Data Science for Digital Forensics & Incident Response

ODSC West ‘21 - Workshop

Jess Garcia

One eSecurity – Founder | SANS – Senior Instructor

jess@one-esecurity.com - @j3ssgarcia

$ whoami

Jess Garcia

@j3ssgarcia

- **Founder and CEO of One eSecurity, a global Digital Forensics and Incident Response (DFIR) company (~15 years).**
- **Leader of the DS4N6 project. Visit: www.ds4n6.io**
- **Senior Instructor at the SANS Institute (~20 years).**

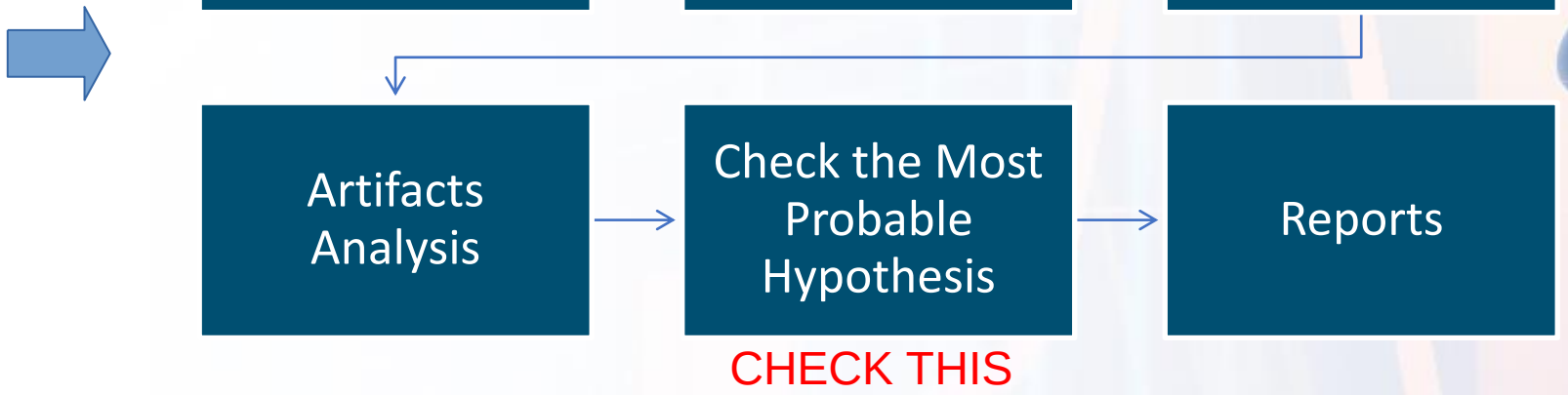
DFIR


SANS DFIR
DIGITAL FORENSICS & INCIDENT RESPONSE
SANSForensics
@SANSForensics
dfir.to/DFIRCast
DFIR
OPERATING SYSTEM & DEVICE IN-DEPTH
FOR308
Digital Forensics Essentials
FOR498
Battlefield Forensics & Data Acquisition
GBFA
FOR500
Windows Forensic Analysis
GCFE
FOR518
Mac and iOS Forensic Analysis & Incident Response
FOR585
Smartphone Forensic Analysis In-Depth
GASF
INCIDENT RESPONSE & THREAT HUNTING
FOR508
Advanced Incident Response, Threat Hunting, & Digital Forensics
GCFA
FOR572
Advanced Network Forensics: Threat Hunting, Analysis, & Incident Response
GNFA
FOR578
Cyber Threat Intelligence
GCTI
FOR610
REM: Malware Analysis Tools & Techniques
GREM
SEC504
Hacker Tools, Techniques, Exploits, & Incident Handling
GCIH

