Experiment No. 5

TITLE: Demonstrate the use of map and reduce tasks.

PREREQUISITE: Operating Systems, Computer Networks, Java

THEORY:

Introduction

Hadoop is a Java-based programming framework that supports the processing and storage of extremely large datasets on a cluster of inexpensive machines. It was the first major open source project in the big data playing field and is sponsored by the Apache Software Foundation. Hadoop 2.7 is comprised of four main layers: Hadoop Common is the collection of utilities and libraries that support other Hadoop modules.

- HDFS, which stands for Hadoop Distributed File System, is responsible for persisting data to disk.
- YARN, short for Yet Another Resource Negotiator, is the "operating system" for HDFS.
- MapReduce is the original processing model for Hadoop clusters. It distributes work within the cluster or map, then organizes and reduces the results from the nodes into a response to a query. Many other processing models are available for the 2.x version of Hadoop.

Step 1 — Installing Java

To get started, we'll update our package list:

```
sudo apt-get update
```

Next, we'll install OpenJDK, the default Java Development Kit on Ubuntu 16.04.

```
sudo apt-get install default-jdk
```

Once the installation is complete, let's check the version.

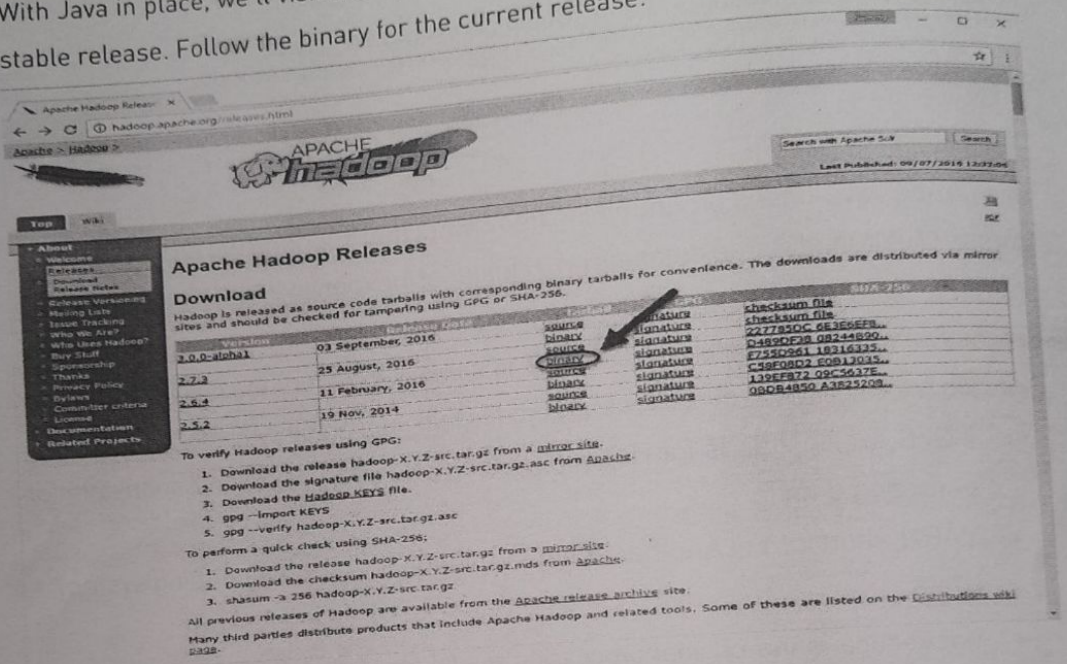
```
java -version
```

Output

```
openjdk version "1.8.0 91 "
OpenJDK Runtime Environment (build 1.8.0 91-8u91-b14-3ubuntu1-16.04.1-b14)
```

Step 2 — Installing Hadoop

With Java in place, we'll visit the Apache Hadoop Releases page to find the most recent stable release. Follow the binary for the current release:



Step 3 — Configuring Hadoop's Java Home

Hadoop requires that you set the path to Java, either as an environment variable or in the Hadoop configuration file. The path to Java, /usr/bin/java is a symlink to /etc/alternatives/java, which is in turn a symlink to default Java binary. We will use readlink with the -f flag to follow every symlink in every part of the path, recursively. Then, we'll use sed to trim bin/java from the output to give us the correct value for JAVA_HOME. To find the default Java path

```
readlink -f /usr/bin/java | sed "s:bin/java::"
```

Output

```
/usr/lib/jvm/java-8-openjdk-amd64/jre/
```

You can copy this output to set Hadoop's Java home to this specific version, which ensures that if the default Java changes, this value will not. Alternatively, you can use

the readlink command dynamically in the file so that Hadoop will automatically use whatever Java version is set as the system default.

