Unsupervised Approach for Detecting Low-Rate Attacks on Network Traffic with Autoencoder

Baskoro Adi Pratomo
Pete Burnap
George Theodorakopoulos
What network traffic looks like

What you see on your monitor

What your computer actually sends and receives
Low-Rate Attacks

- **Exploit**
  - Piece of code that takes advantage of bugs/vulnerabilities in order to cause unintended behaviour
  - i.e. Heartbleed, SQL Injection, etc

- **Backdoor**
  - Software which is planted by attacker to have another entrance to the system
Why doing all of these?

- Statistical properties of traffic obtained from packet header may not be enough
- Classification is most likely to perform badly on imbalanced dataset
- Difficulty of having a labelled dataset for training purposes in real world
Previous works

- PAYL
- McPAD
- OCPAD
- Anagram
- Poseidon
- HMM PAYL
- RePIDS
Methodology

The complicated part
TCP Stream Reassembly

1. TCP breaks up data into datagrams

2. Datagrams travel the Internet possibly over different routes according to IP

3. TCP reassembles datagrams

TCP requests resend of garbled datagrams
What is an Autoencoder?

- Autoencoder neural network
  - An unsupervised learning algorithm that applies backpropagation, setting the target values to be equal to the inputs
- May be used as outlier detection
  - By calculating the total of reconstruction errors
How it learns

Network Traffic

1

Byte Frequencies

2

Training

Model

3a

Average Error as Threshold
Skewed Median
Z-Score

3b
How it detects malicious traffic

1. Extract features from legitimate traffic and compare it with malicious traffic.
2. Calculate reconstruction error.
3. Compare the error with a threshold to determine if traffic is malicious.

Diagram:
- Legitimate Traffic
- Malicious Traffic
- Input
- Features
- Output
- Compare
- Calculate Reconstruction Error
Defining Threshold

- Usual approach:
  - Mean + (2 or 3) * Standard Deviation
- The distribution of Reconstruction Errors of training set is skewed
Defining Threshold (2)

- Skewed Median
  - MedCouple (MC):
    \[ MC(F) = \text{median}_{x_i < m < x_i} h(x_i, x_j) \]
    \[ h(x_i, x_j) = \frac{(x_j - m) - (m - x_i)}{x_j - x_i} \]
  - Threshold:
    \[ Q_3 + e^{3MC} \times 1.5 \times IQR, \text{ if } MC \geq 0 \]
    \[ Q_3 + e^{A_{MC}} \times 1.5 \times IQR, \text{ if } MC < 0 \]

- Median Absolute Deviation (Z-Score method)
  \[ MAD = \text{median}(\{|e_i - \text{median}(E)|\}, e_i \in E, 0 < i < n) \]
  \[ z = \frac{0.6745 \times (|e - \text{median}(R)|)}{MAD} \]
Experimental Results (Dataset)

- UNSW NB15 Dataset
  - Training set:
    - 93406 HTTP connections
    - 41395 SMTP connections
  - Testing set:
    - 131842 legitimate traffic (HTTP & SMTP)
    - 17041 malicious HTTP connections
    - 4631 malicious SMTP connections
Experimental Results (Comparison)

<table>
<thead>
<tr>
<th>Method</th>
<th>DR (%)</th>
<th>FPR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoencoder ($T_{IQR}$ Threshold)</td>
<td>68.91</td>
<td>0.99</td>
</tr>
<tr>
<td>Autoencoder (Z-Score Threshold)</td>
<td>100</td>
<td>8.01</td>
</tr>
<tr>
<td>OCPAD (1-gram)</td>
<td>19.88</td>
<td>0.00</td>
</tr>
<tr>
<td>OCPAD (3-gram) (HTTP only)</td>
<td>29.08</td>
<td>8.85</td>
</tr>
<tr>
<td>PAYL</td>
<td>36.1</td>
<td>0.05</td>
</tr>
</tbody>
</table>
## Experimental Results

(The Effect of Hidden Layers)

<table>
<thead>
<tr>
<th>Number of Neurons in Each Hidden Layer</th>
<th>Threshold Method</th>
<th>DR (%)</th>
<th>FPR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>$T_{IQR}$</td>
<td>68.91</td>
<td>0.99</td>
</tr>
<tr>
<td>200-100-200</td>
<td>$T_{IQR}$</td>
<td>68.21</td>
<td>0.63</td>
</tr>
<tr>
<td>200-100-50-100-200</td>
<td>$T_{IQR}$</td>
<td>69.21</td>
<td>3.16</td>
</tr>
<tr>
<td>200</td>
<td>Z-Score</td>
<td>100</td>
<td>8.11</td>
</tr>
<tr>
<td>200-100-200</td>
<td>Z-Score</td>
<td>100</td>
<td>8.01</td>
</tr>
<tr>
<td>200-100-50-100-200</td>
<td>Z-Score</td>
<td>100</td>
<td>8.1</td>
</tr>
</tbody>
</table>
Conclusion

- The proposed method can detect outliers (malicious traffic)
- Skewed median is more cautious, resulting in lower false positive rate
- Z-score method has got better detection rate with the cost of increasing false positive
Future Works

- Combining 1D Convolutional Network and Autoencoder to capture sequence attributes of network traffic
- Trying Recurrent Neural Network as an outlier detector
Thank you

Any question?