Tracking Object Changes in the Database

If you have external applications that use information in the NIOS database, you can use the Object Change Tracking feature to get informed about changes in the NIOS database. You can then periodically synchronize IPAM, DNS, and DHCP data through the Infoblox API or RESTful API, which returns updated object information. When you enable this feature, the appliance tracks the changes that are made to NIOS objects. It assigns sequence IDs to all the changed objects. These sequence IDs are incremented when there is a change in the high-level objects such as IPv4 and IPv6 fixed addresses, networks, network containers, and others. When you query using the db_objects through the Infoblox API for desired object types, the application returns all the objects of those object types that changed after the sequence ID given in the query.

When you enable the Object Change Tracking feature, you can also specify the lifetime of deleted objects in the database and the total number of objects that must be present in the database. The deleted objects that are saved in the database are purged periodically. Note that you cannot track the user who created or updated an existing object.

Infoblox supports full and incremental synchronization for these changes. Certain events such as Grid Master Candidate promotions and upgrades require a full synchronization.

Users with read-only permission can use this feature. For more information, see *Administrative Permissions for Object Change Tracking*.

To enable and use the Object Change Tracking feature to track and synchronize updates, complete the following:

1. Enable the feature, as described in *Enabling Object Change Tracking.*
2. Select whether you want to use a full synchronization or an incremental synchronization. When you use one of these synchronization methods to synchronize data through Infoblox API or RESTful API, they return updated objects that can be used to update your relational database. For full synchronization, see *Using Full Synchronization*. For incremental synchronization, see *Using Incremental Synchronization.*

Best Practices for Object Change Tracking

- The Object Change Tracking feature is optimized to reduce impact on the DDI services and it runs only on the Grid Master. The synchronization process synchronizes 1000 objects at a time with a 2 second pause in between. There might be a slight impact on the Grid Master Candidate as they get updates from the Grid Master. When protocol services are running on the Grid Master Candidate you might encounter a 5% drop in the protocol performance. This feature does not impact the services that are running on the Grid members.
- NIOS does not update the sequence ID of the respective parent when you modify a child object. For example, inserting an A record under DNS zone does not increase the sequence number of the zone object.
- When you delete a parent object, NIOS saves the child objects in the deleted object table. But, if the associated child objects are either resource records or leases, NIOS will neither save them in the deleted object table nor update the sequence ID for these child objects.
- If you update a parent object, then the sequence ID of its child objects will not change.
- NIOS updates the sequence ID of the host record and IPv4 and IPv6 host addresses, if there are any changes to host addresses, both IPv4 and IPv6. The sequence ID is also updated when you update the host alias records that results in increase in the sequence ID of the host record and all its child objects.

For example, if you create a host record hhh.test.com with an IPv4 address and perform an incremental synchronization to get updates, the response contains an updated host record and host address:

```
curl -k1 -u admin:infoblox -H content-type:application/json -X GET
https://10.32.2.202/wapi/v2.5/db_objects?start_sequence_id=2830689052:0\&object_types=
record:host,record:host_ipv4addr\;_return_fields=last_sequence_id,object,object_type\;
_return_type=json-pretty
```

