Advances in Cloud-Scale Machine Learning for Cyber-Defense

Mark Russinovich
CTO, Microsoft Azure
Microsoft Corporation
@markrussinovich
Intelligence in every software

- Cortana Intelligence Suite
- SQL Server + R
- Microsoft R Server
- Hadoop + R
- Spark + R
- Microsoft CNTK
- Azure Machine Learning
- R Tools/Python Tools for Visual Studio
- Azure Notebooks (JuPyTer)
- Cognitive Services
- Bot Framework
- Cortana
- Office 365
- HoloLens
- Bing
- Skype
- Xbox 360
- Dynamics 365
Microsoft’s daily cloud security scale

- 10s of PBs of logs
- 300+ million active Microsoft Account users
- 1.3+ billion Azure Active Directory logons
- Detected/ reflected attacks >10,000 location-detected attacks
- 1.5 million compromise attempts deflected
WHAT IS ATTACK DISRUPTION?
Red Team Kill Chain

- Recon
- Delivery
- Foothold
- Persist
- Move
- Elevate
- Exfiltrate
Blue Team

Kill Chain

for Attack Detection
Blue Team

Kill Chain

for Attack Disruption

Recon - Delivery - Foothold - Persist

Gather - Detect - Alert - Triage - Context - Plan - Execute

Move - Elevate - Exfiltrate

#RSAC
Challenges for Attack Disruption

False Positives

Manual Triage
False Positives

Lose ability to triage
False Positives **FACT**

You **cannot** salvage a false positive with just Visualization. You need better solutions.
False Positives
 Evolution of security detection techniques

**TRADITIONAL PROGRAMMING**
- Hand-crafted rules by security professionals
- Con: Rules are static, and don’t change with changes in environment => False Positives!

**MACHINE LEARNING**
- System adapts to changes in environment as new data is provided, and re-trained
- Our *supervised learning* approach enables detection *without generating many FPs*
Labels in Microsoft

For supervised learning, Azure gets labeled data through:

- Domain experts, customers who provide feedback from Alerts
- Labels from other product groups (including O365, Windows Seville)
- Surgical Red team exercises (OneHunt)
- Automated Attack bots
- Bug Bounty
- MSRC
False Positives  

Manual Triage
For Attack Disruption, we need to think beyond detection

Everything is manual, with little or no intelligence – We need to change this
Properties of a Successful Machine Learning Solution

- Adaptable
- Successful Detection
- Explainable
- Actionable
Adaptable in Cloud is Difficult

Why?

**Evolving Landscape**
- Frequent deployments
- New services coming online
- Usage spikes

**Evolving Attacks**
- Constantly changing environments leads to constantly changing attacks
  - New services
  - New features for existing services
Explainability

Why?

Surfacing a security event to an end-user can be useless if there is no explanation

Explainability of results should be considered at earliest possible stage of development

Results without explanation are hard to interpret

Best detection signal with no explanation might be dismissed/overlooked

<Example – How do you explain this to an analyst>

<table>
<thead>
<tr>
<th>UserId</th>
<th>Time</th>
<th>EventId</th>
<th>Feature1</th>
<th>Feature2</th>
<th>Feature3</th>
<th>Feature4</th>
<th>...</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a4b43</td>
<td>2016-09-01 02:01</td>
<td>4688</td>
<td>0.3</td>
<td>0.12</td>
<td>3.9</td>
<td>20</td>
<td>...</td>
<td>0.2</td>
</tr>
<tr>
<td>73d87a</td>
<td>2016-09-01 03:15</td>
<td>4985</td>
<td>0.4</td>
<td>0.8</td>
<td>0</td>
<td>11</td>
<td>...</td>
<td>0.09</td>
</tr>
<tr>
<td>9ca231</td>
<td>2016-09-01 05:10</td>
<td>4624</td>
<td>0.8</td>
<td>0.34</td>
<td>9.2</td>
<td>7</td>
<td>...</td>
<td>0.9</td>
</tr>
<tr>
<td>5e9123</td>
<td>2016-09-01 05:32</td>
<td>4489</td>
<td>2.5</td>
<td>0.85</td>
<td>7.6</td>
<td>2.1</td>
<td>...</td>
<td>0.7</td>
</tr>
<tr>
<td>1e6a7b</td>
<td>2016-09-01 09:12</td>
<td>4688</td>
<td>3.1</td>
<td>0.83</td>
<td>3.6</td>
<td>6.2</td>
<td>...</td>
<td>0.1</td>
</tr>
<tr>
<td>33d693</td>
<td>2016-09-01 14:43</td>
<td>4688</td>
<td>4.1</td>
<td>0.63</td>
<td>4.7</td>
<td>5.1</td>
<td>...</td>
<td>0.019</td>
</tr>
<tr>
<td>7152f3</td>
<td>2016-09-01 19:11</td>
<td>4688</td>
<td>2.7</td>
<td>0.46</td>
<td>3.9</td>
<td>1.4</td>
<td>...</td>
<td>0.03</td>
</tr>
</tbody>
</table>
Actionable Detections