To begin, open hadoop-env.sh:

sudo nano /usr/local/hadoop/etc/hadoop/hadoop-env.sh

Then, choose one of the following options:

Option 1: Set a Static Value

```
                    /usr/local/hadoop/etc/hadoop/hadoop-env.sh
...
#export JAVA_HOME=${JAVA_HOME}
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre/
...
```

Copy

Option 2: Use Readlink to Set the Value Dynamically

```
                    /usr/local/hadoop/etc/hadoop/hadoop-env.sh
...
#export JAVA_HOME=${JAVA_HOME}
export JAVA_HOME=$(readlink -f /usr/bin/java | sed "s:bin/java::")
```

Step 4 — Running Hadoop

Now we should be able to run Hadoop:

```
/usr/local/hadoop/bin/hadoop
```

Output

```
Usage: hadoop [--config confdir] [COMMAND | CLASSNAME]
 CLASSNAME           run the class named CLASSNAME
or
 where COMMAND is one of:
 fs                  run a generic filesystem user client
 version             print the version
 jar <jars>          run a jar file
                     note: please use "yarn jar" to launch
```

```
                YARN applications, not this command.
checknative [-a|-h]  check native hadoop and compression libraries availability
distcp <srcurl> <desturl> copy file or directories recursively
archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archive
classpath         prints the class path needed to get the
credential        interact with credential providers
                Hadoop jar and the required libraries
daemonlog         get/set the log level for each daemon
```

The help means we've successfully configured Hadoop to run in stand-alone mode. We'll ensure that it is functioning properly by running the example MapReduce program it ships with. To do so, create a directory called input in our home directory and copy Hadoop's configuration files into it to use those files as our data.

```
mkdir ~/input
cp /usr/local/hadoop/etc/hadoop/*.xml ~/input
```

Next, we can use the following command to run the MapReduce hadoop-mapreduce-examples program, a Java archive with several options. We'll invoke its grep program, one of many examples included in hadoop-mapreduce-examples, followed by the input directory, input and the output directory grep_example. The MapReduce grep program will count the matches of a literal word or regular expression. Finally, we'll supply a regular expression to find occurrences of the word principal within or at the end of a declarative sentence. The expression is case-sensitive, so we wouldn't find the word if it were capitalized at the beginning of a sentence:

```
/usr/local/hadoop/bin/hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep ~/input ~/grep_example 'principal[.]*'
```

When the task completes, it provides a summary of what has been processed and errors it has encountered, but this doesn't contain the actual results.

STANDARD INPUT AND OUTPUT:

Output

```
File System Counters
        FILE: Number of bytes read=1247674
        FILE: Number of bytes written=2324248
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
    Map-Reduce Framework
        Map input records=2
        Map output records=2
        Map output bytes=37
        Map output materialized bytes=47
        Input split bytes=114
        Combine input records=0
        Combine output records=0
        Reduce input groups=2
        Reduce shuffle bytes=47
        Reduce input records=2
        Reduce output records=2
        Spilled Records=4
        Shuffled Maps =1
        Failed Shuffles=0
        Merged Map outputs=1
        GC time elapsed (ms)=61
        Total committed heap usage (bytes)=263520256
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=151
    File Output Format Counters
        Bytes Written=37
```

Laboratory Report

Experiment No - 6

Batch - C-3

Date of Experiment: 31/10/22 Date of Submission: 7/11/22

Title: Design and analyze architecture of Aneka identify different entities to understand the structure

---

Evaluation

1) Attendance [2] 2
2) Lab Performance [2] 2
3) Oral [1] 1

Overall Marks [5] 5

Subject Incharge

Experiment No. 6

TITLE: Design and analyze architecture of Aneka identify different entities to understand the structure of it