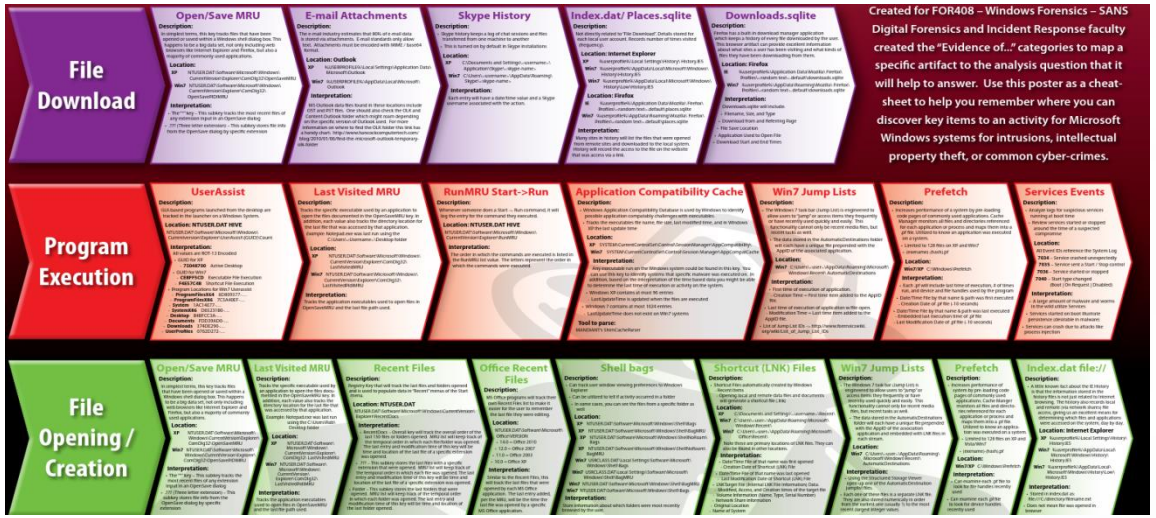
Forensic Investigation

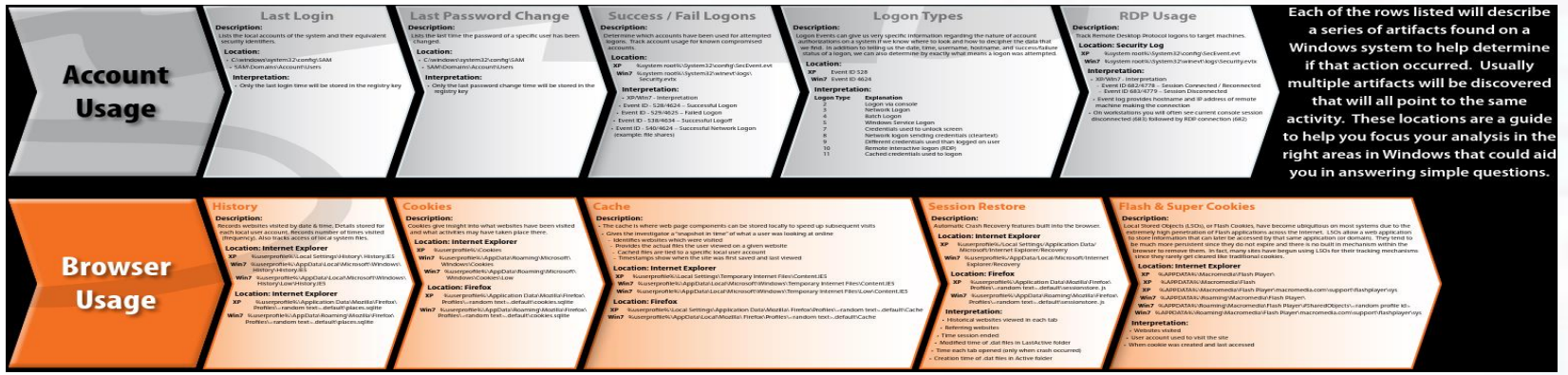
Answer the 5Ws

Who
What
When
Where
Why



Windows Artifacts Categories





Download the SANS poster here: **https://www.sans.org/posters/windows-forensic-analysis/**

Windows Forensic Tools

plaso

- Parse logs and artifacts
- Correlated supertimeline

https://github.com/log2timeline/plaso

fls & mactime

- FLS: lists files and directories
- Mactime: timeline based on FLS

https://github.com/sleuthkit/sleuthkit

SANS

SANS Faculty DFIR Tools

SIFT WORKSTATION

Eric Zimmerman's TOOLS

DS4N6

Find more free SANS tools:
https://www.sans.org/img/free-faculty-tools.pdf

Other Open Source Tools

Velociraptor

GRR RAPID RESPONSE

timesketch Digital Forensics Timeline Analysis

VOLATILITY

Comercial Tools

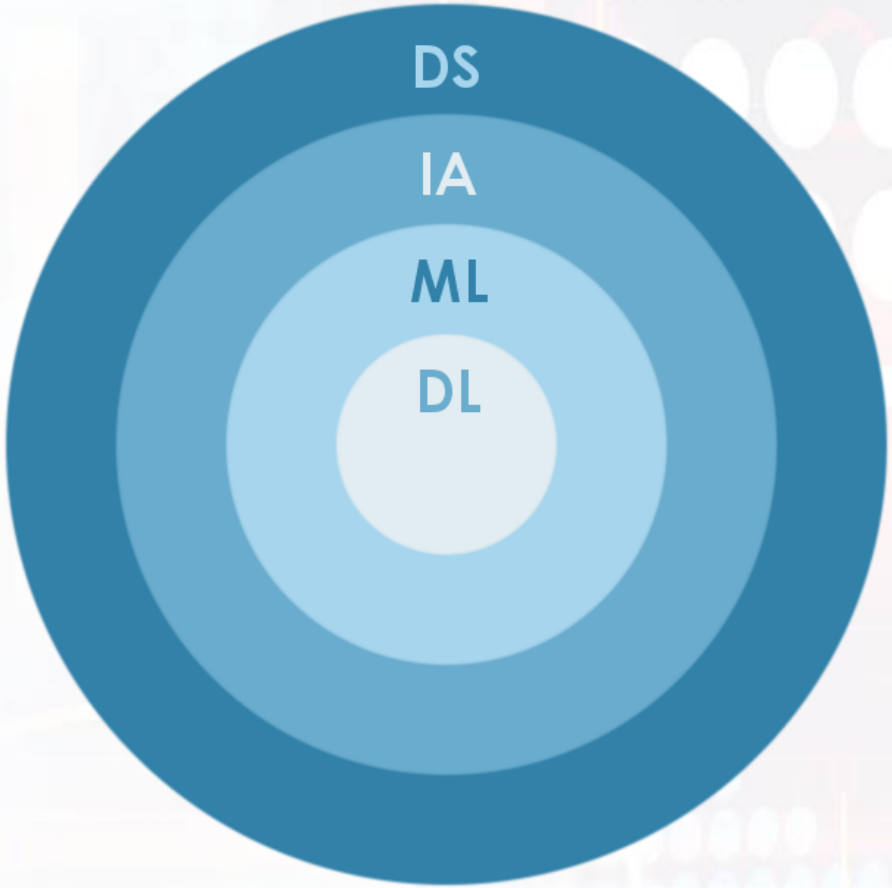
MAGNET FORENSICS

EnCase Forensic

ACCESSDATA ForensicToolkit (FTK)

