

mini
~

**GET OUT OF MY DISC
GET INTO MY DRIVE**



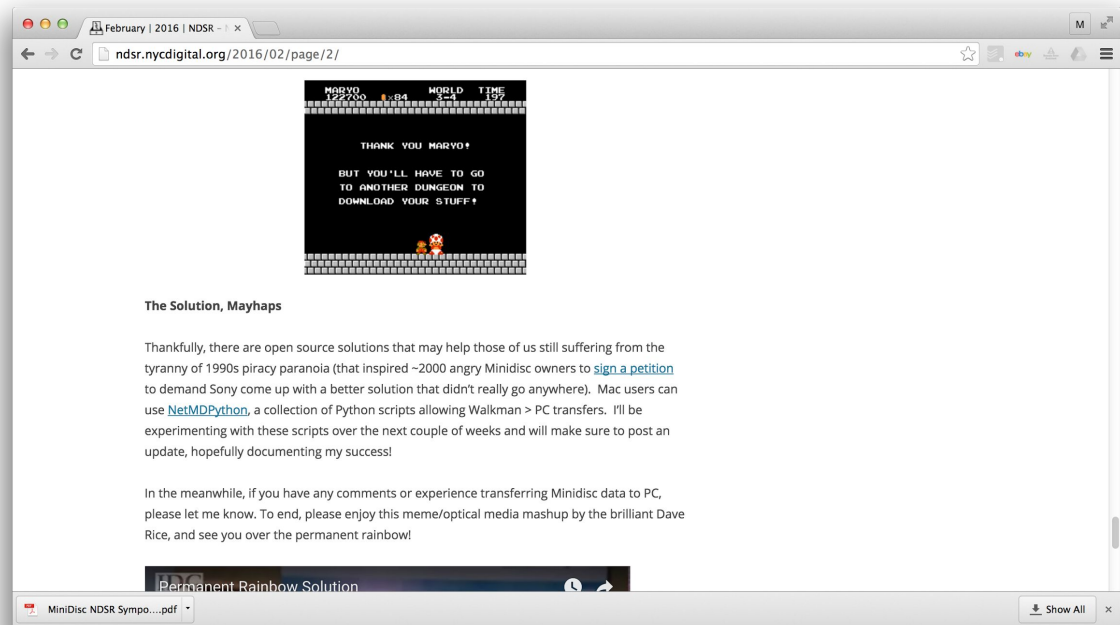
Hi everyone. Nice to see you all here today. My name is Mary Kidd, and I'll be speaking a little bit today about MD-Rs, which is short for MiniDisc Recordables.

NEW YORK
PUBLIC
RADIO



Before I begin, a little bit about myself. I have been working these past 8 months at New York Public Radio's archive as a National Digital Stewardship Resident. For my project, I am devising a "Digital Preservation Roadmap" report, whose goal is to provide NYPR with a detailed investigation of the current landscape of the organization's digital collections and formulate recommendations for the long-term.

One of the residency's requirements is to spend some of our time learning skills outside the scope of our project so I decided to spend that time learning about optical media technology in the archive and focused on CD and MiniDisc Recordables.



Back in February, I wrote an NDSR blog post about my foray into learning about optical media technology and data extraction. It got a nice response. Some readers took the time to share with me their headache-inducing experiences dealing with optical media in their collections. A surprising number of these people worked in radio archives.

This presentation will cover some of those points that I touched upon in the blog post; namely:

- How and why MD-Rs were used by reporters at NYPR;
- How Sony, who developed MD technology, built digital rights management restrictions into its hardware and software;
- And how how DRM can make an archivist's life a bit of a nightmare.

At the end of my blog post, I said that I had just discovered a possible solution called netmdpython, and that I would update my readers with the results of my experiments deploying these scripts. So, this presentation is that update. I will provide some short demos of how I successfully used these scripts to download tracks from MD-Rs.



So let's begin with the question: what is a MiniDisc? Admittedly, when I first heard about the MiniDiscs format, I confused them with the 3" CD, also known as CD3. Here up on the slide is a Fleetwood Mac CD3, in case you've never seen one.

However, MDs are a totally different sort of format and employ a completely different recording technology than CDs, which I will get into in a bit.

And it's okay if you have a vague understanding about what MDs are, like I did. There's no shame in feeling confused over not recognizing an obscure format, because there are a lot of obscure formats out there.



CD-RW



DAT



Compact cassette



DCC



Elcaset (???)



Microcassette

What you see here is just a very small sample of otherwise dozens of formats released in the 80s/90s/early 2000s that were all trying to grab the lion's share of the recordable media market. Philips Digital Compact Cassettes or DCCs, CD-Rewritables, Elcaset (you ever heard of an Elcaset? I haven't), microcassettes, compact cassettes. None of these formats really caught on and over time they began piling up in dusty and dark corners of our memories.

The real doozy here is that green Sony DAT sitting up there in the top row. Sony introduced DATs in 1987, 5 years prior to their debut of MiniDiscs in 1992. In doing so, Sony was butting up its own product lines against one another to see who would come out the winner. To characterize this time in audio recording technology development as competitive is an understatement.

All of these formats touted compactness, portability, sound quality, erase and re-recording abilities. They also all had their own format-specific playback devices. And all of these formats experienced their own day of doom in the market, including...