The response is as follows:

```
[
    {
        "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDIw:38",
        "object": {
           "_ref": "record:host/ZG5zLmhvc3QkLl9kZWZhdWx0LmNvbS5rYXJqYWdpLmhvc3Qx:
hhh.test.com/default",
           "ipv4addrs": [
               {
                  "_ref":
"record:host_ipv4addr/ZG5zLmhvc3RfYWRkcmVzcyQuX2RlZmF1bHQuY29tLmthcmphZ2kuaG9zdDEuMS4x
LjEuMS4:1.1.1.1/ hhh.test.com/default",
                  "configure_for_dhcp": false,
                  "host": " hhh.test.com",
                  "ipv4addr": "1.1.1.1",
                  "mac": "11:11:11:11:11:11"
               }
           ],
           "ipv6addrs": [
               {
                  "_ref":
"record:host_ipv6addr/ZG5zLmhvc3RfYWRkcmVzcyQuX2RlZmF1bHQuY29tLmthcmphZ2kuaG9zdDEuYWE6
OmFhLg:aa%3A%3Aaa/ hhh.test.com/default",
                  "configure_for_dhcp": false,
                  "host": " hhh.test.com",
                  "ipv6addr": "aa::aa"
               }

           ],
           "name": " hhh.test.com",
           "view": "default"
     },
     "object_type": "record:host",
     "unique_id": "c326fcf8058c4022939050af96a0fdb2"
  },
  {
     "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDIx:38",
     "object": {
          "_ref":
"record:host_ipv6addr/ZG5zLmhvc3RfYWRkcmVzcyQuX2RlZmF1bHQuY29tLmthcmphZ2kuaG9zdDEuYWE6
OmFhLg:aa%3A%3Aaa/ hhh.test.com/default",
          "configure_for_dhcp": false,
          "host": " hhh.test.com",
          "ipv6addr": "aa::aa"
      },
      "object_type": "record:host_ipv6addr",
      "unique_id": "2b19a399c3f747538d879fdd33a4de32"
  },
  {
      "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDIy:38",
      "object": {
           "_ref":
"record:host_ipv4addr/ZG5zLmhvc3RfYWRkcmVzcyQuX2RlZmF1bHQuY29tLmthcmphZ2kuaG9zdDEuMS4x
LjEuMS4:1.1.1.1/ hhh.test.com/default",
          "configure_for_dhcp": false,
          "host": " hhh.test.com",
          "ipv4addr": "1.1.1.1",
          "mac": "11:11:11:11:11:11"
      },
      "object_type": "record:host_ipv4addr",
      "unique_id": "3fb8fba003d647ceac5796a28459725a"
   },
   {
      "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDIz:3956302770%3A38",
      "last_sequence_id": "3956302770:38"
   }
]
```


- When you update DNS host, host address, and host alias, NIOS updates the sequence IDs of child objects associated with these parent objects even though the changes do not affect the child objects. For example, when you update an existing comment in the host record, NIOS updates the sequence IDs of child objects associated with the respective host record.

  Example:

```
curl -k1 -u admin:infoblox -H content-type:application/json -X GET
https://10.32.2.202/wapi/v2.5/db_objects?start_sequence_id=2830689052:0\&object_types=
record:host,record:host_ipv4addr\;_return_fields=last_sequence_id,object,object_type,o
bject.comment\;_return_type=json-pretty
```

The response is as follows:

```
[
     {
          "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDI:27",
          "object": {
              "_ref":
"record:host/ZG5zLmhvc3QkLl9kZWZhdWx0LmNvbS50ZXN0LmhoaA:hhh.test.com/default",
              "comment": "hi"
           },
           "object_type": "record:host"
           "unique_id": "1ab6c989d8fd454ca06ffac6ac600fe6"
     },
     {
           "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDM:27",
           "object": {
               "_ref":
"record:host_ipv4addr/ZG5zLmhvc3RfYWRkcmVzcyQuX2RlZmF1bHQuY29tLnRlc3QuaGhoLjEuMC4wLjEu
:1.0.0.1/hhh.test.com/default",
              "configure_for_dhcp": false,
              "host": "hhh.test.com",
              "ipv4addr": "1.0.0.1"
           },
           "object_type": "record:host_ipv4addr"
           "unique_id": "1ab6c989d8fd454ca06ffac6ac600fe7"
     },
     {
           "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDQ:2830689052%3A27",
           "last_sequence_id": "2830689052:27"
     }
```

- When an IPv4 or an IPv6 host address changes, NIOS deletes the old record and creates a new record with the updated IP address. Note that new sequence IDs are generated for the associated objects. When you query a host record, NIOS displays the list of host addresses associated with it.

