Unit XXIX Assignment II

By Nathan Windisch

PIII - Plan an installation and an upgrade

Introduction & Planning

The software that we are going to be installing will be *Everything*. *Everything* is an ultrafast item indexer that searches through any and all folders that you choose. It is, in my opinion, far superior to the Windows search, which is very slow even on an SSD. A non-administrative version of *Everything* on all systems within UTC Reading, allowing users to access their files faster and easier.

Testing

The one issue with *Everything* is that is very powerful. Permissions will need to be set up correctly so that no damage is made to the system, such as editing registry files or even other people's files. Lots of testing will need to be done before the final patch will be rolled out to all systems, including seeing if certain files can be edited, and if the program works with different environments with certain programs and features enabled and disabled.

Delivery

Everything shall be delivered to the user's systems over the course of several weeks, starting with the IT team, then the Senior Leadership Team, then the heads of department, then the teachers and finally the students. This process means that if there are any issues that are encountered during the delivery period, less people will be affected due to the fact that it is an incremental amount of people over a long time, meaning that there is less damage an issue is found within the first few groups of people that it is rolled out to.

Shipping

Everything will be shipped out a few days after it is installed. This is to ensure that all the systems that have the software delivered, work at 100% efficiency and at an optimum speed for a workplace environment. While it is easy to think of Delivery and Shipping as the same thing, it is important to realize that they are different things. Delivery comes first and is used to install the software onto the system, and Shipping comes second and is used to distribute the software amongst the users.

Storage

The storage systems that will be used will be the internal Hard Disk Drives, or HDDs, within the user's computers. *Everything* is quite a small program, as it only organizes and displays files. This means that small amounts of data will be created, such as the user's preferences and files that they want to view on default. This means that the IT team at UTC Reading will need to check the HDDs, and ensure that a few megabytes of storage are available, at least. This is because the program will either not be able to install properly, or will not be able to work correctly after installation due to a lack of storage. A fresh Windows 8 install with about 80,000s files will take approximately 6MB of RAM and less than 3MB of Disk Space. For reference, 1,000,000 files will use about 50MB of RAM and 15MB of Disk Space.

Software Specifications

The software specifications that will be required is Windows XP, Windows Vista, Windows 7, Windows 8 or Windows 10. No other operating systems will be able to run *Everything*. *Everything* also needs to be run as an Administrator or the *Everything* service, in order to access NTFS indexing. Also, the Central Processing Unit will need to be fast enough to perform the calculations, but most CPUs have the power to do that now. Along with the CPU, there will need to be enough Random Access Memory to perform the searches but, again, most computers have enough RAM to do that. The final thing that is needed is the correct type of architecture to run the program. Luckily, *Everything* has both x86 and x64 installs, meaning that it will work with all versions of Windows that I have previously mentioned.

Communication

Communication will be required when rolling out this update, as all participants who are involved will need to be aware of the changes that are occurring. This means that emails will need to be sent out periodically in order to keep everyone up to speed. Also, the Senior Leadership Team will need to be given a biweekly update on the progress that the IT team has been making, including a list of all users and machines that *Everything* has been installed to. Also VoidTool, the developers of the product, will need to be contacted if there are any issues or bugs so that they can both give advice and fix the bugs for future versions of the software.

Logs

Logs will also need to be stored as a way of viewing what went wrong if there are any issues, and also as a method of storing data about which machines and users have installed *Everything*, meaning that it can be used as a tracking device for which users have got the program installed.

Security

Security is vital to UTC Reading and this means that *Everything* will need to be locked down to prevent access to files that users cannot normally access. As a result of this, the logs for the program will need to be carefully monitored and any malicious activity using *Everything* will need to be reported to the Senior Leadership Team. Also, the program's source code will need to be scanned to ensure that there are no viruses or malicious code within it. Luckily, *Everything* is open source, meaning that the code will be easy to view and will require no decompiling.

Confidentiality

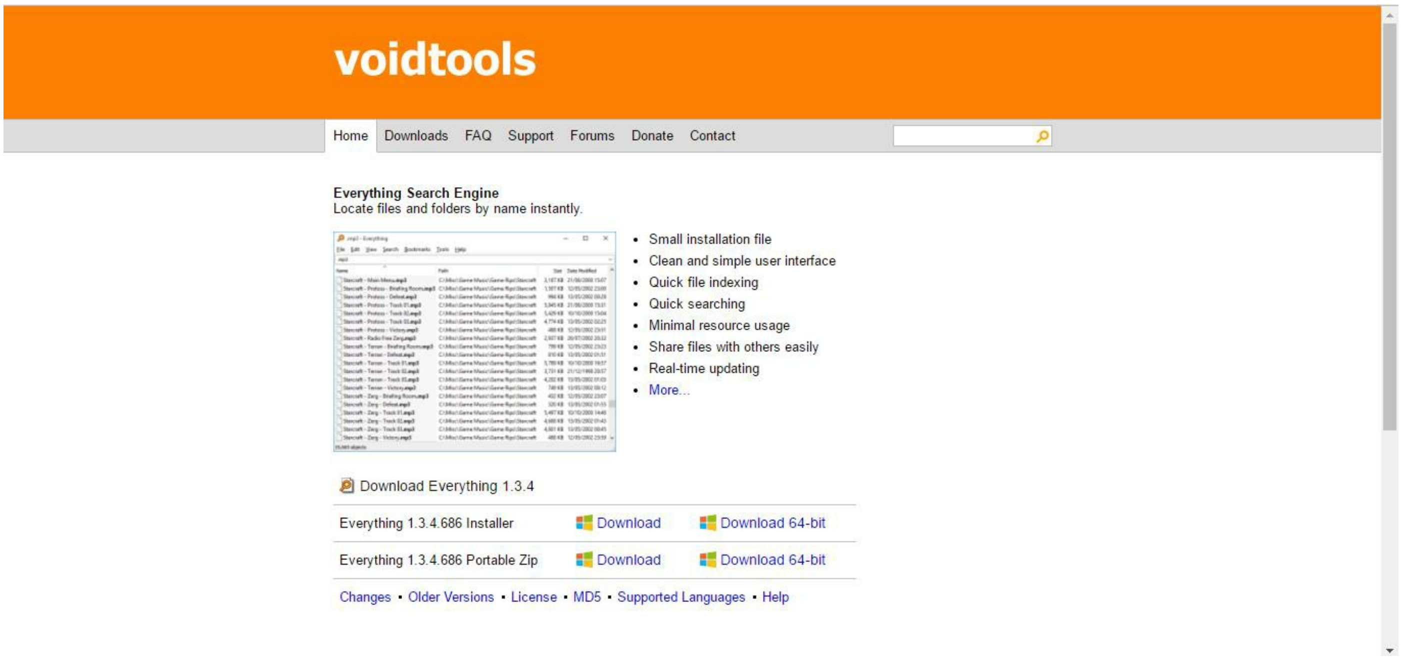
While logs need to be saved to ensure that the software is being used correctly, certain privacy rules must be followed. For instance, login details should only be used by those authorized to use them. Also, system administrators and users should only use *Everything* for accessing their own data. Every user's data is their own, and should only be accessed when certain conditions are met, such as the data being used in a criminal investigation. Also, any remote connections to the network must be sent over a secure connection using SSH or SSL.

Contractual Requirements

Contractual requirements are important to installing any software and *Everything* is no exception. While *Everything* is open source, any changes made to the code for either optimization or security purposes cannot be sold to a third party. All code edits must show that it was based off the *Everything* software, and cannot be sold for profit without the consent of the original author. Also, any Terms of Service contracts that the users of the UTC Reading network sign may need to be updated in order to align with the usage of the new software. For instance, the agreement must state that access of other user's data is prohibited.

PIV - Record and Complete a Software Installation

The software was installed over a few weeks on a variety of different machines. The following report is based on my installation on our test machine running Windows 10 (x64). *Everything* was a rather quick install, taking about {X} minutes to install.



voidtools

Home | Downloads | FAQ | Support | Forums | Donate | Contact

Everything Search Engine
Locate files and folders by name instantly.

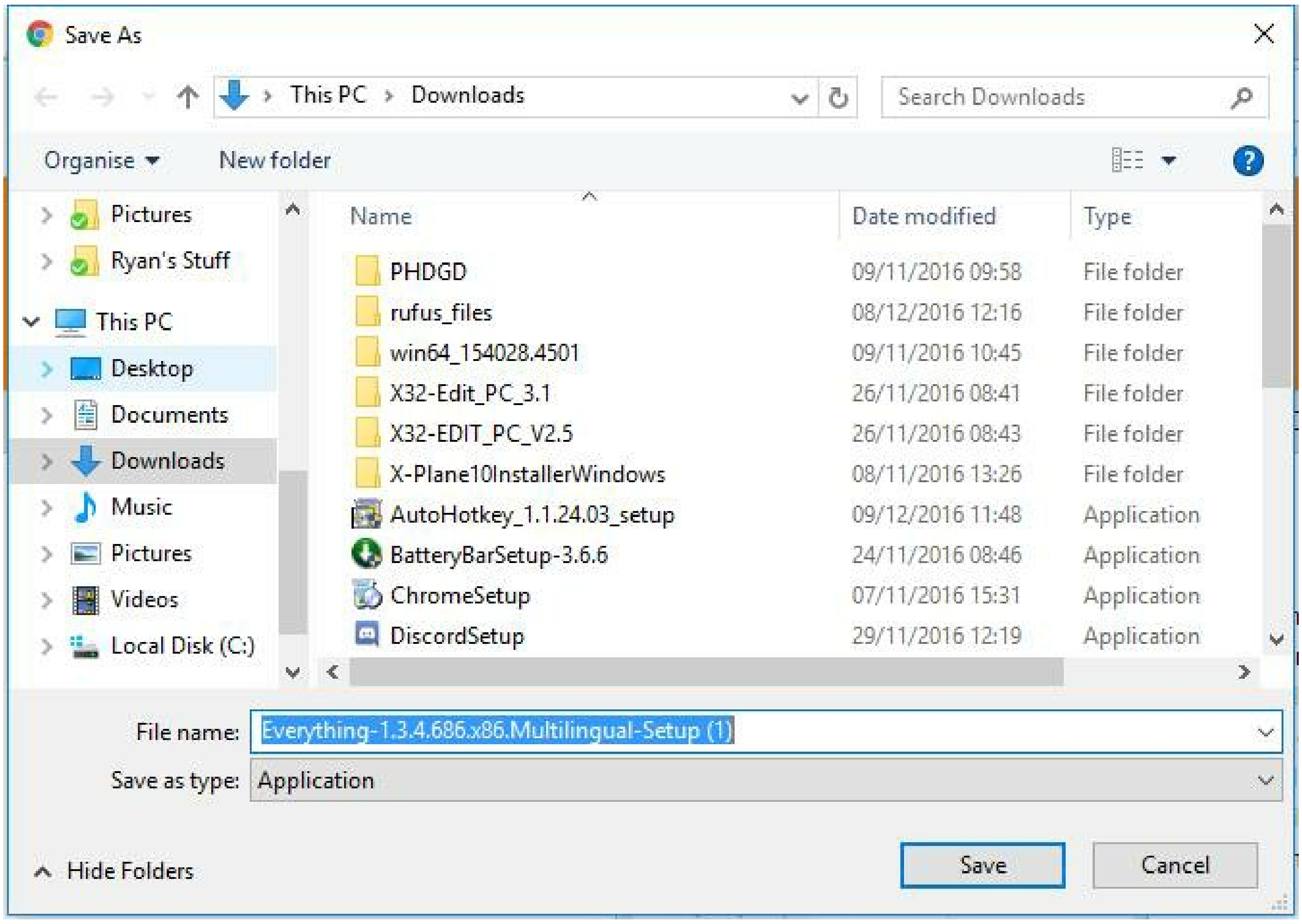
- Small installation file
- Clean and simple user interface
- Quick file indexing
- Quick searching
- Minimal resource usage
- Share files with others easily
- Real-time updating
- More...

Download Everything 1.3.4

Everything 1.3.4.686 Installer	Download	Download 64-bit
Everything 1.3.4.686 Portable Zip	Download	Download 64-bit

Changes • Older Versions • License • MD5 • Supported Languages • Help

The following image is of the website, http://voidtools.com. This is the official website of *Everything*, and contains downloads, development versions, FAQs, documentation and source code.



