Write-Progress Extension Classes

In this document I'm going to cover some incredibly simple classes that aren't complicated at all, to orchestrate the division and calculations for the function “Write-Progress”.

I'll go over SPLATTING and stuff, too.

Beware... if you happen to be the type of person that has estrogen in their bloodstream and YOU'RE EASILY OFFENDED by like CURSE WORDS and stuff...? Close the file. Don't even bother reading this, because... lemme tell ya... you're NOT gonna like the fact that I, will swear a number of times in this document.

Why...? Because I'm an EXPERT. Professionals will curb their language.
EXPERTS do not have to do that.
EXPERTS can tell people to fuck right off at any given moment...?
And, not have to worry about being FIRED, LET GO, CANNED, or otherwise.

Nah. EXPERTS, have a LOT more experience, than PROFESSIONALS.
By the time anybody reading this document reads the entire thing...?

They'll probably read my book... it is listed later in the document.

Overview

Below, is a COPY+PASTE of my actual VSCode editor of the script I've been working on for the EventLog Utility that dynamically allocates runspaces, threads, assigns functions, assemblies, and variables to a runspace control object, so that child (runspaces/threads) can be spun up, and calculate the work in a PARALLEL manner, rather than (in a SERIAL manner/PROCEDURALLY).

I wanted to talk about LOGGING, DateTime objects, status, percentages, progress indicators, and estimated time to completion... all of this stuff is pretty complicated when it comes down to the nitty gritty.
But- this lesson plan is going to take a LOT of the COMPLICATION, out of something pretty complicated.

I'm fairly certain that what is EXHIBITED in this particular lesson plan, is a fair amount of CALCULUS.
But, feel free to dispute that if it isn't.

Anyway, when dividing the workloads between multiple threads, you really are playing with FIRE because unless you know how to feed the custom functions and classes to EACH RUNSPACE...? Then basically you're ASKING for trouble to occur at some point.

The RUNSPACE OBJECTS and dispatchers are well designed.
The problem is, how can anyone tell WHEN the RUNSPACE OBJECTS, will be COMPLETE with the WORKLOAD...?

If you are not familiar with RUNSPACE objects or THREADING...? That's ok, feel free to ask questions if you don't understand what I'm discussing, but want to know more anyway.

For those who DO understand RUNSPACE OBJECTS and THREADING...?
You have an idea of how or why these classes might come in handy.
In THIS PARTICULAR INSTANCE...? I'm not going to talk about the objects that handle the threads.
Because that is COMPLICATED and LENGTHY.

However, what I WILL talk about, are how to use these (2) classes to apply to the function Write-Progress.
Because this function allows a user to see when a particular workload will be COMPLETED.

Here's why this is pretty important.
Without having a way to indicate when/where a process is in it's trajectory...?
Anyone choosing to use these functions in relation to multiple threads is basically gonna have to throw their dick in the wind, and hope that it lands in the right place.

That's not a real great strategy when you're an APPLICATION DEVELOPER, or just a typical user, right...?
Cause the wind could be blowing in a different direction on any given day.
Might land in a different direction, too.
So, you have to have a WAY to CONTROL all of this stuff... and that's sorta what these (2) classes below go right ahead, and DO.

Overview

Percent + Progress /----------\
/--------------------

```powershell
# // ------------------------------------------------------------------------------------------
# // | This is a single percentage calculation, so that the workload can be evenly distributed |
# // | BEFORE any of the ACTUAL work is completed.                                              |
# // ------------------------------------------------------------------------------------------

Class Percent
{
    [UInt32] $Index
    [UInt32] $Step
    [UInt32] $Total
    [UInt32] $Percent
    [String] $String
    Hidden [String] $Output
    Percent([UInt32]$Index,[UInt32]$Step,[UInt32]$Total)
    {
        $This.Index             = $Index
        $This.Step              = $Step
        $This.Total             = $Total
        $This.Calc()
    }
    Calc()
    {
        $Depth                  = ([String]$This.Total).Length
        $This.Percent           = ($This.Step/$This.Total)*100
        $This.String            = "({0:d$Depth}/{1}) {2:n2}%" -f $This.Step, $This.Total, $This.Percent
    }
    [String] ToString()
    {
        If ($This.Output)
        {
            Return $This.Output
        }
        Else
        {
            Return $This.String
        }
    }
}

# // ------------------------------------------------------------------------------------------------
# // | This is a progress container, meant for dividing the work evenly, though < 100 doesn't work yet |
# // ------------------------------------------------------------------------------------------------

Class Progress
{
    [String]  $Activity
    [Object]    $Status
    [UInt32]   $Percent
    [DateTime]   $Start
    [UInt32]     $Total
    [UInt32]      $Step
    [Object[]]    $Slot
    [UInt32[]]   $Range
    Progress([String]$Activity,[UInt32]$Total)
    {
        $This.Activity          = $Activity
        $This.Start             = [DateTime]::Now
        $This.Total             = $Total
        $This.Step              = [Math]::Round($Total/100)
        $This.Slot              = @( )
        ForEach ($X in 0..100)
        {
            $Count              = @($This.Step * $X;$Total)[$X -eq 100]

            $This.AddSlot($X,$Count,$Total)
        }
        $This.Range             = $This.Slot.Step
        $This.Current()
```

```
        }
        AddSlot([UInt32]$Index,[UInt32]$Multiple,[UInt32]$Total)
        {
            $this.Slot             += [Percent]::New($Index,$Multiple,$Total)
        }
        Increment()
        {
            $This.Percent ++
            $This.Current()
        }
        [UInt32] Elapsed()
        {
            Return ([TimeSpan]([DateTime]::Now-$This.Start)).TotalSeconds
        }
        [String] Remain()
        {
            $Remain                = ($This.Elapsed() / $This.Percent) * (100-$This.Percent)
            $Seconds               = [TimeSpan]::FromSeconds($Remain)
            Return "(Remain: {0}, ETA: {1})" -f $Seconds, ([DateTime]::Now+$Seconds)
        }
        Current()
        {
            $This.Status           = $This.Slot[$This.Percent]
            If ($This.Percent -ne 0)
            {
                $This.Status.Output = "{0} [{1}]" -f $This.Status.String, $This.Remain()
            }
            Else
            {
                $This.Status.Output = $This.Status.String
            }
        }
        SetStatus([Object]$Percent)
        {
            $This.Status           = $Percent
            $This.Percent          = $Percent.Percent
            $This.Current()
        }
        [Hashtable] Splat()
        {
            Return [Hashtable]@{

                Activity           = $This.Activity
                Status             = $This.Status
                Percent            = $This.Percent
            }
        }
    }
```

    I'll write a custom class for a PARTICULAR WAY to APPLY these above classes.
    So, lets say that I want to get the FILE HASH on EACH PARTICULAR FILE in a DIRECTORY...?

                                                                                                                    _____________________/
____/ Percent + Progress
  Describing the (Joe Manchin/Ted Cruz) Mentality /____\
/---

    Well, if we have a (JOE MANCHIN/TED CRUZ) mentality...?
    We would want to take AS MUCH TIME AS POSSIBLE, for each individual file, rather than the LEAST amount.

    Because DOING THINGS in the LEAST AMOUNT OF TIME...?
    They BOTH think that's... pretty fuckin' stupid, and that's the end of the conversation in THEIR eyes.

    /--__/--\

    Manchin/Cruz : Seriously...?
                   DO stuff in the LEAST amount of time...?
    Me   : Yeh.
    Manchin/Cruz : Do you have ANY IDEA how stupid that is...?
    Me   : It's not stupid at all.
    Manchin/Cruz : Look, you fuckin' moron...
                   Who the hell do you think you are, trying to tell people HOW to DO stuff, in the LEAST

```
                amount of time...? Huh, tough guy...?
Me           : I'm just some dude that thinks the both of you are WICKED LAZY.
Manchin/Cruz : That's just your opinion, dude.
               Time to kick rocks, pal...
Me           : Whatever dude.

\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/
```

So in reality, they really like taking as MUCH TIME AS POSSIBLE, to GET ANY WORK DONE.
So every day that they show up for “WORK”, they actually just run through the SAME EXACT PROCESS,
as they did the day BEFORE... where they performed the following routine:

1) FIRST hour, they stop at DUNKIN' DONUTS, and buy at least (6) BOSTON CREME DONUTS.
2) Once they get to their desk at Capital Hill...? That's when it's time to kick their feet up on the desk,
   and just stare out the window, with a blank expression on their face, while they eat those (6+) donuts.
3) SECOND HOUR, they're gonna tell EVERYBODY how dumb they are, but, using NICE WORDS.
   They can't risk OPENLY OFFENDING A SINGLE SOUL... Nah. Gotta do it QUIETLY, and SNEAKILY...
4) THEN, from the THIRD to the SEVENTH hour...?
   That's when it's fuckin' nap time.
   BEFORE this nap of theirs, they're gonna start out by grabbing (1-2) JUICE BOX(s)...
   ...once (that/those) juice box(es) (is/are) EMPTY...?
   THEN, it's nap time. 5 hour naps are great... they wake up fully rested...? Life is perfect too.
   ESPECIALLY if they're getting paid MILLIONS OF DOLLARS, to effectively be 100% useless to society.

Who cares if 19 kids and 2 teachers get indiscriminately shot to death by someone that needed some serious
help...? You're getting paid MILLIONS OF DOLLARS, to take a long comfy nap, and perform the SAME EXACT PROCESS
that you did the day before.

Every once in a while though... man. Someone will show up, and catch one of these morons in the act.
Normally, what I just said, would be considered (BEING REALLY LAZY/ACCEPTING BRIBERY), but... who's gonna
catch someone taking a 5 hour nap on Capital Hill EVERY DAY...? Nobody will. That's right.

```
                                                                            ________________________________________________/
\__________________________________________________________________________/ Describing the (Joe Manchin/Ted Cruz) Mentality
  FileHash + FolderHash /¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯\
/¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯
```

Anyway, here's a couple of classes that operate off of a SINGLE PARAMETER.
A PATH variable, and then it'll calculate the hashes for each individual file... I'll show the output below.

```
# // ____________________________________________________________________________________
# // | This is a file object that collects information from EACH GET-CHILDITEM INSTANCE, |
# // | and EXTENDS the CLASS so that a method can collect the hash of each file, and,    |
# // | accommodates OTHER hash algorithms, so you don't have to use the DEFAULT SHA256   |
# // ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

Class FileHash
{
    [UInt32]           $Index
    [String]            $Name
    [DateTime] $LastWriteTime
    [UInt64]          $Length
    Hidden [String] $Fullname
    [String]       $Algorithm
    [String]            $Hash
    FileHash([UInt32]$Index,[Object]$File)
    {
        $This.Index             = $Index
        $This.Name              = $File.Name
        $this.LastWriteTime     = $File.LastWriteTime
        $This.Length            = $File.Length
        $This.Fullname          = $File.Fullname
    }
    GetFileHash([String]$Type)
    {
        # // ______________________________________________________________________________________________
        # // | <Mrs. Quigley told me, this word is NOT based on Al Gore, the former Vice President, so...> |
        # // ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

        $This.Algorithm         = $Type
        $This.Hash              = Get-FileHash -Path $This.Fullname -Algorithm $This.Algorithm | % Hash
```

```
        }
    }

    # // ------------------------------------------------------------------------------------------
    # // | This is effectively a CONTAINER class, meant to provide controls over the input path, |
    # // | and extends the (OPERATING SYSTEM + POWERSHELL)'s default functionality.              |
    # // ------------------------------------------------------------------------------------------

    Class FolderHash
    {
        [String]      $Name
        [String]      $Path
        [String] $Algorithm
        [Object]    $Output
        FolderHash([String]$Path)
        {
            If (!(Test-Path $Path))
            {
                Throw "Invalid path: [$Path]"
            }

            $This.Path              = $Path
            $This.Name              = [DateTime]::Now.toString("yyyy_MMdd-HHmmss")
            $This.Output            = @( )
            $List                   = Get-ChildItem $Path

            $P                      = $This.Progress("Adding [~]",$List.Count)
            $Progress               = $P.Splat()
            Write-Progress @Progress
            ForEach ($X in 0..($List.Count-1))
            {
                If ($X -in $P.Range)
                {
                    $P.Increment()
                    $Progress       = $P.Splat()
                    Write-Progress @Progress
                }

                $This.Add($List[$X])
            }
            $Progress               = $P.Splat()
            Write-Progress @Progress
        }
        [Object] Progress([String]$Activity,[UInt32]$Count)
        {
            If (!$Activity)
            {
                Throw "Must specify an activity or label"
            }
            If ($Count -lt 100)
            {
                Throw "Count must be higher than 100 for the time being..."
            }
            Return [Progress]::New($Activity,$Count)
        }
        Add([Object]$File)
        {
            If ($File.FullName -in $This.Output.FullName)
            {
                Throw "Exception [!] [$($File.Fullname)] already specified."
            }
            $This.Output           += [FileHash]::New($This.Output.Count,$File)
        }
        GetFileHash([String]$Type)
        {
            If ($Type -notin "SHA1 SHA256 SHA384 SHA512 MACTripleDES MD5 RIPEMD160".Split(" "))
            {
                Throw "Invalid Algorithm"
            }

            $This.Algorithm         = $Type
            $P                      = $This.Progress("Hashing [~]",$This.Output.Count)
```

```
        $Progress               = $P.Splat()
        Write-Progress @Progress
        ForEach ($X in 0..($This.Output.Count-1))
        {
            If ($X -in $P.Range)
            {
                $P.Increment()
                $Progress       = $P.Splat()
                Write-Progress @Progress
            }

            $This.Output[$X].GetFileHash($This.Algorithm)
        }
        $Progress               = $P.Splat()
        Write-Progress @Progress -Complete
    }
}
```

I could easily add a SINGLE LINE OF CODE to make this particular process work for LESS THAN 100 items...
But then people who want to learn how to DO that, won't learn (how/when/where) to DO that.

