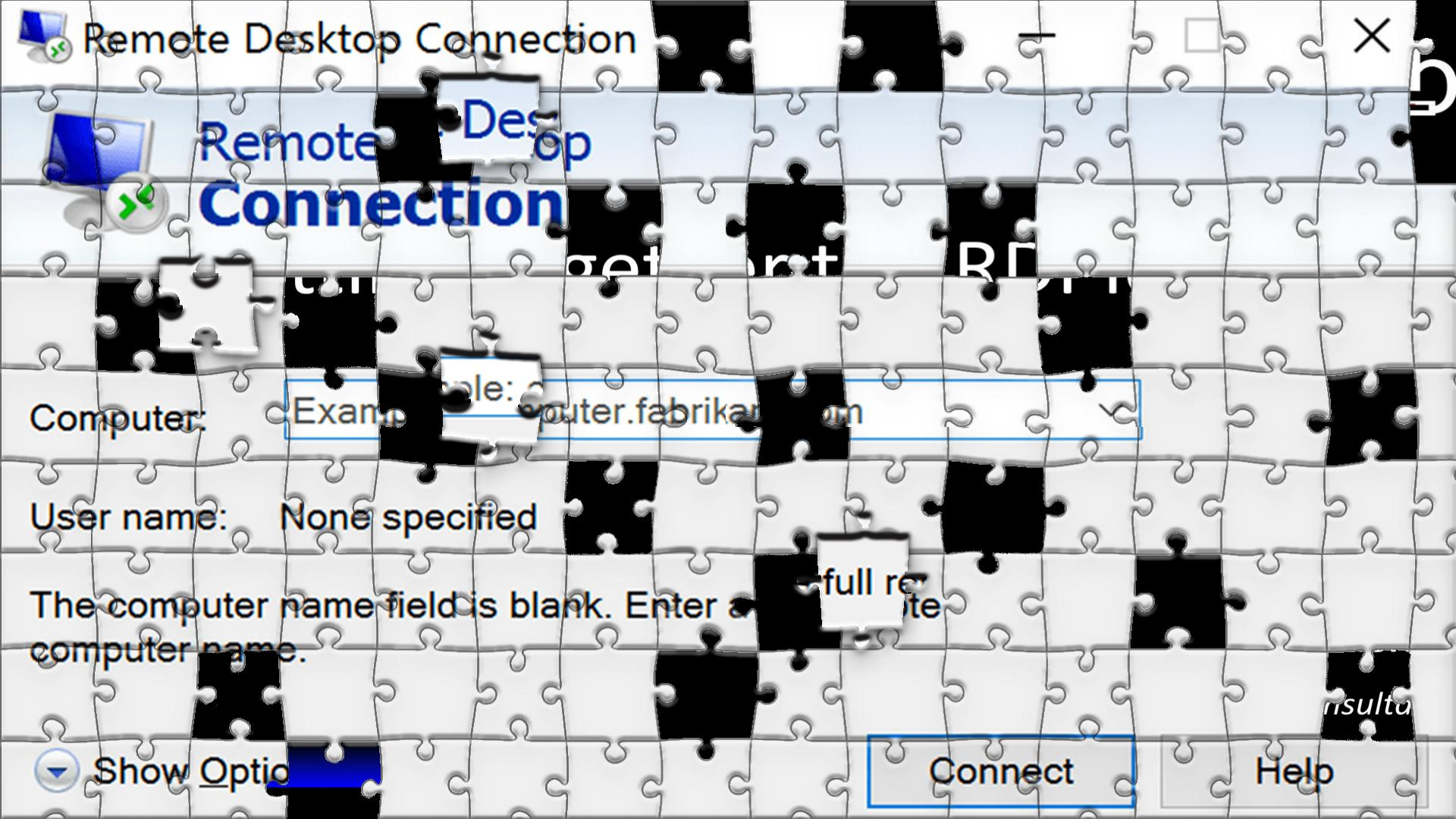

Remote Desktop Connection
Remote Desktop
Connection
Computer:
User name:
None specified
The computer name field is blank. Enter a
computer name.
Show Options
Connect
Help

Putting Together the RDPieces

Brian Moran

Consultant

BriMor Labs

November 18, 2020

A Brief List of Topics

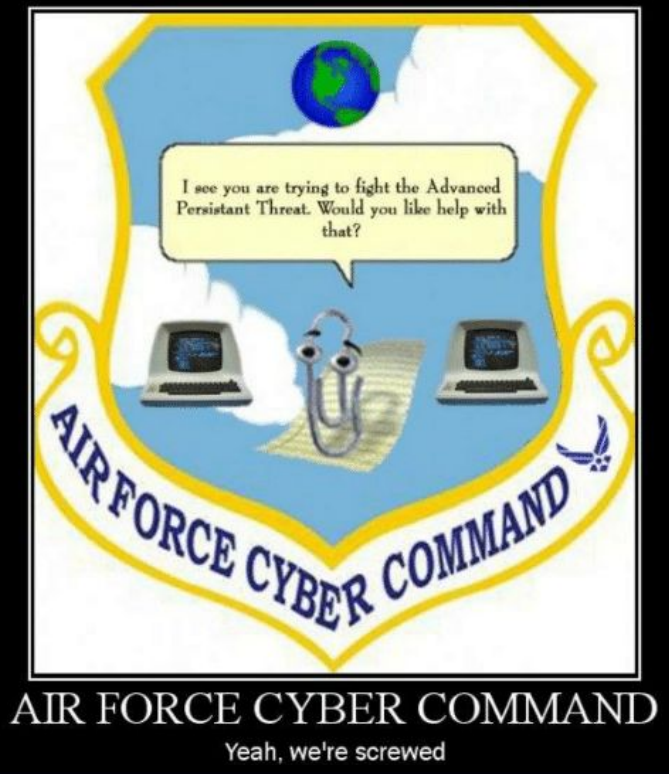
- RDP - WTF?

  (YOU: But Brian, we don’t really see much of this right meow)

  (ME: Perhaps, but this is why you should care)
- Evidence
- Research
- Stuff with Things
- Profit?

I Feel So Seen

- Hello, my name is Brian Moran
- 13+ years Air Force career
  - 17ish years mobile exploitation & DFIR focus
  - Started BriMor Labs in 2014
    - Very happy since!



I Feel So Seen (Cont)

- You may know me from a variety of things, but I am very proud of the #DFIRFitin2020 challenge that was organized with the help of Kat Hedley (@4enzikat0r)
  - You can still join us! Details at https://www.dfirfitin2020.com
- Throughout 2020, our #DFIRFit4Good events have raised over $10,000 for charity!!
- And, yes, there will be a 2021 #DFIRFit challenge
  - Well, if we make it to 2021

What is this RDP Thing?

- “Remote Desktop Protocol (RDP) is a proprietary protocol developed by Microsoft, which provides a user with a graphical interface to connect to another computer over a network connection”
  - This means someone can do stuff with things on another computer, whether it is in the next room or halfway around the world

What Does That Have To Do With My Investigations?

- Lateral movement in an environment
- Remote connection(s) to known/suspected malicious systems
- Unauthorized access
- Ransomware investigations

How I Got Interested in This Topic

- Working what seemed to be a typical ransomware case
  - YARC
- This particular attacker actually cleaned up after themselves
  - Cleared Event Logs
  - Cleared “Recent” data

- This made answering the usual questions (who, what, how, when, data access, data exfil, etc, EXTREMELY difficult)

How I Got Interested in This Topic

- Fortunately, the attacker did not clean up the RDP Bitmap Cache files
  - Since didn't have much else to go on, this was at least evidence of "something had happened"

WTF is RDP Bitmap Cache?

