**Data Exploration and Understanding**

This is the second in a series of posts about the new public Click Security GitHub project [Data Hacking](http://clicksecurity.github.io/data_hacking/). The project utilizes an open architecture based on Python and the most recent advances in data analysis, statistics, and machine learning. We investigate challenging security issues through a set of exercises that use open data sources and popular python modules such as Pandas, Scikit Learn, and stats models. All materials are presented within a set of iPython notebooks that are shared publicly.

**Exercise: Malware Domain List Data Exploration**

The [Data Hacking](http://clicksecurity.github.io/data_hacking/) GitHub project has several posted exercises, this exercise focuses specifically on data from the Malware Domain List website (<http://www.malwaredomainlist.com>.) We'd like to thank them for providing this resource and making their data available to the public.

**Resources**

* [MDL Exploration Notebook](http://nbviewer.ipython.org/github/ClickSecurity/data_hacking/blob/master/mdl_exploration/MDL_Data_Exploration.ipynb)
* [Data Hacking (MDL) GitHub](https://github.com/ClickSecurity/data_hacking/tree/master/mdl_exploration)
* [Malware Domain List Website](http://www.malwaredomainlist.com)

**Python Modules Used:**

* [iPython](http://ipython.org): Architecture for interactive computing and presentation
* [Pandas](http://pandas.pydata.org/): Python Data Analysis Library
* [Scikit Learn](http://scikit-learn.org/) Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
* [matplotlib](http://matplotlib.org): Python 2D plotting library
* [statsmodels](http://statsmodels.sourceforge.net): descriptive statistics, statistical tests, and plotting functions.
* [tldextract](https://pypi.python.org/pypi/tldextract): Accurately separate the TLD from the registered domain and subdomains of a URL.

**Data Source**

* Approximately 90k urls over a period of 4-5 years from the Malware Domain List website (full database available at: <http://www.malwaredomainlist.com/mdlcsv.php>)

When it comes to data analysis, data processing, cleaning and normalization is often 95% of the battle. Never underestimate this part of the process; if you're not careful about it you’ll be sorry later. Another good reason to spend a bit of time on understanding your data is that you may realize that the data isn't going to be useful for your task at hand and perhaps move to another data source.

The [MDL Exploration Notebook](http://nbviewer.ipython.org/github/ClickSecurity/data_hacking/blob/master/mdl_exploration/MDL_Data_Exploration.ipynb) contains all the code and details of the exercise but we'll summarize the work and approach here.

**Summary of Approach**

**Data Ingestion and Cleanup:**

The notebook demonstrates some of the challenges with character encodings, converting the date stamp into an *ISO8601:2004/RFC3339* compatible format, and discovering that one of the columns had nonsensical data. We place the data into a pandas dataframe which quickly allows us organize the data into a form fit for processing:

| **date** | **domain** | **ip** | **reverse** | **description** | **registrant** | **asn** |
| --- | --- | --- | --- | --- | --- | --- |
| 2013/10/30\_16:18 | hotgirlxchicvideos.net/xvidavi/ | 81.177.141.33 | srv110-h-st.jino.ru. | Leads to trojan | Dmitriy Petrov / admin@hotgirlxchicvideos.net | 8342 |
| 2013/10/30\_16:18 | trehomanyself.com/f/go.php?sid=2 | 81.177.141.193 | srv126-h-st.jino.ru. | Leads to trojan | Dmitriy Dmitriy / ivanloop13@trehomanyself.com | 8342 |
| 2013/10/30\_17:50 | www.pornerbros.com/281997/sexy-brunette-is-rea... | 66.152.87.239 | host-66-152-87-239.pornerbros.com. | Leads to exploit | Admin BFEF51CD2BAA41AC86F072F15A853822.PROTECT | 14720 |
| 2013/10/31\_18:23 | www.blueimagen.com/Attachment/Invoice-List2013... | 65.99.225.72 | server79.neubox.net. | Trojan.AdWind | Tools Ideas Enter (staff@toolsideascreativas.com) | 36024 |
| 2013/10/31\_18:23 | tvnotas.us/desktop/Snapshot2013-10-20.jar | 65.99.225.171 | server88.neubox.net. | Trojan.AdWind | Alberto Rodriguez Garcia | 36024 |

