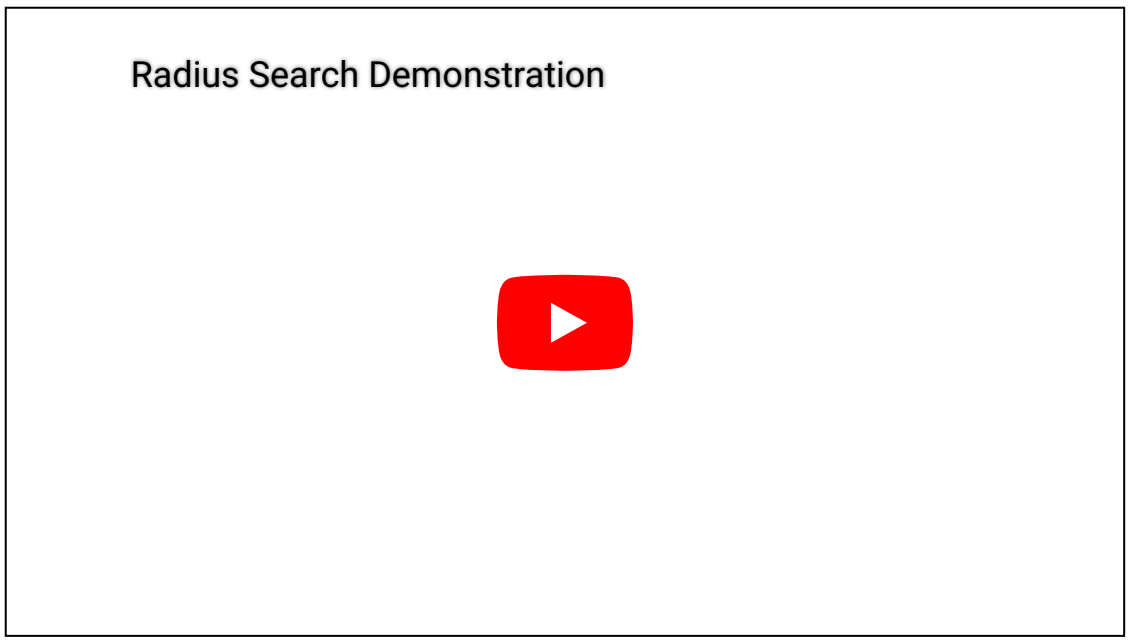
- On the right hand side of the page, click on search.
- Scroll down to view the information you searched for. The search result will bring up facilities which are in that country, city and state.
- You can download your information in a JSON or CSV format.



Querying with the PeeringDB API

Throughout this article up to this point, we have been talking on how to use PeeringDB to find information about potential peers, and then after peering has been arranged, using PeeringDB to obtain the peering details. The PeeringDB website is very helpful in these regards, but using the website still requires a lot of manual work. It's also possible to use the PeeringDB API to automate some parts of this. Why use the PeeringDB API? The PeeringDB API makes it easy to integrate PeeringDB in your environment.

The PeeringDB database can be queried using a REST API. REST allows a client to request information from a server over HTTP or HTTPS. The server then returns the requested information in JSON format.

Object types

Each object has an associated shorthand tag you can use. Object types are not case sensitive and the output is an array.
For example: https://www.peeringdb.com/api/OBJ. The endpoint is: `/api/OBJ`.

Below are the categories of objects types (OBJ) in PeeringDB:
- **Basic Objects**: org, fac, ix, net, poc
- **Derived Objects**: ixlan, ixpfx, netixlan, netfac

Basic objects

Below is a description of what each of the object types mean and what information they return

- **org**: Root object for fac, ix, net, this holds information about an organization.
- **fac**: Describes a facility / colocation record, more useful information are in derived records netfac.
- **ix**: Describes an exchange, more useful information are in derived records ixlan, ixpfx and netixlan.
- **net**: Describes a network / ASN, more useful information are in netfac and netixlan.
- **poc**: Describes various role accounts (point of contact), this is currently only for net objects.
- **as_set**: Array of all AS-SETs corresponding to a network / ASN, this was introduced recently.

Derived objects

Below is a description of what each of the object types mean and what information they return.

- **Ixlan**: Describes the LAN of an ix, one ix may have multiple ixlan. This feature may go away in PeeringDB 3.0.
- **ixpfx**: Describes the IP range (IPv4 and IPv6) for an ixlan, one ixlan may have multiple ixpfx.
- **netixlan**: Describes the presence of a network at an exchange.
- **netfac**: Describes the presence of a network at a facility.

Authentication

Authentication is done through basic HTTP authorization. People who are accessing the API as a guest do not need any authentication. For example:

```
curl -sG https://username:password@www.peeringdb.com/api/net/961
```

```
curl -u username:password https://www.peeringdb.com/api/net/961
```

Note: Access to contact information may be restricted if you are using the API as a guest without authentication. API usage is subject to query limits and these are set at a lower threshold for unauthenticated users.