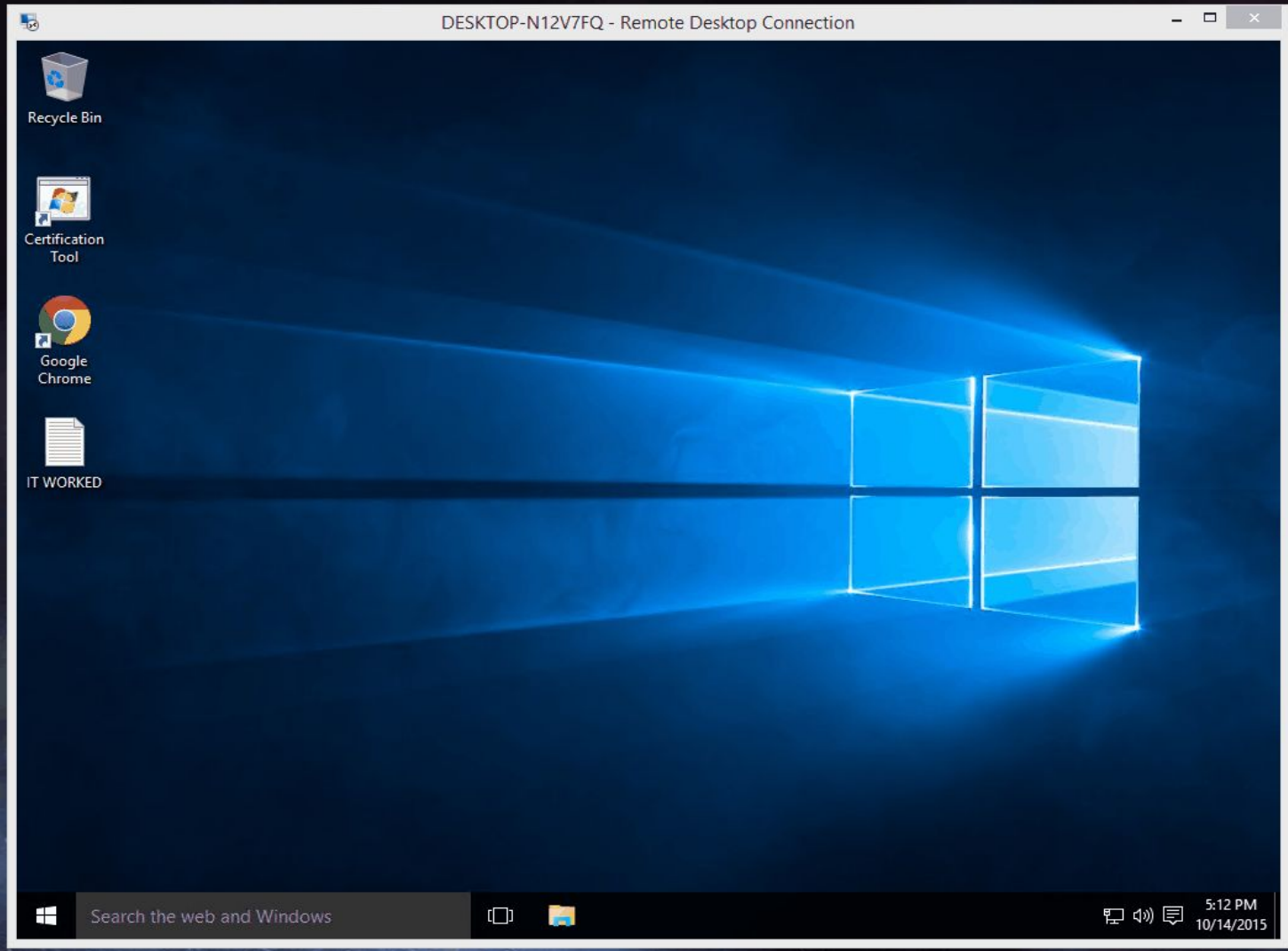
- Let's visit the source (https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/2da3e165-d1ba-4b65-8909-7a0f7f858d69)

"A Persistent Bitmap Cache is a store that contains bitmap images that were sent to the client by using the Cache Bitmap (Revision 2) Secondary Drawing Order ([MS-RDPEGDI] section 2.2.2.2.1.2.3). Unlike the Bitmap Caches described in section 3.2.1.13, Persistent Bitmap Caches are not bound to the lifetime of a given RDP connection and their contents are persisted even after the RDP connection is closed."

Yeah, That Doesn’t Help

- Okay, that admittedly was a lot

  While it is not technically 100% accurate, a better way to think of it is kind of like taking snapshots of the entire screen during an RDP session, which are written to disk on the endpoint that the RDP session originated from


DESKTOP-N12V7FQ - Remote Desktop Connection
Recycle Bin
Certification Tool
Google Chrome
IT WORKED
Search the web and Windows
5:12 PM
10/14/2015

Oh, That Is Better, Thank You!

- The location of the RDP Bitmap Cache files has shifted over the years, but for the most part they can be found under the path “%USERPROFILE%\AppData\Local\Microsoft\Terminal Server Client\Cache\”

More Technical Details

- On older systems, you will usually have a file with a .bmc extension
- Windows 7 and newer systems, you will likely see files that are named “Cache####.bin” (these are incrementally numbered starting at 0000)

- Both file types contain what are essentially small chunks of screenshots that are saved of the remote desktop

More Reading (AFTER This Presentation, please!)

- https://www.allthingsdfir.com/do-you-even-bitmap-cache-bro/
- https://countuponsecurity.com/tag/rdp-bitmap-cache/
- https://cbtgeeks.com/2018/05/22/digital-forensics-on-rdp-cache/
- RDP Cache Forensics - 13Cubed: https://www.youtube.com/watch?v=NnEOk5-Dstw

- https://www.brimorlabsblog.com/2019/06/phinally-using-photoshop-to-phacilitate.html (Hey, that one is mine!)

Well Brian, We Can Read. So Why Are You Here?

- Well, part of the reason is because, like everything else that I do, I want to find an easier way to get usable information from this data source

- I very much enjoy OSDFCON every year, and this is an open source project, so it makes sense
  - *Although this time, it is virtual. Which means not watching giant robots fight, while enjoying a stack of pizzas approximately one Sarah Edwards high, with ~20 of my closest friends*

You Put Files In, You Get Usable Data Out

- Step 1: Extract the data from the RDP Bitmap Cache file(s)
  - I always use the -b flag … but that is up to you

- In my opinion, best current option for this is the Python script from the ANSSI (agence nationale la sécurité des systèmes d'information) github repository

  - https://github.com/ANSSI-FR/bmc-tools
    Note: Use Python v2

You Put Files In, You Get Usable Data Out

- Made a small update to the script to fix a bug
  - The data within the header, referencing file size, is off by 4 bytes (four bytes too long)
    - Most likely counted the “BM” file header (2 bytes) plus hex representation of file size (2 bytes), twice
  - Opened bug request

- Until it is addressed, use this one: https://github.com/brimorlabs/rdpieces/blob/master/modified-bmc-tools.py

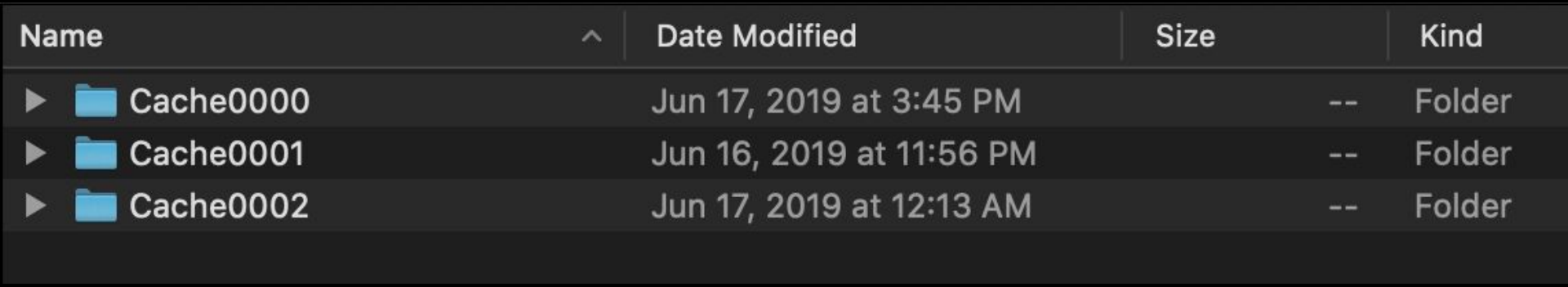
You Put Files In, You Get Usable Data Out

- Alternatively, a PowerShell option is available if you wish to use it.
  - https://github.com/gtworek/PSBits/blob/master/DFIR/DecodeRDPCache.ps1
  - Note: My solution does not currently support the output from this script (if enough interest/requests are made, I can work on building support for this output too though)

