Guides

Getting this funky beast working needed quite a few new skills, so here's how to do those!

- Packaging a PostScript Printer Description (PPD) file as a signed Windows 10 x64 printer driver

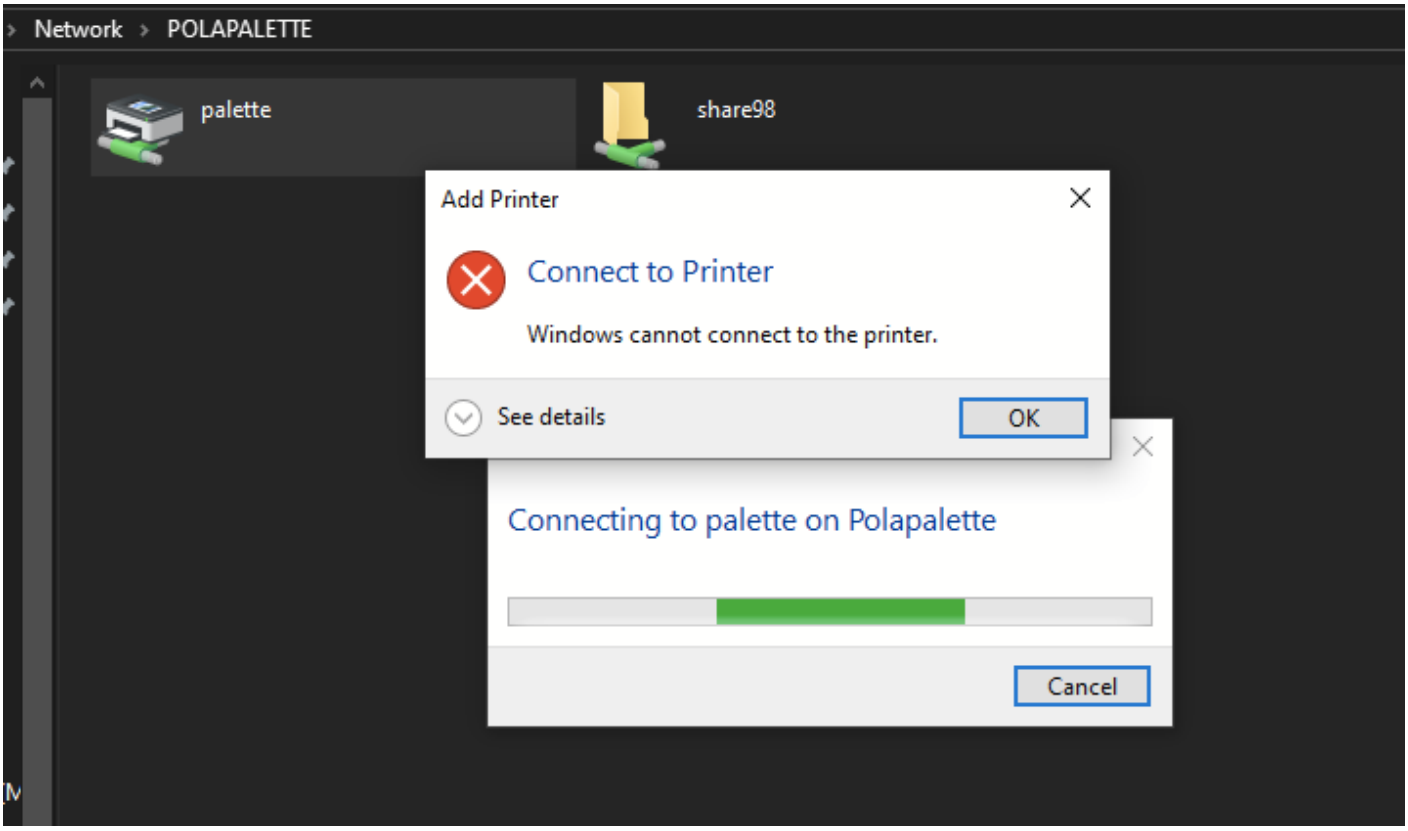
Packaging a PostScript Printer Description (PPD) file as a signed Windows 10 x64 printer driver

Why?

After getting my Polaroid CI-5000 film recorder working as a PostScript-compatible printer on Windows 98SE, I wanted to use that on my Windows 10 editing PC.

I foolishly expected this would *just work* using Windows' printer sharing, like my considerably more modern (but still ancient) HP LaserJet.

NOPE.



Turns out you need a working driver for both the server and client systems, and the newest I had was for Windows NT4.

So, here's a guide on how I turned my film recorder's NT4 driver into a shiny new Windows 10 one!

Disclaimer:

> **Follow this guide at your own risk!**
> **I am not responsible for any mistakes, viruses, explosions, dead film recorders, or anything else!**

I have done my best to ensure this guide is as accurate as possible, and these instructions did work for me!

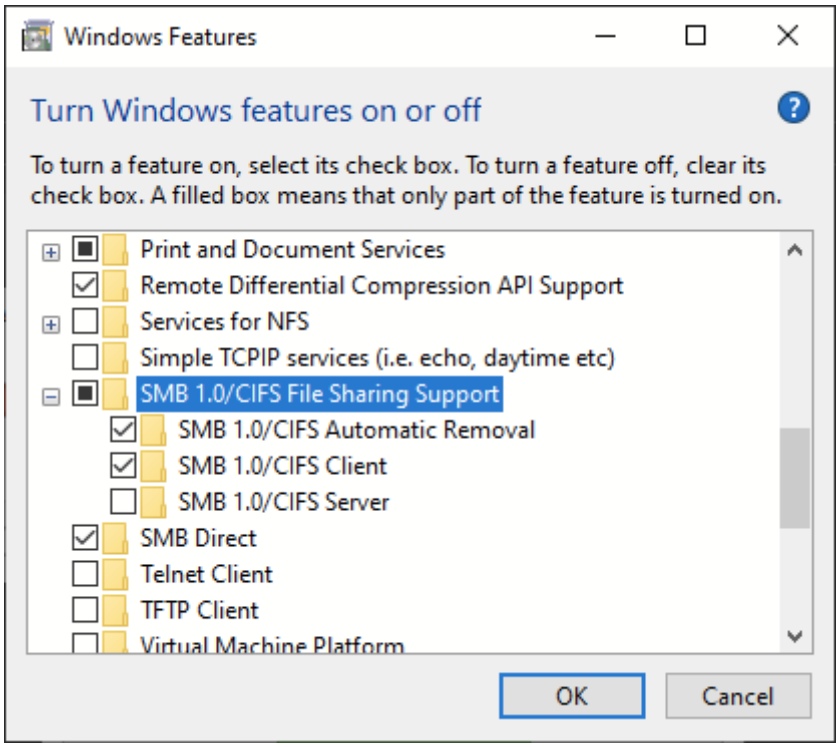
However, I am not a professional Windows driver developer, and only just about understand what I'm doing :)
Therefore, some things may be wrong or poorly explained!

(if I made a mistake though, please contact me and I'll fix it!)

Finally, I will not provide any support, so be warned!