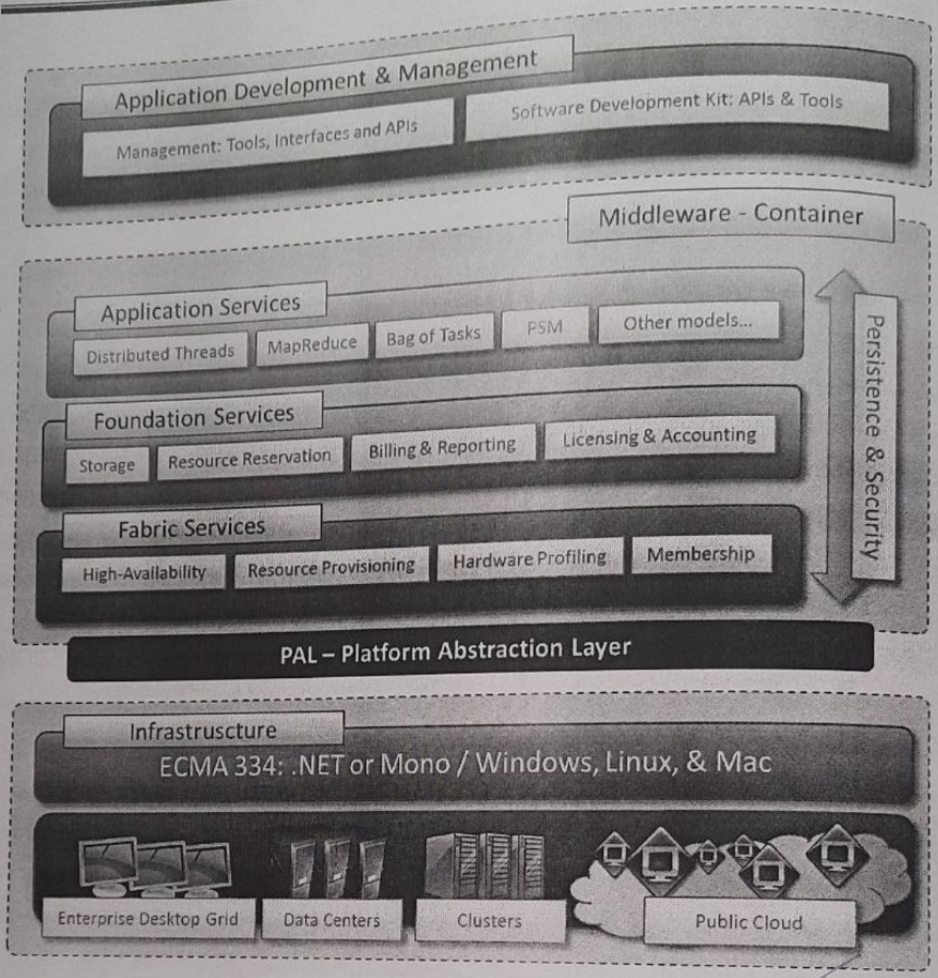
PREREQUISITE: Operating Systems, Computer networks

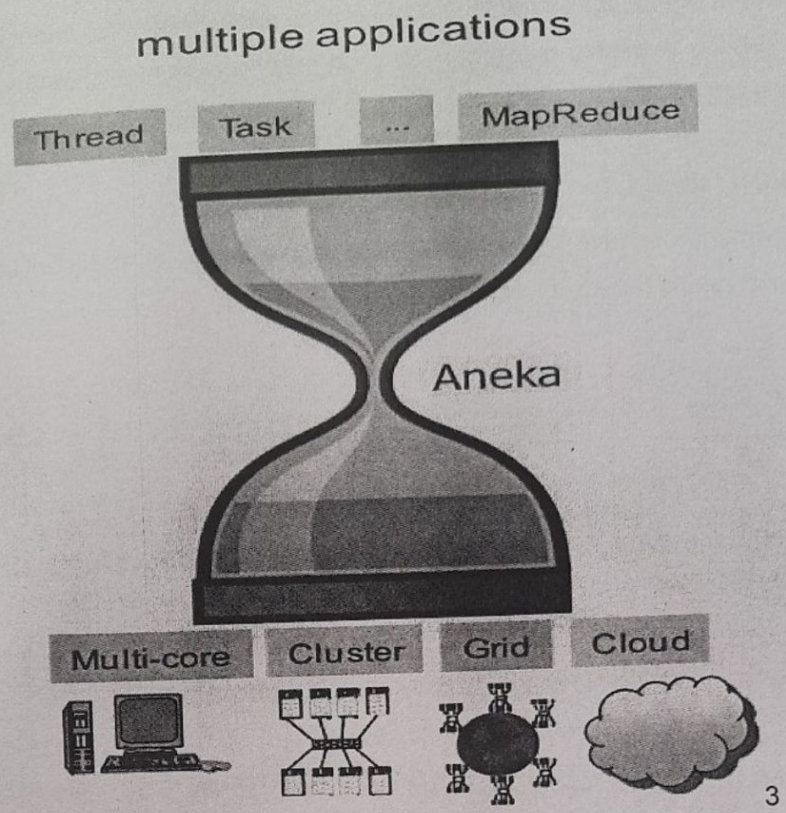
THEORY:

Aneka Architecture

Aneka is a platform and a framework for developing distributed applications on the Cloud. It harnesses the spare CPU cycles of a heterogeneous network of desktop PCs and servers or datacenters on demand. Aneka provides developers with a rich set of APIs for transparently exploiting such resources and expressing the business logic of applications by using the preferred programming abstractions. System administrators can leverage on a collection of tools to monitor and control the deployed infrastructure. This can be a public cloud available to anyone through the Internet, or a private cloud constituted by a set of nodes with restricted access.

