Classifying computer processes in the DARPA OpTC dataset

Experimental Protocol Andrew Veal

Hypotheses (2 marks)

The detection of malware and malicious activity in enterprise networks is an ongoing challenge in cybersecurity **[12]**. Our objective is to build a system that will classify activity associated with a computer process as benign or malicious, using host-based logging data from the computer. We will use supervised learning and 'ground truth' labelled data to build a classification model for the DARPA Operationally Transparent Cyber (OpTC) dataset **[1, 3, 4, 5].**

This is challenging for a number of reasons:

- ***Scale*** – the DARPA OpTC dataset has over 17 billion events: significant data engineering is required to create the feature vectors for the Machine Learning (ML) experiments.
- ***Extreme class imbalance*** – Only 0.0016% of the events in the complete dataset **[1]** are malicious, so it is marginal whether there are sufficient examples for supervised learning.
- ***Limited variability of attacks*** –the attacks used related modus operandi **[5]**; therefore, classifiers trained on this dataset may not generalize well to other attack scenarios **[10]**.

However, the DARPA OpTC dataset is important. As noted in **[12]**, "*The lack of diverse and useful data sets for cyber security research continues to play a profound and limiting role within the relevant research communities and their resulting published research*". For example:

- The utility of many open-source cyber security data sets is compromised by anonymization and a lack of labelled malicious activity – for example, the LANL 2017 data release **[8, 12]**.
- Many researchers use sandbox environments to measure machine activity **[11]** and software system calls **[13]**. This is not representative of large-scale enterprise networks.

Many papers in the literature do not adequately describe the algorithms used, show a naïve application of Machine Learning (ML) or have poor experimental methodologies. For example,

- **[13]** indiscriminately applies 8 different ML algorithms without saying why this is necessary or giving details and they use accuracy as their success metric in settings with class imbalance.
- **[14]** simply apply 10-fold cross validation (stratified or not?) and use Area Under the ROC (AUROC) as the success metric in a paper titled "*Tackling Class Imbalance in Cyber Security Datasets*" – this (as should be well known – see **[6, 7]**) is completely inappropriate.
- **[2]** develop a new classification algorithm 'SK-Tree' for detecting malware in computer process event logs. It is applied to data from one computer in the DARPA OpTC dataset (the host computer with the highest number of malicious events), but it is hard to interpret the results as they simply apply 5-fold cross validation and report the AUROC (again, see **[6, 7]**).

We will establish a simple and interpretable baseline for the classification of computer process activity in the DARPA OpTC dataset. For each process, we will aggregate counts for the observed set of `(object, action)` events. Feature vectors for Machine Learning (ML) will be created by joining the frequency counts with 'ground truth' labels. We will apply a small number of Machine Learning (ML) algorithms, taking care to partition the data into training, validation and test sets using stratified sampling and k-fold cross validation. We hope to investigate simple techniques to mitigate class imbalance **[9]** and to pay special attention to choosing training, validation and test sets from distributions that reflect the data we expect to get in the future **[10]**.

Data (3 marks)

The DARPA OpTC dataset contains events from an isolated enterprise network of 1000 host computers. Sensors logged 'action' events associated with 11 'object' types over 14 days. On 3 consecutive days, a 'red team' introduced malware and malicious activity on 29 computers **[1]**.

The dataset is hosted on a Google drive **[4]** and documentation is available in a Github repository **[3].** A ground truth document **[5]** provides details of the malicious *events*. Note that we define a *malicious process* to be a process which is linked to a malicious *event* detailed in the ground truth **[5].** The starting point for the project was the raw json data **[4]**, ground truth table **[5]** and a table of ‘process create’ event data (extracted from properties in the raw json `PROCESS_CREATE_` events).

Extract Transform Load (ETL) data engineering pipeline to create Machine Learning (ML) data

The raw json data was downloaded to AWS S3 and partitioned into 4554 part files, each o(500MB).