Intro to Data Science

Analyze faster	Scalable
Flexible	Visualization

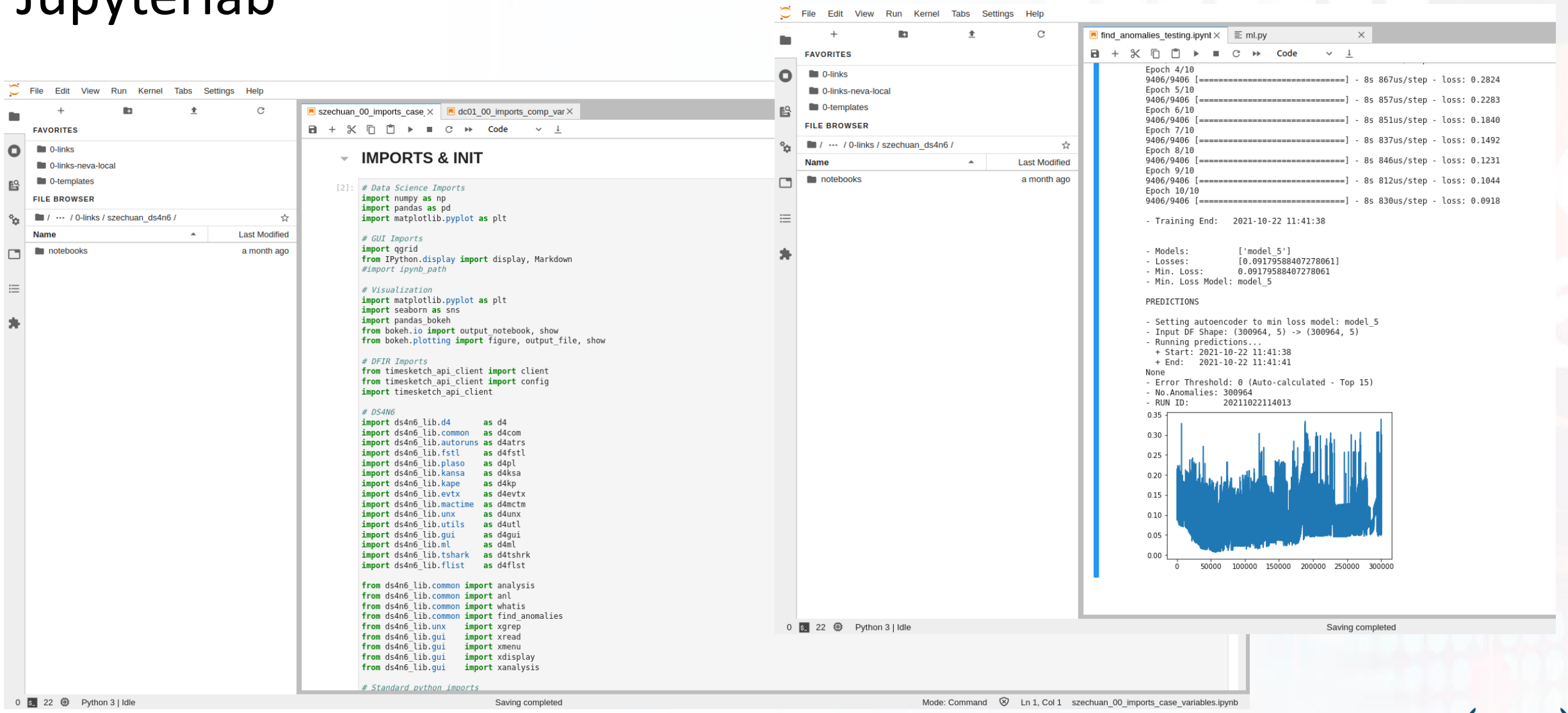


Used in Finance, Medicine, Marketing, Retail...

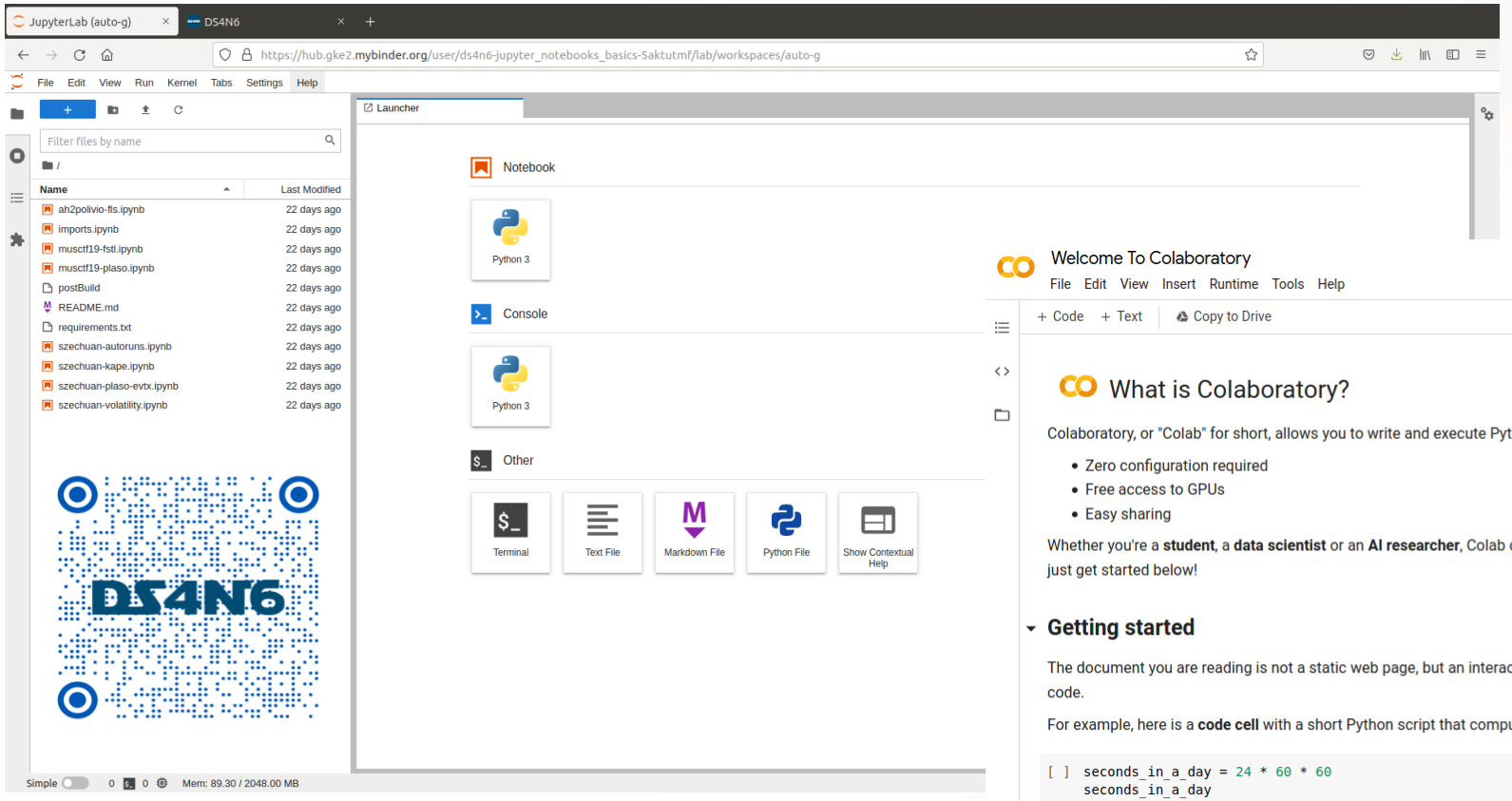
And many more!