Prerequisites:

- A Windows 10 machine with SMB1.0 client support enabled

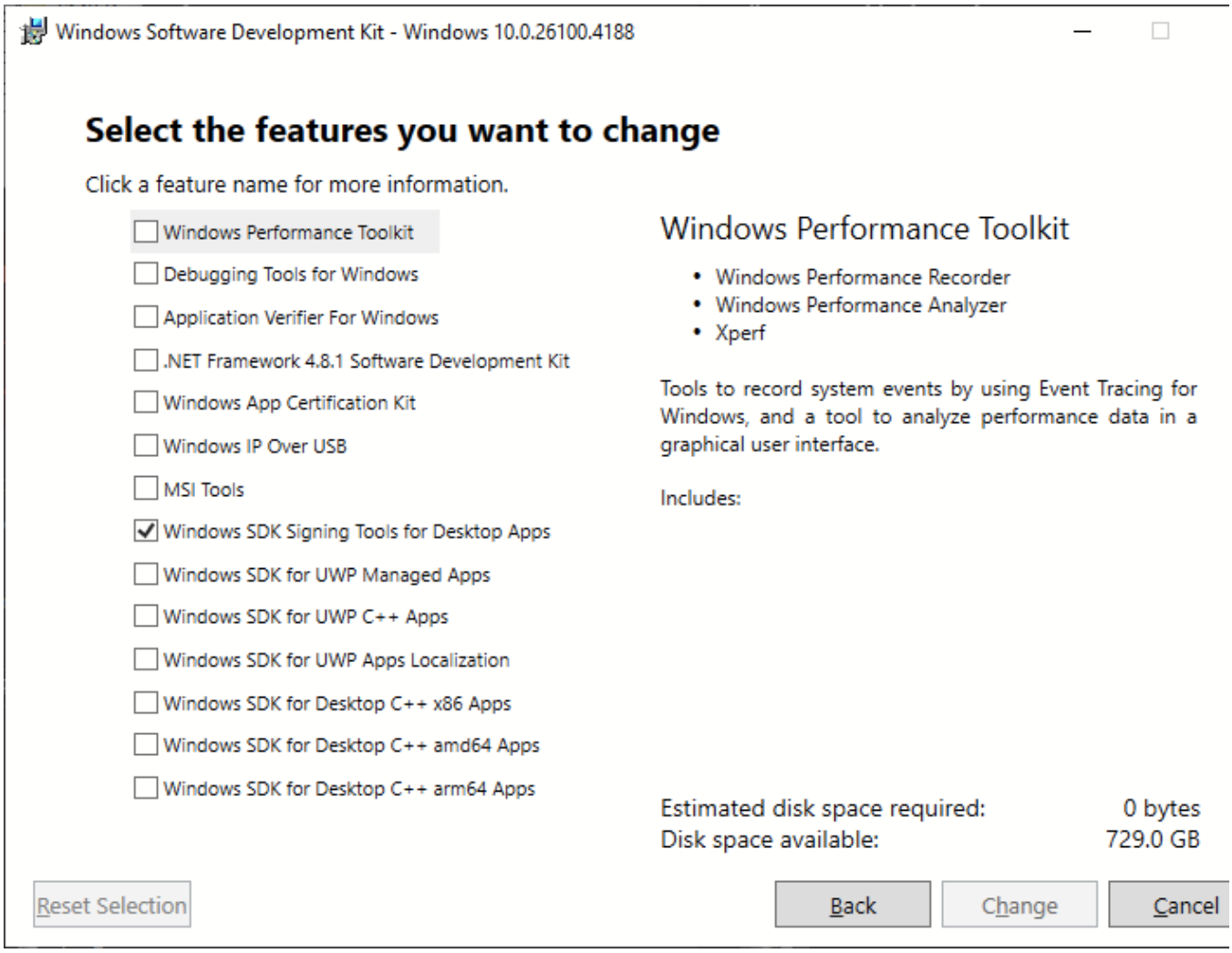


DO NOT ENABLE THIS ON AN INTERNET-CONNECTED MACHINE!!!

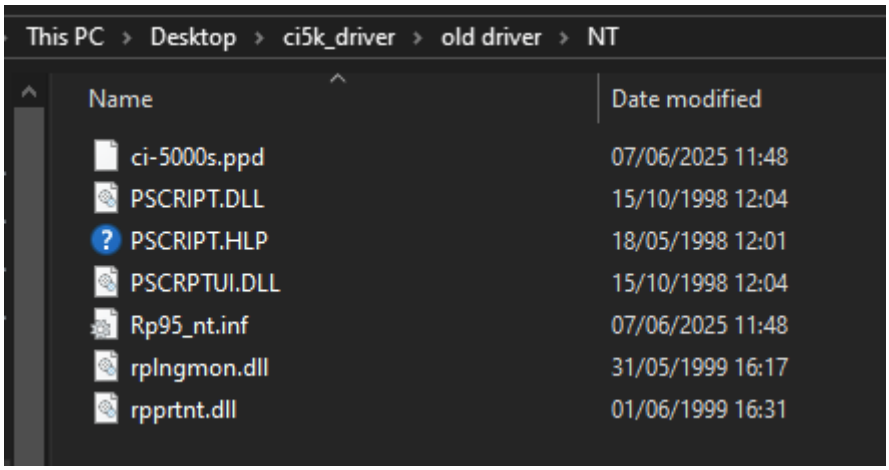
SMB1.0 is *highly* insecure, and opens up your machine to all kinds of fun exploits/ransomware!
ONLY do this if you're 100% sure that your security is up to snuff! (isolated VLAN etc)
There's a reason it's disabled by default :)

- both the Windows SDK and the Windows Driver Kit (WDK) installed on your machine

> You'll only need the Windows SDK Signing Tools for Desktop Apps installed from the SDK, the rest is unnecessary



- the appropriate Windows NT driver and .PPD file for your film recorder (or printer) copied to your Windows 10 machine



> I grabbed these out of my installation of RasterPlus95 on a Win98SE VM.
> If you're using this too, copy the entire `C:\Program Files\Graphx\RasterPlus95\CrossPlatform\NT` folder to a shared directory, as some of the important files are hidden!

Now, you've got all of that together! Awesome!

but first...

How does any of this work?

PostScript is almost like a universal language for printers and printer-like devices, dating right back to the 80s.

Every PostScript device has a PostScript Printer Description (or PPD) file, to tell systems how to use the various features of that device.

It's pretty neat!

I'm oversimplifying a LOT here, but even modern Windows machines have a generic PostScript compatible printer driver built right in, compatible with *any* PostScript device (in theory).

Backwards compatibility, hell yeah!

However, modern Windows won't just run any old PostScript driver...

It's gonna need:

- to be 64-bit compatible
  (10 cannot use 32-bit or 16-bit drivers on a 64-bit system!)
- to be digitally signed
  (makes sure your driver isn't just five viruses in a trenchcoat, and makes it work on 10)
- to have an .inf file in a modern format
  (the spec for .inf files has changed a LOT since Windows NT!)
- to use the modern `PSCRIPT5.DLL` driver
  (instead of NT's 32-bit `PSCRIPT.DLL`)

So, let's get started with cooking one of those up for my film recorder

1: Modernising the driver's .INF file

Straight out of RasterPlus95, our Windows NT .inf file looks like this;

```
[Version]
Signature="$Windows NT$"
Class=Printer
Provider="Graphx"
LayoutFile=layout.inf

[Manufacturer]
"Graphx, Inc."

[Graphx, Inc.]
"Digital Palette CI-5000S"=ci-5000s.ppd

[PSCRIPT]
PSCRIPT.DLL
PSCRPTUI.DLL
PSCRIPT.HLP

[PSCRIPT_DATA]
DriverFile=PSCRIPT.DLL
HelpFile=PSCRIPT.HLP
ConfigFile=PSCRPTUI.DLL
NoTestPage=1

[DestinationDirs]
DefaultDestDir=66000

[ci-5000s.ppd]
CopyFiles=@ci-5000s.ppd,PSCRIPT
DataSection=PSCRIPT_DATA
```

I modified that by trial and error, using this .inf for GhostPDF as a template, to more closely resemble a modern driver.

Compare this working one to the NT one above!
(don't worry, I'll explain everything that I added or changed!)

```
[Version]
Signature="$Windows NT$"
Class=Printer
Provider="Graphx"
```

```
ClassGUID={4D36E979-E325-11CE-BFC1-08002BE10318}
CatalogFile=CI-5000S.cat

DriverVer=06/07/2025,1.0.0.1

[Manufacturer]
"Polaroid"=Polaroid,NTamd64

[Polaroid.NTamd64]
"Digital Palette CI-5000S" = ci-5000s.ppd, "PRINTENUM\LocalPrintQueue"

[PSCRIPT]
PSCRIPT5.DLL
PS5UI.DLL
PSCRIPT.HLP

[PSCRIPT_DATA]
DriverFile=PSCRIPT5.DLL
HelpFile=PSCRIPT.HLP
ConfigFile=PS5UI.DLL
NoTestPage=1

[DestinationDirs]
DefaultDestDir=66000

[ci-5000s.ppd]
CopyFiles=@ci-5000s.ppd,PSCRIPT
DataSection=PSCRIPT_DATA

[SourceDisksNames]
1=%Disk1%,,,""

[SourceDisksFiles]
ci-5000s.ppd =1
PSCRIPT5.DLL =1
PS5UI.DLL =1
PSCRIPT.HLP =1

[DefaultInstall]
CopyINF=CI-5000S.inf
```