In the first ETL stage, we selected the data for the 29 computers attacked on the 3 ‘red team’ days, dropping detailed meta-data contained in ‘properties’ tables **[1, 3, 5]**. The resulting 228 ‘summary’ files, each o(400MB), contained the following fields:

- `object` – one of {`FILE, FLOW, HOST, MODULE, PROCESS, REGISTRY, SERVICE, SHELL, TASK, THREAD, USER_SESSION`}
- `action` – one of a limited set associated with a given object: see **Figure 1** below for types
- `actorID` – the unique identifier (UID) for the parent process
- `objectID` – the unique identifier (UID) for the object (child process if object = `PROCESS`)
- `hostname` – the name of the computer (eg. `SysClient0501.systemia.com`)

In the next stage, we created aggregate counts for every `object.action` pair observed for each `actorID`. The resulting 212 ‘aggregate’ files, each o(400KB), contained following additional field:

- `object.action` counts – dictionary of counts for the observed `object.action` pairs

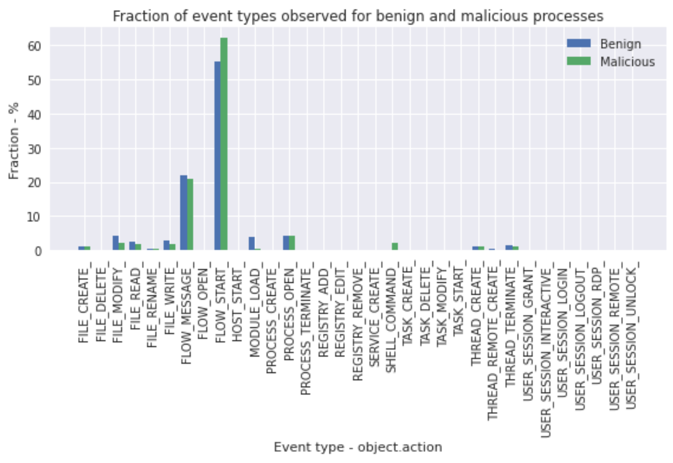
In the final ETL stage, we join reference data from the ground truth table **[5]** and the ‘process create’ table to the ‘aggregate’ data to create a dataframe of feature vectors for Machine Learning (ML). The resulting ML dataframe has shape `(349841, 37)`. The first 4 columns in the ML dataframe are

`[‘actorID’, ‘hostname’, ‘process_name’, ‘label’]`

where `process_name` is the executable name, eg. `Powershell.exe`, and `label` is `0` for benign and `1` for malicious. The remaining 33 columns contain the `object.action` counts for `actorID`.

Class	Number of Processes	Fraction of Processes
Benign	348868	0.997219
Malicious	973	0.002781

Table 1 – Number of benign and malicious processes in the Machine Learning (ML) dataframe



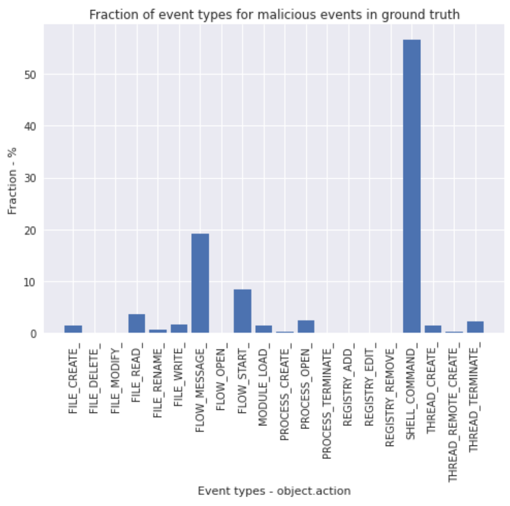


Figure 1 – The fraction of event types for malicious *events* in the ground truth table and the fraction of event types for benign and malicious *processes* in the Machine Learning (ML) dataframe.

Metrics (5 marks)

We shall use *precision*, *recall* and the *F1 score* as success metrics for the classification of malicious examples. In a setting where malicious examples are rare, it is important to account for the impact of false positives: the *precision* of the classifier is the fraction of processes in the validation (or test) set it labelled as malicious that really are malicious; *recall* is the fraction of all malicious processes in the validation (or test) set it correctly labelled. There is a trade-off between having a high *precision* and high *recall*. As we really care about both *precision* and *recall* we shall compute the *F1 score* and use it as our evaluation metric of success for optimizing thresholds **[10]**.