Save As

This PC › Downloads

Search Downloads

Organise ▾ New folder

Pictures
Ryan's Stuff
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)

Name	Date modified	Type
PHDGD	09/11/2016 09:58	File folder
rufus_files	08/12/2016 12:16	File folder
win64_154028.4501	09/11/2016 10:45	File folder
X32-Edit_PC_3.1	26/11/2016 08:41	File folder
X32-EDIT_PC_V2.5	26/11/2016 08:43	File folder
X-Plane10InstallerWindows	08/11/2016 13:26	File folder
AutoHotkey_1.1.24.03_setup	09/12/2016 11:48	Application
BatteryBarSetup-3.6.6	24/11/2016 08:46	Application
ChromeSetup	07/11/2016 15:31	Application
DiscordSetup	29/11/2016 12:19	Application

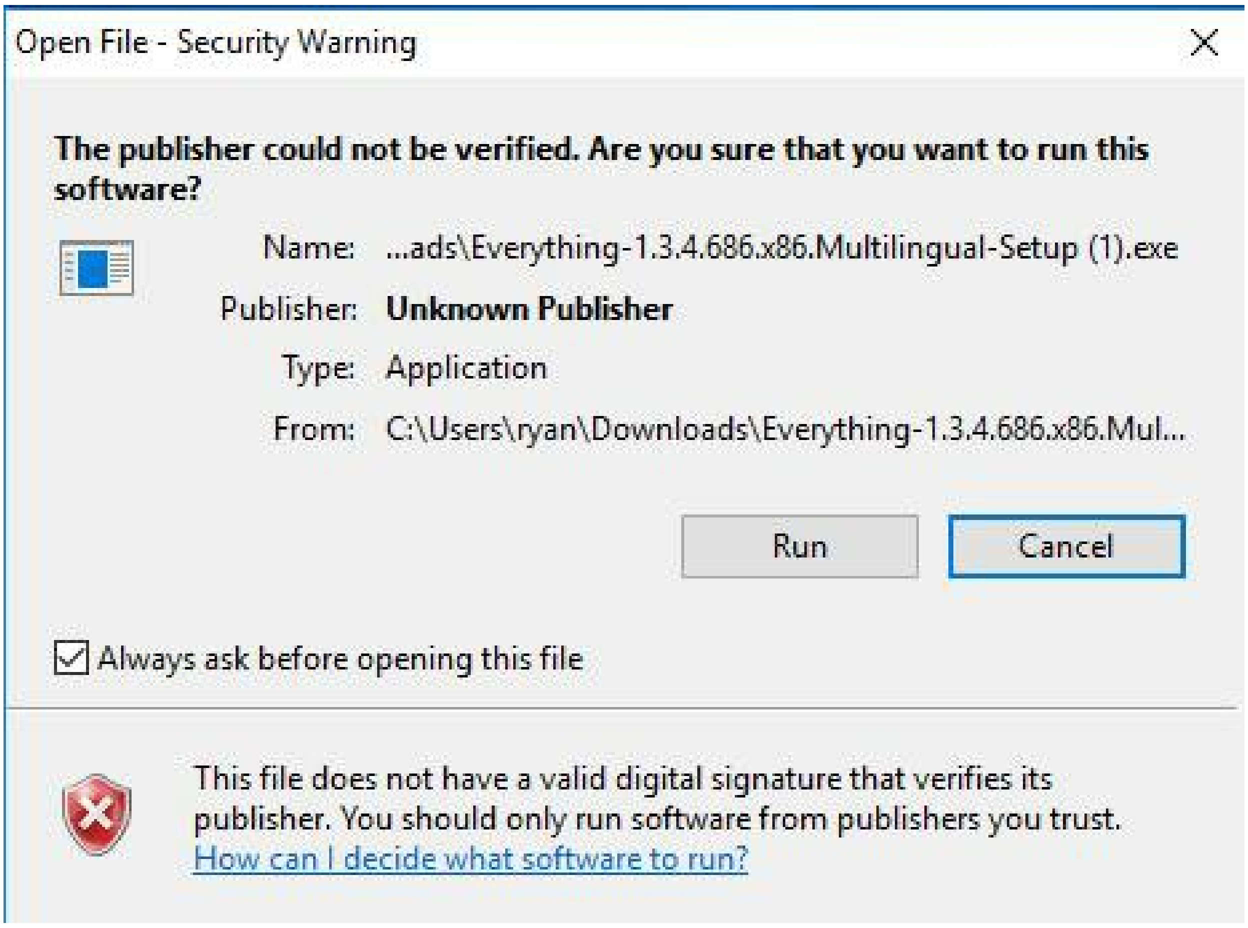
File name: Everything-1.3.4.686.x86.Multilingual-Setup (1)

Save as type: Application

Hide Folders

Save Cancel

This is an image of me saving the .exe file. This is the program that is required to run in order to install *Everything*.


Open File - Security Warning

The publisher could not be verified. Are you sure that you want to run this software?

Name: ...ads\Everything-1.3.4.686.x86.Multilingual-Setup (1).exe

Publisher: Unknown Publisher

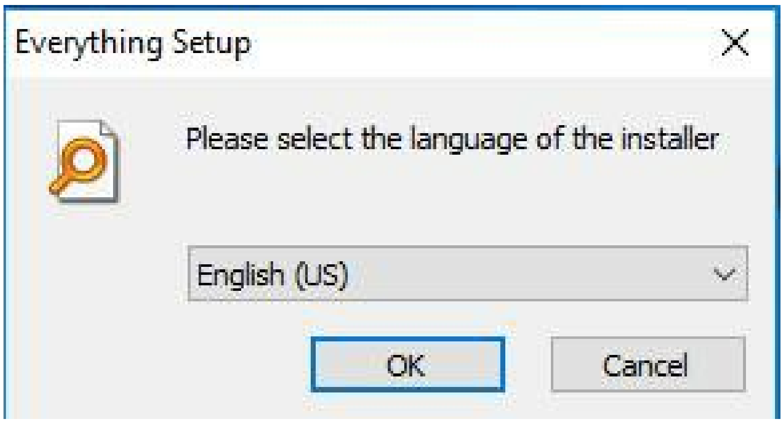
Type: Application

From: C:\Users\ryan\Downloads\Everything-1.3.4.686.x86.Mul...

☑ Always ask before opening this file

This file does not have a valid digital signature that verifies its publisher. You should only run software from publishers you trust.
How can I decide what software to run?

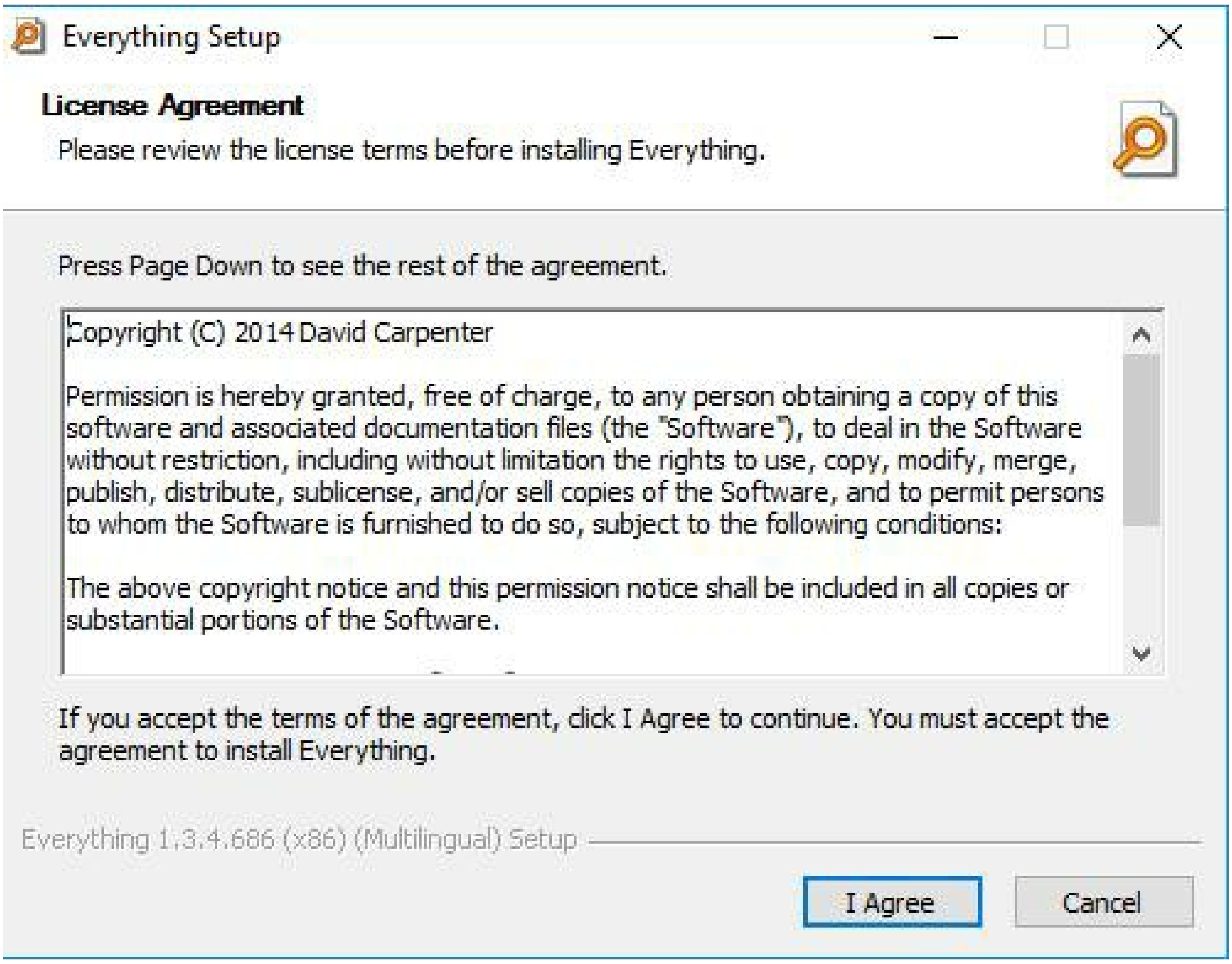
The installation required Administrative rights, meaning that I had to log in as an Administrator account.


Everything Setup

Please select the language of the installer

English (US)

In this step I had to choose the language that the software was going to use. Given that I am fluent in English, as are all the students and staff at UTC Reading, I decided to choose English as the language used.



License Agreement

Please review the license terms before installing Everything.

Press Page Down to see the rest of the agreement.

Copyright (C) 2014 David Carpenter

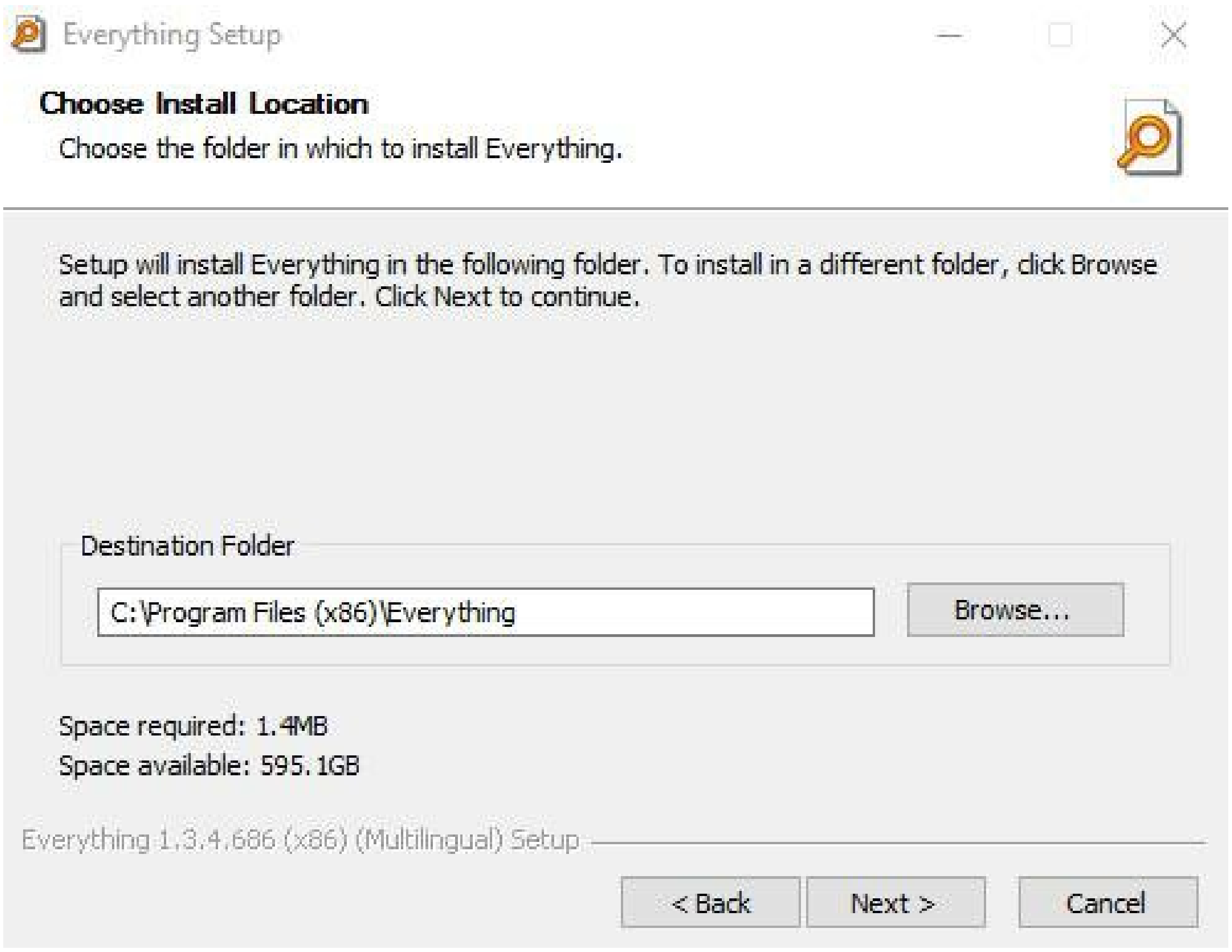
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Everything.