Making a request

When making a request, the URL base is added with `/api/`, followed by the object type and, if applicable, the object primary key (if applicable). For example: `https://www.peeringdb.com/api/OBJ/id`.

If you want to select the output format, either use the Accept: HTTP header or use the extension type parameter:
- Accept Header: `Accept: application/json`
- Extension type: `https://www.peeringdb.com/api/network/42.json`

Operations

Using the GET operation you can retrieve information from the PeeringDB database. You can retrieve both a single object and multiple objects in an array. Let's look at each of them individually.

Single object

To retrieve a single object you need to use this URL: `https://www.peeringdb.com/api/OBJ/ID` with this endpoint GET: `/api/OBJ/id`. The ID is a unique identifier and should be added to the URL when retrieving a single object. Let's look at an example:
- HTTP: `GET /api/OBJ/38` where 38 is the ID
- curl: `curl -H "Accept: application/json" -X GET https://<username>:<password>@peeringdb.com/api/OBJ/38`

There are optional parameters you can add to your URL:
- **str**: which retrieves a comma separated list of field names - only matching fields will be returned in the data
- **int**: which retrieves two nested sets and objects

Nested sets and objects

A nested set or object is any field ending in the suffix: set. For example: net_set will hold network objects. The naming schema of the field will always tell you which type of object the set is holding and will correspond with the object's endpoint on the API `<object_type>_set`. So a set called `net_set` will hold network objects (API endpoint /net).

Note: unlike GET multiple, `depth` here will also expand single relationships in addition to sets. So `net_id` would get expanded into a network object.

Unexpanded

```
{
  ...
  "net_id" : 1
}
```

Expanded

```
{
  ...
  "net_id" : 1
  "net" : {
     ... network object ...
  }
}
```

Depth
- 0: don't expand anything (default)
- 1: expand all first level sets to ids
- 2: expand all first level sets to objects

Multiple objects

To retrieve a single object you need to use this URL: `https://www.peeringdb.com/api/OBJ/` with this endpoint GET: `/api/OBJ/`.
Let's look at an example:
- HTTP: `GET /api/OBJ/` which is the endpoint
- curl: `curl -X GET https://<username>:<password>@www.peeringdb.com/api/OBJ`

There are optional parameters you can add to your URL:
- **limit**: int limits rows in the result set
- **skip**: int skips n rows in the result set
- **depth**: int nested sets will be loaded (slow)

- **fields**: str comma separated list of field names - only matching fields will be returned in the data
- **since**: int retrieve all objects updated since specified time (unix timestamp, seconds)
- **[field_name]**: int|string queries for fields with matching value

Real world use cases

We will show you different use cases on how to use the PeeringDB API.

How do I query by ASN?

To query the ASN 42 using PeeringDB API, you will need to use this URL: GET `https://www.peeringdb.com/api/net?asn=42`, where `asn=42` is the query parameter.

Using curl

Use this curl example to get this specific network. Copy and paste the following to your command line interface: `curl GET https://www.peeringdb.com/api/net?asn=42`.

Using Python

To make use of this Python code, first, you'll have to first install Python if you don't have it installed. Then, install pip and requests. After that create a Python file and copy and paste the following code.

```
import requests
r = requests.get('https://www.peeringdb.com/api/net?asn=42')
print(r.text)

with open('output.csv', 'w+') as f:
    f.write(r.text)
```

From the above code, we make a request to the API using the request module and print out the response which would be in a JSON format. However, reading a JSON file can be quite hectic and tasking so we convert the JSON file to a CSV file. Our CSV file will open in output.csv.

Using jq

You can use jq to make a request to your API and get your output in a CSV format. First, you need to install Jq. Next, we use this curl command to prepare our JSON file. Change to a directory and copy and paste this code on your terminal: `curl https://www.peeringdb.com/api/net?asn=42 > test.json`.

This creates a new file named test.json. To convert the JSON input file to the CSV format, copy and paste the following command: `jq -r '(.data[0] | keys_unsorted), (.data[] | to_entries | map(.value))|@csv' test.json`

Using an online converter

Alternatively you can use an online tool such as https://www.convertcsv.com/json-to-csv.htm to convert the raw JSON file to CSV.

Note: For the purpose of this article we will focus on the curl method but you can conveniently try out the other proposed methods.

How to get all the objects owned by https://www.peeringdb.com/net/961 and convert the data to CSV?

To get all the objects owned by this https://www.peeringdb.com/net/961 using the PeeringDB API. Copy and paste the following to your command line interface: `curl GET https://www.peeringdb.com/net/961`.

I want the list of networks and their 'type' peering at MICE in Minneapolis.

To get the list of networks and their type peering at Mice in Minneapolis, copy and paste the following command in your terminal: `curl -s -X GET https://www.peeringdb.com/api/net?ix=446&__in=Minneapolis | jq '.data[]`.

How to find all the exchanges where my organization has a presence?

Use this curl example. Copy and paste the following to your command line interface: `curl -s -X GET https://www.peeringdb.com/api/netixlan\?ixlan_id=62 | jq '.data[]'`.

PeeringDB