Install and configure Butler-SOS on Windows

The steps below outline how to go about running Butler-SOS on a Windows environment.

1. Download and extract influxdb 1.7

Download InfluxDb

You can follow the link above to view the latest available binaries or simply copy the link below.

```
https://dl.influxdata.com/influxdb/releases/influxdb-1.7.9_windows_amd64.zip
```

> unzip influxdb-1.7.9*windows*amd64.zip

2. Download and install Grafana

The simplest option is to download the .msi package which will run through a wizard to complete installation. You also have the option to manually unblock and unzip the binary.

Download Grafana

Some additional configuration options can be found below. You should be ok with defaults.

```
https://grafana.com/docs/installation/windows/
https://grafana.com/docs/installation/configuration/
```

As part of the Grafana .msi package, nssm is used to register Grafana server as a Windows service, so you can easily stop/start from services.msc

Make sure the Grafana service is running and then navigate to your local grafana server

> by default, the server will run on port 3000 with the credentials of admin/admin
>
> you can change these credentials the first time grafana is opened or at another time

3. Download and configure Butler-SOS

Clone or download the GitHub repository here

Follow instructions on exporting certificates (if you're running butler on your Sense server, you should be able to use the .pem files located at "C:\ProgramData\Qlik\Sense\Repository\Exported Certificates.Local Certificates").

> Instructions: **Go to instructions**

Next you'll need to create the config file called "production.yaml". You'll find a template "production_template.yaml" in the .\src\config directory (wherever you extracted/cloned the git repository). Below you can find my example config file which would only need to be modified slightly.

1. Update the hostname from "RUHANWIN" to your own host. In my case, I'm running Sense and butler on the same machine.
2. Update the logdb port (if you didn't use the Qlik default of 4432)
3. Update the qlogsReaderPwd to your own password set during QLogs setup
4. Update the path to your certificates (either exported from QMC or in .Local Certificates)
5. You can skip MQTT config and leave as is
6. Update the IP address for InfluxDb
7. Update your Sense server hostname(s) and virtual proxies

```
Butler-SOS:
  # All configuration items are mandatory, unless otherwise noted.

  # Logging configuration
  logLevel: info          # Log level. Possible log levels are silly, debug, verbose
  fileLogging: true        # true/false to enable/disable logging to disk file
  logDirectory: logs      # Subdirectory where log files are stored

  # Qlik Sense logging db config parameters
  logdb:
    enableLogDb: true
    # Items below are mandatory if enableLogDb=true
    influxDbRetentionPolicy: DEFAULT  # Name of Influxdb policy used to determine ho
                                      #   log db data should be kept in Influxdb
    pollingInterval: 60000    # How often (milliseconds) should Postgres log db be c
    queryPeriod: 5 minutes    # How far back should Butler SOS query for log entries
    host: RUHANWIN
    port: 4432
    qlogsReaderUser: qlogs_reader
    qlogsReaderPwd: supersecretpassword
    extractErrors: true        # Should error level entries be extracted from log d
    extractWarnings: true      # Should warn level entries be extracted from log db
    extractInfo: true          # Should info level entries be extracted from log db
                              # Warning! Seting this to true will result in LOTS of
                              # being retrrieved by Butler SOS!

  # Certificates to use when querying Sense for healthcheck data. Get these from the
```

```yaml
  cert:
    clientCert: C:\butler-sos\ruhanwin\client.pem
    clientCertKey: C:\butler-sos\ruhanwin\client_key.pem
    clientCertCA: C:\butler-sos\ruhanwin\root.pem
    # If running Butler SOS in a Docker container, the cert paths MUST be the follow
    # clientCert: /nodeapp/config/certificate/client.pem
    # clientCertKey: /nodeapp/config/certificate/client_key.pem
    # clientCertCA: /nodeapp/config/certificate/root.pem

  # MQTT config parameters
  mqttConfig:
    enableMQTT: false
    # Items below are mandatory if enableMQTT=true
    brokerHost: RUHANWIN
    brokerPort: 1883
    baseTopic: butler-sos/          # Topic should end with /

  # Influx db config parameters
  influxdbConfig:
    enableInfluxdb: true
    # Items below are mandatory if enableInfluxdb=true
    hostIP: 10.211.55.3
    hostPort: 8086                  # Optional. Default value=8086
    auth:
      enable: false                    # Does influxdb instance require authenticati
      username: <username>          # Username for Influxdb authentication. Mandator
      password: <password>          # Password for Influxdb authentication. Mandator
    dbName: SenseOps

    # Default retention policy that should be created in InfluxDB when Butler SOS cr
    # Any data older than retention policy threshold will be purged from InfluxDB.
    retentionPolicy:
      name: 2_weeks
      duration: 2w

    # Control whether certain fields are stored in InfluxDB or not
    # Use with caution! Enabling activeDocs, loadedDocs or inMemoryDocs may result i
    includeFields:
      activeDocs: false               # Should data on what docs are active be sto
      loadedDocs: false               # Should data on what docs are loaded be sto
      inMemoryDocs: false             # Should data on what docs are in memory be s

  # Sessions per virtual proxy
  userSessions:
    enableSessionExtract: true      # Query unique user IDs of what users have sess
    # Items below are mandatory if enableSessionExtract=true
    pollingInterval: 30000          # How often (milliseconds) should detailed sessi
```

```
serversToMonitor:
  pollingInterval: 15000          # How often (milliseconds) should the healthchec

  # List of extra tags for each server. Useful for creating more advanced Grafana
  # Each server below MUST include these tags in its serverTags property.
  # The tags below are just examples - define your own as needed
  serverTagsDefinition:
    - server_group
    - serverLocation
    - server-type
    - serverBrand

  # Sense Servers that should be queried for healthcheck data
  servers:
    - host: RUHANWIN:4747
      serverName: RUHANWIN
      serverDescription: RUHANWIN
      logDbHost: RUHANWIN
      userSessions:
        enable: true
        # Items below are mandatory if userSessions.enable=true
        host: RUHANWIN:4243
        virtualProxies:
          - virtualProxy: /                # Default virtual proxy
          - virtualProxy: /anon              # "anon" virtual proxy
      serverTags:
        server_group: DEV
        serverLocation: Canada
        server-type: virtual
        serverBrand: WindowsParallels
```