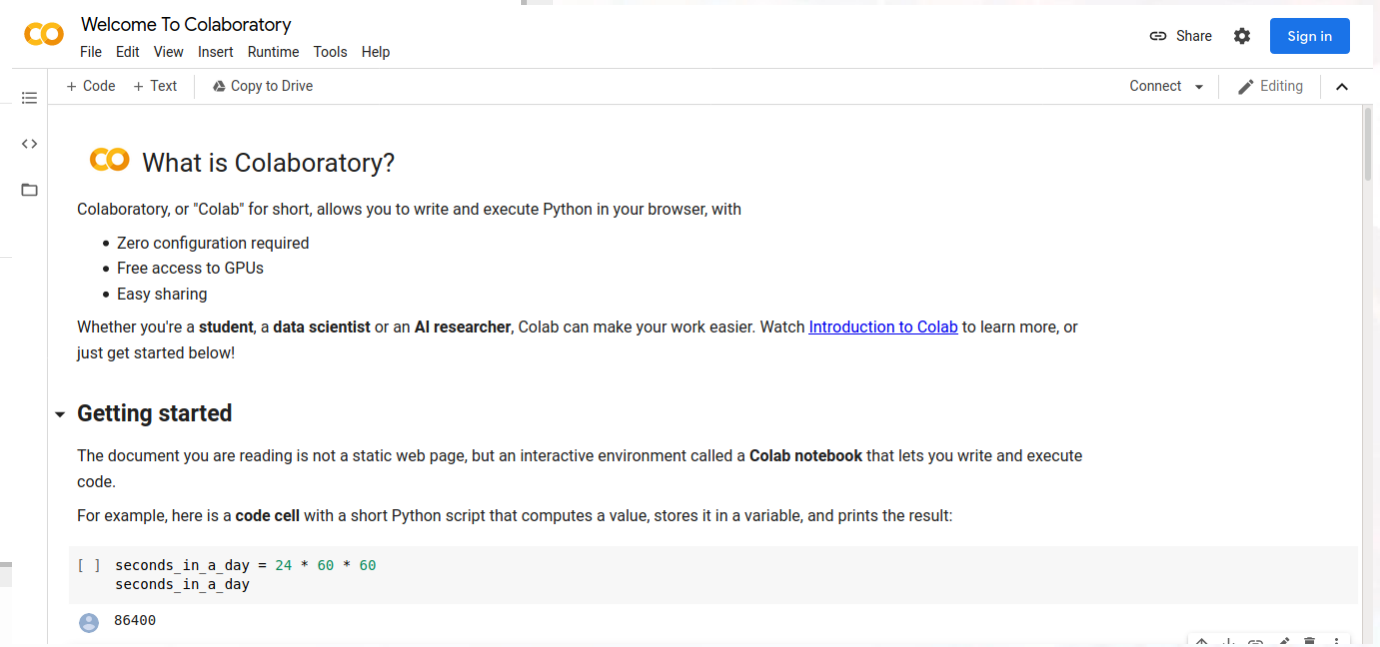
So... What about using it on DFIR?

