

IPNC GUI

Development and Design Guide

Document Version 0.12

Copyright © 2014 Texas Instruments Incorporated. All rights reserved.

Information in this document is subject to change without notice. Texas Instruments may have pending patent applications, trademarks, copyrights, or other intellectual property rights covering matter in this document. The furnishing of this documents is given for usage with Texas Instruments products only and does not give you any license to the intellectual property that might be contained within this document. Texas Instruments makes no implied or expressed warranties in this document and is not responsible for the products based from this document.

IPNC Reference Design on DM36x

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products

Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DSP	dsp.ti.com
Clocks and Timers	www.ti.com/clocks
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
RFID	www.ti-rfid.com
RF/IF and ZigBee® Solutions	www.ti.com/lprf

Applications

Audio	www.ti.com/audio
Automotive	www.ti.com/automotive
Broadband	www.ti.com/broadband
Digital Control	www.ti.com/digitalcontrol
Medical	www.ti.com/medical
Military	www.ti.com/military
Optical Networking	www.ti.com/opticalnetwork
Security	www.ti.com/security
Telephony	www.ti.com/telephony
Video & Imaging	www.ti.com/video
Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

Copyright 2014, Texas Instruments Incorporated

RevisionHistory

Version	Date	RevisionHistory
0.12	5 March 2014	Updated documentation for the build issues in npapi-vlc. Sections 5.1.1.5 and 5.1.2.4 are updated.

TABLE OF CONTENTS

1 Introduction.....	6
1.1 Overview.....	6
1.2 Design Goals.....	6
1.3 Abbreviations.....	6
2 Key Design Elements.....	7
3 Components.....	7
3.1 Video Plugin.....	7
3.1.1 Responsibility.....	7
3.1.2 Interaction.....	7
3.2 User Interface.....	7
3.2.1 Responsibility.....	8
3.2.2 Interaction.....	8
3.2.2.1 Camera Interaction.....	8
3.2.2.2 Video plugin interaction.....	8
4 Source Code Information.....	9
4.1 Video Plugin.....	9
4.1.1 Directory Structure.....	9
4.2 User Interface.....	10
4.2.1 Directory Structure.....	10
5 Building the application.....	10
5.1 Video plugin.....	10
5.1.1 Build System Configuration.....	11
The prerequisites for the set up of build machine are following.....	11
5.1.1 Building for Linux.....	11
5.1.1.1 Preparing the sources.....	11
5.1.1.2 Building live555-20121218:.....	11
5.1.1.3 Building ffmpeg-0.8.14.....	12
5.1.1.4 Building vlc-2.0.6 :.....	12
5.1.1.5 Building npapi-vlc :.....	12
5.1.1.6 Creating the installer package.....	12
5.1.2 Building for Windows.....	12
5.1.2.1 Building Live555 :.....	13
5.1.2.2 Building ffmpeg :.....	13
5.1.2.3 Building vlc-2.0.6 :.....	13
5.1.2.4 Building npapi-vlc :.....	13
5.1.2.5 Creating the Installer package.....	13
5.1.2.5.1 Steps on Linux.....	13
5.1.2.5.2 Steps on Windows.....	14
5.2 User Interface.....	15
5.2.1 Linux.....	15
5.2.2 Creating Release package	15

6 Virtual Box and Ubuntu image Installation.....	16
6.1 Procedure to install the virtual image.....	16
7 Building the ipnc packages through mount folder (shared folder).....	26
7.1 Create the shared folder as follows	26
8 Integrating the application with the firmware.....	29

1 Introduction

1.1 Overview

The GUI for the IPNC Reference Design Kit, is used to configure the IPNC and to view the video streams on your system.

This document describes the key aspects of the IPNC GUI development. These include the key components, build and integration instructions. It also includes some points to customize the User Interface for your needs.

1.2 Design Goals

The key design goals of the application are as follows.

1. The application must work on multiple operating systems (Windows and Linux)
2. The application must work across multiple browsers (Microsoft Internet Explorer (version 7 upwards), Mozilla Firefox and Google Chrome)

1.3 Abbreviations

IPNC	IP Network Camera
GUI	Graphical User Interface
APP	Application (refers to the browser based GUI including the video plugin)
TI	Texas Instruments
API	Application Programming Interface
NPAPI	Netscape Plugin API

2 Key Design Elements

The GUI has been designed using open source technologies. The solution does not have any proprietary software other than TI's IP.

In order to support multiple browsers HTML + Javascript have been chosen. In order to support video playback in the browser we have chosen the Vlc browser plugin (npapi-vlc).

3 Components

3.1 Video Plugin

The video plugin is based on npapi-vlc and libVlc. This open source project builds a wrapper around libVlc which enables video playback across browsers.

libVlc is the media framework which is used by the Vlc application for all its core media functionality.

Npapi-vlc has been extended with a set of APIs and Event Callback (to help it interact with the User Interface code).

3.1.1 Responsibility

The video plugin is responsible for the following.

1. Video playback (streaming, demuxing, decoding and scaling)
2. Handling overlays.
3. Setting and getting zone and region data for different Analytic algorithms and Advanced Video features.
4. Parsing of stream metadata for event information.

3.1.2 Interaction

The video plugin interacts with the HTML/Javascript user interface based on a set of APIs. These APIs are a set of Get/Set properties which are exposed by the plugin. These APIs are available via Active X (for Internet Explorer) and as NPAPI (for all non IE browsers).

The APIs are divided into two categories.

1. Vlc APIs : These APIs are the stock APIs available with the Vlc browser plugin (npapi-vlc). The video plugin does not modify any of these.
2. Tilpnc : These APIs are the extensions to the Vlc plugin (npapi-vlc). They are used by the User Interface code to do get and set region data for various overlays, set resolution and scaling related parameters.

3.2 User Interface

The user interface has been developed using the Twitter Bootstrap and the less CSS frameworks.

The User Interface has been designed as a Single Page Application. Essentially the page load happens only at the beginning. Subsequently the application runs on the browser.

The screen changes are done entirely on the client end using Javascript calls.

3.2.1 Responsibility

The HTML User Interface is responsible for the following.

1. Look and Feel
2. Interacting with the Camera (via Ajax) for configuration and representing the Camera state on the UI.
3. Instantiating and configuring the Video plugin.
4. Authentication and access.

3.2.2 Interaction

The UI code interacts with the Camera and also with the Video plugin.

3.2.2.1 *Camera Interaction*

The camera interaction happens through Ajax. This application uses the existing HTTP interface (ini.htm,vb.htm) to interact with the camera. Typically before the load of each screen the current camera state is fetched. This is parsed and the User Interface is accordingly displayed. Any changes made by the user are submitted back to the camera using Ajax. This avoids any page load.

3.2.2.2 *Video plugin interaction*

Following is a brief description of the general flow of execution which is followed for all screens.

The interaction starts with instantiating the plugin. This is done using the <object> or the <embed> tag.

The video resolution and the window sizes are explicitly passed to the plugin. This is required for handling some call sequence differences (while creating the rendering window).

Subsequently any data that needs to be set on the plugin (depending on the screen) is set. This is followed by setting the identifier for the screen (to enable the plugin to handle any screen specific logic).

Any further interaction is based on the User Interface behavior.

4 Source Code Information

This section provides details of source code structure, how to build, how to create packages, how to generate MSI/CAB installer and install instructions for this application. The intention is not to explain the complete source code but mainly to familiarize the user with the source code so that they can browse it on their own.

4.1 Video Plugin

4.1.1 Directory Structure

Top level directories	
\	
vlc-2.0.6	VLC version 2.0.6 (Open Source)
npapi-vlc	Ability to be used as a browser plugin which supports VLC
live555-20121218	Uses open standards like RTSP for streaming(Open Source)
ffmpeg-0.8.14	For decoding video (Open Source)
npapi-vlc directory structure	
npapi-vlc	
npapi-vlc/npapi	All VLC plugin call handler for non-IE browser
npapi-vlc/active	All VLC plugin call handler for IE browser
npapi-vlc/Ports	Drawing operation definition for both windows and Linux.
npapi-vlc/Wrapper	
npapi-vlc/Wrapper/BrowserIL	Interface and implementation for interacting with the UI Code (NPAPI and Active X calls are routed here)
npapi-vlc/Wrapper/DrawingObjects	Wrapper for all draw call
npapi-vlc/Wrapper/FuntionalObjects	Data gets manipulated coming from camera to represent the zones.
npapi-vlc/Wrapper/ScreenManager	Mouse event call from IPNC GUI screens gets diverted to lower level
npapi-vlc/Wrapper/Utils	Utility functions for Wrapper folder
npapi-vlc/build	Contains all build script for both windows and Linux
npapi-vlc/install_scripts	Contains the install script for both windows and Linux

4.2 User Interface

4.2.1 Directory Structure

Top level directories	
\	
Bootjs	Contains Twitter Bootstrap libraries (Open Source)
Less	Less css framework. Used to generate css files.(Open Source Lib)
Scripts	Contains the scripts to generate packages
Js	Contains the code for all the UI screens.
Img	Contains all the images that gets displayed on user interface.
JS directory structure	
Js/ipnc.js	Handles functionalities like: Updating data modified by user via UI, Parsing received data from camera, handling all types of popup models across the IPNC app.
Js/polish.js	Used to generate DOM files. This file contains helper functions to generate the User Interface templates dynamically. It also handles Radio button, check box, input box operations.
Js/polyZone.js	Handles all types of zone related operations.
Js/config.js	Contains all configuration const like URL, logLevel, UI and PluginVersion etc.
Js/<screen specific files>	Each and every screen has a specific file, sync with the screen name. All the files are implemented for respective page specific operations.