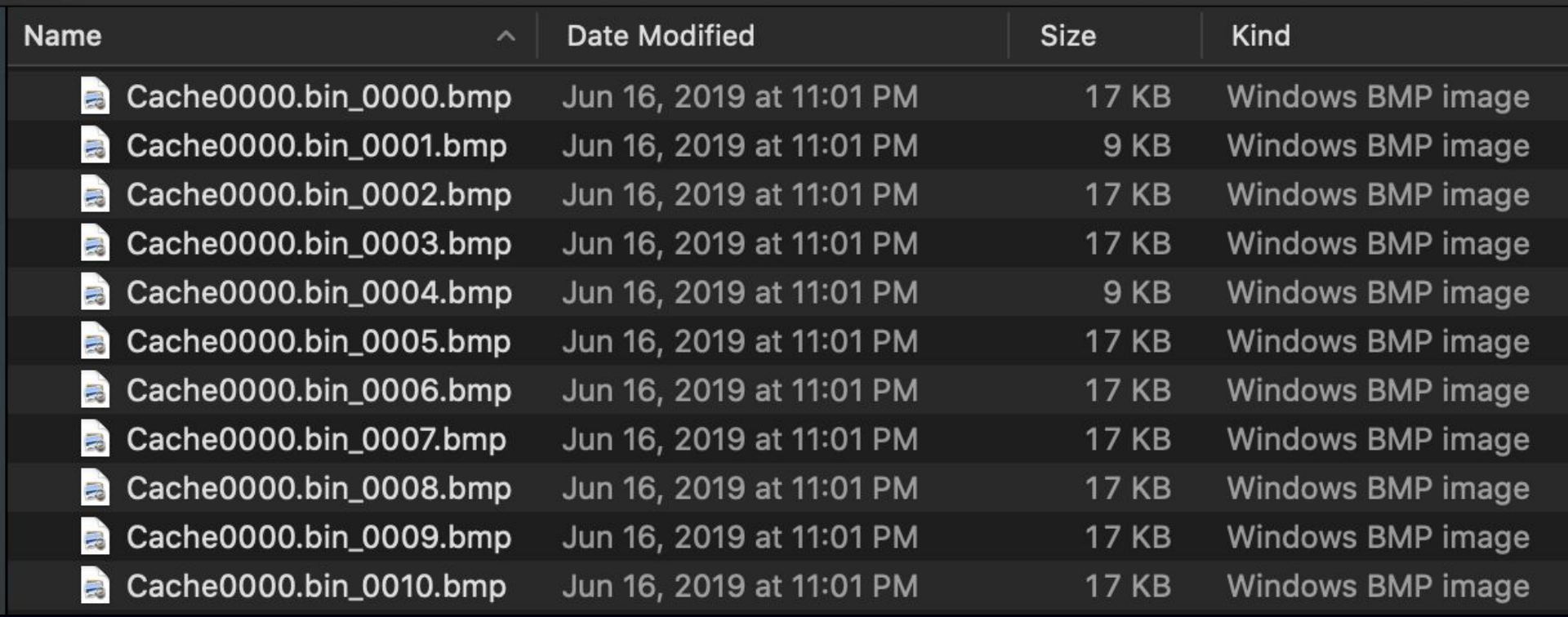
You Put Files In, You Get Usable Data Out

- Step 2: We now have results. Folder structure probably looks like this:

You Put Files In, You Get Usable Data Out

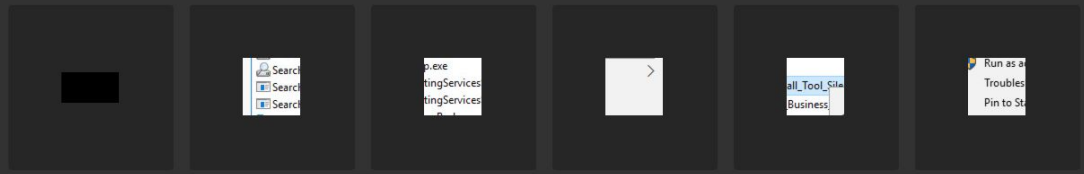


You Put Files In, You Get Usable Data Out



Name	Date Modified	Size	Kind
Cache0000.bin_0000.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0001.bmp	Jun 16, 2019 at 11:01 PM	9 KB	Windows BMP image
Cache0000.bin_0002.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0003.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0004.bmp	Jun 16, 2019 at 11:01 PM	9 KB	Windows BMP image
Cache0000.bin_0005.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0006.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0007.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0008.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0009.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image
Cache0000.bin_0010.bmp	Jun 16, 2019 at 11:01 PM	17 KB	Windows BMP image

Content



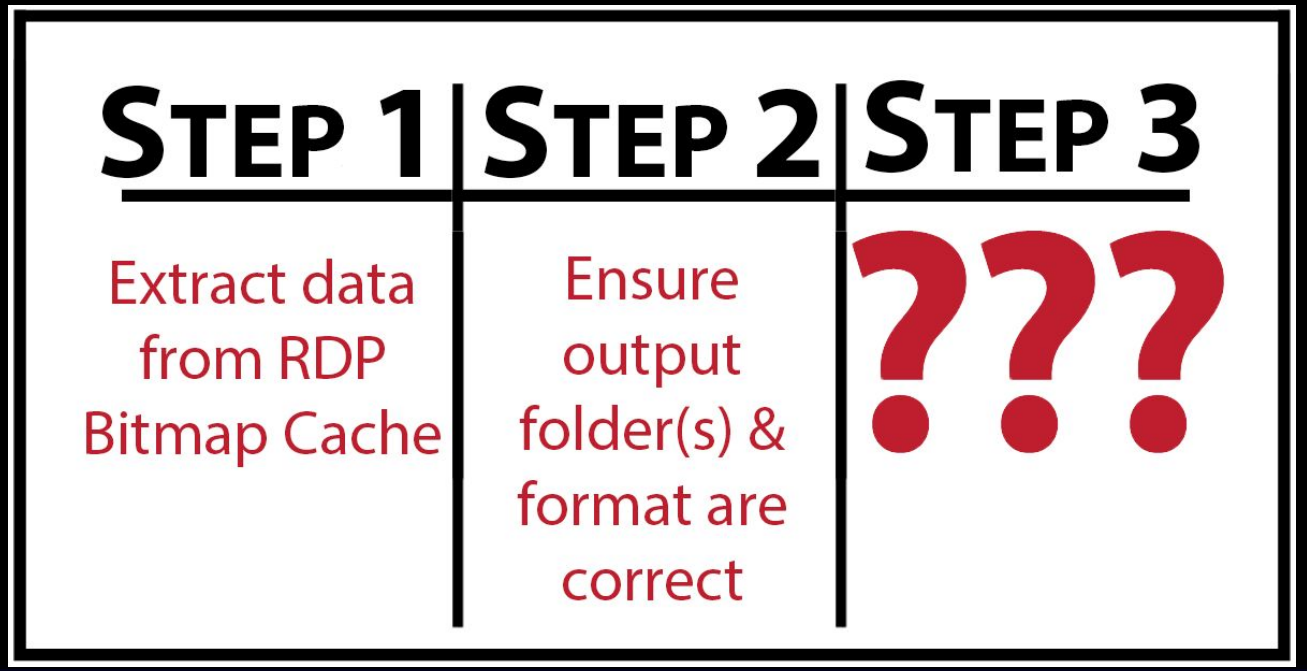
Cache0001.bin _0004.bmp | Cache0001.bin _0010.bmp | Cache0001.bin _0038.bmp | Cache0001.bin _0158.bmp | Cache0001.bin _0164.bmp | Cache0001.bin _0170.bmp

Cache0001.bin _0206.bmp | Cache0001.bin _0212.bmp | Cache0001.bin _0366.bmp | Cache0001.bin _0372.bmp | Cache0001.bin _0399.bmp | Cache0001.bin _0400.bmp

Cache0001.bin _0414.bmp | Cache0001.bin _0428.bmp | Cache0001.bin _0548.bmp | Cache0001.bin _0560.bmp | Cache0001.bin _0574.bmp | Cache0001.bin _0602.bmp

You Put Files In, You Get Usable Data Out

I think this is how this all works?



You Put Files In, You Get Usable Data Out

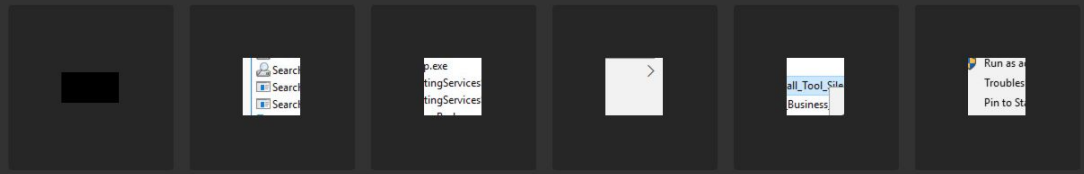
- Step 4: Now you have a whole bunch of bitmap images (usually 6000+) that are 64 x 64*, and one large bitmap file with all of the tiles lined up (see next slide)
  - You can now manually rearrange the individual bitmap images, in hopes of “reconstructing” screen shots that are automatically taken, and stored, during the RDP session
  - This is a challenging, and tedious task

*While a majority are 64 x 64, not all of the images are actually that size. Which makes reconstruction even trickier.