Everything 1.3.4.686 (x86) (Multilingual) Setup

This is the license agreement that must be accepted before the software can be installed. If this license is broken the UTC Reading can be liable for a law suit, costing both time and money.


Everything Setup

Choose Install Location

Choose the folder in which to install Everything.

Setup will install Everything in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

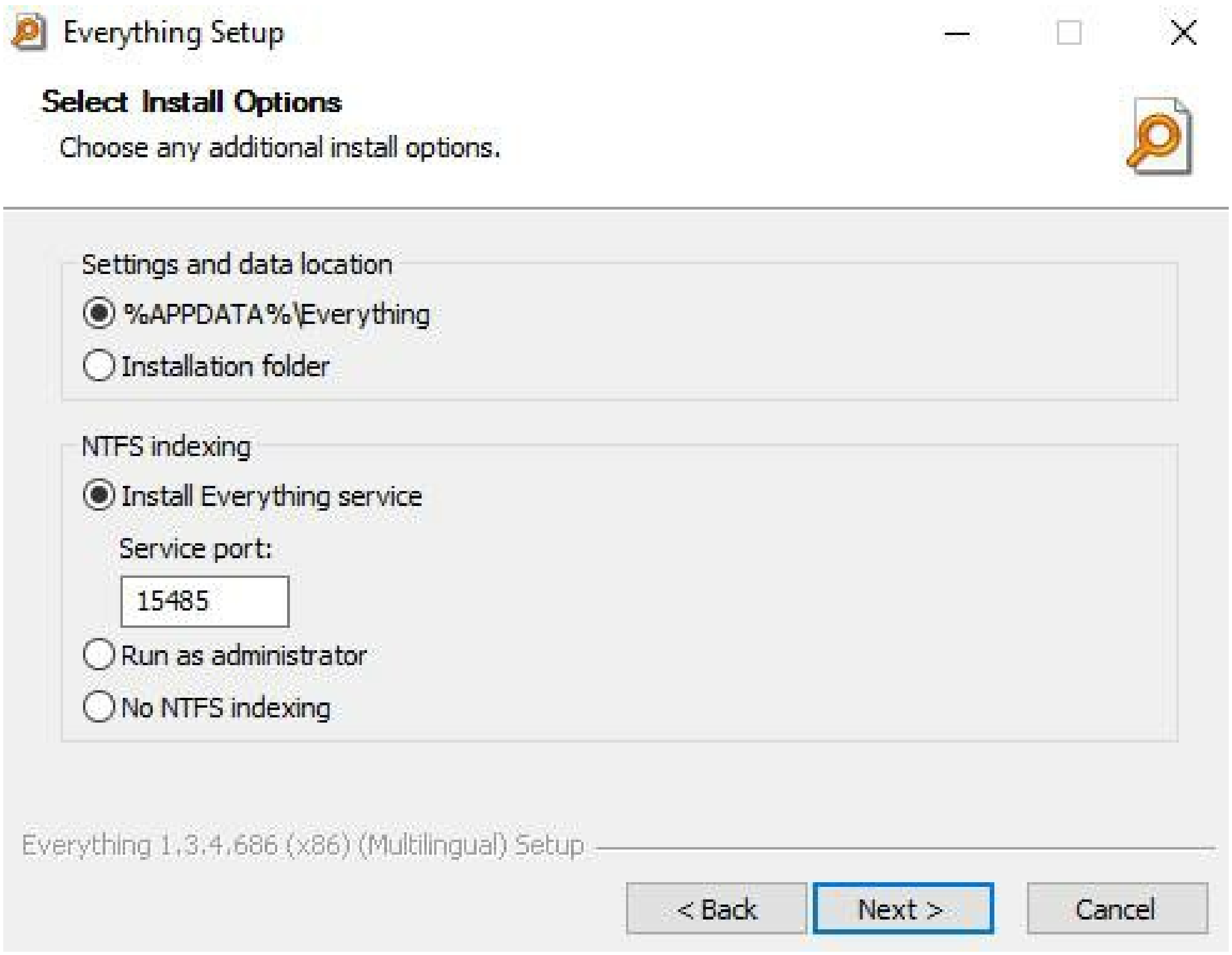
Destination Folder

C:\Program Files (x86)\Everything

Space required: 1.4MB
Space available: 595.1GB

Everything 1.3.4.686 (x86) (Multilingual) Setup

In this step I had to choose what location I would save the program’s files to. I chose the default location as it was the recommended place to store it. It is important to note that I had to pick a space that had enough storage space but it only takes up 1.4MB and I had 595.1GB of space free, which was plenty.


Everything Setup

Select Install Options

Choose any additional install options.

Settings and data location

(●) %APPDATA%\Everything

( ) Installation folder

NTFS indexing

(●) Install Everything service

Service port:

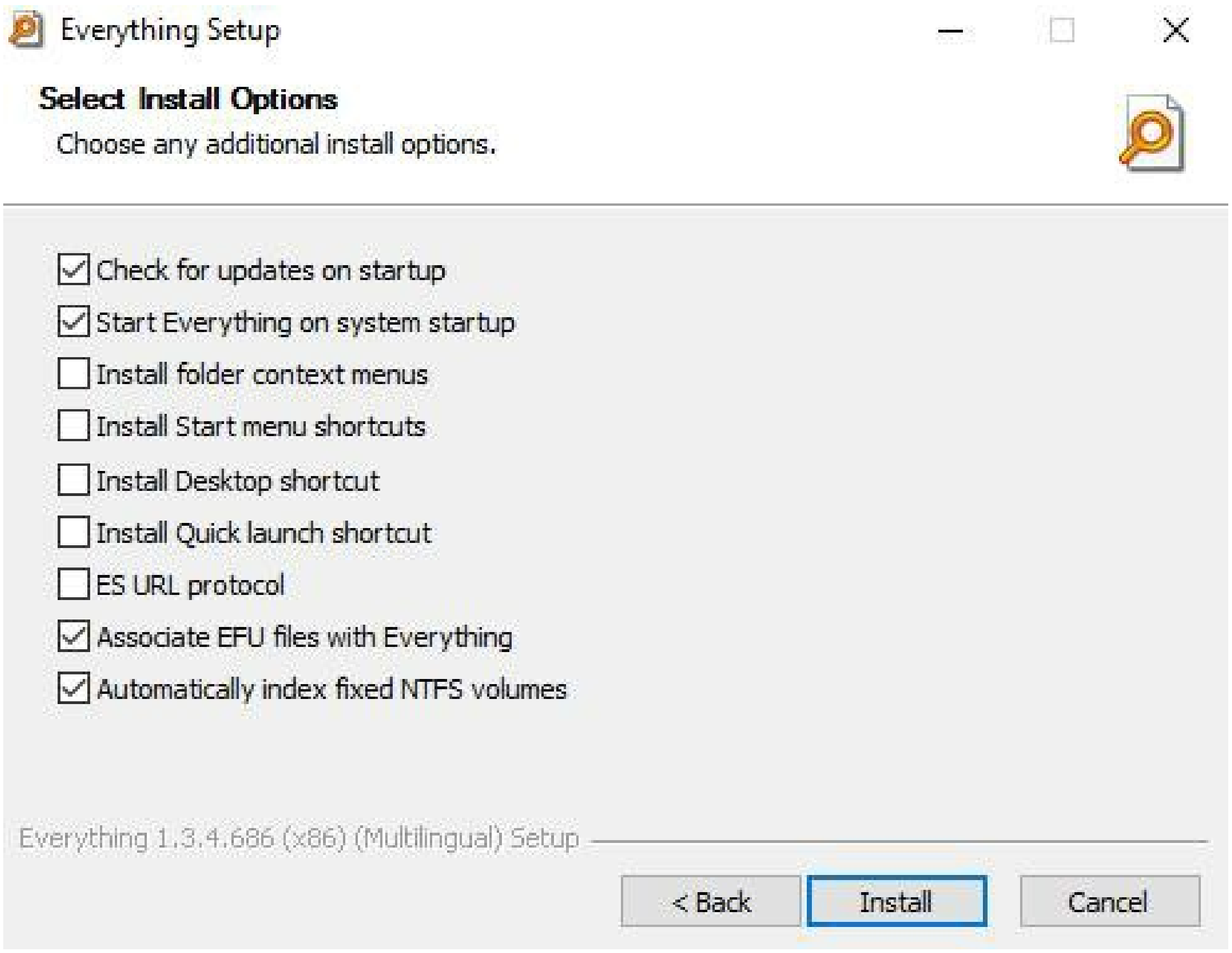
15485

( ) Run as administrator

( ) No NTFS indexing

Everything 1.3.4.686 (x86) (Multilingual) Setup

In this image, and the next one, I chose the place to save the settings and data. I chose to save it under %appdata%, rather than the installation folder as it was the recommended location and it meant that users wouldn't be able to edit the configuration to break the program, as users cannot access the %appdata% folder. I also chose to install *Everything Serice*, which is a way to run *Everything* without administrator access and still use NTFS indexing.


Everything Setup

Select Install Options

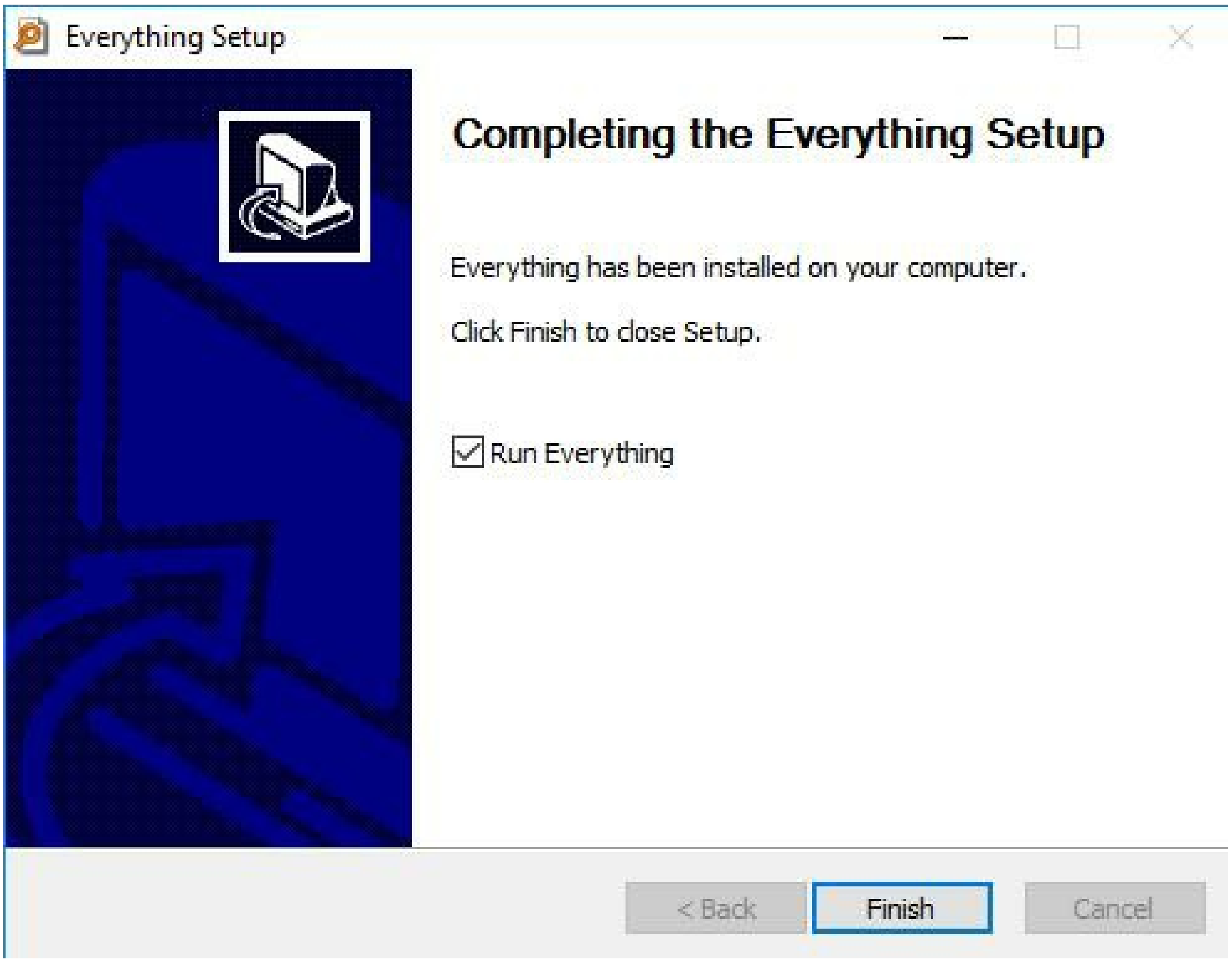
Choose any additional install options.

- [✓] Check for updates on startup
- [✓] Start Everything on system startup
- [ ] Install folder context menus
- [ ] Install Start menu shortcuts
- [ ] Install Desktop shortcut
- [ ] Install Quick launch shortcut
- [ ] ES URL protocol
- [✓] Associate EFU files with Everything
- [✓] Automatically index fixed NTFS volumes

Everything 1.3.4.686 (x86) (Multilingual) Setup

In the image above, I chose to check for updates on start, meaning that the program is always updated. I also chose to run *Everything* on startup, meaning that it would always be run. I decided to not install shortcuts and context menus as I felt that I would take up extra space and would be unused, making them pointless. I decided to associate EFU files with *Everything* as it should speed up the program, which is the same reason as to why I used automatic index fixing.