5 Building the application

5.1 Video plugin

Builds for the plugin require Linux. Both Windows and Linux builds can be done on Linux. The build machine has to be configure before we can go ahead and give builds.

In order to ease the configuration we have provided a pre-configured Virtual Box Image. In case you are using the Virtual Box Image for your builds you can skip the next step (Build System Configuration).

5.1.1 Build System Configuration

The prerequisites for the set up of build machine are following.

1. Ubuntu 12.04 Installation 32 bit
2. Install vlc dependencies

\$ sudo apt-get build-dep vlc

3. Additional packages (Basically for windows build)

Install following dependencies like:

1. gcc-mingw-w64-i686
2. dh-autoreconf
3. gcc-mingw-w64
4. g++-mingw-w64
5. subversion
6. libqt4-dev
7. yasm
8. lua5.1
9. mingw-w64-dev (version 3.0 needs to be installed) install as follows:

\$ sudo dpkg -i mingw-w64-dev_3.0~svn4933-1_all.deb

5.1.1 Building for Linux

Npapi-vlc has a dependency chain. In order to build it, the dependencies would have to be build first.

Following steps should be followed (in the order mentioned below).

5.1.1.1 *Preparing the sources*

Unzip all the four zip archives in a folder. After unzipping you should get the the following folders

1. live555-20121218
2. ffmpeg-0.8.14
3. vlc-2.0.6
4. npapi-vlc

Following heading always should be started from the top folder means where the above four folders are available.

5.1.1.2 *Building live555-20121218:*

1. `cd live555-20121218/`
2. `./genMakefiles linux`

3. `sudo ./build-live555`

5.1.1.3 *Building ffmpeg-0.8.14*

1. `cd ffmpeg-0.8.14`
2. `sudo ./build-linux-release` (Release Build)
3. `sudo ./build-linux-debug` (Debug Build)

5.1.1.4 *Building vlc-2.0.6 :*

1. `cd vlc-2.0.6/build/scripts`
2. `./build-nix-release ../../ffmpeg-0.8.14` (Release Build)
3. `./build-nix-debug ../../ffmpeg-0.8.14` (Debug Build)

5.1.1.5 *Building npapi-vlc :*

1. `cd npapi-vlc/npapi`
2. open Makefile.am and do the following changes:
 - In SOURCES_support option (line no 138) do changes at last two lines .


```
vlcplugin_gtk.h \
$(IPNC_LINUX_sources)
```
 - SOURCES_support option (line no 165) do changes at last two lines


```
../common/win32_vlcwnd.h (remove backward slash)
# $(IPNC_WIN32_sources) (Comment out)
```
3. `cd npapi-vlc/build/scripts`
4. `./build-linux-release ../../vlc-2.0.6` (Release Build)
5. `./build-linux-debug ../../vlc-2.0.6` (Debug Build)

5.1.1.6 *Creating the installer package*

1. `cd npapi-vlc/build/script/`
2. `./linux_release_package ../../vlc-2.0.6 ../../npapi-vlc ../../ffmpeg-0.8.14`
(Release Build)
3. `./linux_debug_package ../../vlc-2.0.6 ../../npapi-vlc ../../ffmpeg-0.8.14`
(Debug Build)

5.1.2 **Building for Windows**

As discussed earlier, Windows builds are also done on Linux (cross compiled using the mingw tool chain).

The Vlc dependencies (live555 and ffmpeg) are present as part of the Vlc archive. This is different from Linux.

Following points should always be started from the top folder where all the 4 folders (ffmpeg, vlc, npapi, live555) are available .

5.1.2.1 *Building Live555 :*

1. `cd vlc-2.0.6/contrib/win32/`
2. `make .live555`

5.1.2.2 *Building ffmpeg :*

- ```

cd vlc-2.0.6/contrib/win32/
make .ffmpeg

```

#### 5.1.2.3 *Building vlc-2.0.6 :*

1. `cd vlc-2.0.6/build/scripts`
2. `./build-win-release` (Release Build)
3. `./build-win-debug` (Debug Build)

#### 5.1.2.4 *Building npapi-vlc :*

1. `cd npapi-vlc/npapi`
2. open Makefile.am and do the following changes:
  - In SOURCES\_support option (line no 138) do changes at last two lines .
 

```

 vlcplugin_gtk.h (remove backward slash)
 # $(IPNC_LINUX_sources) (Comment out)

```
  - SOURCES\_support option (line no 165) do changes at last two lines
 

```

 ../common/win32_vlcwnd.h \ (Add backward slash)
 $(IPNC_WIN32_sources) (Uncomment)

```
3. `cd npapi-vlc/build/scripts`
4. `./build-win-release ../../../vlc-2.0.6` (ReleaseBuild)
5. `./build-win32-debug ../../../vlc-2.0.6` (Debug Build)

#### 5.1.2.5 *Creating the Installer package*

Plugin installation uses MSI installer (for Non-IE) and CAB file (for IE) on Windows. Following steps are involved in generating the installers.

##### 5.1.2.5.1 *Steps on Linux.*

1. cd npapi-vlc/build/scripts
2. ./win32\_release\_package ../.././vlc-2.0.6 ../.././npapi-vlc (Release Build)
3. ./win32\_debug\_package ../.././vlc-2.0.6 ../.././npapi-vlc (Debug Build)

#### 5.1.2.5.2 Steps on Windows

##### System Setup

1. Install WixToolset v3.7  
TBD
2. Install Regspy2  
TBD
3. Install Cabarc  
TBD

##### Copy files from Linux

1. Copy TestReleaseWin32Release.zip to a folder in Windows
2. Unzip the above archive.
3. cd TestReleaseWin32Release/VlcInstallation
1. Above mentioned tools to generate MSI and CAB files are available here.
2. cd WindowsInstallerTools
3. Double click on each of .exe files to install on PC
4. cd win64bit (for 64 bit machine) OR cd win32bit (for 32 bit machine)

##### Generate MSI

###### Generate vlc.wxs

1. <Absolute Path of heat.exe> dir VLC -gg -sfrag -cg vlcFolder -o vlc.wxs.  
(Eg:"C:\Program Files\Wix Toolset v3.7\bin\heat.exe" <all the above params>)
2. Replace SourceDir to VLC and TARGETDIR to IPNC in vlc.wxs file.

##### Following steps to generate MSI file :

1. <Full Path of candle.exe> vlc.wxs axvlc.wxs main.wxs (compilation)  
(Eg:"C:\Program Files\Wix Toolset v3.7\bin\candle.exe" vlc.wxs axvlc.wxs main.wxs)
2. < Full Path of light.exe> vlc.wixobj axvlc.wixobj main.wixobj -o <user defined name with .msi extension>

(Eg:"C:\Program Files\Wix Toolset v3.7\bin\light.exe vlc.wixobj axvlc.wixobj main.wixobj -o win32Installer.msi)

##### Generate axvlc.wxs

*Following steps should be executed under VlcInstallation/VLC*

1. <Absolute Path of RegSpy.exe> axvlc.dll /RegServer > axvlc.reg
2. <Full Path of heat.exe> reg axvlc.reg -gg -sfrag -cg axvlc -o axvlc.wxs (Eg:"C:\Program Files\Wix Toolset v3.7\bin\heat.exe" <all the above params>)

**Generate CAB File**

1. Copy VIPNCWebdll.INF from source code folder (npapi-vlc/doc) to TestReleaseWin32Release/VlcInstallation folder on PC.
2. cd TestReleaseWin32Release/VlcInstallation
3. "Absolute path for CabArc.Exe" -r /m LZX:21 -p n VIPNCWebdlls.CAB <absolute path of all the files present under VLC folder> VIPNCWebdll.INF

**Sign CAB file**

Since the CAB file is not certified user has to do some extra steps to add the publisher as trusted publisher and enable to download unsigned ActiveX control. Follow the steps mentioned in the following [link](#).

