JBox Documentation

Release 0.0.1

Johnny (Chianing) Wang

Oct 25, 2017

Table of Contents:

JBox is an **Archival Software** with **in-line deduplication** and **compression** features, intended to backup data into Object Storage (**Swift**) over internet. It can be triggered by File System Watcher or by crawler which allows to **sync between multiple clients** on the fly.

CHAPTER 1

Introduction

What's JBox can do ?

- **In-Line Deduplication**
- **Compression**
- **Archive over the internet to ObjectStorage, Swift**
- **File Sync with multi-clients** like Cloud Storage Service e.g. DropBox
- **Delta Sync**
- **Versioning (Snapshot)**
- **Timing Purging** - Chunks Garbage Collection
- Pure Java, No Extra Installation Required
- Fully Leverage OpenStack **Swift**.
- No File System Watcher Library Required

Unique

- **In-Line Deduplication** Achive over Object Storge and Sync with Multi-Clients

CHAPTER 2

Execution

How to run JBox ?

1.Get JBox binary and Configuration File

- JBox only works for Linux (Ubuntu or CentOS).
- Download JBox binary directly from JBox github repo - JBox.
- Download JBox Configuration directlry from JBox github repo - JBoxconfig.properties.
- **Make sure JBox and JBoxconfig.properties at the same directory**

2. Copy C++ so

Find the code location and copy c++ `*.so ( shared object ) under /usr/lib/`

```
$ sudo cp ./dll/libclsJavaVariableChunk.so ./usr/lib/*
or
$ cp ./dll/libclsJavaVariableChunk.so /tmp/
```

if you have a question about reference the `*.so` in java you can reference this post.

- how to reference c lib in java via jni.

3. Prepare JBox Configuration

Prepare JBox Configuration `JBoxconfig.properties` with the JBox executable in the same Directory

```
# syncfolders=/hom/user/syncfolder
# it can be any folder and files underneath you would like to sync
syncfolders=/tmp/JBox
```

```
# JBox Properties
# swift auth url
authurl=https://www.xxx.com/auth/v1.0
# swift username
username=xxx
# swift password
password=xxx

# swift container, div, ext, pow, others
# if div then container name rule will be
# file-extension_type_power_div, e.g: pdfvar24128
# else if ext then container name rule will be
# file-extension_type_power, e.g: pdfvar24
# else if pow the cotnainer name rule will be type_power, e.g: var24
# else if others then container name rule will be
# others - put all the chunks into one container e.g: dedupcontainer
# else will be default pow
containername=GenTestNew

# sync time is milliseconds = 1/1000 seconds,
# 5000 milliseconds = 5 seconds
# if p: push mode, then means every sync time
# e.g. 30 min 30*60*1000=1800000 will re-sync
synctime=5000

# s: sync, q: query, r: retrive
# dedup algorithm,
# no - no deduplication, fix - fix chunking, var -variable chunking
type=var

# divider can be 32, 64, 128...2^n,
# if fix and var algorithm then use divider=0 or 1
divider=128

# power default is 0,
# if you prefer specific anchor then you can assigned it
# 10 = 2^10 as anchor
# if type is fix then fix size 2^10
# if type is var
# then var size is between 0.85 * 2^10 ~ 2 * 2^10
power=0

# refactor=0 is
# no refactor,
# 1 is refresh all the time,
# 2 is every 2^x/2^y = 2 then refactor mod
refactor=0

# extra parameters
# maximum multiplier
min=0.25
# minimum multiplier
max=32.0

# refcounter,
# -1: true deletion, 0 : off, 1 : on,
# if > 1 such as 2, 3, 4 ... ~
# means you have more than one client need to deal with.
# if it's -1 means delete right away,
# but this is only for push scenario and no multi clients
# if it's 0 means won't add auto purge feature
# when deleting the object and will keep chunks c+hash forever
# if it's 1 then move all deleted object to backup
# and give X-Delete-At <object purge seconds>
# if it's 2~n, then same with 1 but apply
# how many clients you have
refcounter=-1

# customized min and max instead of calculate by
# mod = size / 64, min=0.85*mod and max=2*mod
clientnum=1

# runmode: 0: master mode,
# only upload to object storage, 1: slaves mode which can sync
runmode=0
```