I was finally finished with the program installation. Here is the final step which gave me the option to run *Everything*.



Everything Setup

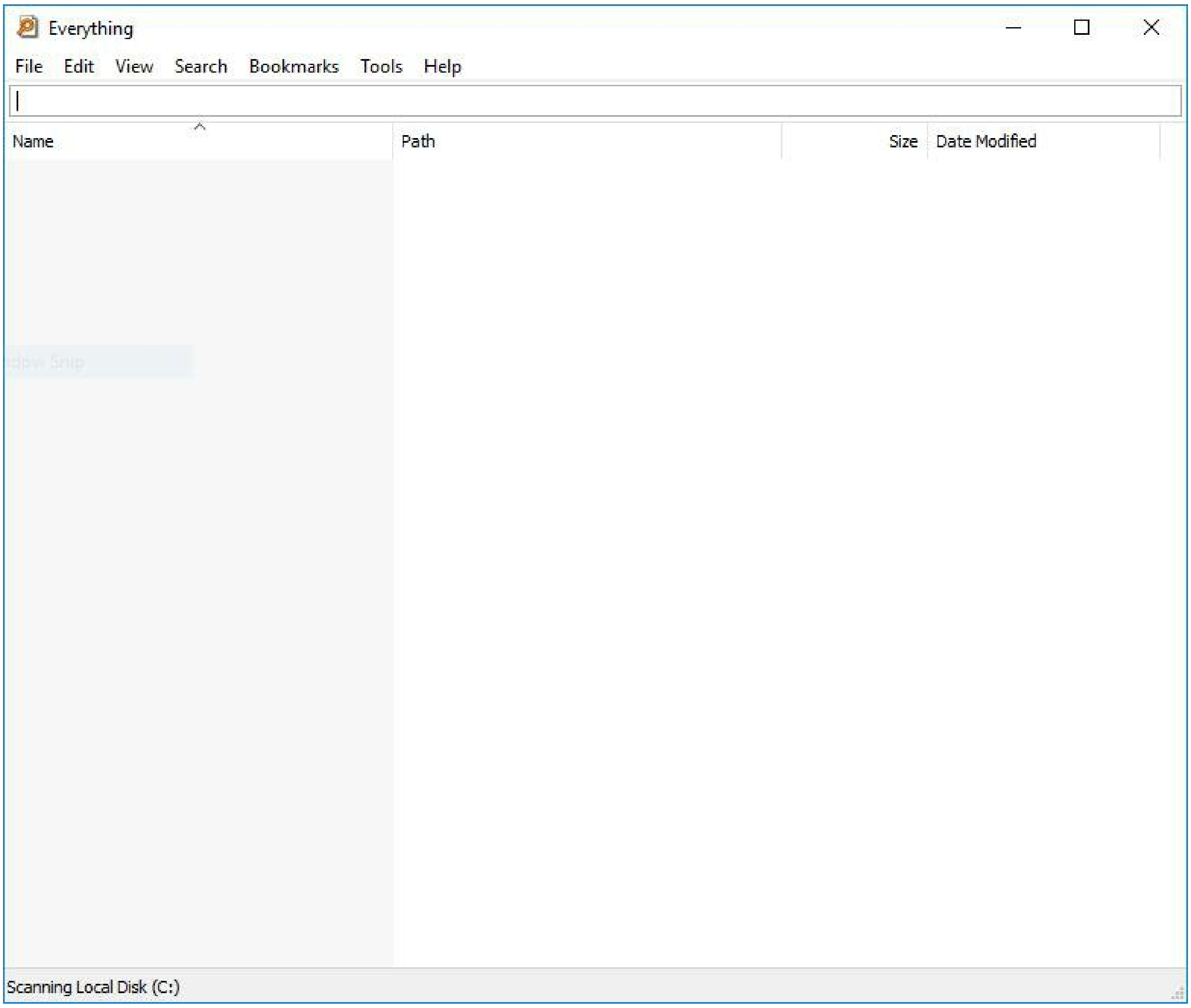
Completing the Everything Setup

Everything has been installed on your computer.

Click Finish to close Setup.

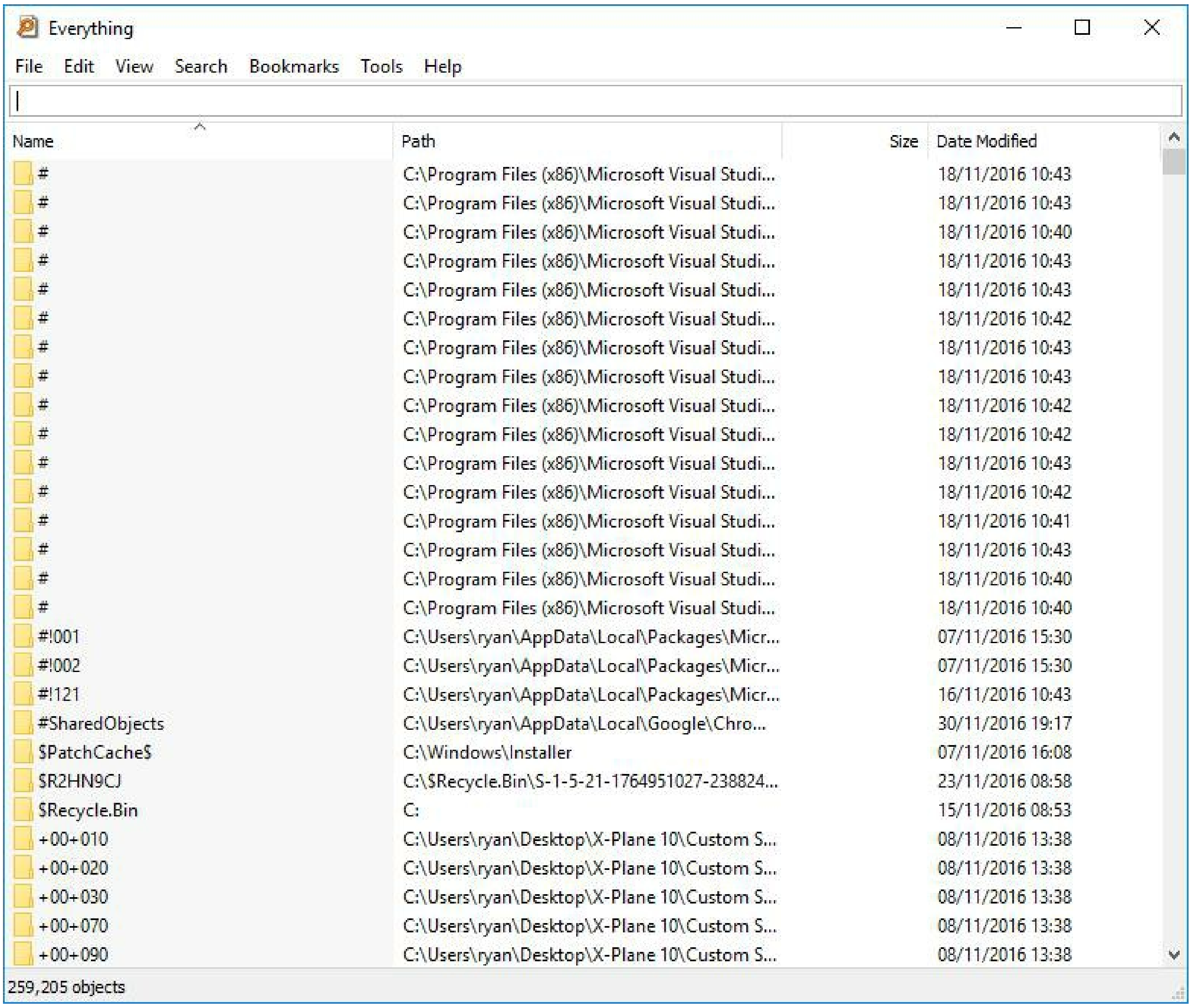
[✓] Run Everything

Also, here is an image of the program without any files being indexed.



Scanning Local Disk (C:)

And here is the program after the files have been indexed.

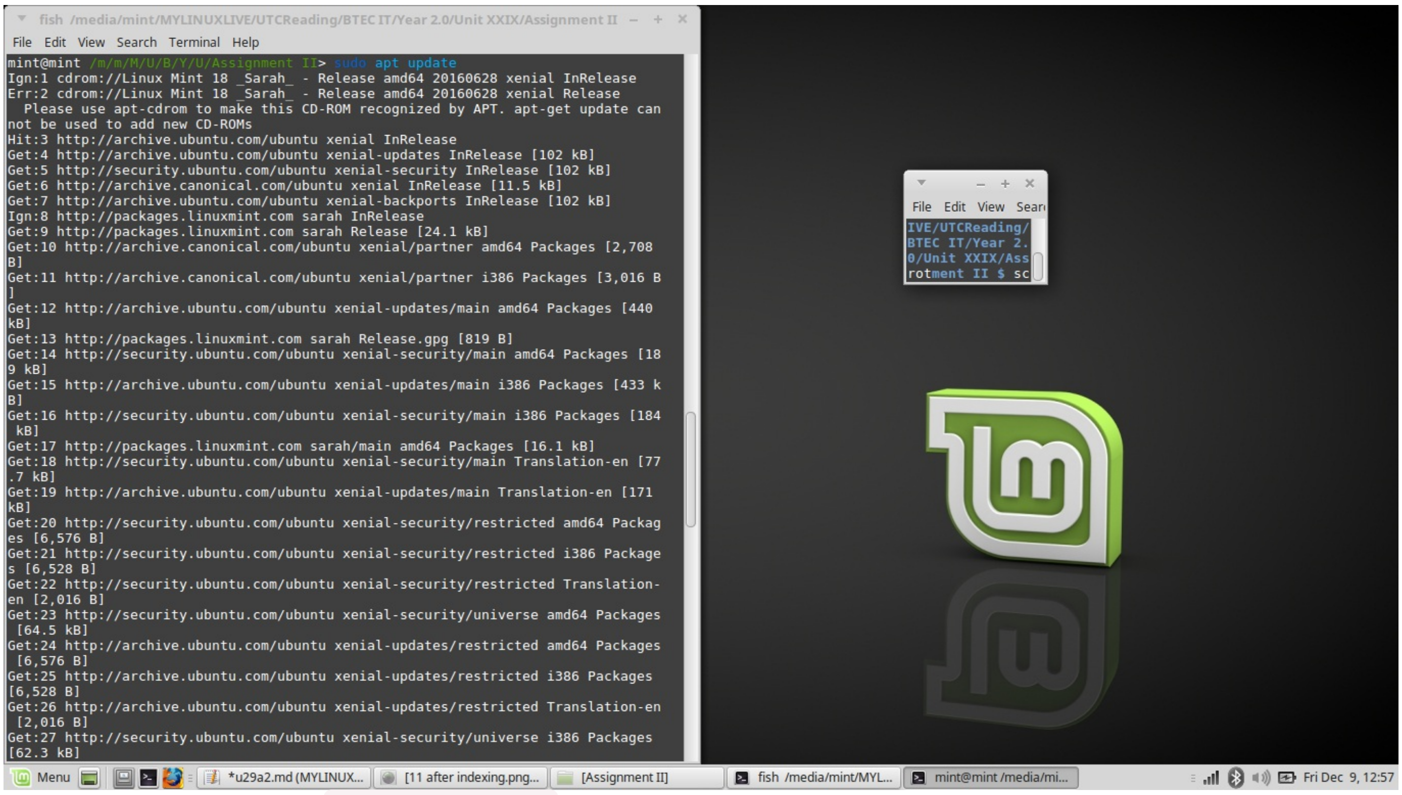


Name	Path	Size	Date Modified
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:40
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:42
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:42
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:42
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:42
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:41
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:43
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:40
#	C:\Program Files (x86)\Microsoft Visual Studi...		18/11/2016 10:40
#!001	C:\Users\ryan\AppData\Local\Packages\Micr...		07/11/2016 15:30
#!002	C:\Users\ryan\AppData\Local\Packages\Micr...		07/11/2016 15:30
#!121	C:\Users\ryan\AppData\Local\Packages\Micr...		16/11/2016 10:43
#SharedObjects	C:\Users\ryan\AppData\Local\Google\Chro...		30/11/2016 19:17
$PatchCache$	C:\Windows\Installer		07/11/2016 16:08
$R2HN9CJ	C:\$Recycle.Bin\S-1-5-21-1764951027-238824...		23/11/2016 08:58
$Recycle.Bin	C:		15/11/2016 08:53
+00+010	C:\Users\ryan\Desktop\X-Plane 10\Custom S...		08/11/2016 13:38
+00+020	C:\Users\ryan\Desktop\X-Plane 10\Custom S...		08/11/2016 13:38
+00+030	C:\Users\ryan\Desktop\X-Plane 10\Custom S...		08/11/2016 13:38
+00+070	C:\Users\ryan\Desktop\X-Plane 10\Custom S...		08/11/2016 13:38
+00+090	C:\Users\ryan\Desktop\X-Plane 10\Custom S...		08/11/2016 13:38