## 5.2 User Interface

### 5.2.1 Linux

Steps to create the setup and build the code for UI :

- Create a folder (lets take "work") on home folder
- cd work
- Copy less.js library ( open source)
- mkdir src
- cd src
- Copy "src" folder (available in UI Source code) contents here
- make

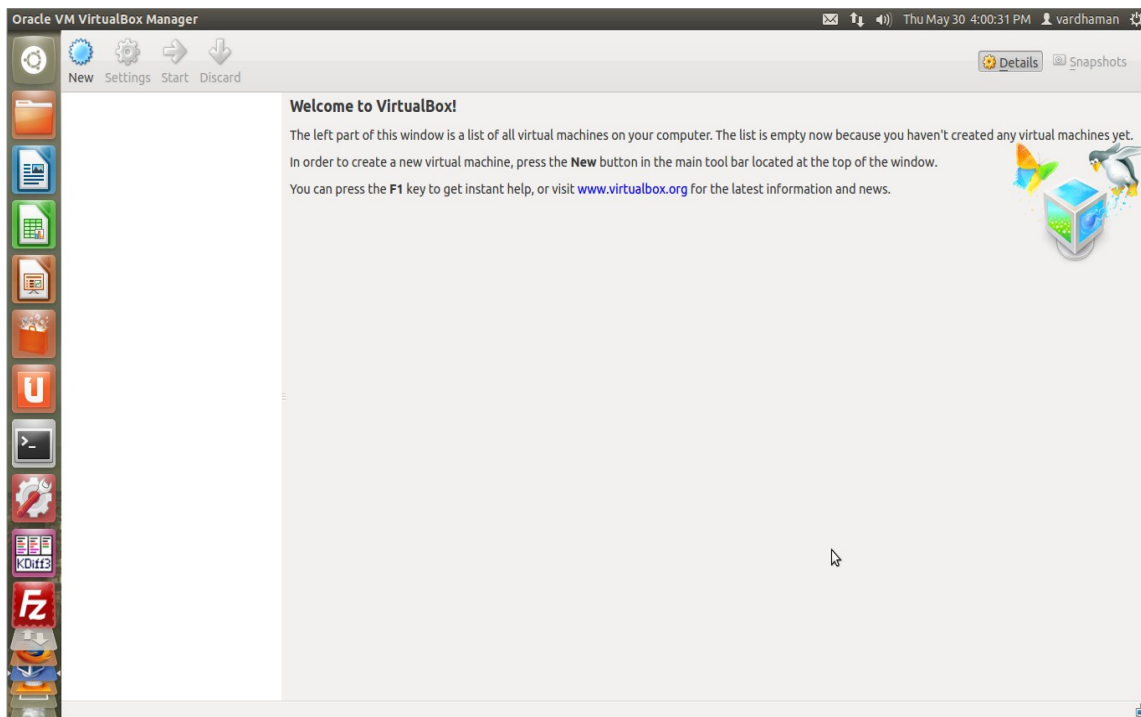
### 5.2.2 Creating Release package

- cp <cab file> ~/work (E.g: cp VIPNCWebdlls.CAB ~/work)
- cd ~/work/src/scripts
- ./makeRelease <targetName> ( E.g : ./makeRelease Ui )
  - \* <targetName> is a user created folder where all the packaged data will be copied.

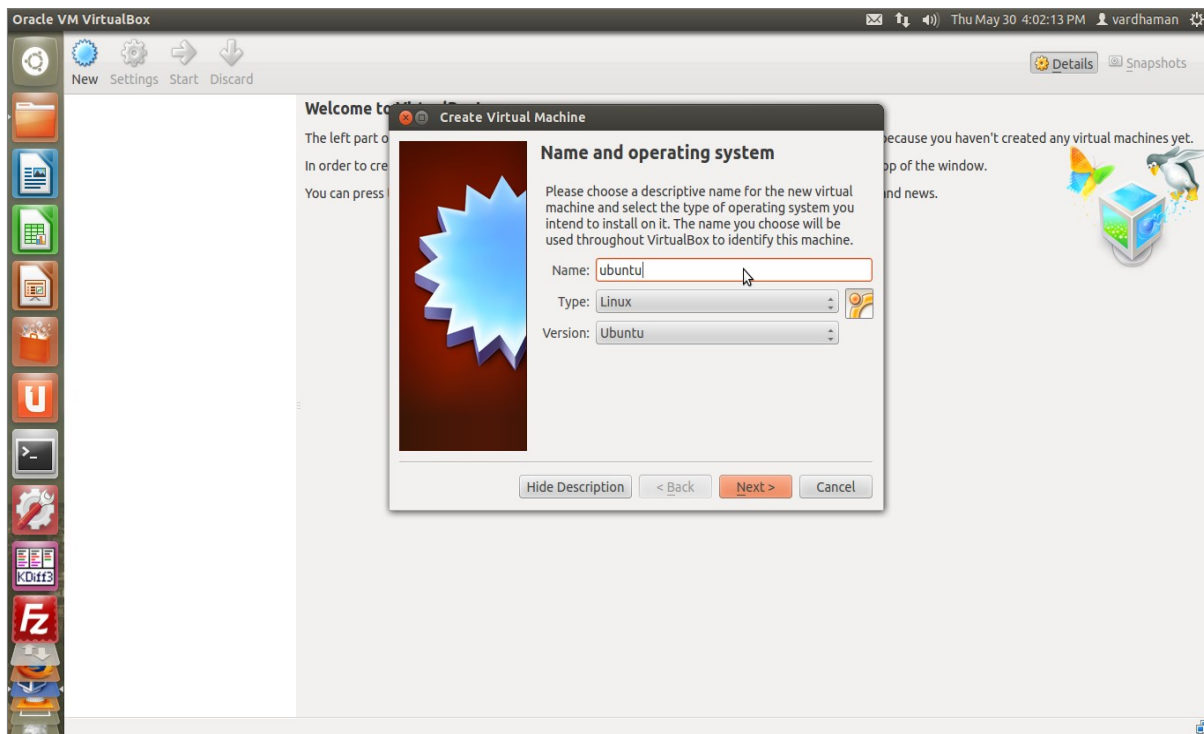
## 6 Virtual Box and Ubuntu image Installation

### 6.1 Procedure to install the virtual image

- Download the virtual box debian file from Internet for particular system configuration.
- Install this file as below  
**sudo dpkg -i <.deb file>**
- Launch the VirtualBox application
- whenever the application launches ,will get the below window.

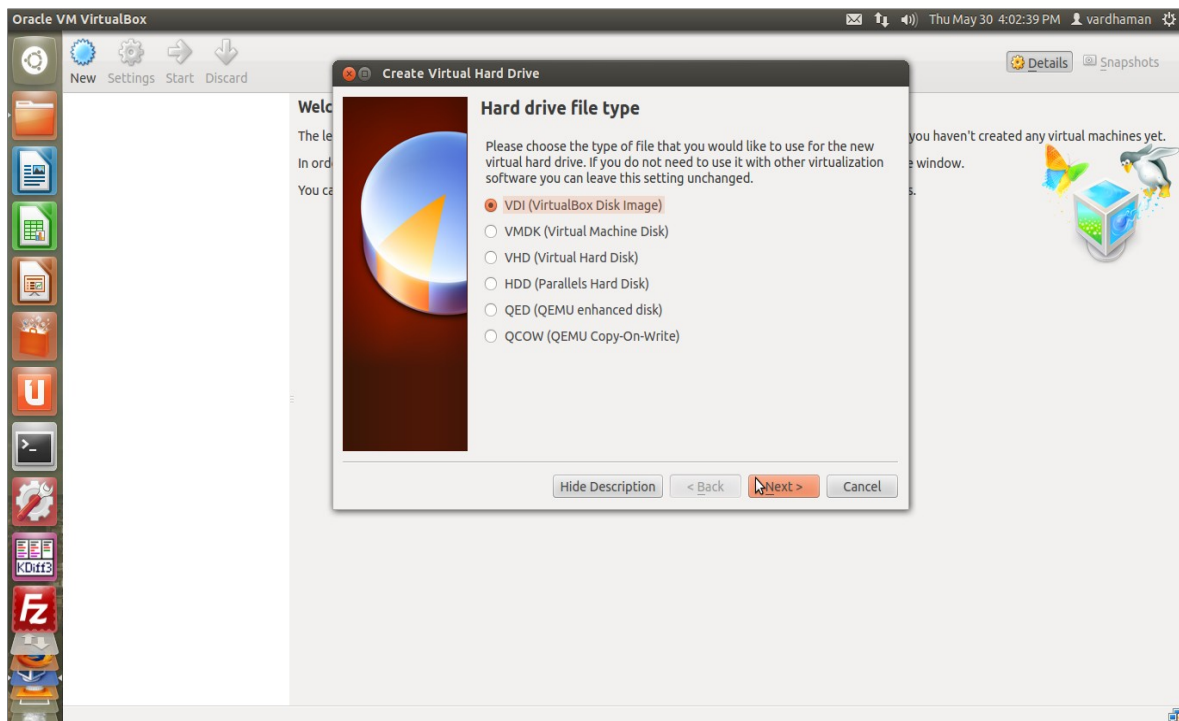
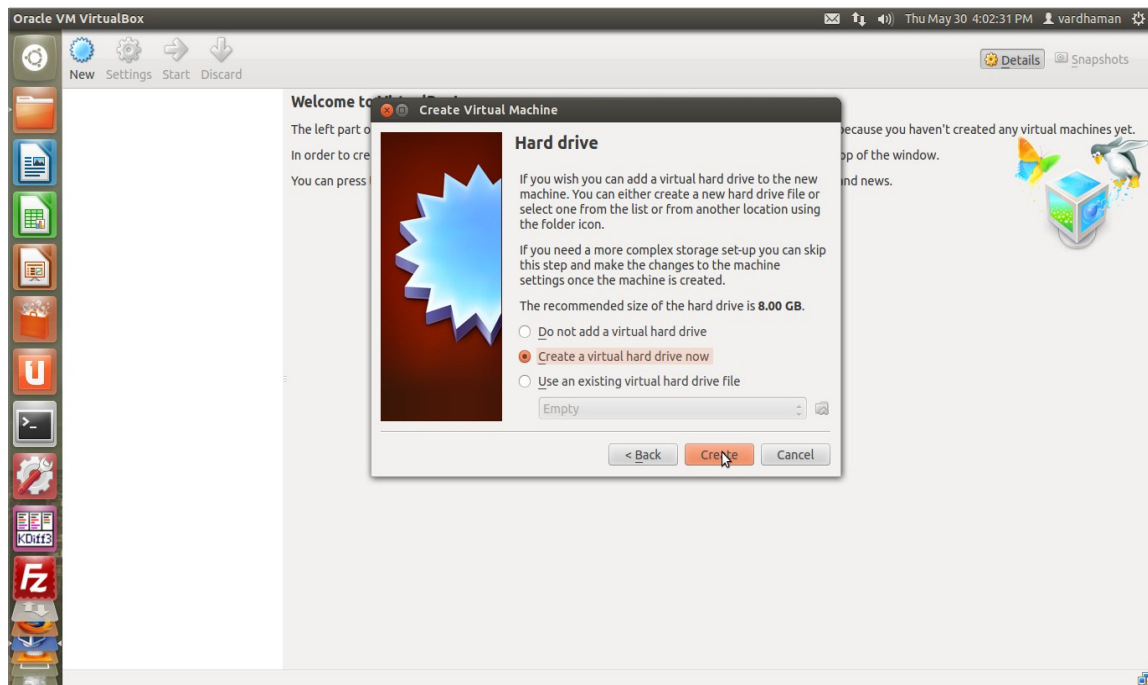