Example:

```
curl -k1 -u admin:infoblox -H content-type:application/json -X GET
https://10.32.2.202/wapi/v2.5/db_objects?start_sequence_id=2830689052:0\&object_types=
record:host,record:host_ipv4addr\;_return_fields=last_sequence_id,object,object_type\;
_return_type=json-pretty
```

The response is as follows:

```
[
     {
           "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDA:28",
           "object": {
               "_ref":
"record:host/ZG5zLmhvc3QkLl9kZWZhdWx0LmNvbS50ZXN0LmhoaA:hhh.test.com/default",
```

```
              "ipv4addrs": [
                  {
                    "_ref":
"record:host_ipv4addr/ZG5zLmhvc3RfYWRkcmVzcyQuX2RlZmF1bHQuY29tLnRlc3QuaGhoLjEuMC4wLjIu
:1.0.0.2/hhh.test.com/default",
                    "configure_for_dhcp": false,
                    "host": "hhh.test.com",
                    "ipv4addr": "1.0.0.2"
                  }
              ],
              "name": "hhh.test.com",
              "view": "default"
       },
       "object_type": "record:host"
       "unique_id": "1ab6c989d8fd454ca06ffac6ac600fe6"
   },
   {
        "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDE:28",
        "object": {
             "_ref": "deleted_objects/Li5kZWxldGVkX29iamVjdHMkMA",
             "object_type": "record:host_ipv4addr",
           "unique_id": "68cad3406a244aa0b34b3ea3be383598"
        },
        "object_type": "deleted_objects"
        "unique_id": "1ab6c989d8fd454ca06ffac6ac600fe4"
    },
    {
        "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDI:28",
        "object": {
            "_ref":
"record:host_ipv4addr/ZG5zLmhvc3RfYWRkcmVzcyQuX2RlZmF1bHQuY29tLnRlc3QuaGhoLjEuMC4wLjIu
:1.0.0.2/hhh.test.com/default",
            "configure_for_dhcp": false,
            "host": "hhh.test.com",
            "ipv4addr": "1.0.0.2"
         },
         "object_type": "record:host_ipv4addr"
         "unique_id": "1ab6c989d8fd454ca06ffac6ac600fe9"
    },
    {
         "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDM:2830689052%3A28",
         "last_sequence_id": "2830689052:28"
    }
```


- NIOS returns the shared records per zone when you query for them. For example, consider two zones, z1 and z2, and a shared record group, srg1, which contains a shared record sr1. The shared record group srg1 is associated with the zones z1 and z2. Hence, the shared record sr1 is shared between two zones, z1 and z2, but NIOS saves only a single copy of the shared record sr1 in the database. When you query for shared record updates, NIOS returns two records for z1 and z2, as sr1 is associated with both the zones, z1 and z2, and saves these records with a UUID and a sequence ID in the table. The response also contains new records with zone pointing to the zone it belongs to, UUID and sequence ID of the new records and a `shared_record_group`, which indicates that it is a shared record and not a normal resource record. When you enable Object Change Tracking feature, NIOS clears this table and recreates a new table for the existing shared records.

- Note that NIOS generates two leases in the database when a client sends a DHCP request to the NIOS DHCP server, which is configured with failover association. You can perform either a full or an incremental synchronization in this scenario. Infoblox recommends that you ignore the lease that is generated by the failover DHCP server with `binding_state` as `BACKUP` and consider the lease with `binding_state` as `ACTIVE`.
- File distribution service automatically removes old files from `wapi_output` directory if there is no sufficient space for new files. NIOS allocates 50% each of the file distribution area to `wapi_output` directory and normal files. For example, if the file distribution area is allocated 500 MB, then 250 MB is used for `wapi_output` directory and the remaining 250 MB is used for normal files.
- NIOS does not support WAPI paging mechanism. As the synchronization results are displayed in the order of `sequence_id`, you must `specify _max_results` to achieve paging mechanism for synchronization.

- NIOS deletes all previous scheduled tasks after a restore or an upgrade operation.

Enabling Object Change Tracking

To enable object change tracking:

1. **Grid**: From the **Grid** tab, select the **Grid Manager** tab -> **Members** tab, and then click **Grid Properties** -> **Edit** from the Toolbar.
   Standalone appliance: From the **System** tab **System Manager** tab, expand the Toolbar and click **System Properties** -> **Edit**.
2. In the *Grid Properties* editor, click **Object Change Tracking** and complete the following:
   - **Enable Object Change Tracking**: Select this check box to enable object change tracking for the objects.
   - **Status**: Indicates whether object change tracking is enabled or not. You can view what percentage of the operation is complete.