259,205 objects

PV - Record and Complete a Software Upgrade

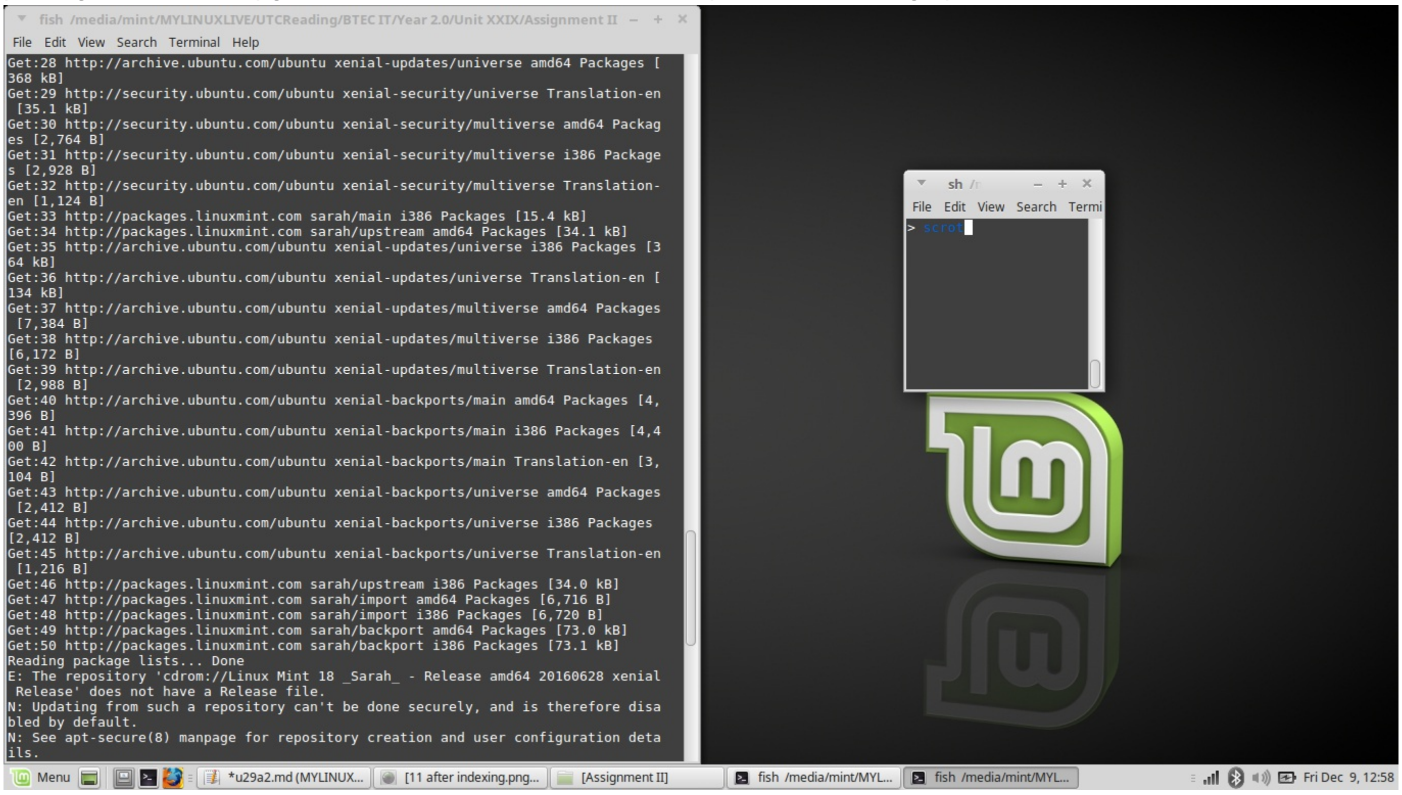
The version of *Everything* that I installed was the latest version, meaning that I could not upgrade anything. Because of this, I decided to update my Linux Mint distribution on my secondary Hard Disk Drive.



```
mint@mint /m/m/M/U/B/Y/U/Assignment II> sudo apt update
Ign:1 cdrom://Linux Mint 18 _Sarah_ - Release amd64 20160628 xenial InRelease
Err:2 cdrom://Linux Mint 18 _Sarah_ - Release amd64 20160628 xenial Release
  Please use apt-cdrom to make this CD-ROM recognized by APT. apt-get update can
not be used to add new CD-ROMs
Hit:3 http://archive.ubuntu.com/ubuntu xenial InRelease
Get:4 http://archive.ubuntu.com/ubuntu xenial-updates InRelease [102 kB]
Get:5 http://security.ubuntu.com/ubuntu xenial-security InRelease [102 kB]
Get:6 http://archive.canonical.com/ubuntu xenial InRelease [11.5 kB]
Get:7 http://archive.ubuntu.com/ubuntu xenial-backports InRelease [102 kB]
Ign:8 http://packages.linuxmint.com sarah InRelease
Get:9 http://packages.linuxmint.com sarah Release [24.1 kB]
Get:10 http://archive.canonical.com/ubuntu xenial/partner amd64 Packages [2,708
B]
Get:11 http://archive.canonical.com/ubuntu xenial/partner i386 Packages [3,016 B
]
Get:12 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 Packages [440
kB]
Get:13 http://packages.linuxmint.com sarah Release.gpg [819 B]
Get:14 http://security.ubuntu.com/ubuntu xenial-security/main amd64 Packages [18
9 kB]
Get:15 http://archive.ubuntu.com/ubuntu xenial-updates/main i386 Packages [433 k
B]
Get:16 http://security.ubuntu.com/ubuntu xenial-security/main i386 Packages [184
 kB]
Get:17 http://packages.linuxmint.com sarah/main amd64 Packages [16.1 kB]
Get:18 http://security.ubuntu.com/ubuntu xenial-security/main Translation-en [77
.7 kB]
Get:19 http://archive.ubuntu.com/ubuntu xenial-updates/main Translation-en [171
kB]
Get:20 http://security.ubuntu.com/ubuntu xenial-security/restricted amd64 Packag
es [6,576 B]
Get:21 http://security.ubuntu.com/ubuntu xenial-security/restricted i386 Package
s [6,528 B]
Get:22 http://security.ubuntu.com/ubuntu xenial-security/restricted Translation-
en [2,016 B]
Get:23 http://security.ubuntu.com/ubuntu xenial-security/universe amd64 Packages
 [64.5 kB]
Get:24 http://archive.ubuntu.com/ubuntu xenial-updates/restricted amd64 Packages
 [6,576 B]
Get:25 http://archive.ubuntu.com/ubuntu xenial-updates/restricted i386 Packages
[6,528 B]
Get:26 http://archive.ubuntu.com/ubuntu xenial-updates/restricted Translation-en
 [2,016 B]
Get:27 http://security.ubuntu.com/ubuntu xenial-security/universe i386 Packages
[62.3 kB]
```

fish /media/mint/MYLINUXLIVE/UTCReading/BTEC IT/Year 2.0/Unit XXIX/Assignment II
File Edit View Search Terminal Help
IVE/UTCReading/BTEC IT/Year 2.0/Unit XXIX/Assignment II $ sc
Menu
*u29a2.md (MYLINUX...
[11 after indexing.png...
[Assignment II]
fish /media/mint/MYL...
mint@mint /media/mi...
Fri Dec 9, 12:57

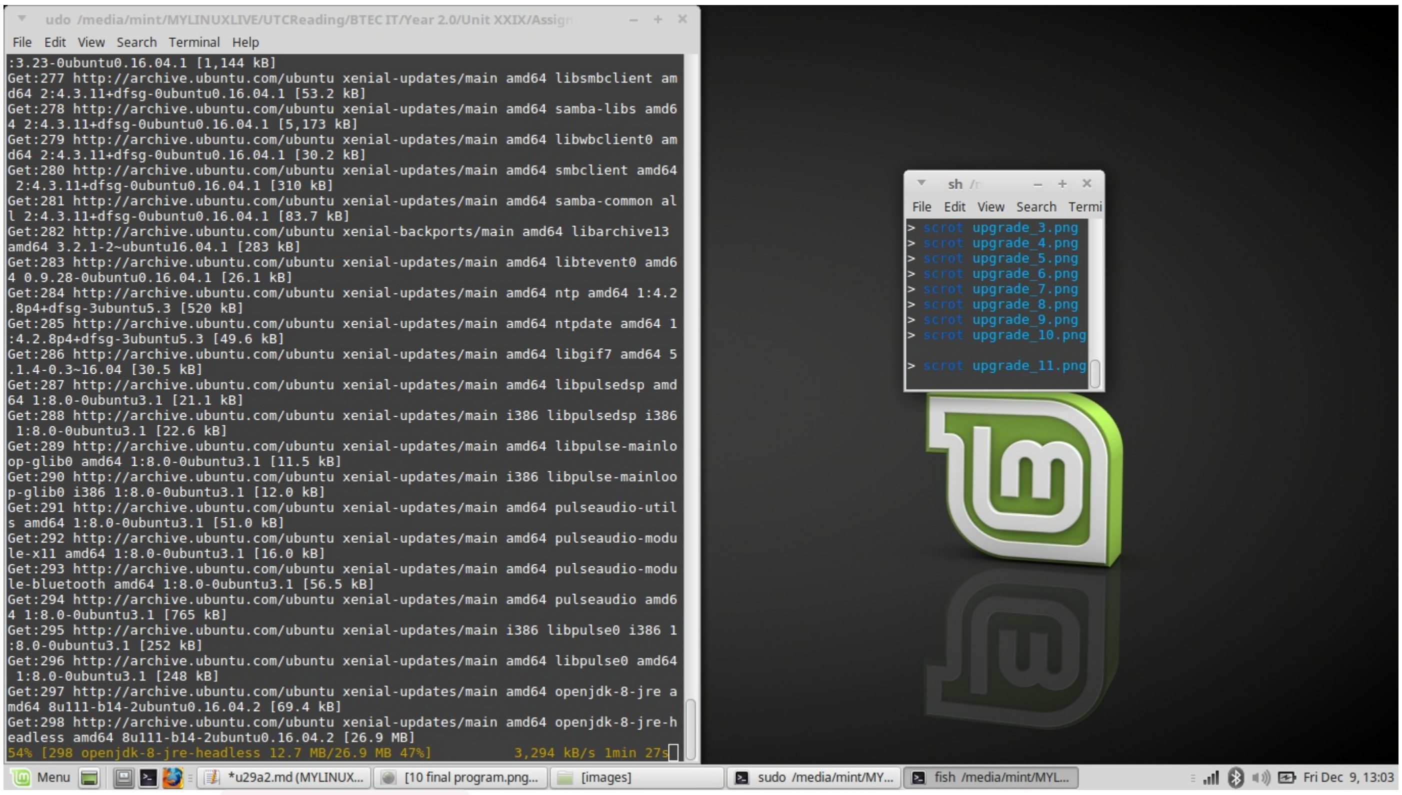
At first, I ran the command `sudo apt update`. This command pings all the file servers for packages that are installed on the machine, including the operating system. It then updates the list of packages that are installed on the machine, and performs changes to the list accordingly. This means that the machine knows where to go to download the packages when the upgrade command is sent. In short, the pointers are being updated.



```
Get:28 http://archive.ubuntu.com/ubuntu xenial-updates/universe amd64 Packages [
368 kB]
Get:29 http://security.ubuntu.com/ubuntu xenial-security/universe Translation-en
 [35.1 kB]
Get:30 http://security.ubuntu.com/ubuntu xenial-security/multiverse amd64 Packag
es [2,764 B]
Get:31 http://security.ubuntu.com/ubuntu xenial-security/multiverse i386 Package
s [2,928 B]
Get:32 http://security.ubuntu.com/ubuntu xenial-security/multiverse Translation-
en [1,124 B]
Get:33 http://packages.linuxmint.com sarah/main i386 Packages [15.4 kB]
Get:34 http://packages.linuxmint.com sarah/upstream amd64 Packages [34.1 kB]
Get:35 http://archive.ubuntu.com/ubuntu xenial-updates/universe i386 Packages [3
64 kB]
Get:36 http://archive.ubuntu.com/ubuntu xenial-updates/universe Translation-en [
134 kB]
Get:37 http://archive.ubuntu.com/ubuntu xenial-updates/multiverse amd64 Packages
 [7,384 B]
Get:38 http://archive.ubuntu.com/ubuntu xenial-updates/multiverse i386 Packages
[6,172 B]
Get:39 http://archive.ubuntu.com/ubuntu xenial-updates/multiverse Translation-en
 [2,988 B]
Get:40 http://archive.ubuntu.com/ubuntu xenial-backports/main amd64 Packages [4,
396 B]
Get:41 http://archive.ubuntu.com/ubuntu xenial-backports/main i386 Packages [4,4
00 B]
Get:42 http://archive.ubuntu.com/ubuntu xenial-backports/main Translation-en [3,
104 B]
Get:43 http://archive.ubuntu.com/ubuntu xenial-backports/universe amd64 Packages
 [2,412 B]
Get:44 http://archive.ubuntu.com/ubuntu xenial-backports/universe i386 Packages
[2,412 B]
Get:45 http://archive.ubuntu.com/ubuntu xenial-backports/universe Translation-en
 [1,216 B]
Get:46 http://packages.linuxmint.com sarah/upstream i386 Packages [34.0 kB]
Get:47 http://packages.linuxmint.com sarah/import amd64 Packages [6,716 B]
Get:48 http://packages.linuxmint.com sarah/import i386 Packages [6,720 B]
Get:49 http://packages.linuxmint.com sarah/backport amd64 Packages [73.0 kB]
Get:50 http://packages.linuxmint.com sarah/backport i386 Packages [73.1 kB]
Reading package lists... Done
E: The repository 'cdrom://Linux Mint 18 _Sarah_ - Release amd64 20160628 xenial
 Release' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disa
bled by default.
N: See apt-secure(8) manpage for repository creation and user configuration deta
ils.
```

```
> scrot
```