The Aneka based computing cloud is a collection of physical and virtualized resources connected through a network, which are either the Internet or a private intranet. Each of these resources hosts an instance of the Aneka Container representing the runtime environment where the distributed applications are executed. The container provides the basic management features of the single node and leverages all the other operations on the services that it is hosting. The services are broken up into fabric, foundation, and execution services. Fabric services directly interact with the node through the Platform Abstraction Layer (PAL) and perform hardware profiling and dynamic resource provisioning. Foundation services identify the core system of the Aneka middleware, providing a set of basic features to enable Aneka containers to perform specialized and specific sets of tasks. Execution services directly deal with the scheduling and execution of applications in the Cloud.


Application Development & Management
Management: Tools, Interfaces and APIs
Software Development Kit: APIs & Tools
Middleware - Container
Application Services
Distributed Threads
MapReduce
Bag of Tasks
PSM
Other models...
Foundation Services
Storage
Resource Reservation
Billing & Reporting
Licensing & Accounting
Fabric Services
High-Availability
Resource Provisioning
Hardware Profiling
Membership
Persistence & Security
PAL – Platform Abstraction Layer
Infrastruscture
ECMA 334: .NET or Mono / Windows, Linux, & Mac
Enterprise Desktop Grid
Data Centers
Clusters
Public Cloud



3

One of the key features of Aneka is the ability of providing different ways for expressing distributed applications by offering different programming models; execution services are mostly concerned with providing the middleware with an implementation for these models. Additional services such as persistence and security are transversal to the entire stack of services that are hosted by the Container. At the application level, a set of different components and tools are provided to:

1) simplify the development of applications (SDK);
2) porting existing applications to the Cloud; and
3) monitoring and managing the Aneka Cloud.

A common deployment of Aneka is presented at the side. An Aneka based Cloud is constituted by a set of interconnected resources that are dynamically modified according to the user needs by using resource virtualization or by harnessing the spare CPU cycles of desktop machines. If the deployment identifies a private Cloud all the

resources are in house, for example within the enterprise. This deployment is extended by adding publicly available resources on demand or by interacting with other Aneka public clouds providing computing resources connected over the Internet.

CONCLUSION / RESULT:

In this Experiment, we have analyzed the architecture of Aneka and identified different entities to understand the structure of it

SES
SHIRPUR EDUCATION SOCIETY
॥ सा विद्या या विमुक्तये ॥

Laboratory Report

Experiment No - 7

Batch - C-3

Date of Experiment: 7/11/22 Date of Submission: 14/11/22

Title: Installation and configuration of Aneka master node and execution of Convolution imaging application

---

Evaluation

1) Attendance [2] 02

2) Lab Performance [2] 01

3) Oral [1] 01

Overall Marks [5] 04

Subject Incharge

Experiment No. 7

TITLE: Installation and configuration of Aneka master node and execution of Convolution imaging application

PREREQUISITE: Operating Systems, Computer Nertworks

THEORY:

Aneka is a Cloud Application Development Platform for developing and running compute and data intensive applications. As a platform it provides users with both a runtime environment for executing applications developed using any of the three supported programming models, and a set of APIs and tools that allow you to build new applications or run existing legacy code. The purpose of this document is to help you through the process of installing and setting up an Aneka Cloud environment. This document will cover everything from helping you to understand your existing infrastructure, different deployment options, installing the Management Studio, configuring Aneka Daemons and Containers, and finally running some of the samples to test your environment.

An Aneka Cloud is composed of a collection of services deployed on top of an infrastructure. This infrastructure can include both physical and virtual machines located in your local area network or Data Centre. Aneka services are hosted on Aneka Containers which are managed by Aneka Daemons. An Aneka Daemon is a background service that runs on a machine and helps you to install, start, stop, update and reconfigure Containers.

A key component of the Aneka platform is the Aneka Management Studio, a portal for managing your infrastructure and clouds. Administrators use the Aneka Management Studio to define their infrastructure, deploy Aneka Daemons, and install and configure Aneka Containers. The figure below shows a high-level representation of an Aneka Cloud, composed of a Master Container that is responsible for scheduling jobs to Workers, and a group of Worker Containers that execute the jobs. Each machine is typically configured with a single instance of the Aneka Daemon and a single instance of the Aneka Container.

Installation

This section assumes that you have a copy of the Aneka distribution with you. If you do not have a copy already, you can download the latest version from Manjrasoft's Website.

