Let AI Auto-Generate Neural-ASR Rules for OT-specific Attacks via the NLP Approach

Mars Cheng, Head of Cyber Threat & Product Defense Center

Jr-Wei Huang, Sr. Threat Researcher, Cyber Threat & Product Defense Center

TXOne Networks Inc.

August 9, 2025 @DEF CON 33

Mars Cheng from Taiwan



- **Head of Cyber Threat & Product Defense Center at TXOne Networks**
- **Executive Director**, Association of Hackers in Taiwan (HIT/HITCON)
- General Coordinator of HITCON CISO Summit 2025
- Experience
  - ICS/SCADA, IoT, Malware Analysis, Threat Hunting, and Enterprise Security
  - Spoke at Black Hat USA/Europe/MEA, RSA Conference, DEF CON, TROOPERS, HITCON, FIRST, CODE BLUE, SecTor, HITB, SINCON, ICS Cyber Security Conference USA and Asia, and CYBERSEC
  - Instructor of Global Cybersecurity Camp (GCC) 2024, HITCON Training 2022/2021/2020/2019, CCoE Taiwan, Ministry of National Defense, Economic Affairs, Education, and Finance in Taiwan, and Listed companies
  - Cyber Security Auditor of the Taiwan Government
  - General Coordinator of HITCON CISO Summit 2024, 2023, HITCON PEACE 2022 and 2021
  - 10+ CVEs and 3 SCI Journals
  - Member of CHROOT Study Group

HIT

Hacks In Taiwan Conference

txOne networks

Jr-Wei Huang from Taiwan



- **Sr. Threat Researcher of Cyber Threat & Product Defense Center at TXOne Networks Inc.**
  - Specializing in threat hunting, detection engineering, and malware analysis.
  - Hands-on experience in developing EDR product features and designing effective detection strategies
  - He has spoken at major conferences such as HITCON, JASEC, and CYBERSEC Taiwan, covering topics including macOS and Windows security, blue team operations, and detection engineering
  - He has also delivered lectures and training sessions for universities and private companies across Taiwan

Behind this research, Why did we kick off this research?

- The pain points that the factory face are the following situations
  - Can’t Patch
  - Can’t update
  - Of course, can’t update virus patterns regularly
- But the factory is still eager to protect and keep low FP to prevent operational disruption
- But different OT factories have entirely different situations
- And AI ware is sweeping in!
- So…

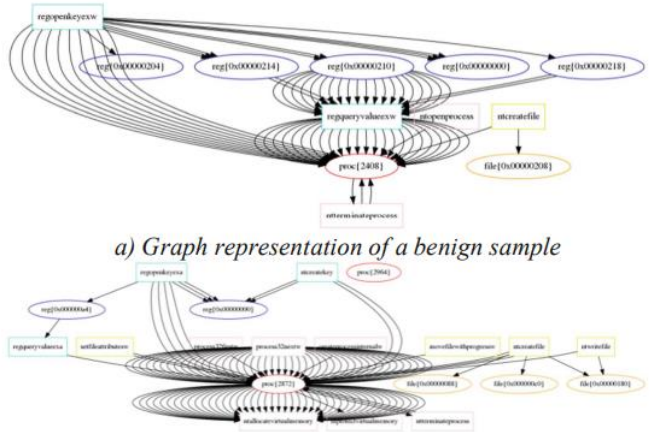
Word Embedding for Malware Dynamic Behaviors

- (2020) Detecting Malware Based on Dynamic Analysis Techniques Using Deep Graph Learning
  - Feature: called Windows APIs, files, CMD/PowerShell commands, registry, etc.
  - Use Word Embedding (Skip-gram) to embedding process behaviors
    - Record benign/malicious process behaviors into a graph representation
    - Focus on the behaviors of a single process and detect it as benign or malicious.
    - Accuracy rate ~96.19%

Detecting Malware Based on Dynamic Analysis Techniques Using Deep Graph Learning

Nguyen Minh Tu[1], Nguyen Viet Hung[1](✉), Phan Viet Anh[1], Cao Van Loi[1], and Nathan Shone[2]

[1] Le Quy Don Technical University, Hanoi, Vietnam
hungnv@lqdtu.edu.vn
[2] Liverpool John-Moore University, Liverpool, UK



a) Graph representation of a benign sample

b) Graph representation of a malware sample

Fig. 1. Graph representation of a malware and benign sample.

Not All OT/ICS Attack Behaviors can be Capture as Malicious, and lack of the information of process child/parent relationship

DrSec: Self-supervised Learning on the AV/DER Event Logs

- (2024) DrSec: Flexible Distributed Representations for Efficient Endpoint Security
  - Feature: Alert Logs from Carbon Black's EDR and SIEM rule from 04/27/2021 to 07/26/2021. Total ~91M processes and ~2.55B events
  - Represent behavior activities into sequences for Doc2Vec
  - Good. Train by unsupervised learning; no labeled dataset is required!

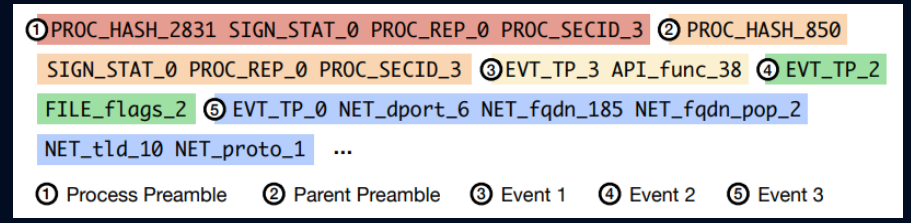
Not All OT/ICS Attack Behaviors can be Capture as Malicious, and lack of the information of process child/parent relationship

DrSec: Flexible Distributed Representations for Efficient Endpoint Security

Mahmood Sharif*†, Pubali Datta‡, Andy Riddle§, Kim Westfall§, Adam Bates§, Vijay Ganti¶, Matthew Lentz‖†, and David Ott†
*Tel Aviv University †VMware ‡University of Massachusetts Amherst §University of Illinois Urbana-Champaign
¶Google ‖Duke University
Email: mahmoods@tauex.tau.ac.il, pdatta@umass.edu, {rriddle2, kw26, batesa}@illinois.edu, ganti.vijay1@gmail.com, mlentz@cs.duke.edu, dott@vmware.com

$$
\begin{aligned}
1 \quad & S \rightarrow S_{proc} \cdot S_{parent} \cdot (E_{net} | E_{cproc} | E_{file} | E_{api} | E_{reg})^* \\
2 \quad & S_{proc} \rightarrow A_{phash} \cdot A_{sign} \cdot A_{rep} \cdot A_{secid} \\
3 \quad & S_{parent} \rightarrow A_{phash} \cdot A_{sign} \cdot A_{rep} \cdot A_{secid} \\
4 \quad & E_{net} \rightarrow \texttt{EVT_TP_0} \cdot A_{dport} \cdot A_{fqdn} \cdot A_{pop} \cdot A_{tld} \cdot A_{proto} \\
5 \quad & E_{cproc} \rightarrow \texttt{EVT_TP_1} \cdot A_{phash} \cdot A_{sign} \cdot A_{rep} \cdot A_{secid} \\
6 \quad & E_{file} \rightarrow \texttt{EVT_TP_2} \cdot A_{acctype} \\
7 \quad & E_{api} \rightarrow \texttt{EVT_TP_3} \cdot A_{fname} \\
8 \quad & E_{reg} \rightarrow \texttt{EVT_TP_4} \cdot A_{regval}
\end{aligned}
$$

Figure 4: **The process abstraction-language defined as a regular grammar** for balancing between the descriptiveness and generalizability of raw telemetry streams.