You Put Files In, You Get Usable Data Out

You Put Files In, You Get Usable Data Out

Content

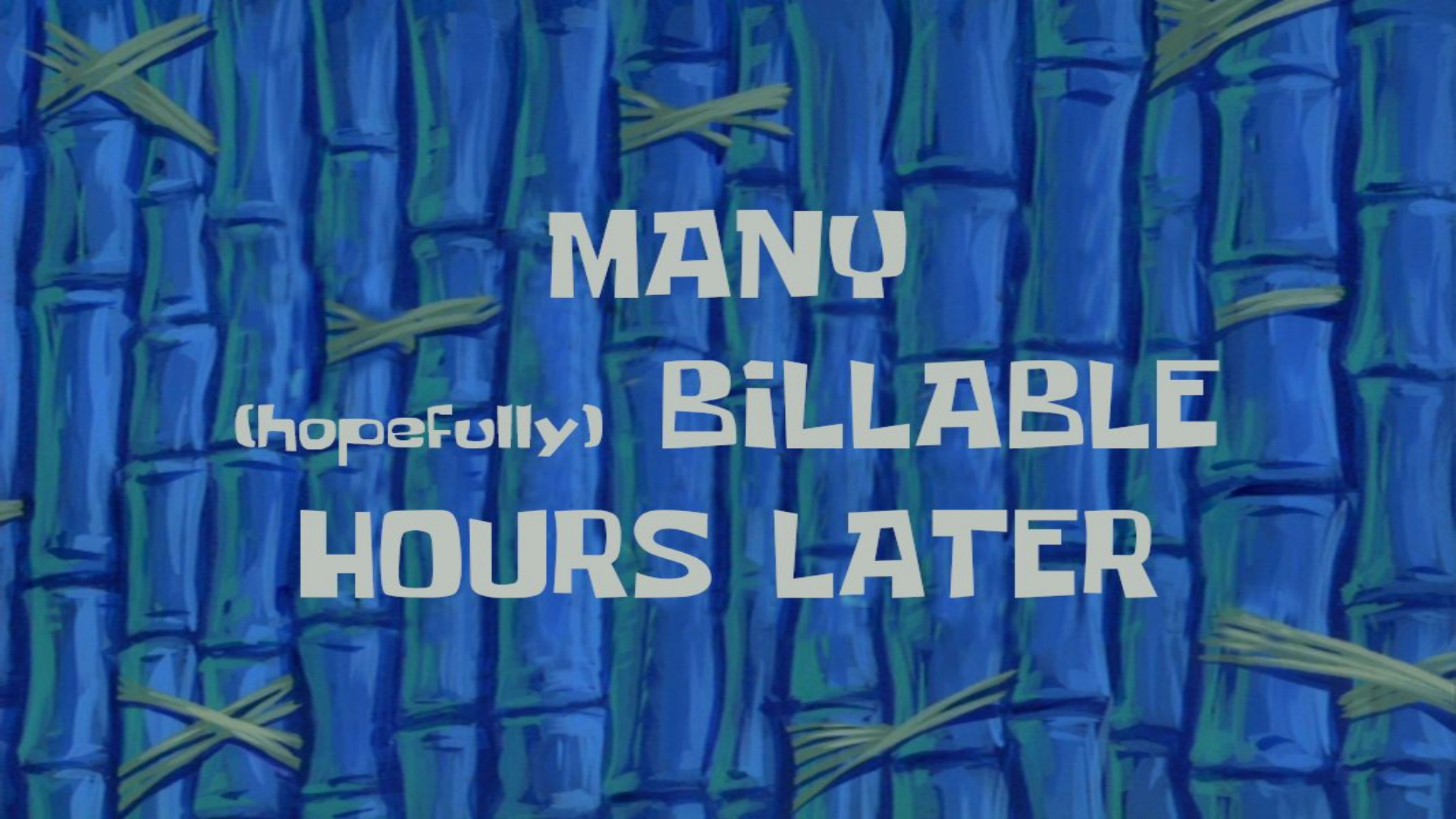


Cache0001.bin _0004.bmp | Cache0001.bin _0010.bmp | Cache0001.bin _0038.bmp | Cache0001.bin _0158.bmp | Cache0001.bin _0164.bmp | Cache0001.bin _0170.bmp

Cache0001.bin _0206.bmp | Cache0001.bin _0212.bmp | Cache0001.bin _0366.bmp | Cache0001.bin _0372.bmp | Cache0001.bin _0399.bmp | Cache0001.bin _0400.bmp

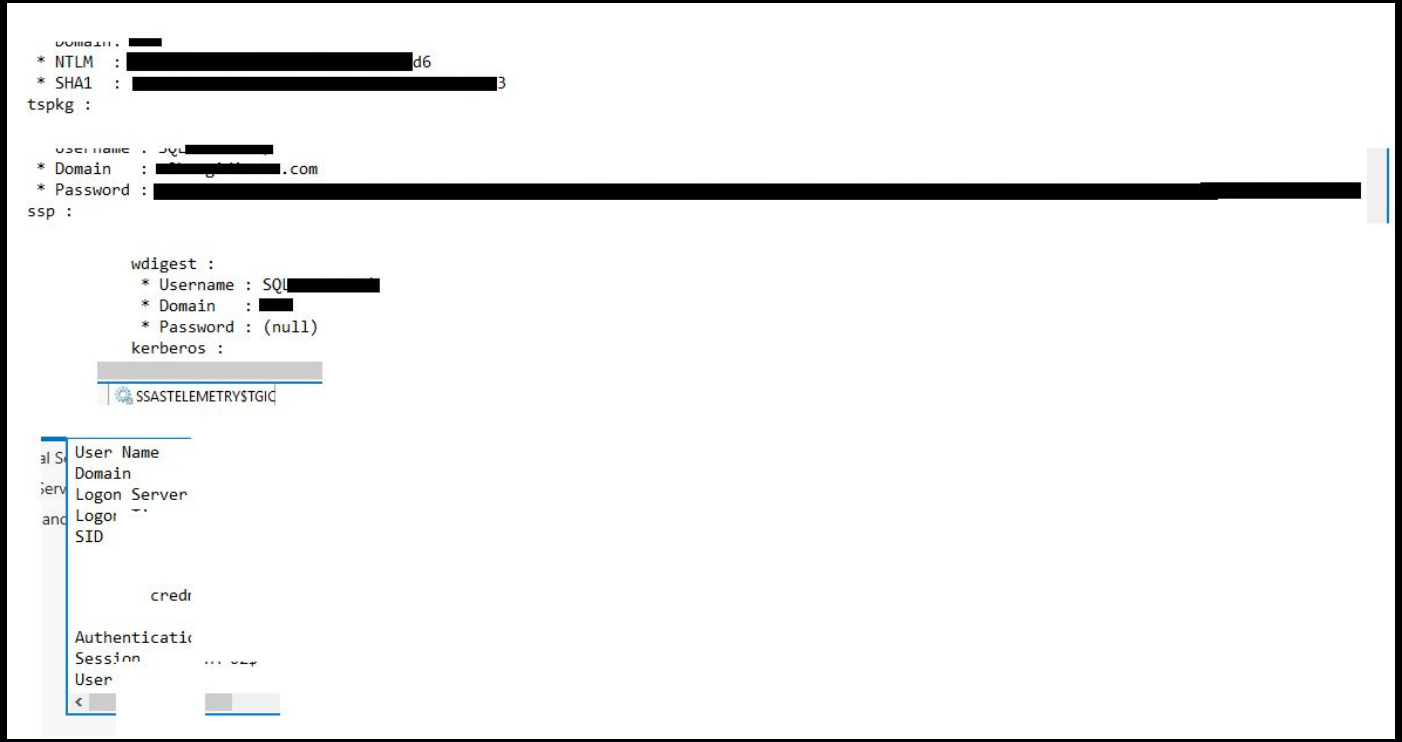
Cache0001.bin _0414.bmp | Cache0001.bin _0428.bmp | Cache0001.bin _0548.bmp | Cache0001.bin _0560.bmp | Cache0001.bin _0574.bmp | Cache0001.bin _0602.bmp

You Put Files In, You Get Usable Data Out


MANY
(hopefully) BILLABLE
HOURS LATER

You Put Files In, You Get Usable Data Out

You Put Files In, You Get Usable Data Out - Manual Reconstruction



You Put Files In, You Get Usable Data Out

- On average, it takes between 20-40 hours to go through and manually rebuild RDP Bitmap Cache data
  - Fine if you have the time (or cough cough billable hours) to do that
- Wanted to make an easier way to at least make slices, and focus on individual slices rather than rebuilding the entire picture

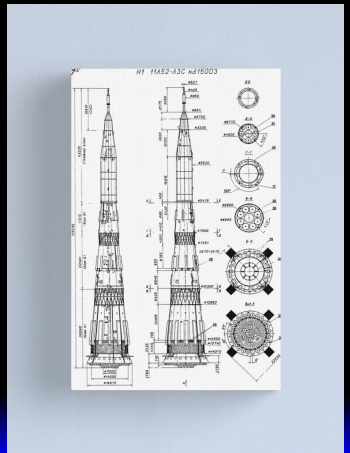
You Put Files In, You Get Usable Data Out

- First thought was

You Put Files In, You Get Usable Data Out

- Started mapping out math, data visualization, statistics, etc. that I thought would be needed