Follow further instructions on Windows based setup here.

You should have already extracted Butler SOS so you can skip that step and simply install node.js and run `npm i` from your .\src directory to install the node modules. Ignore Python errors.

Open Windows command line as administrator and run: `set NODE_ENV=production`

TIP

> Run `SETX NODE_ENV production` to enable persistance of the variable even after closing the cmd window or shutting down the machine
>
> You can configure butler-sos as a Windows service using nssm (same as grafana setup) however for now, we are going to run it all manually in sequence for testing

4. The good part - let's test!

First, navigate to your influxDb directory from a new Windows command prompt window. My example below:

```
c:\butler-sos\influxdb-1.7.9_windows_amd64\influxdb-1.7.9-1>
```

Next, start the influxd.exe service

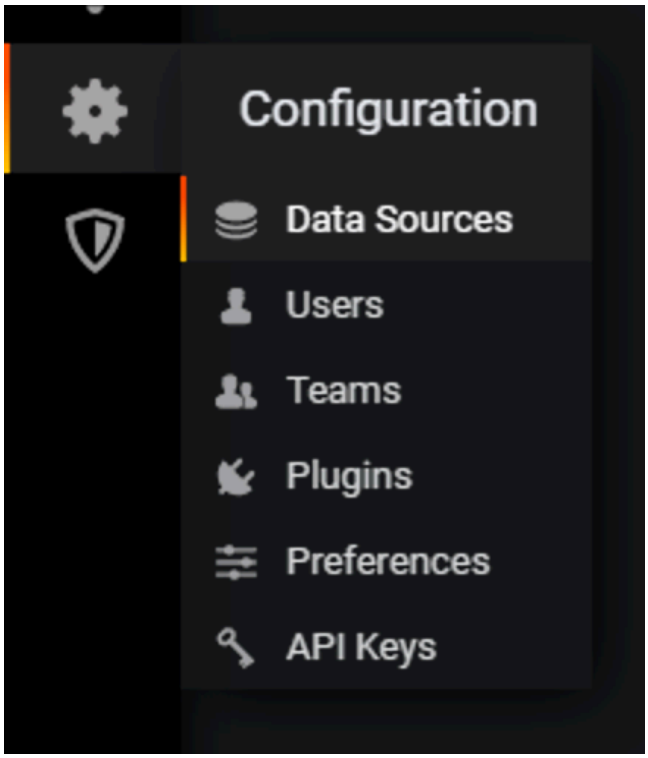
```
c:\butler-sos\influxdb-1.7.9_windows_amd64\influxdb-1.7.9-1>influxd.exe
```