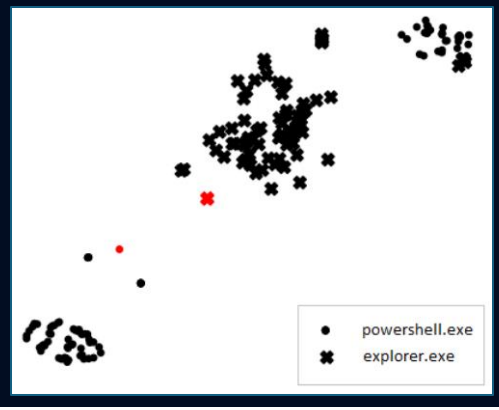
From Multimodal Graph Embedding to Uncover Unknown Attacks

- (2021) NLP-based Entity Behavior Analytics for Malware Detection
  - Instead of considering process behaviors as time-wise activity sequences. Transform all a single process behaviors into Graph-based DAG (Directed Acyclic Graph) of 5-dim access events representation (Registry, Network, Process, DNS, File)
  - Learning all process behaviors (benign + malicious) to successfully discover 2 unknown threats as suspicious events for human experts to check.
    - Accuracy ~94%

Still Lack of the information of process child/parent relationship

NLP-based Entity Behavior Analytics for Malware Detection

Pejman Najafi, *Hasso Plattner Institute*, Potsdam, Germany, pejman.najafi@hpi.de
Daniel Koehler, *Hasso Plattner Institute*, Potsdam, Germany, daniel.koehler@hpi.de
Feng Cheng, *Hasso Plattner Institute*, Potsdam, Germany, feng.cheng@hpi.de
Christoph Meinel, *Hasso Plattner Institute*, Potsdam, Germany, christoph.meinel@hpi.de



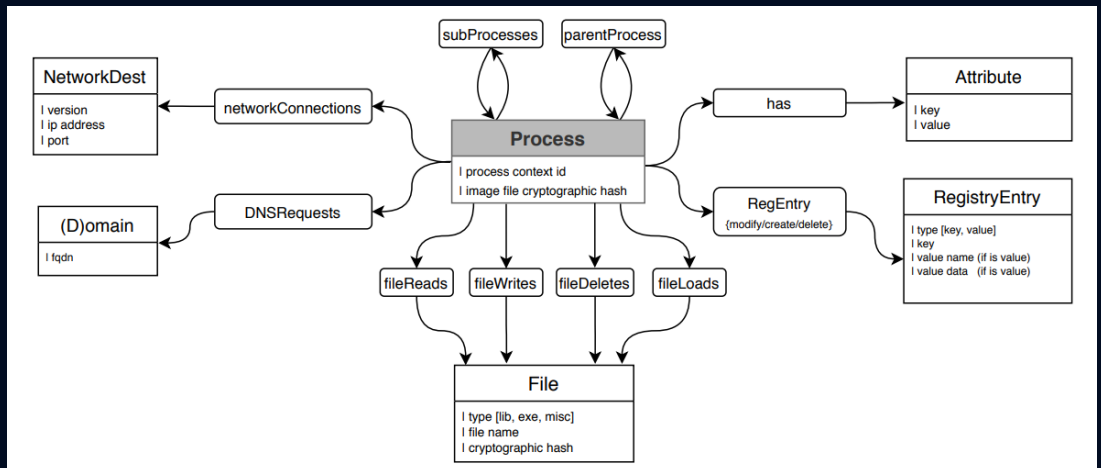
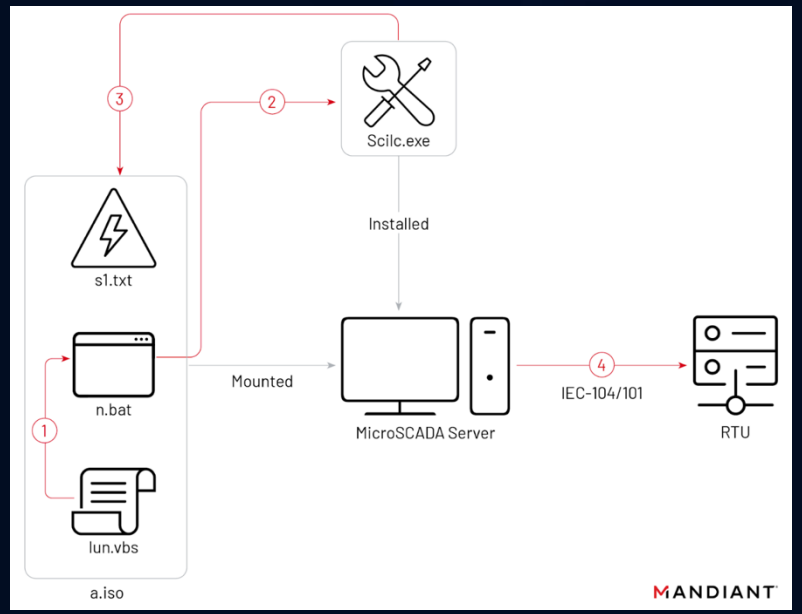


Fig. 2: The schema used to generate graph streams.

Why Process Relationship does Matter for Semantics Detection?

- Russian Sandworm Disrupts Power in Ukraine Using a Novel Attack Against OT
  - OT-specific Living off the land (LOTL) techniques to bypass AV/EDR detection
  - There is no dropper to download the backdoor
    - Abuse the native benign MicroSCADA (scilc.exe) tool to send attack payloads to install Sandworm's backdoor, a new variant of CADDYWIPER.



It is difficult for modern AV/EDR solutions to be aware that it's a LOTL attack if there is a lack of expert rules

However, such LOTL attacks can be infinite variants due to the complex OT/ICS ecosystem, making it impossible to write all the attack rules by security vendors.

Why Process Relationship does Matter for Semantics Detection?

- Expert Contextual Comprehension of Process Chain Integrity is Important!

Case Study 1

C:\Windows\System32\Svchost.exe Great. Seems like OS native services.

C:\Users\Jack\AppData\Temp\SVCH0st.exe Weird! Is that a backdoor installed on the victim?

Case Study 2

Explorer.exe (a.iso) → Wscript.exe → lun.vbs → scilc.exe Now we know it's Sandworm LOTL