Oregon Trail Tombstone Generator

New Tombstone Share on Facebook Tweet <http://bit.ly/1U2nj1L>



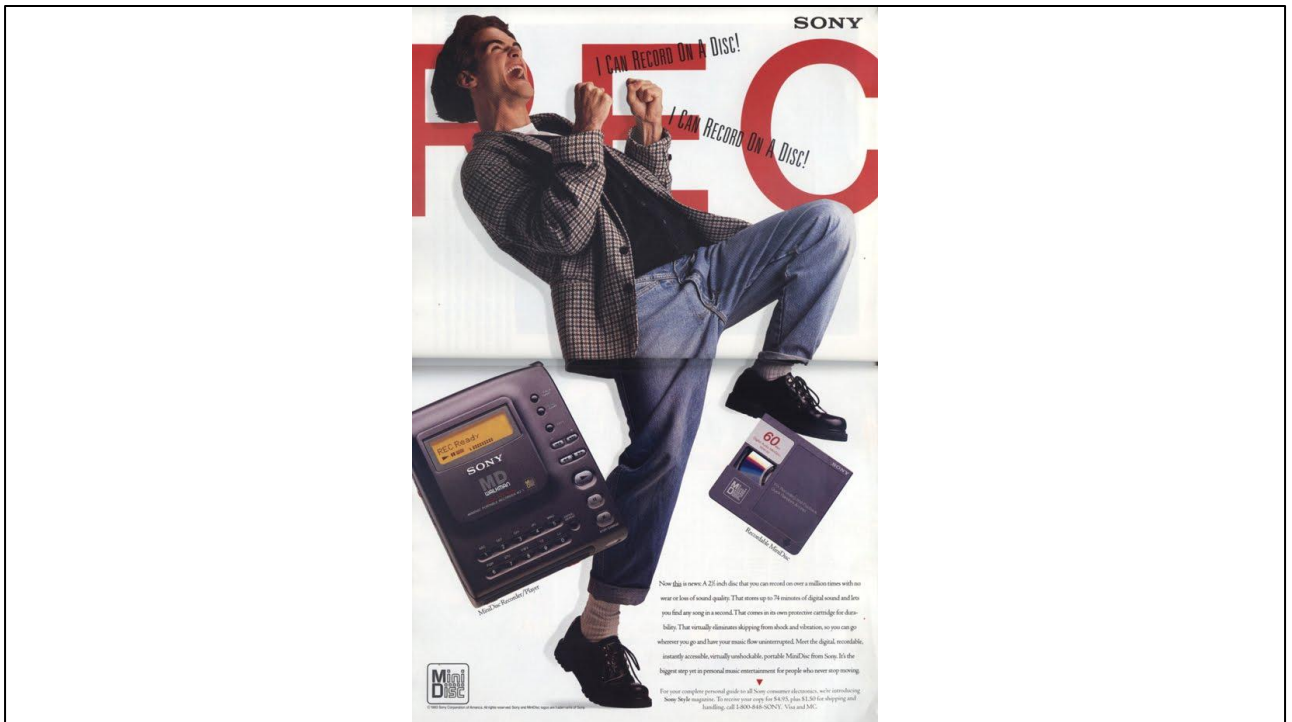
Our subject for today: The Minidisc.



The NYPR Archive has over 3,000 MD-Rs in its holdings. Sizeable chunks of long-time WNYC reporter's careers are stored in MD-R format. What you see here is a photograph of an MD-R I pulled out of a box at random. A Hillary Clinton Press Briefing from 13 years ago.

So, why did reporters at NYPR use MD-Rs?

- MiniDisc dimensions were approximately 2.75" tall x 2.65" wide and 0.18" thick and weighed in at around 0.6 oz. So they were small. Their portability appealed to reporters running around in the field trying to capture sound bytes for newscasts on deadline.
- For being so small, MD-Rs could pack in a lot of data. Earlier versions of MD-Rs could store up to 160 megabytes - or 74 minutes - of CD-quality audio: later versions offered up to 80 minutes of audio. And for the most part, the sound was considered as good as CD quality.
- Also, MiniDiscs were fast: a reporter could record, erase, rewrite and rename tracks with a few presses of a button. This was preferred over compact cassettes, where you would have to sit and wait for a tape to fast-forward or rewind.



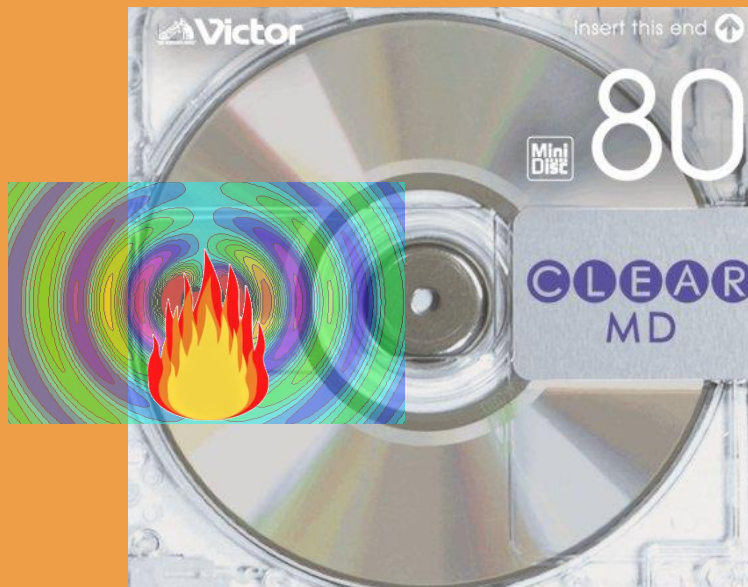
Some more advantages:

- The magnetic recorded area was sandwiched into a double-shuttered protective plastic cartridge. So, unlike their CD cousins, you didn't have to worry about accidentally scratching or smudging the disc.
- MDs also offered shock resistance. I am sure some of you in this audience at one point or another owned a CD Walkman. You may remember having to hold the Walkman still on a shaky bus ride or in your car in order to prevent the audio signal from "skipping", which was a huge pain in the neck. MD technology solved this problem by reading the data into a memory buffer at a higher speed than was required before being read out to the digital-to-analog converter.



Like CDs, MDs were sold either as a mass produced pre-recorded MD or a blank MiniDisc-Recordable. Here's a photo of a WHAM! Best of MiniDisc case, with liner notes and cartridge, going for about \$30.00 right now on eBay. This is an example of a commercial, pre-recorded MD.

Pre-recorded MDs were made and played back using pretty much the same sort of technology as standard pre-recorded CDs you would buy at a record store. On the other hand, MD-Rs employed "magneto-optical" recording and playback technology.