- Press new at the left corner to add virtual image of particular operating system.

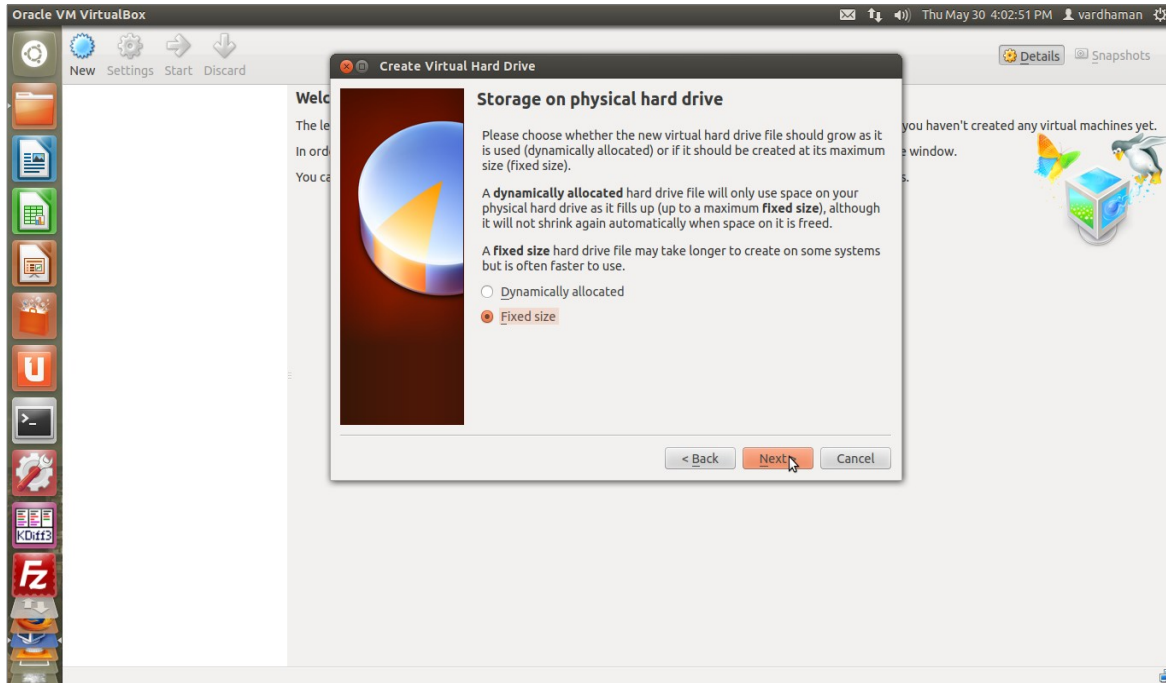


- Press next, Provide the memory size and press next.

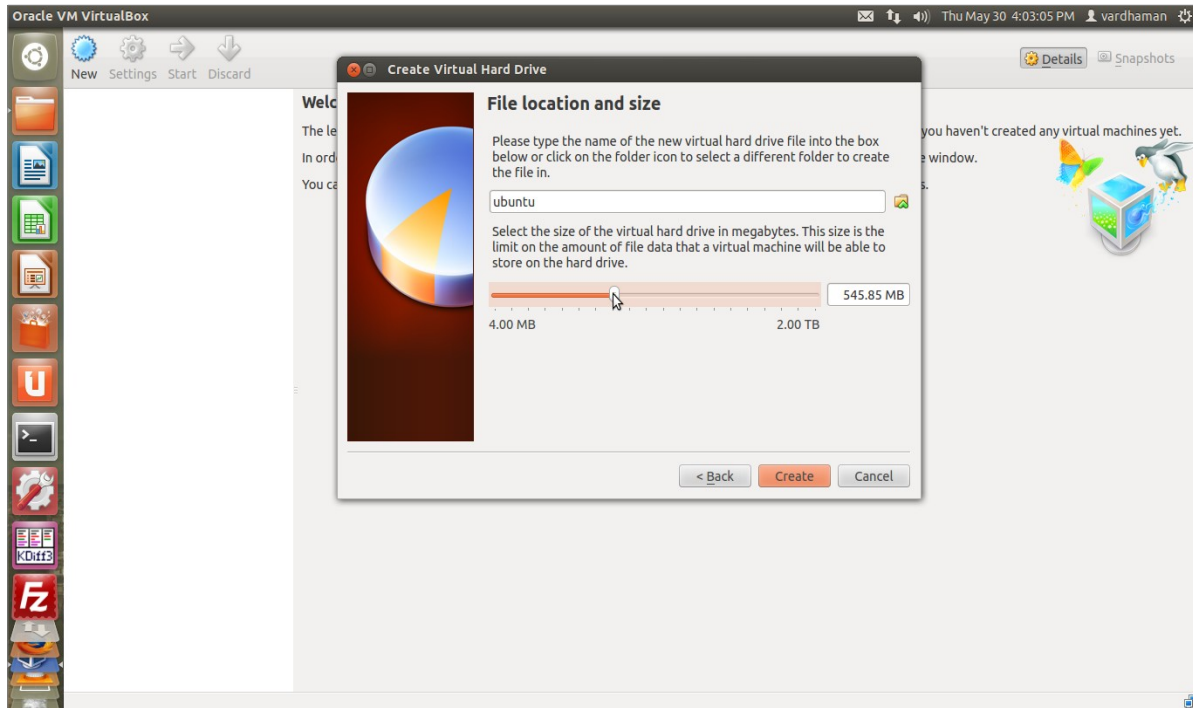
- Press Create → next as shown below.



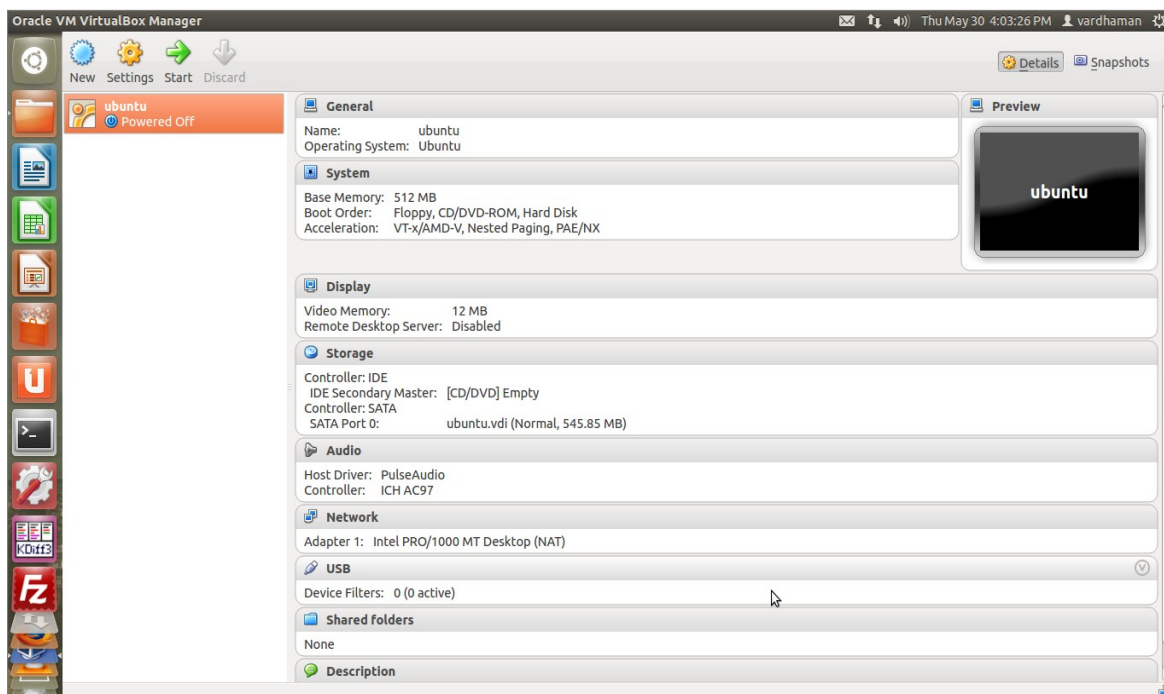
- You will get window asking for hard drive size option like fixed or dynamic, select on your choice and press next.