How it started



How it’s going

You Put Files In, You Get Usable Data Out

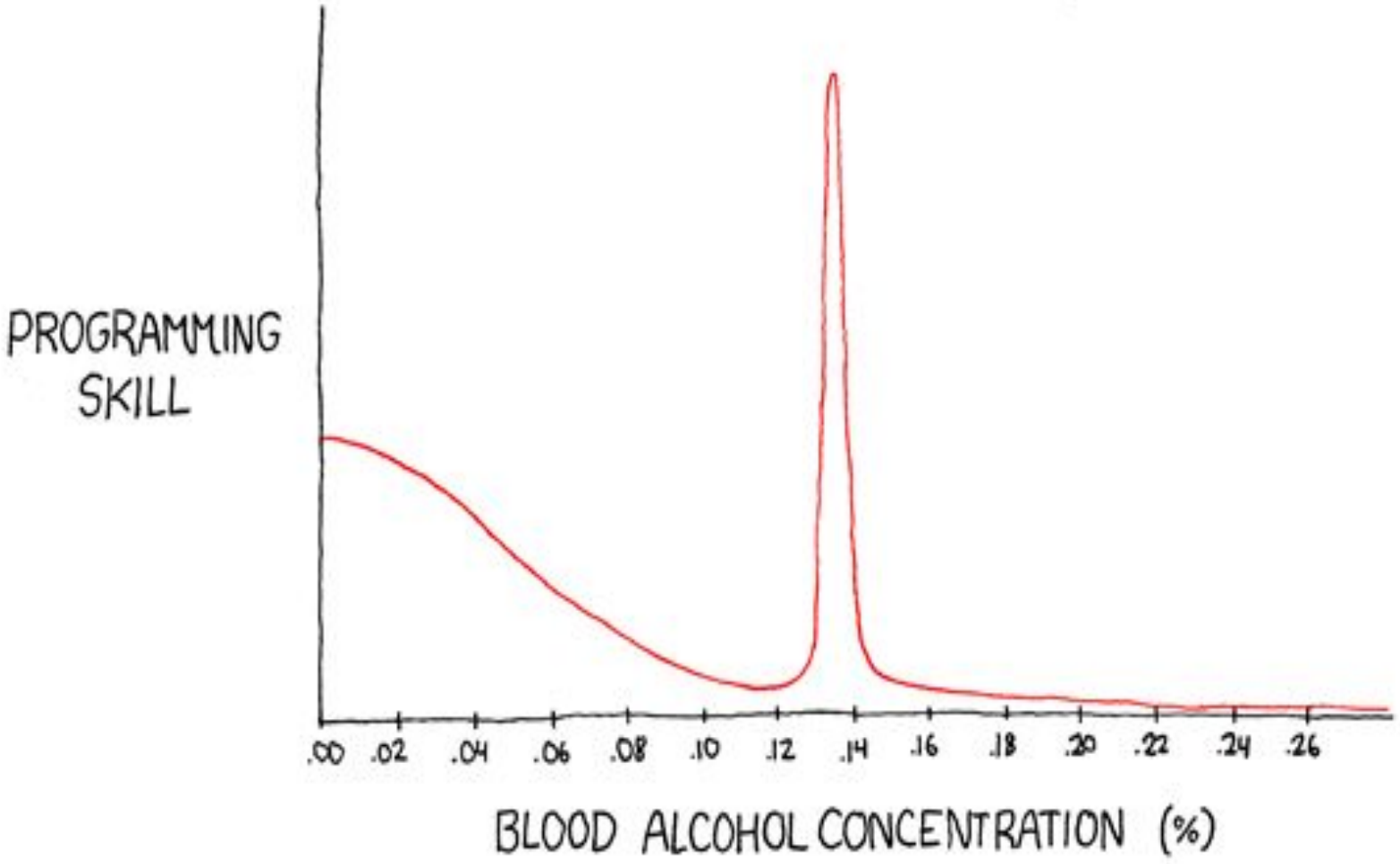
- Enter imagemagick, which as it turns out, does almost everything that I was hoping to find out, and more, already
  - https://imagemagick.org/index.php

You Put Files In, You Get Usable Data Out

- Now that I found where the mathiness would come from, I had to work on ensuring that my formulas worked fairly well, were broad enough to capture less than ideal circumstances, but at the same time, didn't accidentally match too much
- So relieved that I wouldn't have to do terribly complex data manipulation
  - However, it is worth noting that Python absolutely sucks for doing even moderately advanced mathiness, but Perl handles it all like a champ. Long Live Perl!

You Put Files In, You Get Usable Data Out

- The next hurdle was deciding “how” to do this most efficiently
  - Thankfully, my photography hobby (which I do not focus on nearly enough anymore) came into play
  - Alcohol helped too


PROGRAMMING SKILL
.00 .02 .04 .06 .08 .10 .12 .14 .16 .18 .20 .22 .24 .26
BLOOD ALCOHOL CONCENTRATION (%)

You Put Files In, You Get Usable Data Out

- When matching puzzle pieces, you generally look for shapes that go together … and the shapes are determined by the edges …

… hmmm … this line of thinking might actually take me somewhere. Maybe. Possibly.

Probably need another drink … I mean, RDP Bitmap Cache inspiration juice

You Put Files In, You Get Usable Data Out

- So, maybe if I just take the edges of each slice, and figure out how many colors, the color variance/standard deviation, and some file name spatial awareness, maybe I could generate some useful data
- After trial and error, deciding that the edge should be 5 pixels in width/height, depending on if we are matching left/right or top/bottom
  - It’s not perfect, but it is at least a decent solution!

You Put Files In, You Get Usable Data Out

- Used imagemagick to make a total of four new files (filename + L/R/T/B) for each 5x64 or 64x5 slice
  - Code has been updated to account for numbers less than 64 now too, since that's how these work (apparently)
- Pushed the resulting mathiness to a SQLite database that is in memory (for returning faster results)

- The formula can (and undoubtedly will) evolve over time, but it's much easier building SQLite queries than computing mathematical statistics of files!

Introducing: RDPieces.pl

- Perl script that automates everything I just talked about
  - Runs cross platform (Windows, macOS, *nix)
    - macOS/*nix may require some additional modules
    - On Windows, use Strawberry Perl (the best Perl)
  - Requires imagemagick to be installed
  - Script cleans up after itself, deleting temp data directory

- At some point, might make a cool logo for it
  - If large scale ransomware cases ever stop
    - Well, at least slows down

You Put Files In, You Get Usable Data Out

- In my testing, there are roughly 400 results to review per bitmap cache (compared to ~6400 files)
- Put limits on the maximum/minimum size of the slices, because that is how math works

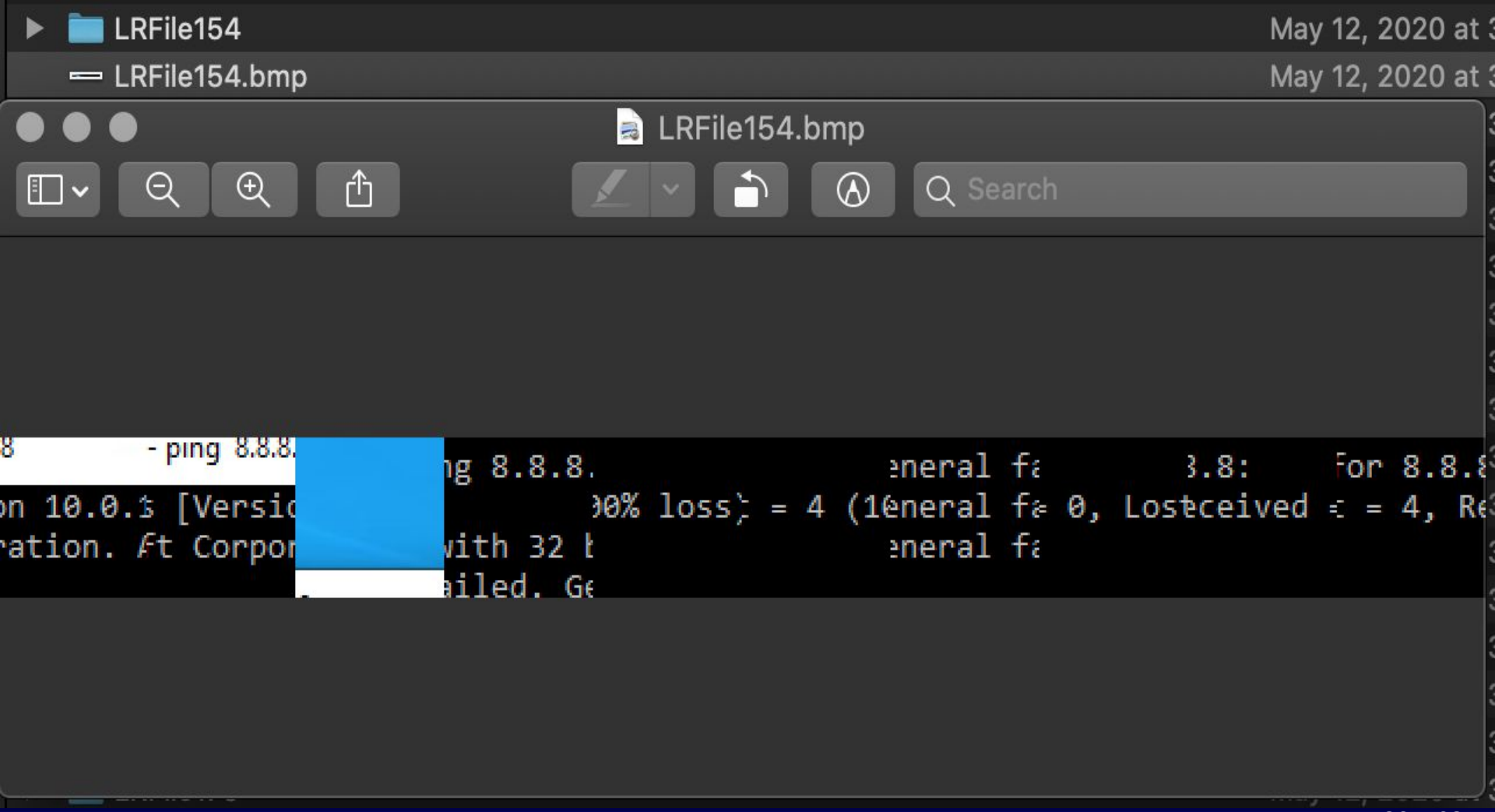
- Script also saves a folder with the rebuilt bitmap image, and the original files used to build the bitmap image, if you want to manually manipulate the files a bit.
  - Much easier than doing it all manually