So, how does “magneto-optical” recording and storage technology work? Simply put:

- When recording, a laser heats a spot on the recording layer to a specific temperature.
- While this is happening, a head in contact with the other side of the disc bathes the heated spot in a magnetic field. The magnetic particle's orientation corresponds to 0s and 1s in the data.
- Once the heat is taken away, the particles are frozen into position.

And that is how you record an audio signal into the MD substrate. To erase an old recording and record something new, simply re-heat and re-bathe the area in a new magnetic field.

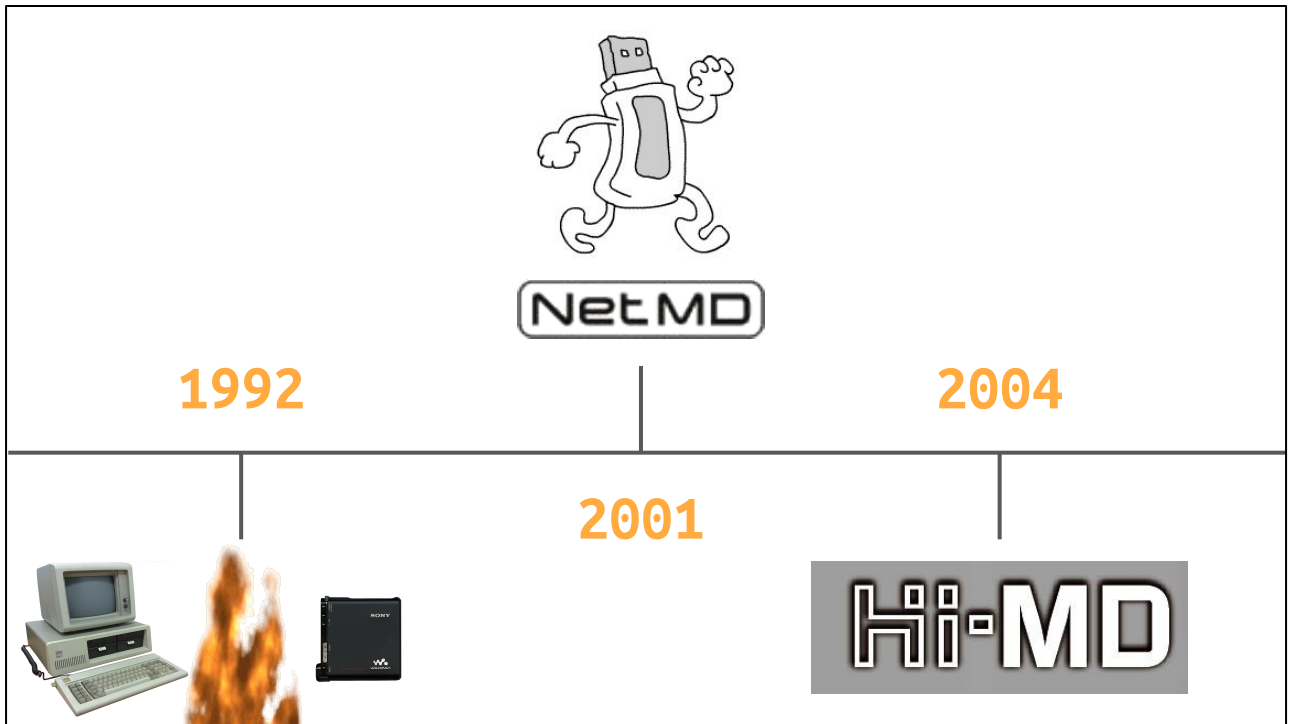
During playback the MD machine focuses the laser on the same spot again, but at lower power, and the data is read back by measuring changes in polarization of light reflected from the previously magnetized regions.

ATRAC©



So, a little more about how so much data could be stored on such a small surface. Sony developed a special perceptual audio codec for MDs called Adaptive Transform Acoustic Coding or ATRAC. ATRAC effectively reduced a 16-bit stereo sample into 292 kbps: a fifth less than of the original data rate with minimal reduction in sound quality. Without ATRAC, an MD-R could store only about 15 minutes of recorded audio.

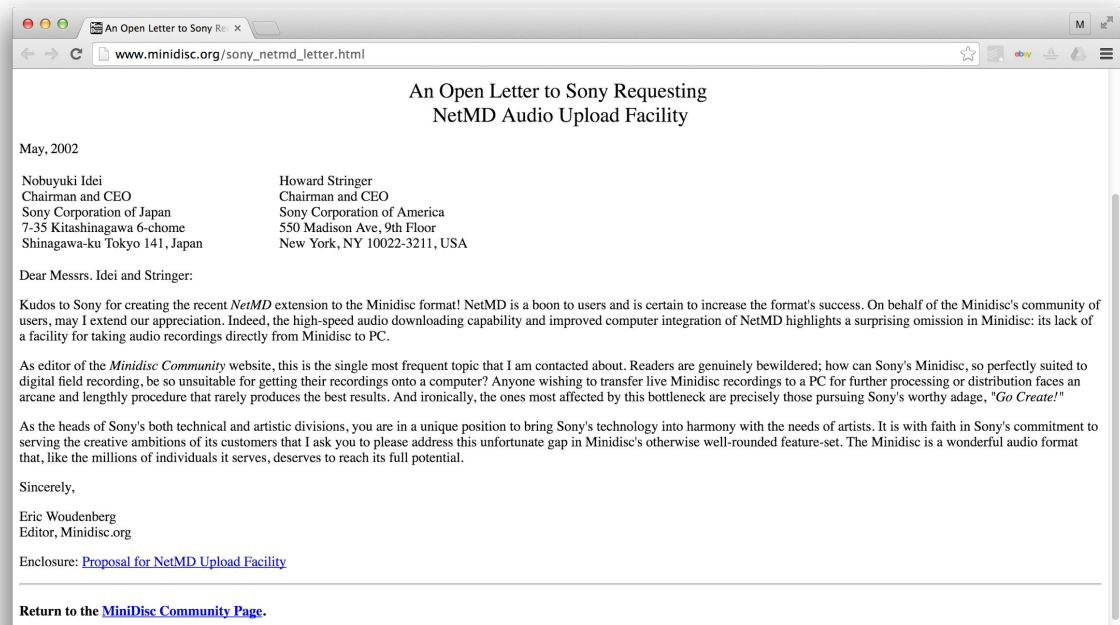
Perceptual audio coding is a compression technology based on imperfections in how the human ear perceives sound. An underlying data algorithm is programmed to discern and effectively filter out what it knows our ears cannot hear. MP3s are a popular type of this sort of audio encoding. The biggest difference between MP3 and ATRAC is that ATRAC is proprietary to Sony. So, when Sony decided to stop supporting this codec around 2008, MiniDisc owners got angry and worried over what would happen to their ATRAC data.