The first changes I made are in the `[Version]` block;

- removed non-existent `layout.inf`
  I couldn't actually find this to tell you what it does/did, sorry!
- added `ClassGUID`
  this one tells modern Windows that this is specifically a printer driver
- added a `CatalogFile`
  this is where the driver's digital signature will be stored!
- added the `DriverVer`(sion)
  this verifies that the driver was written whilst the signing certificates were valid

Then, onto the `[Manufacturer]` block;

- changed "GraphX, Inc" to "Polaroid"
  for some reason it would NOT work with the original manufacturer name, sorry GraphX!
- added NTAMD64 decorated model sections (Polaroid, NTAMD64)
  to tell Windows this is a 64-bit driver

then the hardware ID block, `[Polaroid, NTAMD64]`;

- added the hardware ID `"PRINTENUM/LocalPrintQueue"`
  to tell Windows to use this driver specifically for local print queues

Also in this block is the .PPD description file (in my case ci-5000s.ppd), that's staying unmodified :)

Then, the `[PSCRIPT]` and `[PSCRIPT_DATA]` blocks
These tell Windows which PostScript base driver to use

- updated the original `PSCRIPT.DLL`, `PSCRIPTUI.DLL`, and `PSCRIPT.HLP` with their modern `PSCRIPT5` equivalents
  Those are `PSCRIPT5.DLL`, `PS5UI.DLL` and a new `PSCRIPT.HLP`

> I found these new base drivers in the Windows 10 print driver store, under `"C:\Windows\System32\spool\drivers\x64\3"`
>
> You'll need to copy these to your working directory with these .inf and .ppd files :)

I then added location pointers to the file.
These tell Windows where to find all the files this .inf references :)

- the `[SourceDisksNames]` block simply says 'look in the same folder as the .inf file'
- the `[SourceDisksFiles]` block specifies the location of every file so far referenced is in that current folder

Finally, the `[DefaultInstall]` block tells Windows to copy this .inf file to its' driver store.

Got all of that?
Nor do I really, but I've linked all the helpful websites I used at the bottom! No ChatGPT here!

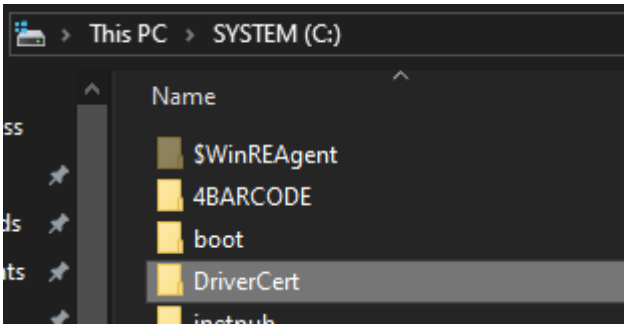
and just like that, we have a valid .inf file ready to be packaged and signed!

2: Generating a self-signed cert to sign the new driver

Before we digitally sign *anything*, we're gonna need a certificate to sign it *with*.

I wasn't going to use this driver outside of my own PC, so instead of buying a proper code signing cert, I generated one using some nifty PowerShell commands :)

First up, we're going to need a folder to put our newly generated certs in.
I created one at the root of my C drive called `"C:\DriverCert"`



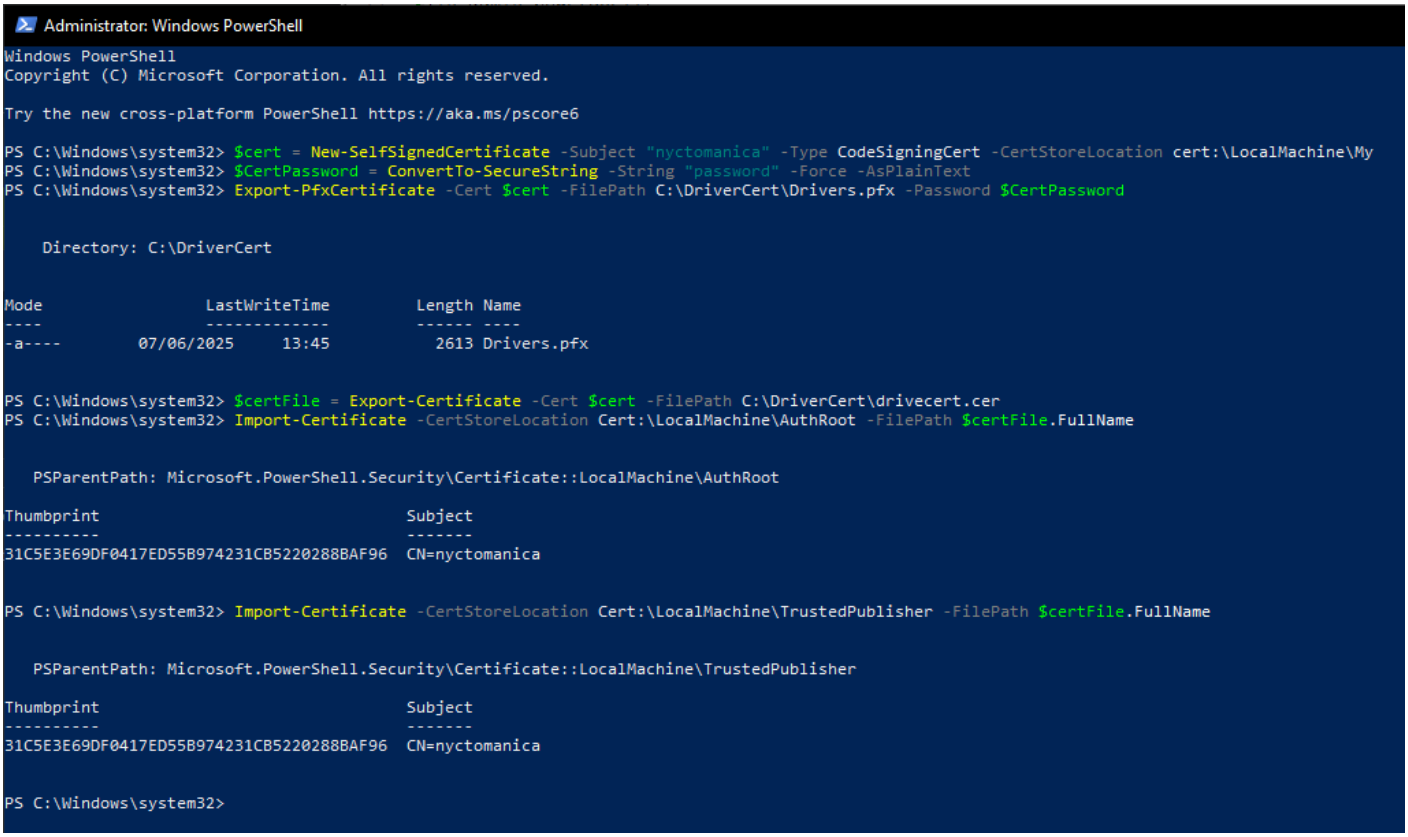
Then, in an admin PowerShell prompt, run the following commands;

```
$cert = New-SelfSignedCertificate -Subject "nyctomanica" -Type CodeSigningCert -CertStoreLocation
cert:\LocalMachine\My
$CertPassword = ConvertTo-SecureString -String "password" -Force -AsPlainText
Export-PfxCertificate -Cert $cert -FilePath C:\DriverCert\Drivers.pfx -Password $CertPassword
$certFile = Export-Certificate -Cert $cert -FilePath C:\DriverCert\drivecert.cer
Import-Certificate -CertStoreLocation Cert:\LocalMachine\AuthRoot -FilePath $certFile.FullName
Import-Certificate -CertStoreLocation Cert:\LocalMachine\TrustedPublisher -FilePath $certFile.FullName
```

