,
    "UniqueID": filter_id,
}
r = requests.post('https://i.nuos.io/nuapi/filters/attach', json=payload,␣
↪headers=headers)
print(r.json())
```

After you have attached the Filter to the Source Group, all Key Agents that fit the filter's rule will be included in the Source Group.

3.3.6 Setting Up Destination Groups

Note: If you have other methods of traffic mirroring or are running the Decryptor on the same instance as the Key Agent, then you do not need a Destination Group.

A Destination Group needs to be created so that the Key Agent traffic mirroring services understand to which destination to send the traffic to.

We will create a Destination Group through `POST /destgroups`. Please make sure to note down the Destination Group response's UniqueID.

Bash

```
curl -X POST https://i.nuos.io/nuapi/destgroups -H "Authorization: Bearer <YOUR-TOKEN>
↪" -H "Content-type: application/json"
--data '{"ProjectID": "YOUR PROJECTID", "Name": "Tool", "DestinationList": ["YOUR␣
↪DESTINATION EC2 IP"]}'
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
project_id = 'YOUR PROJECTID'
destination_list = ["YOUR DESTINATION EC2 IP"]
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
payload = {
    "ProjectID": project_id,
    "DestinationList": destination_list,
    "Name": "Tool",
}
r = requests.post('https://i.nuos.io/nuapi/destgroups', json=payload, headers=headers)
response = r.json()
print("UniqueID: %s" % response['item']['UniqueID'])
```

3.3.7 Creating a Connection

Note: If you have other methods of traffic mirroring or are running the Decryptor on the same instance as the Key Agent, then you do not need a Connection.

A Connection will allow the Key Agent to tap traffic to the destination specified when the Destination Group was created.

We will create a Connection through `POST /connections`. You are asked to input the TapType and TapID, but these values are irrelevant when you want to do TLS decryption. You may leave the TapType as "VXLAN" and TapID as any number.

Bash

```
curl -X POST https://i.nuos.io/nuapi/connections -H "Authorization: Bearer <YOUR-
↪TOKEN>" -H "Content-type: application/json"
--data '{"ProjectID": "YOUR PROJECTID", "SourceGroupID": "YOUR SOURCE GROUP UNIQUEID",
↪ "DestinationGroupID": "YOUR DESTINATION GROUP UNIQUEID", "TapType": "VXLAN", "TapID
↪": "1"}'
```

Python

```
import requests

token = 'REPLACE WITH YOUR TOKEN'
project_id = 'YOUR PROJECTID'
sg_id = "YOUR SOURCE GROUP UNIQUEID"
dg_id = "YOUR DESTINATION GROUP UNIQUEID"
headers = {'Authorization': 'Bearer {TOKEN}'.format(TOKEN=token)}
payload = {
    "ProjectID": project_id,
    "SourceGroupID": sg_id,
    "DestinationGroupID": dg_id,
    "TapType": "VXLAN",
    "TapID": "1",
}
r = requests.post('https://i.nuos.io/nuapi/connections', json=payload,␣
↪headers=headers)
response = r.json()
print("UniqueID: %s" % response['item']['UniqueID'])
```

3.3.8 See Traffic Decrypting!

Your environment is all set up! The Key Agent should have started to extract keys and the Decryptor should have begun to decrypt the packets that are coming in through its designated network interface.

On your `Destination` instance you can start a `tcpdump` to see decrypted traffic:

```
tcpdump -Ani nurx0 port 80
```

On your `Source` instance you can generate some traffic so that the Decryptor can decrypt the traffic:

```
# run some https traffic on the client
curl https://example.com
```

CHAPTER 4

Accounts

4.1 /accounts

4.1.1 GET

4.1.1.1 Description

To get accounts.

4.1.1.2 Response Example


```
{
    "status": "success",
    "items": [
        {
            "CreationDate": 1546973194967,
            "ModifiedDate": 1546973194967,
            "UniqueID": "1529227729398x327613111017111017",
            "AccountID": "jeviedUL91"
        }
    ],
    "response": {
        "err": ""
    }
}
```

CHAPTER 5

Connections

5.1 /connections

5.1.1 GET

5.1.1.1 Description

To get a connection.

5.1.1.2 Optional Query Strings for GET

Name	Description	Data Type
uniqueid	UniqueID of the Connection.	String

5.1.1.3 Request Query String Example