Case Study 3 by Microsoft Defender ASR Policy

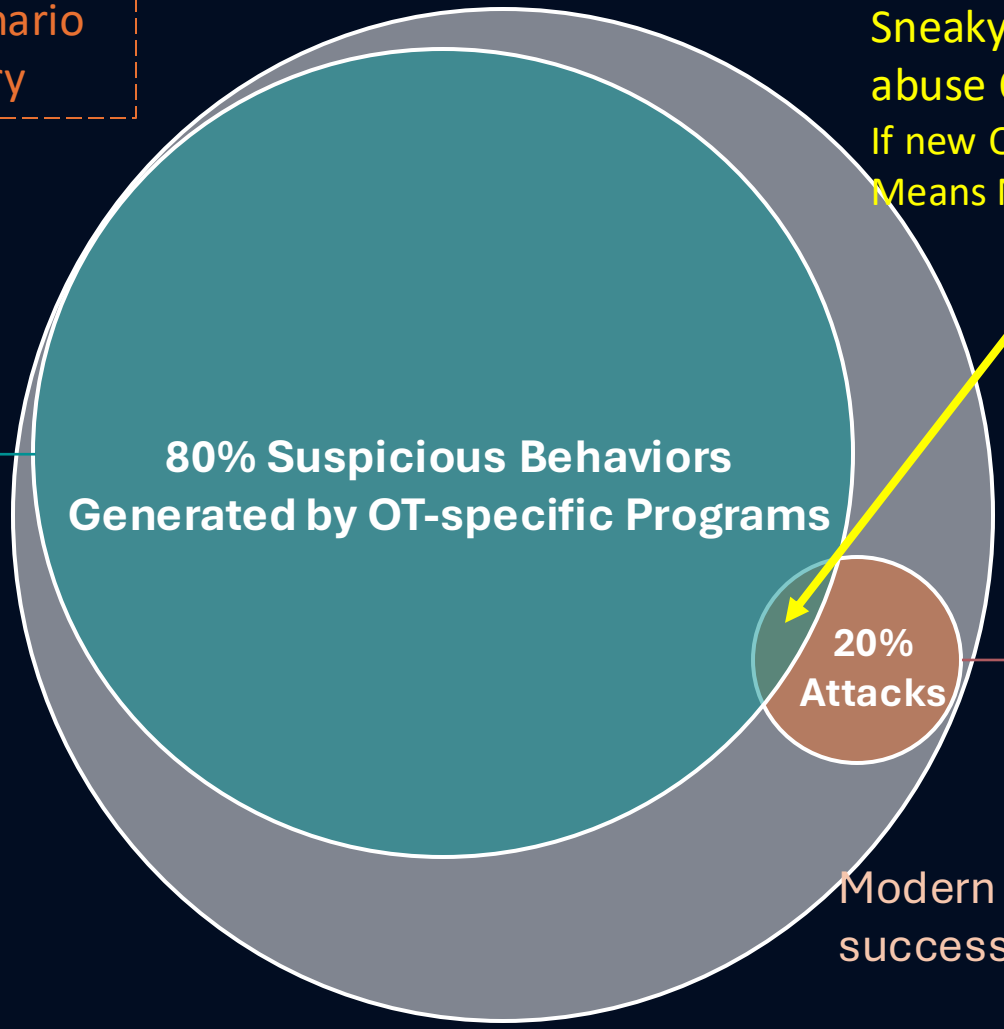
Adobe Reader / MS Office → PowerShell Commands Weird! A new 0-day RCE exploits here?

ASR rule name:	Standard protection rule?
Block abuse of exploited vulnerable signed drivers	Yes
Block Adobe Reader from creating child processes	
Block all Office applications from creating child processes	
Block credential stealing from the Windows local security authority subsystem (lsass.exe)	Yes

```
l_2_0["%programfiles(x86)%\\adobe\\reader\\acrord32.exe"] = 2
l_2_0["%programfiles(x86)%\\adobe\\reader\\reader\\acrord32.exe"] = 2
l_2_0["%programfiles%\\adobe\\acrobat 10.0\\acrobat\\acrobat.exe"] = 2
l_2_0["%programfiles%\\adobe\\acrobat 11.0\\acrobat\\acrobat.exe"] = 2
l_2_0["%programfiles%\\adobe\\acrobat 2015\\acrobat\\acrobat.exe"] = 2
l_2_0["%programfiles%\\adobe\\acrobat 2017\\acrobat\\acrobat.exe"] = 2
l_2_0["%programfiles%\\adobe\\acrobat 5.0\\acrobat\\acrobat.exe"] = 2
```

Modern AV/EDR FP Issue: Capability-Chain != Threat Behaviors

⚠ Require lots of humans to review logs
⚠ It's painful to practical OT/ICS Scenario to review all assets of the whole factory



Sneaky OT/ICS actors
abuse OT-specific benign behaviors
If new OT-specific attacks haven't been discovered
Means No Expert Rules and No Detection here ☹

Modern Multi-Layers AV/EDR
Solution have Global Visibility but
are afraid to Detect as Malicious

Modern Multi-Layers AV/EDR Solution
success captures accurate threat behaviors

txOne networks

Challenge of Malicious Based Detection

a. Recon → b. Initial Access → c. Execution → d. Persistence → e. EoP → f. Lateral Movement → g. Impacts.

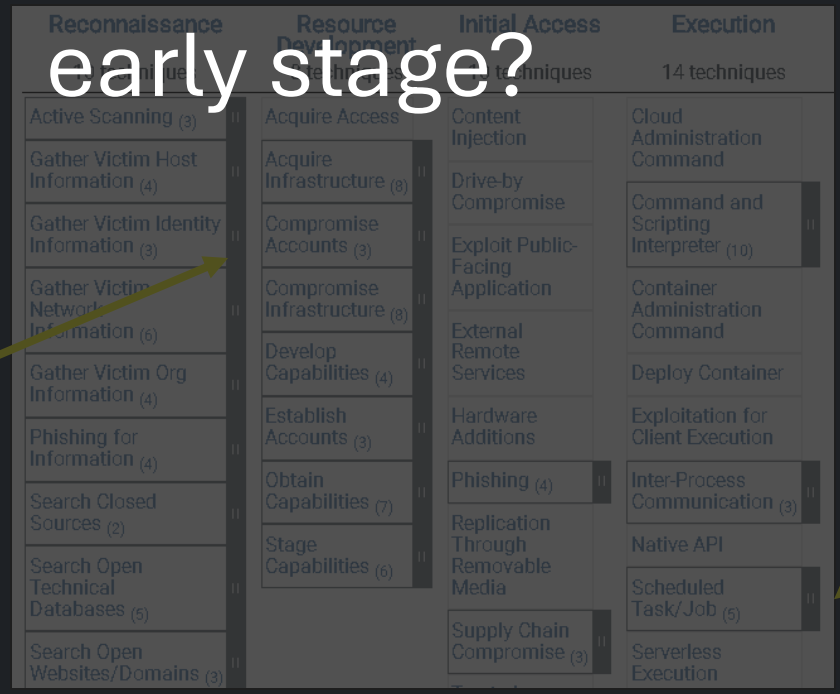
- Cyber Kill Chain and MITRE – We are only aware that bad things happen at the end of the activity
  - Multiple Layer Protection of AV/EDR primarily blocks and detects while phasing d. → f.
    - The clever operators do their best on obfuscation to strip all the possible patterns and prevent detection since stage a. → c.

Can we do it better to be aware of anomalous signals in the early stage?