Jupyterlab


File Edit View Run Kernel Tabs Settings Help
FAVORITES
0-links
0-links-neva-local
0-templates
FILE BROWSER
/ ··· / 0-links / szechuan_ds4n6 /
Name
Last Modified
notebooks
a month ago
szechuan_00_imports_case
dc01_00_imports_comp_var
Code
IMPORTS & INIT
[2]:
# Data Science Imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# GUI Imports
import qgrid
from IPython.display import display, Markdown
#import ipynb_path
# Visualization
import matplotlib.pyplot as plt
import seaborn as sns
import pandas_bokeh
from bokeh.io import output_notebook, show
from bokeh.plotting import figure, output_file, show
# DFIR Imports
from timesketch_api_client import client
from timesketch_api_client import config
import timesketch_api_client
# DS4N6
import ds4n6_lib.d4 as d4
import ds4n6_lib.common as d4com
import ds4n6_lib.autoruns as d4atrs
import ds4n6_lib.fstl as d4fstl
import ds4n6_lib.plaso as d4pl
import ds4n6_lib.kansa as d4ksa
import ds4n6_lib.kape as d4kp
import ds4n6_lib.evtx as d4evtx
import ds4n6_lib.mactime as d4mctm
import ds4n6_lib.unx as d4unx
import ds4n6_lib.utils as d4utl
import ds4n6_lib.gui as d4gui
import ds4n6_lib.ml as d4ml
import ds4n6_lib.tshark as d4tshrk
import ds4n6_lib.flist as d4flst
from ds4n6_lib.common import analysis
from ds4n6_lib.common import anl
from ds4n6_lib.common import whatis
from ds4n6_lib.common import find_anomalies
from ds4n6_lib.unx import xgrep
from ds4n6_lib.gui import xread
from ds4n6_lib.gui import xmenu
from ds4n6_lib.gui import xdisplay
from ds4n6_lib.gui import xanalysis
# Standard python imports
0 22 Python 3 | Idle
Saving completed
Mode: Command
Ln 1, Col 1
szechuan_00_imports_case_variables.ipynb
find_anomalies_testing.ipynt
ml.py
Epoch 4/10
9406/9406 [==========] - 8s 867us/step - loss: 0.2824
Epoch 5/10
9406/9406 [==========] - 8s 857us/step - loss: 0.2283
Epoch 6/10
9406/9406 [==========] - 8s 851us/step - loss: 0.1840
Epoch 7/10
9406/9406 [==========] - 8s 837us/step - loss: 0.1492
Epoch 8/10
9406/9406 [==========] - 8s 846us/step - loss: 0.1231
Epoch 9/10
9406/9406 [==========] - 8s 812us/step - loss: 0.1044
Epoch 10/10
9406/9406 [==========] - 8s 830us/step - loss: 0.0918
- Training End: 2021-10-22 11:41:38
- Models: ['model_5']
- Losses: [0.09179588407278061]
- Min. Loss: 0.09179588407278061
- Min. Loss Model: model_5
PREDICTIONS
- Setting autoencoder to min loss model: model_5
- Input DF Shape: (300964, 5) -> (300964, 5)
- Running predictions...
+ Start: 2021-10-22 11:41:38
+ End: 2021-10-22 11:41:41
None
- Error Threshold: 0 (Auto-calculated - Top 15)
- No.Anomalies: 300964
- RUN ID: 20211022114013
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
50000
100000
150000
200000
250000
300000

Doing DS on the Cloud: binder & colab





Visit bit.ly/3Ff2VOm Or scan the QR code to test the DS4N6 repository

Data Structures: Dataframes

```
df.head()
```

Timestamp	D4_DataType_	D4_Orchestrator_	D4_Tool_	D4_Plugin_	D4_Hostname_	EventRecordID	EventID_	evtxFileName_	ProviderName	ProviderGuid	System > EventID	Version	Level	Task	Opcode	Keywords	ProcessID	Threa
2020-09-18 05:41:16	evtx	NaN	plaso	windows_evtx_record	NaT	16	4624	Security.evtx	Microsoft-Windows-Security-Auditing	{54849625-5478-4994-A5BA-3E3B0328C30D}	4624	2	0	12544	0	0x8020000000000000	660	6
2020-09-18 05:41:17	evtx	NaN	plaso	windows_evtx_record	NaT	47	4624	Security.evtx	Microsoft-Windows-Security-Auditing	{54849625-5478-4994-A5BA-3E3B0328C30D}	4624	2	0	12544	0	0x8020000000000000	660	7
2020-09-18 05:41:17	evtx	NaN	plaso	windows_evtx_record	NaT	50	4624	Security.evtx	Microsoft-Windows-Security-Auditing	{54849625-5478-4994-A5BA-3E3B0328C30D}	4624	2	0	12544	0	0x8020000000000000	660	7
2020-09-18 05:41:18	evtx	NaN	plaso	windows_evtx_record	NaT	51	4624	Security.evtx	Microsoft-Windows-Security-Auditing	{54849625-5478-4994-A5BA-3E3B0328C30D}	4624	2	0	12544	0	0x8020000000000000	660	7
2020-09-18 05:41:18	evtx	NaN	plaso	windows_evtx_record	NaT	54	4624	Security.evtx	Microsoft-Windows-Security-Auditing	{54849625-5478-4994-A5BA-3E3B0328C30D}	4624	2	0	12544	0	0x8020000000000000	660	7

Data Structures: Dictionaries

```
pldfs_srv.keys()
```

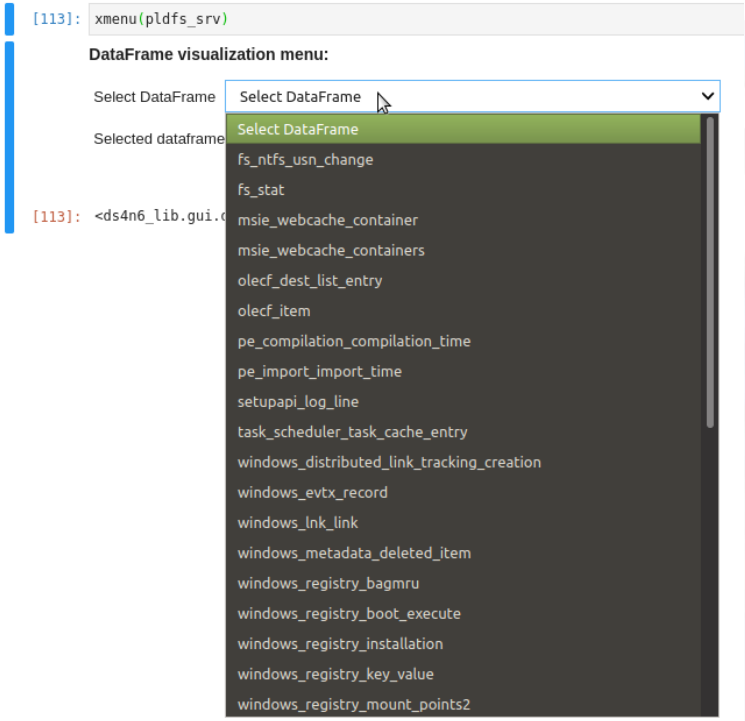
```
dict_keys(['windows_registry_installation', 'windows_registry_userassist', 'windows_lnk_link', 'fs_stat', 'olecf_item', 'pe_compilation_compilation_time', 'windows_registry_key_value',
'windows_shell_item_file_entry', 'windows_distributed_link_tracking_creation', 'task_scheduler_task_cache_entry', 'windows_evtx_record', 'windows_registry_network', 'fs_ntfs_usn_change
', 'setupapi_log_line', 'windows_registry_run', 'windows_registry_msie_zone_settings', 'windows_registry_mount_points2', 'windows_registry_bagmru', 'windows_registry_boot_execute', 'win
dows_registry_timezone', 'windows_registry_service', 'windows_registry_winlogon', 'windows_registry_sam_users', 'olecf_dest_list_entry', 'msie_webcache_container', 'windows_registry_mru
list', 'msie_webcache_containers', 'windows_registry_shutdown', 'windows_registry_usb', 'windows_registry_typedurls', 'windows_registry_mrulistex', 'windows_metadata_deleted_item', 'win
dows_registry_mstsc_connection', 'windows_registry_mstsc_mru', 'pe_import_import_time'])
```

```
pldfs_srv['task_scheduler_task_cache_entry']['Timestamp_'].iloc[0]
```

```
Timestamp('2013-08-10 14:48:44.446487')
```

```
pldfs_srv['task_scheduler_task_cache_entry'][['Timestamp_','task_name']].iloc[0:10]
```