Detections must result in downstream action

Good explanation without being actionable is of little value

**EXAMPLES**

- Policy decisions
- Reset user password
Framework for a Successful Detection

Successful Detections incorporate domain knowledge through disparate datasets and rules.
Case Study 1

Successful detection through combining disparate datasets

PROBLEM STATEMENT
Detect compromised VMs in Azure

HYPOTHESIS
If the VM is sending spam, then it is most likely compromised.

SOLUTION
Use supervised Machine Learning to leverage labeled spam data from Office365 and combine with IPFIX data from Azure.
Case Study 1

Technique Overview

Azure

EXAMPLES
Automated
• All ports with traffic
• Number of connections
• Which TCP flags combination exist
• Many more...

IPFIX Features

Spam Tags come from O365!

Microsoft

RSA Conference 2017
Case Study 1

Technique Overview

IPFIX data

- Spam labeled IPFIX data
- Benign IPFIX data

Machine Learning

New Case

Automated Compromise Detection
### Dataset

#### Why is Network Data Good for Detection?
- No installation required – running on all Azure tenants
- No overload on the VM
- Resilient – cannot be maliciously turned off
- OS independent

#### Feature Sources
- External IPs
- External Ports
- TCP flags

#### Feature Types
- Existence (binary)
- Counts
- Normalized counts

---

**Case Study 1**
Case Study 1

Machine Learning Deep Dive: **Gradient Boosting**

Input data for 1st iteration

Weak learner at 1st iteration

Results
The data points that were incorrectly categorized by the weak learner in the first iteration (the positive examples) are now weighted more.

Simultaneously, the correct points are down weighted.
The data points that were incorrectly categorized in the second iteration (the negative examples) are now weighted more.

Simultaneously, the correct points are down weighted.

Final result is a combination of learners from each iteration.
Model Performance and Productization

Model trained in regular intervals
Size of data: 360GB per day
Within minutes

Classification runs multiple times a day
Completed within seconds

<table>
<thead>
<tr>
<th>Dataset</th>
<th>True Positive Rate</th>
<th>False Positive Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only using Azure IPFIX data</td>
<td>55%</td>
<td>1%</td>
</tr>
<tr>
<td>Using Azure IPFIX and O365 data</td>
<td>81%</td>
<td>1%</td>
</tr>
</tbody>
</table>

26 points improvement
Case Study 2

Successful detection through combining rules and machine learning

**PROBLEM STATEMENT**
Rule based malware detection place hard constraints if something is a malware or not. While they are specific, they have a lot of False Positives, False negatives and are not adaptable.

**HYPOTHESIS**
Can we combine the hard logic of rule based detections with the soft - logic of machine learning systems?

**SOLUTION**
Build two ML models:
1) Model 1 that baselines malware behavior
2) Model 2 that incorporates rules as features
Combine result of two models
Case Study 2

MALWARE DETECTION BACKGROUND
ATP Architecture

Conventional A/V

Detonation Chamber
- Spin up multiple VMs
- Multiple OS and Office versions
- Instrument attachment behavior

Safelinks
- Protects against malicious URLs in Real Time (on click)
Case Study 2

Technique Overview

**PRE-ANALYSIS**
- Hash/Fuzzy Hash
- PE Analyzer
- File Type Analyzer
- PhotoSimilarity
- ...

**DETONATION**
- SysMon
- ETW Logger
- API hooks
- Crash dump
- ...

**POST-ANALYSIS**
- YARA
- Threat Intel
- Network Analysis
- Macro Evidence
- ...

- Fingerprint Model
- Behavioral Model
- Combiner
- Combined Verdict
Case Study 2