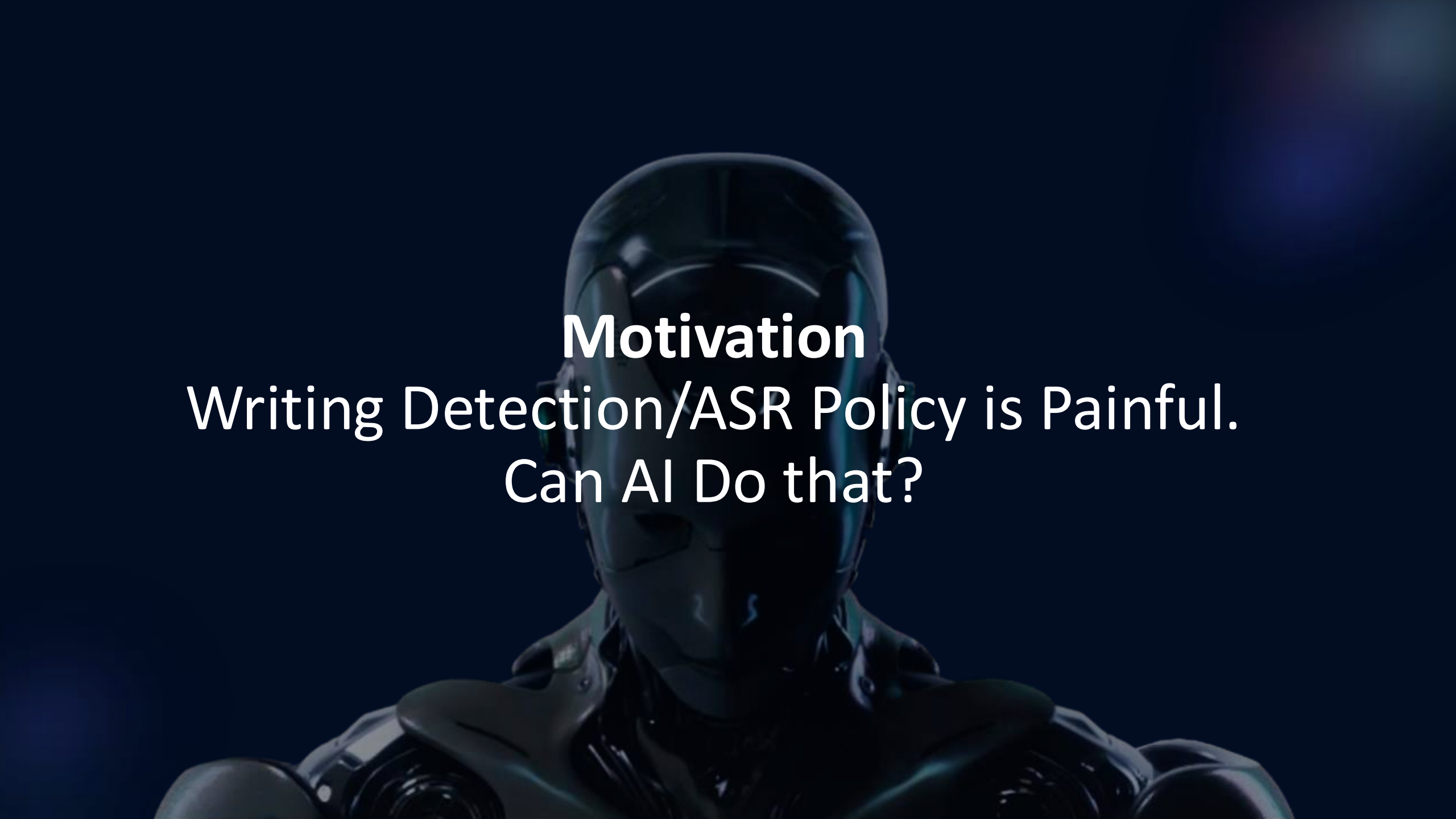
If a shell is controlled by normal engineers
Why does he need to recon the victim's information?
e.g. $whoami or Ncat to sniffer broadcasting traffic

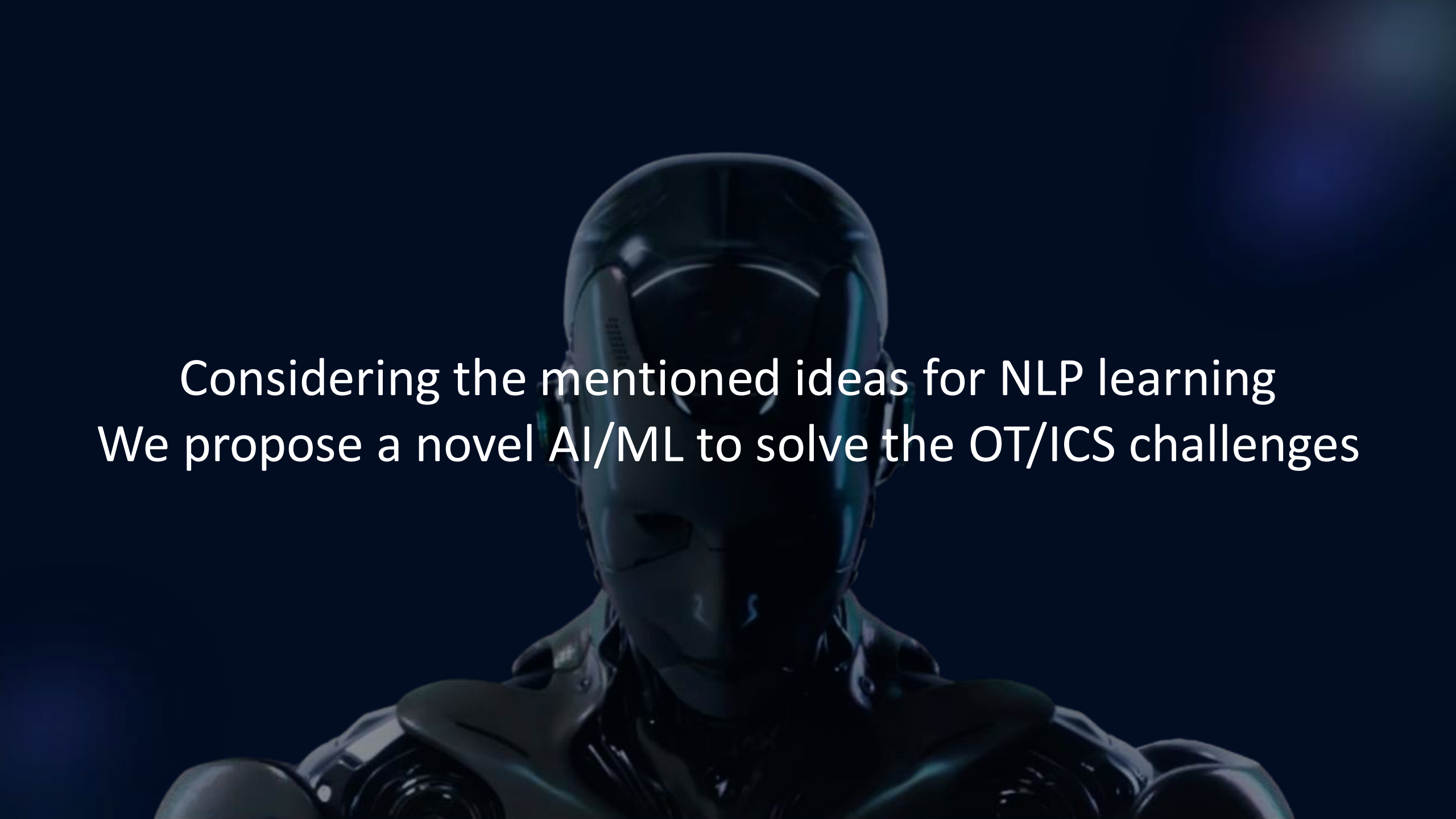
A remote logon by a retired employee account?

PSExec to install a program at midnight? At that time, all engineers have gotten off work. So weird, right?



txOne networks


Motivation
Writing Detection/ASR Policy is Painful.
Can AI Do that?