We used tldextract to pull the TLD from the full urls (see [tldextract](https://pypi.python.org/pypi/tldextract) for usage):

import tldextract

df['domain'] = ['.'.join(tldextract.extract(uri)[-2:]) for uri in df['domain'].values]

uol.com.br 458 woyo8g.com 310 geocities.com 263

y83h2.com 105 ipq.co 70 dnset.com 69

lookseekpages.com 55 1accesshost.com 52 ovfr6.com 41

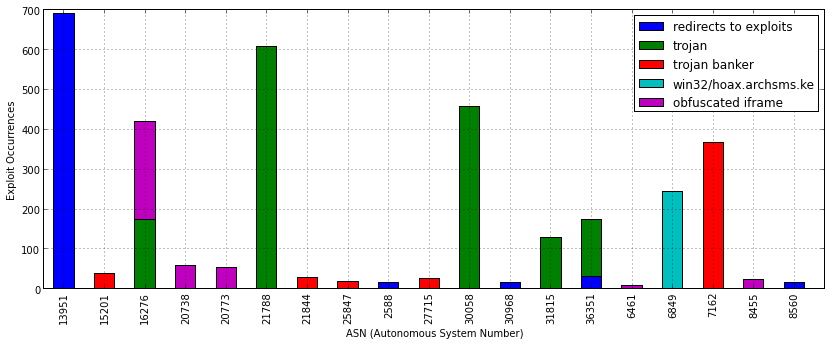
kogaryu.com 40 real-host.ru 37 ibnsites.com 36

zapto.org 36 hpg.com.br 35 thechocolateweb.ru 35

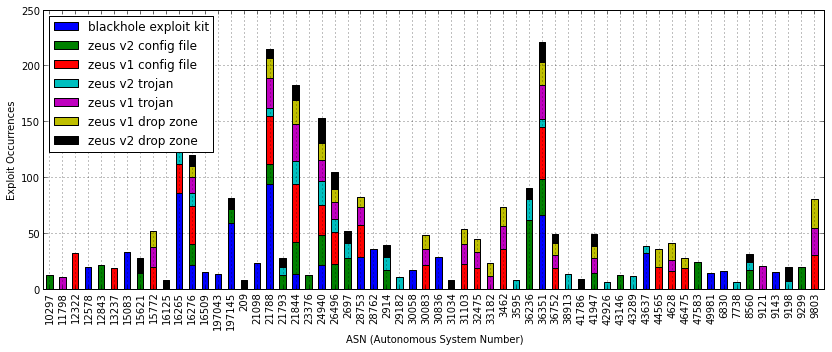
...

**Statistics and Plotting:**

Now that the data is cleaned up and processed we’ll take a look as some statistical independence tests and plot the results. G-test is for goodness of fit to a distribution and for independence in contingency tables. It's related to chi-squared, multinomial and Fisher's exact test please see [G Test: Wikipedia](http://en.wikipedia.org/wiki/G_test) for more information. The data hacking repository has a simple stats module we’re going to use to see how various exploits are related to ASNs (Autonomous System Number) By using the g-test functionality in our stats module and pulling the highest scoring domains we produced the plot below. We’re interested in seeing whether exploits are highly correlated to particular ASNs.

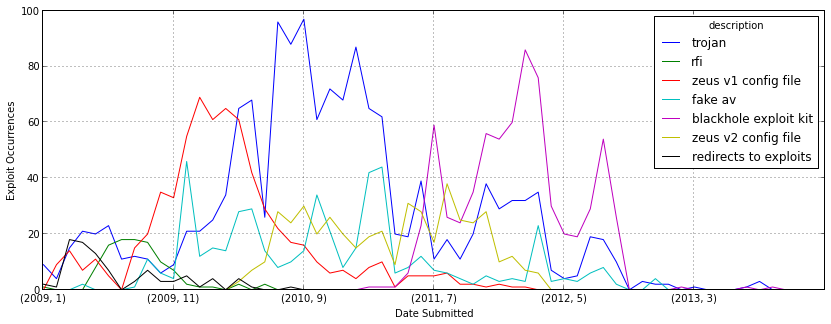


The association of a particular exploit to an ASN might warrant further investigation, the opposite may also be interesting, which exploits are wide spread and have no particular association with an ASN. The plot below shows the lowest scoring g-test results where exploits are basically independent of any particular association with an ASN.



