*Title In Progress*

Brandon Julian  
University of Colorado, BoulderBoulder, Colorado  
brandon.julian@colorado.edu

*Abstract*—I understand this is not the point of an abstract, however I am going to preface this submission by saying I am making a slight pivot from my original project idea submission. I am still attempting to interpret/reverse-engineer Hybrid Analysis™ Falcon Sandbox© incident response threat assessments, but I am switching from executable malware files to malicious URL detection. Hybrid Analysis’s API also contains URL reports and I thought it would be fitting to instead attempt to apply myself there within the scope of my skills. (*Abstract*)

Keywords—Malicious URL detection, features, static-analysis, dynamic-analysis, (key words)

# Introduction

As the internet continues to grow, malicious intent also grows with it. The potential for users to be scammed or taken advantage of increases with the ever-evolving attack methods of the unscrupulous. Very Dramatic.

# Data Collection

## Hybrid Analysis

To quote their FAQ, “This webpage is a free malware analysis service for the community. Using this service you can submit files for in-depth static and dynamic analysis.”[[1]](#footnote-1) Which is true, malware files can be submitted to their site for analysis. However, Hybrid Analysis (*H.A.)* also produces a product they call Falcon Sandbox. Which is a much more verbose system that does most of the dynamic analysis behind the scenes. What is very cool about their website-product integration is when items are submitted for analysis by a user of their product Falcon Sandbox it by default generates a high-level report of the item in question. To include interesting features of the item, whether that be signs of attempted access escalation, known malicious artifacts are present, or process spawning. It also includes a threat score out of one-hundred and a final label in the list of malicious, suspicious, no specific threat, no verdict, or whitelisted.

## Hybrid Analysis API – Search Query

At the beginning of the project I spent most of my time attempting to get their API and associated python wrapper up and running on my machine. That ended up being a tumultuous process due to extremely poor documentation on H.A.’s side and lack of my own patience. With the help of Dr. David Eargle of the Leeds MS. BA. program, he pointed out that the API was simply processing a POST request. And therefore, could be used outside of their python wrapper and web API interface[[2]](#footnote-2). Using the *requests* package, we wrote a function to submit a post request. This is an upside even though I sunk an enormous amount of time into fiddling with their API because now using *requests* the process can be automated to return hundreds of searches. To be clear, the search query returns basic information of a file that was submitted to their platform for analysis in a nested JSON format. It includes some metadata, a verdict, threat score, and the most important piece which is a job-id. The job-id can then be used in their report download get request to download the full report of each file returned in the search. It is this full report that contains H.A’s features and analysis on the item. See screenshots in the appendix (section IV) for examples of the code and return.

## Hybrid Analysis API – Report Download

After the struggle that was implementing a scalable way to search their database, a scalable way to extract the features and therefore the reports needed to be made. Using the previous POST request format, I was able to create an extremely simple version for their report download. The only hang up I had was realizing that the report download was a GET request and not a POST. With that slight adjustment it worked as intended.

## Next Steps

Now with the initial framework developed I need to create a pipeline that (1) queries their database (2) saves the query results as there is necessary target information there (3) extracts job-id from the query (4) uses the job-id to get the report for the file (5) generate feature extraction code and decide on ways to reduce dimensionality as well as possibly implement my own lexical feature creation. I suspect (5) will soak up the next chunk of my time as I also need to draw connections between their web platform and the return from the report download as there are some slight nuances to how features are described (e.g “mutex” vs “mutant”).