Considering the mentioned ideas for NLP learning
We propose a novel AI/ML to solve the OT/ICS challenges

The Reality of OT Environment

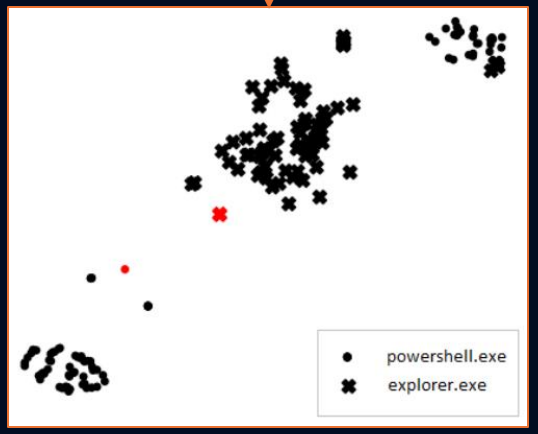
- In OT/ICS, successful threat actors will abuse native benign OT-specific behaviors for attacks, and It is difficult for modern IT AV/EDR to immediately determine malicious behaviors/traffic and block.
- Even a benign OT/ICS service could still possibly trigger suspicious behaviors, which could be considered attacks by IT AV/EDR

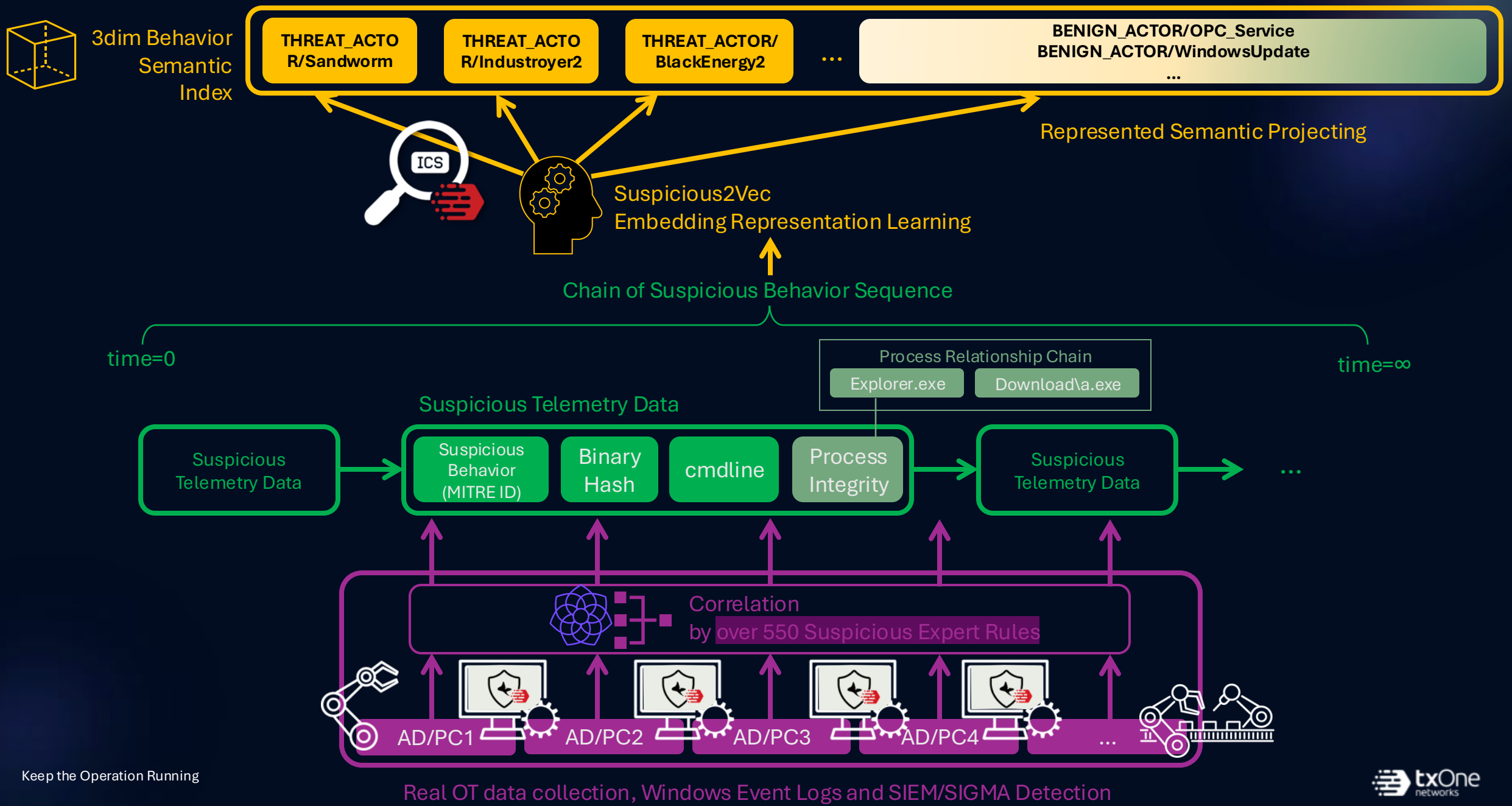
Both benign and malicious actors will use Chain-of-Suspicious Behaviors

Trigger many security events, which is painful for human experts to validate

The Goal of Our AI/ML Model

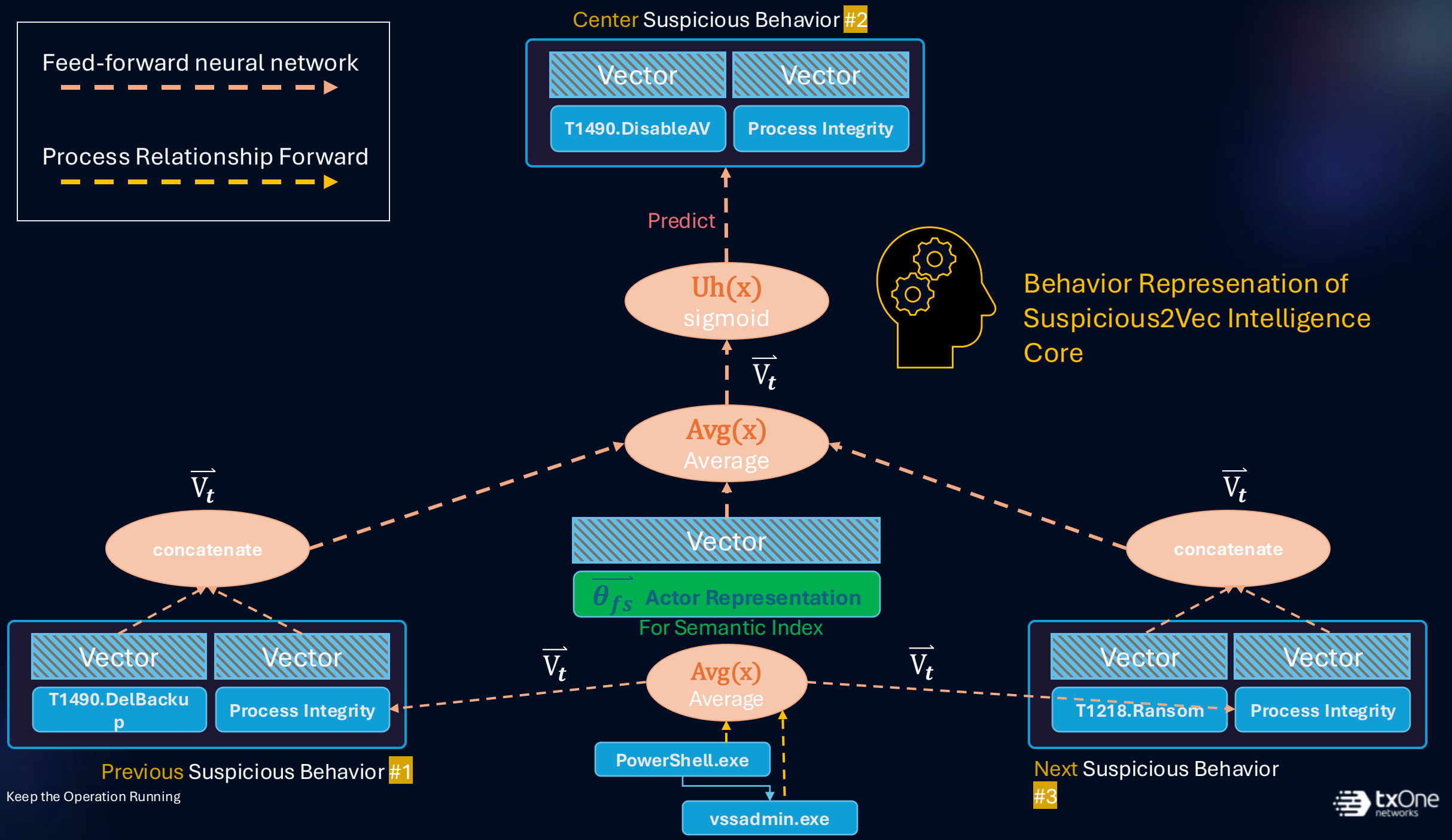
- Single One AI/ML Architecture to **learn the actual client's suspicious behaviors**, no matter whether benign or malicious
- It **must be lightweight** and robust because OT/ICS device computing is expensive to correlate complex events
- The model can learn the suspicious behavior usages from real-world OT data
- Projecting benign or suspicious behaviors into numerical word-embedding vectors
  - We can use SVD decomposition to transform any behaviors into 2-dim (X, Y) map
  - Behavior relationship observation