You should see output similar to the below:

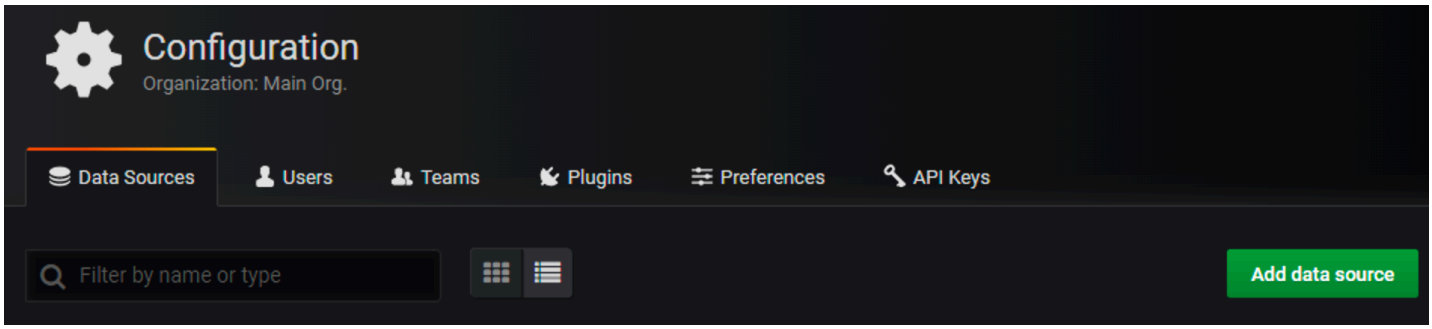
```
 8888888           .d888 888                    8888888b.  888888b.
   888            d88P"  888                    888  "Y88b 888  "88b
   888            888    888                    888    888 888  .88P
   888   88888b.  888888 888 888  888 888  888 888    888 8888888K.
   888   888 "88b 888    888 888  888  Y8bd8P' 888    888 888  "Y88b
   888   888  888 888    888 888  888   X88K   888    888 888    888
   888   888  888 888    888 Y88b 888 .d8""8b. 888  .d88P 888   d88P
 8888888 888  888 888    888  "Y88888 888  888 8888888P"  8888888P"

2019-11-25T23:37:33.470827Z     info    InfluxDB starting       {"log_id": "0JLEgG7C
2019-11-25T23:37:33.471829Z     info    Go runtime      {"log_id": "0JLEgG7G000", "v
2019-11-25T23:37:33.582830Z     info    Using data dir  {"log_id": "0JLEgG7G000", "s
2019-11-25T23:37:33.582830Z     info    Compaction settings     {"log_id": "0JLEgG7C
2019-11-25T23:37:33.583837Z     info    Open store (start)      {"log_id": "0JLEgG7C
2019-11-25T23:37:33.631831Z     info    Reading file    {"log_id": "0JLEgG7G000", "e
2019-11-25T23:37:33.694885Z     info    Opened shard    {"log_id": "0JLEgG7G000", "s
2019-11-25T23:37:33.695832Z     info    Reading file    {"log_id": "0JLEgG7G000", "e
2019-11-25T23:37:33.795832Z     info    Opened shard    {"log_id": "0JLEgG7G000", "s
2019-11-25T23:37:33.796830Z     info    Open store (end)        {"log_id": "0JLEgG7C
2019-11-25T23:37:33.796830Z     info    Opened service  {"log_id": "0JLEgG7G000", "s
2019-11-25T23:37:33.796830Z     info    Starting monitor service        {"log_id": "
2019-11-25T23:37:33.796830Z     info    Registered diagnostics client   {"log_id": "
2019-11-25T23:37:33.797832Z     info    Registered diagnostics client   {"log_id": "
2019-11-25T23:37:33.797832Z     info    Registered diagnostics client   {"log_id": "
2019-11-25T23:37:33.797832Z     info    Registered diagnostics client   {"log_id": "
2019-11-25T23:37:33.797832Z     info    Starting precreation service    {"log_id": "
2019-11-25T23:37:33.797832Z     info    Starting snapshot service       {"log_id": "
2019-11-25T23:37:33.797832Z     info    Starting continuous query service       {"lo
2019-11-25T23:37:33.797832Z     info    Starting HTTP service   {"log_id": "0JLEgG7C
2019-11-25T23:37:33.797832Z     info    opened HTTP access log  {"log_id": "0JLEgG7C
2019-11-25T23:37:33.798831Z     info    Listening on HTTP       {"log_id": "0JLEgG7C
2019-11-25T23:37:33.798831Z     info    Starting retention policy enforcement servic
2019-11-25T23:37:33.798831Z     info    Listening for signals   {"log_id": "0JLEgG7C
2019-11-25T23:37:33.797832Z     info    Storing statistics      {"log_id": "0JLEgG7C
2019-11-25T23:37:33.799832Z     info    Sending usage statistics to usage.influxdata
```

