Release H: Influxdb Security

Introduction

Influxdb uses API token for secure interaction with the database.

These tokens fall into 3 categories:

1. Operator Token - Grants full read and write access to all organizations and all organization resources in he database.
2. All Access Token - Grants full read and write access to all resources in an organization.
3. Read/Write Token - Grants full read and write access to specific buckets in an organization.

The token used to do the initial setup is an operator token. This type of token is required to setup users and assign them to organizations.

You can retrieve the authorizations using the following command:

influx auth

```
influx auth list -t UQ5ayNlN33QK-c6G6fOBU7lKVVynpEF_TijIGxNuhWkSs75TMjnv4yucHjOmDXhSMJmxZzYe9xjQI1MDlEAPuw==
ID                      Description
Token                                                                   User Name       User
ID                Permissions
0ae29cdcfe76b000        influxdb's Token        UQ5ayNlN33QK-
c6G6fOBU7lKVVynpEF_TijIGxNuhWkSs75TMjnv4yucHjOmDXhSMJmxZzYe9xjQI1MDlEAPuw==     influxdb
0ae29cdce7f6b000        [read:/authorizations write:/authorizations read:/buckets write:/buckets read:
/dashboards write:/dashboards read:/orgs write:/orgs read:/sources write:/sources read:/tasks write:/tasks read:
/telegrafs write:/telegrafs read:/users write:/users read:/variables write:/variables read:/scrapers write:
/scrapers read:/secrets write:/secrets read:/labels write:/labels read:/views write:/views read:/documents
write:/documents read:/notificationRules write:/notificationRules read:/notificationEndpoints write:
/notificationEndpoints read:/checks write:/checks read:/dbrp write:/dbrp read:/notebooks write:/notebooks read:
/annotations write:/annotations read:/remotes write:/remotes read:/replications write:/replications]
0ae2b000d3bdb000        Rest Token              Tt3ORWWBeR8niojjUTVQ4PndTQDdD0lEKMD-
sJjtH__kMqD1vWuFybLqZN972zkbB8rvGVYFZ4yDKkcz2pkOQA==     influxdb        0ae29cdce7f6b000        [read:orgs
/4a43da86fa150161/annotations write:orgs/4a43da86fa150161/annotations read:orgs/4a43da86fa150161/authorizations
write:orgs/4a43da86fa150161/authorizations read:orgs/4a43da86fa150161/buckets write:orgs/4a43da86fa150161
/buckets read:orgs/4a43da86fa150161/checks write:orgs/4a43da86fa150161/checks read:orgs/4a43da86fa150161
/dashboards write:orgs/4a43da86fa150161/dashboards read:orgs/4a43da86fa150161/dbrp write:orgs/4a43da86fa150161
/dbrp read:orgs/4a43da86fa150161/documents write:orgs/4a43da86fa150161/documents read:orgs/4a43da86fa150161
/labels write:orgs/4a43da86fa150161/labels read:orgs/4a43da86fa150161/notebooks write:orgs/4a43da86fa150161
/notebooks read:orgs/4a43da86fa150161/notificationEndpoints write:orgs/4a43da86fa150161/notificationEndpoints
read:orgs/4a43da86fa150161/notificationRules write:orgs/4a43da86fa150161/notificationRules read:/orgs
/4a43da86fa150161 read:orgs/4a43da86fa150161/remotes write:orgs/4a43da86fa150161/remotes read:orgs
/4a43da86fa150161/replications write:orgs/4a43da86fa150161/replications read:orgs/4a43da86fa150161/scrapers
write:orgs/4a43da86fa150161/scrapers read:orgs/4a43da86fa150161/secrets write:orgs/4a43da86fa150161/secrets
read:orgs/4a43da86fa150161/sources write:orgs/4a43da86fa150161/sources read:orgs/4a43da86fa150161/tasks write:
orgs/4a43da86fa150161/tasks read:orgs/4a43da86fa150161/telegrafs write:orgs/4a43da86fa150161/telegrafs read:
/users/0ae29cdce7f6b000 write:/users/0ae29cdce7f6b000 read:orgs/4a43da86fa150161/variables write:orgs
/4a43da86fa150161/variables read:orgs/4a43da86fa150161/views write:orgs/4a43da86fa150161/views]
```

They can also be obtained from the rest endpoint: http://<influxdb host>:<influxdb port>/api/v2/authorizations

The first token listed above (influxdb's Token) is he setup token and grants access to all resources in the database.