Keep the Operation Running

txOne networks



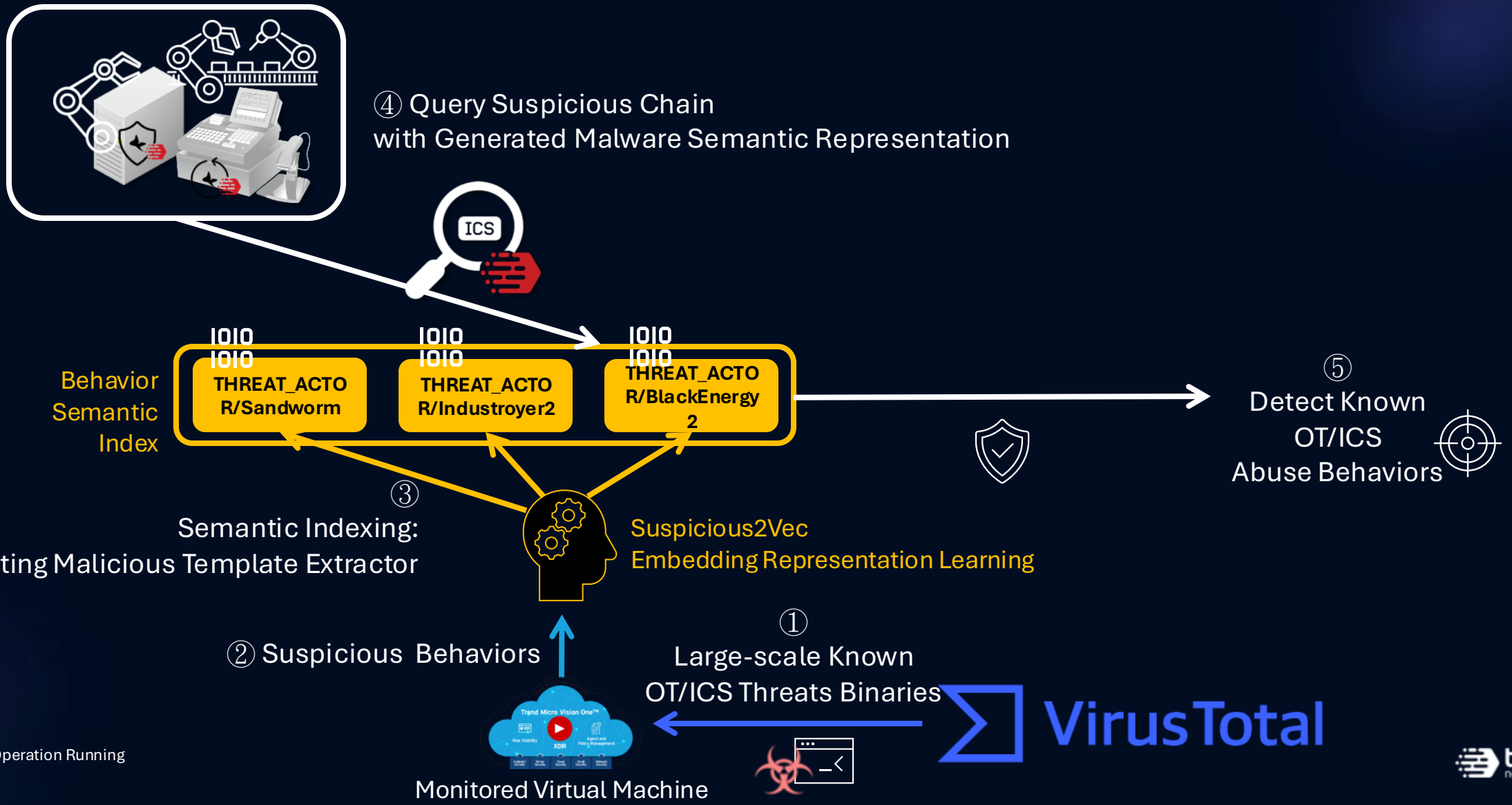
Keep the Operation Running

Experiment and Model Training

- Dataset
  - Human-expert defined **over 550 suspicious rules**
  - Filter suspicious behaviors from all events, including EDR feedback, SIGMA, SIEM, and Windows Event logs
  - From July 2023 to July 2024, a **total of 2,000,000 Benign/Malicious/Suspicious telemetry data** from real OT environments to train
    - Include 52,438 unique binaries, no matter benign or malware
- Model Training
  - 11th Gen Intel(R) Core(TM) i7-1185G7, VRAM 6GB (NVIDIA)
  - Word Embedding dimension: 100 dim
  - 1,062 unique tokens
- Behavior Semantic Indexing
  - Total collect 3,000 unduplicated IT and OT/ICS variant malware from VirusTotal, which are positive by 50+ vendors (Listed Threat Actors on MITRE)
    - Project them into 2 x 100 = 200 dim vector as the representation of behaviors

```
In [6]: event2vec._vocab_itos
Out[6]:
{0: 'reg-ENT_T1547.001-4',
 1: 'SVIQ.EXE',
 2: 'Explorer.exe',
 3: 'Fun.exe',
 4: 'dc.exe',
 5: 'explorer.exe',
 6: 'reg_ransom-d01',
 7: 'winlogon.exe',
 8: 'smss.exe',
 9: 'reg_ransom-w01',
 10: 'reg_ransom-w02',
 11: 'userinit.exe',
 12: 'firefox.exe',
 13: 'services.exe',
 14: 'wininit.exe',
 15: 'addfile_119_low-l02',
 16: 'rundll32.exe',
 17: 'svchost.exe',
 18: 'taskhost.exe',
 19: 'AEGIS-ENT_T1053.005-1',
 20: 'reg_119-004',
 21: 'watchdogGuard.exe',
 22: 'Watchdog.exe',
 23: 'powershell.exe',
```