	Timestamp_	task_name
0	2013-08-10 14:48:44.446487	Plug and Play Cleanup
1	2013-08-16 14:48:44.477735	Sqm-Tasks
2	2013-08-16 14:48:44.649610	SynchronizeTimeZone
3	2013-08-17 14:48:44.758985	KernelCeipTask
4	2013-08-18 14:48:43.837110	SmartScreenSpecific
5	2013-08-18 14:48:44.743361	AnalyzeSystem
6	2013-08-18 14:48:44.758985	UsbCeip
7	2013-08-19 14:48:44.087110	SQM data sender
8	2013-08-19 14:48:44.149611	SynchronizeTime
9	2013-08-19 14:48:44.321488	Metadata Refresh



Pandas

- Most used DS library in Python
- Other commonly used are:
  - Numpy
  - SciPy
  - Scikit-Learn
  - Keras
  - Matplotlib

```
df.loc["2020-09-18 05:41:16":"2020-09-18 05:41:18", ["TargetUserSid", "Computer"]]
```

Timestamp	TargetUserSid	Computer
2020-09-18 05:41:16	S-1-5-18	WIN-2IH1TBB9I4Q
2020-09-18 05:41:17	S-1-5-18	WIN-2IH1TBB9I4Q
2020-09-18 05:41:17	S-1-5-96-0-0	WIN-2IH1TBB9I4Q
2020-09-18 05:41:18	S-1-5-20	WIN-2IH1TBB9I4Q
2020-09-18 05:41:18	S-1-5-96-0-1	WIN-2IH1TBB9I4Q

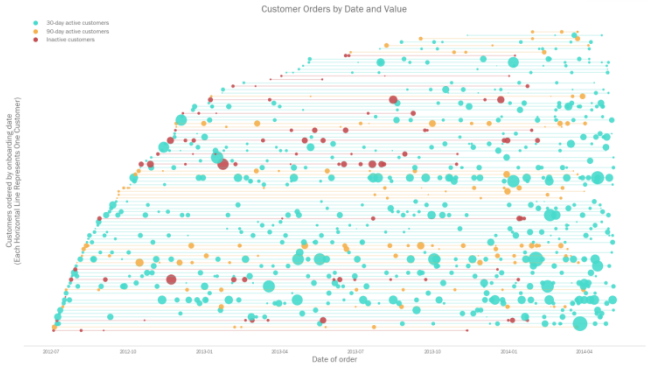
```
df['Computer'].iloc[0]
```

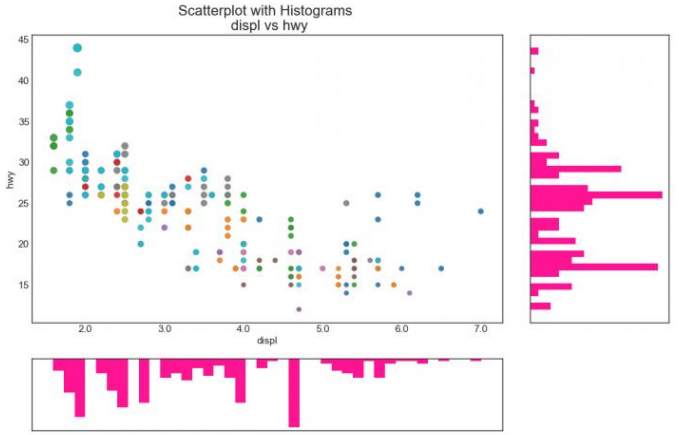
```
'WIN-2IH1TBB9I4Q'
```

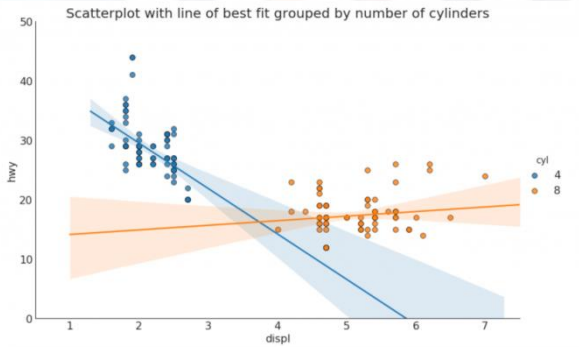
```
df.query('TargetUserSid == "S-1-5-18"').head()
```