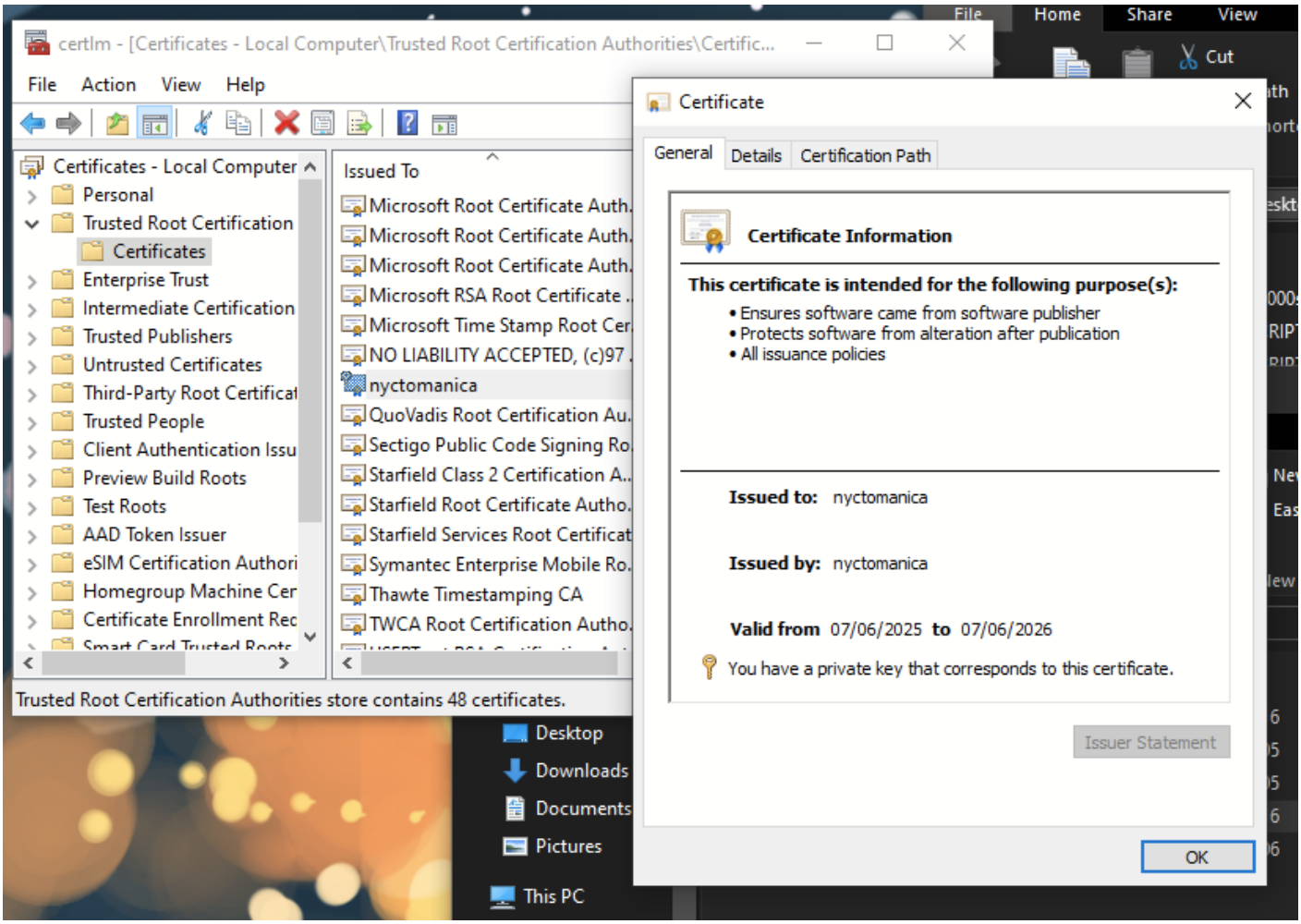
These generate a new self-signed code signing certificate in the new folder I just made on C, then install it into the system's Root Certificate Store.

> You'll probably want to change the Subject to not be "nyctomanica", and the password notto be "password"

and boomski, we have one lovely self-signed cert!



Check to make sure that's in the root store by opening `certlm.msc` using Run, then look for your certs' subject name :)



Sweet!

Looks like we're ready to package up our driver with a security catalog(ue)!

3: Packaging the driver, and creating a security catalog(ue)

Before we create a security catalog for our driver, it's probably a good idea to put all of the driver files into a nice new folder.

> My nice new folder is at `C:\Users\nycto\Desktop\ci5k_driver\NT`, but yours will be different :)

I'm now going to use a tool called `Inf2Cat`, part of the Windows Driver Kit we installed earlier :)

This checks to make sure that our driver's .inf file is valid, then creates a new (unsigned) security catalog for our driver, ready to be signed!

In a non-admin command prompt, `cd` to the location of the `Inf2Cat` tool.

> For me, this was at `"C:\Program Files (x86)\Windows Kits\10\bin\10.0.26100.0\x64"` , but for you this may be different

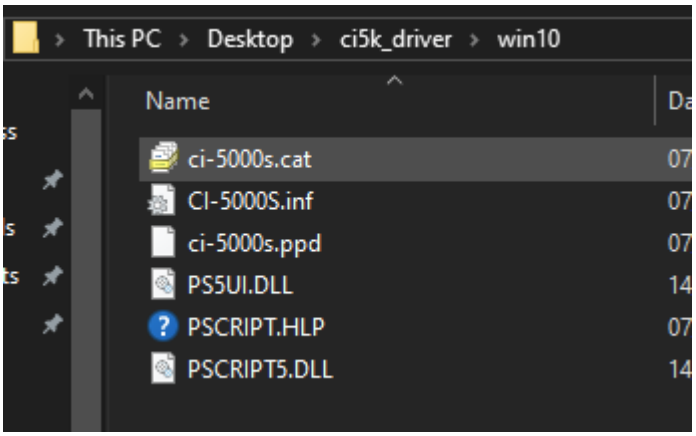
then, run `Inf2Cat.exe /driver:C:\Users\nycto\Desktop\ci5k_driver\NT /os:10_X64`

```
C:\Program Files (x86)\Windows Kits\10\bin\10.0.26100.0\x86>Inf2Cat.exe /driver:C:\Users\[redacted]\Desktop\ci5k_driver\NT /os:10_X64
.............................
Signability test complete.

Errors:
None

Warnings:
None

Catalog generation complete.
C:\Users\[redacted]\Desktop\ci5k_driver\NT\ci-5000s.cat
```

> yes, I ran cmd *inside* Powershell, because I already had it open :)

With a bit of luck, you'll be rewarded with a lovely new .cat file, and no errors or warnings!
(I say this because oh boy, I got PLENTY of warnings until I properly figured out the syntax of that inf file!)

But, there it is, now ready to be signed!



so, let's go sign that cat :)

4: Signing the driver

Finally, we've almost got a working driver for that pesky film recorder (or printer)!

But for it to be usable by Windows 10, it needs to be digitally signed to verify that it's safe. (back in the Old Times this wasn't actually necessary, but Windows' driver security has been tightened up a LOT since then)

So, we're going to use a tool from the Windows SDK called `signtool` to do just that!

In a non-admin command prompt (or `cmd` in PowerShell), make sure you're in the same folder as `Inf2Cat`

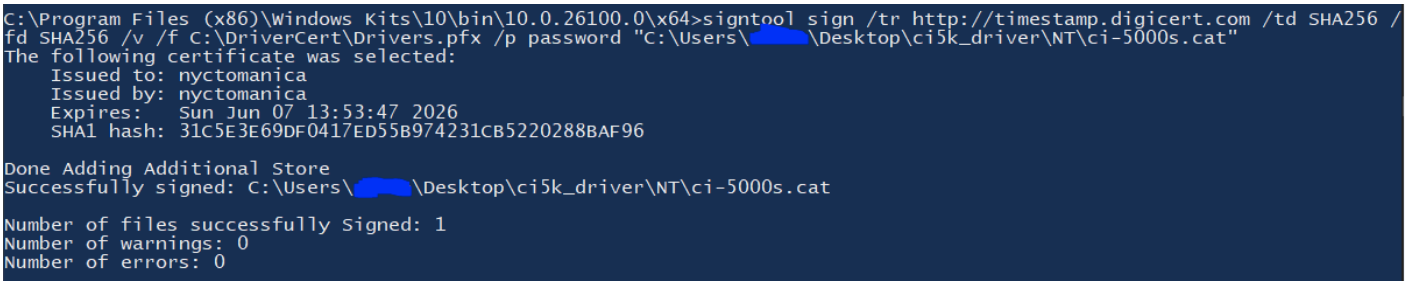
> again, this was at `"C:\Program Files (x86)\Windows Kits\10\bin\10.0.26100.0\x64"` , but for you this may be different