Installing Aneka Cloud Management Studio

Aneka installation begins with installing Aneka Cloud Management Studio. The Cloud Management Studio is your portal for creating, configuring and managing Aneka Clouds. Installing Aneka using the distributed Microsoft Installer Package (MSI) is a quick process involving three steps as described below.

Step 1 – Run the installer package to start the Setup Wizard

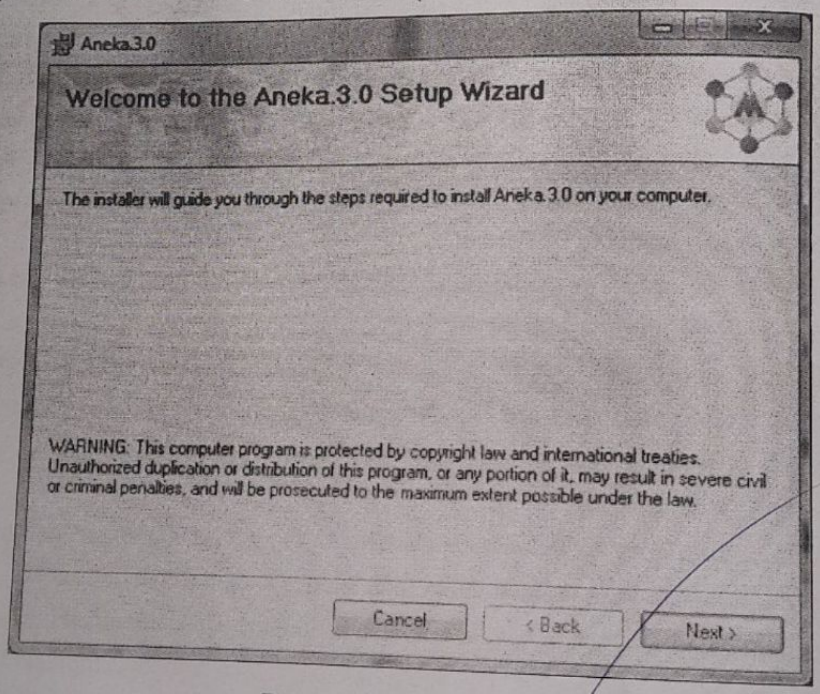


Figure - Welcome Page

The Welcome Page is self-explanatory and you can proceed by clicking next.

Step 2 – Specifying the installation folder

In Step 2 you specify the installation folder. By default Aneka is installed in C:\Program Files\Manjrasoft\Aneka.3.0.

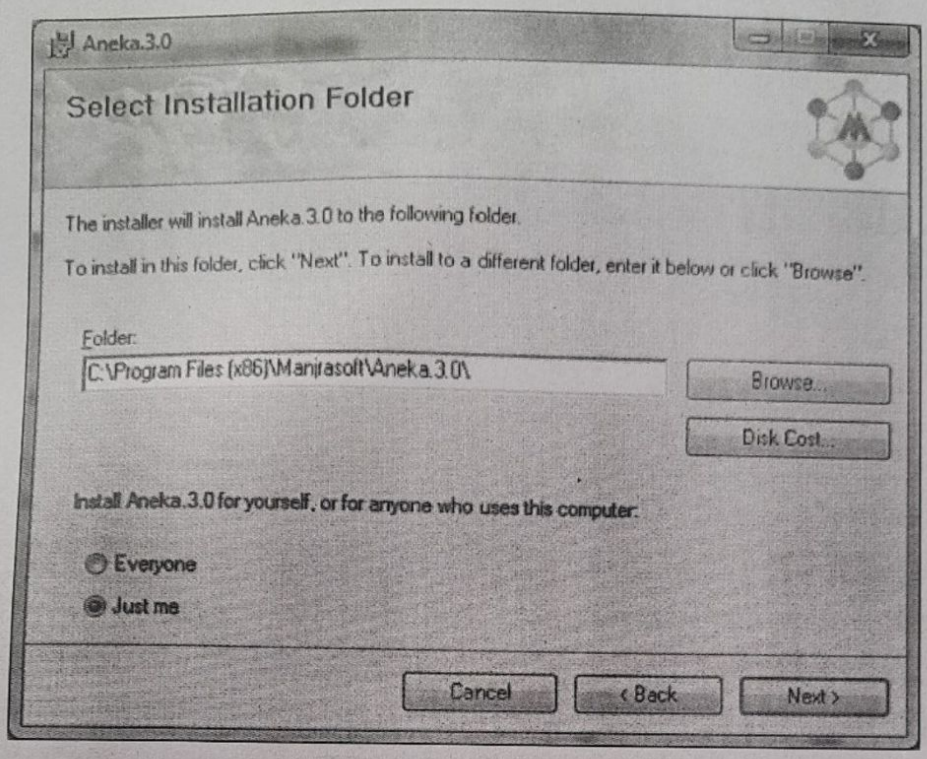


Figure - Specifying the installation folder

Step 3 – Confirm and start the installation

At this point you are ready to begin the installation. Click —"Next" to start the installation or —"Back" to change your installation folder.