To understand consumer frustration with minidiscs, it's good to know a little history of the development of MD technology.

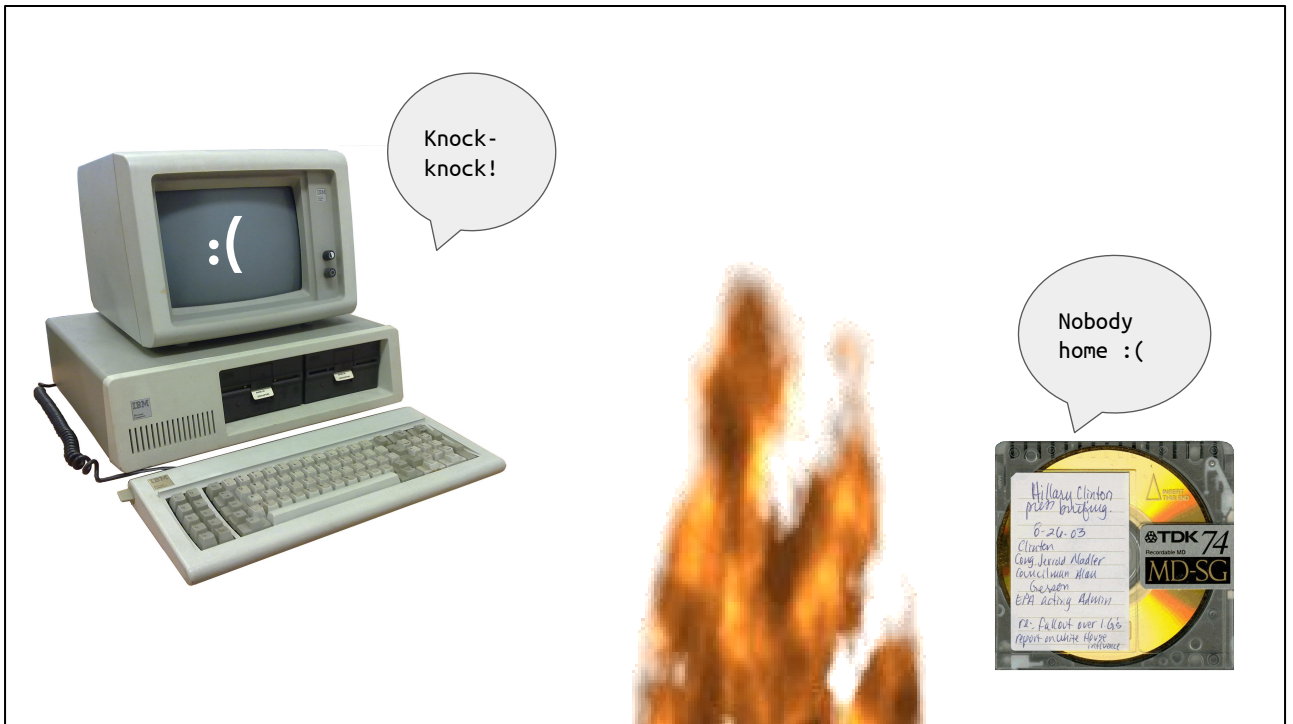
- In 1992, MD products were released to the consumer market. At this time, there was no way you could transfer ATRAC data from the Walkman to your computer and visa versa. Your only option was to transfer audio in real-time which is a huge pain in the neck, because they take a long time, and you lose all sorts of metadata, like track information.
- In 2001, Sony released a the MZ-N1 portable recorder equipped with so-called "NetMD interface" technology. NetMD players came equipped with a USB socket. In conjunction with proprietary music library management software (initially, this was OpenMG Jukebox, which was later superseded by SonicStage), users could transfer ATRAC data from their PC to an MD over a USB connection in faster than real-time.
- However, users could not transfer ATRAC data from a NetMD player to their PC. This one-way transfer protocol was deliberately built into NetMD technology to prevent commercial recordings from being freely distributed over networks. However, this also prevented someone like me from being able to transfer my own personal recordings into my own computer.
- In response to user frustration over these restrictions, in 2004, Sony released another update: The Hi-MD player, along with SonicStage 3.4. Both could distinguish between tracks that were personally recorded using the microphone input and those recorded for mass sale and distribution. Users

- were finally allowed to transfer personally recorded material from their MD player to their computer and convert the ATRAC data into WAV format.



Before Hi-MD players were released, some MD-R users got together to sign a petition in 2002 that basically called out MD technology as difficult, arcane and working directly against the needs of artists.

Learning about the history of MD technology was important to me as a audio archivist. For example, if I hadn't known about the transfer limitations of earlier players, I could have made the mistake of accidentally purchasing the wrong player from eBay that wouldn't allow a player > PC transfer. Believe me though, researching this information is exhaustive: it means pouring through lots of rage-inspired messages posted into tech forums in all-caps. But at the same time, I truly empathize with these users.



NYPR's archive owns SonicStage 4.3 and a Hi-MD player. Yet, when I first plugged in NYPR's MD Walkman into my work computer to try my first ATRAC data transfer, I could see the tracks listed, but when I clicked on Import, nothing happened.