Then, run the following command to sign that .cat file;

```
signtool sign /tr http://timestamp.digicert.com /td SHA256 /fd SHA256 /v /f C:\DriverCert\Drivers.pfx /p password "C:\Users\nycto\Desktop\ci5k_driver\NT\ci-5000s.cat"
```

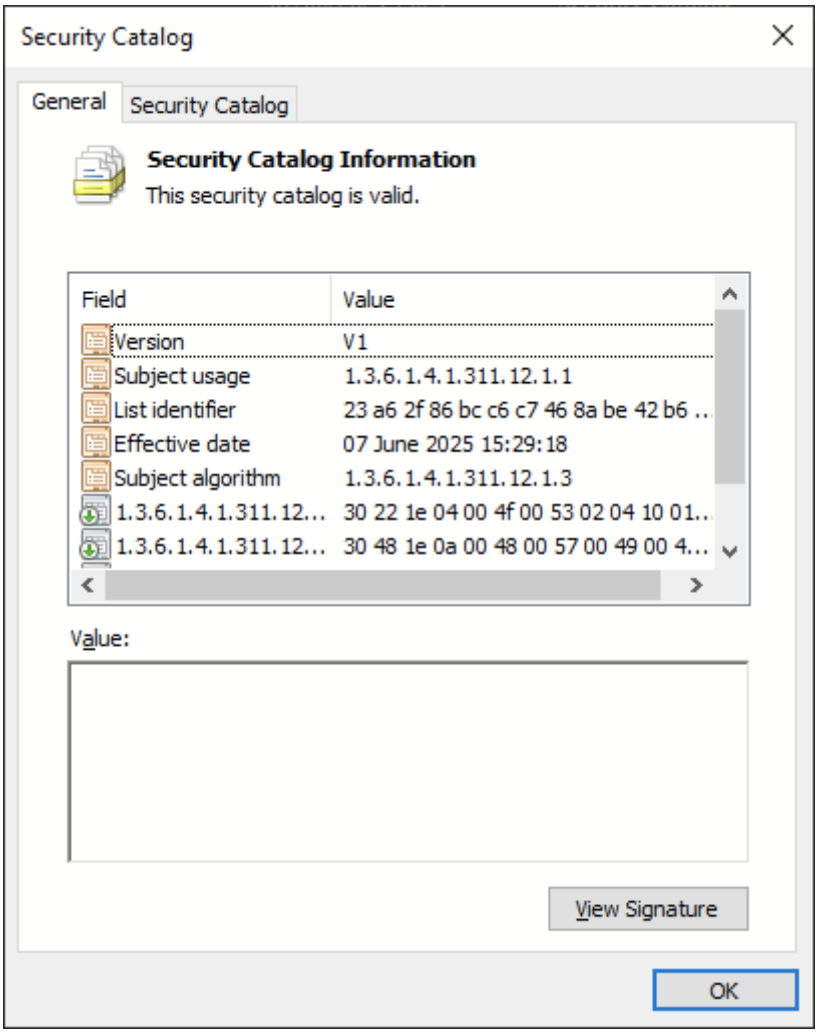
This signs the security catalog using SHA256 signatures and the certificate we generated earlier, as well as an accurate timestamp from Digicert.

You should be rewarded with something like this!



Then, to verify that the catalog is indeed signed and valid, just open the .cat file.

If you see something like this, you're ready to install your shiny new driver!!



> if it's not valid, double check that your self signed cert is indeed in the system trust store, and try again :)
> It'll work if everything's in order!

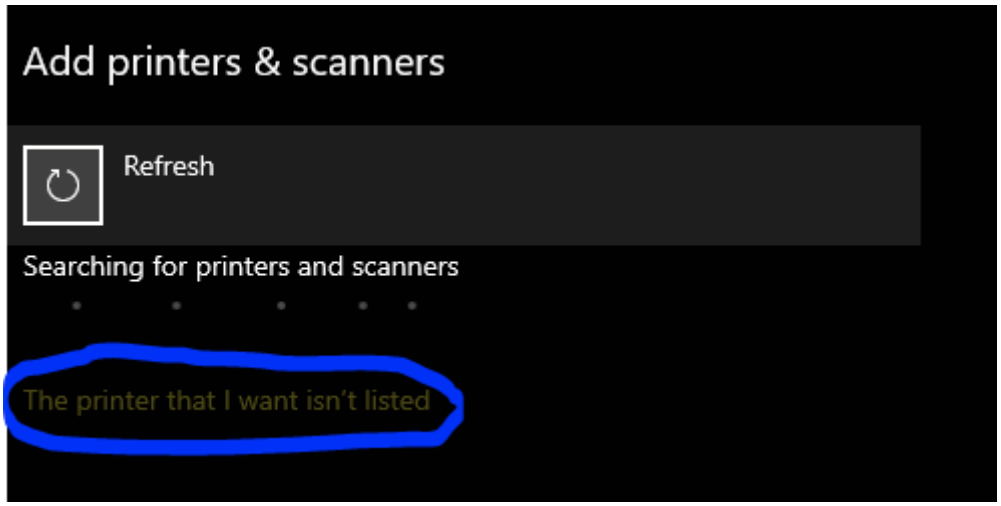
So, let's give it a go!

5: Testing out the new driver!

Now armed with our nicely packaged and signed driver, it's time to give it a shot!

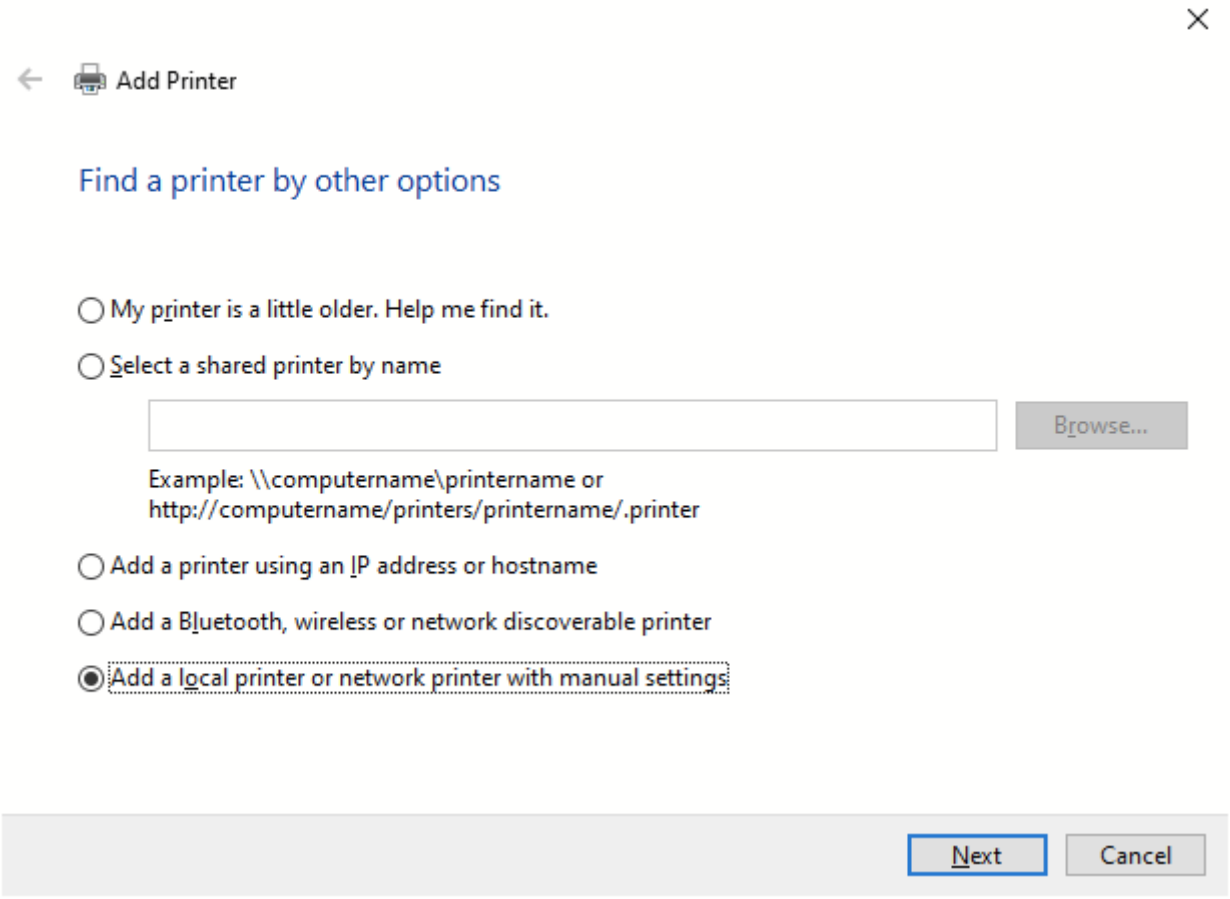
Open up Settings, then Devices, then Printers and Scanners.