```
?uniqueid="1529227729398x327613111017111017"
```

5.1.1.4 Response Example

```
{
    "status": "success",
    "items": [
        {
            "Description": "This is a sample description",
            "TapID": "42",
            "DestinationGroupID": "1529227729398x327613111017111017",
```

(continues on next page)

(continued from previous page)


```
            "ProjectID": "1529227729398x327613111017111017",
            "BPF": "not port 22",
            "UniqueID": "1529227729398x327613111017111017",
            "ModifiedDate": 1556318922110,
            "SourceGroupID": "1529227729398x327613111017111017",
            "TapType": "VXLAN",
            "CreationDate": 1556318922110,
            "AccountID": "jeviedUL91"
        },
    ],
    "response": {
        "err": ""
    }
}
```


5.1.2 POST

5.1.2.1 Description

To create a new connection.

You are asked to input the TapType and TapID, but these values are irrelevant when you want to do TLS decryption. For TLS decryption, you may leave the TapType as "VXLAN" and TapID as any number.

5.1.2.2 Request Body Parameters

Name	Description	Required	Data Type
ProjectID	User's ProjectID.	Yes	String
SourceGroupID	The Source Group UniqueID that you want to include in the connection.	Yes	String
DestinationGroupID	The Destination Group UniqueID that you want to include in the connection.	Yes	String
TapType	There are only two options: "VXLAN", "GRE". Case sensitive. This TapType is only required when Nubeva's packet brokering service is used.	Yes	String
TapID	ID of the tap that is being used. This ID can be any number if Nubeva's packet brokering service is not used.	Yes	String

5.1.2.3 Request Payload Example


```
{
    "ProjectID": "1529227733333x927613333000111017",
    "SourceGroupID": "1529227733123x827613333123111017",
    "DestinationGroupID": "1529227729398x327613111017111017",
    "TapType": "VXLAN",
    "TapID": "1",
}
```

5.1.2.4 Response Example


```
{
    "status": "success",
    "item": {
        "Description": "Connection Description",
        "TapID": "1",
        "DestinationGroupID": "1529227729398x327613111017111017",
        "ProjectID": "1529227733333x927613333000111017",
        "BPF": "icmp",
        "UniqueID": "1565238777402x926425807383818768545090",
        "ModifiedDate": 1565238777614,
        "SourceGroupID": "1529227733123x827613333123111017",
        "TapType": "VXLAN",
        "CreationDate": 1565238777614,
        "AccountID": "jeviedUL91"
    },
    "response": {
        "err": "",
        "uniqueid": "1565238777402x926425807383818768545090"
    }
}
```


5.2 /connections/delete

5.2.1 POST

5.2.1.1 Description

To delete a connection.

5.2.1.2 Request Body Parameters

Name	Description	Required	Data Type
UniqueID	The uniqueid of the Connection.	Yes	String

5.2.1.3 Request Payload Example


```
{
    "UniqueID": "1529227733333x927011101703333000",
}
```


5.2.1.4 Response Example


```
{
    "status": "success",
    "response": {
        "msg": "Deleted",
```


(continues on next page)

(continued from previous page)

```
        "uniqueid": "1529227733333x927011101703333000"
    }
}
```

CHAPTER 6

Decryptors

6.1 /decryptors

6.1.1 GET

6.1.1.1 Description

To get decryptor agents.

6.1.1.2 Query Strings for GET

Name	Description	Required	Data Type
uniqueid	UniqueID of the Decryptor.	No	String
projectid	ProjectID of the Project that the Decryptor belongs to.	Yes	String

6.1.1.3 Request Query String Example

```
?uniqueid="1529227729398x327613111017111017"&projectid=
↪"1560938293x144687868664388668111017"
```

6.1.1.4 Response Example

```
{
    "status": "success",
    "items": [
        {
            "VPCID": "vpc-d910f4b3",
```

(continues on next page)

(continued from previous page)