4. run JBox with arguments

```
// q: query
// r: retrive, download
// w: watch folder event then trigger sync
// s: use timer ( crawler ) then trigger sync
// p: push sync and only happen on time

$ JBox <q, r, w, s, p> or <help>
```

Command Line Help

- More detail you can try `$ JBox h`

PS: Setup Swift

- For run JBox, you need to have an OpenStack Environment, Swift All In One aka (SAIO) is an option if you didn't want to purchase any public cloud solution. The SAIO setup can be found in SAIO. or my post before OpenStack - Swift Dev Box - SAIO on Ubuntu 14.04 via VirtualBox.

PS: Install Java

- how to install Linux 32 bit Java.
- how to install Linux 64 bit Java.

CHAPTER 3

Development

How to join JBox coding ?

JBox is the Java code which is composed with `Eclipse IDE`. It's Eclipse project and easy to debug and test. Here are the steps how to open it in eclipse.

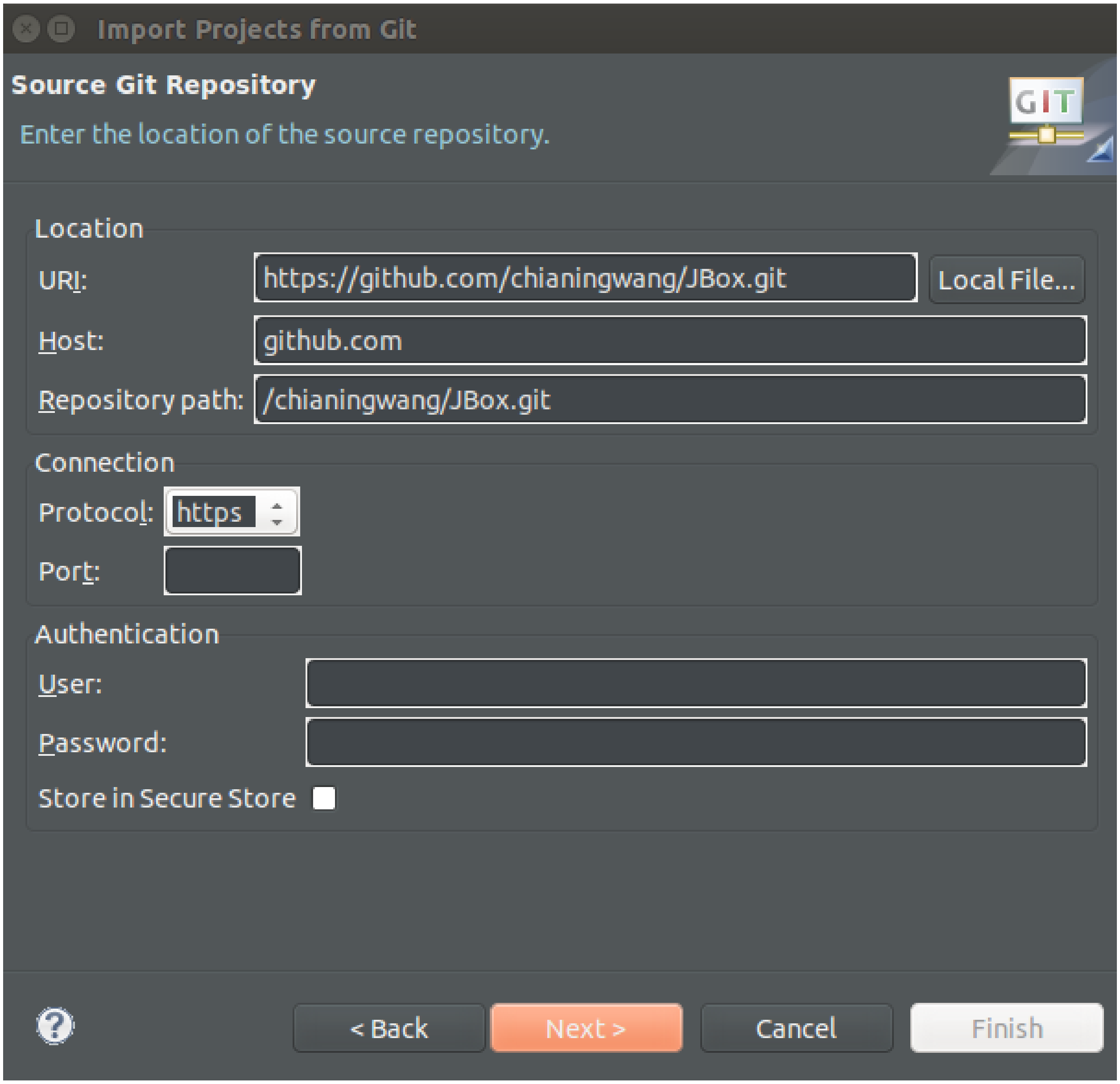
Installation and Setup

1. download the JBox source code or import into Eclipse directly

```
$ git clone https://github.com/chianingwang/JBox.git
$ cd ./JBox
```

In eclipse, right click at Package Explore: `Import --> Git --> Project from Git --> Clone URl` then paste **https://github.com/chianingwang/JBox.git** `--> next --> master --> next --> Import existing projects --> next` , then done if you miss the project file you can find .prject and .classpath under prj folder.

- Import JBox in eclipse



URI: https://github.com/chianingwang/JBox.git

Host: github.com

Repository path: /chianingwang/JBox.git

Protocol: https

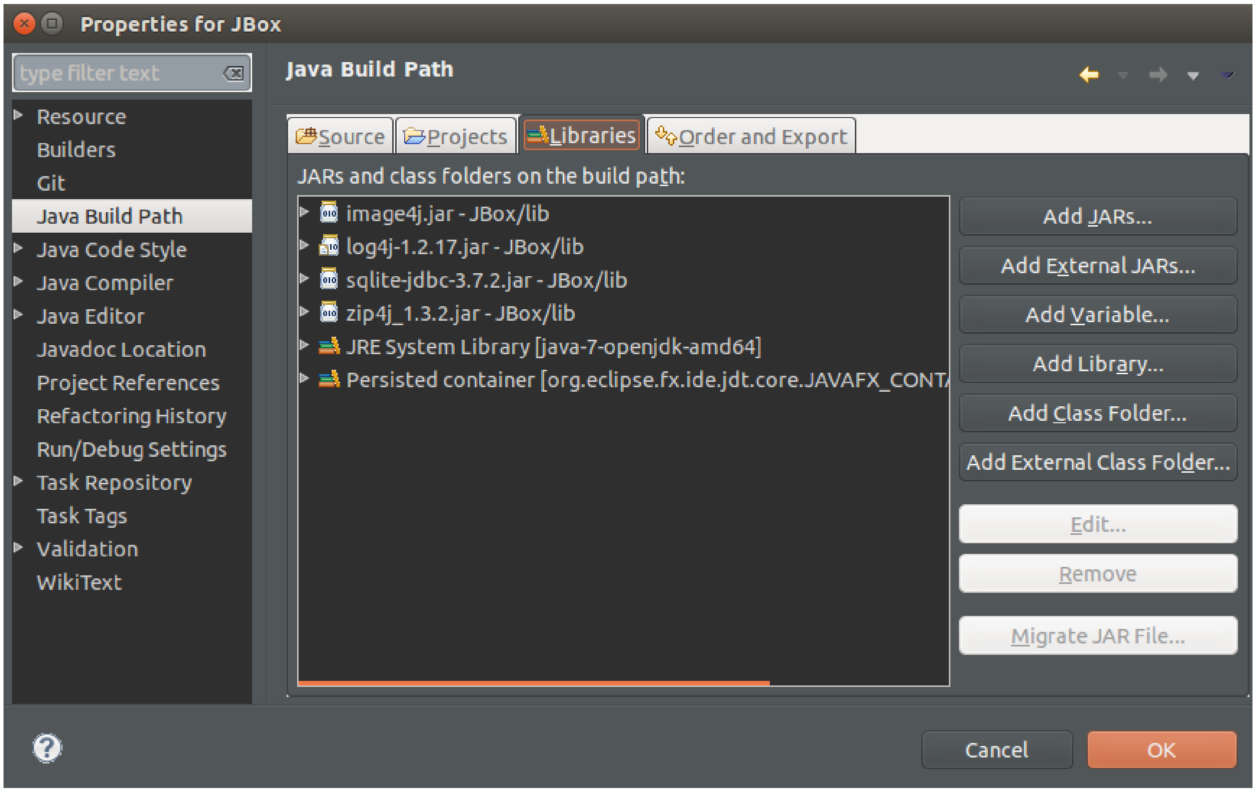
Port:

User:

Password:

2. double check reference library

- double check required lib
- Double Check Required Library (JAR).



Java Build Path

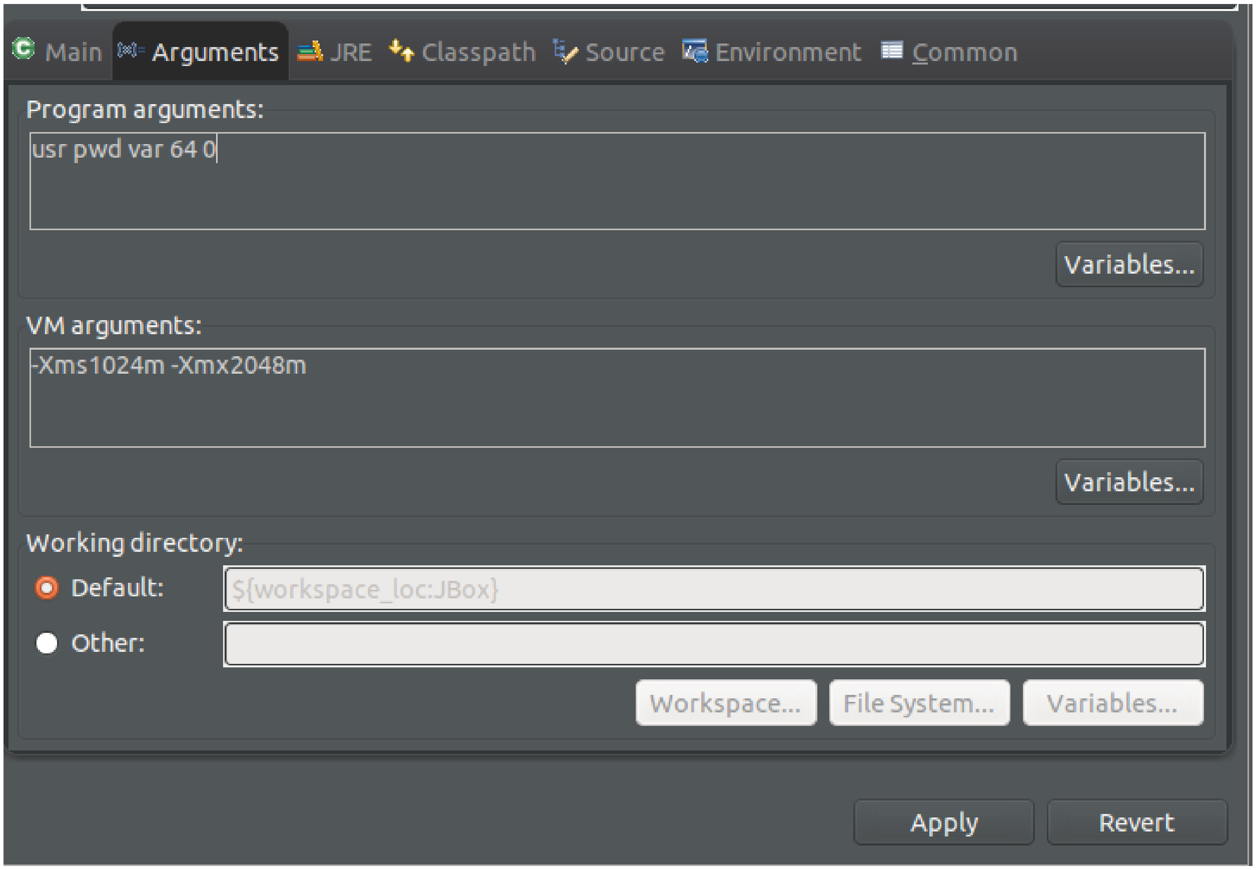
JARs and class folders on the build path:

- image4j.jar - JBox/lib
- log4j-1.2.17.jar - JBox/lib
- sqlite-jdbc-3.7.2.jar - JBox/lib
- zip4j_1.3.2.jar - JBox/lib
- JRE System Library [java-7-openjdk-amd64]
- Persisted container [org.eclipse.fx.ide.jdt.core.JAVAFX_CONT/

3. add run/debug configuration

Right click project and select `run configurations --> New Launcha Configuration --> Argument --> Prog`

- Setup Run Paramenter: e.g. usr pwd var 64 0 0
- Enlarge the Java VM cache size: VM arguements : -Xms1024m -Xmx2048m
- Configure Run Paramenters.



Program arguments:

```
usr pwd var 64 0
```

VM arguments:

```
-Xms1024m -Xmx2048m
```

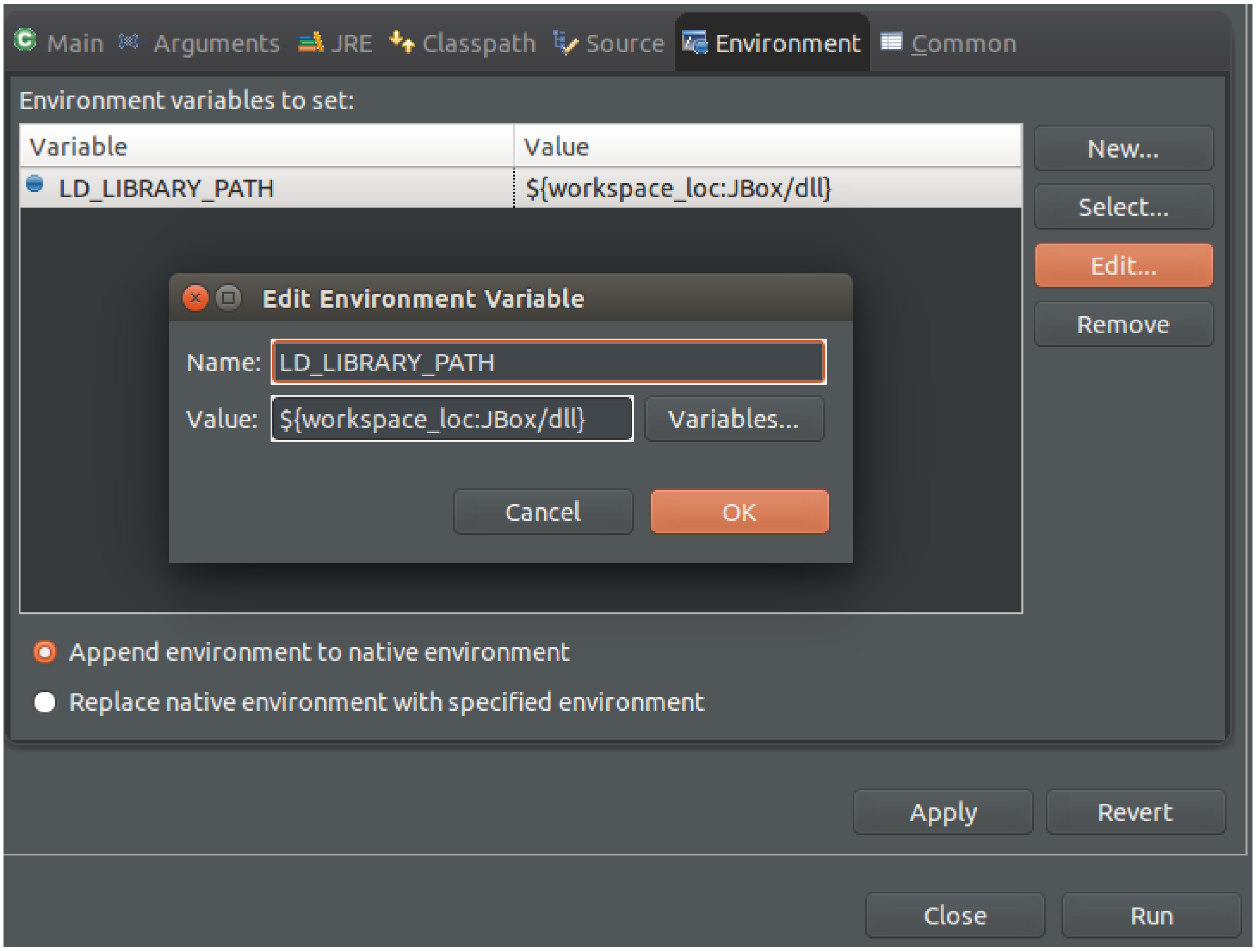
Working directory:

● Default: ${workspace_loc:JBox}

○ Other:

4. reference required `*.so ( c++ ) object`

- Add Library reference path
- Configure Reference Object Directory.



Environment variables to set:

Variable	Value
LD_LIBRARY_PATH	${workspace_loc:JBox/dll}

Edit Environment Variable

Name: LD_LIBRARY_PATH

Value: ${workspace_loc:JBox/dll}

● Append environment to native environment

○ Replace native environment with specified environment

5. Start to debug or run JBox

CHAPTER 4

Technology

In this sections we would like to discuss the technologies we applied in JBox.

What's technologies JBox adopt ?

JBox adopts **2-tier metadata structure** in order to effectively operate file system and allows to sync with multiple clients. During the file syncing, **copy on write(CoW)** makes sure metadata can be updated mutually exclusive and **Reference Counter** supports object purge to save more storage space. JBox reduces upload bandwidth and storage consumption by chunk compression and **variable chunk deduplication** which allows **Delta Sync** and **Versioning (Snapshot)** feature. **JBox** has **Dedup-Map** to make archive configurable to fit different kinds of the backup stream. It does not only control the **Dedup Anchor** for numbers of the chunks per file but also provide different kinds of deduplication skins, to try to balance between efficiency and performance.

JBox adopts the technologies and provides the features as below.

- **JBox fully leverage OpenStack swift**
- Using Swift as Repository
- Using KeyStone as Access Control
- **2-tier metadata structure** to make file system operates effectively and allows to sync with multiple clients.
- **2-tier metadata structure** can provide **light weight inotify** feature to trigger file sync execution.
- file sync is with **multiple clients** and always make a **newest backup copy in ObjectStorge, Swift**.
- **COW (copy on write)** make sure metadata update mutual exclusion
- It's **chunk-level variable deduplication** by default which allows backup stream has **Delta Sync** and **Versioning (Snapshot)** feature.
- Delta Sync only transfers the chunk containing the modification.
- It's **in-line deduplication**, which is dedup before saving the data.

- JBox **compresses** the chunk (object) before upload which reduces bandwidth and Object Storage, Swift consumption.
- JBox use **dedup-map** to make archive configurable, it allows to configure as below.
- **Dedup Anchor** for number of the chunks per file
- **Refector** limit interval for Dedup Anchor growing
- **File Level Deduplication** vs. **Chunk Level Deduplication**
- **Fixed Chunking** vs. **Variable Chunking** Deduplication
- In Config.java and will allow maintaining dedup-map.cfg for the user to adjust dynamically.
- It's using reference counter to support *metadata and object purge*.
- Purge lead time for chunk level metadata (fxxxxx)
- Purge lead time for object (c0xxxxx or c1xxxxx)
- Rename purged object as the cold storage tier, if no further reference, then purge, if objects get reference again, then rename it back w/o upload.
- **Virtual Storage Tiering** when screen the existing chunk, scan **Hot Chunks** first which is chunk(object) being the reference at least one in Swift, it can't find it then move to **Cold Chunk**, if screen can't find in both then upload new chunk to Swift.
- Phase 1: Hot Chunk is existing referenced chunk, Cold Chunk is purged chunk but hasn't delete in Swift. Dedup Screen from Hot to Cold.
- Phase 2: Hot Chunk is the chunk been referenced with certain time (e.g. 3 month), Cold Chunnk is other than that existing referneced chunk, plus Purged Chunk is the purged chunk but haven't delete in Swift yet. Dedup Screen from Hot to Cold, then Cold to Purged.