Not gonna put anybody's name in here, but it is something that I would expect from somebody like
JOE MANCHIN or TED CRUZ.

```
/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\

Manchin/Cruz : Hey buddy, you've got a LOT OF NERVE to try and make me THINK HARD...
Me           : Nah I don't, you're lazy.
               You need to use your head more often.
Manchin/Cruz : LOL, YEAH RIGHT, BUDDY~!
               DREAM ON, PAL~!

\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/
```

And then, they expected me to do ALL of the work FOR them, rather than for THEM...
to figure out how to do it on their OWN. Cool, huh...?

STUDENT LOAN FORGIVENESS, and NOT ACCEPTING BRIBERY FROM EXXON MOBIL...?

Those things are NOT in THEIR best interest AT ALL...

```
/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\

Somebody : Nah...?
Me       : Nah.
           Guys like JOE MANCHIN, their FRIENDS are RICH/POWERFUL because of the amount of DEBT that people
           have. Student loan forgiveness would IMMEDIATELY ELIMINATE the WEALTH that SOME PEOPLE HAVE off of...
           DEBT INTEREST.
Somebody : ...really?
Me       : Yep.
           The reason why 50% of the governors around the country wrote a letter to JOE BIDEN telling him to
           REVERSE the STUDENT LOAN FORGIVENESS bill, is literally because some people are "WEALTHY" because
           those DEBTS still generate this COOL stuff called "INTEREST".
           That means when the DEBTS are all FORGIVEN/ELIMINATED...?
           There's no more INTEREST.
           Without that INTEREST...?
           They don't get a FREE RIDE anymore.
           It's basically VERY SIMILAR to how ENRON defrauded the entire STATE of CALIFORNIA back in the 90's.
           But also, how GEORGE W. BUSH convinced AMERICA to attack the MIDDLE EAST for OIL in KUWAIT...
           ...did a PERFECT JOB of making the TERRORIST ATTACK of 9/11 APPEAR TO BE UNWARRANTED...
           ...so that invading AFGHANISTAN would be a PRETEXT for attacking SADDAM HUSSEIN/IRAQ/KUWAIT...
           ...being 100% legitimate in APPEARANCE.
Somebody : WOW, REALLY...?
Me       : Yup.
           That's how LAZY (some people/oil corporations) really are.

\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/
```

Now, you can't just go around like I do, telling people why the USA-PATRIOT Act of 2001 was ACTUALLY passed.
Basically digital slavery. Nobody really cares...? At least, until you show them what the reasons are, for
being in MY particular situation.

```
    You can be A LOT MORE EXPERIENCED AND TALENTED than a lot of people in any given industry...?
    But some sad son of a bitch will constantly spam the shit out of you with DOWNVOTES, thumbs downs, moronic
    comments and opinions, and entitled brats that don't realize how often they are lied to on a daily basis.

    In the end...? There's a LOT OF RESISTANCE out there in the world...
    However...? You can just DO WHAT I DO, and show people some REALLY COOL STUFF, like:
    _______________________________________________________________
    | 09/26/2022 | God Mode Cursor | https://youtu.be/tW80Zj_H6Fw |
    ---------------------------------------------------------------
    I am FAIRLY CERTAIN, that whoever is causing this particular artifact to show up in VSCode...?
    They're one of the most important people in the world.
                                                                                                 ______________________/
\_______________________________________________________________________________________________/ FileHash + FolderHash
  Instantiation /¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯\
/¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

    Now, lets run through the portion of the code where anybody could run this code, and it'll do some WICKED cool
    stuff that will be available as a VIDEO.
```

```
    # // ______________________________________________________________________
    # // | It's 9:00 AM. Time to open, and take care of business...             |
    # // | Let's make a MOCK FOLDER somewhere with a bunch of EMPTY TEMP FILES |
    # // ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

    $Path                           = "$Home\Desktop\Temp"
    If (!(Test-Path $Path))
    {
        New-Item $Path -ItemType Directory -Verbose
    }

    # // ___________________________________________________________________________
    # // | Create a bunch of empty text file names, converting an input number into |
    # // | CAPITALIZED HEXADECIMAL FORMAT with a .txt extension, and prefix with    |
    # // | the ABOVE PATH... (ForEach-Object w/ .NET format + string interpolation) |
    # // ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

    $List                           = 0..4095 | % { "{0}\{1:X3}.txt" -f $Path, $_ }

    # // ________________________________
    # // | Establish a PROGRESS TRACKER |
    # // ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

    $P                              = [Progress]::New("Creating [~]",$List.Count)

    <#
    __________________________________________________
    | WHAT THE PROGRESS TRACKER OBJECT LOOKS LIKE... |
    ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯
    PS Prompt:\> $P

    Activity : Creating [~]
    Status   : (4096/4096) 100.00% [(Remain: 00:00:00, ETA: 9/29/2022 6:24:41 PM)]
    Percent  : 100
    Start    : 9/29/2022 6:24:36 PM
    Total    : 4096
    Step     : 41
    Slot     : {(0000/4096) 0.00%, (0041/4096) 1.00% [(Remain: 00:01:39, ETA: 9/29/2022 6:26:17 PM)]...}
    Range    : {0, 41, 82, 123...}
    ____________________________________
    | WHAT THE SLOT ARRAY LOOKS LIKE... |
    ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯
    PS Prompt:\> $P.Slot | Format-Table

    Index Step Total Percent Output
    ----- ---- ----- ------- ------
        0    0  4096       0 (0000/4096) 0.00%
        1   41  4096       1 (0041/4096) 1.00% [(Remain: 00:01:39, ETA: 9/29/2022 6:26:17 PM)]
        2   82  4096       2 (0082/4096) 2.00% [(Remain: 00:00:49, ETA: 9/29/2022 6:25:27 PM)]
        3  123  4096       3 (0123/4096) 3.00% [(Remain: 00:00:32.3330000, ETA: 9/29/2022 6:25:10 PM)]
        4  164  4096       4 (0164/4096) 4.00% [(Remain: 00:00:24, ETA: 9/29/2022 6:25:02 PM)]
        5  205  4096       5 (0205/4096) 5.00% [(Remain: 00:00:19, ETA: 9/29/2022 6:24:57 PM)]
```

```
  6  246  4096       6 (0246/4096) 6.00% [(Remain: 00:00:15.6670000, ETA: 9/29/2022 6:24:53 PM)]
  7  287  4096       7 (0287/4096) 7.00% [(Remain: 00:00:13.2860000, ETA: 9/29/2022 6:24:51 PM)]
  8  328  4096       8 (0328/4096) 8.00% [(Remain: 00:00:11.5000000, ETA: 9/29/2022 6:24:49 PM)]
  9  369  4096       9 (0369/4096) 9.00% [(Remain: 00:00:10.1110000, ETA: 9/29/2022 6:24:48 PM)]
 10  410  4096      10 (0410/4096) 10.00% [(Remain: 00:00:09, ETA: 9/29/2022 6:24:47 PM)]
 11  451  4096      11 (0451/4096) 11.00% [(Remain: 00:00:08.0910000, ETA: 9/29/2022 6:24:46 PM)]
 12  492  4096      12 (0492/4096) 12.00% [(Remain: 00:00:07.3330000, ETA: 9/29/2022 6:24:45 PM)]
 13  533  4096      13 (0533/4096) 13.00% [(Remain: 00:00:06.6920000, ETA: 9/29/2022 6:24:45 PM)]
 14  574  4096      14 (0574/4096) 14.00% [(Remain: 00:00:12.2860000, ETA: 9/29/2022 6:24:50 PM)]
 15  615  4096      15 (0615/4096) 15.00% [(Remain: 00:00:11.3330000, ETA: 9/29/2022 6:24:49 PM)]
 16  656  4096      16 (0656/4096) 16.00% [(Remain: 00:00:10.5000000, ETA: 9/29/2022 6:24:49 PM)]
 17  697  4096      17 (0697/4096) 17.00% [(Remain: 00:00:09.7650000, ETA: 9/29/2022 6:24:48 PM)]
 18  738  4096      18 (0738/4096) 18.00% [(Remain: 00:00:09.1110000, ETA: 9/29/2022 6:24:47 PM)]
 19  779  4096      19 (0779/4096) 19.00% [(Remain: 00:00:08.5260000, ETA: 9/29/2022 6:24:47 PM)]
 20  820  4096      20 (0820/4096) 20.00% [(Remain: 00:00:08, ETA: 9/29/2022 6:24:46 PM)]
 21  861  4096      21 (0861/4096) 21.00% [(Remain: 00:00:07.5240000, ETA: 9/29/2022 6:24:46 PM)]
 22  902  4096      22 (0902/4096) 22.00% [(Remain: 00:00:07.0910000, ETA: 9/29/2022 6:24:45 PM)]
 23  943  4096      23 (0943/4096) 23.00% [(Remain: 00:00:06.6960000, ETA: 9/29/2022 6:24:45 PM)]
 24  984  4096      24 (0984/4096) 24.00% [(Remain: 00:00:06.3330000, ETA: 9/29/2022 6:24:45 PM)]
 25 1025  4096      25 (1025/4096) 25.00% [(Remain: 00:00:06, ETA: 9/29/2022 6:24:44 PM)]