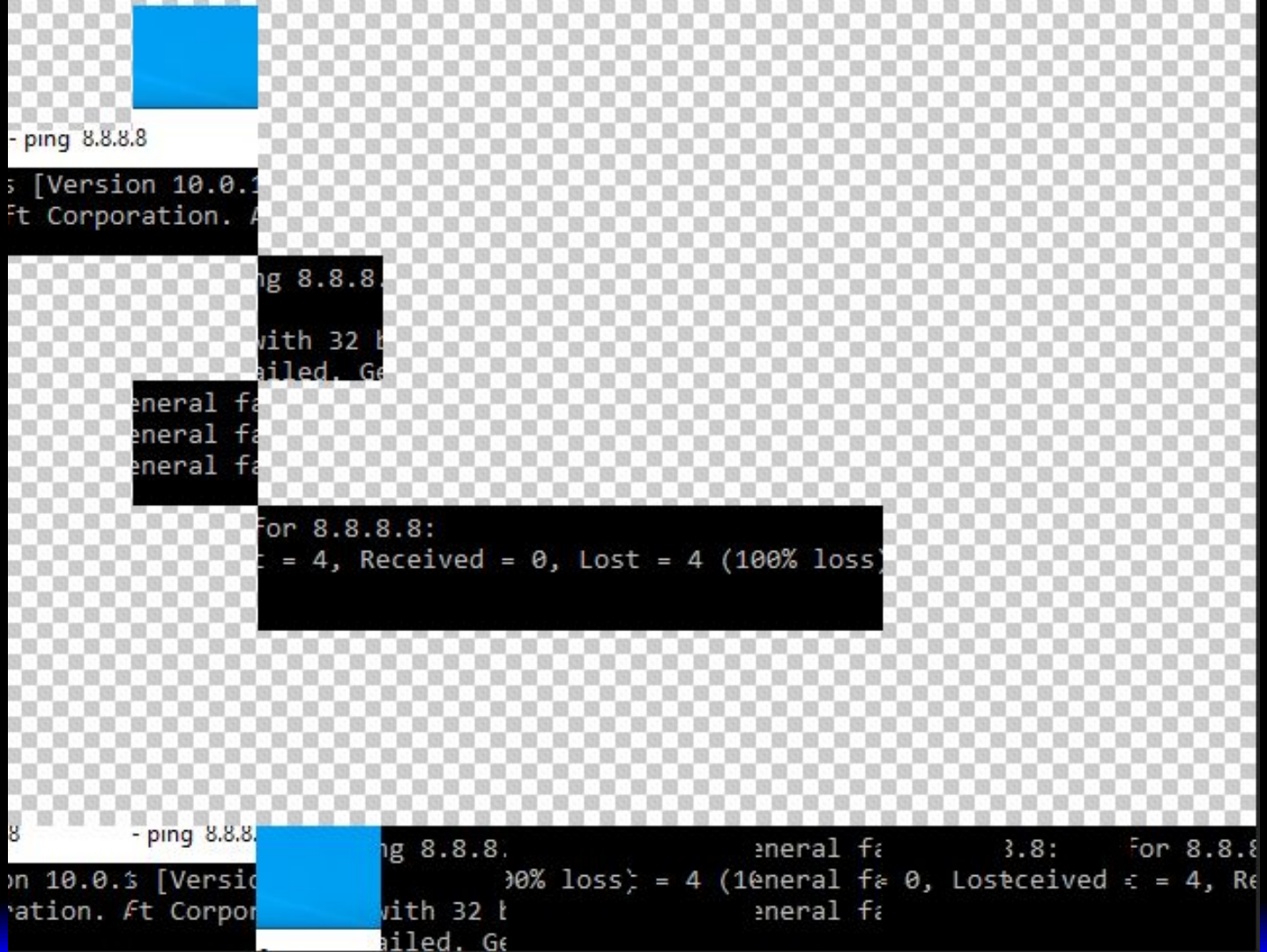
```
***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6082.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6083.bmp
***** NOTICE: Histogram contains only 4 color(s), moving to next file

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6084.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6085.bmp
***** NOTICE: Histogram contains only 5 color(s), moving to next file

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6086.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6087.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6088.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6089.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6090.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6091.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6092.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6093.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6094.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6095.bmp
***** NOTICE: Histogram contains only 2 color(s), moving to next file
```

```
***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6083.bmp
***** NOTICE: Histogram contains only 4 color(s), moving to next file

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6084.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6085.bmp
***** NOTICE: Histogram contains only 5 color(s), moving to next file

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6086.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6087.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6088.bmp

***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-Research/Testing/UAC
-IP-Test/Cache0000.bin_6089.bmp
```

Example of RDPieces.pl running

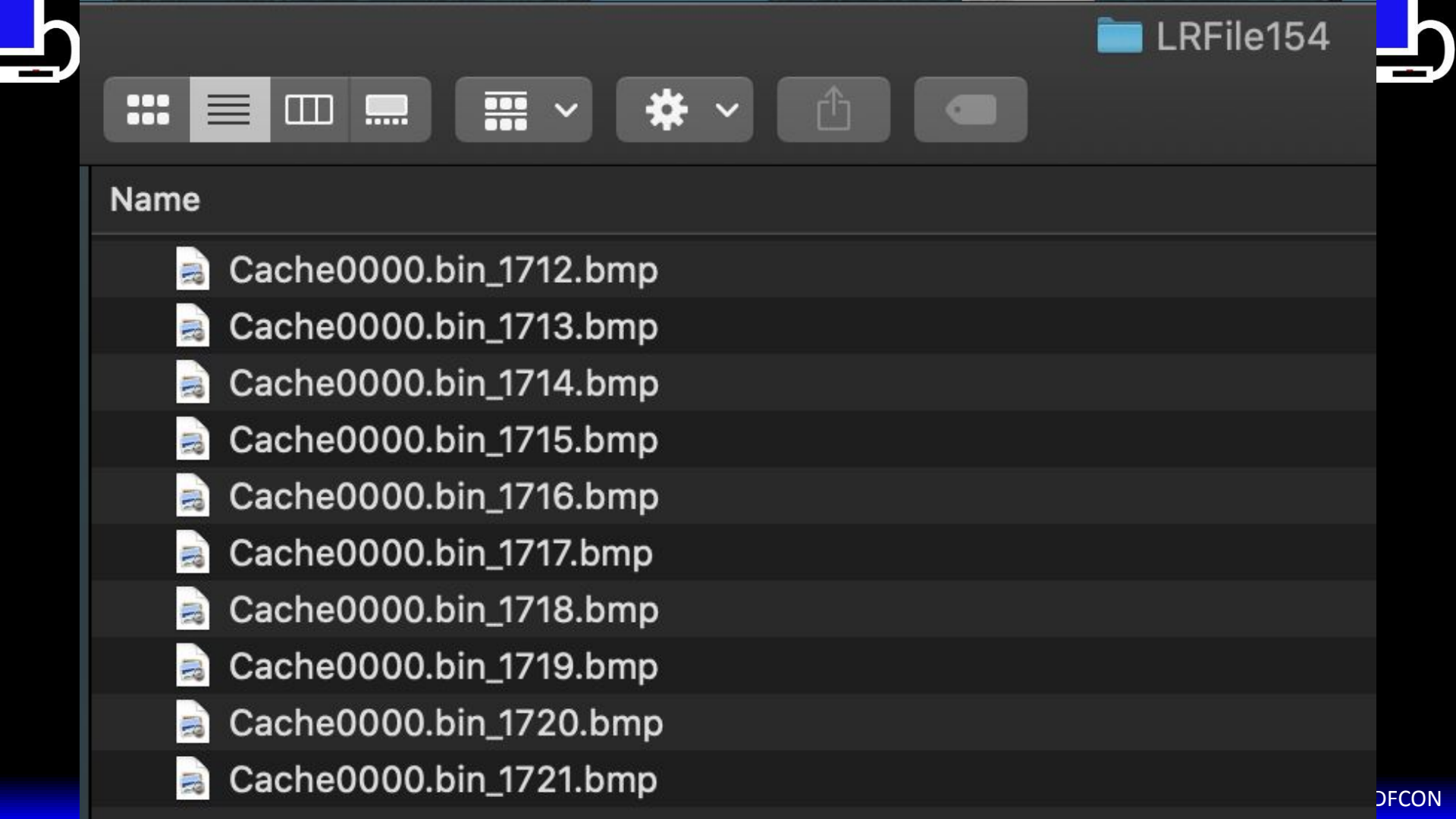
```
***** Processing /Users/brimorlabs/Desktop/RDP-Bitmap-Cache-
Research/Testing/UAC-IP-Test/Cache0000.bin_6100.bmp

Now deleting the directory /Users/brimorlabs/Desktop/foobar/
Data/

A total of 12 files have been copied

The script took 00:00:24 to complete
```