Automatic Semantic Index Generator of Behavior Chains



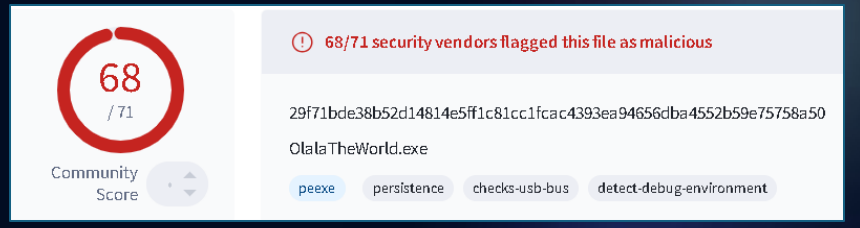
Testing Result by Our Model

- In the whole year experiment, we used over 500+ expert rules to capture 2,000,000 suspicious behaviors generated by 52,438 unique binaries.
  - Include benign OT/ICS and malware samples
- Our neural-network AI model successfully captured 12 variant malware targeting OT-specific fields from 50,000+ samples

1	SHA256	File type	First sub	VT famliy	Mal cnt	Total cnt	Trendmicro
616	afcf5351e4bf704ba97df84cbfffdafae2337c91	exe	2016-04-21T22:43:15	trojan.msil/comine	61	73	TROJ_GEN.R002C0DF721
636	42ac5ff66e21f7e3b7efb150ab931e3c56247253	exe	2006-06-02T23:05:31	trojan.perlovga/genom	68	74	WORM_PERLOVGA.G
815	5dd980ace26b78a30db927cac904363be2c106c0	exe	2010-10-15T16:07:28		68	71	WORM_IMAUT.SME
857	1c74b239d3363f7d9d481c911efb7a2eb1ae5b6d	exe	2022-01-29T20:23:29	trojan.phorpiex/mint	59	71	TROJ_GEN.R002C0PAT22
1367	5cb46b805d3f3c37abf444acfcc2a1c99cd46b78	exe	2006-12-12T03:09:29	worm.jampork	66	75	WORM_VB.DVP
1419	1d2e81739269951b13e205be293e3b0f412b8162	exe	2013-03-18T19:44:46		58	71	BKDR_ANDROM.DF
3102	05c3250f63c021b78dbcc4981f94ec752ae3e48e	exe	2013-03-10T22:16:26		64	71	WORM_IMAUT.SME
3461	b341437a1f94d645e5628d8491f068de1a049fb9	exe	2022-10-26T16:21:36		55	70	TROJ_RANSOMNOTE.RTF
3823	575f68a1e6d8a784e5ab56e3c483488e402c8622	exe	2016-11-04T04:26:08		55	71	TROJ_GEN.R002C0RKO20
3829	49c204e5a0981aa88443cb5c927bcc8bdc98a9d2	exe	2006-06-01T17:20:54	trojan.bkdr	63	72	BKDR_SMALL.DDE
3859	4797283149254a51b852c02d776a5a97f33b2209	exe	2021-09-17T08:56:54	trojan.citeary/doina	64	75	

Case Study #1: Variant WORM_IMAUT.SME

Interpretation: WORM_IMAUT.SME



- **Variant Sample:** 5dd980ace26b78a30db927cac904363be2c106c0 (2023-09-11)
- **Suspicious Behaviors Chain:**
  1. **(Rule) reg-ENT_T1547.001-4**

     Writing Run key makes the program run every time the user logs on. (HKCU\Software\Microsoft\Windows\CurrentVersion\Run)
  2. **(Rule) reg_ransom-w02**

     This entry is a copy of DefaultconnectionSettings, and specifies the configuration used by network connections other than the default connection. Via overwrite HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Connections = SavedLegacySettings
  3. **(Rule) reg_ransom-w01**

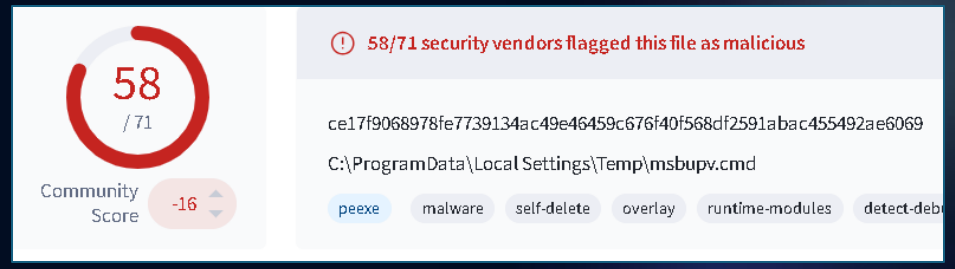
     Set it to 1 to enable a proxy server and 0 to disable one, connecting directly to the internet. Write Registry: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings = ProxyEnable
  4. **(Rule) reg_ransom-d01**

     Clean up Internet Explorer settings (HKCU). Cleanup Registry: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings
- **Visualized Reasoning:**
  - **Suspicious Behavior** ① → ② → ③ → ④
  - This worm tried to register itself as a bootup service for persistence, and also registers itself as the default network proxy to sniff all the local network traffic
  - It also wipes out the default proxy cache of Internet Explorer to ensure traffic monitoring works



Case Study #2: Variant Worm.Gamarue.I

interpretation: Worm.Gamarue.I



- **Variant Sample:** 1d2e81739269951b13e205be293e3b0f412b8162 (2023-09-20)
- **Suspicious Behaviors Chain:**
  1. **(Rule) reg_ransom-d01**

     Clean up Internet Explorer settings (HKCU). Cleanup Registry: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings
  2. **(Rule) reg_119-022.2**

     Disable "Show System Files". Write Registry: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced = ShowSuperHidden
  3. **(Rule) reg_119-022.1**

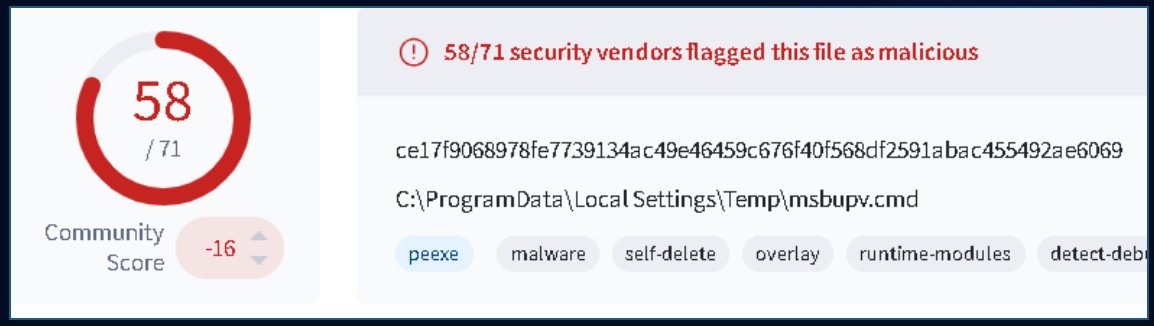
     Disable "Show Hidden Files". Write Registry: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced = Hidden
  4. **(Rule) reg_ransom-w02**

     This entry is a copy of DefaultconnectionSettings, and specifies the configuration used by network connections other than the default connection. Write Registry: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\Connections = SavedLegacySettings
  5. **(Rule) reg_ransom-w01**

     Set it to 1 to enable a proxy server and 0 to disable one, connecting directly to the internet. Write Registry: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings = ProxyEnable

interpretation: Worm.Gamarue.I

- **Variant Sample:** 1d2e81739269951b13e205be293e3b0f412b8162 (2023-09-20)
- **Suspicious Behaviors Chain:**
  1. **(Rule) reg_ransom-d01**
  2. **(Rule) reg_119-022.2**
  3. **(Rule) reg_119-022.1**
  4. **(Rule) reg_ransom-w02**
  5. **(Rule) reg_ransom-w01**



- **Visualized Reasoning:**
  - **Suspicious Behavior ① → ② + ③ → ④ → ⑤**
  - The rootkit set itself as super hidden system file, to prevent victim see the binary files on NTFS by Explorer.exe
  - This worm tried to register itself as a bootup service for persistence, and also register itself as default network proxy to sniffer all the local network traffic.
  - It also wipe out the default proxy cache of Internet Explorer, to make sure traffic monitoring works.