SubjectUserName	SubjectDomainName	SubjectLogonId	TargetUserSid	TargetUserName	TargetDomainName	TargetLogonId	@CorrelationActivityID	LogonType	LogonProcessName
-	-	0x0000000000000000	S-1-5-18	SYSTEM	NT AUTHORITY	0x00000000000003e7	<NA>	0	-
MINWINPC$	<NA>	0x00000000000003e7	S-1-5-18	SYSTEM	NT AUTHORITY	0x00000000000003e7	{4B73C7C8-8D7E-0001-15CA-734B7E8DD601}	5	Advapi
MINWINPC$	<NA>	0x00000000000003e7	S-1-5-18	SYSTEM	NT AUTHORITY	0x00000000000003e7	{4B73C7C8-8D7E-0001-15CA-734B7E8DD601}	5	Advapi
MINWINPC$	<NA>	0x00000000000003e7	S-1-5-18	SYSTEM	NT AUTHORITY	0x00000000000003e7	{4B73C7C8-8D7E-0001-15CA-734B7E8DD601}	5	Advapi
MINWINPC$	<NA>	0x00000000000003e7	S-1-5-18	SYSTEM	NT AUTHORITY	0x00000000000003e7	{4B73C7C8-8D7E-0001-15CA-734B7E8DD601}	5	Advapi

Matplotlib

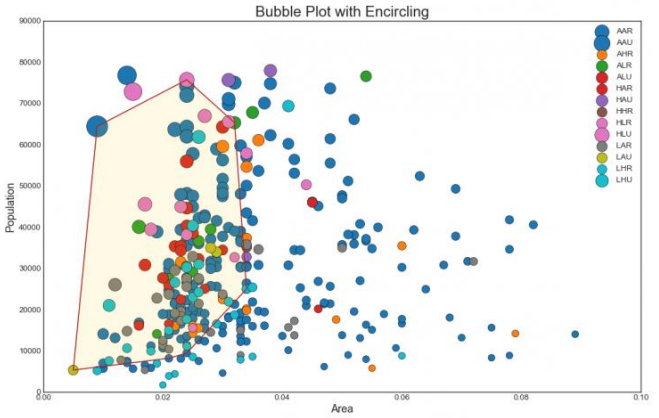


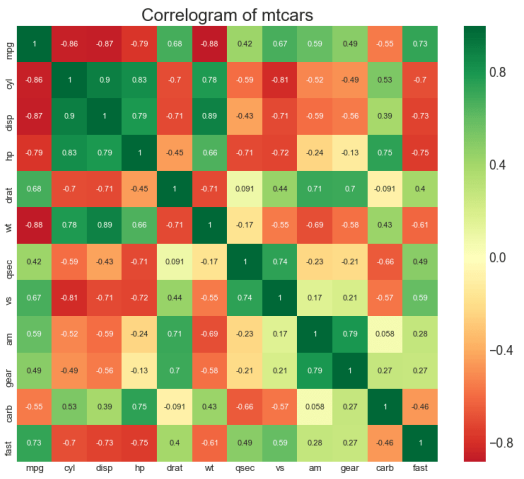




Other visualization libraries:

- seaborn
- plotly
- bokeh
- NetworkX
- altair





DS4N6. The Road So Far

DS4N6

CHRYSALIS

D4ML

EVTX
PF
MFT
REG
WEB

HAM

ADAM

ADversAry eMulator

DAISY

DAISY VM

CHRYSALIS

Function	Usage	Type	Description
whatis()	whatis(obj)	CLI	Identifies the forensic data type of an object (DataFrame -df- or DataFrame Collection -dfs-)
xread()	xread(options)	GUI	Reads tool output data (e.g. plaso output) and stores it in a df/dfs
xmenu()	xmenu(obj)	GUI	Used to easily select a dataframe from dfs, or a column from a df, displaying the selected data and allowing manual (Excel-like) analysis on it
xanalysis()	xanalysis(obj, options)	GUI	Displays a menu with the advanced analysis functions available for the data type (i.e. forensic artifact) given
xdisplay()	xdisplay()	GUI	Used to select the display settings for the dataframes that will be displayed (max. rows, max. columns, etc.)
simple()	df.simple(options)	CLI	Simplifies forensic output (df) showing only the most interesting columns for analysis.
xgrep()	xgrep(obj, options)	CLI	UNIX-like grep for the DataFrame world. Allows the user to search for a regular expression in a DF column or full DF