 26 1066  4096      26 (1066/4096) 26.00% [(Remain: 00:00:05.6920000, ETA: 9/29/2022 6:24:44 PM)]
 27 1107  4096      27 (1107/4096) 27.00% [(Remain: 00:00:05.4070000, ETA: 9/29/2022 6:24:44 PM)]
 28 1148  4096      28 (1148/4096) 28.00% [(Remain: 00:00:05.1430000, ETA: 9/29/2022 6:24:44 PM)]
 29 1189  4096      29 (1189/4096) 29.00% [(Remain: 00:00:04.8970000, ETA: 9/29/2022 6:24:43 PM)]
 30 1230  4096      30 (1230/4096) 30.00% [(Remain: 00:00:04.6670000, ETA: 9/29/2022 6:24:43 PM)]
 31 1271  4096      31 (1271/4096) 31.00% [(Remain: 00:00:04.4520000, ETA: 9/29/2022 6:24:43 PM)]
 32 1312  4096      32 (1312/4096) 32.00% [(Remain: 00:00:04.2500000, ETA: 9/29/2022 6:24:43 PM)]
 33 1353  4096      33 (1353/4096) 33.00% [(Remain: 00:00:04.0610000, ETA: 9/29/2022 6:24:43 PM)]
 34 1394  4096      34 (1394/4096) 34.00% [(Remain: 00:00:03.8820000, ETA: 9/29/2022 6:24:43 PM)]
 35 1435  4096      35 (1435/4096) 35.00% [(Remain: 00:00:03.7140000, ETA: 9/29/2022 6:24:42 PM)]
 36 1476  4096      36 (1476/4096) 36.00% [(Remain: 00:00:03.5560000, ETA: 9/29/2022 6:24:42 PM)]
 37 1517  4096      37 (1517/4096) 37.00% [(Remain: 00:00:03.4050000, ETA: 9/29/2022 6:24:42 PM)]
 38 1558  4096      38 (1558/4096) 38.00% [(Remain: 00:00:03.2630000, ETA: 9/29/2022 6:24:42 PM)]
 39 1599  4096      39 (1599/4096) 39.00% [(Remain: 00:00:03.1280000, ETA: 9/29/2022 6:24:42 PM)]
 40 1640  4096      40 (1640/4096) 40.00% [(Remain: 00:00:03, ETA: 9/29/2022 6:24:42 PM)]
 41 1681  4096      41 (1681/4096) 41.00% [(Remain: 00:00:02.8780000, ETA: 9/29/2022 6:24:42 PM)]
 42 1722  4096      42 (1722/4096) 42.00% [(Remain: 00:00:02.7620000, ETA: 9/29/2022 6:24:42 PM)]
 43 1763  4096      43 (1763/4096) 43.00% [(Remain: 00:00:02.6510000, ETA: 9/29/2022 6:24:42 PM)]
 44 1804  4096      44 (1804/4096) 44.00% [(Remain: 00:00:03.8180000, ETA: 9/29/2022 6:24:43 PM)]
 45 1845  4096      45 (1845/4096) 45.00% [(Remain: 00:00:03.6670000, ETA: 9/29/2022 6:24:43 PM)]
 46 1886  4096      46 (1886/4096) 46.00% [(Remain: 00:00:03.5220000, ETA: 9/29/2022 6:24:43 PM)]
 47 1927  4096      47 (1927/4096) 47.00% [(Remain: 00:00:03.3830000, ETA: 9/29/2022 6:24:42 PM)]
 48 1968  4096      48 (1968/4096) 48.00% [(Remain: 00:00:03.2500000, ETA: 9/29/2022 6:24:42 PM)]
 49 2009  4096      49 (2009/4096) 49.00% [(Remain: 00:00:03.1220000, ETA: 9/29/2022 6:24:42 PM)]
 50 2050  4096      50 (2050/4096) 50.00% [(Remain: 00:00:03, ETA: 9/29/2022 6:24:42 PM)]
 51 2091  4096      51 (2091/4096) 51.00% [(Remain: 00:00:02.8820000, ETA: 9/29/2022 6:24:42 PM)]
 52 2132  4096      52 (2132/4096) 52.00% [(Remain: 00:00:02.7690000, ETA: 9/29/2022 6:24:42 PM)]
 53 2173  4096      53 (2173/4096) 53.00% [(Remain: 00:00:02.6600000, ETA: 9/29/2022 6:24:42 PM)]
 54 2214  4096      54 (2214/4096) 54.00% [(Remain: 00:00:02.5560000, ETA: 9/29/2022 6:24:42 PM)]
 55 2255  4096      55 (2255/4096) 55.00% [(Remain: 00:00:02.4550000, ETA: 9/29/2022 6:24:42 PM)]
 56 2296  4096      56 (2296/4096) 56.00% [(Remain: 00:00:02.3570000, ETA: 9/29/2022 6:24:42 PM)]
 57 2337  4096      57 (2337/4096) 57.00% [(Remain: 00:00:02.2630000, ETA: 9/29/2022 6:24:42 PM)]
 58 2378  4096      58 (2378/4096) 58.00% [(Remain: 00:00:02.1720000, ETA: 9/29/2022 6:24:42 PM)]
 59 2419  4096      59 (2419/4096) 59.00% [(Remain: 00:00:02.0850000, ETA: 9/29/2022 6:24:42 PM)]
 60 2460  4096      60 (2460/4096) 60.00% [(Remain: 00:00:02, ETA: 9/29/2022 6:24:41 PM)]
 61 2501  4096      61 (2501/4096) 61.00% [(Remain: 00:00:01.9180000, ETA: 9/29/2022 6:24:41 PM)]
 62 2542  4096      62 (2542/4096) 62.00% [(Remain: 00:00:01.8390000, ETA: 9/29/2022 6:24:41 PM)]
 63 2583  4096      63 (2583/4096) 63.00% [(Remain: 00:00:01.7620000, ETA: 9/29/2022 6:24:41 PM)]
 64 2624  4096      64 (2624/4096) 64.00% [(Remain: 00:00:01.6880000, ETA: 9/29/2022 6:24:41 PM)]
 65 2665  4096      65 (2665/4096) 65.00% [(Remain: 00:00:01.6150000, ETA: 9/29/2022 6:24:41 PM)]
 66 2706  4096      66 (2706/4096) 66.00% [(Remain: 00:00:01.5450000, ETA: 9/29/2022 6:24:41 PM)]
 67 2747  4096      67 (2747/4096) 67.00% [(Remain: 00:00:01.4780000, ETA: 9/29/2022 6:24:41 PM)]
 68 2788  4096      68 (2788/4096) 68.00% [(Remain: 00:00:01.4120000, ETA: 9/29/2022 6:24:41 PM)]
 69 2829  4096      69 (2829/4096) 69.00% [(Remain: 00:00:01.3480000, ETA: 9/29/2022 6:24:41 PM)]
 70 2870  4096      70 (2870/4096) 70.00% [(Remain: 00:00:01.2860000, ETA: 9/29/2022 6:24:41 PM)]
 71 2911  4096      71 (2911/4096) 71.00% [(Remain: 00:00:01.2250000, ETA: 9/29/2022 6:24:41 PM)]
 72 2952  4096      72 (2952/4096) 72.00% [(Remain: 00:00:01.1670000, ETA: 9/29/2022 6:24:41 PM)]
 73 2993  4096      73 (2993/4096) 73.00% [(Remain: 00:00:01.1100000, ETA: 9/29/2022 6:24:41 PM)]
 74 3034  4096      74 (3034/4096) 74.00% [(Remain: 00:00:01.4050000, ETA: 9/29/2022 6:24:41 PM)]
 75 3075  4096      75 (3075/4096) 75.00% [(Remain: 00:00:01.3330000, ETA: 9/29/2022 6:24:41 PM)]
 76 3116  4096      76 (3116/4096) 76.00% [(Remain: 00:00:01.2630000, ETA: 9/29/2022 6:24:41 PM)]
 77 3157  4096      77 (3157/4096) 77.00% [(Remain: 00:00:01.1950000, ETA: 9/29/2022 6:24:41 PM)]
```

```
   78 3198  4096      78 (3198/4096) 78.00% [(Remain: 00:00:01.1280000, ETA: 9/29/2022 6:24:41 PM)]
   79 3239  4096      79 (3239/4096) 79.00% [(Remain: 00:00:01.0630000, ETA: 9/29/2022 6:24:41 PM)]
   80 3280  4096      80 (3280/4096) 80.00% [(Remain: 00:00:01, ETA: 9/29/2022 6:24:41 PM)]
   81 3321  4096      81 (3321/4096) 81.00% [(Remain: 00:00:00.9380000, ETA: 9/29/2022 6:24:41 PM)]
   82 3362  4096      82 (3362/4096) 82.00% [(Remain: 00:00:00.8780000, ETA: 9/29/2022 6:24:41 PM)]
   83 3403  4096      83 (3403/4096) 83.00% [(Remain: 00:00:00.8190000, ETA: 9/29/2022 6:24:41 PM)]
   84 3444  4096      84 (3444/4096) 84.00% [(Remain: 00:00:00.7620000, ETA: 9/29/2022 6:24:41 PM)]
   85 3485  4096      85 (3485/4096) 85.00% [(Remain: 00:00:00.7060000, ETA: 9/29/2022 6:24:41 PM)]
   86 3526  4096      86 (3526/4096) 86.00% [(Remain: 00:00:00.6510000, ETA: 9/29/2022 6:24:41 PM)]
   87 3567  4096      87 (3567/4096) 87.00% [(Remain: 00:00:00.5980000, ETA: 9/29/2022 6:24:41 PM)]
   88 3608  4096      88 (3608/4096) 88.00% [(Remain: 00:00:00.5450000, ETA: 9/29/2022 6:24:41 PM)]
   89 3649  4096      89 (3649/4096) 89.00% [(Remain: 00:00:00.4940000, ETA: 9/29/2022 6:24:41 PM)]
   90 3690  4096      90 (3690/4096) 90.00% [(Remain: 00:00:00.4440000, ETA: 9/29/2022 6:24:41 PM)]
   91 3731  4096      91 (3731/4096) 91.00% [(Remain: 00:00:00.3960000, ETA: 9/29/2022 6:24:41 PM)]
   92 3772  4096      92 (3772/4096) 92.00% [(Remain: 00:00:00.3480000, ETA: 9/29/2022 6:24:41 PM)]
   93 3813  4096      93 (3813/4096) 93.00% [(Remain: 00:00:00.3010000, ETA: 9/29/2022 6:24:41 PM)]
   94 3854  4096      94 (3854/4096) 94.00% [(Remain: 00:00:00.2550000, ETA: 9/29/2022 6:24:41 PM)]
   95 3895  4096      95 (3895/4096) 95.00% [(Remain: 00:00:00.2110000, ETA: 9/29/2022 6:24:41 PM)]
   96 3936  4096      96 (3936/4096) 96.00% [(Remain: 00:00:00.1670000, ETA: 9/29/2022 6:24:41 PM)]
   97 3977  4096      97 (3977/4096) 97.00% [(Remain: 00:00:00.1240000, ETA: 9/29/2022 6:24:41 PM)]
   98 4018  4096      98 (4018/4096) 98.00% [(Remain: 00:00:00.0820000, ETA: 9/29/2022 6:24:41 PM)]
   99 4059  4096      99 (4059/4096) 99.00% [(Remain: 00:00:00.0400000, ETA: 9/29/2022 6:24:41 PM)]
  100 4096  4096     100 (4096/4096) 100.00% [(Remain: 00:00:00, ETA: 9/29/2022 6:24:41 PM)]
#>

# // ------------------------------------------------------------------------------------------------
# // | Create the LOOP to create 4096 empty text files, use the [System.IO.File] class to           |
# // | ACCELERATE the job, since the Set-Content function takes a while longer to do this at scale. |
# // ------------------------------------------------------------------------------------------------

$Progress                       = $P.Splat()
Write-Progress @Progress
ForEach ($X in 0..($List.Count-1))
{
    If ($X -in $P.Range)
    {
        $P.Increment()
        $Progress               = $P.Splat()
        Write-Progress @Progress
    }

    [System.IO.File]::Create($List[$X]).Dispose()
    [System.IO.File]::WriteAllLines($List[$X],$List[$X])
}
$Progress                   = $P.Splat()
Write-Progress @Progress -Complete

# // ------------------------------------
# // | Instantiate the class (or else...) |
# // ------------------------------------

$Base                       = [FolderHash]::New($Path)
$Base.GetFileHash("SHA256")
```