```
            "LastReceived": 1565153783158,
            "InstanceID": "i-0089033370e111017",
            "Alerts": [
                "Testing Error",
            ],
            "PublicHostname": "",
            "CPUFlags": [],
            "Latitude": "36.0588",
            "OperatingSystem": "Debian",
            "AccountID": "jeviedUL91",
            "CloudAccountID": "301110170490",
            "FQDN": "",
            "NuTags": [
                "1555461210x2111017544271637815128",
            ],
            "SecurityGroups": "",
            "Address": "3 Jevie Road, San Jose, CA 95124, USA",
            "SystemHostname": "",
            "InstanceType": "t3.small",
            "ModifiedDate": 1565153783158,
            "PrismsLabel": "",
            "PublicIPv4": "34.216.31.125",
            "Filters": null,
            "CloudRegion": "us-west-2",
            "Hypervisor": "",
            "Longitude": "-115.3104",
            "Bps": 0,
            "NetworkID": "/subscriptions/2a111017-d04f-4193-b156-19f23111017/
↪resourceGroups/nub/providers/Microsoft.Network/virtualNetworks/test",
            "AvailabilityZone": "us-west-1b",
            "CPUModel": "",
            "ProjectID": "1560938293x144687868664388668111017",
            "Disabled": null,
            "CPUArch": "x86_64",
            "IPv4": "172.31.36.118",
            "UniqueID": "1565153458x6873421110174754260084",
            "IPv6": "",
            "CreationDate": 1565153458364,
            "PrivateHostname": "ip-172-31-3-115",
            "Cloud": "AWS"
        }
    ],
    "response": {
        "err": ""
    }
}
```


6.2 /decryptors/commands

6.2.1 GET

6.2.1.1 Description

To get the launch command for your Project's decryptors.

6.2.1.2 Query Strings for GET

Name	Description	Required	Data Type
projectid	ProjectID of the Project that the Decryptor belongs to.	Yes	String

6.2.1.3 Request Query String Example

```
?projectid="1560938293x144687868664388668111017"
```

6.2.1.4 Response Example

```
{
    "status": "success",
    "item": {
        "linux": "docker run -v /:/host -v /var/run/docker.sock:/var/run/docker.sock -
↪-cap-add NET_ADMIN --name nubeva-rx -d --restart=on-failure --net=host nubeva/nurx -
↪-accept-eula --nutoken jeviejksky_
↪1u7XE3XTJl3ajevie111017jdGLngGL1T3JdG77jxaVaGdOoxqDUd9lT3goqlGaXljG"
    },
    "response": {
        "err": ""
    }
}
```

CHAPTER 7

Destination Groups

7.1 /destgroups

7.1.1 GET

7.1.1.1 Description

To get destination groups.

7.1.1.2 Optional Query Strings for GET

Name	Description	Data Type
uniqueid	UniqueID of the Destination Group.	String

7.1.1.3 Request Query String Example

```
?uniqueid="1529227729398x3276131110171111017"
```

7.1.1.4 Response Example

```
{
    "status": "success",
    "items": [
        {
            "Name": "Jevie's ntop",
            "DestinationList": [
                "10.0.1.29"
```

(continues on next page)

(continued from previous page)


```
            ],
            "ProjectID": "1529227729398x311101751301452254077147",
            "ModifiedDate": 1550559676121,
            "UniqueID": "1529227729398x327613111017111017",
            "CreationDate": 1550559676121,
            "BpsList": [1, 2, 3],
            "AccountID": "jeviedUL91"
        },
    ],
    "response": {
        "err": ""
    }
}
```


7.1.2 POST

7.1.2.1 Description

To create a new destination group.

7.1.2.2 Request Body Parameters

Name	Description	Required	Data Type
ProjectID	User's ProjectID.	Yes	String
Name	Name of the Destination Group.	Yes	String
DestinationList	A list containing one or more destinations. Destinations can be any domain to IPv4 and IPv6 address.	Yes	List of Strings

7.1.2.3 Request Payload Example


```
{
    "ProjectID": "1529227733333x927613333000111017",
    "DestinationList": ['11.10.17.3', '2001:0db8:85a3:0000:0000:8a2e:0370:7114'],
    "Name": "Netflow Tool",
}
```


7.1.2.4 Response Example


```
{
    "status": "success",
    "item": {
        "Name": "Netflow Tool",
        "DestinationList": [
            "11.10.17.3",
            "2001:0db8:85a3:0000:0000:8a2e:0370:7114"
        ],
        "ProjectID": "1529227733333x927613333000111017",
```


(continues on next page)

(continued from previous page)