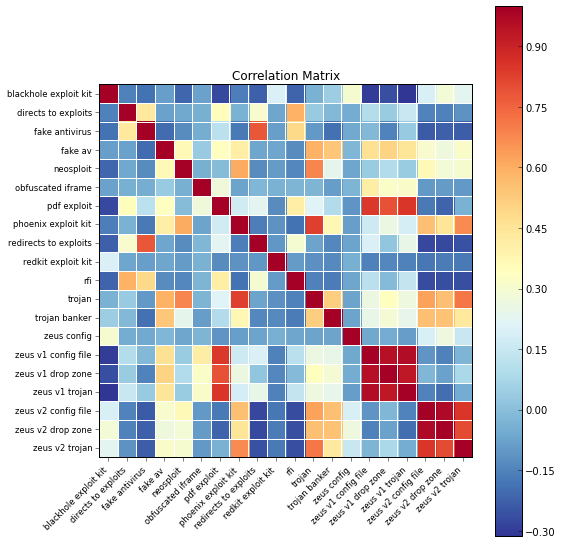
**Dataframes, Pivoting, and Time Series Analysis:**

The pandas python module has nice support for pivoting and time series analysis. The notebook breaks down all the details of taking the Malware Domain List data and pivoting it on volume of exploits over time (only a few lines of python). The plot below shows the volume of particular exploits impacting new domains. Tracking the ebb and flow of exploits over time might be useful depending on the type of analysis you're doing.



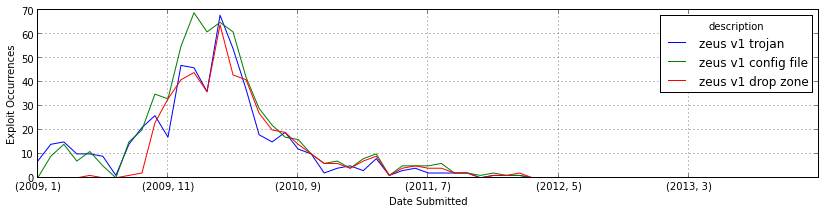
**Correlations of Volume Over Time:**

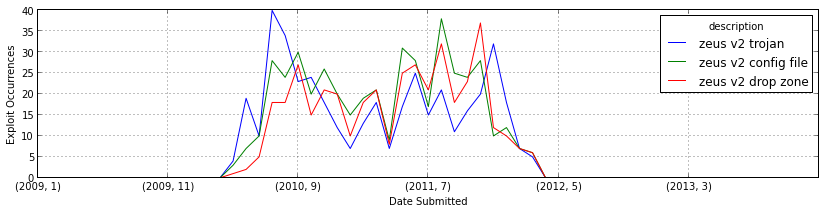
The Pandas module also has a slick correlation function that can be called on the dataframe directly. Pearson’s r is the default correlation function but other variants can be passed in as a parameter. For Pearson’s r a score of 1.0 means perfectly positively correlated, 0 means no correlation and -1 is perfect negative correlation (see[*Pearson's Correlation*](http://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient))*.* So we call corr() on the dataframe and then utilize a nice plot from the [statsmodels](http://statsmodels.sourceforge.net) library. In the plot below we expect the diagonal to have perfect correlation (1.0) but any cell with a high score off the diagonal means that the volume of different exploits are temporally correlated.

****

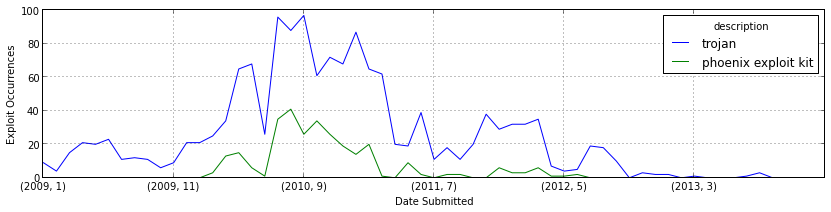
**Discussion of Correlation Matrix:**

The two sets of 3x3 red blocks on the lower right make intuitive sense, Zeus config file, drop zone and trojan show almost perfect volume over time correlation.

****

****

We also notice a correlation between 'trojan' and 'phoenix exploit kit' in the matrix plot above (although not as strong and the zeus correlations).

****

Looking above we see that the generic 'trojan' label and the fairly specific 'phoenix exploit kit' have a reasonable volume over time correlation of .834. So it certainly might be something to dive into depending on your particular interest, again the win here is that with a few lines of python code we can 'see' these kinds of relationships.

**Conclusions:**

So this exercise was an exploration of the dataset. At this point we have a good idea about what's in the dataset, what cleanup issues we might have and the overall quality of the dataset. We've run some simple correlative statistics and produced some nice plots. Most importantly we should have a good feel for whether this dataset is going to suite our needs for whatever use case we may have.

In the next exercise we're going look at some syslog data. We'll take it up a notch by computing similarities with 'Banded MinHash', running a hierarchical clustering algorithm and exercising some popular supervised machine learning functionality from Scikit Learn http://scikit-learn.org/.

Please visit the new Click Security [Data Hacking](http://clicksecurity.github.io/data_hacking/) GitHub site for additional exercises, code, and iPython notebooks.

-Cheers

Brian Wylie (Lifeform at Click Security)