So NOW, if we use the ol' Batman underground $Base variable, we'll get some stuff that looks like this...

```
<#

------------------------------
| WHAT THE OUTPUT LOOKS LIKE |
------------------------------

PS Prompt:\> $Base

Path                          Algorithm Output
----                          --------- ------
C:\Users\mcadmin\Desktop\Temp SHA256 {000.txt, 001.txt, 002.txt, 003.txt...}

------------------------------------------------------------
| WHAT THE FIRST 101 ENTRIES IN THE OUTPUT TABLE LOOKS LIKE |
------------------------------------------------------------

PS Prompt:\> $Base.Output[0..100] | Format-table
```

Index	Name	LastWriteTime	Length	Algor.	Hash
0	000.txt	9/29/2022 5:56:35 PM	39	SHA256	03F49E97AEB0A4C6CDBBFF22C3F08BCDD11997E961618B0AC65FFD0D7ADDA3FD
1	001.txt	9/29/2022 5:56:35 PM	39	SHA256	11C2554A361357AD69FD6405B3B97D584EC539179E5FAFD5FAF82BB67F04DE25
2	002.txt	9/29/2022 5:56:35 PM	39	SHA256	BE6D3E36EED84DE4C88DFB98B71100827E388E18B683E16B3F250548713AA03F
3	003.txt	9/29/2022 5:56:35 PM	39	SHA256	1569B3C084015EF14489CD10576D71395A69CD3EF39548E5CFD1F14F448BAAA6
4	004.txt	9/29/2022 5:56:35 PM	39	SHA256	B24FD54E81610E9618B3A2BACDC7743000FA545BF27BCC5DD0DB89FC91E36D90
5	005.txt	9/29/2022 5:56:35 PM	39	SHA256	B6C795911E6F0EE2D9FA0AE6153EC949CB6F3FF0C29EB1AFA178DD7E66C3A07F
6	006.txt	9/29/2022 5:56:35 PM	39	SHA256	70BE7FF981F2F9D06406A0D3040C646F5CAA89192C1194370CC15776F8710CBC
7	007.txt	9/29/2022 5:56:35 PM	39	SHA256	F707A9609BFA8C2BC43C763C7E89B937B6D1335E2D5C0FBC2A0A483F14420572
8	008.txt	9/29/2022 5:56:35 PM	39	SHA256	9D09D6ADDD317BEF5AF833E8D812C6559999BA15EF2D2B3D3BED078B0CFB9EBF
9	009.txt	9/29/2022 5:56:35 PM	39	SHA256	A3A19134F187F1BE502219CC0B2F34D47B307309500BEC12F2CAF88902B3EB37
10	00A.txt	9/29/2022 5:56:35 PM	39	SHA256	83AE6442332B7152BE9A0A806890A5950A421697B7E6CC9D509DD93396349D00
11	00B.txt	9/29/2022 5:56:35 PM	39	SHA256	CA66F17CB24F770214DBB5D7D0B5585556DE3C030F5C296CECED554706211149C
12	00C.txt	9/29/2022 5:56:35 PM	39	SHA256	42C3E025C5EB586263AAA622D1ECD0C7BF697A5D5F272CD716C7427FA5CFD95C
13	00D.txt	9/29/2022 5:56:35 PM	39	SHA256	A00F503BA4973FC82B22B60464DCBBCDDBC42C0F80B0B85AC50DF3AD84B1A48E
14	00E.txt	9/29/2022 5:56:35 PM	39	SHA256	C4DE18529CB2D5AB9C7B6CD32E37CC2ED10E9941E73CF93B3142101B512C6F9C
15	00F.txt	9/29/2022 5:56:35 PM	39	SHA256	02D61C24176293CD5BDA32790D5C2AB6659BBF13CE394DACC8E8FF3AEF56EC28
16	010.txt	9/29/2022 5:56:35 PM	39	SHA256	C9CAC1ED169DE3B5D7747D7DFAEF69DCA57964036BC303A419C5AB2B9B859F9B
17	011.txt	9/29/2022 5:56:35 PM	39	SHA256	552755E67C3C69D2FB50575B4E1A989B8912811F636B74EB998A88CCE0CBF093
18	012.txt	9/29/2022 5:56:35 PM	39	SHA256	22DE110A2CC9554025A13BEF37136BAB8D6DF6C207EB26B3CEBC218C090665CC
19	013.txt	9/29/2022 5:56:35 PM	39	SHA256	D1E74C1A8520CA892C239C7F88DB9355A628088FE876837F726E1038E1ABCB77
20	014.txt	9/29/2022 5:56:35 PM	39	SHA256	84A576EB5D008C43586E8AC2C2B135F988334B79BC4893007C37DB7DEFB0B714
21	015.txt	9/29/2022 5:56:35 PM	39	SHA256	86AC1631096934 3506AC5B08803293D5899D6676E01E235376BABFA486046CDF
22	016.txt	9/29/2022 5:56:35 PM	39	SHA256	5A9DA6312977EEB7C623DF2275A132AFCE6AC235B86BB072BB1A5468A6B8265F
23	017.txt	9/29/2022 5:56:35 PM	39	SHA256	10ACB2760C7E79807674EDEA1CC41C01824F81FF4DC77A1801C9373DD4AF1313
24	018.txt	9/29/2022 5:56:35 PM	39	SHA256	F655940646E02E1EB26FA3BA77B5E4DCB07CBE9BA793C47F87827F4346878 38B
25	019.txt	9/29/2022 5:56:35 PM	39	SHA256	72E75756409287F83D7B1147F9995A5B2F8994937ACA7533AFD8CC0E7C3706E9
26	01A.txt	9/29/2022 5:56:35 PM	39	SHA256	43B19D3F44C613350D1126641AD09DB86C120C147F169A5F408FB38CFE9ED061
27	01B.txt	9/29/2022 5:56:35 PM	39	SHA256	C8BB37BEC213348D2E43F620EA66556F562E65E195B35EEA02BA7B6628E8361B
28	01C.txt	9/29/2022 5:56:35 PM	39	SHA256	BD20795A6230235301F548668CD04983859E760DB5E7DAA29798847E9428A6CB
29	01D.txt	9/29/2022 5:56:35 PM	39	SHA256	A36D8B3EA560F1464DB5D5D22E4C540570957BFD548B143388700C746E3211E3
30	01E.txt	9/29/2022 5:56:35 PM	39	SHA256	3EBD2E73A2521597ED6CE33C7E7A33D0633772C842F91B045E6570D6ACD913D5
31	01F.txt	9/29/2022 5:56:35 PM	39	SHA256	B66065602D9553F1FC94A6B813CE86382085108F8844D9A12B6C252CC7EA9C44
32	020.txt	9/29/2022 5:56:35 PM	39	SHA256	F5A94698AD3C870759D89C92052C5C3618719171602C4EAD0F60CB368242C0E9
33	021.txt	9/29/2022 5:56:35 PM	39	SHA256	02BE1C43309EDBF6C61C6DB21A1985B5EFED21958531C3F9892F76B634A137CA
34	022.txt	9/29/2022 5:56:35 PM	39	SHA256	91C4F0B6C236893240BEA389ABA282E1E078A0E285FEDC7806B0ED596946DADF
35	023.txt	9/29/2022 5:56:35 PM	39	SHA256	2103D8021B403E3E5A9349E4ED8A1EF4010CE3FFD8911C5977C5269613B9D60B
36	024.txt	9/29/2022 5:56:35 PM	39	SHA256	EBF9F01A15314B7923688167FA39A8692B00DAEC2FCD2B828180A295E6A90B39
37	025.txt	9/29/2022 5:56:35 PM	39	SHA256	44851D18F74B67A5FF4E35C5CA2F09F815FEC1B84621107849CF8A2C3BC42685
38	026.txt	9/29/2022 5:56:35 PM	39	SHA256	8BAE54065EF49D661B045B80C76E328521A058FE936754D359A12FCD468E9180
39	027.txt	9/29/2022 5:56:35 PM	39	SHA256	645E45E6DF945F431BAC463F2F8BF1C9144488A201385EBDEB7E09B872C7BCC7
40	028.txt	9/29/2022 5:56:35 PM	39	SHA256	21F4479545B518E2D2EA4BB6F5FD0F99F9CD90B1D2DDB11411405DB05BD6EAC6
41	029.txt	9/29/2022 5:56:35 PM	39	SHA256	FBBE4BE9FC448A59C19F589DFF042F2A9B8EF8A8B6AF62D9518CBA104AFD4A38
42	02A.txt	9/29/2022 5:56:35 PM	39	SHA256	48F73CDC2E0B95C705B9BD7E4F7FA710E3A02C6DFE26F25DD29BD5CFFD40B544
43	02B.txt	9/29/2022 5:56:35 PM	39	SHA256	D06EF4834A67A8A415C4A52D540831E9D0CF2B2DBE30BE56FE50F23F42BA971A
44	02C.txt	9/29/2022 5:56:35 PM	39	SHA256	14E93CEE157C2F8A30E95E590DD9C09D2FC582F1D730BD1C77558EB8D3C54D2A
45	02D.txt	9/29/2022 5:56:35 PM	39	SHA256	D1089412E8962F3654C78C11C9416C5E3875AD68DD5DF9D29F791A1349733 68B
46	02E.txt	9/29/2022 5:56:35 PM	39	SHA256	43D9C14B195CB6EE928CCBE3C34A4F3B44E52EB6F38EE22991364C6D2CCE70AE
47	02F.txt	9/29/2022 5:56:35 PM	39	SHA256	66F37EBA9B475967B2E44266D0D5AB5C990DB60AC930A97CF9288400B7D86F28
48	030.txt	9/29/2022 5:56:35 PM	39	SHA256	59BE8FDFEC4FAFC1AABFFE0B5F11C500E08109753BEAAFAAE5418385817E22F4
49	031.txt	9/29/2022 5:56:35 PM	39	SHA256	7EC66FED921DFF7AEC5DBE665A00E7C64B4F5989C887423D4542DE4551032562
50	032.txt	9/29/2022 5:56:35 PM	39	SHA256	5B869D7F9BD8FAF9C360D827C6DD19245FCEEF458D3C3C4C443C3BB14FC82CBE
51	033.txt	9/29/2022 5:56:35 PM	39	SHA256	8554C1BE2E851B67FA84C65B9ABD2C2AD8AAECE006E1D936C8A98367B71571A9
52	034.txt	9/29/2022 5:56:35 PM	39	SHA256	535D1EB6F4FAE1DC40A1509B8C4CBB14BB1FBCDE8FE60B6A1232CFF276FEF66E
53	035.txt	9/29/2022 5:56:35 PM	39	SHA256	8E984D85978FE5210E3407569BA1F9ED16401743DB303D628156A0F3A5673CFC
54	036.txt	9/29/2022 5:56:35 PM	39	SHA256	88582672E9330DE70D13F760CE698E8BBAB765026F60A421AD125536BD0C2F15
55	037.txt	9/29/2022 5:56:35 PM	39	SHA256	441A694FBDC03350DB51A9D68D093BF8CB7BF85621ED89B813D7B7C9735A88D8
56	038.txt	9/29/2022 5:56:35 PM	39	SHA256	38EE63010000A715F25245101531D5EC3E7E32E702E3591D62E8F4232A75559B
57	039.txt	9/29/2022 5:56:35 PM	39	SHA256	8D038AA3967AF3F4A5819FAB163A1A22BBA5EA3B90EA628A88189C7E8A5F745D
58	03A.txt	9/29/2022 5:56:35 PM	39	SHA256	C86B3121B1D985CA5FEB88008A45022EEF28B769CFF68DE1E7332BFEEDC78901
59	03B.txt	9/29/2022 5:56:35 PM	39	SHA256	5DA1AE3A85474E3AE0BE81D37CF7FB6C73EE86716B28E95AFEAE9BDAAE0E6B2C
60	03C.txt	9/29/2022 5:56:35 PM	39	SHA256	9BCACC1FE1D9B3CD5861B2A05F6FDADD72ACAB0FEBD18B7D46D7134C938764 07
61	03D.txt	9/29/2022 5:56:35 PM	39	SHA256	FB0E301B5FC1CD2B4BD488F9780C7748200A368BA5D9C74D873CCECD52929000
62	03E.txt	9/29/2022 5:56:35 PM	39	SHA256	EB68861F3AE5936B10FB04CBF5D991DA2A600EBA17684DF942356E5FDDB48E74
63	03F.txt	9/29/2022 5:56:35 PM	39	SHA256	28BB18339F0C0C19A8A41B79262392BEC17243270F1451537A569E25D6A3DAA7
64	040.txt	9/29/2022 5:56:35 PM	39	SHA256	611209BCF92EF3E7D4A3938BBD3B0F7D484599FC7ECB128E6A8882F56EC87393
65	041.txt	9/29/2022 5:56:35 PM	39	SHA256	A2C2027D8179661A3A4B872C750CC39F9F9BD9FE288BF0675AC8E8AB0CC8FC30
66	042.txt	9/29/2022 5:56:35 PM	39	SHA256	FD9A3E82A5B3AAEBF6B1B1909C0A11D1B0199168D1988C7490CB766E831C99B5
67	043.txt	9/29/2022 5:56:35 PM	39	SHA256	8DE1E43830857D85A99915E8AEDC589BF73DA4D6BD8B3EF414CB512C9F595E2E
68	044.txt	9/29/2022 5:56:35 PM	39	SHA256	D857E53A8A08C58B45674D5D65824C2F6F16D52C8DA3F5040165460429 0D9DAC
69	045.txt	9/29/2022 5:56:35 PM	39	SHA256	52212527C52F1B5112C7438330763A2298AFF0120ED303EB710BC0FF90586807

```
   70 046.txt 9/29/2022 5:56:35 PM     39 SHA256 E3DC0CA0B59B9B2839E96EF68CF8779954F4DF0B705C5D126D970E280681F3BA
   71 047.txt 9/29/2022 5:56:35 PM     39 SHA256 6186AF047AA38088F3CC4C7DDD1E883EE86C0EC076C6D12E094A8889666B340C
   72 048.txt 9/29/2022 5:56:35 PM     39 SHA256 C3B6B54FFDD2E841171609483CED45AB96E9960A3D8753B7714C0145FCCB22CD
   73 049.txt 9/29/2022 5:56:35 PM     39 SHA256 08664B0DBBE1091365EB78AA2E0C76CE1B881D93DD11D205234751D6127120D2
   74 04A.txt 9/29/2022 5:56:35 PM     39 SHA256 49D1CCF8F900DDF680F66E70740E35C5B691C3091E13A7EE24576ECD4051B390
   75 04B.txt 9/29/2022 5:56:35 PM     39 SHA256 B6306BE323313A5C1AA12692EE94BA129030A6B27B9B01507935572657CA5A7B
   76 04C.txt 9/29/2022 5:56:35 PM     39 SHA256 6EBA9DAEF78D559870ACC2D0E184554EFE94839DA202218B05C0851979460794
   77 04D.txt 9/29/2022 5:56:35 PM     39 SHA256 FBC3CF345212B3E0CAC8685C5FAE681F80DBFD4E25829D09C5E2B6F0E83E08FE
   78 04E.txt 9/29/2022 5:56:35 PM     39 SHA256 8A1AC368CB1603FCD967F9548BC5FA7264FA2C7252E680627ECE1F363905FFE2
   79 04F.txt 9/29/2022 5:56:35 PM     39 SHA256 FCC16D56A6091928013D59A607FAA2555D57A615C0862BC99112DA871078039C
   80 050.txt 9/29/2022 5:56:35 PM     39 SHA256 818E8ED1045133DDC53B791650EFC5183B407F4EF7EBE25407BFCB740EAE1190
   81 051.txt 9/29/2022 5:56:35 PM     39 SHA256 E0EFC9A2CBC0385DF6478E7159ACF9CDFB626AACC75478D6FE91358366025DC1
   82 052.txt 9/29/2022 5:56:35 PM     39 SHA256 3E2D4277A1D9895F8E8BACCED01A78B1FAFDEB0A864AE6C1C3442473928A8461
   83 053.txt 9/29/2022 5:56:35 PM     39 SHA256 FA56E39B8D0976A03C88D77F14E1735727714C466A0A54D520F762D15F957CA0
   84 054.txt 9/29/2022 5:56:35 PM     39 SHA256 8B70C6D2D00B0382FF489FEA8EF70CC2BF67322B80CA0B83ED233CC707305891
   85 055.txt 9/29/2022 5:56:35 PM     39 SHA256 7E0AA702849FB94FD471ABB24CC68BD3E0F4CC65863EF14C8493611EE0702C3E
   86 056.txt 9/29/2022 5:56:35 PM     39 SHA256 54A7F8B9BC04B0850294F24C8148D0CA7A35B7B318FD147AC0448A3DDA6B9FE8
   87 057.txt 9/29/2022 5:56:35 PM     39 SHA256 CE692A0480AF2F61AC7AD16EDB0D24614D3DBE9E3D2106C29723CFC1270862D7
   88 058.txt 9/29/2022 5:56:35 PM     39 SHA256 C4636451675D18804CBC49C52C1307BEEB657A93F5FCCF10E06B25AD1E24AAFB
   89 059.txt 9/29/2022 5:56:35 PM     39 SHA256 3C08C097A71AF3E04BC889D75A129A82CD34CA00C274C79F7DF43DD614925195
   90 05A.txt 9/29/2022 5:56:35 PM     39 SHA256 C79B5274B3D496A7C9B1DFA19F49EE0C7C552B111C2EC212EE08EC3E426D034B
   91 05B.txt 9/29/2022 5:56:35 PM     39 SHA256 41801043085C2C0D77FA54295E4D9E92761E534D244DC62B8C3DE641EF754720
   92 05C.txt 9/29/2022 5:56:35 PM     39 SHA256 3B86FC399D7EF138E9799FF7106375B1CE2F761B54DD21E703EDEF3770DFFB78
   93 05D.txt 9/29/2022 5:56:35 PM     39 SHA256 D0CA63AD99C7C5606E9A3302ECC97A4BF500D13B439842A185F2CB925DBD005F
   94 05E.txt 9/29/2022 5:56:35 PM     39 SHA256 EAA656D7A4118CE3504D21C9FEAEB39CDC63D1DE367BF77BBC409C0F7CFAA0B7
   95 05F.txt 9/29/2022 5:56:35 PM     39 SHA256 CB1DE30716229563CB45387A2D78D2683143D893EBC5684E3D89B7E33B655453
   96 060.txt 9/29/2022 5:56:35 PM     39 SHA256 733AD675C7062518C35B6DCAE7488940D542D6674789EC97722655E851A42CA8
   97 061.txt 9/29/2022 5:56:35 PM     39 SHA256 3A182E238996E4C0FDB2D162E421F3B0563CC4E67AD84F319F170390DF12EEC8
   98 062.txt 9/29/2022 5:56:35 PM     39 SHA256 CDC3F241F958805806B673C2B37FE14D5E7164401591E8D90C3E0E309CD33051
   99 063.txt 9/29/2022 5:56:35 PM     39 SHA256 0105DAE062AA754E69EE48B2600A6585D01B18CE75894CF6BBDD8E0FC280FD05
  100 064.txt 9/29/2022 5:56:35 PM     39 SHA256 73887423ED34687191A2067D6E8AABC084627C306DEEA2414518CCDF7D96F012
#>
```

Pretty cool, huh...?