```
        "ModifiedDate": 1565238290612,
        "UniqueID": "1529227729398x327613111017111017",
        "CreationDate": 1565238290612,
        "BpsList": null,
        "AccountID": "jeviedUL91"
    },
    "response": {
        "err": "",
        "uniqueid": "1529227729398x327613111017111017"
    }
}
```


7.2 /destgroups/delete

7.2.1 POST

7.2.1.1 Description

To delete a destination group.

7.2.1.2 Request Body Parameters

Name	Description	Required	Data Type
UniqueID	The uniqueid of the Source Group.	Yes	String
ProjectID	The ProjectID that the Source Group belongs in.	Yes	String

7.2.1.3 Request Payload Example


```
{
    "UniqueID": "1529227729398x327613111017111017",
    "ProjectID": "1529227729398x327613111017111017"
}
```


7.2.1.4 Response Example


```
{
    "status": "success",
    "response": {
        "msg": "Deleted",
        "uniqueid": "1529227729398x327613111017111017"
    }
}
```

CHAPTER 8

Filters

8.1 /filters

8.1.1 GET

8.1.1.1 Description

To get filters.

Filters are used to filter which key agents should be included in the source group. Each filter is combined by an AND.

8.1.1.2 Optional Query Strings for GET

Name	Description	Data Type
uniqueid	UniqueID of the filter.	String

8.1.1.3 Request Query String Example

```
?uniqueid="1529227729398x327613111017111017"
```

8.1.1.4 Response Example

```
{
    "status": "success",
    "items": [
        {
            "ProjectID": "1529227729398x327613111017111017",
            "FilterType": "Metadata",
```

(continues on next page)

(continued from previous page)


```
            "Value": "ami-005bdb005fb00e791",
            "ModifiedDate": 1559797380836,
            "UniqueID": "1529227729398x327613111017111017",
            "Operator": "Equals",
            "CreationDate": 1559797380836,
            "Type": "AMI",
            "AccountID": "jeviedUL91"
        },
    ],
    "response": {
        "err": ""
    }
}
```


8.1.2 POST

8.1.2.1 Description

To create a new filter. You many only create filters with values belonging to active key agents. Active key agents have to be running at the point of time the filter with desired values are created.

Filters are used to filter which key agents should be included in the source group. Each filter is combined by an AND.

8.1.2.2 Request Body Parameters

Name	Description	Required	Data Type
ProjectID	User's ProjectID.	Yes	String
Operator	Indicates if the operator between Type and Value. Only TWO Options are available: "Equals" / "Not Equals"	Yes	String
FilterType	Indicates whether to use Metadata or Custom Tags for the filter. Only TWO Options are available: "Metadata" / "Custom Tags"	No	String
Value	Any value that can be found through /nuapi/filters/options.	Yes	String
Type	Any type that can be found through /nuapi/filters/options.	Yes	String

8.1.2.3 Request Payload Example

```
{
    "ProjectID": "1529227733333x927613333000111017",
    "Operator": "Equals",
    "FilterType": "Metadata",
    "Value": "AWS",
    "Type": "Cloud",
}
```

8.1.2.4 Response Example

```
{
    "status": "success",
    "item": {
        "ProjectID": "1529227733333x927613333000111017",
        "FilterType": "Metadata",
        "Value": "AWS",
        "ModifiedDate": 1565239827549,
        "UniqueID": "1529227729398x327613111017111017",
        "Operator": "Equals",
        "CreationDate": 1565239827549,
        "Type": "Cloud",
        "AccountID": "jeviedUL91"
    },
    "response": {
        "err": "",
        "uniqueid": "1529227729398x327613111017111017"
    }
}
```

8.2 /filters/delete

8.2.1 POST

8.2.1.1 Description

To delete a filter.

8.2.1.2 Request Body Parameters

Name	Description	Required	Data Type
UniqueID	The uniqueid of the Filter.	Yes	String

8.2.1.3 Request Payload Example

```
{
    "UniqueID": "1529227733123x111017827613333123",
}
```

8.2.1.4 Response Example

```
{
    "status": "success",
    "response": {
        "msg": "Deleted",
        "uniqueid": "1529227733123x111017827613333123"
    }
}
```

8.3 /filters/attach

8.3.1 POST

8.3.1.1 Description

To attach a filter to a Source Group.

Filters are used to filter which key agents should be included in the source group. Each filter is combined by an AND.

8.3.1.2 Request Body Parameters

Name	Description	Required	Data Type
SourceGroupID	The Source Group UniqueID that you want to attach the filter to.	Yes	String
UniqueID	The Filter UniqueID.	Yes	String

8.3.1.3 Request Payload Example