Hit "Add a printer or scanner", then wait a sec until "The printer that I want isn't listed" appears.

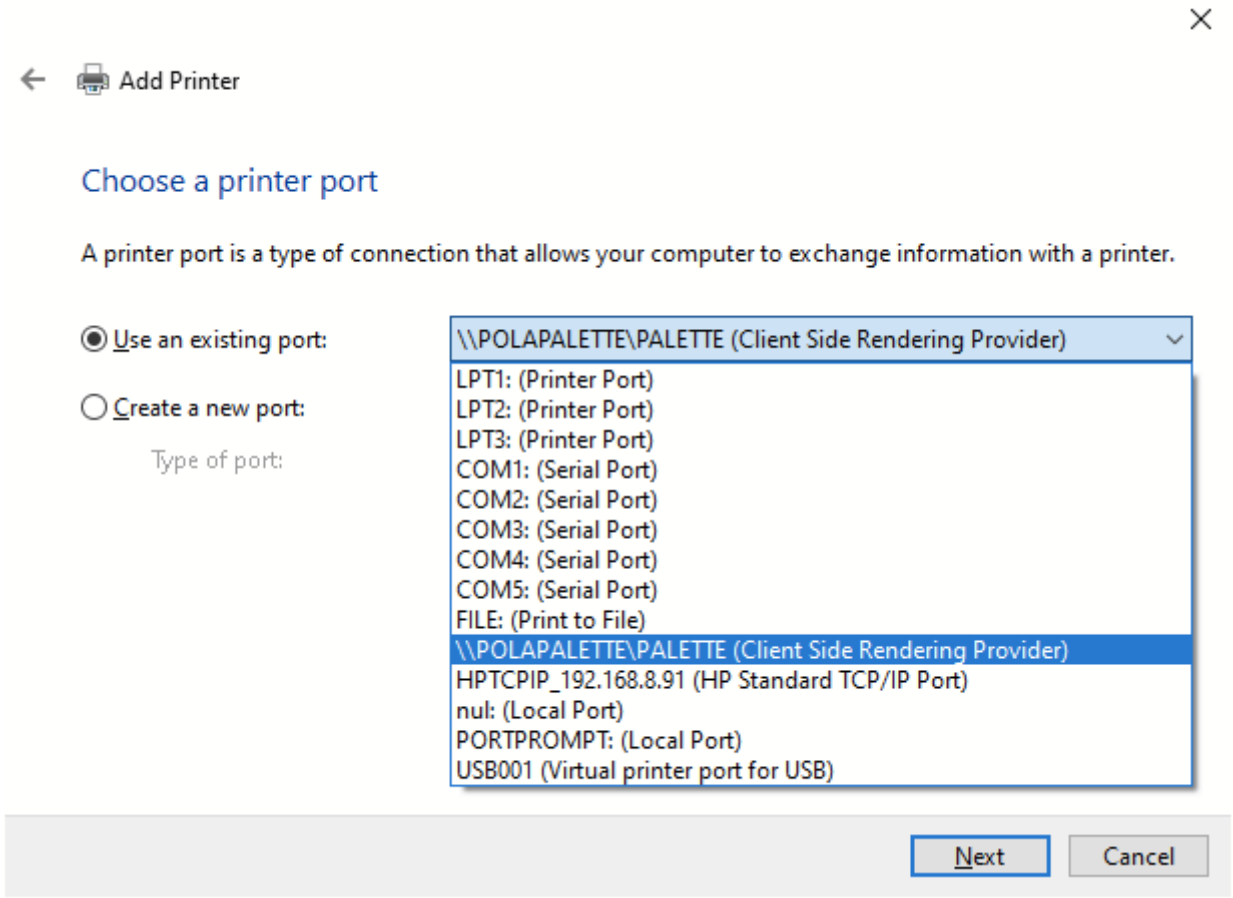


Click that, and you'll get the old style "Add a printer" dialogue!

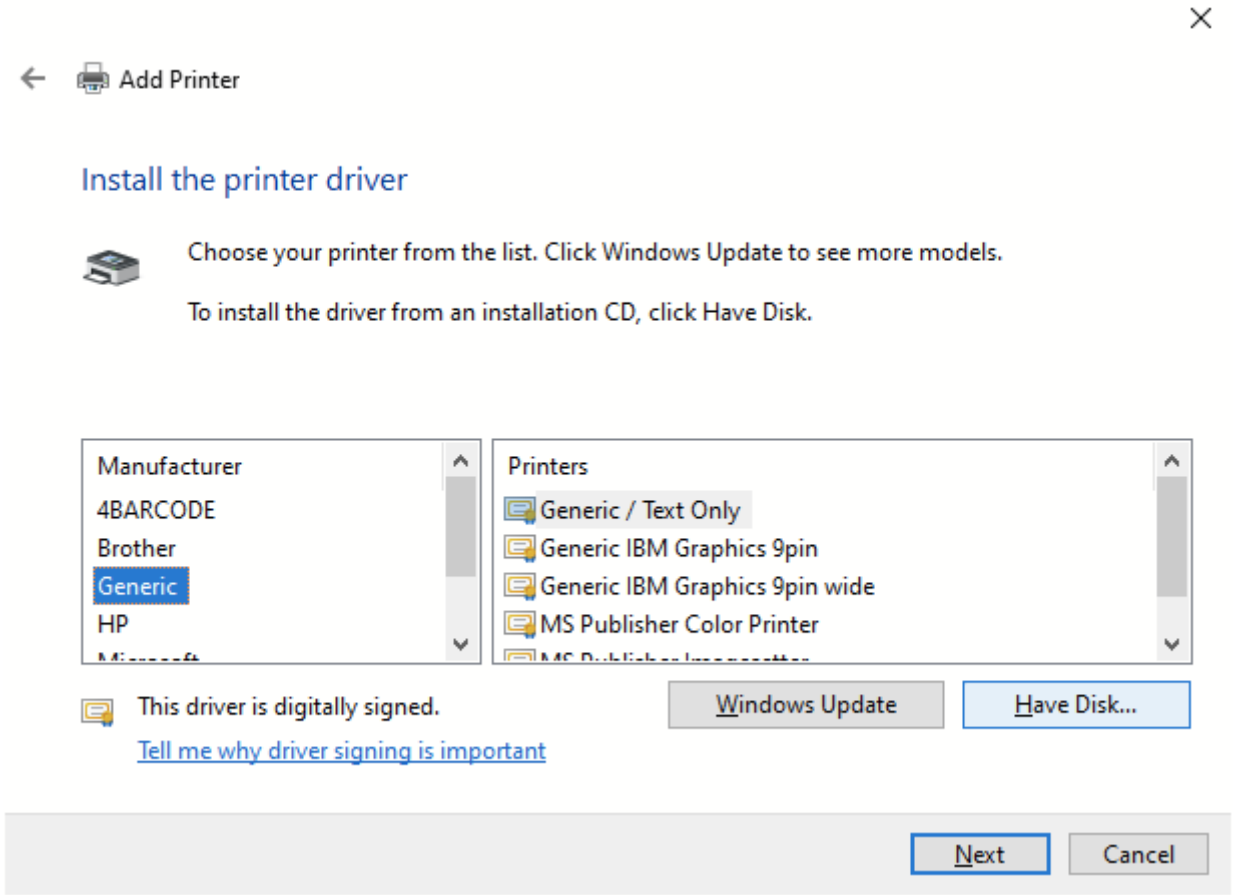
We're going to want the bottom option, "add with manual settings"