Anyway, I know that SOME people will probably have some questions, so I don't mind answering a few.
STILL, I know that some people will probably say something like...

```
/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\

Somebody : *grumble grumble* You shouldn't make fun of Joe Manchin or Ted Cruz...
           They worked really hard to get where they are.
Me       : LMFAO... *laughs profusely for the next 20 minutes*
Somebody : Dude, you've spent 20 minutes laughing hysterically...
Me       : LMFAO...
           BRO...
           You should be a comedian~!
           I am ABSOLUTELY SERIOUS~!
           You're REALLY GOOD~!
Somebody : I wasn't even trying to be funny...
Me       : Well, I guess you're a real NATURAL THEN~!
Somebody : Joe Manchin and Ted Cruz are really cool guys...
           Obviously, they wouldn't be where they are if they didn't work hard to get where they are.
Me       : *CONTINUES LAUGHING PROFUSELY*

\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/
```

And, that's just the way life goes.
The sad reality is, nah. Some people never work hard at all to get where they are.
Look at Prince Andrew. Perfect example of someone who's never worked hard a single day of his life...?
But somehow, respected by a lot of people for some ridiculous reason.

```
/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\

Prince Andrew : *stupid look on his face* I've NEVER seen that woman in my entire life~!
Reporter      : *shows him the photo of him standing next to Ms. Guiffre* That's YOU right there, isn't it...?
Prince Andrew : That's my DOPPLEGANGER...
                That's NOT me.
Reporter      : Oh, DOPPLEGANGER you say...?
Prince Andrew : Yeh.
                *winces* Dude looks IDENTICAL to me in every way.
```

```
Reporter      : She claims that this is DEFINITELY you...
Prince Andrew : Well, it's definitely NOT me...
Reporter      : So, you're saying that this woman is lying...?
Prince Andrew : Obviously.
                What do I look like, some type of dude that just has sex with 17 year old girls or whatever...?
Reporter      : *raises one eyebrow*
Prince Andrew : *crosses arms*
                That's not me.
                Never in a million years, would I EVER be caught dead, with a girl THAT young...
                So...

\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/¯¯\__/
```

Look, that whole skit right there...? That's how hard Ted Cruz and Joe Manchin have worked, to get where they are. So, if I start laughing PROFUSELY in a PowerShell lesson...?

It's because I'm making every effort to call out a couple of highly respected, yet outright fucking lazy ass lying cocksuckers... since there are a number of people out there that are ALSO highly respected, that would NEVER DARE SAY THOSE WORDS IN PUBLIC...

...and they need like uh... how do you say it...?
Like a LUTHER, the ANGER TRANSLATOR, by Key and Peele.
I'm not gonna rattle off who out there, needs an ANGER TRANSLATOR...?
But I can say with sheer certainty, that I would NEVER say the same shit about CHUCK SCHUMER, or even KIRSTEN GILLIBRAND... So...

The point being...? Some people look a lot more guilty than others, when 19 children and 2 teachers are indiscriminately shot to death my someone that needed serious help, while Ted Cruz was busy being pounded in the butthole by Exxon Mobil's (cash/penis). I talk a lot of shit about this dude in my book...

```
        ____    ____    ________________________________________________________________    ____    ____
    ___//¯¯\\__//¯¯\\__//¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯\\__//¯¯\\__//¯¯\\___
___/    \__/    \__//¯¯¯          Secure Digits Plus LLC (π) | Fighting Entropy        ¯¯¯\\__/    \__/    \___
//¯¯\    /¯¯\    /¯¯\\___                 Top Deck Awareness - Not News                ___//¯¯\    /¯¯\    /¯¯\\
\\__//¯¯\\__//¯¯\\__//¯¯\\_______________________________________________________________//¯¯\\__//¯¯\\__//¯¯\\__//
//¯¯\\__//¯¯\\__//¯¯¯    ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯   ¯¯¯\\__//¯¯\\__//¯¯\\
\\__/    \__//   ________________________________________________________________________________   \\__/    \__//
//¯¯\    /¯¯\\   | Used to be news...? Now its Not News. Not News. Part of the Not News Network |   //¯¯\    /¯¯\\
\\__//¯¯\\__//   ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯   \\__//¯¯\\__//
//¯¯\\__//¯¯\\       ________________________________________________________________________      //¯¯\\__//¯¯\\
\\__/    \__//   ___//¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯\\___   \\__/    \__//
//¯¯\    /¯¯\\__//¯¯¯    Provisional Author: Michael C. Cook Sr. | "The Buck Stops Here"     ¯¯¯\\__//¯¯\    /¯¯\\
\\__/¯¯¯¯    ¯¯¯¯                                                                               ¯¯¯¯    ¯¯¯¯\__//
//   Thesis   : Fox News is not news, it is propaganda. They should change the name to Not News              \\
||   Subject  : (STEM/Science, Technology, Engineering, Mathematics)                                          ||
||              History, Politics, Psychology, Philosophy, Morality, Comedy, Commerce, Industry               ||
||   Conflict : Treason, Corruption, Censorship, Injustice, Identity Theft, Cybercrime, Antitrust, Fraud      ||
||   Affair   : Political/Foreign+Domestic+Local, Social/Personal                                             ||
||   Agency   : NSA/Nat. Sec. Agency, CIA/Central Intel Agency, NIST/Nat. Institute of Standards and Tech.,   ||
||             FBI/Fed. Bureau of Invest., DHS/Dept. of Homeland Sec., CISA/Cybersec. Infra. Sec. Agency,     ||
||             NYSP/New York State Police, SCSO/Saratoga County Sheriffs Office                               ||
\\__________________________________________________________________________________________________________//
 | PDF | https://github.com/mcc85s/FightingEntropy/blob/main/Docs/2022_0823_TDA_Not_News.pdf                |
 | PS1 | https://github.com/mcc85s/FightingEntropy/blob/main/Framing/Write-Book.ps1                          |
 ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯
```

The reason why I've had to write a PROGRAM that PUBLISHES the BOOK, is because every single time that I save the thing to like a bunch of (.doc) files...? They get FLAGGED with a (CIA/SCRIBBLES) beacon, which means that when someone OPENS it, it'll tell somebody at like, IDK, the NATIONAL SECURITY AGENCY, or whatever... that someone opened the file. I'm not even REMOTELY kidding about that.

Here's a folder of a bunch of screenshots I took, of this remote party, adding a SINGLE LINE OF PIXELS to a SINGLE LINE OF WORDS, in literally EVERY word editor out there. OPEN OFFICE, LIBRE OFFICE, MICROSOFT WORD, this happens even on LINUX, so I know it's not MICROSOFT that's doing that.

```
 ____________________________________________________________________________________________________________
 | (Central Intelligence Agency/Vault 7 → EXPLOIT, "SCRIBBLES")                                             |
 | 06/01/22 | Pictures               | https://drive.google.com/drive/folders/1wukJYJanYKBfUX34WIhEeoamnsfuTDlg |
 | 02/15/22 | A Matter of Nat'l Sec. | https://youtu.be/e4VnZObiez8                                             |
 ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

                                                                                                    _______________/
\_________________________________________________________________________________________________/ Instantiation
  Q&A /¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯\
/¯¯¯¯¯

    ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯
```

```
/-----------------------------------------------------------------------------------------------------------------\
| Q: What if I don't feel like COPYING and PASTING all of that ridiculous, badly written code...?                 |
[=================================================================================================================]
| A: Well, I can make things a lot easier by putting it all into a CONTINUOUS script block...                     |
\_________________________________________________________________________________________________________________/
 -----------------------------------------------------------------------------------------------------------------

 _________________________________________________________________________________________________________________
/-----------------------------------------------------------------------------------------------------------------\
| Q: WHEN, though...? I've been sitting around waiting for the last *checks watch* 30 seconds...                  |
|    What gives, dude...? I thought you were SUPPOSEDLY an EXPERT SECURITY ENGINEER or whatever...?               |
|-----------------------------------------------------------------------------------------------------------------|
| A: Well, hold up there, frosted flakes. Even TONY the OVERZEALOUS, HYPER-SON-OF-A-BITCH TIGER has               |
|    to slow the fuck down sometimes. Can't just EXPECT things to be DONE already all the time...                 |
\_________________________________________________________________________________________________________________/
 -----------------------------------------------------------------------------------------------------------------

 _________________________________________________________________________________________________________________
/-----------------------------------------------------------------------------------------------------------------\
| Q: Fine, dude. How long do you think it'll take for you to stop writing this ridiculous Q&A section...?        |
|-----------------------------------------------------------------------------------------------------------------|
| A: I'm about to accomplish a whole, kill-two-birds-with-one-stone phenomenon.                                   |
|    So, that's why it's pretty important, to accumulate some REAL COOL STUFF, called: PATIENCE                  |
|    Even ZACH GALIFIANAKIS would say “Hold your horses, Pablo... Alright...?”                                    |
\_________________________________________________________________________________________________________________/
 -----------------------------------------------------------------------------------------------------------------

 _________________________________________________________________________________________________________________
/-----------------------------------------------------------------------------------------------------------------\
| Q: What if (PAUL MANCHIN/TED CRUZ) doesn't wanna use these lame lookin' classes right here...?                  |
|-----------------------------------------------------------------------------------------------------------------|
| A: Alright, well... those guys are too lazy to even COPY + PASTE STUFF... So...                                 |
|    But, lets have this FANTASY where they DO STUFF that could be considered “PRODUCTIVE”.                      |
|                                                                                                                 |
|    Well, PowerShell is WICKED FLEXIBLE and stuff. So, we can actually WRAP all of these ABOVE                  |
|    classes into a function. I will do that in the next section, below.                                          |
|                                                                                                                 |
|    I'm literally gonna add like 6 lines of code, and then copy + paste ALL of the code that I                  |
|    just wrote...?                                                                                               |
|                                                                                                                 |
|    ...inside of the function. I'm going to remove ALL of the commentary, however.                              |
\_________________________________________________________________________________________________________________/
 -----------------------------------------------------------------------------------------------------------------

 _________________________________________________________________________________________________________________
/-----------------------------------------------------------------------------------------------------------------\
| Q: Why do you seem to have an axe to grind against Ted Cruz and Joe Manchin...?                                 |
|-----------------------------------------------------------------------------------------------------------------|
| A: Because THEIR idea of “working hard”, or being “thorough”, is to bust a nut after having sex with a girl    |
|    for a grand total of about 2 seconds... and then that's when they're basically done.                         |
|                                                                                                                 |
|    That's because a lot of people in the government don't really CARE about stuff.                             |
|    That's not to say that the GOVERNMENT ITSELF is bad...? It comes down to INDIVIDUALS.                       |
|    So for instance, if TED CRUZ is busy being pounded in the butt by Exxon Mobil, instead of investing         |
|    capital into projects that do a better job of SEEKING OUT PEOPLE THAT NEED COUNSELING...?                    |
|                                                                                                                 |
|    Then, I wouldn't call the dude a lazy fuck. As it stands...? 19 kids and 2 teachers died on his watch.      |
|    So I blame him a hell of a lot more than I do GILLIBRAND or SCHUMER for the incident in Buffalo, at TOPS.   |
|    The situations are NOT IDENTICAL IN THE LEAST. The BUFFALO incident involved someone who had a PLAN, and    |
|    the incident was a HATE CRIME. Not the case in UVALDE. UVALDE, and the RENO NEVADA incident were VERY       |
|    SIMILAR, as well as NEWTOWN CT. These incidents are happening because people just suddenly SNAP.            |
|                                                                                                                 |
|    *squinting* Wanna know why...? Read the book.                                                                |
\_________________________________________________________________________________________________________________/
 -----------------------------------------------------------------------------------------------------------------

                                                                                                          _____/
\_________________________________________________________________________________________________________/ Q&A
  [CmdletBinding()] /-----------------------------------------------------------------------------------------------------------\
/-------------------

    Here is the CODE BLOCK, that anybody can HIGHLIGHT, and then COPY + PASTE it into their code editor.
    And, rip it apart...?

    Examine it...?
    Use it...?
    Learn cool shit from it...?
    Do whatever the hell you want to it...
```

Just don't go around telling people that YOU originally wrote this thing...
I won't really care, but somebody WILL, and then they'll say you're a little Ted Cruz, or Joe Manchin.

Copying and pasting somebody else's work/code, and then claiming that YOU wrote it...?
That's the (Ted Cruz/Joe Manchin) mentality, right there, dude.

The point being, give credit where credit is due.
You don't have to be the original author of something to learn something from it, and you can even show it to other people too. You can even say 'Yeah, this MICHAEL C. COOK guy literally taught me some cool shit about PowerShell, and then called Ted Cruz and Joe Manchin a couple of lazy lying cocksuckers who take 5 hour naps every day they go to “work”.'

And, I ENCOURAGE people to do this.
Because at the end of the day...?
I might've used some pretty offensive language...?

But THEIR ACTIONS should be considered FAR MORE OFFENSIVE to people...
...in a country where SUPPOSEDLY, people have this RIGHT to EXPRESS THEMSELVES FREELY.

It's called “The First Amendment”, and I talk about the CONSTITUTION OF THE UNITED STATES OF AMERICA, in the book. People would be TRULY IMPRESSED by my PROFILE, PROLIFIC NATURE, and ABILITY TO WRITE MY OWN ORIGINAL CONTENT ON A DAILY BASIS. ALMOST AS IF I COULD BE GETTING PAID A LOT OF MONEY TO DO THAT...