- Select the file location and memory size (minimum 8GB), press create.



- You will get new window and Ubuntu at the left corner.



- Press settings at the left corner. You will get new pop up window.

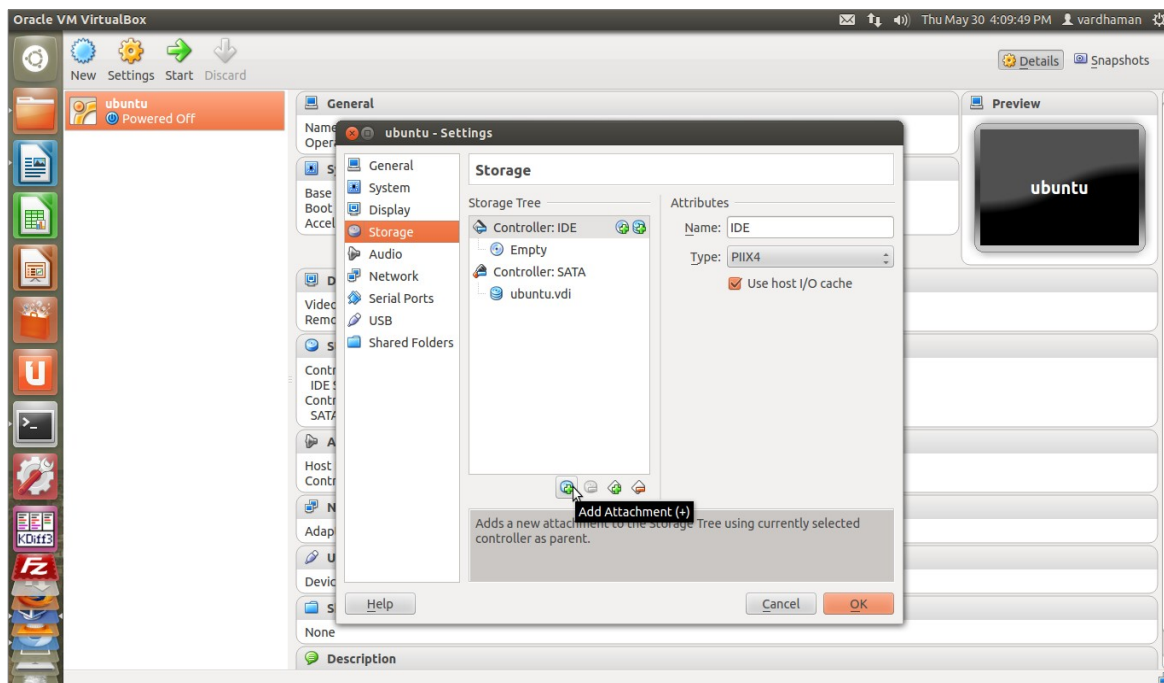
- If it asks for the user group settings , then need to add the user name to the vbox user group. It can be done as follows on host machine, then open terminal

```
vim /etc/group
```

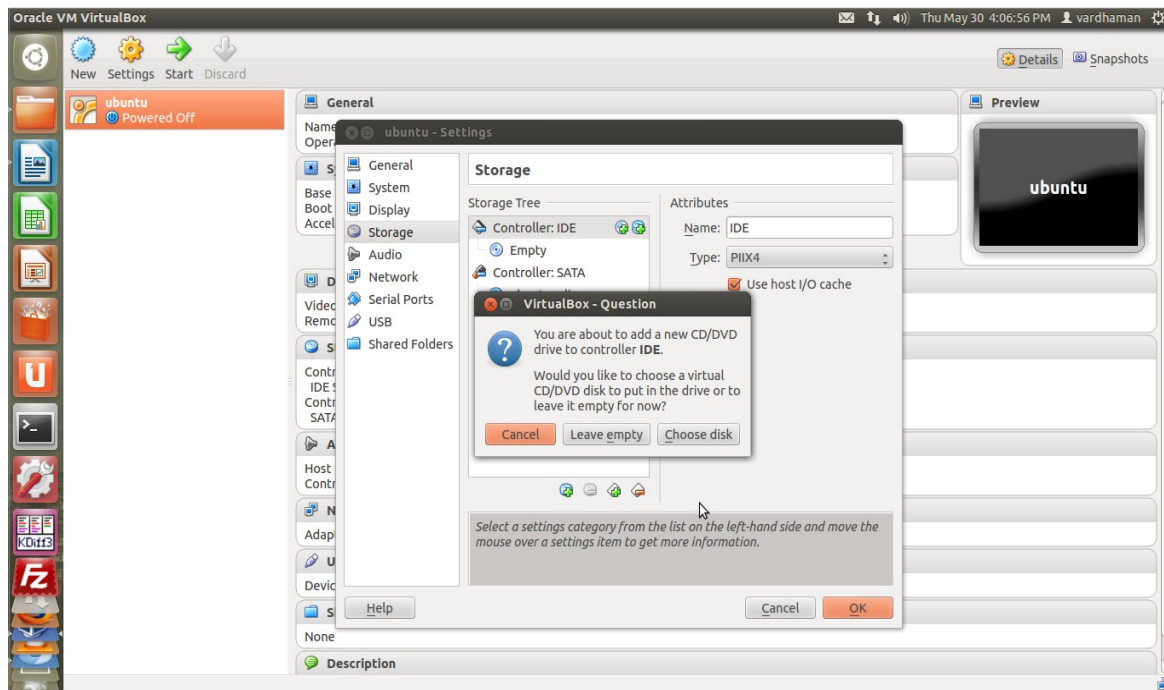
add the user name of the host machine to vboxusers.

Ex: vboxusers:x:126:<username>

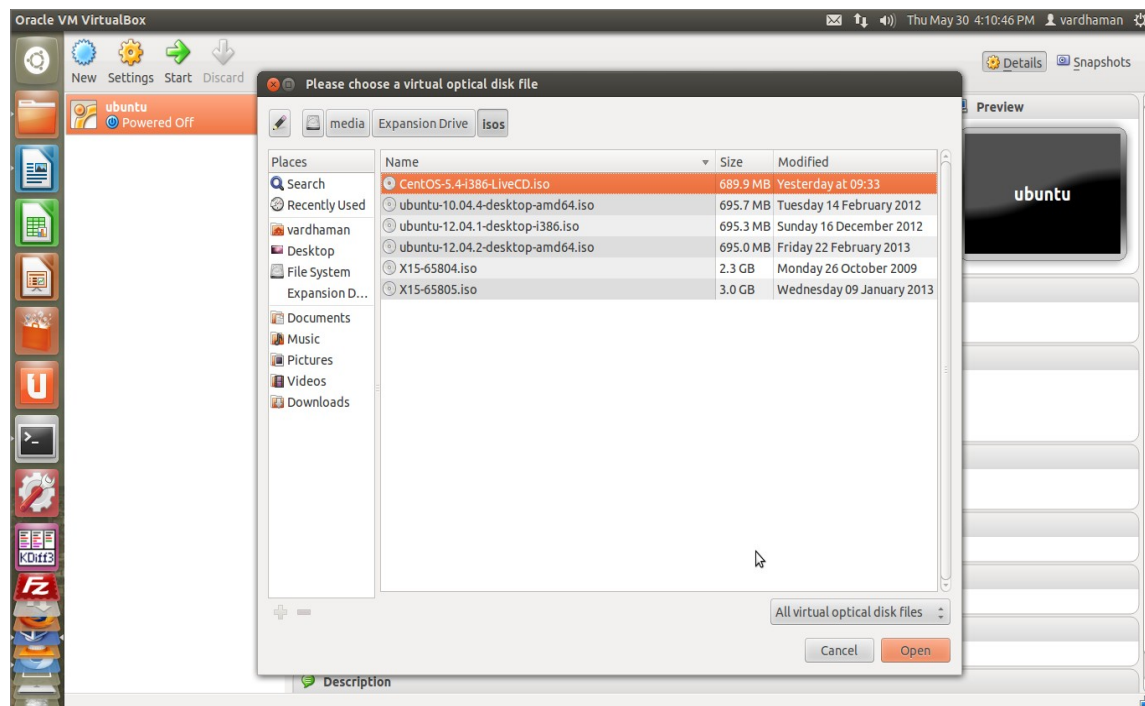
- Press “settings”, select “storage” at the left side list.
- Select Icon of add attachment , Select the either of them wherever “iso” file is present (DVD or Hard disk) (If you are using “iso” to install).