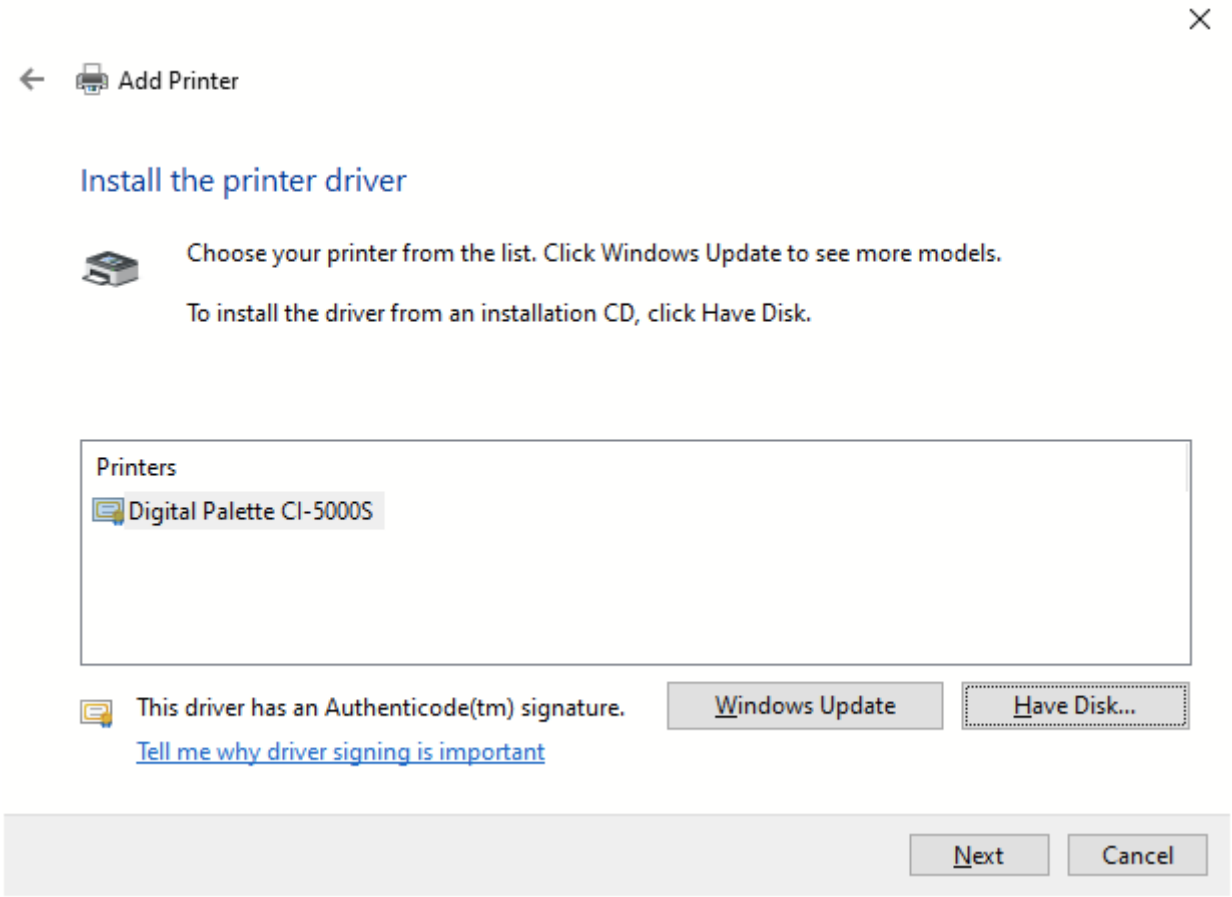
Hit "Next", then find the printer you want to add



"Next" again, then hit "Have Disk"



Navigate to the new driver folder, select the .inf file, "Open", "OK", and you should be presented with something a bit like this!

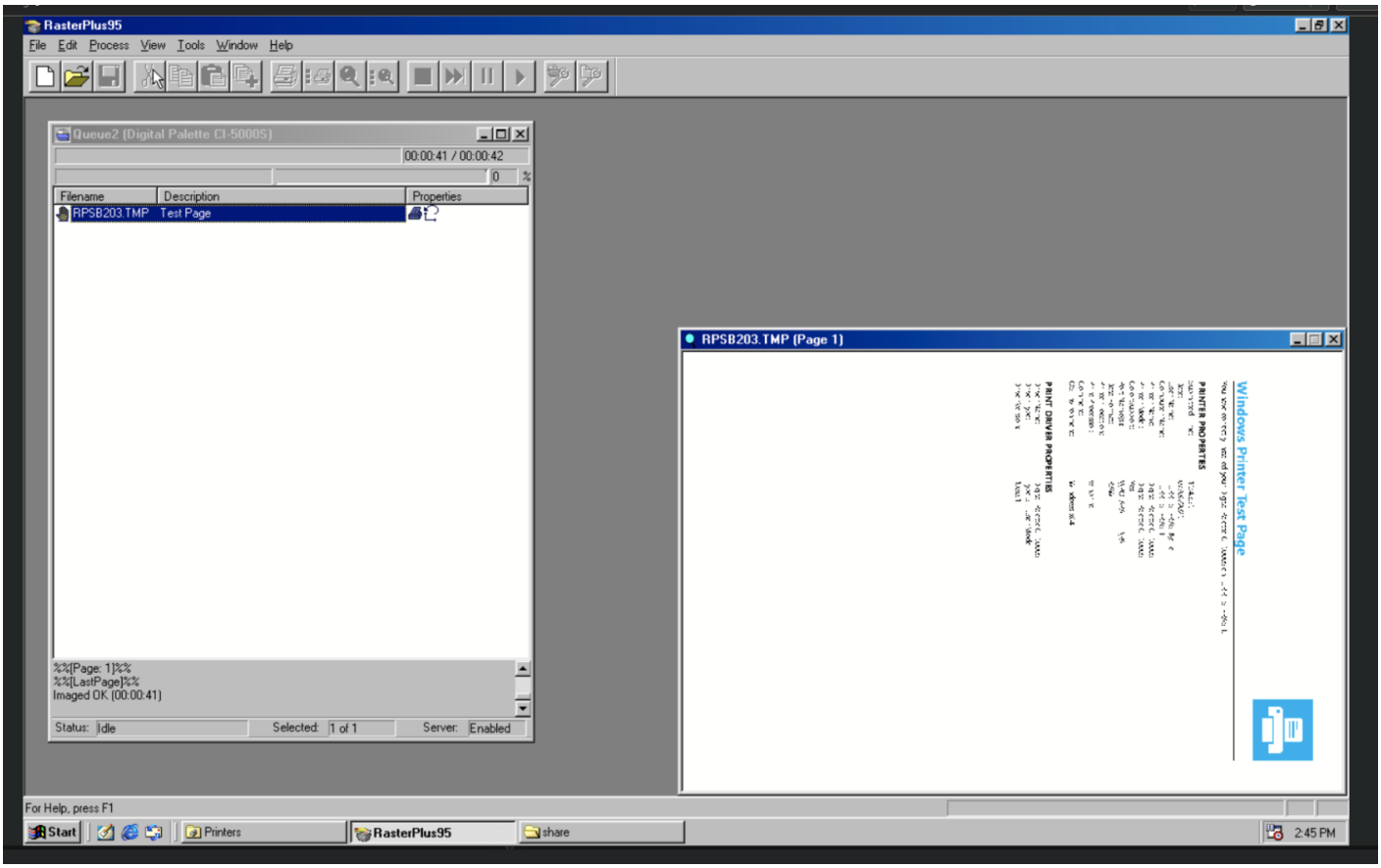


The signature worked, and the driver is accepted as valid! Sweeeet!

hit next again, then print a test page!