That may be something for some people to consider... when they make the time to read my book.

```powershell
Function Get-FolderHash
{
    [CmdLetBinding()]
    Param(
        [Parameter(Mandatory,Position=0)][String]$Path,
        [Parameter(          Position=1)][String]$Algorithm="SHA256"
    )

    Class Percent
    {
        [UInt32] $Index
        [UInt32] $Step
        [UInt32] $Total
        [UInt32] $Percent
        [String] $String
        Hidden [String] $Output
        Percent([UInt32]$Index,[UInt32]$Step,[UInt32]$Total)
        {
            $This.Index             = $Index
            $This.Step              = $Step
            $This.Total             = $Total
            $This.Calc()
        }
        Calc()
        {
            $Depth                  = ([String]$This.Total).Length
            $This.Percent           = ($This.Step/$This.Total)*100
            $This.String            = "({0:d$Depth}/{1}) {2:n2}%" -f $This.Step, $This.Total, $This.Percent
        }
        [String] ToString()
        {
            If ($This.Output)
            {
                Return $This.Output
            }
            Else
            {
                Return $This.String
            }
        }
    }

    Class Progress
    {
        [String]  $Activity
```

```
        [Object]    $Status
        [UInt32]   $Percent
        [DateTime]   $Start
        [UInt32]     $Total
        [UInt32]      $Step
        [Object[]]    $Slot
        [UInt32[]]   $Range
        Progress([String]$Activity,[UInt32]$Total)
        {
            $This.Activity          = $Activity
            $This.Start             = [DateTime]::Now
            $This.Total             = $Total
            $This.Step              = [Math]::Round($Total/100)
            $This.Slot              = @( )
            ForEach ($X in 0..100)
            {
                $Count              = @($This.Step * $X;$Total)[$X -eq 100]

                $This.AddSlot($X,$Count,$Total)
            }
            $This.Range             = $This.Slot.Step
            $This.Current()
        }
        AddSlot([UInt32]$Index,[UInt32]$Multiple,[UInt32]$Total)
        {
            $this.Slot             += [Percent]::New($Index,$Multiple,$Total)
        }
        Increment()
        {
            $This.Percent ++
            $This.Current()
        }
        [UInt32] Elapsed()
        {
            Return ([TimeSpan]([DateTime]::Now-$This.Start)).TotalSeconds
        }
        [String] Remain()
        {
            $Remain                 = ($This.Elapsed() / $This.Percent) * (100-$This.Percent)
            $Seconds                = [TimeSpan]::FromSeconds($Remain)
            Return "(Remain: {0}, ETA: {1})" -f $Seconds, ([DateTime]::Now+$Seconds)
        }
        Current()
        {
            $This.Status            = $This.Slot[$This.Percent]
            If ($This.Percent -ne 0)
            {
                $This.Status.Output = "{0} [{1}]" -f $This.Status.String, $This.Remain()
            }
            Else
            {
                $This.Status.Output = $This.Status.String
            }
        }
        SetStatus([Object]$Percent)
        {
            $This.Status            = $Percent
            $This.Percent           = $Percent.Percent
            $This.Current()
        }
        [Hashtable] Splat()
        {
            Return [Hashtable]@{

                Activity            = $This.Activity
                Status              = $This.Status
                Percent             = $This.Percent
            }
        }
    }

    Class FileHash
```

```
{
    [UInt32]           $Index
    [String]            $Name
    [DateTime] $LastWriteTime
    [UInt64]          $Length
    Hidden [String] $Fullname
    [String]       $Algorithm
    [String]            $Hash
    FileHash([UInt32]$Index,[Object]$File)
    {
        $This.Index             = $Index
        $This.Name              = $File.Name
        $this.LastWriteTime     = $File.LastWriteTime
        $This.Length            = $File.Length
        $This.Fullname          = $File.Fullname
    }
    GetFileHash([String]$Type)
    {
        $This.Algorithm         = $Type
        $This.Hash              = Get-FileHash -Path $This.Fullname -Algorithm $This.Algorithm | % Hash
    }
}

Class FolderHash
{
    [String]      $Name
    [String]      $Path
    [String] $Algorithm
    [Object]    $Output
    FolderHash([String]$Path)
    {
        If (!(Test-Path $Path))
        {
            Throw "Invalid path: [$Path]"
        }

        $This.Path              = $Path
        $This.Name              = [DateTime]::Now.toString("yyyy_MMdd-HHmmss")
        $This.Output            = @( )
        $List                   = Get-ChildItem $Path

        $P                      = $This.Progress("Adding [~]",$List.Count)
        $Progress               = $P.Splat()
        Write-Progress @Progress
        ForEach ($X in 0..($List.Count-1))
        {
            If ($X -in $P.Range)
            {
                $P.Increment()
                $Progress       = $P.Splat()
                Write-Progress @Progress
            }

            $This.Add($List[$X])
        }
        $Progress               = $P.Splat()
        Write-Progress @Progress
    }
    [Object] Progress([String]$Activity,[UInt32]$Count)
    {
        If (!$Activity)
        {
            Throw "Must specify an activity or label"
        }
        If ($Count -lt 100)
        {
            Throw "Count must be higher than 100 for the time being..."
        }
        Return [Progress]::New($Activity,$Count)
    }
    Add([Object]$File)
    {
```

```
            If ($File.FullName -in $This.Output.FullName)
            {
                Throw "Exception [!] [$($File.Fullname)] already specified."
            }
            $This.Output           += [FileHash]::New($This.Output.Count,$File)
        }
        GetFileHash([String]$Type)
        {
            If ($Type -notin "SHA1 SHA256 SHA384 SHA512 MACTripleDES MD5 RIPEMD160".Split(" "))
            {
                Throw "Invalid Algorithm"
            }

            $This.Algorithm         = $Type
            $P                      = $This.Progress("Hashing [~]",$This.Output.Count)
            $Progress               = $P.Splat()
            Write-Progress @Progress
            ForEach ($X in 0..($This.Output.Count-1))
            {
                If ($X -in $P.Range)
                {
                    $P.Increment()
                    $Progress       = $P.Splat()
                    Write-Progress @Progress
                }

                $This.Output[$X].GetFileHash($This.Algorithm)
            }
            $Progress               = $P.Splat()
            Write-Progress @Progress -Complete
        }
    }

    If (!(Test-Path $Path))
    {
        New-Item $Path -ItemType Directory
    }

    $List                           = 0..4095 | % { "{0}\{1:X3}.txt" -f $Path, $_ }
    $P                              = [Progress]::New("Creating [~]",$List.Count)
    $Progress                       = $P.Splat()
    Write-Progress @Progress
    ForEach ($X in 0..($List.Count-1))
    {
        If ($X -in $P.Range)
        {
            $P.Increment()
            $Progress               = $P.Splat()
            Write-Progress @Progress
        }

        [System.IO.File]::Create($List[$X]).Dispose()
        [System.IO.File]::WriteAllLines($List[$X],$List[$X])
    }
    $Progress                       = $P.Splat()
    Write-Progress @Progress

    $Base                           = [FolderHash]::New($Path)
    $Base.GetFileHash($Algorithm)
    $Base
}
```

Now, legally speaking...? Nobody is ALLOWED to call either 1) Joe Manchin, or 2) Ted Cruz, a couple of wicked lazy bastards. That's PRACTICALLY against the law... So, don't do it. (I'm just kidding, people SHOULD do this.)

Anyway, look. Now that the function is there...?
We can like, CALL upon it, by capturing it to a VARIABLE, or just call the function, and you'll get the output.
You won't be able to DO anything with the OUTPUT if you DO NOT capture it to a variable...