For the 2-Tier Metadata and what's the algorithm logic to identify new/update/copy/rename/move/delete can be found in here.

- Archival and Sync via ObjectStorage Swift - JBox. explain, why JBox doesn't need to adopt any extra library to do the thing like Linux inotify. In such, JBox doesn't need to reference specific file system monitor library such as FileSystemWatcher in Windows for C# or JNotify in Linux for Java.

dedup parameters definition

1. Deduplication Algorithm, var=variable chunk (content aware), fix=fix chunk and no=no chunk, it's file level
2. divider have to be number base on power of 2

```
# divider=64 example
# e.g. divider = 64
# then file size / 64 and
# get between lower bound power of 2 to upper bound power of 2,
# then Dedup Anchor = upper bound of the power of 2.
# Deduplication average size will be around Dedup Anchor.
# Here is pseudo code concept
if var in c,
then
  chunk size will be 0.85 x Dedup Anchor ~ 2 x Dedup Anchor
  number of chunk between 32 ~ 75
else if fix in c,
then
```

```
chunk size will be Dedup Anchor
number of chunk will <= 64
```

3. refactor=0 which is no refactoring or any number n

```
# Dedup Anchor 2^x will be wipe out if new Dedup 2^y,
# then (2^y) / (2^x) > n </p>
# refactor=3 example
# e.g. if Dedup Anchor = 18 ,
# then JBox will divide file size by 2^18,
# however if file grow and when we found file size
# is power of 2 upper bound is 2^22,
# then (2^22)/(2^18) = 4 > 3, then
# JBox Dedup Anchor will be wiped out
# then use 22 as Dedup Anchor.
```

4. refcounter flag, if we would like to turn on then set 1, otherwise 0.

CHAPTER 5

Algorithm

This section we would like to talk about the algorithm we adopt in JBox.

Deduplicatioin Chunking Algorithm

Mainly purpose for JBox is back up your data from local to Object Storage, thus we adopt compression and deduplication to reduce as much as possible your backup data set on the remote repository which is Object Storage, OpenStack Swift.

Fix Chunking

When we do the deduplication, the chunk size is all fixed.

Variable Chunking

When we do the deduplication, the chunk size is variable which means it will change base on the backup data stream content.

Dynamic Anchor Variable Chunking

Like we learn from the previous section, even the chunk size is variable but we still need boundary to limit the chunk size. The **Dynamic Anchor Variable Chunking** is base on the file size and compression ratio to dynamic decide Variable Chunk Boundary but keep it as Anchor in metadata, when file content change, deduplication will always apply the same rule.

CHAPTER 6

License

Apache License 2.0

Apache License Version 2.0, January 2004 http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

“License” shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

“Licensor” shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

“Legal Entity” shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, “control” means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

“You” (or “Your”) shall mean an individual or Legal Entity exercising permissions granted by this License.

“Source” form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

“Object” form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

“Work” shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

“Derivative Works” shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

“Contribution” shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, “submitted” means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as “Not a Contribution.”

“Contributor” shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a “NOTICE” text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets “{}” replaced with your own identifying information. (Don’t include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same “printed page” as the copyright notice for easier identification within third-party archives.

Copyright {yyyy} {name of copyright owner}

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

CHAPTER 7

Help

Needs Help ?

If you need any help or have any question, please log a **issue** in Github JBox Repo.

OR email to chianingwang@hotmail.com

CHAPTER 8

Indices and tables