fish /media/mint/MYLINUXLIVE/UTCReading/BTEC IT/Year 2.0/Unit XXIX/Assignment II
File Edit View Search Terminal Help
sh
Menu
*u29a2.md (MYLINUX...
[11 after indexing.png...
[Assignment II]
fish /media/mint/MYL...
fish /media/mint/MYL...
Fri Dec 9, 12:58



```
:3.23-0ubuntu0.16.04.1 [1,144 kB]
Get:277 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libsmbclient am
d64 2:4.3.11+dfsg-0ubuntu0.16.04.1 [53.2 kB]
Get:278 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 samba-libs amd6
4 2:4.3.11+dfsg-0ubuntu0.16.04.1 [5,173 kB]
Get:279 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libwbclient0 am
d64 2:4.3.11+dfsg-0ubuntu0.16.04.1 [30.2 kB]
Get:280 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 smbclient amd64
 2:4.3.11+dfsg-0ubuntu0.16.04.1 [310 kB]
Get:281 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 samba-common al
l 2:4.3.11+dfsg-0ubuntu0.16.04.1 [83.7 kB]
Get:282 http://archive.ubuntu.com/ubuntu xenial-backports/main amd64 libarchive13
amd64 3.2.1-2~ubuntu16.04.1 [283 kB]
Get:283 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libtevent0 amd6
4 0.9.28-0ubuntu0.16.04.1 [26.1 kB]
Get:284 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 ntp amd64 1:4.2
.8p4+dfsg-3ubuntu5.3 [520 kB]
Get:285 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 ntpdate amd64 1
:4.2.8p4+dfsg-3ubuntu5.3 [49.6 kB]
Get:286 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libgif7 amd64 5
.1.4-0.3~16.04 [30.5 kB]
Get:287 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpulsedsp amd
64 1:8.0-0ubuntu3.1 [21.1 kB]
Get:288 http://archive.ubuntu.com/ubuntu xenial-updates/main i386 libpulsedsp i386
 1:8.0-0ubuntu3.1 [22.6 kB]
Get:289 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpulse-mainlo
op-glib0 amd64 1:8.0-0ubuntu3.1 [11.5 kB]
Get:290 http://archive.ubuntu.com/ubuntu xenial-updates/main i386 libpulse-mainloo
p-glib0 i386 1:8.0-0ubuntu3.1 [12.0 kB]
Get:291 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 pulseaudio-util
s amd64 1:8.0-0ubuntu3.1 [51.0 kB]
Get:292 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 pulseaudio-modu
le-x11 amd64 1:8.0-0ubuntu3.1 [16.0 kB]
Get:293 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 pulseaudio-modu
le-bluetooth amd64 1:8.0-0ubuntu3.1 [56.5 kB]
Get:294 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 pulseaudio amd6
4 1:8.0-0ubuntu3.1 [765 kB]
Get:295 http://archive.ubuntu.com/ubuntu xenial-updates/main i386 libpulse0 i386 1
:8.0-0ubuntu3.1 [252 kB]
Get:296 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libpulse0 amd64
 1:8.0-0ubuntu3.1 [248 kB]
Get:297 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 openjdk-8-jre a
md64 8u111-b14-2ubuntu0.16.04.2 [69.4 kB]
Get:298 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 openjdk-8-jre-h
eadless amd64 8u111-b14-2ubuntu0.16.04.2 [26.9 MB]
54% [298 openjdk-8-jre-headless 12.7 MB/26.9 MB 47%]            3,294 kB/s 1min 27s
```

```
> scrot upgrade_3.png
> scrot upgrade_4.png
> scrot upgrade_5.png
> scrot upgrade_6.png
> scrot upgrade_7.png
> scrot upgrade_8.png
> scrot upgrade_9.png
> scrot upgrade_10.png

> scrot upgrade_11.png
```

After this I ran the `sudo apt upgrade` command. This command then goes to all the pointers that I previously mentioned and downloads the files. After that, the system upgrades all the packages that require updates, by extracting the files.

PVI - Importance of the User Acceptance process

The user acceptance process is highly important as it allows for the user to see the changes that we have made to their system and they can tell us if they are happy with the changes or not. We need to be able to see what they do not like about the system modifications, if there are any, so that we know how to improve.

The first step in this process is ensuring that the configuration meets the customer’s initial request. Given that the initial request was to perform an upgrade which decreased wait time for file browsing, I would consider this a success. *Everything* allows the user to browse file at lightning speeds, similar to that of an SSD. The user can also enter more advanced search terms using regular expression in order to find the files that they need.

The second step in this process is to hand the devices back to the customer. Given that I work as IT support within UTC Reading, this was relatively easy as I did not have to move any physical devices per se, as all of the hardware was untouched. The software was installed via group policy, so I only needed to show the customer one of the machines as they are all completely identical.

The third and final step to this process is to ensure that the customer is satisfied with our work. All in all, the head of the college, Joanne Harper, was pleased with the upgrade, although she did have concerns with security. While all users can view all files metadata, such as names, extensions, sizes and more, within the program, they cannot view the contents of the file unless stated within our group policy software, ensuring that no files can be tampered with.

MIII - Design and Implement Procedure to Preserve Data Integrity

There are a few things required in the IT world, and one of which is data integrity. Data integrity is the "accuracy and consistency of data within a data structure". This means that all data sent and received within a network needs to be secure. This can be confirmed by using things called checksums. Checksums use mathematics to generate two numbers that are intrinsically linked, the first being a larger number that is bound to the file, and the second is a file that is sent along with it. If the numbers do not match after a mathematical equation is performed on them, such as multiplication, division or modulo, then the file that was sent has been compromised. Compromising can either be in the form of the data not being sent properly, or by the data being manipulated during transfer, either by a system process or an outside force.

Another point of data integrity is the security of the data once it is on the system. A way that this can be achieved is by performing backups. One type of backup is a full backup, where ALL of the data specified is backed up to the external servers. This type of backup may only be performed once every few months.

Another type of data backup is differential, where only the data that has been changed since the last backup occurs is stored elsewhere. This type of backup should be done once every few weeks or so. An example of this is as follows:

- A full backup is performed on Day #1.
- A differential backup is performed on Days #2 and #3. All of these backups will contain the data that has changed from Day #1, meaning that Day #2 will only contain Day #2's work, where as Day #3 will have Day #2 and Day #3's work.
  This means that differential backups can take up lots of space very quickly, which is why it shouldn't be done all the time.

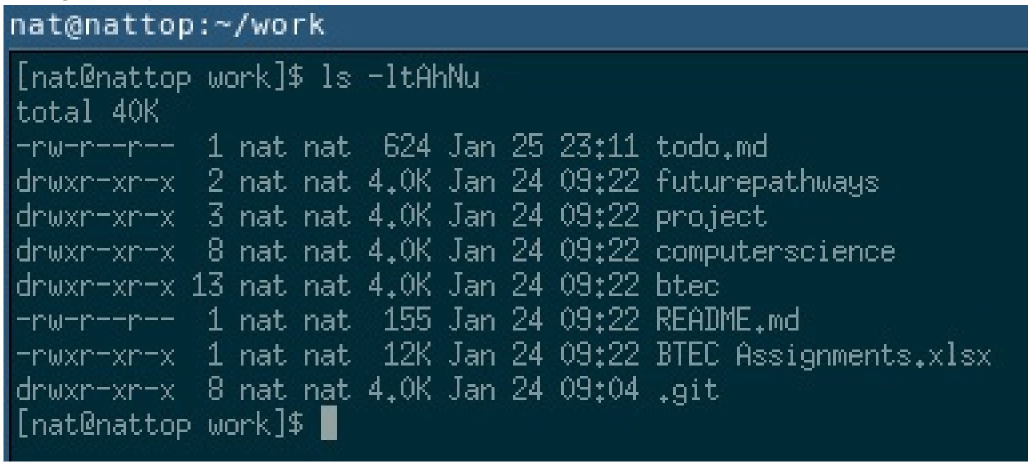
The final type of backup that I shall be covering is incremental, where only the data that has changed since the last incremental backup occurs is stored elsewhere. This type of backup should be performed daily or every other day. An example of this is as follows:

- A full backup is performed on Day #1.
- An incremental backup is performed on Days #2 and #3. All of these backups will contain the data that has changed from Day #x-1, meaning that Day #2 will only contain Day #2's work, and Day #3 will have only have Day #3's work.

Another type of data integrity is restore points. Restore points are a way that Microsoft systems revert to a previous state, including system files, installed applications, registry settings and system settings, from a different point in time. These files are stored using NTFS compression, meaning that they are very compact and can only be used if necessary.

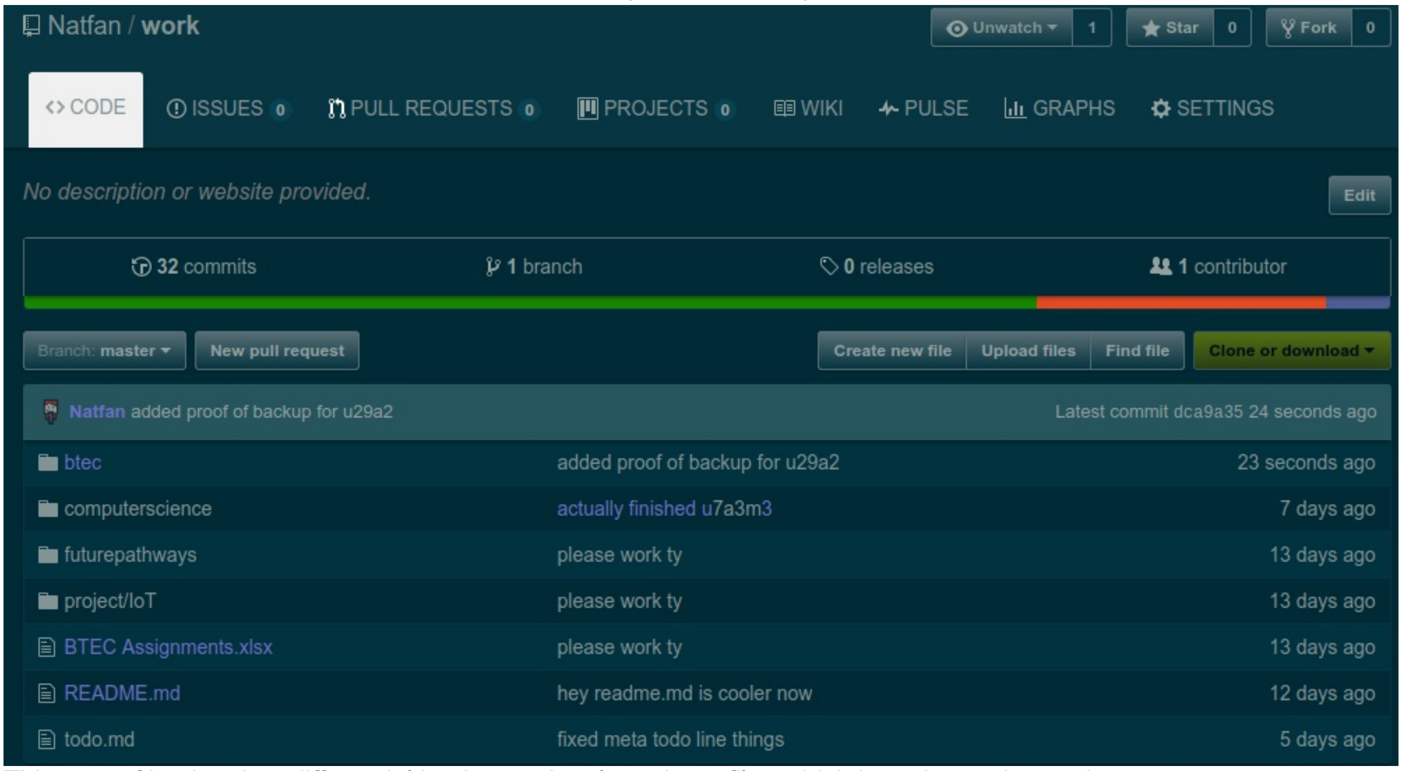
My Linux distribution is running on a live system, meaning that no changes are saved. This means if I mess up an installation all I need to do is restart my machine and all the system files shall be reset, along with any other files and packages I have saved on the system.