Next, open Windows Services (Go to Start, type "services"), scroll to "Grafana" and start the service. You should be able to reach http://localhost:3000 on your machine after starting the service successfully.

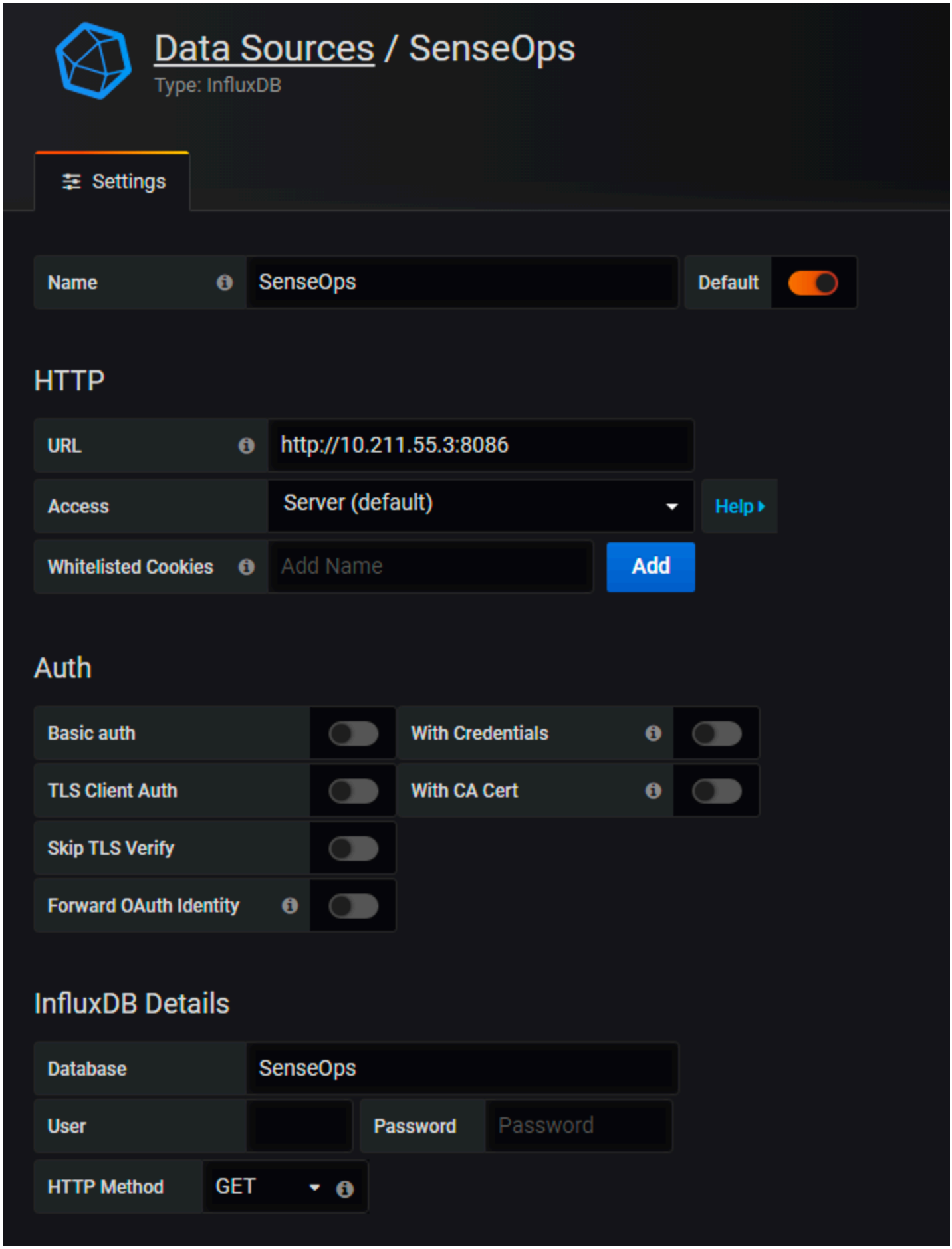
From the Grafana client, create the data source for SenseOps by following the screenshots below.