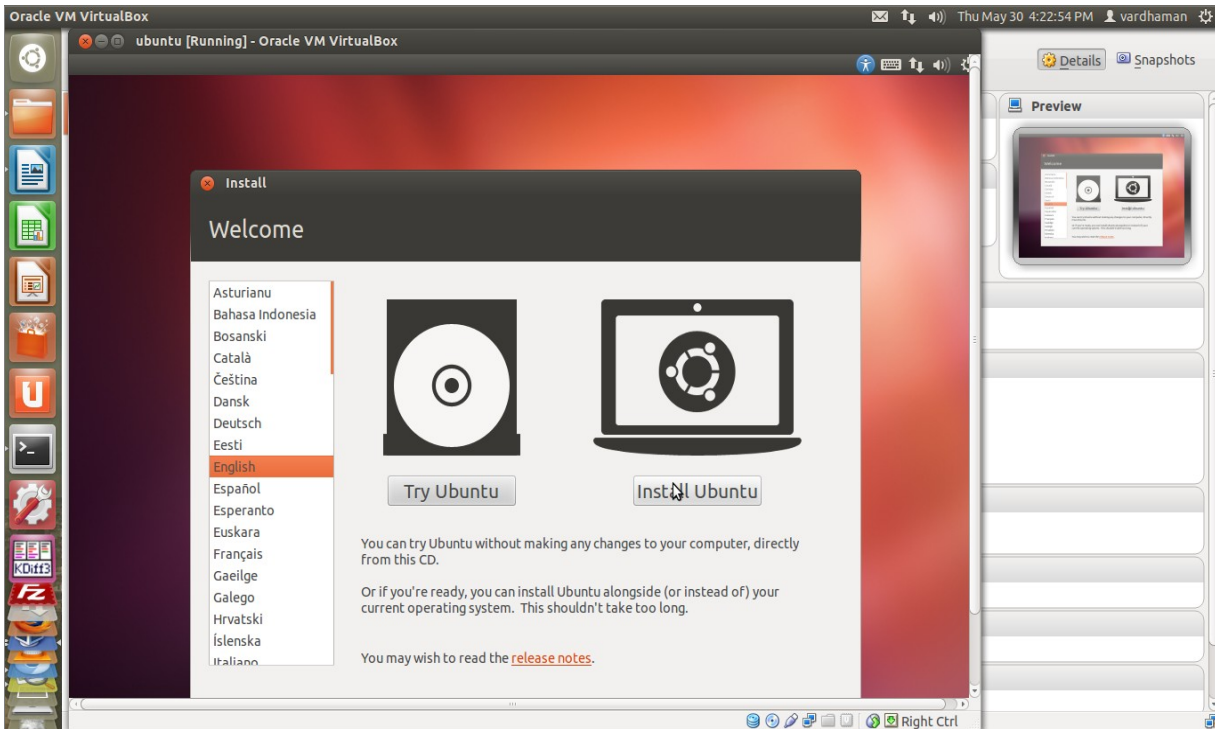
- You will get pop up , select “choose disk”.



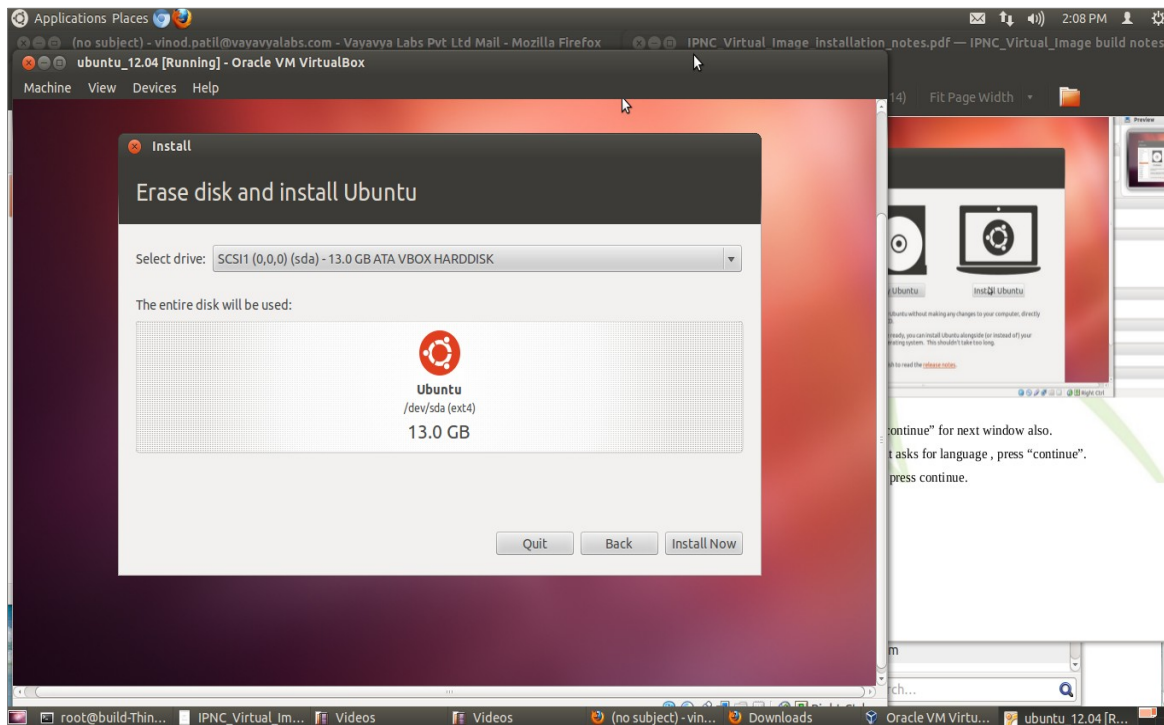
- Select the matching “iso” file with the system and click “open”.



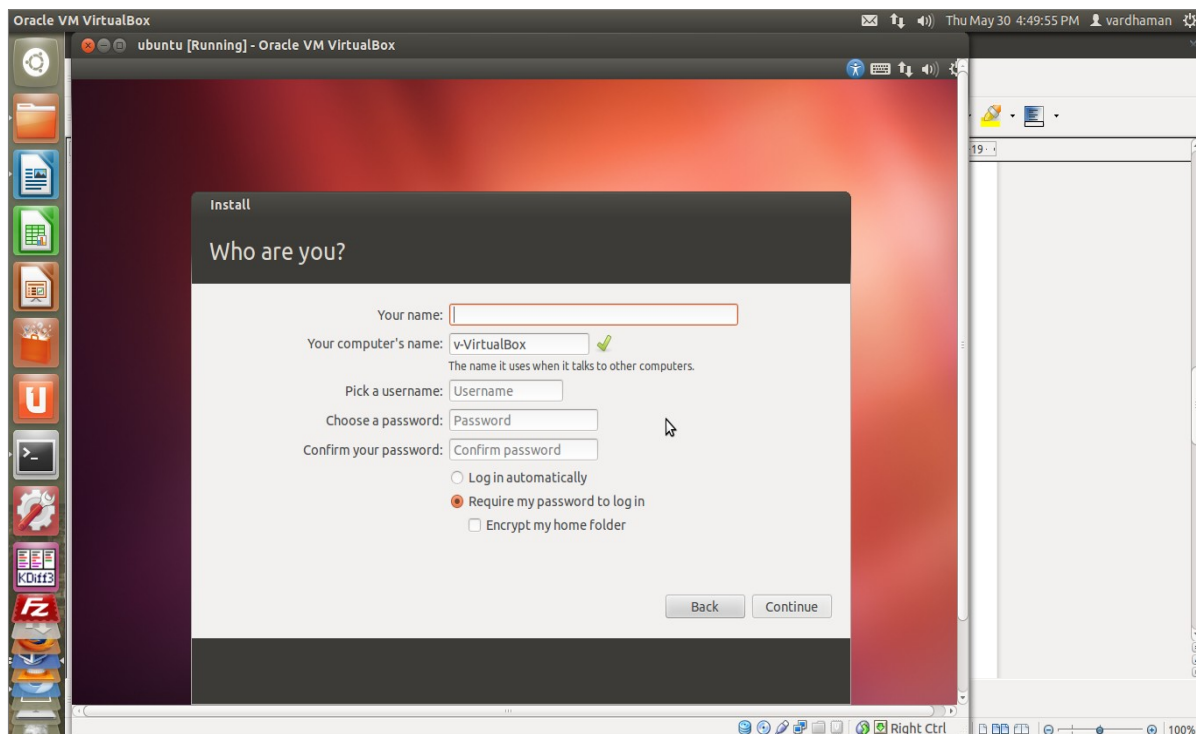
- Click “start” button on the upper left side besides settings button.
- It will search for bootable device, gives pop up , select “Install ubuntu”.