The following images are an example of data being backed up. The first one is my local folder where I keep all my college and personal work.



```
nat@nattop:~/work
[nat@nattop work]$ ls -ltAhNu
total 40K
-rw-r--r--  1 nat nat  624 Jan 25 23:11 todo.md
drwxr-xr-x  2 nat nat 4.0K Jan 24 09:22 futurepathways
drwxr-xr-x  3 nat nat 4.0K Jan 24 09:22 project
drwxr-xr-x  8 nat nat 4.0K Jan 24 09:22 computerscience
drwxr-xr-x 13 nat nat 4.0K Jan 24 09:22 btec
-rw-r--r--  1 nat nat  155 Jan 24 09:22 README.md
-rwxr-xr-x  1 nat nat  12K Jan 24 09:22 BTEC Assignments.xlsx
drwxr-xr-x  8 nat nat 4.0K Jan 24 09:04 .git
[nat@nattop work]$
```

The next image is a screenshot of my data uploaded to the servers on http://github.com, a website used by developers to share their code but I use it to be able to access my work from any computer.



Natfan / work

Unwatch 1 | Star 0 | Fork 0

CODE | ISSUES 0 | PULL REQUESTS 0 | PROJECTS 0 | WIKI | PULSE | GRAPHS | SETTINGS

No description or website provided.

32 commits | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

Natfan added proof of backup for u29a2 — Latest commit dca9a35 24 seconds ago

btec	added proof of backup for u29a2	23 seconds ago
computerscience	actually finished u7a3m3	7 days ago
futurepathways	please work ty	13 days ago
project/IoT	please work ty	13 days ago
BTEC Assignments.xlsx	please work ty	13 days ago
README.md	hey readme.md is cooler now	12 days ago
todo.md	fixed meta todo line things	5 days ago

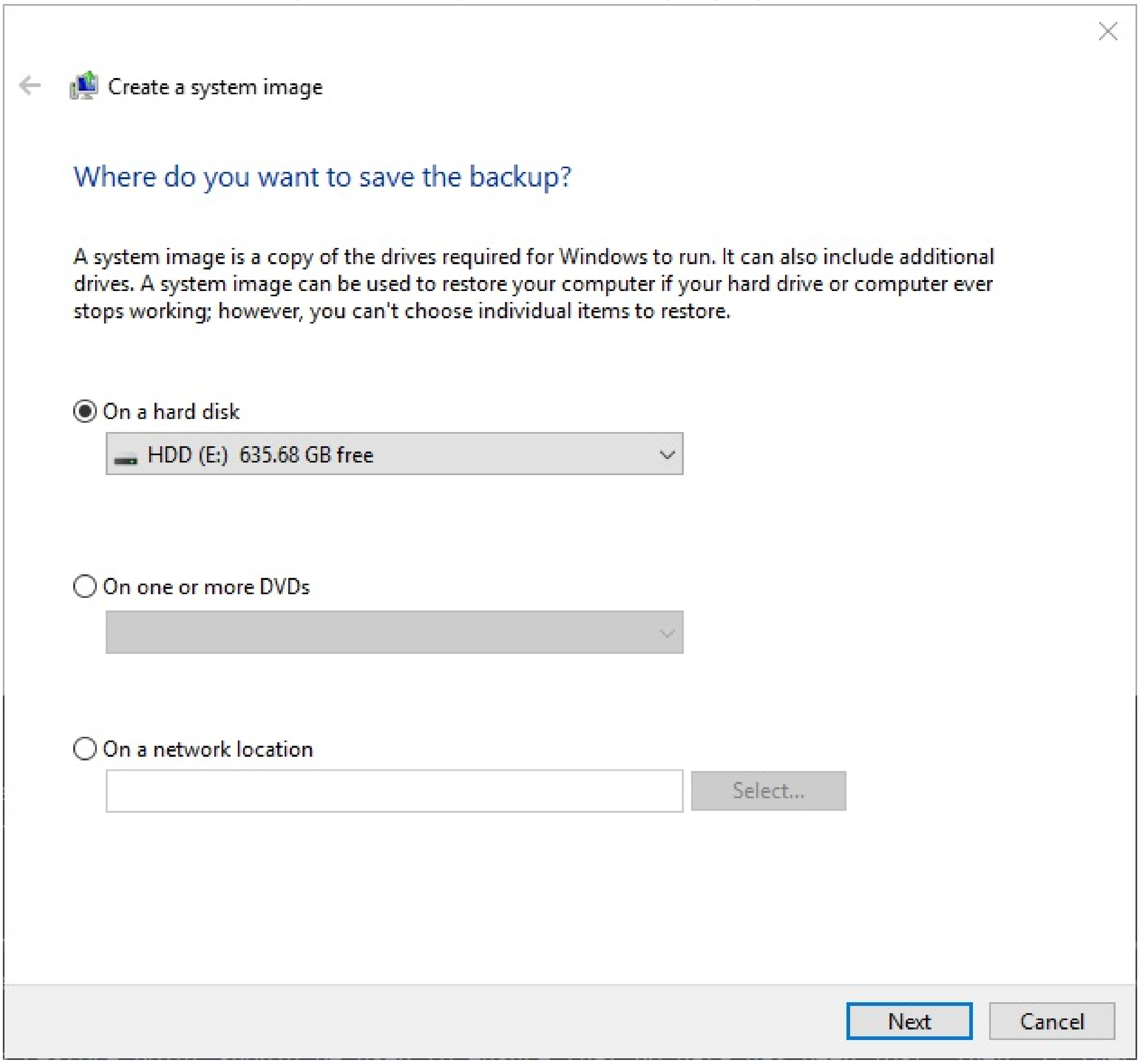
This type of backup is a differential backup as it only updates files which have been changed.

MIV - Design a Procedure to Back Out of Software Upgrades

Back out procedures are highly important when installing new software. They are to be used if the software installed creates errors that prevent the system from running at an optimum standard. The back out procedure that UTC Reading should follow should be:

- Create a system restore point at the start of the upgrade.
- Perform the upgrade
- Run tests on the system
- If required, restore from the system restore point.

The **system Restore Point** should be made to take up as little data as possible, to ensure that the Hard Disk Drive can be used for actual work. Another feature of this backup procedure could be that the system restore points are saved to an off site location, for even greater security to prevent from physical threats such as a fire. This method can also ensure that the finer configurations of the system are saved using **Registry Backups**



Create a system image

Where do you want to save the backup?

A system image is a copy of the drives required for Windows to run. It can also include additional drives. A system image can be used to restore your computer if your hard drive or computer ever stops working; however, you can't choose individual items to restore.

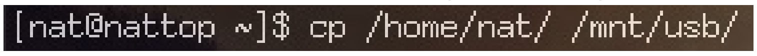
[✓ On a hard disk]
HDD (E:) 635.68 GB free

On one or more DVDs

On a network location

This is an example of generating a restore point using the restore point system within Microsoft's Windows.

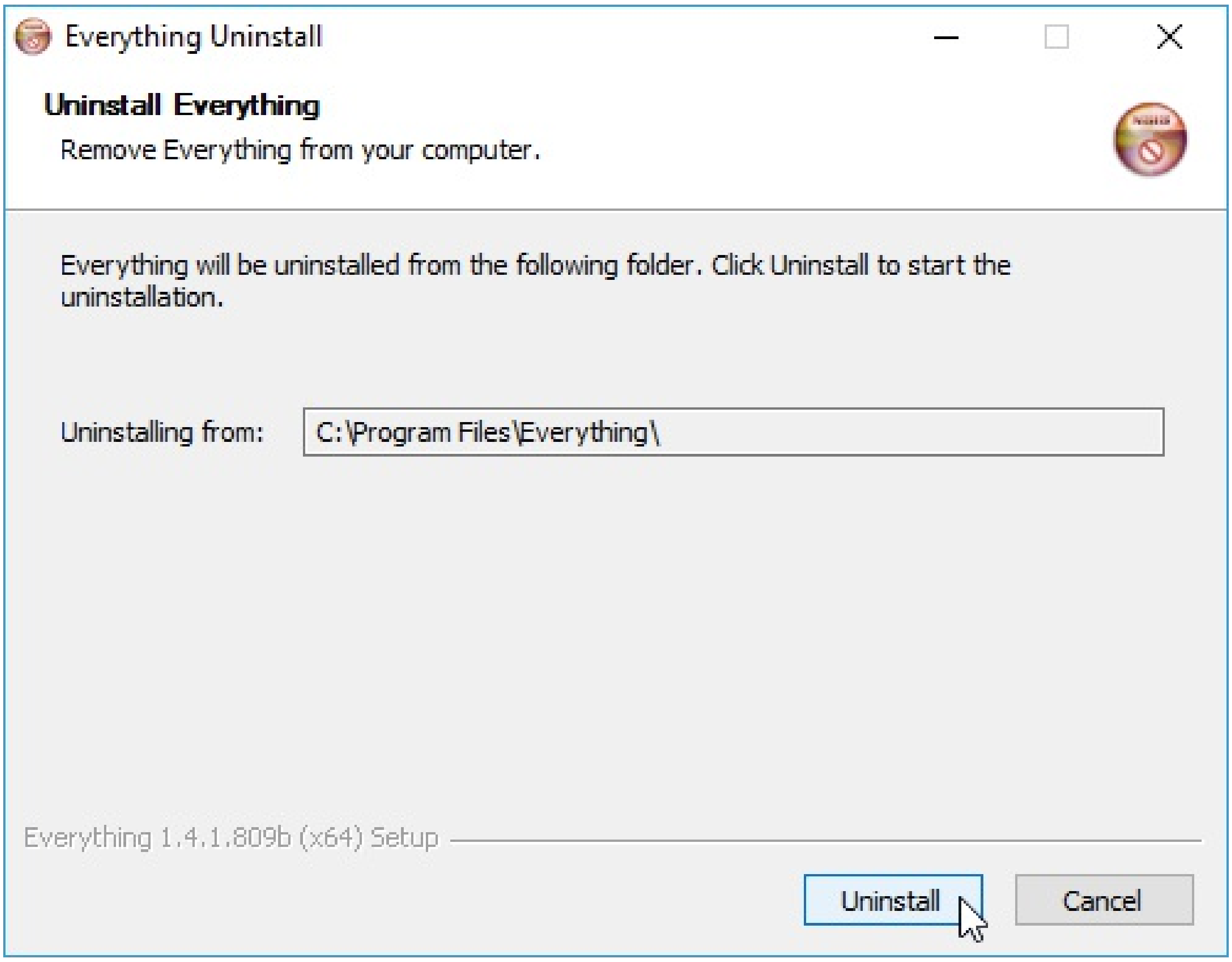
Another thing that can be done is **Hard Drive Cloning**, which is where the data on a Hard Disk Drive or Solid State Drive is saved to another location for greater storage.



```
[nat@nattop ~]$ cp /home/nat/ /mnt/usb/
```

This is an example of copying the home directory of my Linux install onto a mounted USB drive. This home directory is where all of my important data is saved.

Also, you must ensure that the software that has just been installed **can be uninstalled** from the system, which may fix the issue.



Everything Uninstall

Uninstall Everything

Remove Everything from your computer.

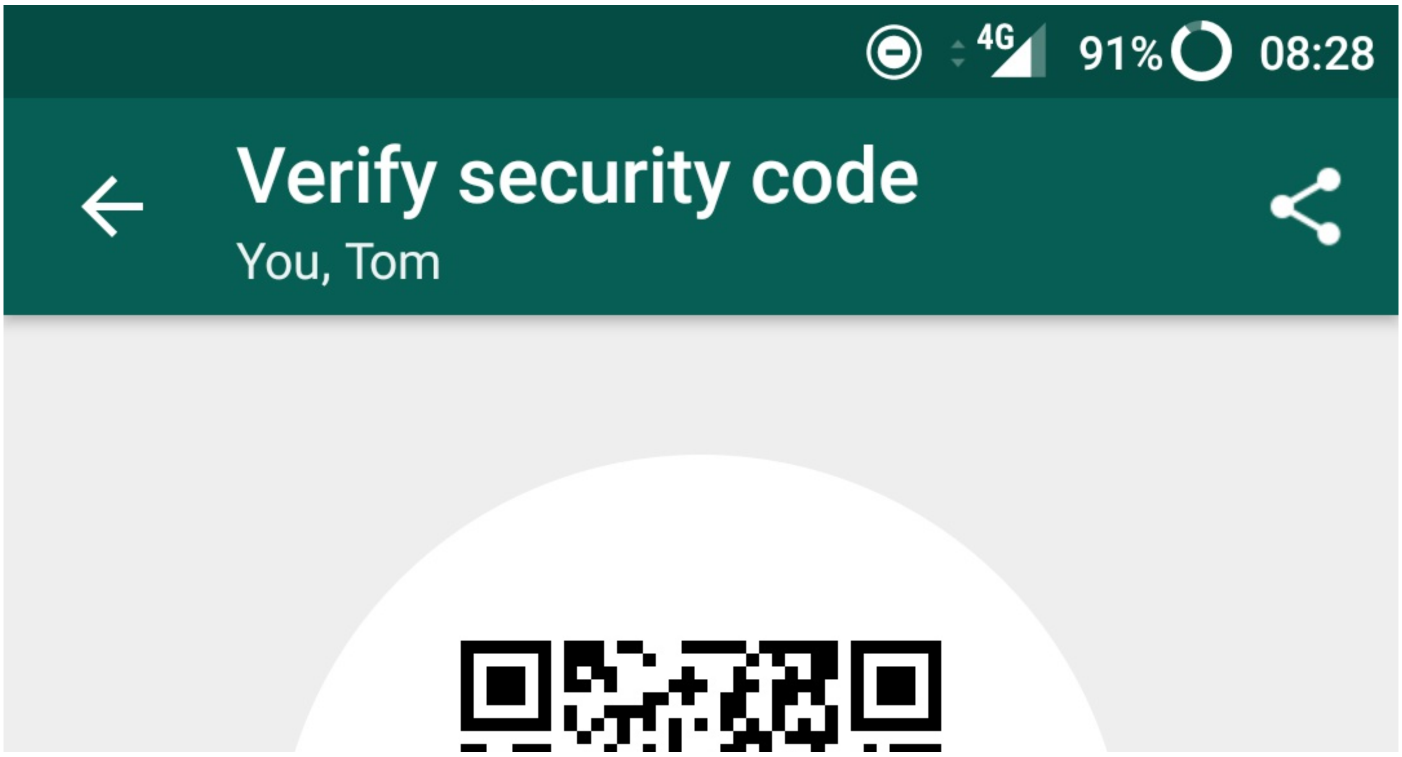
Everything will be uninstalled from the following folder. Click Uninstall to start the uninstallation.