Note: This operation might take a longer time to complete when you have a large database. For example, 1 million objects on an IB-4010 appliance takes approximately 1 hour. Infoblox recommends that you enable this feature during off-peak hours.

- **Maximum number of deleted objects that will be tracked**: Specify the total number of deleted objects that must be present in the database. The minimum value is 2,000 and the maximum value is 20,000. The default value is 4,000.

3. Save the configuration.

Using Full Synchronization

Full synchronization synchronizes the entire set of objects irrespective of updates. You can perform a full synchronization only on the Grid Master and Grid Master Candidate. Note that the tasks created on the Grid Master and Grid Master Candidates will be executed on them respectively and you can download the synchronized data on to the member where the task was created. The Grid Master Candidate can execute only read-only tasks. To save the output of a full synchronization, you must specify the output location. The output of a full synchronization is in the following formats: JSON, XML, ROWJSON, or ROWXML.

You can specify one of the following output locations:

`"FILE_DISTRIBUTION"` and `"LOCAL".`

- When `_output_location = "FILE_DISTRIBUTION"`

When you set the output location as mentioned above, you must specify a file name or a prefix for the file. NIOS saves the output file in the file distribution area and displays an error message if you do not specify a file name or a prefix. For more information about file distribution area, see *Managing Directories*.

When you select the above output location for a Grid Master Candidate, you must select **Allow Upload to Grid Members** to upload files to the file distribution area. Infoblox recommends that you use the Grid Master Candidate to offload the Grid Master. You must enable a full synchronization during off-peak hours if protocol services are running on the Grid Master Candidate. For more information, see *Enabling Upload to Grid Members*.

Example:

```
curl -k1 -u admin:infoblox -X POST 'https://..../wapi/v2.5/fileop?_function=read' -H 'Content-Type:
application/json; charset=UTF-8' -d '{"_filename": "test.json", "_output_location":"FILE_DISTRIBUTION","
_object": "db_objects", "all_object_types_supported_in_version": "2.5","_encoding": "ROWJSON"}'
```

You can specify either `_object = db_objects` or `all_object_types_supported_in_version=2.5` to retrieve all objects. When you specify `_object = db_objects`, NIOS returns only standard Restful API fields as response. To retrieve all fields of all objects, you can specify `all_object_types_supported_in_version=2.5.` To fetch the output file, start file distribution service and use the following URL:

```
curl -k1 -u admin:infoblox -X GET http://<ipaddress>/wapi_output/test.json -H 'Content-Type:application/json'
```

- When `_output_location = "LOCAL"`
  - If you schedule a full synchronization and set the output location as mentioned above, a file name is generated based on the task ID and the output file is saved at another location. You do not have to specify a file name or a prefix.
  - If you do not schedule a synchronization, then NIOS returns an URL and a token, which is used for the download complete function, in the response.
  - If you do not specify an output location, NIOS saves the file in the file distribution area.
  - Use the above output location for saving full synchronization files when file distribution area does not have enough space for storage.

Example:

```
curl -k1 -u admin:infoblox -X POST 'https://127.0.0.1/wapi/v2.5/fileop?_function=read&_schedinfo.
schedule_now=true' -H 'Content-Type: application/json; charset=UTF-8' -d '{"_output_location": "LOCAL",
"_object": "db_objects", "all_object_types_supported_in_version": "2.5", "_encoding": "JSON"}'
```

NIOS saves the output file in the file distribution area if you do not specify an output location as shown in the code below:

```
curl -k1 -u admin:infoblox -X POST 'https://..../wapi/v2.5/fileop?_function=read' -H 'Content-Type:
application/json; charset=UTF-8' -d '{"_filename": "test.json", "_output_location":"FILE_DISTRIBUTION","
_object": "db_objects", "all_object_types_supported_in_version": "2.5","_encoding": "ROWJSON"}'
```

