InfluxDB Administration

Michael Desa & Jack Zampolin

By the end of this section, participants will be able to…

1. Describe the various parts of the InfluxDB Configuration file
2. Backup and Restore your InfluxDB instance
3. Manage Migrating to a new version of InfluxDB

Configuration

[meta]

```
# Where the metadata/raft database is stored
dir = "/var/lib/influxdb/meta"

retention-autocreate = true

# If log messages are printed for the meta service
logging-enabled = true
pprof-enabled = false

# The default duration for leases.
lease-duration = "1m0s"
```

Controls parameters for InfluxDB’s metastore, which stores information on users, databases, retention policies, shards, and continuous queries.

[data]

```
enabled = true

dir = "/var/lib/influxdb/data"

…

wal-dir = "/var/lib/influxdb/wal"
wal-logging-enabled = true
data-logging-enabled = true
…
```

Controls where the actual shard data for InfluxDB lives and how it is flushed from the WAL.

- "dir" may need to be changed to a suitable place for your system.
- The defaults should work for most systems.

[retention]

```
  enabled = true
  check-interval = "30m"
```

The retention service controls the enforcement of retention policies for evicting old data.

[shard-precreation]

```
enabled = true
check-interval = "10m"
advance-period = "30m"
```

The shard-precreation service controls the precreation of shards, so they are available before data arrives. Only shards that, after creation, will have both a start- and end-time in the future, will ever be created. Shards are never precreated that would be wholly or partially in the past.

[monitor]

```
store-enabled = true # Whether to record statistics internally.
store-database = "_internal"
store-interval = "10s" # The interval at which to record statistics
```

The monitor service writes diagnostic data to the database. This includes information about writes, queries, the http listener, the individual shards, etc…

[admin]

```
enabled = true
bind-address = ":8083"
https-enabled = false
https-certificate = "/etc/ssl/influxdb.pem"
```

Controls the availability of the built-in, web-based admin interface. If HTTPS is enabled for the admin interface, HTTPS must also be enabled on the [http] service.

[http]

```
enabled = true
bind-address = ":8086"
auth-enabled = false
log-enabled = true
write-tracing = false
pprof-enabled = false
https-enabled = false
https-certificate = "/etc/ssl/influxdb.pem"
```

Controls how the HTTP endpoints are configured. These are the primary mechanism for getting data into and out of InfluxDB.

Plugin configuration

InfluxDB supports the following write protocols at custom endpoints:

```
[[graphite]]
[opentsdb]
[collectd]
```

InfluxDB also supports writing line protocol via UDP:

```
[[udp]]
```

Backup and Restore

Backing up a Database

```
influxd backup [-database <name_of_database>] <backup_dir>
```

- Using the backup command without database flag will only backup the meta service
- Backups can be incremental in time
- Both the whole system as well as discrete sections of the database can be backed up

Backing up a Database (since)

```
  influxd backup [-database <name_of_database>] -since [time]
<backup_dir>
```

```
influxd backup -database NOAA_water_database -since 2016-02-01T00:00:00Z
```

- Using the backup command without database flag will only backup the meta service
- Backups can be incremental in time
- Both the whole system as well as discrete sections of the database can be backed up

Restoring from a backup

```
influxd restore -metadir <meta dir> -datadir <data dir> -database [db] <backup dir>
```

- Restoring from a backup is an offline process

Backup and Restore

Backup a database

```
$ influxd backup -database NOAA_water_database backup_dir
```

Stop the process

```
$ sudo service influxdb stop
```

Wipe the underlying directory

```
$ rm -rf ~/.influxdb              (OSX)
$ sudo rm -rf /var/lib/influxdb/* (Linux)
```

Restore the database

```
$ influxd restore -metadir ~/.influxdb/meta/ -datadir ~/.influxdb/data/ -database NOAA_water_database backup_dir/
```

Make sure that the DB was restored

Restart the process

```
$ influxd (OSX)
$ sudo service influxdb start (Linux)
```

Open up the CLI

```
$ influx
```

Query the Data

```
> use NOAA_water_database
> show measurements
```

Exercise (5-10 min)

Backup and restore the `telegraf` database.

Migrating Old Data

Why Upgrade to 0.12.X?

- Improved compression
- Improved write/query performance
- Improved query management

influxdata

Migration Example 0.9 to 0.12 (Downtime Unacceptable)

1. Install 0.11 on top of 0.9
2. Copy "cold" B1/BZ1 shards to /tmp
3. Run ./influx_tsm to convert "cold" shards
4. Run influx_tsm to convert shards, e.g.:
   1. `influx_tsm -backup /tmp/influxdb_backup /tmp/data`
5. Remove "cold" B1/BZ1 shards and replace with converted TSM shards
6. Wait until current "hot" shards become "cold", repeat steps 2-4 on new "cold" shards
7. Backup the metastore information
   1. `influx backup /tmp/influxdb_backup`
8. Shut down InfluxDB (this downtime is unavoidable, as InfluxDB 0.12 uses non-backwards compatible metastore format)
9. Install 0.12 on top of 0.11
10. Restore the metastore information
    1. `influxd restore -metadir=/var/lib/influxdb/meta /tmp/influxdb_backup`
11. Start InfluxDB 0.12

Migration Example 0.8 to 0.12 (Downtime Necessary)

1. Stop traffic to InfluxDB 0.8
2. Install InfluxDB 0.8.9 and restart
3. Run the Export tool
   1. `curl -o export.gz --compressed http://localhost:8086/ \export/<db>/<shard_space>`
4. Launch InfluxDB 0.12 on a new server
5. Use the CLI to import the exported data
   1. `influx -import -path=export.gz -compressed > failures`
6. Update clients and dashboards for new API, etc
7. Resume traffic to InfluxDB 0.12

influxdata