```
{
    "SourceGroupID": "1529227733123x82761333312311101 7",
    "UniqueID": "1529227733123x111017827613333123",
}
```

8.3.1.4 Response Example

```
{'response': {'msg': 'Attached',
              'uniqueid': '1529227733123x111017827613333123'},
 'status': 'success'}
```

8.4 /filters/detach

8.4.1 POST

8.4.1.1 Description

To detach a Filter from a Source Group

8.4.1.2 Request Body Parameters

Name	Description	Required	Data Type
SourceGroupID	The Source Group UniqueID that you want to attach the filter to.	Yes	String
UniqueID	The Filter UniqueID.	Yes	String

8.4.1.3 Request Payload Example

```
{
    "SourceGroupID": "1529227733123x827613333123111017",
    "UniqueID": "1529227733123x111017827613333123",
}
```

8.4.1.4 Response Example

```
{'response': {'msg': '',
              'uniqueid': '1565010927786x603119357640980715321111'},
 'status': 'success'}
```

8.5 /filters/options

8.5.1 POST

8.5.1.1 Description

To identify what values are available for filters. The response will only give back filter options and values for metadata and NuTags belonging to active key agents. If you do not see any options, please make sure that your key agents are actively running.

8.5.1.2 Request Body Parameters

Name	Description	Required	Data Type
ProjectID	User's ProjectID that you want to find all available Metadata / Custom Tags values for.	Yes	String

8.5.1.3 Request Payload Example

```
{
    "ProjectID": "1529227733333x927613333000111017",
}
```

8.5.1.4 Response Example

```
{
    "status": "success",
    "CustomTags": [
        {
            "Type": "Name",
            "Value": "TLS Ubuntu 18.04 t3"
        }
    ],
```

(continues on next page)

(continued from previous page)


```
    "response": {
        "err": ""
    },
    "Metadata": [
        {
            "Type": "CloudAccountId",
            "Value": [
                "301110170490"
            ]
        },
        {
            "Type": "PublicIPv4",
            "Value": [
                "54.182.211.80"
            ]
        },
        {
            "Type": "InstanceID",
            "Value": [
                "i-0089033370e501234"
            ]
        },
        {
            "Type": "PublicHostname",
            "Value": [
                "",
                "ec2-54-182-211-80.us-west-2.compute.amazonaws.com"
            ]
        },
        {
            "Type": "CloudRegion",
            "Value": [
                "us-west-2"
            ]
        },
        {
            "Type": "VPCId",
            "Value": [
                "vpc-4a3a7123"
            ]
        },
        {
            "Type": "CIDR",
            "Value": [
                "172.31.26.123"
            ]
        },
        {
            "Type": "OperatingSystem",
            "Value": [
                "Debian"
            ]
        },
        {
            "Type": "AMI",
            "Value": [
                "ami-005bdb005fb00e791"
```


(continues on next page)

(continued from previous page)

```
                ]
            },
            {
                "Type": "AvailabilityZone",
                "Value": [
                    "us-west-2a"
                ]
            },
            {
                "Type": "PrivateHostname",
                "Value": [
                    "ip-172-31-26-123"
                ]
            },
            {
                "Type": "CPUArch",
                "Value": [
                    "x86_64"
                ]
            },
            {
                "Type": "IPv4",
                "Value": [
                    "172.31.26.123"
                ]
            },
            {
                "Type": "InstanceType",
                "Value": [
                    "t3.small"
                ]
            },
            {
                "Type": "Cloud",
                "Value": [
                    "AWS"
                ]
            }
    ]
}
```

CHAPTER 9

Key Agents

9.1 /keyagents

9.1.1 GET

9.1.1.1 Description

To get key agents.

9.1.1.2 Query Strings for GET

Name	Description	Required	Data Type
uniqueid	UniqueID of the Key Agent.	No	String
projectid	ProjectID of the Project that the Key Agent belongs to.	Yes	String

9.1.1.3 Request Query String Example

```
?uniqueid="1529227729398x327613111017111017"&projectid=
↪"1560938293x144687868664388668111017"
```

9.1.1.4 Response Example

```
{
    "status": "success",
    "items": [
        {
            "VPCID": "vpc-d910f4b3",
```

(continues on next page)

(continued from previous page)