Note the following about full synchronization:

- A full synchronization is required initially for a complete snapshot of the database.
- Depending on the size of requested data, a full synchronization may take a longer time to complete. For example, when 1 million objects are requested with all fields on an IB-4010 appliance, full synchronization takes a couple of hours. When 1 million objects are requested with standard RESTful API fields on an IB-4010 appliance, full synchronization takes less than an hour.
- Infoblox recommends that you request only object types with standard RESTful API fields and specify any additional required fields during a full synchronization.
- To dump full synchronization updates into a file using `fileop->read` operation when the updated objects are large in number, you can specify `_object=db_objects` and all the necessary parameters.
- Infoblox recommends a full synchronization in the following cases:
  - After an upgrade, restore or master promotion, which resets the sequence ID.
  - During off-peak hours for busy Grids with high rate of change that is greater than 500/sec.
  - When the maximum time or maximum number to track deleted objects is met, with the default being 4 hours and 4000 objects, then synchronization API returns an error if the last deleted sequence ID is newer than the current sequence ID.

Following are a few samples of API requests for full synchronization:

- If you specify `_output_location = "local"` and schedule the task, then you must use `fileop ->get_file_url` to retrieve the URL:

```
curl -k1 -u admin:infoblox -X POST 'https://127.0.0.1/wapi/v2.5/fileop?_function=get_file_url' -H "Content-
Type: application/json" -d '{"task_id":1}'
```

The response is as follows:

```
"url":
"https://10.32.2.202/http_direct_file_io/req_id-OBJECTS-1/nios-db-objects-1.json"
```

You can fetch this file using the following API request:

```
curl -k1 -u admin:infoblox -X GET https://10.32.2.202/http_direct_file_io/req_id-OBJECTS-1/nios-db-objects-1.
JSON -H 'Content-Type: application/json; charset=UTF-8'
```

- If you specify `_output_location = "local"` and do not schedule the task, then NIOS returns an URL and a token:

```
curl -k1 -u admin:infoblox -X POST 'https://127.0.0.1/wapi/v2.5/fileop?_function=read'
-H 'Content-Type: application/json; charset=UTF-8' -d '{ "_output_location":"LOCAL","_object": "db_objects",
"all_object_types_supported_in_version": "2.5","_encoding": "JSON"}'
```

The response is as follows:

```
{
    "token":

"eJy1kU1vwyAMhu/8kfaSD/JB0t5aZZU2Ta3UTtrRSoB2nlJggUztv5+ZtJ123QFk/L7mMUZK6+4w\n6QujTVr
jwzTLYCfmOFtKNGc7jPaWWqPjCnenPev60MNRn5krmAQYZhwDGgCmUAbmSrZUrmKnhb45\nnO4Q8KoXzNVsxyt
RtKVYFU0qqqpuBWf+tJinkWRBBW8hOL/OMp6nZZ2KtG2ymAKF1FuAM44a0GaT\n/gBUSXd43T8fNl3C85xnBq1
P1AB2eCevT5J8leR1UuRcQFGsy3pN1KfTYU+sJmJRUdQS9a/rSFpF\nk6KnUsxz8mWe5tJfdBau7n/64vyHCdp
Iq9BcYrYg+Pbx21D+Gq5WxanyOOhu87KB48Munmvmw9Fx\nET+BNyTi0DtA4+YAn3ryaE20tWzvh/QLT4Gbhw=
=\n",
    "url":
"https://10.35.6.87/http_direct_file_io/req_id-DOWNLOAD-1001/nios-db_objects--09-05-20

16_22:35:27.JSON"
}
```

- To save the file in a local area instead of the file distribution area:

```
curl -k1 -u admin:infoblox -X POST 'https://127.0.0.1/wapi/v2.5/fileop?_function=read&_schedinfo.
schedule_now=true' -H 'Content-Type: application/json; charset=UTF-8' -d '{"_output_location": "LOCAL",
"_object": "db_objects", "all_object_types_supported_in_version": "2.5", "_encoding": "JSON"}'
```