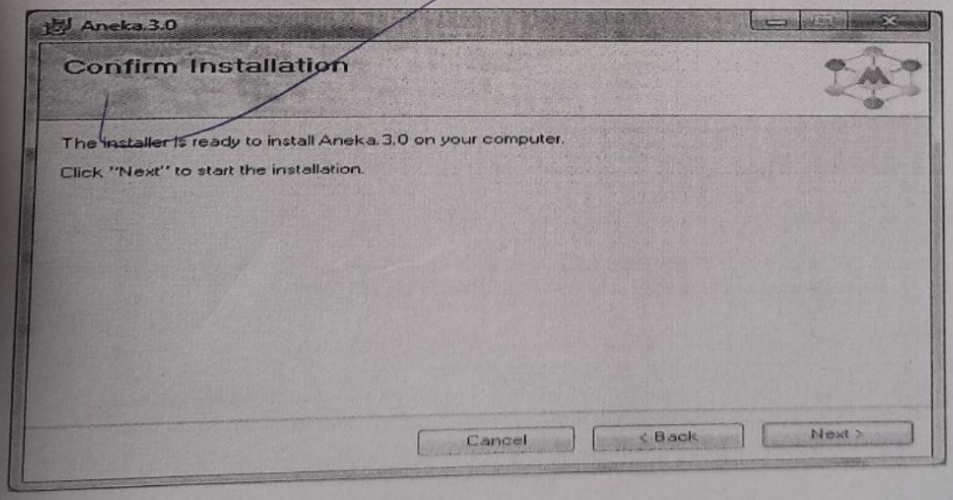


Figure - Confirm Installation

Once the installation is complete, close the wizard and launch Aneka Management Studio from the start menu.