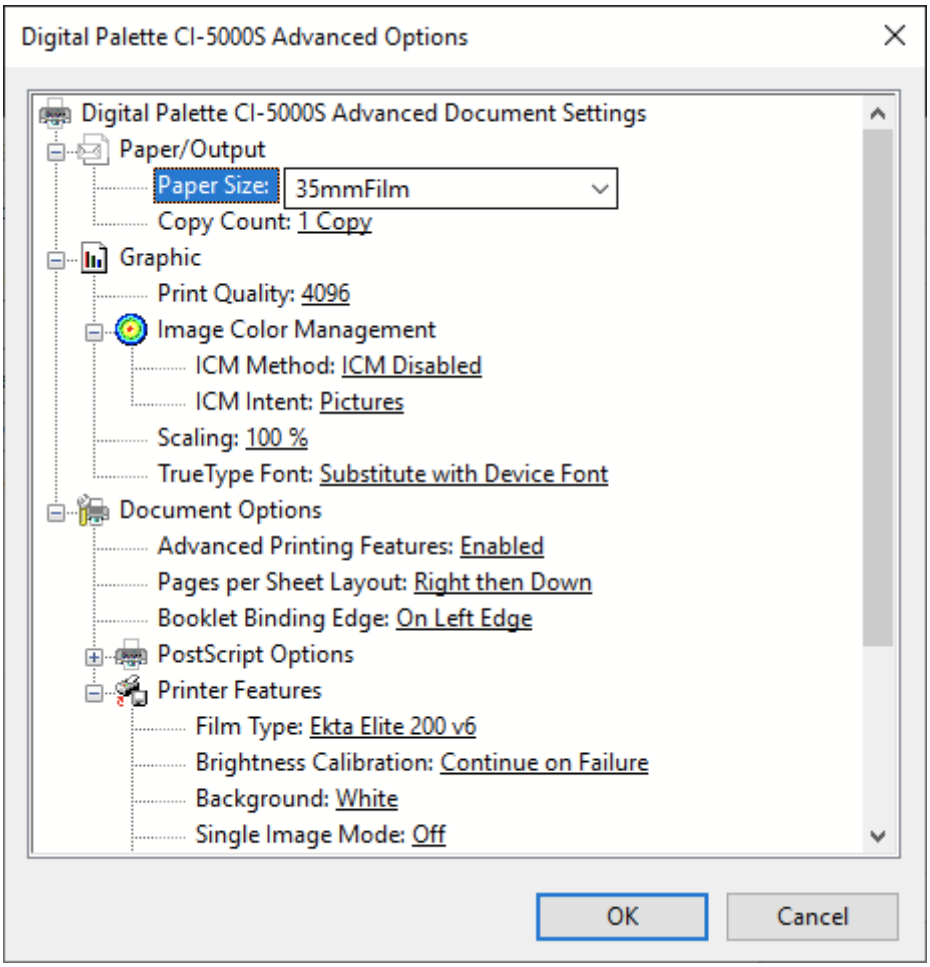
aaaannnnddddd after Win98 gets it's act together...

SUCCESS!



Tadaaa!
One highly cursed screenshot of the Windows 10 printer test page, ready to be recorded to 35mm film in RasterPlus95!

Not only that, but we also have all of the options for the film recorder available in the Windows 10 printer driver!!



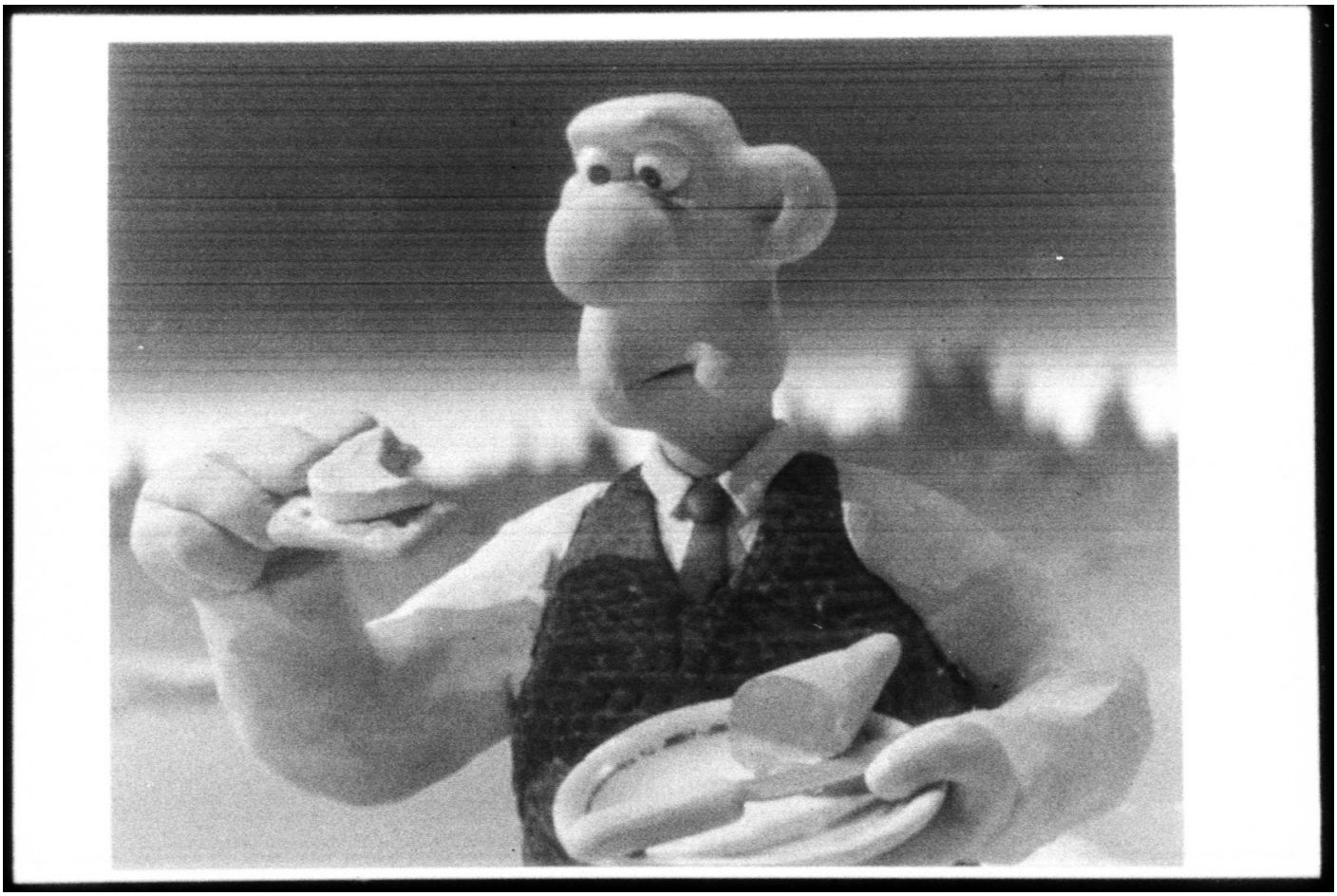
If you're interested (of course you are!), I'll be doing a full project post soon about getting my film recorder running!

so keep your eyes peeled over at nycto.io!

Happy printing! (or film recording!)

I hope that this may help someone one day :)

and even if it doesn't, here's a scan of some 35mm Kentmere 400 exposed in *crisp 4K* with the Polaroid Digital Palette to brighten your day!
say CHEESE!!



or should that be Camem-bert, Gromit?

(oh lordy, this machine needs some work!)

References:

Here's some of the many resources I found super useful during this adventure!

1. https://superuser.com/questions/1496842/windows-10-how-to-use-a-ppd-file-for-a-network-postscript-printer
2. https://stackoverflow.com/questions/19520540/signed-ghostscript-postscript-print-driver/19585322#19585322
3. https://github.com/plangrid/ghostpdl/blob/master/lib/ghostpdf.inf
4. https://woshub.com/how-to-sign-an-unsigned-driver-for-windows-7-x64/
5. https://community.osr.com/t/missing-sourcediskfiles-sourcedisknames-in-inf-file/35942

and of course we cannot forget the great Phil Pemberton, with his significantly more up-to-date versions of RasterPlus95 than my recorder came with, and many other resources for these

wonderful machines!

His site is at https://www.philpem.me.uk/code/filmrec/start
(and if you're reading this, HELLO! I've got a few floppies and things you may be interested in, please get in touch!!)