I opened up Windows Explorer to see if the player was listed as a drive in the file tree, but nothing appeared, even though when I plugged in the player I would get that little USB connection notification in my toolbar. What I was experiencing here was a so called "audio/data firewall" that prevents me from being able to directly access my MD.

What frustrated me was that I was using all the "right" hardware and software to perform a perfectly legal, innocent MD > PC transfer. Yet, the software and I had no other way of getting in.



The last Sony MD Walkmen were sold in March 2013. In addition to this, development of new versions of SonicStage - including updates that allowed you to upload ATRAC data to your PC - ceased around 2008. So, even if you were to hoard the entire internet's inventory of used players and later versions of SonicStage, you would still be left having to keep around an earlier versions of Windows to use it. Also, Mac users are totally out of luck, because SonicStage is Windows-compatible only. So, you'd probably have to use an emulator, but then, who wants to deal with software licensing rules and restrictions? Not I.

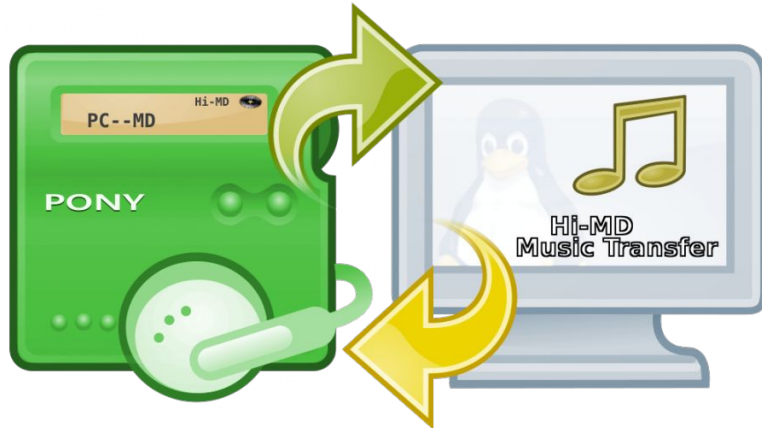
Now, I suppose I could have just gone over to my coworker's station and tried SonicStage on there to see if it worked. However, as an archivist, the general idea of using licensed software to complete an archival transfer makes me cringe.

mini THE SOLUTION



So, here's how I got by Sony DRM.

Linux-minidisc project



The “linux-minidisc project” was started in 2008 by a few developers who decided that they, and I quote, “...want[ed] to create a [way to perform] simple transfer[s]... for HiMD and NetMD Walkm[e]n which will run under [their] preferred operating systems and [that was] free of any of the annoying ‘features’ of the original software.” They also claim to have started this project as a way to collect and consolidate as much information about MD technology. The image you see above is their logo. To date, over 30 people have contributed code to this project.

Today I will be showing you how netmdpython works. Netmdpython is a collection of Python scripts created by the linux-minidisc project developers that enable users to transmit data directly from an MiniDisc Walkman to their PC over a USB connection.



So, what do you need to use netmdpython? The required hardware is both basic yet specific.

- You'll need an MD-R with stuff recorded onto it.
- You'll also need an MD-R player, but not just any player. Netmdpython will only work with certain Walkmans: the MZ-RH1 and MZ-M200 "Hi-MD" Walkmen, specifically, which were released around 2005/6. I was lucky enough that NYPR coincidentally owned the "right" players. But in the grand scheme of things, this requirement is quite limiting.
- Moving on: you'll also need a MD to PC USB cable.
- In terms of operating systems: netmdpython works on Linux, Unix and MacOS X, but will not work with Windows. The reason why it will not work in Windows is because one of the compiler libraries, libusb, has not been updated to be supported on current versions of Windows.
- I was lucky that NYPR's archive has a Mac. But again, this restriction is quite limiting, and kind of ironic, considering SonicStage was Windows-compatible only.

1. Download and install:
 - a. **Macports** to simplify installation of stuff below
 - b. **Python** to run scripts
 - c. **ffmpeg** to convert ATRAC > WAV
2. Set up bash **path variable**
3. From command line install **git-core** and **libusb**

For the actual set up of netmdpython, you need to setup a few things on your system.

- First, you will need to download and install Macports, which allows you to easily download and install the latest version of git-core and libusb from the command line.
- libusb is a C library that provides generic access to USB devices. It is intended to be used by developers to facilitate the production of applications that communicate with USB hardware.
- Once Macports is installed, you'll need to modify your .bashrc in your home-directory by adding "/opt/local/bin" to the PATH-environment.
- You'll also need to install the latest version of Python to run the transfer scripts, and ffmpeg to transform the stuff you transfer over from ATRAC into WAV.
- After all this stuff is setup, connect the Walkman to your Mac with a USB cable. Launch your terminal and navigate to the linux-minidisc folder.

There are great, more detailed instructions available through the netmdpython page. Our symposium git page will have a link out to this page in case you want to check that out later.

```
netmd — bash — 80x24
PB7-ARCH-MBPRO-DR53:netmd archives$ python lsmd.py
Disk (writable media)
Time used: 01:44:36+023 (100.00%)
63 tracks
000: 00:03:28+042 sp mono unprotected
001: 00:02:12+018 sp mono unprotected
002: 00:01:34+061 sp mono unprotected
003: 00:01:24+050 sp mono unprotected
004: 00:13:10+005 sp mono unprotected
005: 00:02:02+017 sp mono unprotected
006: 00:00:40+078 sp mono unprotected
007: 00:01:11+080 sp mono unprotected
008: 00:00:37+001 sp mono unprotected
009: 00:00:12+054 sp mono unprotected
010: 00:00:25+062 sp mono unprotected
011: 00:00:40+026 sp mono unprotected
012: 00:00:03+001 sp mono unprotected
013: 00:01:06+037 sp mono unprotected
014: 00:00:17+063 sp mono unprotected
015: 00:02:36+061 sp mono unprotected
016: 00:01:02+015 sp mono unprotected
017: 00:00:32+017 sp mono unprotected
018: 00:00:21+061 sp mono unprotected
019: 00:00:05+033 sp mono unprotected
```