The response is as follows:

```
scheduledtask/b25lLnF1ZXVlZF90YXNrJDE:1/WAITING_EXECUTION
```

- You can also schedule a full synchronization task as follows:

```
curl -k1 -u admin:infoblox -X POST 'https://127.0.0.1/wapi/v2.5/fileop?_function=read&_schedinfo.
schedule_now=true' -H 'Content-Type: application/json; charset=UTF-8' -d '{"_filename": "test2.txt",
"_object": "db_objects", "all_object_types_supported_in_version": "2.5", "_encoding": "JSON"}' scheduledtask
/b25lLnF1ZXVlZF90YXNrJDE:1/WAITING_EXECUTION
```

NIOS returns the scheduled task reference so that you can query and find out when will the scheduled task be complete:

```
curl -k1 -u admin:infoblox -X GET

https://127.0.0.1/wapi/v2.5/scheduledtask/b25lLnF1ZXVlZF90YXNrJDE:1/WAITING_EXECUTION
```

The response is as follows:

```
{ "_ref": "scheduledtask/b25lLnF1ZXVlZF90YXNrJDE:1/COMPLETED", "approval_status": "NONE",
"execution_status": "COMPLETED", "task_id": 1 }
```

Note that the result is in either JSON or XML format. You can fetch the output file from the `wapi-output` directory of the file distribution area when the scheduled task is complete.

Using Incremental Synchronization

Incremental synchronization synchronizes only those objects that are updated since the previous synchronization. With incremental synchronization, you can query NIOS for any updates and when updates are found, NIOS returns the changed objects since the previous incremental synchronization. Incremental synchronization synchronizes the NIOS database for object changes such as addition, updates, or deletion, mostly via scripts. The incremental synchronization query consists of object types, a cookie of the form `<id of db:sequence_id>`, a sequence ID and an optional `exclude_deleted` object to indicate if the deleted objects must be excluded from the results. NIOS includes the deleted records in the results by default.

Note: Infoblox recommends that you use `fileop->read` for incremental synchronization when the updated objects are large in number.

Note the following about incremental synchronization:

- You cannot request an incremental synchronization to get updated (created, updated, or deleted) objects after a full synchronization.
- For performance reasons, NIOS returns the objects that changed and not the object data that changed. Infoblox recommends that you figure out what changed by comparing the object with the previous version in the client database.
- Incremental synchronization may take a longer time to complete depending on the number of changes. For example, if 50K objects are requested with all fields on an IB-4010 appliance, an incremental synchronization approximately takes 2 minutes. When 50K objects are requested with standard RESTful API fields on an IB-4010, the synchronization process completes in approximately 1 minute.
- The size of the memory increases when the number of objects increases. Infoblox recommends that you use `_max_results` to limit the number of results or use the asynchronous `fileop->read` operation.
- Infoblox recommends that for a busy Grid with a high Rate of Change that is greater than 500/sec, you must run an incremental synchronization more frequently. Example: If the expected Rate of Change for objects in the Grid is 50/sec, you can schedule an incremental synchronization every 5-10 minutes. If the Rate of Change for objects is 200/sec, schedule an incremental synchronization every minute or every 2 minutes.
- Infoblox recommends that you include all object types when you query using an incremental synchronization, so that it returns the highest sequence ID for all objects that can be used for subsequent incremental updates. If you query only selected object types such as A record and zone, NIOS returns the highest sequence ID for those object types only.

  Consider an example where A record, Network, and A record are the object types with sequence ID 4, 5 and 6 respectively. When you query for an A record object type with a previous sequence ID using incremental synchronization, NIOS returns A record with the highest sequence ID, which is 6 and does not return the network object. Meanwhile, if you add or update a network object, then in the subsequent incremental synchronization it does not return the network object and you will lose updates made to the network object. Hence, Infoblox recommends that you include all object types in the query to ensure that you do not lose updates made to other object types.

- You cannot perform a full or an incremental synchronization during Grid Master Candidate promotions, restores, and upgrades.
- The output of an incremental synchronization is either in JSON or XML format.