But if you DO capture it to a variable...?
You could, like, TOTALLY do some real cool stuff with it...

```
$Base                              = Get-FolderHash -Path $Home\Desktop\Temp -Algorithm SHA384

<#
------------------------------
| WHAT THE OUTPUT LOOKS LIKE |
------------------------------

PS Prompt:\> $Base

Name             Path                           Algorithm Output
----             ----                           --------- ------
2022_0929-184226 C:\Users\mcadmin\Desktop\Temp SHA384    {000.txt, 001.txt, 002.txt, 003.txt...}

------------------------------------------------------------
| WHAT THE FIRST 101 ENTRIES IN THE OUTPUT TABLE LOOKS LIKE |
------------------------------------------------------------

PS Prompt:\> $Base.Output[0..100] | Format-Table

Index Name    LastWriteTime        Length Algor. Hash
----- ----    -------------        ------ ------ ----
    0 000.txt 9/29/2022 6:42:23 PM     39 SHA384 11C8BE810A4C334A3FB902FED23C0B7A4E67AB062FB13F913C8B27590C008...
    1 001.txt 9/29/2022 6:42:23 PM     39 SHA384 08CF5EAED13BA7A22564A9B38778C55CF4268048EA14793E3A54BC8ECFFA9...
    2 002.txt 9/29/2022 6:42:23 PM     39 SHA384 3F1FFCE64C88EE74EAF4974C0F6F07523AD1990247547B42D7E2ECD2DD5A2...
    3 003.txt 9/29/2022 6:42:23 PM     39 SHA384 044B3F38F3CAEEB68227BB700A1128410E4B1C5380007042F13F1DB7CF847...
    4 004.txt 9/29/2022 6:42:23 PM     39 SHA384 CEC40508DC4BDAD5217BE26AA58A57363DCF49FF5F064ACA6E7DB53D2C574...
    5 005.txt 9/29/2022 6:42:23 PM     39 SHA384 5EA856AFF5C96C3C2B4774DC48CB4CCAE07731C5D9986C7A0DF839405CAC6...
    6 006.txt 9/29/2022 6:42:23 PM     39 SHA384 93D11647ECB4D1FEEB013AFB8E4B6A455E5DC362E14D93D8737408113D702...
    7 007.txt 9/29/2022 6:42:23 PM     39 SHA384 3F07FBC903CFB9EBB9D7208309FDE7168F2D4D9DA32A5A59536173F230A72...
    8 008.txt 9/29/2022 6:42:23 PM     39 SHA384 C285ACAC54717667CE6A9DBF5A0EE0B20472897210EA50FB1941A2EB63E52...
    9 009.txt 9/29/2022 6:42:23 PM     39 SHA384 C79481EADAE3188C83ACFEC89E58085CFF98A31281F348ACF446222ADB981...
   10 00A.txt 9/29/2022 6:42:23 PM     39 SHA384 CF0142D183BE2BA0086133E856F4880DF4C40D7648C03D7B1C698BD0ACF3D...
   11 00B.txt 9/29/2022 6:42:23 PM     39 SHA384 691C2823F943916B2FF9BDE68FE91736056A8F1F422859463052F2156A5B5...
   12 00C.txt 9/29/2022 6:42:23 PM     39 SHA384 B5295527036B81FA8D988346B7AB4DAF8BE93C3A7DB5EE83D699CB73B6E4E...
   13 00D.txt 9/29/2022 6:42:23 PM     39 SHA384 BF7EA7394F31D2E72B968CBFD3FBF61FD4A408D4310D487E1C95CC74C65B4...
   14 00E.txt 9/29/2022 6:42:23 PM     39 SHA384 D21C145436DABD9D230DC18660F4B42C367680E9A234F343EDFC976944364...
   15 00F.txt 9/29/2022 6:42:23 PM     39 SHA384 6A27657CAA73ACA7218EA23726150911A29D9441208E34CCE338A096E6A08...
   16 010.txt 9/29/2022 6:42:23 PM     39 SHA384 6FBE9A0E0149E10C11BFA957F8BF91E119C2FEF58AF720F20EED983DD0329...
   17 011.txt 9/29/2022 6:42:23 PM     39 SHA384 E52DB99E04B0AB4D6D72D0259D5D93FF0EBA0A455B779BA019B69013F9D4A...
   18 012.txt 9/29/2022 6:42:23 PM     39 SHA384 1BE2438A25815C8C39625D55BE1316A122C06CD2D9D34E1DA312099CD8CF3...
   19 013.txt 9/29/2022 6:42:23 PM     39 SHA384 379376E705DD8D03E0F449E9BD6A89BD11A248DD12D8CAA6EA6D6E5C88DE6...
   20 014.txt 9/29/2022 6:42:23 PM     39 SHA384 C4E2DC722BE399E6858C45618B40A919B3C126A7D32CF852256217F451801...
   21 015.txt 9/29/2022 6:42:23 PM     39 SHA384 CBE6155C359F15D4B68C66C20879477FBC29CCC5A11DF9F5405F223ABC929...
   22 016.txt 9/29/2022 6:42:23 PM     39 SHA384 5512909AAC15A088F376882905E75277152A331320D31DD7870F2926A202F...
   23 017.txt 9/29/2022 6:42:23 PM     39 SHA384 8E443879A7BF5BD9011839433F4F449AFB2924EFDC6F83EF6A7081BC08D0...
   24 018.txt 9/29/2022 6:42:23 PM     39 SHA384 928F4EAFD0A61CC12FE0F54629302D7F3453E268CF2A79EDA297D06BB4E98...
   25 019.txt 9/29/2022 6:42:23 PM     39 SHA384 683050D99EC8FA11951AB2B4083CED57C4B3812B0E044F8AD2AB288E83E21...
   26 01A.txt 9/29/2022 6:42:23 PM     39 SHA384 B096B77D3F55C3C384C3FDFF9760C2C6AD9C4DA6E79B6447F94230A604F6B...
   27 01B.txt 9/29/2022 6:42:23 PM     39 SHA384 B3999219D668EFDBF8D820E166AA0927710F04804278A74B0E9D2FD4E8623...
   28 01C.txt 9/29/2022 6:42:23 PM     39 SHA384 223B013B19C436B3DBE7F97880B7F5CE05BBA2FF3270C4DBD1E9B76DA12AD...
   29 01D.txt 9/29/2022 6:42:23 PM     39 SHA384 72B22A3391150F9A8A0FD5026D8F58B77DF06AB2DAAF7BBCF366BE2BC1CBF...
   30 01E.txt 9/29/2022 6:42:23 PM     39 SHA384 48B06D6AA8AA9046292DB90223C9B8F0DF48B72FA19FBC754A5B3A6A06467...
   31 01F.txt 9/29/2022 6:42:23 PM     39 SHA384 5CA7C43E8441401BC0871AC7FF50B12F4A5CCB714FF4D270D746319AD61C4...
   32 020.txt 9/29/2022 6:42:23 PM     39 SHA384 BCDCC70B9CD06394B067192D94544DFE991CAC1172DEDC1C8C70CFE233133...
   33 021.txt 9/29/2022 6:42:23 PM     39 SHA384 4AD2A8AC42FA33F5BB24C2604EF65FC7FD91FCFAB39D2ECC525F1A427C0E2...
   34 022.txt 9/29/2022 6:42:23 PM     39 SHA384 7FDD3F84863E9EF2DDB21899467B2A0613E98C28B772A7457C31F8F0D4315...
   35 023.txt 9/29/2022 6:42:23 PM     39 SHA384 84CEC4C5D52CECB58E57F99F4930755F338575F0D81F5CCD027873E1E3778...
   36 024.txt 9/29/2022 6:42:23 PM     39 SHA384 A13D7E1D5DA2FED694EE8301505582189BB7C1B32861E0976A31E4927D3DD...
   37 025.txt 9/29/2022 6:42:23 PM     39 SHA384 6A7535EA07654826C60949B276A93526983C6092CAF1E0DE103623CF4A5CD...
   38 026.txt 9/29/2022 6:42:23 PM     39 SHA384 58FB3886274B2D4A81D786C25C9DBB27BC40B8E98C1874DBE8834E8422D0C...
   39 027.txt 9/29/2022 6:42:23 PM     39 SHA384 CBD6BA358C7BA38D77A64505CB8AFB455904A7834B70EA1B1C959ACC58241...
   40 028.txt 9/29/2022 6:42:23 PM     39 SHA384 60348D8B6699752967241074D0D9185D3FF2FA454DE483515BF4DC37995F4...
   41 029.txt 9/29/2022 6:42:23 PM     39 SHA384 683DB7EDCCFC6CDD0EE7083355717F1EFA463F1F4191BDB563FA55895B4EF...
   42 02A.txt 9/29/2022 6:42:23 PM     39 SHA384 F94A73EA33FA1D792C40D3CD46DFF9F8AC873AB1EBCE3DC474C8DD36CD2ED...
   43 02B.txt 9/29/2022 6:42:23 PM     39 SHA384 EDBA222B4702EC1C9620C0D3C760A78CF8680DA912E6B4DCA5D295A743672...
   44 02C.txt 9/29/2022 6:42:23 PM     39 SHA384 3CD2DE881BB49C03734ADE8987029E8352A7BC72D6E30832A962BF5807C2D...
   45 02D.txt 9/29/2022 6:42:23 PM     39 SHA384 C2379944A812D632D829031EE634064A5098D5EF0A990FC8424974 4FBF134C...
   46 02E.txt 9/29/2022 6:42:23 PM     39 SHA384 A1521C782DE35F5977079B5847F45620D670EFA56574280256 48AB954A859...
   47 02F.txt 9/29/2022 6:42:23 PM     39 SHA384 CDFCC49A6B28956662B3446767 40B9D0E1AA08BCAB34E8FD9B2F769975D8F...
   48 030.txt 9/29/2022 6:42:23 PM     39 SHA384 5DDCA04081000234E6E32D5ACBA68CE996E38FCFBE0DA86BB5B482C25D6CF...
   49 031.txt 9/29/2022 6:42:23 PM     39 SHA384 2017B191636105F61C6864748CF45AEB63624C993D73FB976564F1C00CC7B...
   50 032.txt 9/29/2022 6:42:23 PM     39 SHA384 5E7CA9129AA90A031E1419F0B3900DD422B1A537217E8279EC3A43D65AF17...
   51 033.txt 9/29/2022 6:42:23 PM     39 SHA384 97F9BDE760A4C269C243B8CB0B13C48A5BCB66044D6F6C00E7EA8FF5AB964...
```

```
    52 034.txt 9/29/2022 6:42:23 PM     39 SHA384 C01337FEC63D81DB97BB176E80C13E32243A01673205EE7E4615B1516AF4B...
    53 035.txt 9/29/2022 6:42:23 PM     39 SHA384 2E69D5E0A78C2592B09742E9DC39726A16E69DCB33E53F05F50AEE5085FD0...
    54 036.txt 9/29/2022 6:42:23 PM     39 SHA384 2C310AF86FD447E9199002A3EF6254D922F373533E16F5995FC0EF28A8066...
    55 037.txt 9/29/2022 6:42:23 PM     39 SHA384 4E70CC363913EA07D962C560302C5359BB9EDFA35874A3C854FAD0D21EFDF...
    56 038.txt 9/29/2022 6:42:23 PM     39 SHA384 804EB0DED8291EA03517EFD2B270C935C06FF1B5FFE3517BF32C3EA2050EF...
    57 039.txt 9/29/2022 6:42:23 PM     39 SHA384 823B39EFA8058E9CF8E0F1F8E99C544C3B87DFDB786D681902A7573F2FE29...
    58 03A.txt 9/29/2022 6:42:23 PM     39 SHA384 1E7F6B06EDF569311EEED6E786A5CFEFD29ECC445F007473033D8706196B8...
    59 03B.txt 9/29/2022 6:42:23 PM     39 SHA384 4BD704136C09E26FD2960E8478D7D97E49D24592E82F05CED5DEAADFF2AE9...
    60 03C.txt 9/29/2022 6:42:23 PM     39 SHA384 FB3562723510S5A6E8EE61B70A1244C66996A2AABF77D71B22CA6CF979C2C...
    61 03D.txt 9/29/2022 6:42:23 PM     39 SHA384 03865EBFF9879DD4BDD9AB128E187630AEBDC002B41FD0CFEDE2EFD24D9D1...
    62 03E.txt 9/29/2022 6:42:23 PM     39 SHA384 0984651E3E8BF2FAD10B91BC68579950BD71562D8D9A95E939FA3E624B9EF...
    63 03F.txt 9/29/2022 6:42:23 PM     39 SHA384 BCC5E4340E3C70028C65AFFF99DB7E543A959EA523EEA1FBD9A9B20C2D4A8...
    64 040.txt 9/29/2022 6:42:23 PM     39 SHA384 B8FDCA46E17365B1CEEF25EC4B4CFDA049336579784EF28EF8AAFC02784C8...
    65 041.txt 9/29/2022 6:42:23 PM     39 SHA384 2A43A42328A6F7E36C3E47F53F374C982023563CE712BF1014F930B2B852C...
    66 042.txt 9/29/2022 6:42:23 PM     39 SHA384 56F6A646A0BB096A78EF92E3AED56B960108612C31BC88F6253AD5BD5998D...
    67 043.txt 9/29/2022 6:42:23 PM     39 SHA384 33F31247CDA7453265EFCFABE4ED05B7F7A02F82C8C1723C292D543044FC6...
    68 044.txt 9/29/2022 6:42:23 PM     39 SHA384 F7BB85B307D079CA44B0DB0CB5849019A5011E74613C8A5C12AEF4EAEFC70...
    69 045.txt 9/29/2022 6:42:23 PM     39 SHA384 47620A42F5BA4BF93313ED78C1C9233F0ADD02D48AEBBC7A754F51BA683CA...
    70 046.txt 9/29/2022 6:42:23 PM     39 SHA384 485BC512149619EDEC8B3E938B0FE0274BAF660FC9BCF6A8FC0E0B4CDE04E...
    71 047.txt 9/29/2022 6:42:23 PM     39 SHA384 60BEE1577DBA2A2178994ACC85B5E60B5628A47E455B3FA7D4EA2F1720231...
    72 048.txt 9/29/2022 6:42:23 PM     39 SHA384 AC570640E296F471CBCCA153B3ED338183F3DFADFECA139D1DFA7F80A61C9...
    73 049.txt 9/29/2022 6:42:23 PM     39 SHA384 B26F5E66B26A5848B59C472897A6F3EDD9DCB241079A95DD0532F30F089C3...
    74 04A.txt 9/29/2022 6:42:23 PM     39 SHA384 E9F7E50D0B7B051A230539FD3BB118850366B5E07C8B7CDA532909B597597...
    75 04B.txt 9/29/2022 6:42:23 PM     39 SHA384 67A17FF17C9F9F078D099DF5AC7A9A65FF7F8F41B3C7FC823239C10293379...
    76 04C.txt 9/29/2022 6:42:23 PM     39 SHA384 CAFC4E35B8E4D5ED66F91BA770D195BDEC1E9446FC666C9AED0BC3967983A...
    77 04D.txt 9/29/2022 6:42:23 PM     39 SHA384 6AD23104392EFE0526D6D30375DD48DE7D99E44465CAB6F03F29FF74269D3...
    78 04E.txt 9/29/2022 6:42:23 PM     39 SHA384 327BE333470DFBE6A9EA87A75658808BF2AA4873C5A91B6A98C3DACDE4715...
    79 04F.txt 9/29/2022 6:42:23 PM     39 SHA384 B9EE51583FF534C0C7AE64596DF50D6C93E2007A97520AAF2620797CA4C0F...
    80 050.txt 9/29/2022 6:42:23 PM     39 SHA384 7B317F99B242086F21B0F89199EB08847655DDAF7298513FA0E56DB65FE3E...
    81 051.txt 9/29/2022 6:42:23 PM     39 SHA384 D7132DFFA3151E93A53ADC56C6F05C3B526A6EB27F144DBB34A399DD2446D...
    82 052.txt 9/29/2022 6:42:23 PM     39 SHA384 E1ED598315D59ACDC4E95B5A8055379C5B67D68D82C9D5ED905241FC225D6...
    83 053.txt 9/29/2022 6:42:23 PM     39 SHA384 7E2AA6EBFF0FA07B4B828518E5108F53B83701047ABBA7A43E52BE3923319...
    84 054.txt 9/29/2022 6:42:23 PM     39 SHA384 6A65C5ECB0E67E77A578A067F98E56C4C07685D850B0D18BA254C3F9875C5...
    85 055.txt 9/29/2022 6:42:23 PM     39 SHA384 7A6D6DA9CBC1FE6CABF28CC2EBB5D219CB3B600FE3A708706EBB11283BE0B...
    86 056.txt 9/29/2022 6:42:23 PM     39 SHA384 B3FED62DA3D4DAD0ED5483D9B65412756AB2FF1A3CD9A9308EAF147CC7BB5...
    87 057.txt 9/29/2022 6:42:23 PM     39 SHA384 179266BFC794E2AF37D3F683D642701F6FDDD2DB559AD8471917566235F0A...
    88 058.txt 9/29/2022 6:42:23 PM     39 SHA384 BBFC0A4AF46E55CA74C5F17B35E1602EFD49778B9499FCB352D5E5835B219...
    89 059.txt 9/29/2022 6:42:23 PM     39 SHA384 AF06FD816E35B28A024B16ABA62F172DF5FDE51C228DE58A76B966FC168D8...
    90 05A.txt 9/29/2022 6:42:23 PM     39 SHA384 C30EBEDEB9B8E7B2D176225F0E8779D4F85F2AE5D21CA8700F418943DC8CF...
    91 05B.txt 9/29/2022 6:42:23 PM     39 SHA384 F7FE69E3FEE98A6CC9E2EE39D7B0F77F7E09F5A7C86B742094EA8AA49B5CF...
    92 05C.txt 9/29/2022 6:42:23 PM     39 SHA384 B24FC3329F656D3BE190F3C56958E0FB2F194909D2DDD46F2BD8D69A8FA11...
    93 05D.txt 9/29/2022 6:42:23 PM     39 SHA384 61797A95942DB06A445F2E99DF61187E6DBE210041F5D23768BCBECF8E665...
    94 05E.txt 9/29/2022 6:42:23 PM     39 SHA384 C985F3BCB40665DDFA8168F0776E8EDC5B3780DD70441FB214E609C9BE609...
    95 05F.txt 9/29/2022 6:42:23 PM     39 SHA384 F0BF82FCD27EF6EE073EF781D670521341E939EC30A5E3C14793B598624B9...
    96 060.txt 9/29/2022 6:42:23 PM     39 SHA384 34F4C0F61A15A88847D0E3191380B29020709715B5749A79321C235E935904...
    97 061.txt 9/29/2022 6:42:23 PM     39 SHA384 E51143A7B31E6FF2F9E68C35CB96C1BB5AC5C52B27FC40E709CD2E968998B...
    98 062.txt 9/29/2022 6:42:23 PM     39 SHA384 3C36A2C3E4384F230C41D31229EC81E9D8B3868BCB00F142D4138F9385B74...
    99 063.txt 9/29/2022 6:42:23 PM     39 SHA384 496BA4DB13AFC14A7BACE8931405C74EE5D8F2F65F20FF6F843571B688870...
   100 064.txt 9/29/2022 6:42:23 PM     39 SHA384 950BDE31663B58FB965C117C9CE63AC48D4545064127 9E7CCDA9C88DD8F98...
#>
```

SHA384 is slightly longer, but as you can see, it's producing totally different hash values.
I had to “TRUNCATE” each of the output strings (TRUNCATE means to make shorter, somehow)

Also, it's worth noting that I could run this same bit of code on ANY MACHINE using the SAME LOGIC, and get the SAME EXACT RESULT, EACH and EVERY time. So, if the result is NOT the same...? Then, that's an indication that even the most advanced VIRUS PROTECTION or ENDPOINT ENCRYPTION PROTECTION software in the world, can't actually protect against... wanna know why...? That's because if THAT IS HAPPENING...?