```
            "LastReceived": 1565153783158,
            "InstanceID": "i-0089033370e111017",
            "Alerts": [
                "Testing Error",
            ],
            "PublicHostname": "",
            "CPUFlags": [],
            "Latitude": "36.0588",
            "OperatingSystem": "Debian",
            "AccountID": "jeviedUL91",
            "CloudAccountID": "301110170490",
            "FQDN": "",
            "NuTags": [
                "1555461210x2111017544271637815128",
            ],
            "SecurityGroups": "",
            "Address": "3 Jevie Road, San Jose, CA 95124, USA",
            "SystemHostname": "",
            "InstanceType": "t3.small",
            "ModifiedDate": 1565153783158,
            "PrismsLabel": "",
            "PublicIPv4": "34.216.31.125",
            "Filters": null,
            "CloudRegion": "us-west-2",
            "Hypervisor": "",
            "Longitude": "-115.3104",
            "Bps": 0,
            "NetworkID": "/subscriptions/2a111017-d04f-4193-b156-19f23111017/
↪resourceGroups/nub/providers/Microsoft.Network/virtualNetworks/test",
            "AvailabilityZone": "us-west-1b",
            "CPUModel": "",
            "ProjectID": "1560938293x144687868664388668111017",
            "Disabled": null,
            "CPUArch": "x86_64",
            "IPv4": "172.31.36.118",
            "UniqueID": "1565153458x6873421110174754260084",
            "IPv6": "",
            "CreationDate": 1565153458364,
            "PrivateHostname": "ip-172-31-3-115",
            "Cloud": "AWS"
        }
    ],
    "response": {
        "err": ""
    }
}
```


9.2 /keyagents/commands

9.2.1 GET

9.2.1.1 Description

To get the launch command for your Project's key agents.

9.2.1.2 Query Strings for GET

Name	Description	Required	Data Type
projectid	ProjectID of the Project that the Key Agent belongs to.	Yes	String

9.2.1.3 Request Query String Example

```
?projectid="1560938293x144687868664388668111017"
```

9.2.1.4 Response Example

```
{
    "status": "success",
    "item": {
        "linux": "docker run -v /:/host -v /var/run/docker.sock:/var/run/docker.sock -
↪-cap-add NET_ADMIN --cap-add SYS_ADMIN --cap-add SYS_RESOURCE --name nubeva-agent -
↪d --restart=on-failure --net=host nubeva/nuagent --accept-eula --nutoken jeviejksky_
↪1u7XE3XTJl3ajevie111017jdGLngGL1T3JdG77jxaVaGdOoxqDUd9lT3goqlGaXljG"
    },
    "response": {
        "err": ""
    }
}
```

CHAPTER 10

Projects

10.1 /projects

10.1.1 GET

10.1.1.1 Description

To get projects.

10.1.1.2 Optional Query Strings for GET

Name	Description	Data Type
uniqueid	UniqueID of the Project.	String

10.1.1.3 Request Query String Example

```
?uniqueid="1529227729398x327613111017111017"
```

10.1.1.4 Response Example

```
{
    "status": "success",
    "items": [
        {
            "SrcGroups": [
                "1560938293x144687868664118668111017",
            ],
```

(continues on next page)

(continued from previous page)


```
            "Description": "",
            "CredObj": {
                "Domain": "NubevaCreds-DDBTable-11AAQEX4K3HEN",
                "Type": "ddb",
                "Region": "us-west-2"
            },
            "DestGroups": [
                "1560938293x144687868664388138111017",
            ],
            "ModifiedDate": 1560938293191,
            "UniqueID": "1560938293x144687868664388668111017",
            "Name": "asdf",
            "SensorKey": "jeviejksky",
            "CreationDate": 1560938293191,
            "ProjectKey":
↪"1u7XE3XTJl3ajevie111017jdGLngGL1T3JdG77jxaVaGdOoxqDUd9lT3goqlGaXljG",
            "AccountID": "jeviedUL91"
        },
    ],
    "response": {
        "err": ""
    }
}
```

CHAPTER 11

Source Groups

11.1 /srcgroups

11.1.1 GET

11.1.1.1 Description

To get source groups.

11.1.1.2 Optional Query Strings for GET

Name	Description	Data Type
uniqueid	UniqueID of the Source Group.	String

11.1.1.3 Request Query String Example