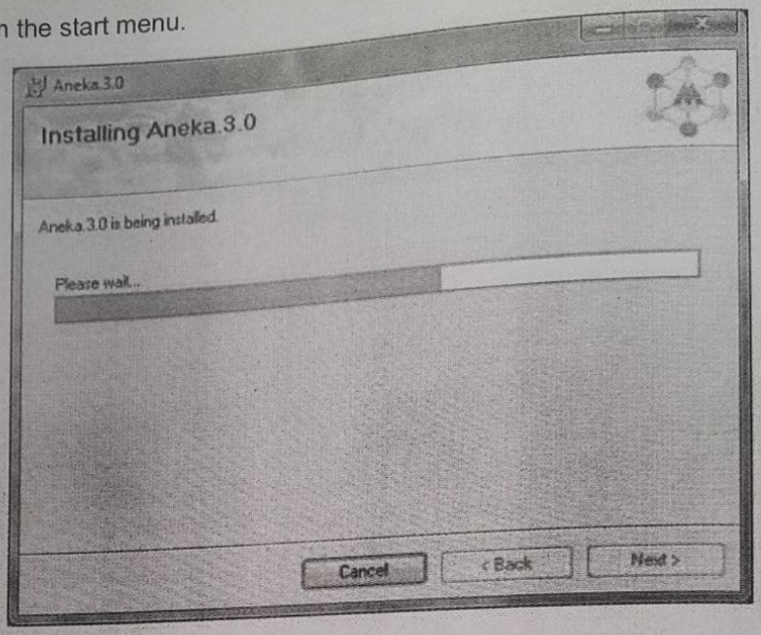


Figure - Installation Progress

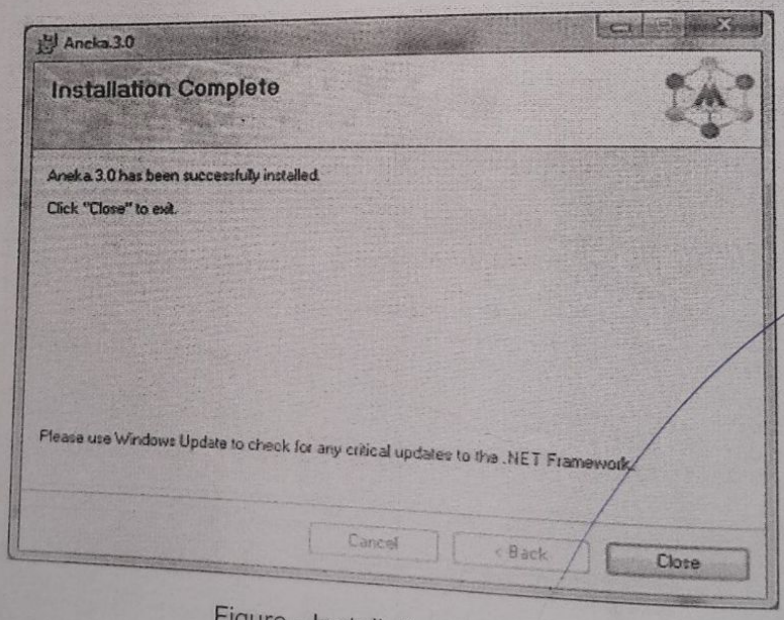


Figure - Installation Complete

Figure - Start Menu

Aneka Cloud Management Studio

The Aneka Cloud Management Studio is your portal for managing your infrastructure and clouds. It provides facilities for defining your underlying cloud infrastructure and creating one or more Aneka Clouds on top. It lets you create and manage Aneka user accounts, monitor the overall performance of your Cloud, obtain detailed reporting information on resource usage, data transfers, billing and application (job) execution. It also provides facilities for troubleshooting your deployments by allowing you to access and examine remote logs.

rting up Management Studio

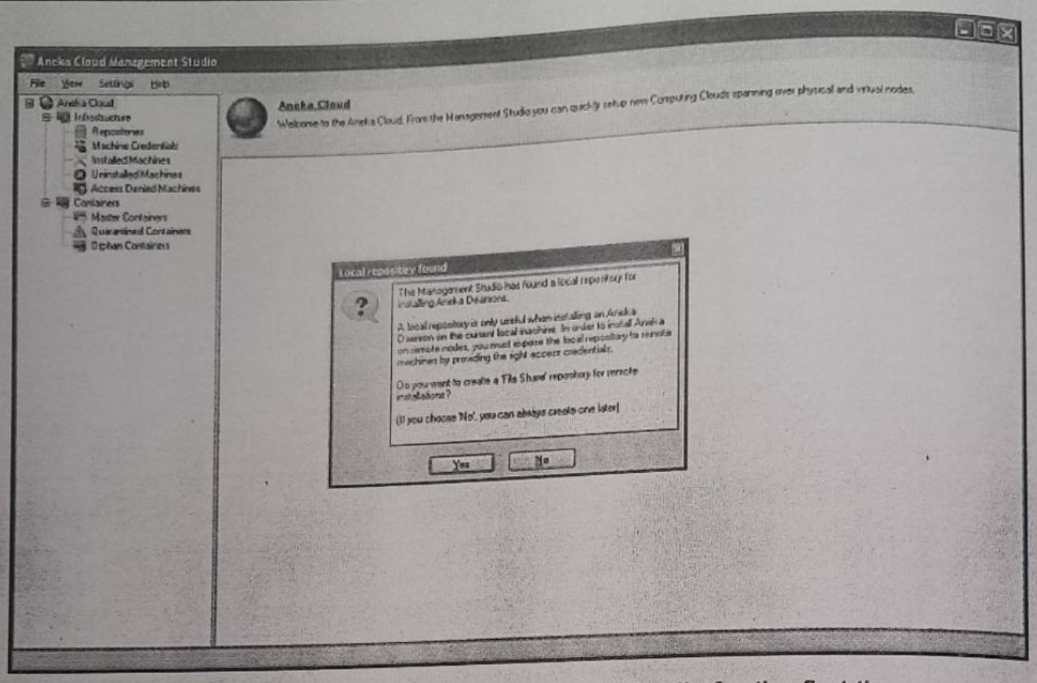


Figure - Starting Aneka Cloud Management Studio for the first time.

When Aneka Cloud Management Studio is started up for the first time you'll be asked to create a Remote Repository for performing remote installations. Setting up a Remote Repository requires selecting a suitable repository type and supplying valid credentials which remote machines can use to connect and download required files. You may however choose to create this repository at a late time before making remote installations. If no repository is defined, you will be restricted to making local installations only.