Dataset

(SAMPLE)

```xml
<?xml version="1.0" encoding="utf-8"?>
<Journal Types ManagedTargetProcessId="2100" TargetFileNames="Sample.xml" TargetProcessesByName/>

<Events>
  <LoadImage TaskName="ImageLoad" ProcessId="2100" ProcessName="Process:wscript" ParentProcessId="2100">
    <ImageName>\Device\HarddiskVolume2\Windows\System32\wscript.exe</ImageName>
    <ImageSize>155648</ImageSize>
    <TimeDateStamp>1247528568</TimeDateStamp>
    <ImageCheckSum>176852</ImageCheckSum>
  </LoadImage>

  <CallFunction TaskName="CallFunction" ProcessId="2100" ProcessName="Process:wscript" ParentProcessId="2100">
    <FunctionName>CreateMutexA</FunctionName>
    <Param0>Local\_IMSFTHISTORY\_</Param0>
  </CallFunction>

  <CallFunction TaskName="CallFunction" ProcessId="2100" ProcessName="Process:wscript" ParentProcessId="2100">
    <FunctionName>CreateMutexW</FunctionName>
    <Param0>Local\IETId!Mutex</Param0>
  </CallFunction>

  <SetRegistryValue TaskName="SetValueKey" ProcessId="2100" ProcessName="Process:wscript">
    <KeyName>\REGISTRY\MACHINE\SOFTWARE\Microsoft\Tracing\WScript_RASAPI32</KeyName>
    <ValueName>EnableFileTracing</ValueName>
    <Data>0x00000000</Data>
    <DataType>REG_DWORD</DataType>
  </SetRegistryValue>
</Events>
```
Case Study 2

Machine Learning Deep Dive: **Fingerprint Model**

Information gets more granular

<table>
<thead>
<tr>
<th>Call Order</th>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
<th>Level 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Process</td>
<td>LoadImage</td>
<td>SYSTEM</td>
<td>.exe</td>
<td>wscript</td>
</tr>
<tr>
<td>2</td>
<td>Api</td>
<td>CallFunction</td>
<td>CreateMutexA</td>
<td><em>!MSFTHISTORY!</em></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Api</td>
<td>CallFunction</td>
<td>CreateMutexW</td>
<td>!_ETId!Mutex</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Registry</td>
<td>SetRegValue</td>
<td>Tracing</td>
<td>wscript_rasapi32</td>
<td>EnableTracing</td>
</tr>
<tr>
<td>5</td>
<td>Registry</td>
<td>DeleteRegValue</td>
<td>InternetOption</td>
<td>internet settings</td>
<td>ProxyBypass</td>
</tr>
<tr>
<td>6</td>
<td>Process</td>
<td>CreateProcess</td>
<td>NOT_SANDBOX_CHECK</td>
<td>LaunchedViaCom</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Network</td>
<td>AccessNetWork</td>
<td>Wininet_Getaddrinfo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Api</td>
<td>CallFunction</td>
<td>CreateMutexW</td>
<td>RANDOM_STR</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Network</td>
<td>ResolveHost</td>
<td>piglyeleutqq.com</td>
<td>UNKNOWN</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Api</td>
<td>CallFunction</td>
<td>Connect</td>
<td>UNKNOWN</td>
<td></td>
</tr>
</tbody>
</table>
Machine Learning Deep Dive: Fingerprint Model Observations

Benefits of the Action-Chain prototype

- It can be **RESILIENT** to malware obfuscation because it captures the runtime semantics by considering the more **IMPORTANT** details
- Feature extraction is **NON-PARAMETRIC**
  - Would generalize to many situations

Model

Current: L1 Logistic Regression followed by L2 Logistic Regression; weighted samples through cross-validation
Machine Learning Deep Dive: Behavioral Model

Incorporates security domain knowledge into the model

Source of features
- YARA rules
- Static analysis
- Aggregates from Data:
  - Registry keys/values that are changed/created/deleted
  - Mutexes created
  - Number of spawn processes per process detail info

The model works well to detect new types malware
Case Study 2

Model Performance and Productization

Model trained in regular intervals
Size of data: 270GB per day
Completed within minutes

Classification runs multiple times a day
Completed within milliseconds

<table>
<thead>
<tr>
<th>Dataset</th>
<th>True Positive Rate</th>
<th>False Positive Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>YARA rules only</td>
<td>82.6%</td>
<td>0.0178%</td>
</tr>
<tr>
<td>Machine Learning Model 1 + Model 2</td>
<td>93.6%</td>
<td>0.0127%</td>
</tr>
</tbody>
</table>

10 points improvement
For Attack Disruption, We Need to Think Beyond Detection

Current use of Machine Learning

Everything is manual, with little or no intelligence – We need to change this
**Triage incidents, not alerts**

**Anomalous DLL:** rundll32.exe launched as sposql11 on CFE110095

- alert type
- process
- user
- host

**New process uploading:** rundll32.exe to 40.114.40.133 on CFE110095

- alert type
- process
- remote host
- host

**Large transfer:** 50MB to 40.114.40.133 from sqlagent.exe on SQL11006

- alert type
- remote host
- process
- host
Triage incidents, not alerts
Conclusion

Attack Disruption means to shorten blue team kill chain

- **Speed**: Real-time detection
- **Quality**: Reduce false positives
- **React**: Fast triage
# Attack Disruption Checklist

- Combine different datasets
- Labels, Labels, Labels
- Scalable ML solution and expertise

Example Azure services you can leverage:

| Azure Event Hubs | Azure Data Lake | Azure Machine Learning |
Thank you