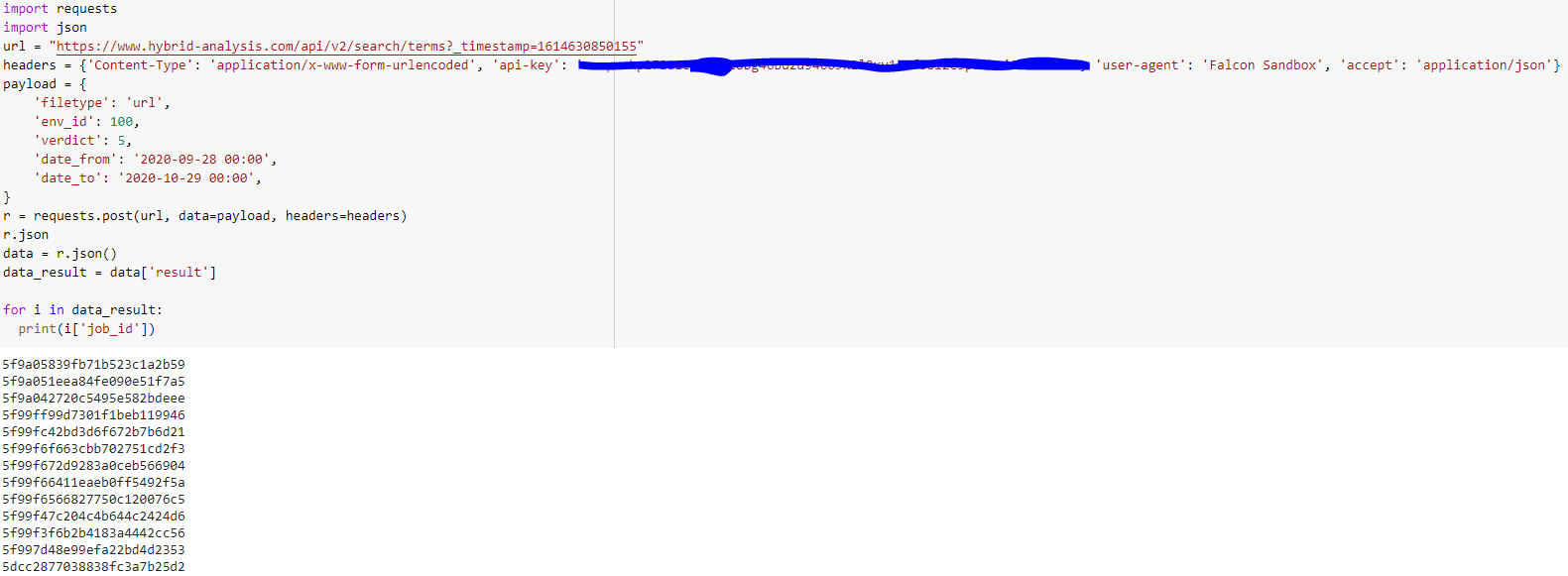
# Previous work, inspiration

## Previous Work

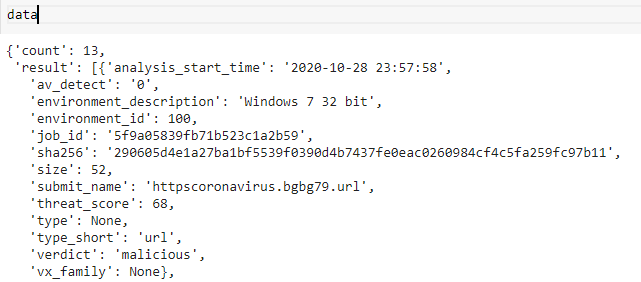
Since I recently decided on the project pivot, I did not collect 3 outside publications just yet. However, I did find this: Doyen Sahoo, Chenghao Liu, and Steven C.H. Hoi. 2019. Malicious URL Detection using Machine Learning: A Survey. 1, 1 (August 2019).[[3]](#footnote-3) That paper is an extremely readable and well documented approach to URL detection and will inspire the possible lexical feature creation I hope to implement in my final project. It also very well could influence my choices in feature extraction, but my hopes there lie in using all of the available information from H.A’s reports. What I find to be most beneficial of Sahoo et al. is the education on the domain of URL detection.