```
?uniqueid="1529227729398x327613111017111017"
```

11.1.1.4 Response Example

```
{
    "status": "success",
    "items": [
        {
            "Filters": [
                "1559321827x41671886568381528587660 0"
            ],
```

(continues on next page)

(continued from previous page)


```
            "KeyAgentList": [
                "1550559598x763183078525274495646546",
            ],
            "Description": "",
            "ProjectID": "1529227729398x327613111017111017",
            "TLS": true,
            "ModifiedDate": 1559321836602,
            "UniqueID": "1529227729123x327613111017111017",
            "Name": "aws",
            "CreationDate": 1559321836602,
            "AccountID": "jeviedUL91"
        }
    ],
    "response": {
        "err": ""
    }
}
```


11.1.2 POST

11.1.2.1 Description

To create a new source group. Source groups include the active key agents that are running.

11.1.2.2 Request Body Parameters

Name	Description	Required	Data Type
ProjectID	User's ProjectID.	Yes	String
Name	Name of the Source Group.	Yes	String
Description	Description of the Source Group.	No	String

11.1.2.3 Request Payload Example


```
{
    "ProjectID": "1529227733333x927613333000111017",
    "Name": "Internal-1",
    "Description": "Here is my personal internal instance",
}
```


11.1.2.4 Response Example


```
{
    "status": "success",
    "item": {
        "TLS": true,
        "Description": "Here is my personal internal instance",
        "KeyAgentList": [
            "1529227729398x327613111017111017",
```


(continues on next page)

(continued from previous page)


```
        ],
        "ProjectID": "1529227729398x327613111017111017",
        "ModifiedDate": 1565237303629,
        "UniqueID": "1529227729123x327613111017111017",
        "Filters": [
            "1529227733123x111017827613333123",
        ],
        "Name": "Internal-1",
        "CreationDate": 1565237303629,
        "AccountID": "jeviedUL91"
    },
    "response": {
        "err": "",
        "uniqueid": "1529227729123x327613111017111017"
    }
}
```


11.2 /srcgroups/delete

11.2.1 POST

11.2.1.1 Description

To delete a source group.

11.2.1.2 Request Body Parameters

Name	Description	Required	Data Type
UniqueID	The uniqueid of the Source Group.	Yes	String
ProjectID	The ProjectID that the Source Group belongs in.	Yes	String

11.2.1.3 Request Payload Example


```
{
    "UniqueID": "1529227733123x827613333123111017",
    "ProjectID": "1529227729398x327613111017111017"
}
```


11.2.1.4 Response Example


```
{
    "status": "success",
    "response": {
        "msg": "Deleted",
        "uniqueid": "1529227733123x827613333123111017"
    }
}
```

CHAPTER 12

Tokens

12.1 /tokens

12.1.1 POST

12.1.1.1 Description

To create a new token. Please store token safely because you cannot retrieve the same token again.

The provided token value will be the Authorization Bearer Header token.

NOTE: This endpoint does NOT require the Authorization header.

12.1.1.2 Request Body Parameters

Name	Description	Required	Data Type
Email	The User's email. Must have an account with Nubeva.	Yes	String
AccountID	The User's AccountID	Yes	String

12.1.1.3 Request Payload Example

```
{
    "Email": "nubeva@example.com",
    "AccountID": "jeviedUL91"
}
```

12.1.1.4 Response Example


```
{
    "status": "success",
    "response": {
        "Token":
↪"c1V0TzElI3hUYUZIeTQhREhzOUBPMWpVU1F4TSZEWXVjeHgmc1M1UktLSndpRFAmJmpqWnFtMUVWM0NhUGclamZEI3E2OWpBZ0
↪",
        "err": "",
        "TokenType": "Bearer"
    }
}
```


12.2 /tokens/delete

12.2.1 POST

12.2.1.1 Description

To delete a token.

12.2.1.2 Request Body Parameters

Name	Description	Required	Data Type
Token	The token provided by Nubeva to authenticate the user to perform API actions.	Either Token or UniqueID is required	String

12.2.1.3 Request Payload Example


```
{
    "Token":
↪"c1V0TzElI3hUYUZIeTQhREhzOUBPMWpVU1F4TSZEWXVjeHgmc1M1UktLSndpRFAmJmpqWnFtMUVWM0NhUGclamZEI3E2OWpBZ0
↪",
}
```


12.2.1.4 Response Example


```
{
    "status": "success",
    "response": {
        "msg": "Deleted"
    }
}
```