With extreme class imbalance it is important that we analyse bias-variance diagnostics at each stage of model selection and optimization. Following **[10]**, we can interpret the bias as the error rate of the algorithm on the training set. In the model selection experiments we report in **Table 2** we can see that simple linear classifiers (Logistic Regression (LR) and linear Support Vector Machine (SVM)) underfit the data and have a high bias. Again, following **[10]**, we can interpret the variance as how much worse you do on the test set compared to the training set. In fact, the results for the Random Forest are not bad:

- Training set accuracy for Random Forest was 0.9917 – see Table 2 for rest of metrics
- Out of bag estimate (validation set) accuracy for Random Forest was 0.9908
- Test set accuracy for Random Forest was 0.9903

The Random Forest already looks (in context) quite a good classifier: train, validation and test accuracy are all good and the out of bag estimate is close to test set accuracy. However, we will do some further tuning of (regularization) hyper-parameters (eg. **max_depth** and **n_estimators**) and plot validation curves to optimize the Random Forest classifier's hyper-parameters. Another technique to reduce variance of the Random Forest is to do feature selection to decrease the number of input features **[10]**: looking at **Figure 2**, where we rank feature importance, it appears that the top 20 features will capture the key discriminating factors (and the ground truth only used 20 features).

As noted already, there is a trade-off between *precision* and *recall* – in **Figure 2** we see the *Precision-Recall (PR) curve* as we vary the Random Forest probability threshold for classifying processes as malicious (calculated using 5-fold stratified cross-validation). The *Area Under Curve (AUC)* for the *Precision-Recall (PR) curve* is a useful measure of classifier performance when dealing with imbalanced classes (see **Table 2**). In **Figure 2** we show the point on the *PR curve* where the *F1 score* is a maximum and we report the values of the *F1 score*, *precision*, *recall* and *threshold* at that point. Using the *threshold* value for the classifier, we can rerun our test set experiments to get the results at the bottom of **Table 2**.

As noted previously, AUROC should never be used to measure performance **[6, 7]** but as it is prevalent in the literature **[2, 14]** we have given AUROC values in **Table 2** for information only.

Varying the probability thresholds has driven down the false positive errors but we will need to examine misclassification errors in detail. By looking at off-diagonal error cases in the confusion matrix, we will investigate whether hard to classify cases are associated with low event counts (spoiler alert – they are) or specific executables (is `PING.EXE` predominantly benign or malicious?).

Coming up with an unbiased and effective data selection and partition strategy for train/validation/test sets is important. We will look at a few different strategies:

- *simple stratified train/test split and stratified k-fold cross validation* – naïve method (we do not take account of *hostname* or *process_name* or *date* when we split) that we have applied to start with – good enough to do the basic model selection steps reported in this paper.
- *stratified train/validation/test split on days* – it turns out that the set of computers attacked on day 1 is different from the set of computers attacked on days 2 and 3: we will be able to partition the train/validation/test sets so that we don't mix day 1 with days 2 and 3.

- *stratified train/validation/test split on process names* – there is an argument that we should not include processes with the same process name in both the train and test sets: we should be learning behaviour associated with malicious activity independent of the process name.

Finally, we hope to look at methods tailored to imbalanced classes **[9]**, including: under-sampling the majority benign class, using balanced class weights, bagging and boosting.

Models (3 marks)

Our baseline model is the Random Forest classifier. **Table 2** shows results for three models and it clearly outperforms linear models. It offers bagging and balanced class weights to mitigate the class imbalance and gives an out-of-bag estimate for accuracy. It has hyper-parameters we can tune and feature importance metrics to aid feature selection. Exploratory Data Analysis (EDA) of the data suggests that decision trees could do well and they offer the additional benefit of explanatory results – for example, the following 'rule' detects only malicious processes in the ML `dataframe df`:

```
df[df['SHELL_COMMAND_'] > 2800][['label']].apply(lambda x: Counter(x))
{1: 12}
```

We plan to look at Gradient Boosted Trees – which might be expected to do better **[11]** – and also K Nearest Neighbour (KNN), if we can successfully under-sample the majority class (it doesn't scale).

With more time, we would investigate Deep Learning, but this is out of scope: we don't have time.

General Reasoning (3 marks)

Our core hypothesis is that we can distinguish between malicious and benign processes using the frequency count of the `(object, action)` events associated with each process as a feature vector. If we examine **Figure 2** and **Table 2** we achieve 88% precision with a recall of 51% on the test set with a Random Forest without any tuning of hyperparameters – that suggests our hypothesis is reasonable.