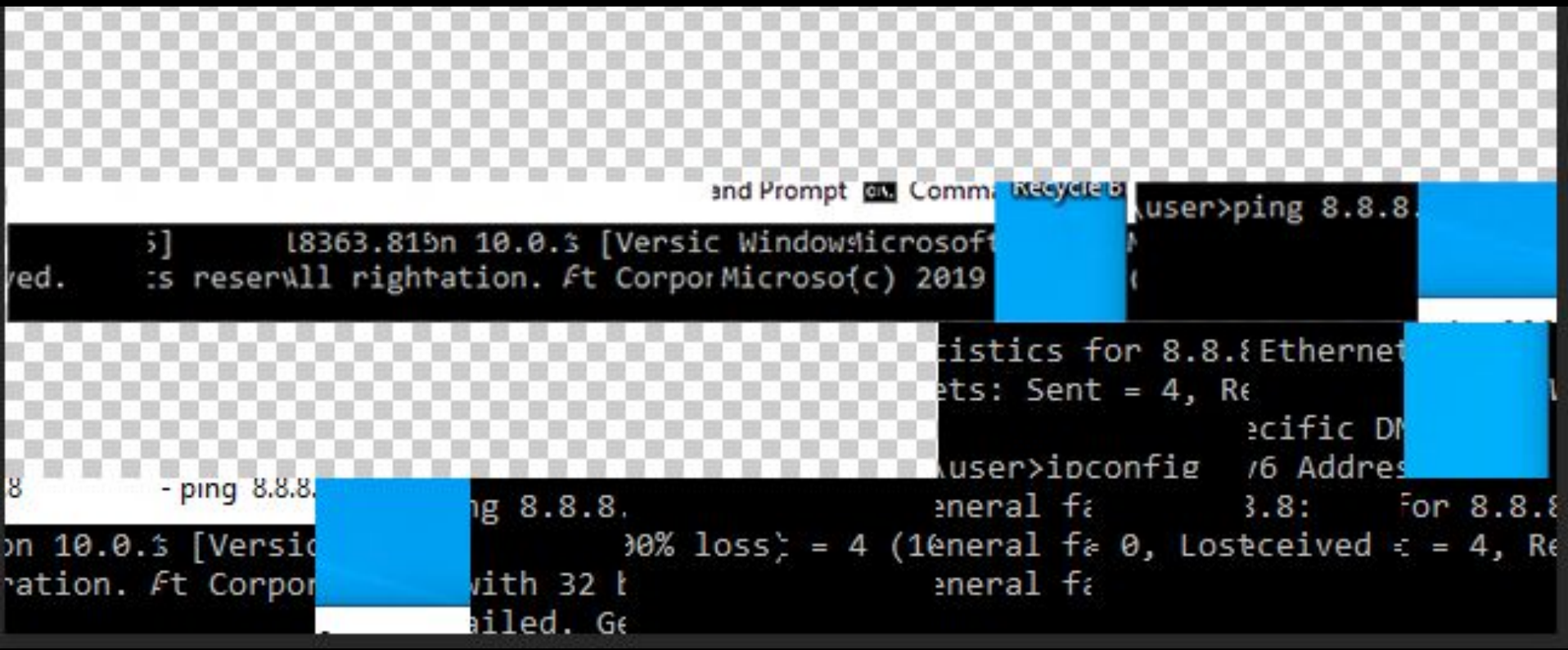

LRFile154
May 12, 2020 at 3
LRFile154.bmp
May 12, 2020 at 3
LRFile154.bmp
Search

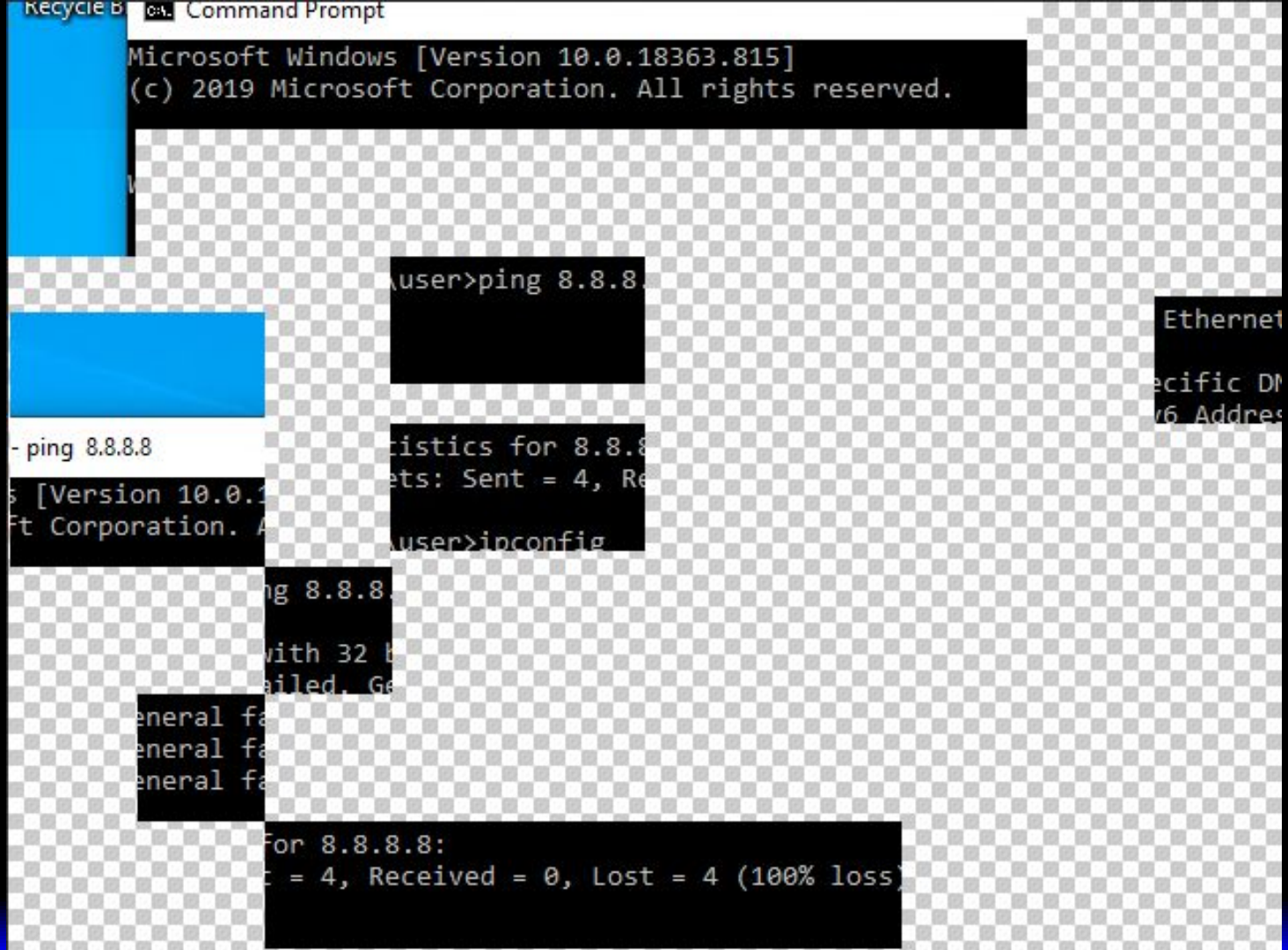

- ping 8.8.8.8
[Version 10.0.1
ft Corporation. A
ng 8.8.8.
with 32 b
ailed. Ge
eneral fa
eneral fa
eneral fa
For 8.8.8.8:
= 4, Received = 0, Lost = 4 (100% loss)
- ping 8.8.8.
on 10.0.1 [Versio
ration. ft Corpor
ng 8.8.8.
00% loss) = 4 (10
with 32 b
ailed. Ge
eneral fa
eneral fa
eneral fa
3.8:
For 8.8.8
0, Lost
eceived = 4, Re