# Appendix

## Hybrid Analysis – Search Query



*Fig 1. POST Request, Job-id extraction*

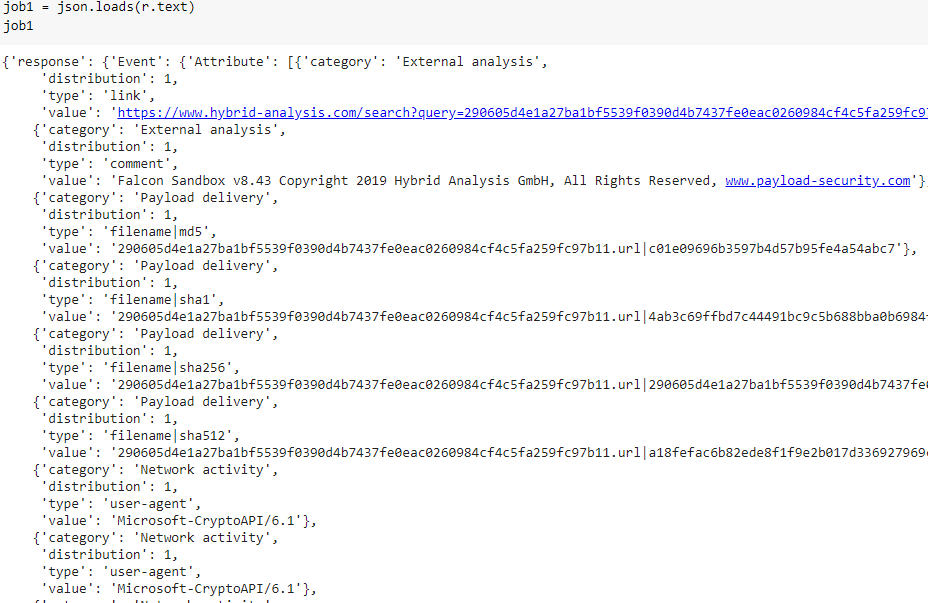
**

*Fig 2. POST Request JSON return*

## Hybrid Analysis – Report Download

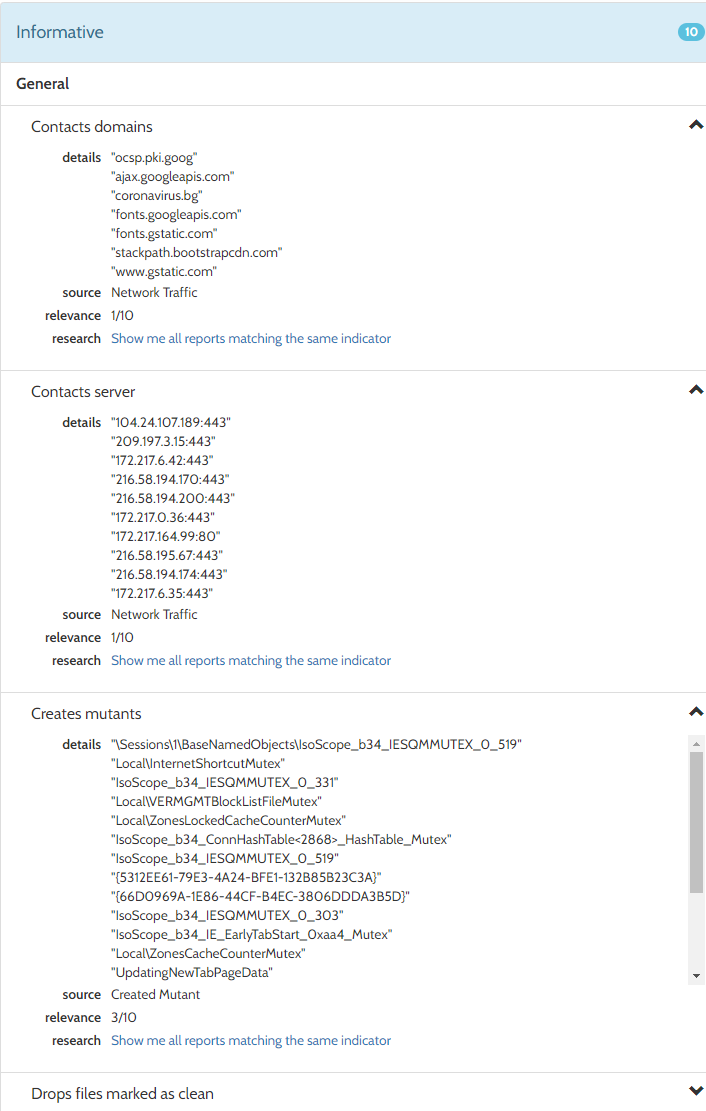


*Fig 3. Get Request*



*Fig 4. Get Request JSON return*

## Web examples, how the API results and graphical report relate



1. https://www.hybrid-analysis.com/faq [↑](#footnote-ref-1)
2. https://www.hybrid-analysis.com/docs/api/v2 [↑](#footnote-ref-2)
3. https://arxiv.org/pdf/1701.07179.pdf [↑](#footnote-ref-3)