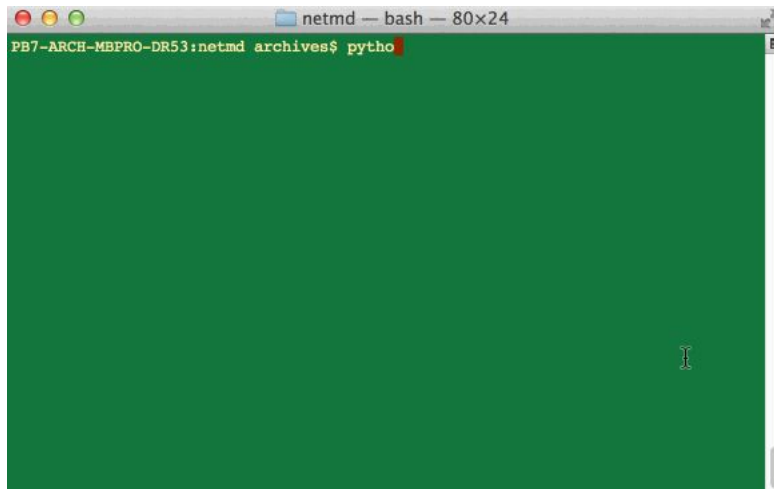
lsmd.py
List, count and
describe the
contents of an
MD-R

So, the first script I will show you is called `lsmd.py`, which is a script that allows you to get a directory listing of what's inside your MD-R. To launch it, I type into the terminal prompt "python, space, `lsmd.py`".

Once you press enter, a list will appear right below the command.

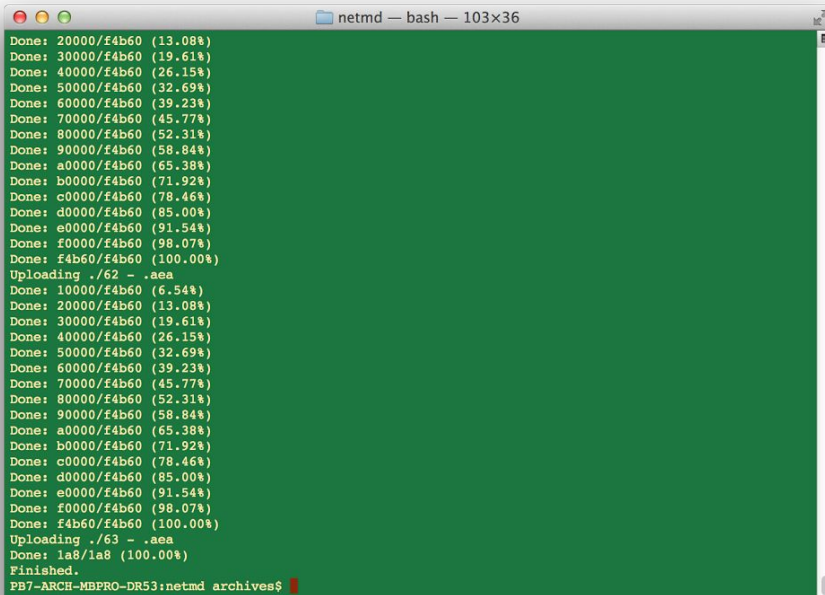
- Each recorded track appears in sequence and is assigned a track number.
- Next to that is the length of the track.
- Next to that is information about how the track was recorded: so here, "sp" refers to Standard Play, which means it was recorded at a high quality. Mono indicates it was probably recorded with a microphone through a single channel.
- Lastly, each track is listed as either protected or unprotected. What this means is that the script can differentiate between whether or not something was recorded personally using the player's mic input, rather than something commercially recorded in the studio. Netmdpython can only transfer unprotected tracks and not protected tracks.

Note that this command doesn't actually upload any audio data. However, it's an easy way to get a manifest of what's on your MD-R. I used to check my upload to make sure I got all 63 tracks. You could also export this list into text file and use it to populate fields in a catalog.

A screenshot of a terminal window. The title bar at the top reads "netmd — bash — 80x24". The terminal content shows a prompt "PB7-ARCH-MBPRO-DR53:netmd archives\$" followed by the command "pytho" with a red cursor at the end. The terminal background is dark green.

```
netmd — bash — 80x24
PB7-ARCH-MBPRO-DR53:netmd archives$ pytho
```

So, here is lsmd.py in action. Takes just a few seconds.



```
netmd — bash — 103x36
Done: 20000/f4b60 (13.08%)
Done: 30000/f4b60 (19.61%)
Done: 40000/f4b60 (26.15%)
Done: 50000/f4b60 (32.69%)
Done: 60000/f4b60 (39.23%)
Done: 70000/f4b60 (45.77%)
Done: 80000/f4b60 (52.31%)
Done: 90000/f4b60 (58.84%)
Done: a0000/f4b60 (65.38%)
Done: b0000/f4b60 (71.92%)
Done: c0000/f4b60 (78.46%)
Done: d0000/f4b60 (85.00%)
Done: e0000/f4b60 (91.54%)
Done: f0000/f4b60 (98.07%)
Done: f4b60/f4b60 (100.00%)
Uploading ./62 - .aea
Done: 10000/f4b60 (6.54%)
Done: 20000/f4b60 (13.08%)
Done: 30000/f4b60 (19.61%)
Done: 40000/f4b60 (26.15%)
Done: 50000/f4b60 (32.69%)
Done: 60000/f4b60 (39.23%)
Done: 70000/f4b60 (45.77%)
Done: 80000/f4b60 (52.31%)
Done: 90000/f4b60 (58.84%)
Done: a0000/f4b60 (65.38%)
Done: b0000/f4b60 (71.92%)
Done: c0000/f4b60 (78.46%)
Done: d0000/f4b60 (85.00%)
Done: e0000/f4b60 (91.54%)
Done: f0000/f4b60 (98.07%)
Done: f4b60/f4b60 (100.00%)
Uploading ./63 - .aea
Done: 1a8/1a8 (100.00%)
Finished.
PB7-ARCH-MBPRO-DR53:netmd archives$
```

