Lab 2 - The Great Host/Lexical URL Reputation Bake-off ™

Summary

Modern URL classification systems can classify most URLs without ever needing URL content. This is important because it is far less expensive to classify a URL without having to crawl, download, store, and analyze content. Furthermore, it is often impossible to access content of a URL due to single-shot and auto-cloaking malicious websites.

Typically this classification is done using a combination rule-based and machine-learning techniques. Since we don't have time to delve into machine learning, let's build a rule-based micro-classifier which uses weighted scoring for different URL features. We will run the classification set through our newly built micro URL classifier and submit them to Piazza and see how we did against our peers. Prizes might be involved!

Instructions

1. Form teams of 3. Feel free to trade team members with other teams to assure your team has a good mix of skills. Try to make sure your team has at least one person who feels comfortable writing simple Perl or Python scripts. Perl or Python will be needed in order to read the provided JSON files.
2. Open up the "Lab 2" folder on your desktop. You already likely have doe this as you are now reading this PDF file.
3. Note that there are two files of URL records, in standard JSON format. One file is the training file, which includes a flag that notes if the URL is pre-known as malicious. The other file is the classification set, which does NOT have a malicious field, which you will be classifying using your micro URL classification system you will build.
4. Note that there is a utility provided for you, this is a template (Perl or Python - your choice) that you can use . It is a simple parser for the JSON files. You will be extending one of these scripts and turning them into your classification system.
5. Explore the training file and note the features listed for each URL. If you have questions about any of the features please feel free to ask and/or discuss.
6. Talk with your team members about what you think makes a URL more suspicious. Here are some hints to get you thinking:
   a. Young domains are likely **MORE malicious** than old domains
   b. Domains which don't return IP addresses could be fast-flux domains. These domains are likely to be **MORE malicious**. For example, how often does a DNS query for google.com fail?
   c. URLs which are listed in the Alexa top 1,000,000 are **LIKELY** to be **LESS malicious** than those that are not.
   d. URLs with a very low Alexa rank are likely to be **LESS malicious** that those with a high Alexa rank. This is known as "URL Prevalence"
   e. Another hint: 50% of the URLs in each file are malicious. Use this to help validate your results.
   f. What about file extension. How often do you *really* download raw .exe file directly from the web, instead of a software package.
   g. What about query string?
   h. How about the number of domain tokens? Path tokens?
   i. What port does the URL use? Do your favorite safe URLs usually use non-standard ports?
   j. What about odd combinations?? If a URL has a keyword in it such as 'paypal', but has a very young domain age and no Alexa rating, is it likely to be malicious? (Think phishing.)
   k. The more features you use, the more accurate your results will likely be.
7. Think up of additional features, extrapolated from the existing data set, which you can use.

8. Create a strategy for classifying the raw URL set. Consider using a point system where malicious features add to a score, and safe features detract from a that score. Figure out a threshold to use to decide if a URL has exceeded a score which makes it malicious. Try to have a basic strategy by the end of the lab.
9. Perform some validation on your strategy by aggregating features of the training set.
10. These are suggestions. Use your own ideas for classification. Perhaps some statistical analysis on the provided features?

Deliverable (The fun part) - Please provide by Monday:

1. Create a 1-page document which summarizes your strategy. What features are you using, what are their weights? How did you figure out a threshold?
2. Implement your strategy by extending the readcorpus script and turning it into your URL micro-classifier. Run it against the classification set (the one without malicious_url flag populated) and have it write a results file of in the format : `<url string>, <malicious bit>` (where 1 =malicious, and 0=safe)
3. Zip up your strategy document and results text file and upload to Piazza. When all files are received we will have a leader board with everyone's results. Your results file should have about 2000 URLs in it, as that is the size of the classification set.
4. The files will be scored for accuracy, and teams will receive prizes for 1st, 2nd, and 3rd place.
5. The corpus files, script templates, and this document will be posted on Piazza so you can access them after the lab, if needed.

A final word

The point of this lab is to introduce a thought process, not just create an accurate classification run of the classification corpus. Have fun and don't worry about being 100% correct, as using weighted scoring is a basic technique. Given more time, you would employ other techniques such as machine learning. (which some of you may already have experience with) Feel free to be aggressive if it fits your strategy. Remember, you should expect about 50% of the URLs to be malicious.