Comparison of 2 Variant Worm Types

WORM_IMAUT.SME: ① → ② → ③ → ④

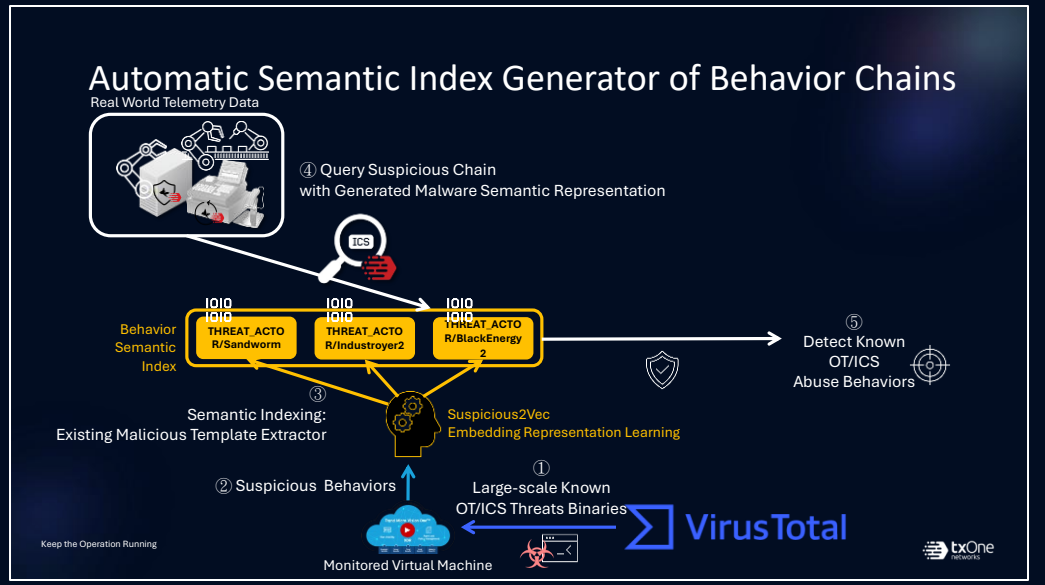
Worm.Gamarue.I: ① → ② + ③ → ④ → ⑤

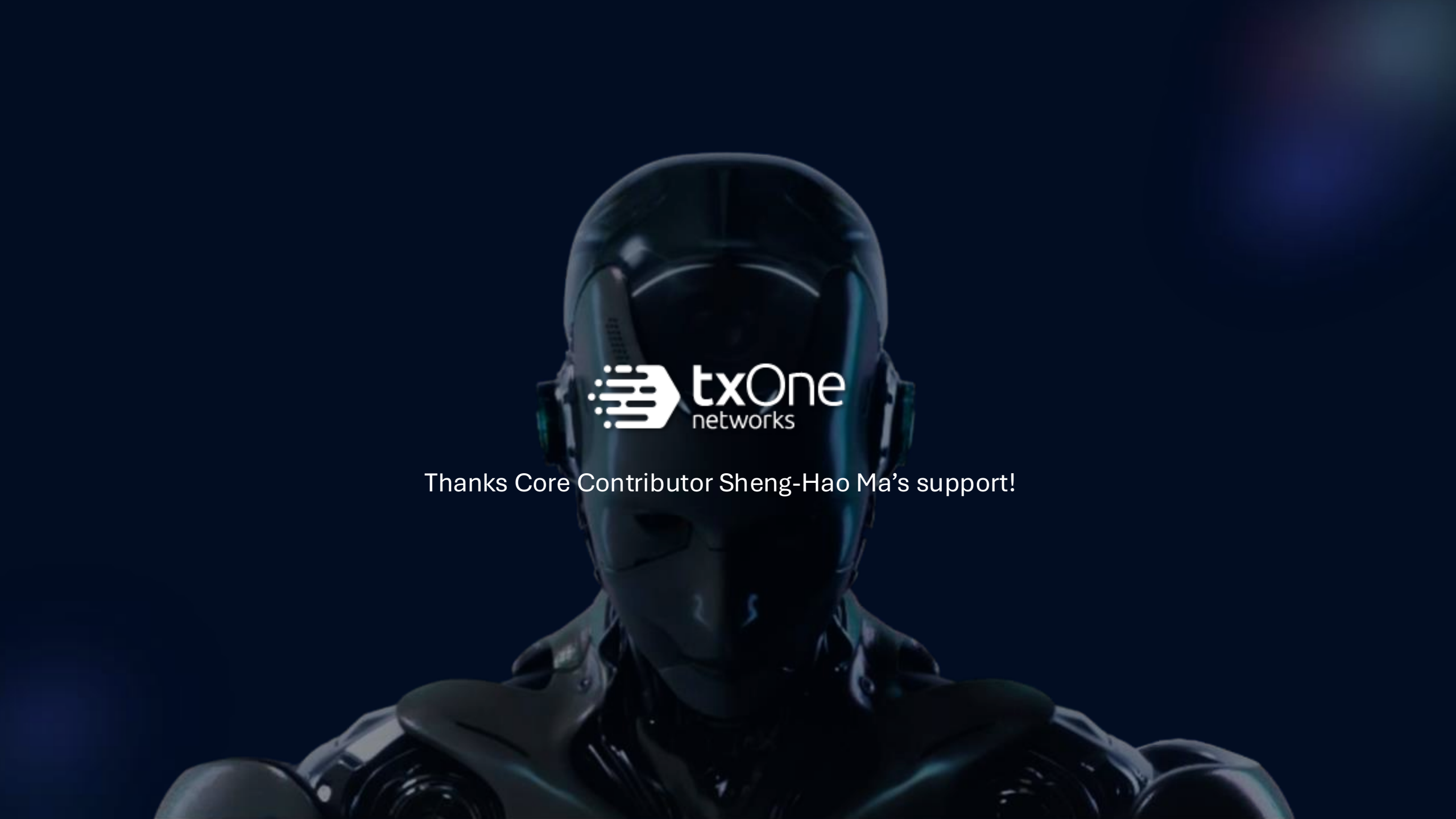
$$Representation\ of\ \overrightarrow{V_{Worm_type}}$$

txOne networks

Takeaways

- Through the environmental characteristics of OT and its real data training, the model can hunt these variants of malicious programs, regardless of age, and confirm that they are effective
- Fast-evolved variant hacking tools are challenging to define by existing expert rules, so it is essential to rapidly and dynamically evolve and confirm that detection capabilities are always available and applicable to OT environments with limited computing resources
- Continue to strengthen its models and experiments to ensure that the models can be widely used in real OT environments, such as extend coverage to all OT known attack techniques





Thanks Core Contributor Sheng-Hao Ma’s support!