It means that there's a FIRMWARE LEVEL ATTACK, HAPPENING. MOST FIRMWARE LEVEL ATTACKS are NEVER DETECTED.
PEGASUS/PHANTOM are FIRMWARE LEVEL ATTACKS that EXPLOIT 100% of EVERY SMARTPHONE IN THE WORLD that is made by either APPLE, or GOOGLE. Though, firmware level attacks are RARE, they're security related for sure.

Like, anybody can run this on any given system, and the hash values will be EXACTLY the same.
That's mainly because there's an individual/unique hash code for every single combination of characters/content.

What THAT means is, it's a UNIQUE SIGNATURE which can be used for FILE INTEGRITY VALIDATION.
So like, if I download Microsoft Windows ISO from Microsoft, right...?

And for whatever reason, Hyper-V tells me that the HASH in the ISO isn't valid...?
That means that someone between MY COMPUTER NETWORK, and MICROSOFT...

...performed a MAN-IN-THE-MIDDLE ATTACK, and most people just aren't intelligent enough to catch those when they happen. That's not really most people's FAULTS... that's just how SNEAKY some MAN-IN-THE-MIDDLE attacks truly are.

The reason that they USUALLY go UNDETECTED, is mainly because people like to lie to one another on a CONSTANT BASIS, sometimes your BEST FRIEND will actually be committing ESPIONAGE to you, and you'll just find it WICKED HARD TO BELIEVE...? Cause, most people will say “Now, why would my BEST FRIEND be doing that...?”

Ask yourself some questions... Don't ask these questions FACETIOUSLY, either.
1) Do you have something that someone else would want...?
2) Are you intelligent enough to detect when people lie to you...?
3) Are you very meticulous about what you do...?
4) Do you ever get the feeling that someone seems to know what you're thinking or doing...?

Those are all REALLY GOOD QUESTIONS TO ASK, but few people ever actually DO that.
The reason being...? Most people give each other the benefit of a doubt, that everybody in society is ALWAYS GOOD, ALL THE TIME, DOESN'T LIE, ALWAYS TELLS THE TRUTH, COULDN'T POSSIBLY BE A LYING DOUCHEBAG...

...but that's what most people think.
In order to NOT BE MOST PEOPLE...
...you have to start asking yourself questions that FEW people EVER ask themselves...
...and that is MOST DEFINITELY SECURITY RELATED.

The IDENTITY THEIVES and CYBERCRIMINALS that guys like JIM BROWNING go after...?
THOSE problems are a MUCH SMALLER COMPONENT of a MUCH LARGER PROBLEM.
Absolute power corrupts absolutely. (← That's the much larger problem)

At some point, once MORE PEOPLE stop DOUBTING ME and my ABILITIES and what I have to say...?
People will realize that I'm literally working a lot harder than TED FUCKIN' CRUZ does, on ANY GIVEN DAY.
But, I still EXPECT plenty of RESISTANCE.

That's why I like to talk about:
1) grabbing a METAPHORICAL baseball bat, and then...
2) METAPHORICALLY breaking somebody's legs when they start to provide this OBVIOUS RESISTANCE.

I do this as a manner of my RHETORIC, but some people get ALARMED by it.
Wanna know who gets ALARMED by METAPHORICAL baseball bats...?
Uh- people with MALICIOUS INTENTIONS. That's who starts to be pretty afraid of my RHETORICAL/METAPHORICAL baseball bats. Ya know...? They don't really care for that type of rhetoric at all... Nah.

The truth is...? It's actually NOT ILLEGAL, to talk about:
1) grabbing a METAPHORICAL baseball bat, and then...
2) METAPHORICALLY breaking somebody's legs when they start to provide this OBVIOUS RESISTANCE...

But, that's because some people really are that fuckin' stupid.
There's literally no reason to be afraid of METAPHORS or RHETORIC, unless you get the feeling that YOU are the PERSON that's being SPOKEN to. It will actually sound pretty funny to people that have no malicious intent at all, but people that start sweating bullets...? They're gonna piss their pants and start saying stuff like “THAT DUDE IS CRAZY~! HE'S BULLYING ME~!” (← That's how fuckin' STUPID they are.)

Because, I am trying to teach people about how many lazy fucks like TED FUCKIN' CRUZ, actually exist.
Ted Cruz is like a COCKROACH, and COCKROACHES are TERRIFYING.
When you see (1) COCKROACH...?
You can bet your ass that there's dozens, hundreds, or thousands of them in a nearby NEST.
COCKROACHES are VERY RARELY by themselves.

And, some people will say “I don't see how any of this is related to POWERSHELL or SECURITY”...
By itself, someone saying that just (1) time in a bunch of comments...?
That's not a CLEAR indication that person has malicious intent, they may just need CLARIFICATION.

However, all of this shit IS definitely SECURITY RELATED, and if someone on a PowerShell Subreddit continues to submit comments that sound like “I fail to see how this is related”, or like, has something NEGATIVE to say about the content...?

Well, once it becomes a PATTERN of behavior...?
That's when it's a good idea to bring up those metaphorical baseball bats, and describing scenarios where someone who thinks they're being slick, begins to realize that “MAYBE IT'S A BAD IDEA TO CONTINUE PROVIDING RESISTANCE, CAUSE THIS DUDE IS ONTO ME...”.

Because, someone WILL detect that shit for CERTAIN, especially if they work with PowerShell a LOT more than your typical (COPY + PASTE) Ted Cruz.

Anyway, I'm gonna take a step back from having to call some people out AHEAD OF TIME...

To the normal, regular, every day user...?
It's a good idea to ask yourself these questions.
Stop thinking like DALE CARNEGIE: HOW TO WIN FRIENDS AND INFLUENCE PEOPLE.
The reason why CYBERCRIMINALS really like that book, is because of how easy it is to lie to people.

When you're a SECURITY ENGINEER, or an APPLICATION DEVELOPER...?
You need to rely on SPECIFICS, EXACTNESS, PRECISION, and ACCURACY.
Sometimes people don't really give a shit about ACCURACY, and a lot of people who think this way, are
POLICE OFFICERS. The LAW ENFORCEMENT system of the GOVERNMENT, as well as the JUSTICE SYSTEM, they don't
really care for ACCURACY. A lot of people in the MEDICAL INDUSTRY, as well as general, all-around COMMERCE...?

They do not fucking care for ACCURACY at all.
Because once people begin to develop a system where they have a CONSISTENTLY HIGH LEVEL OF ACCURACY...?
They're gonna start to realize how many lazy, lying cocksuckers are out there in the world.

Getting hash values from files, allows people who work with DATA, to have a way to VALIDATE the CONTENT
of EACH INDIVIDUAL FILE. So, if the CONTENT of the FILE is the SAME, then the HASH VALUE will MATCH UP.

What that means is this.
If the CONTENT of EACH FILE, is basically the SAME EACH TIME...?
Then the HASH VALUE will be the SAME on ANY COMPUTER.

So, if the HASH VALUE DOES NOT MATCH UP...?
That means that there's a MALICIOUS COCKSUCKER in the middle, performing a MAN-IN-THE-MIDDLE attack,
whether it's a FIRMWARE LEVEL ATTACK by INTEL with MELTDOWN/SPECTRE...?

Or, it's an entire company leaking it's user data to CAMBRIDGE ANALYTICA...?
Or, it's a ROGUE FBI MAN that has a knack for writing custom firmware and then depositing it as a UEFI BIOS
package for the ASUS CORPORATION...
OR, maybe it's a CEO like TIM COOK having some way of distributing ZERO DAY VULNERABILITIES to the NSO GROUP...
...the end result is this.

If someone can make some money off of taking advantage of somebody's ignorance...?
Then, they will always find some way of getting in the middle.
That's just how it is.
Ted Cruz got to be as rich as he is, because an oil corporation tells him what to do.
Politicians aren't supposed to accept bribery because that is ILLEGAL.

There is a LAW that says he's NOT supposed to DO that...?
But there are (0) police officers that are willing to uphold the law EQUALLY, EVENLY, ALL ACROSS THE BOARD,
even though they will fuckin' tell people that and even tell people that they have integrity.

Well, 19 kids and 2 teachers being shot to death because some lazy cocksucker like TED CRUZ gets kickbacks
from Exxon Mobil, is a pretty good reason to start talking about metaphorical baseball bats, as well as lazy
police officers, politicians, and judges... that stand to have their legs metaphorically broken at some point.

In the end, this stuff in PARTICULAR toward the END of this document DOES have relatively very little to do
with PowerShell, specifically. However, here's how it relates.

Suppose a company has SECURITY FOOTAGE to review.
The SECURITY COMPANY has to VALIDATE whether the SECURITY FOOTAGE has been TAMPERED with.

Doing that MANUALLY is IMPOSSIBLE.
You'll need to use SOFTWARE to VALIDATE whether the CONTENT is being CHANGED.

The way that HASHING actually works, is when a FILE is in a CONTIGUOUS STREAM OF DATA.
So, if I were to combine the CONTENT of each individual file here, I'm going to get an individual record
that has been TAMPERED WITH.

PowerShell is a PRETTY FUCKING AWESOME WAY TO DO THAT.
Because, here's how this works.

```
                                                                                                    _____________________/
\___________________________________________________________________________________________________/ [CmdletBinding()]
  Contiguous /-------------------------------------------------------------------------------------------------------------\
/------------
```

I've written an ADDITIONAL CLASS, though, to be perfectly clear, this could EASILY be implemented in the
FolderHash class above. Still, it is good practice to have separate processes rather than to jumble them
TOGETHER in some cases. I won't get into specifics as to how/why that is the case, but it can be.

```
Class Contiguous
```

```
{
    [String]        $Name
    [String]    $Fullname
    [String]        $Path
    [String]   $Algorithm
    [String]        $Hash
    [Object]       $Value
    Contiguous([Object]$InputObject)
    {
        If ($InputObject.GetType().Name -ne "FolderHash")
        {
            Throw "This is not the correct input object type."
        }
        $This.Name        = $InputObject.Name
        $This.Path        = $InputObject.Path
        $This.Algorithm   = $InputObject.Algorithm
        $This.Fullname    = "{0}\{1}-{2}.cat" -f $This.Path, $This.Name, $This.Algorithm
        $Swap             = @{ }
        ForEach ($X in 0..($InputObject.Output.Count-1))
        {
            $Swap.Add($X,$InputObject.Output[$X].Hash)
        }
        $This.Value       = @($Swap[0..($Swap.Count-1)])

        If (Test-Path $This.Fullname)
        {
            [System.IO.File]::Delete($This.Fullname)
        }

        [System.IO.File]::Create($This.Fullname).Dispose()
        [System.IO.File]::WriteAllLines($This.Fullname,$This.Value)

        $This.Hash        = Get-FileHash -Path $This.Fullname -Algorithm $This.Algorithm | % Hash
    }
}

$Cont = [Contiguous]::New($Base)
```

So NOW, I've got that whole $Base variable being passed over to the class “Contiguous”.
I'll explain what this class does.

It literally obtains the HASH VALUE of EACH FILE in the $Base.Output (variable.property)...
...throws em all together into a single file that uses the BASE PATH, BASE NAME, and BASE ALGORITHM...
...to provide a unique name for the output file...
...and then it writes all of those hash values to that particular file...
...and then once all of that is TOGETHER...?
...it like, fuckin', you'd never believe it...
...it gets the hash value from THAT file.

So it's like, a dude named Michael, who's WAAAAAAAAAY COOLER than TED CRUZ on any given day, lookin' in the mirror, and saying:

/¯¯__/¯¯\

Michael : Mirror mirror, on the wall...
Mirror : Yeh...?
          Sup, dude...?
Michael : Just had a question to ask ya...
Mirror : Go ahead, dude.
Michael : Alright.
          I know that you're gonna be a real straight shooter here if I ask you this question...
Mirror : Damn straight, dude.
          You know I don't play any fuckin' games, bruh.
Michael : Who's the FAIREST of them all...?
Mirror : Oh, wow.
          You know...?
          This grandmaster <Hash value>, is the fairest of them all, dude.
Michael : Yeah.
          That's what I thought...
Mirror : That's cause I am a mirror, my good friend.
Michael : I'm impressed.

```
 Mirror  : Good.
           You SHOULD be, cause I'm literally tellin' ya how it really is.

 \__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/--\__/
```

Now, check it out. Here's the output from that class up above...

```
 ------------------------------
 | WHAT THE OUTPUT LOOKS LIKE |
 ------------------------------


 Name      : 2022_0929-184226
 Fullname  : C:\Users\mcadmin\Desktop\Temp\2022_0929-184226-SHA384.cat
 Path      : C:\Users\mcadmin\Desktop\Temp
 Algorithm : SHA384
 Hash      : BF1B1E5C081B827071A58C2EEA9A4B35DAF0BB5D86D66D45C8E97555D380A59D55F422A78F540A5C1831E19AE572481B
 Value     : {11C8BE810A4C334A3FB902FED23C0B7A4E67AB062FB13F913C8B27590C008F860072D0BC8FE22C183ACEC6990B8F459F...}
```

The class literally has ALL of the hash values in the $Cont.Value (variable/property).
That's (4096) entries. Here's a VIDEO I made, of ME, doing this whole process.

09/29/22	Script	https://github.com/mcc85s/FightingEntropy/blob/main/Scripts/ProgressLesson.ps1
09/29/22	Video	https://youtu.be/K738wZSKcjo

Anyway, that is it for now, as I am nearing completion of this utility that I've had to take a bit of a break from developing, because of how fucking complicated it is...

```
-------------------------------------------------------------------------------------------------------
| Threading Dispatcher objects | Logging | Status objects | Console objects | Error handling |
| Xaml Objects | Dynamic threading capabilities | Archiving | Event logs | Runspacing |-------
| Advanced kitchen sink blueprint design | etc. |--------------------------------------
-------------------------------------------------
```


Michael C. Cook Sr.
Security Engineer
Secure Digits Plus LLC