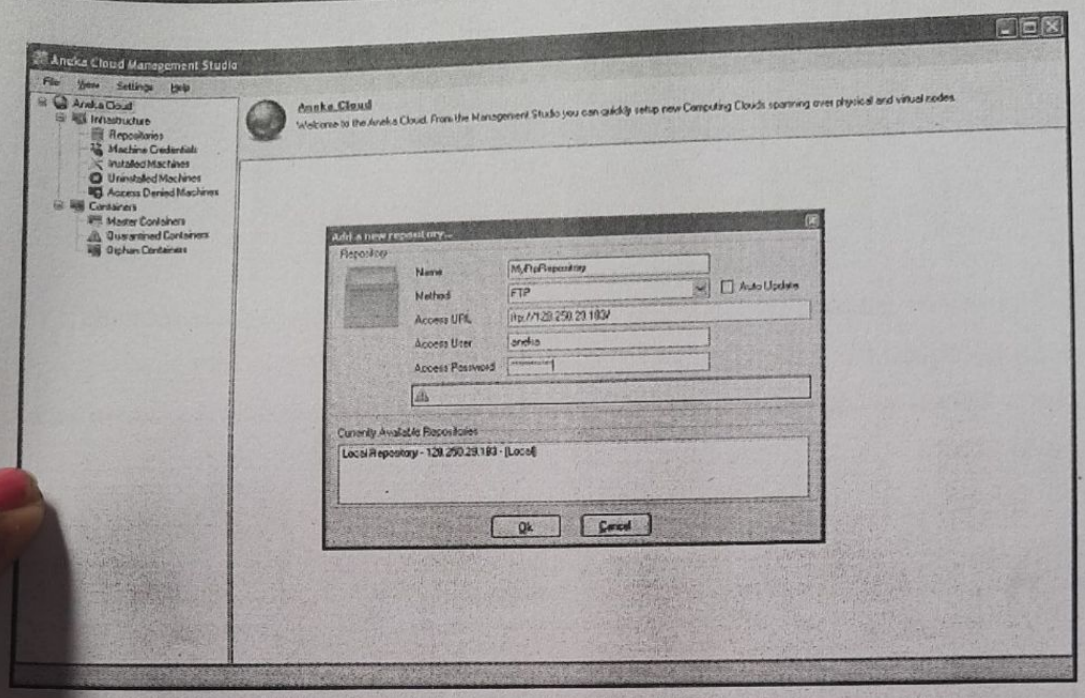


Figure - Creating a repository for remote installations

Shutting down Aneka Management Studio

When attempting to shut down Aneka Management Studio, you will be given the choice of saving all configuration data from the current session. It is highly recommended that you save this information and restore it the next time you start using the Management Studio.

The Configuration File

The configuration file, ManagementStudio.config, contains all information that describes your infrastructure, your Clouds, the machine credentials, repositories and authentication keys (see section on installing the Master Container) that you defined when using Aneka Management Studio. It is recommended that you save this information when you exit Management Studio so that you can restore it at a later session, and get up-to-speed with your Cloud management without having to redefine all settings again. Some configuration information, such as authentication keys, must be maintained safely if you are to add new Containers to your existing Cloud. Losing an authentication key however, is not detrimental as you will be able to reconfigure your clouds with a new key.

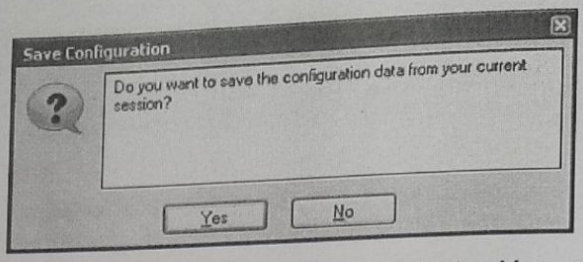


Figure - Request to save configuration data when closing Management Studio

The configuration file is always encrypted before being written to disk for security reasons. When saving configuration data you will be required to specify a password as show in Figure.

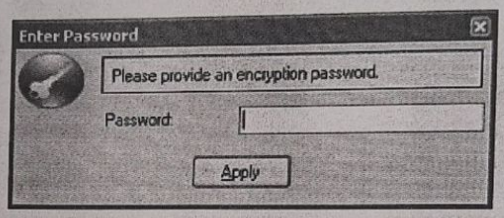


Figure - Password to encrypt configuration data

When starting up Aneka Management Studio at a later session, you will be given the choice of restoring your configuration data. If you choose to do so, you must re-enter the same password you used when saving.

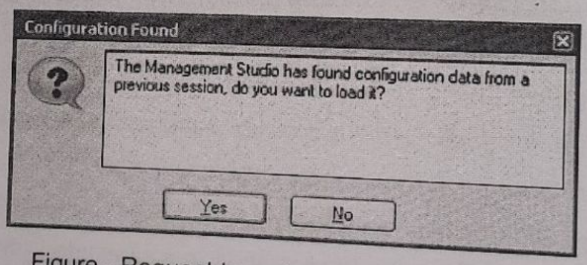


Figure - Request to restore configuration data

CONCLUSION / RESULT:

In this Experiment, we have Installed and configured Aneka master node and executed Convolution imaging application.