In addition, **Figure 2** shows we can identify the event types which are more important for the classification of malicious processes.

We should offer a few words of caution – our simplified train/validation/test strategy may be biased: we may have data leakage. We plan to investigate more principled train/validation/test splits.

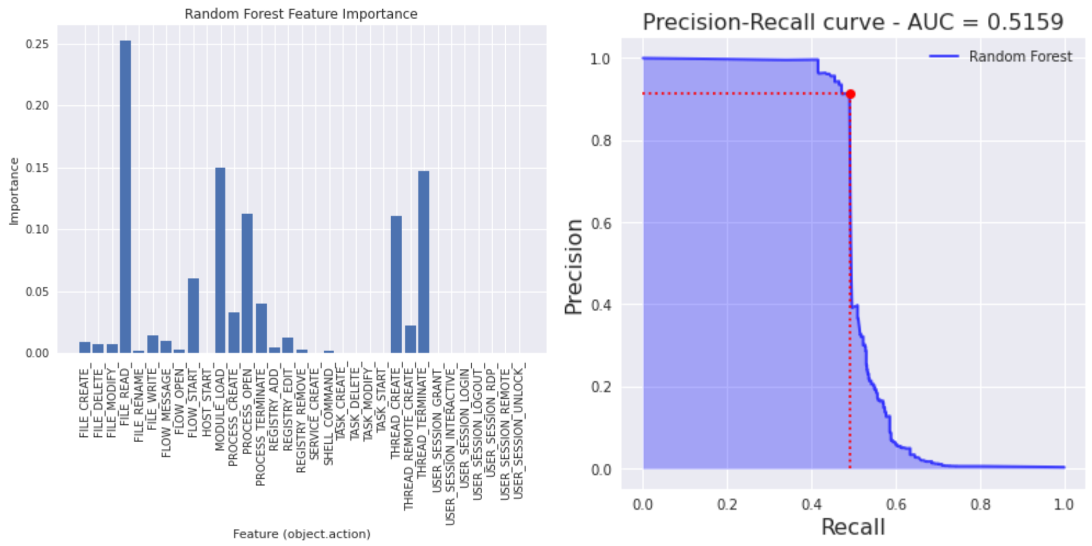


Figure 2 – Preliminary results for our baseline Random Forest classifier

Summary of progress so far (3 marks)

So far, we have:

- created the ML dataset, done basic EDA and partially completed the model selection.
- implemented a simple train/validation/test split strategy using stratified sampling and stratified 5-fold cross-validation and used a hold-out test set to evaluate the performance of models.
- finished model selection results for linear classifiers (LR, LinearSVM) and Random Forest.

We plan to:

- evaluate Gradient Boosted Tree (GBT) classifiers and do hyperparameter optimization.
- examine misclassification errors to understand hard to classify cases.
- try basic under-sampling methods using the imbalanced learn package **[9].**
- split the data on hostname to partition 'red team' day 1 from days 2 and 3 **[5].**
- split the data on `process_name` to guard against 'data leakage' from train to test sets.

Issues with our experimental protocol are:

- We are taking a summary view of process activity, aggregating frequency counts of high level (`object, action`) events only. We are not using the full richness of the dataset: each `(object, action)` event has detailed properties. For example, `PROCESS.CREATE` event table contains meta-data for `timestamp`, `user`, `image path` and `command line`.
- We need to pay special attention to the choice of training, validation and test sets, so that the distributions (as far as possible) reflect the data we expect to get in the future **[10]**.

Metric	Logistic Regression	Linear SVM	Random Forest
Training set results			
Precision	0.0074	0.0072	0.2432
Recall	0.4082	0.4332	0.9413
F1 score	0.0145	0.0142	0.3865
Test set results			
Precision	0.0081	0.0080	0.1551
Recall	0.4452	0.4760	0.5582
F1 score	0.0159	0.0157	0.2427
Cross-Validation results			
Area Under Precision-Recall	0.0238	0.0163	0.5159
F1 score maximum	0.0818	0.0653	0.6386
Precision at F1 maximum	0.0672	0.0735	0.9151
Recall at F1 maximum	0.1043	0.0587	0.4905
AUROC (for info only)	0.6834	0.6327	0.8106
Test set using threshold			
Threshold value	0.8780	142.16	0.8000
Precision	0.0740	0.0747	0.8810
Recall	0.1199	0.0616	0.5068
F1 score	0.0915	0.0675	0.6435

Table 2 – Results for three finished models: Random Forest outperforms linear models.