Queries

The following example shows how to create a new bucket using one of these tokens:

Create bucket

```
#!/bin/sh
INFLUX_ORG_ID="4a43da86fa150161"
INFLUX_TOKEN="UQ5ayNlN33QK-c6G6fOBU7lKVVynpEF_TijIGxNuhWkSs75TMjnv4yucHjOmDXhSMJmxZzYe9xjQI1MDlEAPuw=="

curl --request POST "http://localhost:8086/api/v2/buckets"
--header "Authorization: Token ${INFLUX_TOKEN}"
--header "Content-type: application/json"
--data '{
    "orgID": "'"${INFLUX_ORG_ID}"'",
    "name": "iot-bucket",
    "retentionRules": [
      {
        "type": "expire",
        "everySeconds": 86400,
        "shardGroupDurationSeconds": 0
      }
    ]
  }'
```

We can also use the version 1 APIs with a username and password if required.

First create a v1 user and assign read an write permissions for a particular bucket:

influx v1 auth create -o EST --username influxv1 --password influxv1 --read-bucket 272626d8c766fd27 --write-bucket 272626d8c766fd27 -t UQ5ayNlN33QK-c6G6fOBU7lKVVynpEF_TijIGxNuhWkSs75TMjnv4yucHjOmDXhSMJmxZzYe9xjQI1MDlEAPuw==

We can then write to the bucket like this:

Write

```
curl -iv -XPOST "http://localhost:8086/write?db=ts-bucket&precision=ns" \
  --user "influxv1":"influxv1"
  --header "Content-Type: text/plain; charset=utf-8" \
  --header "Accept: application/json" \
  --data-binary '
    airSensors,sensor_id=TLM0201 temperature=73.97038159354763,humidity=35.23103248356096,co=0.48445310567793615
    airSensors,sensor_id=TLM0202 temperature=75.30007505999716,humidity=35.651929918691714,co=0.5141876544505826
    '
```

and read the data back like this:

Query

```
curl -v --get "http://localhost:8086/query" \
  --user "influxv1":"influxv1" \
  --data-urlencode "db=ts-bucket" \
  --data-urlencode "q=SELECT * FROM airSensors WHERE \"time\" > now()-1h"
```

Note: If we are using a measurement with special characters like SubNetwork=CountryNN,MeContext=MEC-Gbg-1,ManagedElement=RNC-Gbg-1ManagedElement=RNC-Gbg-1,ENodeBFunction=1 we need to surrounds the measurement name in quotes : "q=SELECT * FROM \" SubNetwork=CountryNN,MeContext=MEC-Gbg-1,ManagedElement=RNC-Gbg-1ManagedElement=RNC-Gbg-1,ENodeBFunction=1\""

JWT Authorization in Influxdb V1

If we include the following environment variables in our influxdb (v1) docker container we can enable authorization and use JWTs to retrieve data:

INFLUXDB_HTTP_SHARED_SECRET: "my super secret pass phrase"
INFLUXDB_ADMIN_USER: influxadmin
INFLUXDB_ADMIN_PASSWORD: influxadmin
INFLUXDB_HTTP_AUTH_ENABLED: "true"

The following python program shows this in action:

Influxdb JWT

```
import requests
import jwt
from datetime import datetime, timedelta, timezone

def get_jwt(username, secret, algorithm):
   payload_data = {
       "username": username,
       "exp": datetime.now(tz=timezone.utc) + timedelta(minutes=15)
   }

   encoded = jwt.encode(
       payload=payload_data,
       key=secret,
       algorithm=algorithm
   )
   return encoded

url = "http://localhost:8085/query"
username = "influxadmin"
secret = 'my super secret pass phrase'
algorithm="HS256"
jwt = get_jwt(username, secret, algorithm)

headers = { "Authorization": "Bearer "+jwt.decode('utf-8') }

querystring = {"pretty": "true", "db": "ts_pms_metrics",
               "q": "SELECT \"eventName\", \"domain\", \"sourceName\", \"measuredEntityUserName\", \"
startEpochMicrosec\", \"startEpochDate\", \"lastEpochMicrosec\", \"lastEpochDate\", \"measuredEntityDn\", \"
measObjInstId\", \"sMeasType\" ,\"sValue\", \"suspectFlag\" FROM \"pms_data\" WHERE \"time\" > now()-20s"}

response = requests.request("GET", url=url, headers=headers, params=querystring)

print(response.text)
```

To create a new user in influxdb v1 use the following commands:

Influxdb v1 user

```
/ # influx  -username influxadmin -password influxadmin -execute 'SHOW DATABASES'
name: databases
name
----
_internal
ts_db
ts_db1
ts_test
ts_host_metrics
ts_pms_metrics
ts_pms_metrics2
null
ts_pms_bucket
ts_pms_bucket2
ts_pms_bucket3
ts_pms_bucket4
ts_pms_metrics3
telegraf
ts_pms_metrics_v1
/ # influx  -username influxadmin -password influxadmin -database ts_pms_metrics
Connected to http://localhost:8086 version 1.7.11
InfluxDB shell version: 1.7.11
> CREATE USER influxweb WITH PASSWORD 'influxweb' WITH ALL PRIVILEGES
```

The first command shows the available databases

The second one logs into the ts_pms_metrics database using the admin user.

The last command creates a new user "influxweb"

Note: If you restart influxdb you'll need to remove these variables otherwise it will get stuck in a loop trying to create the admin user again

INFLUXDB_ADMIN_USER: influxadmin
INFLUXDB_ADMIN_PASSWORD: influxadmin

Note: JWT authorization is no longer supported in Influxdb v. 2

Client Libraries

There are a number of client libraries available for interacting with influxdb, the full list can be seen here: InfluxDB client libraries

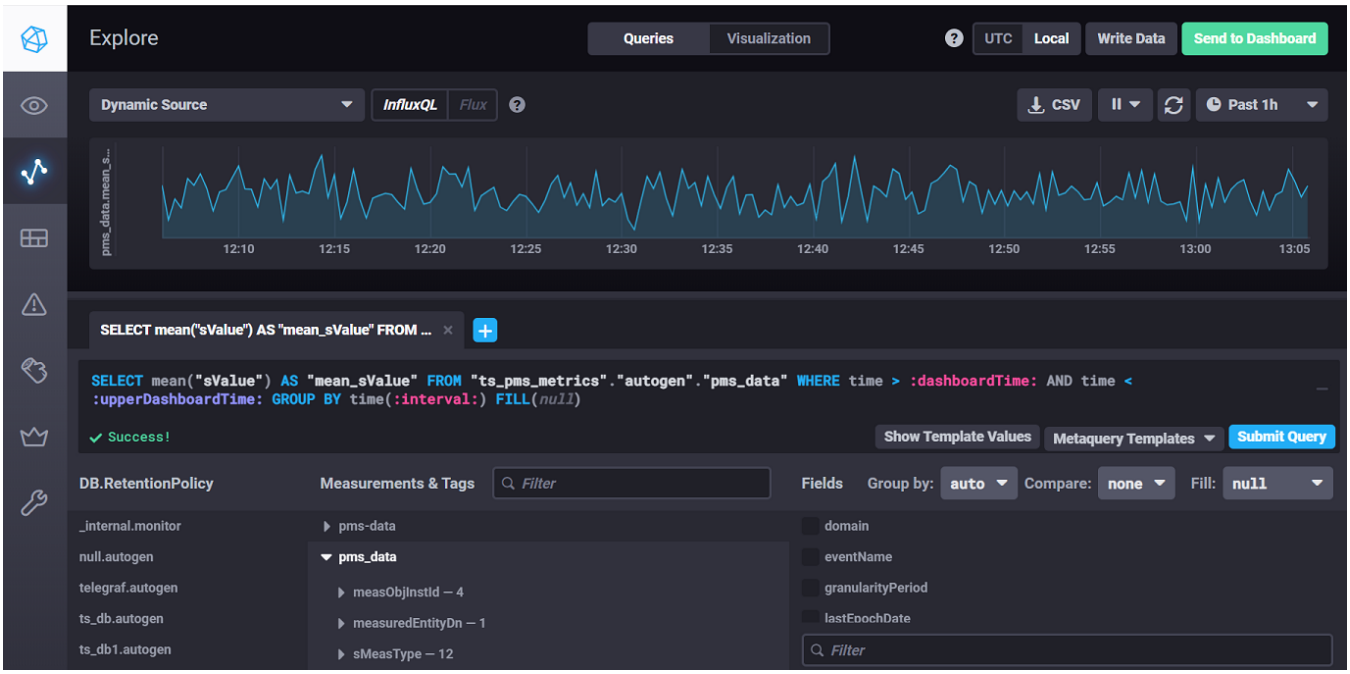
Here is an example of the python client library:

influxdb_client

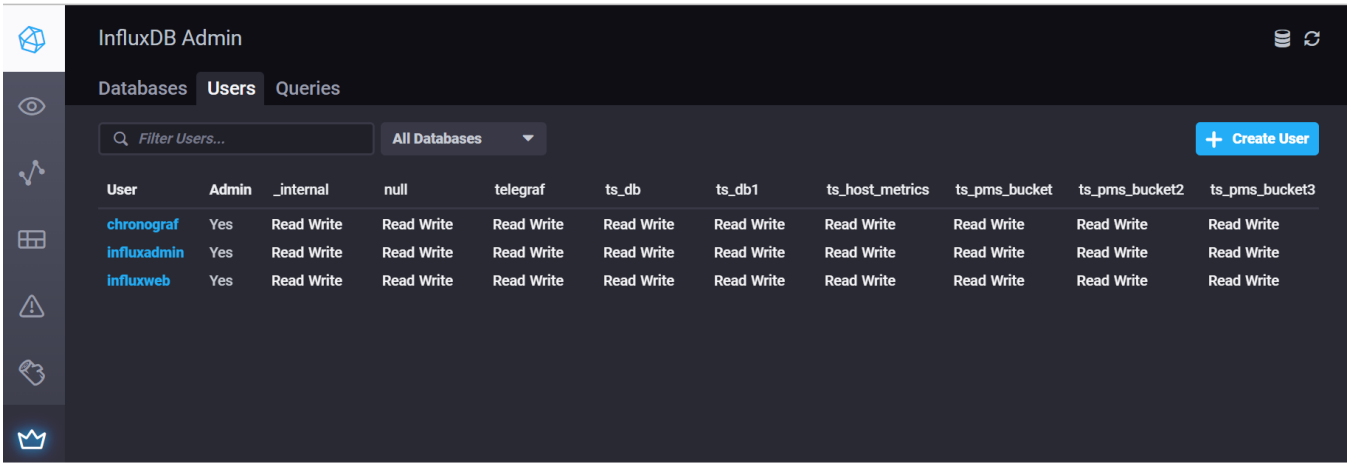
```
from influxdb_client import InfluxDBClient, Point, PermissionResource, Permission
from influxdb_client.domain import Authorization

my_org = "iot"
my_url = "http://localhost:8086"
my_username = "influxdb"
my_password = "influxdb"
my_bucket_name = "iot-bucket"
client = InfluxDBClient(url=my_url, username=my_username, password=my_password, org=my_org)
my_org_id = ""

organizations_api = client.organizations_api()
orgs = organizations_api.find_organizations()

# Check if org already exists
my_org_list = [o for o in orgs if o.name == my_org]
if len(my_org_list):
   org = my_org_list[0]
   my_org_id=org.id
   print("Found " + org.name + ", " + my_org_id)
else:
   print("Creating " + my_org)
   org = organizations_api.create_organization(name=my_org)
   my_org_id=org.id

buckets_api = client.buckets_api()
# Check if bucket already exists
bucket = buckets_api.find_bucket_by_name(bucket_name=my_bucket_name);
if not bucket is None:
   print("Found " + bucket.name)
else:
   print("Creating " + my_bucket_name)
   bucket = buckets_api.create_bucket(bucket_name=my_bucket_name)

# Create a new Authorization token for the bucket
bucket_resource = PermissionResource(org_id=my_org_id, id=bucket.id, type="buckets")
read_bucket = Permission(resource=bucket_resource, action="read")
write_bucket = Permission(resource=bucket_resource, action="write")
auth = Authorization()
auth.org_id=my_org_id
auth.permissions=[read_bucket, write_bucket]
auth.description=bucket.name+' Token'

authorizations_api = client.authorizations_api()
authorizations_api.create_authorization(authorization=auth)

# Find available authorizations
authorizations = authorizations_api.find_authorizations()
for auth in authorizations:
    print(auth.description + " - " + auth.token + " - " + auth.status + " - " + auth.org_id)
```

Chronograf

Chronograf can be used to visualize your data in either V1 or V2, although it's very similar to what comes out of the box with V2.



You can also use it to create users:



You can use the following yaml to run it in your cluster: chronograf.yaml

Links

Manage security and authorization

Manage API tokens

API Quick Start

InfluxDB OSS API Service

InfluxDB Tech Tips; Creating Tokens with the InfluxDB API

Write data with the InfluxDB API V1

Authentication and authorization in InfluxDB V1

Using influx - InfluxDB V1 command line interface

Community Templates

Install Chronograf

Chronograf configuration options