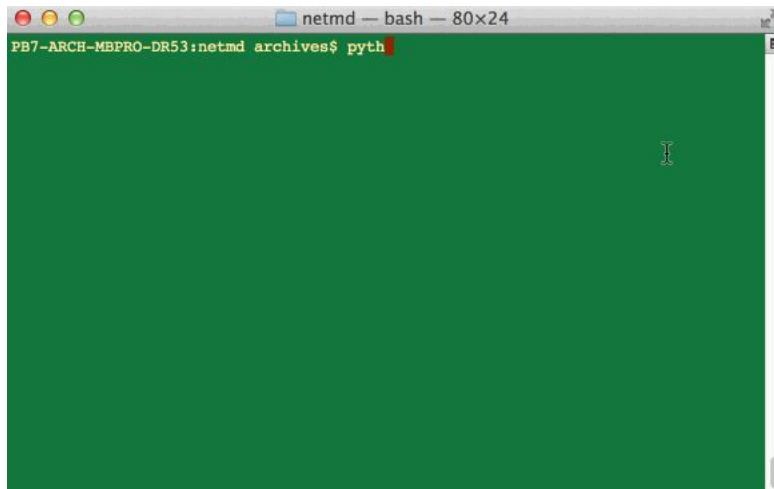
upload.py
Transfer ATRAC
data to PC. Each
track has an *.aea
extension.

Next, I will show you upload.py. This is the script that actually uploads data off the MD-R over USB onto your computer.

The screenshot here shows the last two tracks - 62 and 63 - that were transferred. Each track takes a few seconds to upload depending on how long it is. You can see in parenthesis the progress each track is making from start to finish. So this is kind of like, a text-based way of monitoring the status of your transfer.

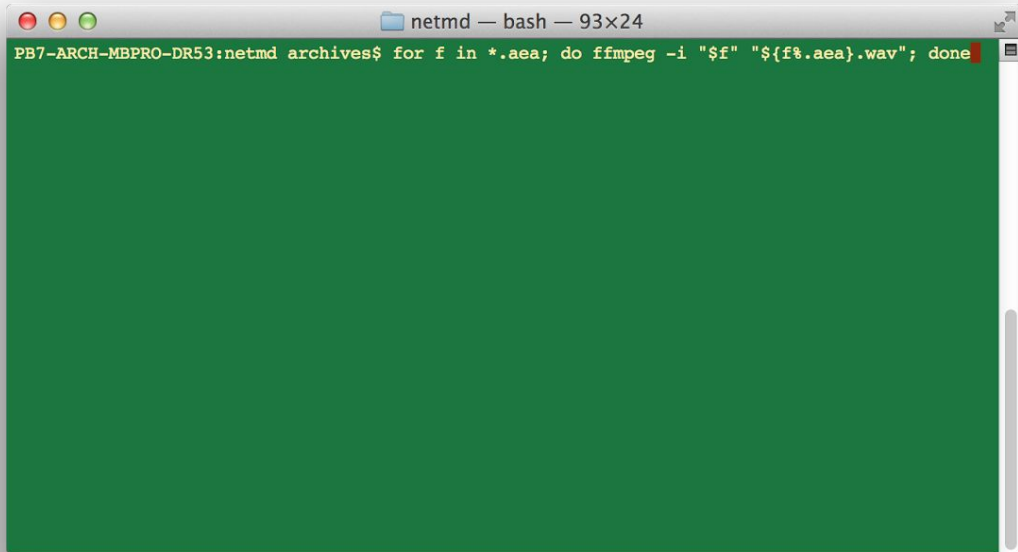
The tracks end up on your computer with an AEA extension. AEA just means ATRAC.

Something to note: when I first launched the upload script, it didn't complete the transfer. It got through about 50 tracks and then just stopped. So, I unplugged everything and restarted the system, and then, it worked! But, it wasn't apparent what had happened and I received no error message. And it's happened to me a few times since, even when I was extra careful to not bother the USB cable during the transfer. So, it's a little buggy.

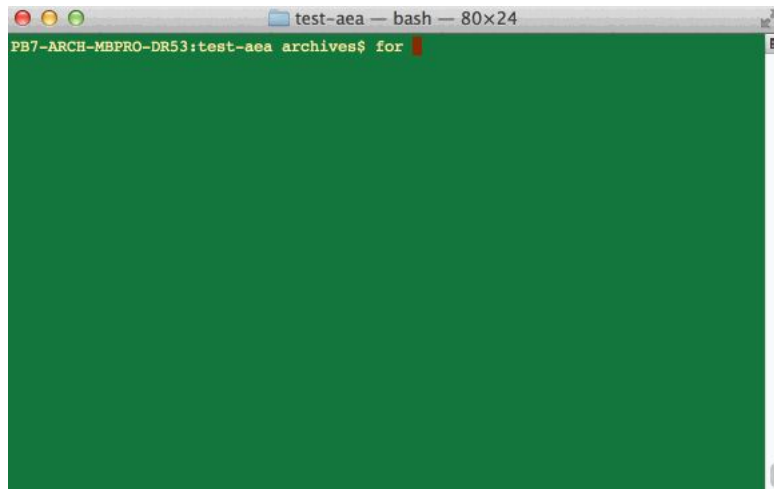


Here is a little video of me launching upload.py. As you can see, it runs through each track and provides a percent status.


```
for f in *.aea; do ffmpeg -i "$f" "${f%.aea}.wav"; done
```



So, that's basically it for the netmypython scripts. And now, you have a folder full of AEA tracks. What's great is that the people who developed netmdpython worked together with the people over at ffmpeg to make sure that users of netmdpython could use ffmpeg convert tracks into non-proprietary audio formats. So, what you see above is the ffmpeg command I would use to convert all AEA tracks in a folder into WAV format.