> Data Sources



> Add data source



SenseOps example

> Save and Test

Finally, open a separate command line window and navigate to your butler-sos\src directory. My example below:

```
c:\butler-sos\src>
```

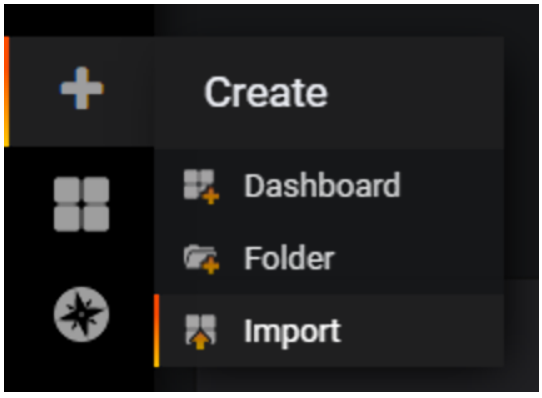
From here, run butler-sos.js:

```
c:\butler-sos\src>node butler-sos.js
```

You should see output similar to this:

```
2019-11-25T23:46:01.582Z info: CONFIG: Influxdb enabled: true
2019-11-25T23:46:01.586Z info: CONFIG: Influxdb host IP: 10.211.55.3
2019-11-25T23:46:01.586Z info: CONFIG: Influxdb host port: 8086
2019-11-25T23:46:01.587Z info: CONFIG: Influxdb db name: SenseOps
2019-11-25T23:46:01.864Z info: ------------------------------------
2019-11-25T23:46:01.866Z info: Starting Butler SOS
2019-11-25T23:46:01.866Z info: Log level is: info
2019-11-25T23:46:01.867Z info: App version is: 5.0.1
2019-11-25T23:46:01.867Z info: ------------------------------------
2019-11-25T23:46:01.874Z info: MAIN: Docker healthcheck server now listening
2019-11-25T23:46:01.881Z info: CONFIG: Found InfluxDB database: SenseOps
```

You are now ready to import the example grafana dashboards located in the original github repository.



> Import
>
> Navigate to the grafana folder in the git repository and import the .json configurations for the overview dashboard and detail dashboard