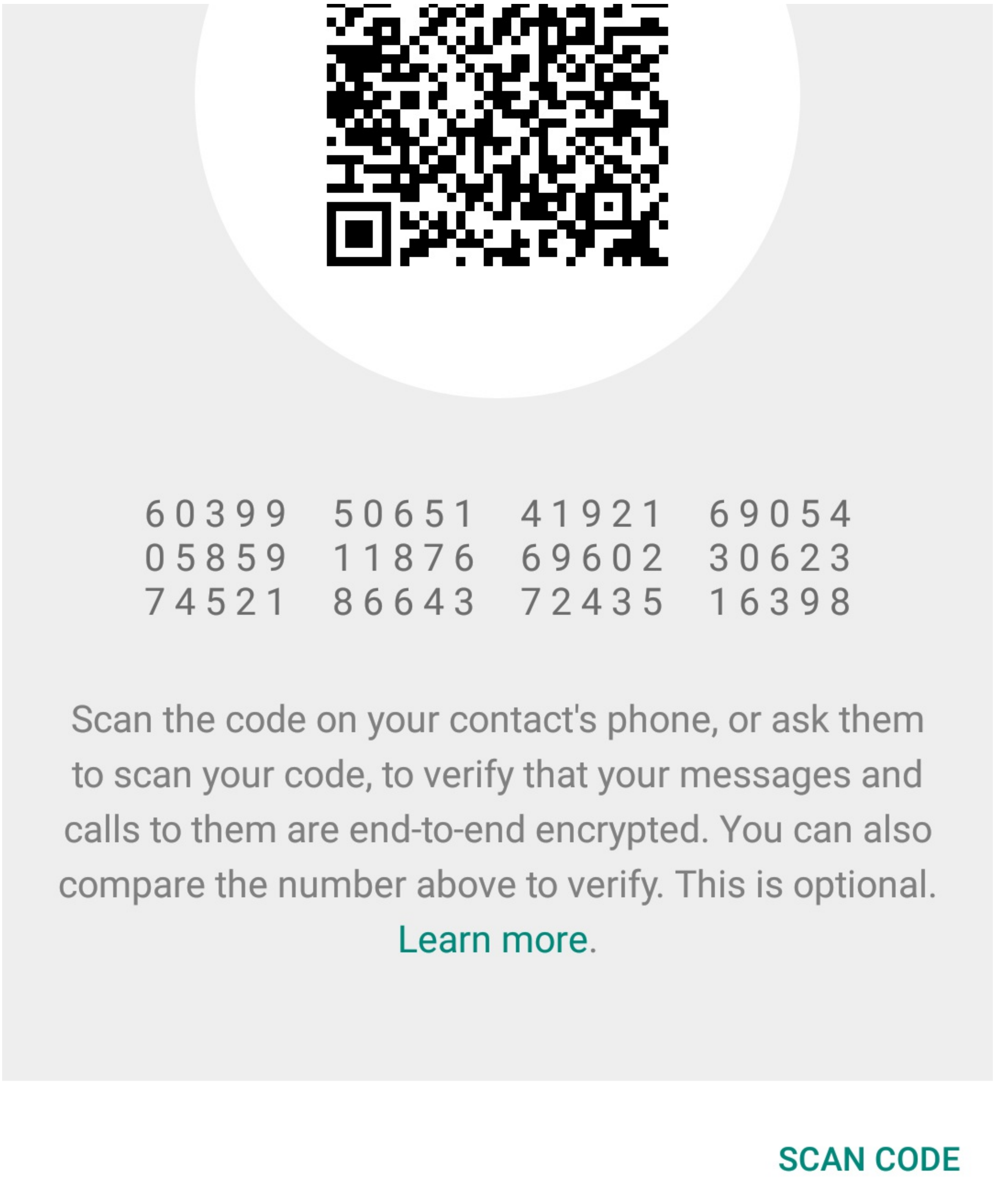
Uninstalling from: C:\Program Files\Everything\

Everything 1.4.1.809b (x64) Setup

This is an example of the *Everything* software being uninstalled.

Finally, **data integrity** must be considered when performing backups. All the data in the backups must be identical to the ones taken from the drives before the upgrade was performed.

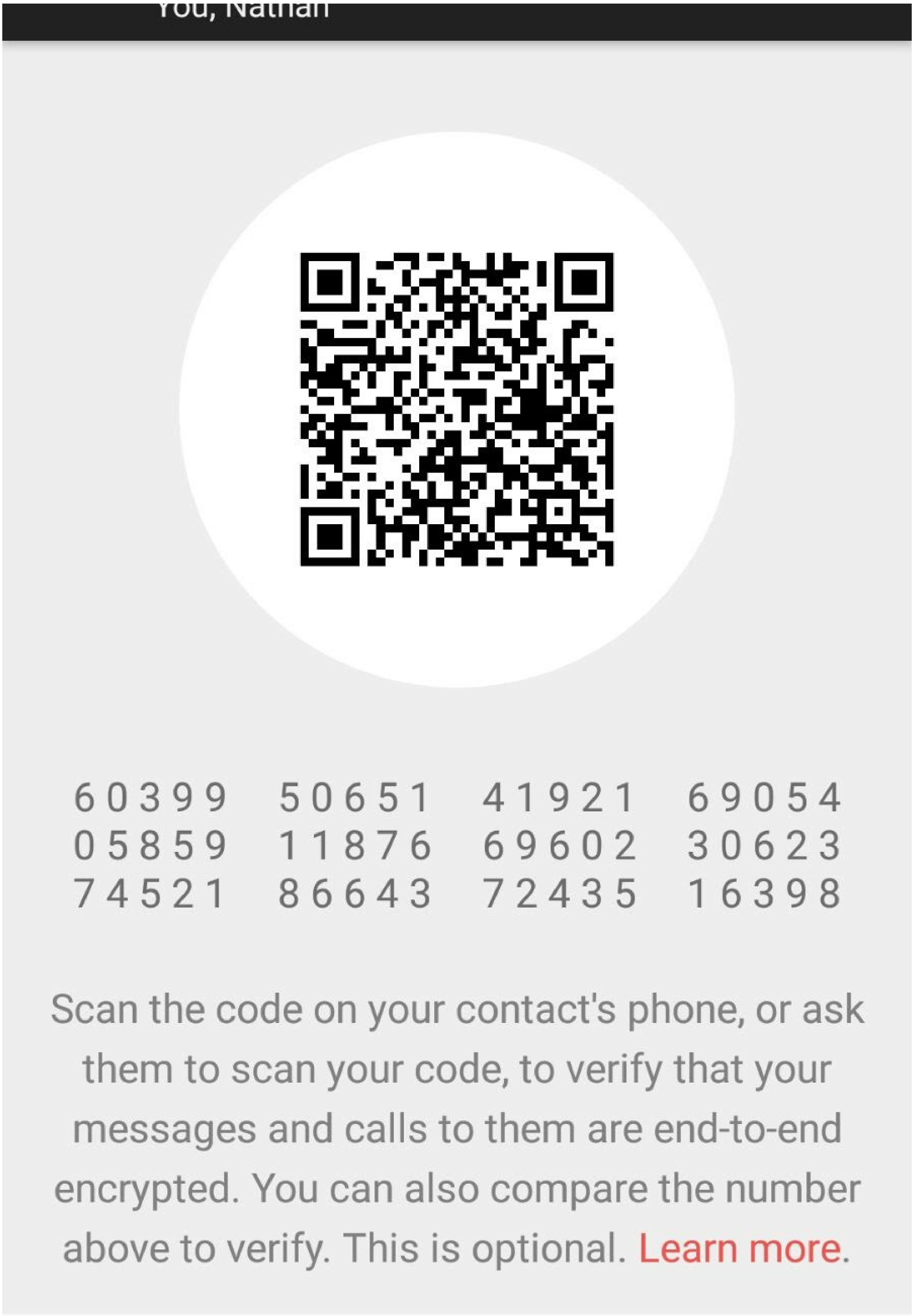




```
60399   50651   41921   69054
05859   11876   69602   30623
74521   86643   72435   16398
```

Scan the code on your contact's phone, or ask them to scan your code, to verify that your messages and calls to them are end-to-end encrypted. You can also compare the number above to verify. This is optional. Learn more.

This is an example of data encryption from the chatting application "WhatsApp". The first image is a screenshot of my side of the conversation with my colleague *Tom Hutching*, with the checksum in the center of the page. The second image is his side of the conversation. As you can see, his checksum numbers are the same as mine.



You, Nathan

```
6 0 3 9 9   5 0 6 5 1   4 1 9 2 1   6 9 0 5 4
0 5 8 5 9   1 1 8 7 6   6 9 6 0 2   3 0 6 2 3
7 4 5 2 1   8 6 6 4 3   7 2 4 3 5   1 6 3 9 8
```

Scan the code on your contact's phone, or ask them to scan your code, to verify that your messages and calls to them are end-to-end encrypted. You can also compare the number above to verify. This is optional. Learn more.

The next step that should be taken is the actual system upgrade itself. If multiple changes are being made, only one should be performed at this stage. This change should only be performed on the one computer that the upgrade is happening on, due to the fact that there are multiple things that could go wrong if the entire network was updated, the main one being that, if something went wrong, all computers that are being worked on would be unusable.

The next step is to run tests on the system, specifically around the files that were changed during the installation. Can files that the user should be able to access be changed? Can system files that the user shouldn't be able to access be changed? Does the machine run faster or slower than it did before? All these questions and more should be answered when testing the system.

Finally, if there is an issue with the semi-updated system, the restore point should be used to ensure that no data has been lost. Then, if there is no possible way of fixing or reproducing the errors that were found, then the upgrade should be called off. The back out protocol should be run, and possible alternate software solutions should be sought out to ensure that the upgrade can be performed in the future.

If all things went well and there was only one change that had to be made, then the upgrade should be considered a success. The change should be applied to all systems that require the change, and they should be updated one at a time, to lower the amount of outage caused by the upgrade.

If there were multiple changes to consider, but the first upgrade went well, then the IT Team should go back to the first step and create a system restore point of the semi-upgraded system. Then, the steps should be performed again but with the secondary software. If the secondary software fails to install, then the IT Team should revert to the newest restore point and **back out** of the upgrade. Otherwise, a new restore point should be made and the cycle should begin again, if there are anymore changes to be made.

DI - Justify a Particular Installation up Upgrade

The upgrade that we are performing on the system are necessary as they allow for both faster access to files, along with updated software. The cost of not having these upgrades occurring may be more than the cost of performing them, due to the fact that lots of time can be lost searching for files. Also, both of the upgrades are free so, forgetting about backup costs and the possible price of failure due to incompatibility errors, there will be a definite profit incurred when upgrading the system and installing *Everything*. Hardware requirements should already be fine, due to the fact that the two changes that we have made to the system are very lightweight and not very resource intensive.

Security threats should not be a concern from *Everything*, as it is open source program meaning that there should be no viruses as all the code is clearly visible. As far as updating the Linux system, only packages that you trust should be installed, meaning that the upgrade should not add any viruses that were not there before. As long as the files have been backed up, data and system integrity should be sound and any corruption should not affect the system too greatly.

Another factor that should be considered is that the software that we are installing is greatly beneficial to the users. *Everything* allows them to search for and find files at an incredible speed, therefore negating any time spent searching for files, and instead using that time for work related activities such as writing up BTEC assignments or filling out past papers or writing code. The possibilities that the time that was previously lost is vast, meaning that both the students and the teachers have a lot more time on their hands to perform activities that are beneficial to both them and their work, along with the college as well.

Finally, possible file corruption must be considered when upgrading the system. If both the upgrade and the backout protocol fail then will the possible cost of all files affected by the change being corrupted be worth the upgrade? When working with large systems such as UTC Reading’s servers then every possible outcome must be considered in order to ensure that even if anything bad happens then we have protocols and procedures in place to save any work that may have been temporarily lost or corrupted within the errors that occured, resulting in a minimal loss of files.