A screenshot of a terminal window. The title bar at the top reads "test-aea — bash — 80x24". The terminal content shows the prompt "PB7-ARCH-MBPRO-DR53:test-aea archives\$" followed by the command "for" and a red cursor. The rest of the terminal area is filled with a solid green color, likely representing redacted output or a placeholder.

```
PB7-ARCH-MBPRO-DR53:test-aea archives$ for
```

Here is the command running.



- Only works with 1 type of player
- Scripts are experimental
- Doesn't work on Windows OS :(
- Python + command line + Git + ... =
intimidating

So, that's what netmdpython looks like in action. What's great about it is it worked for NYPR! However, it is not the perfect solution for a number of reasons.

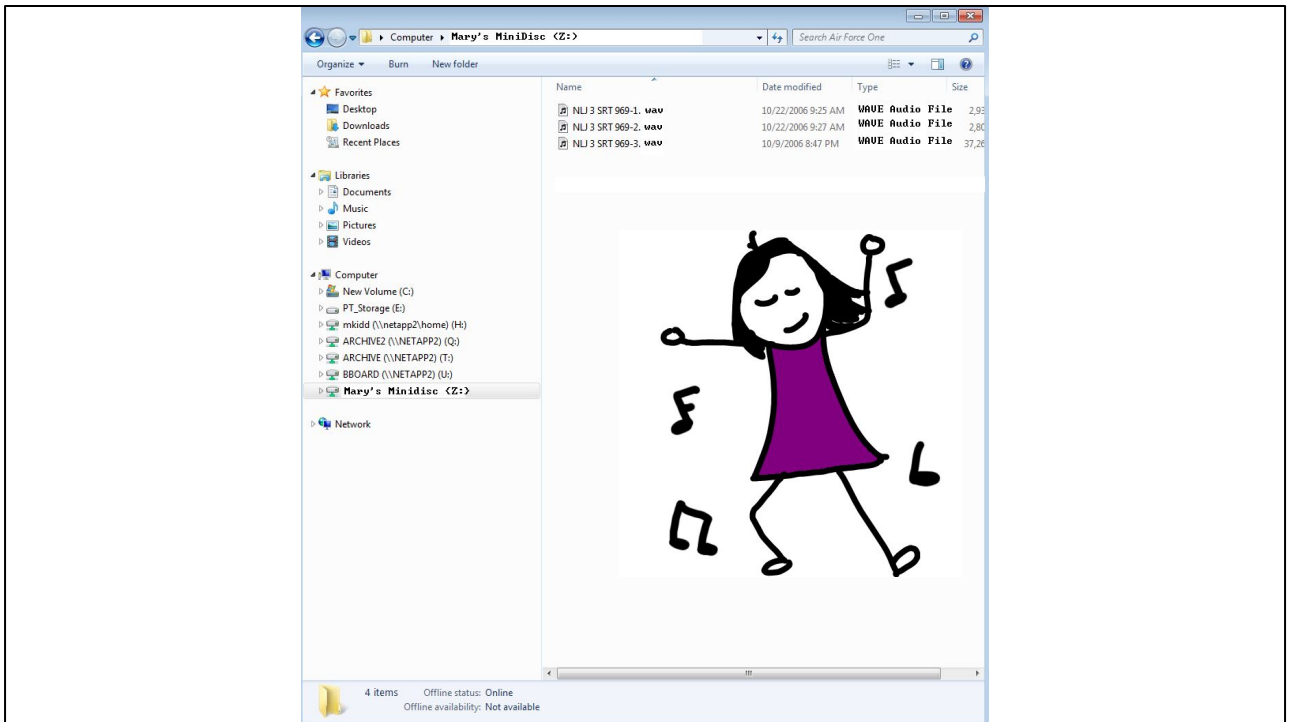
- As I mentioned before, the last MD Walkman was sold in March 2013. These scripts only work with 2 of the dozens of MD players released onto the market, so if you don't have the right player, you may have to troll eBay. Over time, though, these specific players may become rarer and more expensive.
- Also, these scripts only work with certain OS's and excludes Windows users. Lots and lots of people and organizations use Windows.
- These scripts are experimental. Meaning, there's no guarantee that they will work smoothly, or not break or erase something. One of the linux-minidisc's disclaimers is to use netmdpython with quote "toy media only". I obviously went against that and took a risk, with my mentor's permission of course. But, I must emphasize that this isn't a widely acceptable archival transfer practice. I personally haven't found an example of another archive using these scripts, and going at these things alone is a bit daunting.
- Implementing netmdpython can be intimidating for users who do not know what Python is, are not used to using and installing libraries from git, or using the command line.
- That being said, there is a linux-minidisc project mailing list and an IRC chatroom where you can ask questions.



To put things in perspective, we should talk about smartphones. Have you plugged in your smartphone into your computer by USB lately? Ever notice that when you peer into your smartphone using your computer, you see maybe 1 folder with just photos and nothing else? Most of us here today are carrying one of these things in our pockets. My smartphone contains a lot of personal recordings. Voice memos, some containing voice memo interviews with NYPR staff for my NDSR report. And these recordings are mine.

To get my voice memos off my phone, I have to use iTunes to “sync” my phone data to my computer. Without iTunes, I can’t see or interact with them in any other way. iTunes is just another SonicStage. My voice memos hide behind the same audio data firewall protecting my Taylor Swift MP3s.

Reporters at NYPR use voice memos to record from the field. It’s the standard now. But what will happen in 50 years and we start using something else besides an iPhone or an Android? What’s scarier is MDs were a market failure, yet the archive ended up with thousands of MD-Rs. iPhones are ubiquitous, a market success. And one day, we may have to break down the doors of our smartphones or our clouds and demand what’s rightfully ours.



Thank you!

I will take any questions now.

I have set up a table with a laptop with netmdpython setup, along with a player and some actual MD-Rs from the NYPR archive [over there](#) for anyone who wants to check out MiniDisc players, see what MD-Rs look like. I even have a CD3 in case you want to see what an MD-R is not.