- Press “continue” , press “continue” for next window also.
- Select continue,after that it asks for language , press “continue”.
- Asks for install , press “Install Now”.



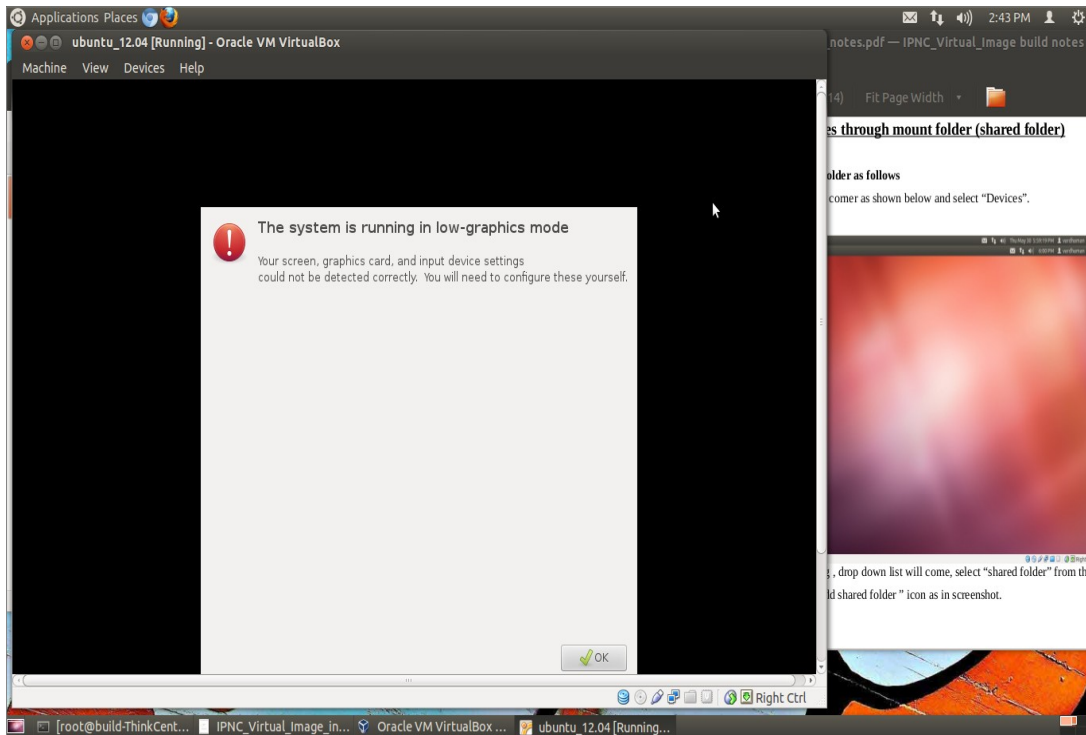
- Give the login details and press continue.



It will start installing the ubuntu( which ever OS selected).

Once installation done it asks for restarting. Just press restart.

After rebooting, if it throws any errors as below screenshot

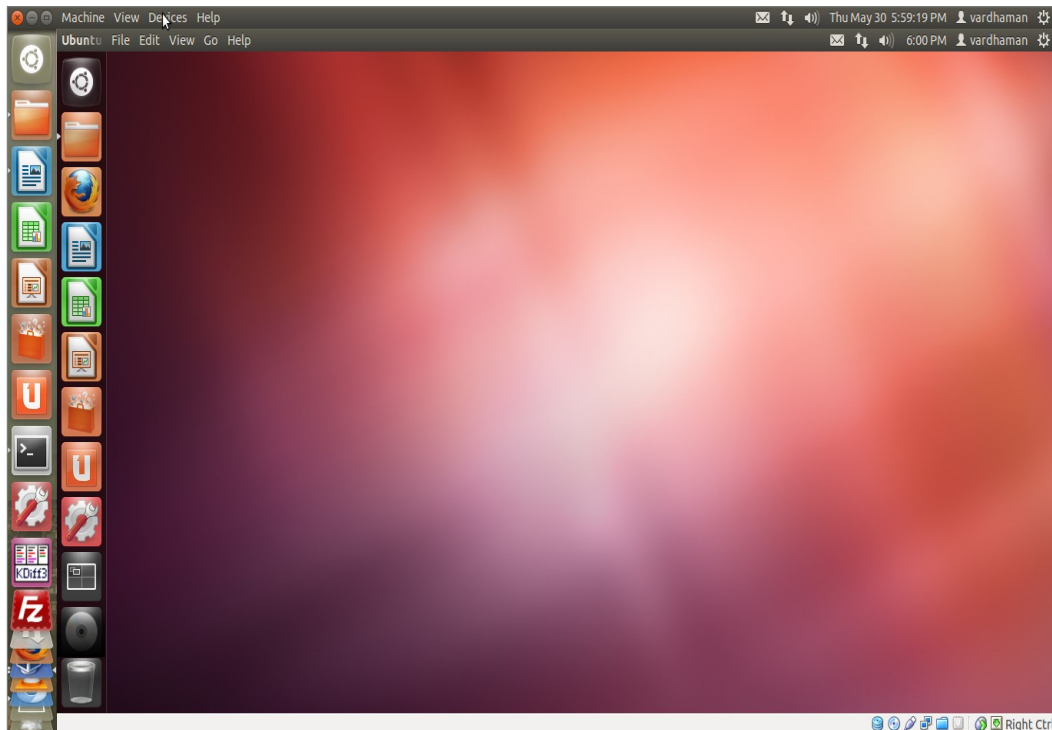


- Just press “Ok”, Next one more pop up windows comes asking four options.
- Select last option “exit to console login”.
- Enter the username and password
- `$sudo apt-get install gdm.`
- After installation it asks for return, just press enter key. It will reboot automatically, if it does not then reboot it manually.
- Once its done, he will ask again as above screenshot, just press “Ok” and “Ok”.

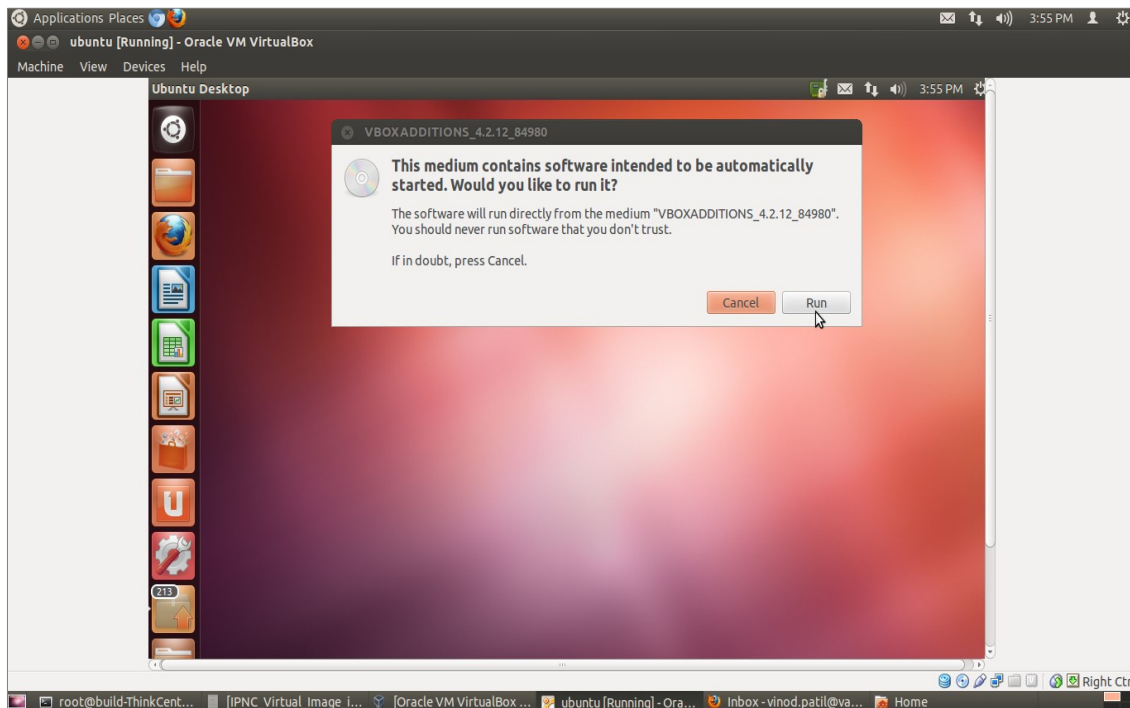
## 7 Building the ipnc packages through mount folder (shared folder)