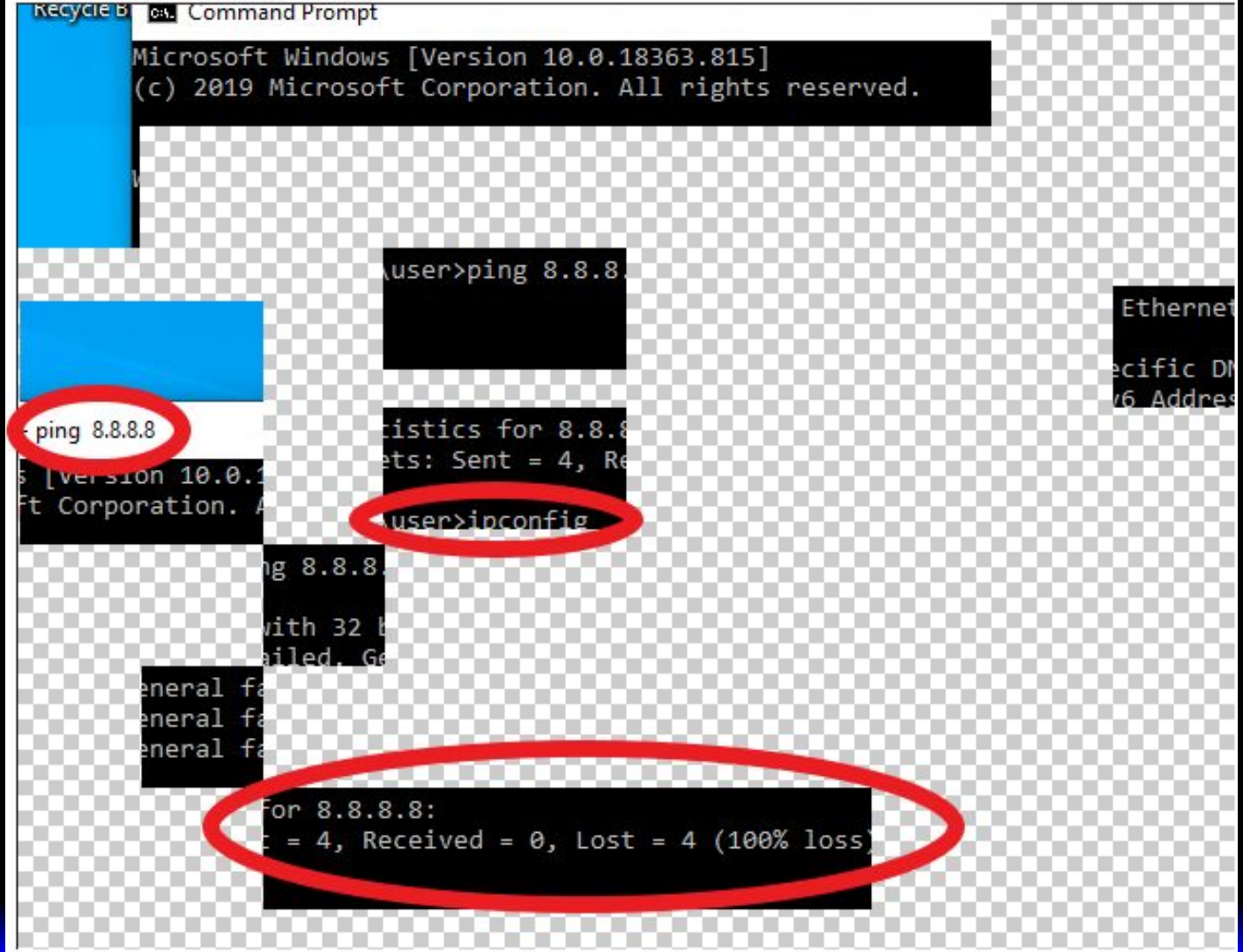
You Put Files In, You Get Usable Data Out

- Because we have the original files that the slice was comprised of, we can then go back and try to rebuild a more complete picture with other slices and/or images


LRFile154
Name
Cache0000.bin_1712.bmp
Cache0000.bin_1713.bmp
Cache0000.bin_1714.bmp
Cache0000.bin_1715.bmp
Cache0000.bin_1716.bmp
Cache0000.bin_1717.bmp
Cache0000.bin_1718.bmp
Cache0000.bin_1719.bmp
Cache0000.bin_1720.bmp
Cache0000.bin_1721.bmp


and Prompt
Comma
Recycle B
user>ping 8.8.8.
8363.819n 10.0.3 [Versic Windows icrosoft
ed.
:s reser ll right tation. ft Corpor Microso(c) 2019
tistics for 8.8.8 Ethernet
ets: Sent = 4, Re
ecific DN
user>ipconfig
/6 Addres
- ping 8.8.8.
ng 8.8.8.
eneral fa
3.8:
For 8.8.8
on 10.0.3 [Versio
00% loss) = 4 (10eneral fa 0, Lost ceived = 4, Re
ration. ft Corpor
with 32 b
eneral fa
ailed. Ge


Recycle Bi
Command Prompt
Microsoft Windows [Version 10.0.18363.815]
(c) 2019 Microsoft Corporation. All rights reserved.
user>ping 8.8.8.
Ethernet
- ping 8.8.8.8
tistics for 8.8.8
ets: Sent = 4, Re
user>ipconfig
ng 8.8.8.
with 32 b
ailed. Ge
eneral fa
eneral fa
eneral fa
For 8.8.8.8:
= 4, Received = 0, Lost = 4 (100% loss)


Command Prompt
Microsoft Windows [Version 10.0.18363.815]
(c) 2019 Microsoft Corporation. All rights reserved.
user>ping 8.8.8
ping 8.8.8.8
tistics for 8.8.8
ets: Sent = 4, Re
user>ipconfig
ng 8.8.8.
with 32
neral fa
neral fa
neral fa
For 8.8.8.8:
= 4, Received = 0, Lost = 4 (100% loss)

You Put Files In, You Get Usable Data Out

- Hey, that isn't too shabby, right?

- We can see that
  - Windows command prompt was used
  - User ran commands "ping" and "ipconfig"

You Put Files In, You Get Usable Data Out - The Next Generation

- This is going to be a continuing project
  - Very much welcome feedback, comments, thoughts on ways to improve it
- My only caveat is that I want to keep this project entirely open source.
  - If Microsoft will not release technical details of how they are doing/rebuilding it, at least we as a community can band together to try to come up with a solution!

Cool Story Bro, Where Can I Get It?

- You can download the Perl script here:
  - https://github.com/brimorlabs/rdpieces

- Again, my only caveat is that I want to keep this project entirely open source
  - I am sure there are different, and probably better, ways to perform mathiness
  - Sharing is caring

BriMor Labs

RDPieces.pl

This script will parse extracted RDP Bitmap Cache directory(ies) and attempt to rebuild some of the screenshots automatically. A user is required to extract the bmp files already, best done by using the script from https://github.com/ANSSI-FR/bmc-tools

Usage example: rdpieces.pl -source "RDPBitmapFiles" -output "Rebuilt Images"

SUPPORTED PLATFORMS:

- Windows
- macOS
- *nix

REQUIREMENTS:

- Needs output from ANSSI bmc-tools Python script (use Python 2)
- May require some additional Perl modules
- On Windows, highly suggest using Strawberry Perl
- Users must have Imagemagick installed on their system, as that program does most of the heavy lifting. Please visit https://imagemagick.org and install imagemagick if you have not done so already

You Put Files In, You Get Usable Data Out

- Heather (@LitMoose) summed it up best:

“It is like putting together an adult jigsaw puzzle, but for forensic analysts”

She also said something to the effect of “*it’s kind of relaxing*”, which makes me question things about her 😂

Naughty By Nature said it best …

Questions?

Brian Moran

Twitter: @brianjmoran

Email: brian@brimorlabs.com

Wear a mask

Wash your hands

Practice social distancing

Avoid large (in-person) gatherings