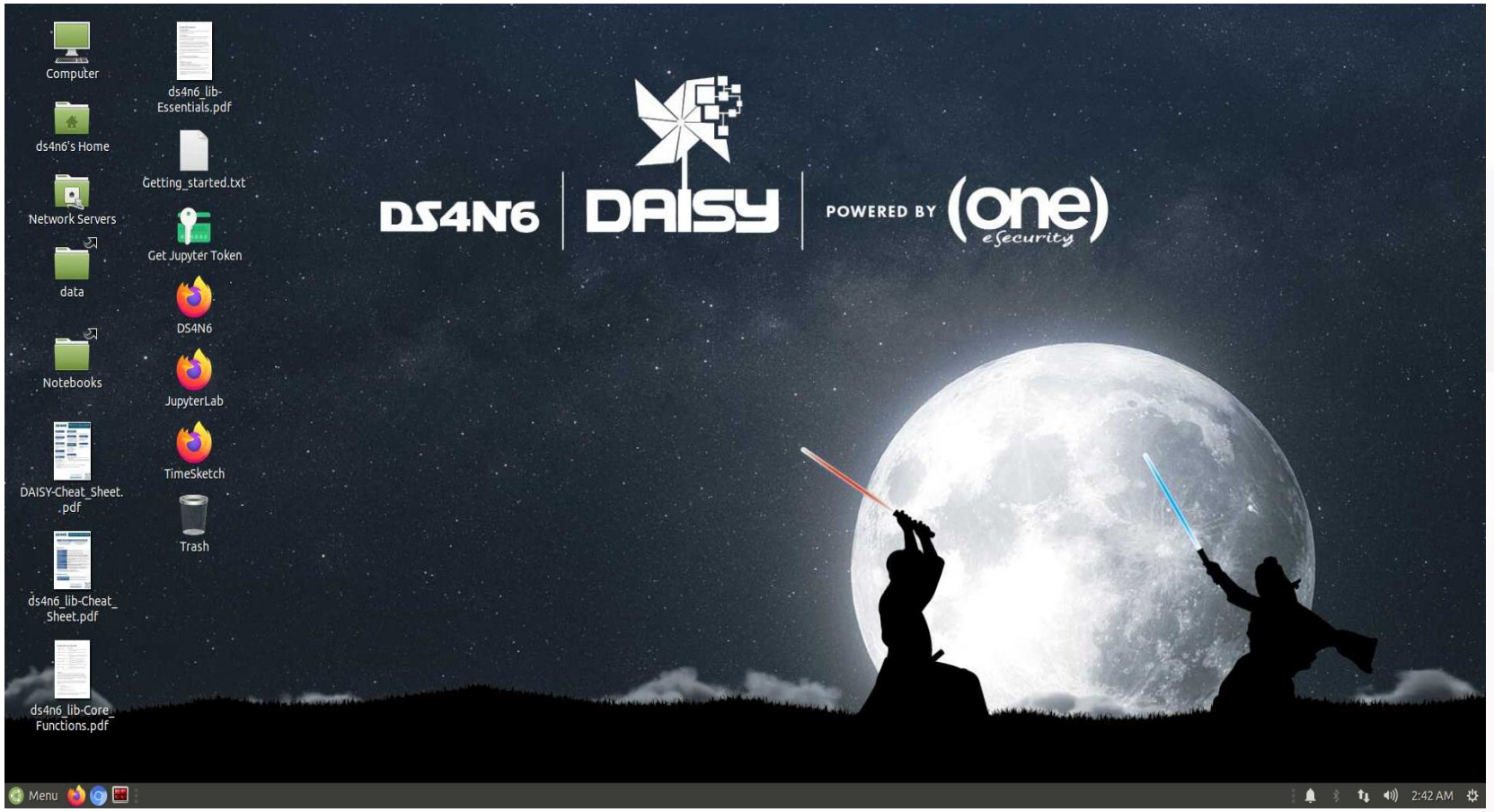
For more information, visit https://www.ds4n6.io/tools/chrysalis.html

DAISY

DFIR

Data	D	
	A	Artificial
	I	Intelligence
Science	S	
	Y	

www.ds4n6.io/daisy



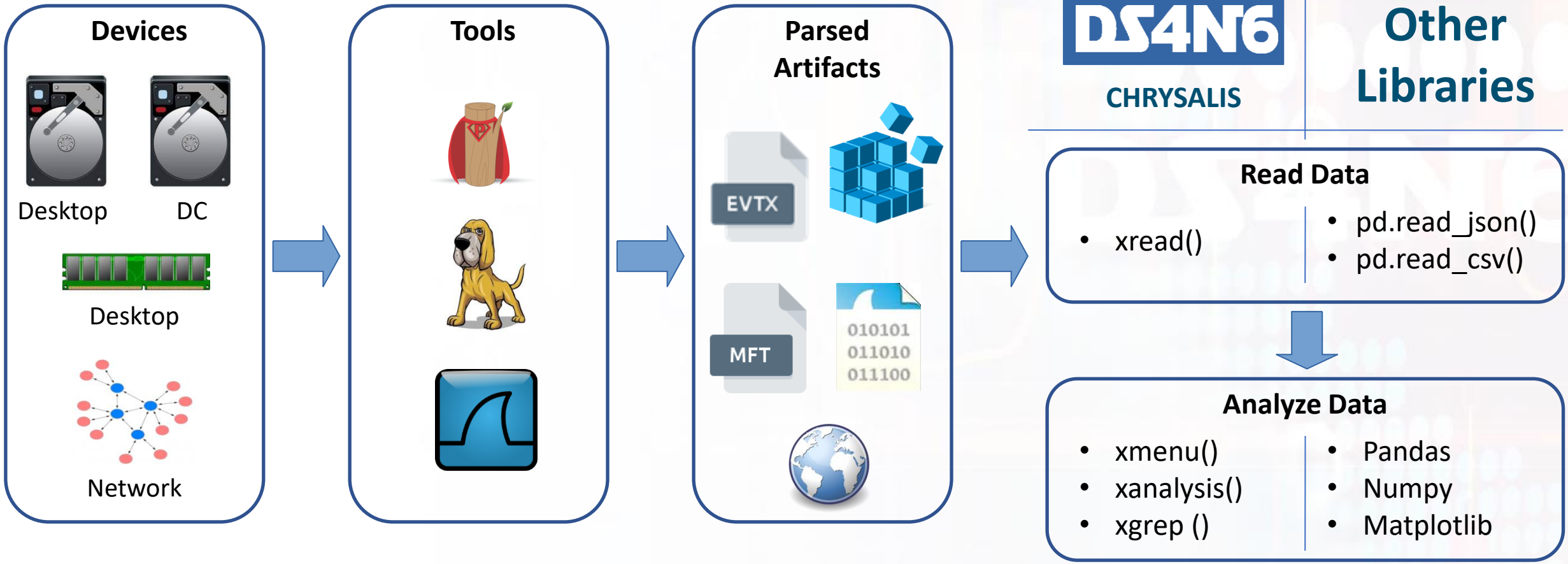
The Case: The Stolen Szechuan Sauce

Read the whole case here:
https://dfirmadness.com/the-stolen-szechuan-sauce/

- The FBI contacts us: our secret Szechuan Sauce recipe is on the Dark Web
- The recipe was on the Domain Controller. There are more interesting files in the share
- We receive the Domain Controller disk, an affected desktop computer disk and memory and a pcap with the network traffic of the attack date
- The computers were in Colorado (UTC -6)

We need to find out WHO, WHEN and HOW did it!

The Case: The Stolen Szechuan Sauce Case



All the details about this Workshop:

ds4n6.io/odscwest21

HANDS ON

DS4N6

www ds4n6.io

@ds4n6_io

DS4N6

(one) eSecurity

www one-esecurity.com

One_eSecurity