### 7.1 Create the shared folder as follows

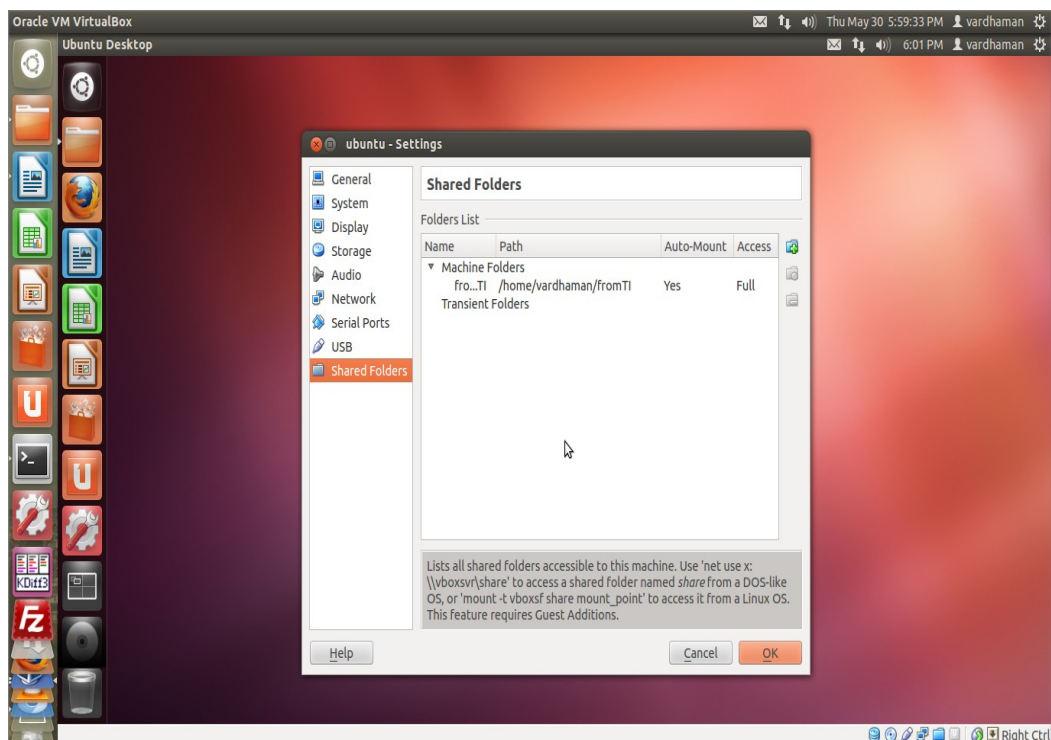
- Go to the left corner as shown below and select “Devices”.



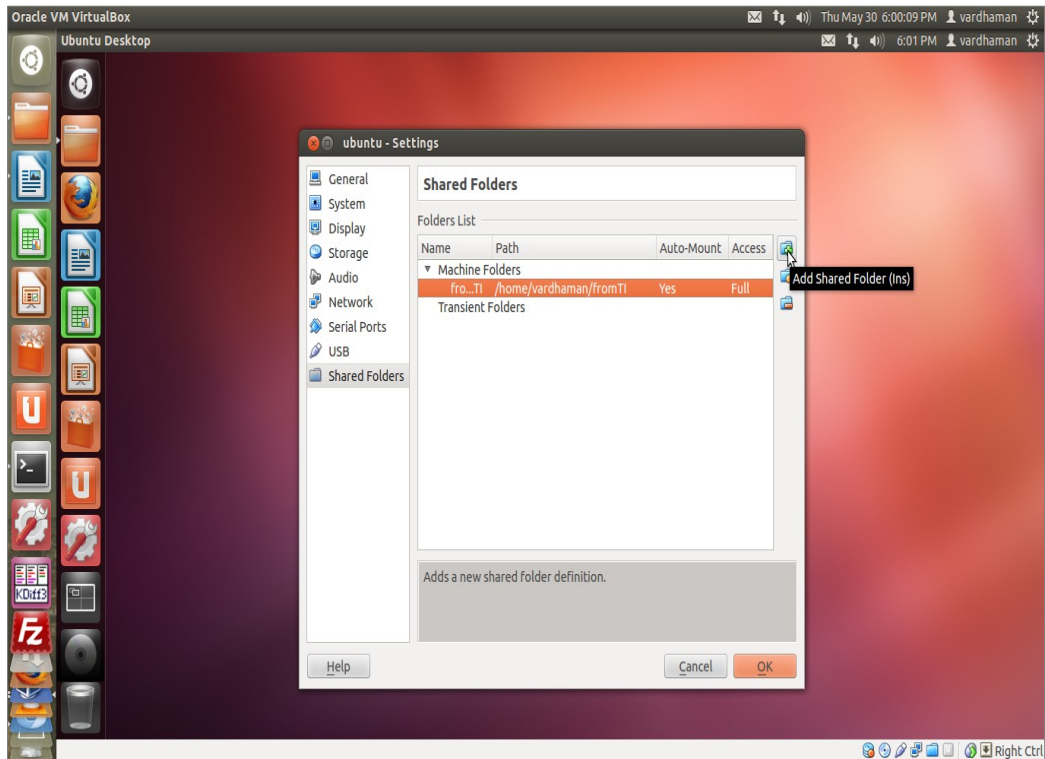
- After selecting , drop down list will come, select “shared folder” from this list.
- If it asks for “Guest addition” settings , then click “Devices” , Select last option “Install Guest additions”.
- Click on “Run”. Once its gets completed it asks to press return, just press enter key.



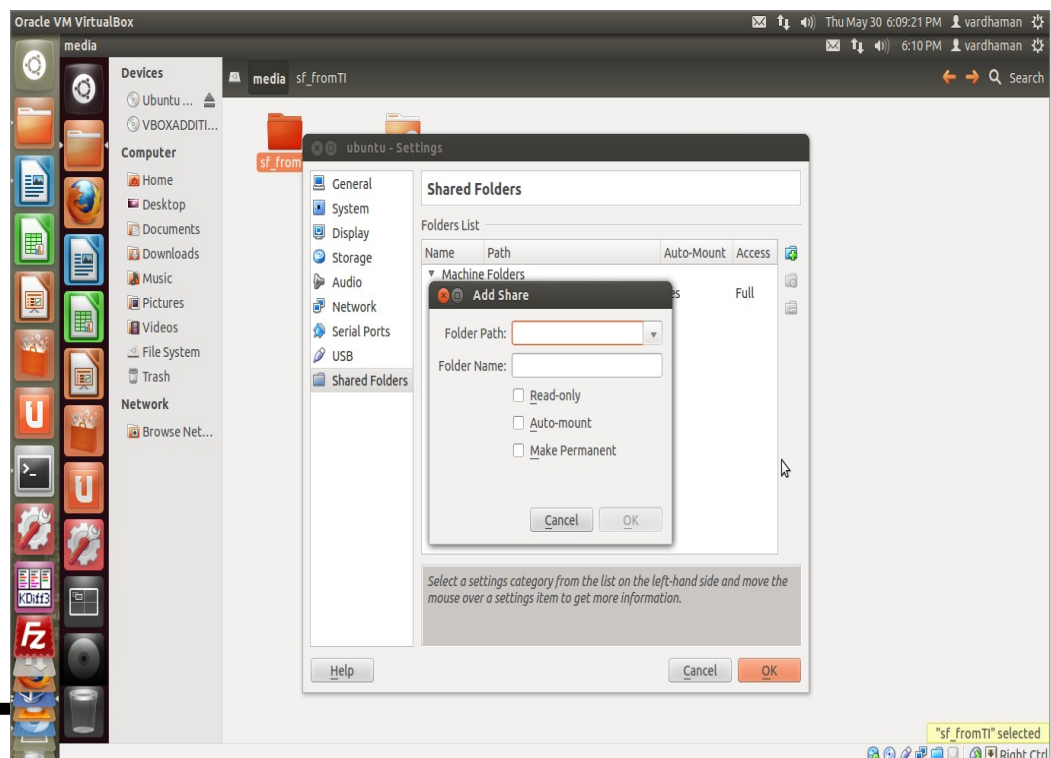
➤ Now select “Devices” → “shared folders” .



- Select the “Add shared folder ” icon as in screen shot.



- In the pop up window, select the folder which need to be shared ( ipnc source codes), give the other details (Auto mount, Permanent etc). And press “Ok”.



After pressing “ok”, restart the virtual box machine .After rebooting the shared folder will be present in the “media” folder. The IPNC packages can be built through this shared folder.

**NOTE:** For building the IPNC packages refer the build document provided with the release.

## 8 Integrating the application with the firmware

Following steps to view integrated browser with this application :

1. Copy “Ui” folder's content available in release folder to /var/www on RFS.  
(E.g : After untarring release folder → cd drop2ti → cd UI → cd Ui - > copy all content to /var/www).

P.S : Please verify the sync path of Ui folder with TI's released package

2. Type “http://<Camera Ip>/index.html” to enjoy the app.

(E.g: <http://192.168.3.142/index.html>) )