The following are a few samples of API requests for incremental synchronization:

- To get updates with all_objects_supported_in_version=2.5:

```
curl -k1 -u admin:infoblox -H content-type:application/json -X GET

https://10.35.6.87/wapi/v2.5/db_objects?start_sequence_id=992684967:0\&all_object_type

s_supported_in_version=2.5\;return_fields=last_sequence_id,object,object_type,unique_id\;_return_type=json
```

  The response for the above API request contains all fields of all object types.

- To get an A record and an auth zone object update:

```
curl -k1 -u admin:infoblox -H content-type:application/jason -X GET
https://10.32.2.202/wapi/v2.5/db_objects?start_sequence_id=183566281:0\&object_types=record:a,zone_auth\;
_return_fields=last_sequence_id,object,object_type,unique_id\;_return_type=json
```

  The response is as follows:

```
[{"_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDA:19", "object": {"_ref":
"zone_auth/ZG5zLnpvbmUkLl9kZWZhdWx0LmNvbS5mb28:foo.com/default", "fqdn": "foo.com",
"view": "default"}, "object_type": "zone_auth", "unique_id":
"087af628fa03418faa0577b953807efc"}, {"_ref":
"db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDE:21", "object": {"_ref":
"record:a/ZG5zLmJpbmRfYSQuX2RlZmF1bHQuY29tLmZvbyxhcmVjMSwxLjIuMy4y:arec1.foo.com/default", "ipv4addr":
"1.2.3.2", "name": "arec1.foo.com", "view": "default"}, "object_type":
"record:a", "unique_id": "89314c3fae2841f49600e1b686b0c7b7"},{"_ref":
"db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDI:183566281%3A21", "last_sequence_id":
"183566281:21"},
```

- NIOS returns deleted objects as synthetic deleted objects. To get updates on deleted host objects:

```
curl -s -k1 -u admin:infoblox -H content-type:application/json -X GET
https://10.34.19.220/wapi/v2.5/db_objects?start_sequence_id=1207501969:0\&object_types
=record:host,bulkhost,record:host_ipv4addr,record:host_ipv6addr\&exclude_deleted=false
```

  The response is as follows:

```
[{ "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDM2YQ:130",
        "object": {
            "_ref": "deleted_objects/Li5kZWxldGVkX29iamVjdHMkMA",
            "object_type": "record:host_ipv6addr"
        },
        "object_type": "deleted_objects",
        "unique_id": "f7d87ec4b9204e16a3d14bfb340c402d" },
{ "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDM2Yg:130",
        "object": {
           "_ref": "deleted_objects/Li5kZWxldGVkX29iamVjdHMkMQ",
           "object_type": "record:host_ipv4addr" },
        "object_type": "deleted_objects",
        "unique_id": "68316853600f41f083e2be134628ce62" },
{ "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDM2Yw:130",
        "object": {
           "_ref": "deleted_objects/Li5kZWxldGVkX29iamVjdHMkMg",
           "object_type": "record:host" },
        "object_type": "deleted_objects",
        "unique_id": "7a5e0622e9db441caa47c12ca30700d5" },
{ "_ref": "db_objects/Li5hbGxfY2hhbmdlZF9vYmplY3RzJDM2ZA:1207501969%3A130",
        "last_sequence_id": "1207501969:130" }]
```

- To dump incremental synchronization updates into a file using `fileop->read` operation when the updated objects are large in number, you can specify `_object=db_objects` and all the necessary parameters for incremental synchronization:

```
curl -k1 -u admin:infoblox -X POST 'https://10.34.19.100/wapi/v2.5/fileop?_function=read' -H 'Content-Type:
application/json; charset=UTF-8' -d '{"_output_location": "LOCAL", "_object": "db_objects",
"start_sequence_id" : "1973553522:0","all_object_types_supported_in_version": "2.5", "_max_results":
1000000,"_encoding": "JSON"}'
```

For more information about supported objects in the Infoblox API and Restful API, refer to the Infoblox API Documentation and the Infoblox WAPI Documentation.