References (1 mark)

[1] Md. Monowar Anjum, Shahrear Iqbal and Benoit Hamelin. 2021. Analysing the Usefulness of the DARPA OpTC Dataset in Cyber Threat Detection Research. arXiv:2103.03080v2. Retrieved from https://arxiv.org/abs/2103.03080

Accepted for *ACM Symposium on Access Control Models and Technologies (SACMAT)*, 16-18 June, 2021, Barcelona, Spain [virtual event]. ACM Inc., New York, NY. DOI: https://doi.org/10.1145/3450569.3463573

[2] Thomas Cochrane, Peter Foster, Varun Chhabra, Maud Lemercier, Terry Lyons and Cristopher Salvi. 2021. SK-Tree: a systematic malware detection algorithm on streaming trees via the signature kernel**.** arXiv:2102.07904v3. Retrieved from https://arxiv.org/abs/2102.07904

[3] DARPA. 2020. Operationally Transparent Cyber (OpTC) Data Release. README. Retrieved from http://github.com/FiveDirections/OpTC-data

[4] DARPA. 2020. Operationally Transparent Cyber (OpTC) Data Release. Retrieved from https://drive.google.com/drive/u/0/folders/1n3kkS3KR31KUegn42yk3-e6JkZvf0Caa

[5] DARPA. 2020. OpTC Red Team Ground Truth. Retrieved April 7, 2021 from https://github.com/FiveDirections/OpTC-data/blob/master/OpTCRedTeamGroundTruth.pdf

[6] David J. Hand. 2009. Mismatched Models, Wrong Results, and Dreadful Decisions. Keynote at *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, June, 2009 Paris, France recorded June 2009, published September 14, 2009
http://videolectures.net/kdd09_hand_mmwrdd/ (video)
http://videolectures.net/site/normal_dl/tag=45840/kdd09_hand_mmwrdd_01.pdf (slides)

[7] David J. Hand. 2009. Measuring classifier performance: a coherent alternative to the area under the ROC curve. *Mach Learn*, 77 (2009)**,** 103–123. DOI: https://doi.org/10.1007/s10994-009-5119-5

[8] LANL. 2017. Unified Host and Network Data Set. Retrieved from https://csr.lanl.gov/data/2017/

[9] Guillaume Lemaitre, Fernando Nogueira and Christos K. Aridas. 2016. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research,* 7 (2016) 1-5 DOI: https://dl.acm.org/doi/pdf/10.5555/3122009.3122026

[10] Andrew Ng. 2018. Machine Learning Yearning: Technical Strategy for AI Engineers, In the Era of Deep Learning. Draft Version. Retrieved May 6, 2021 from https://www.deeplearning.ai/programs/

[11] Matilda Rhode, Pete Burnap and Kevin Jones. 2018. Early-stage malware prediction using recurrent neural networks. *Computers & Security,* 77 (August 2018), 578-594. DOI: https://doi.org/10.1016/j.cose.2018.05.010

[12] Melissa J. M. Turcotte, Alexander D. Kent and Curtis Hash. 2018. Unified Host and Network Data Set. In *Data Science for Cyber-Security*, Chapter 1 (November 2018), 1-22. World Scientific DOI: https://doi.org/10.1142/9781786345646_001

[13] Aaron Walker and Shamik Sengupta. 2019. Insights into Malware Detection via Behavioral Frequency Analysis using Machine Learning. In *Proceedings of the 2019 IEEE Military Communications Conference (MILCOM)*, 12-14 November, 2019, Norfolk, VA, USA. IEEE *Explore,* 1-6. https://doi.org/10.1109/MILCOM47813.2019.9021034.

[14] Charles Wheelus, Elias Bou-Harb and Xingquan Zhu. 2018. Tackling Class Imbalance in Cyber Security Datasets. In *Proceedings of the 2018 IEEE International Conference on Information Reuse and Integration (IRI)*, 6-9 July, 2018, Salt Lake City, UT, USA. IEEE *Xplore*, 229-232. https://doi.org/10.1109/IRI.2018.00041