Towards Explicable and Adaptive DDoS Traffic Classification

Yebo Feng, Jun Li
University of Oregon
{yebof, lijun}@cs.uoregon.edu
Introduction & Background

• Decades of research and industry efforts have led to a myriad of DDoS detection and classification approaches. Nowadays, many researchers begin to harness machine learning in classifying DDoS attacks. However, such methods have two negative aspects:

1. The prediction results are inexplicable. An unexplainable result may lead to unexpected collateral damage when conducting access control.

2. Learning-based methods are not adaptive. A model trained in one environment cannot easily apply to another environment.
Introduction & Background

• In this poster, we propose a learning-based DDoS traffic detection and classification method.
  1. It utilizes a modified k-nearest neighbors algorithm to detect DDoS threats.
  2. It then conducts fine-grained traffic classification using risk degree sorting with grids.
  3. To improve efficiency, we use a k-dimensional tree to partition the searching space, shortening the time for queries significantly.

• Compared with the previous learning-based approaches, this method is highly explicable and adaptive.
Methodology

• Our approach has two phases, which are DDoS detection and classification.
• It monitors the traffic in batches. Each batch $t$ is a uniform time bin, which is also the most basic detection unit.
• During each batch $t$, our approach will extract features to form a traffic profile $S (S = \{ feature\ 1\ ,\ feature\ 2\ ,\ ...,\ feature\ n \} )$ and input it into the detection module.
• In the classification phase, our approach will generate traffic profile $D (D = \{ feature\ 1\ ,\ feature\ 2\ ,\ ...,\ feature\ n \} )$ for each source IP and determine whether it is malicious according to the features.
Phase one: detection of DDoS traffic

• We use k-nearest neighbors (KNN) algorithm in the detection of DDoS traffic.

• Although it takes no time to train the model, the prediction requires a time complexity of $O(n\log n)$ to complete. Hence, we leverage the KD tree to partition the searching space, reducing the number of data points to enumerate.

• Furthermore, our approach generates a decision tree out of the KD tree, reducing the time complexity for traffic monitoring to nearly $O(1)$. 
Towards Explicable and Adaptive DDoS Traffic Classification

Yebo Feng, Jun Li

Diagram:
- Feature 1 vs Feature 2
- KD tree partition
- Area scan
- Tree generation and pruning

Legend:
- Green: Benign profile
- Red: Malicious profile
- Blue: Unconfirmed area
- Pink: Confirmed malicious area
- Green: Confirmed benign area
Phase two: DDoS classification

• Design philosophy: the traffic profile is currently in a malicious position, and we need to conduct access control on some of the sources, so that the traffic profile can return to a benign area.
  • First step: calculate the shortest path \( p \) of the current position to a benign area.
• We conduct the classification for malicious sources by building traffic profiles for each IP address.

• Then, mark IP as malicious in a particular order (according to $p$) until the overall traffic profile returns to a benign area.

```python
malicious_IP = set()
eliminated_val = 0
D = sort(IP)  # Grid cutting
for i in D:
    Malicious_IP.add(i)
    eliminated_val += i.feature2
    if eliminated_val >= p:
        return malicious_IP
```
Adaptability

• Users do not need to retrain the proposed model to fit it into a different network environment. They can easily use some prior knowledge to refit the model:
  • If we have the traffic measurement information about the new environment, we can normalize the KNN searching space from the trained environment to the new environment according to the two networks’ traffic distributions.
  • If the traffic monitoring system can obtain labeled traffic with the system running, we can efficiently conduct online learning on the proposed model.
  • In some circumstances, the user of this method may know some incomplete threshold values or rules in the new network environment. They can then build a decision tree based on the preliminary knowledge and merge it with the trained classifier, a tree-like data structure.
Thanks!