black hat EUROPE 2025

# MODELS AS MALWARE

## Attacking and Defending the AI Supply Chain

# THE AI SUPPLY CHAIN

- is a SOFTWARE supply chain!

(generally, for Python)

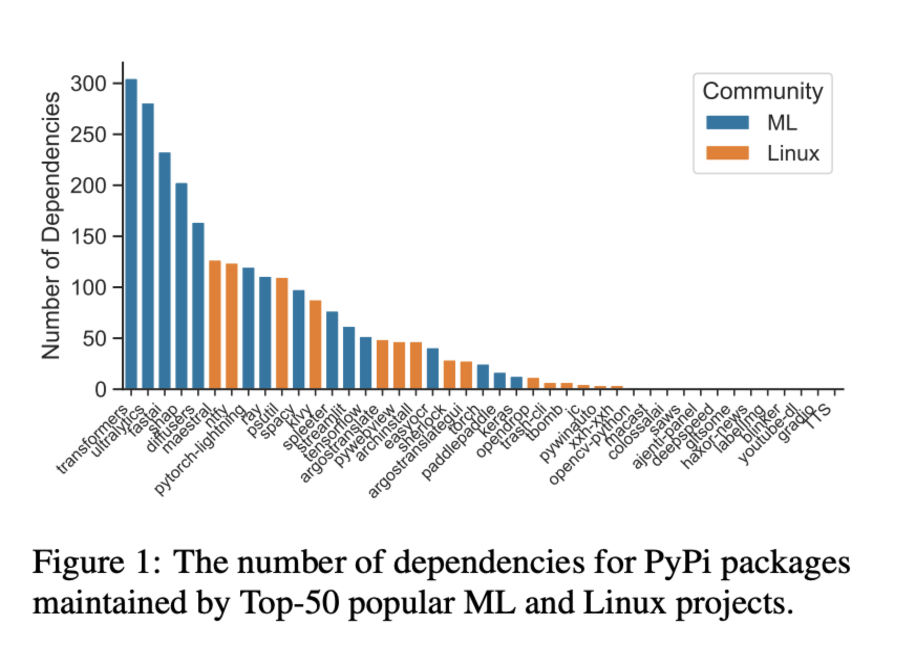


Figure 1: The number of dependencies for PyPi packages maintained by Top-50 popular ML and Linux projects.

Supply-chain attacks in machine learning frameworks Y Gao, I Shumailov, K Fawaz - MLSys 2025

# LINES BETWEEN DATA AND CODE BLUR

- a product of complexity (ML algorithms) and convenience

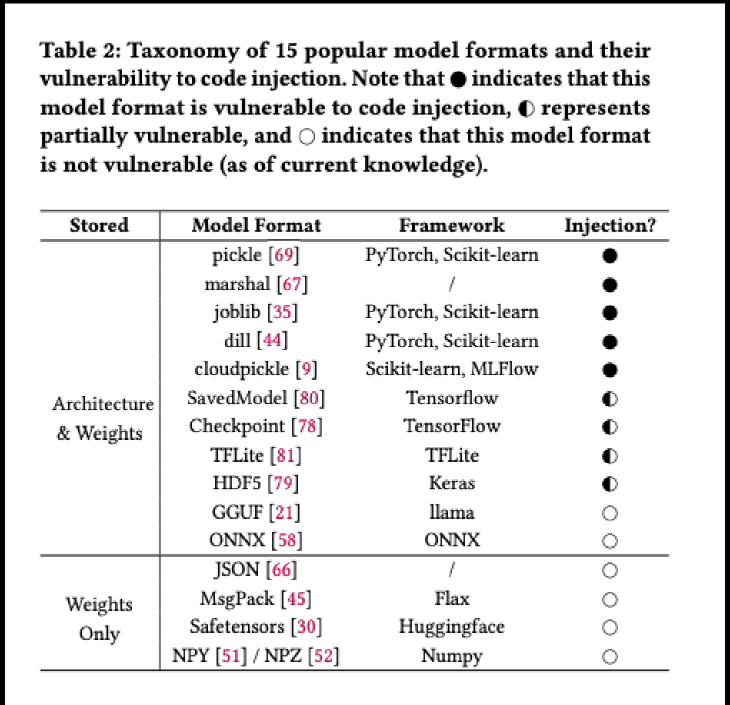


**Table 2: Taxonomy of 15 popular model formats and their vulnerability to code injection. Note that ● indicates that this model format is vulnerable to code injection, ◐ represents partially vulnerable, and ○ indicates that this model format is not vulnerable (as of current knowledge).**

| Stored | Model Format | Framework | Injection? |
|---|---|---|---|
| Architecture & Weights | pickle [69] | PyTorch, Scikit-learn | ● |
| | marshal [67] | / | ● |
| | joblib [35] | PyTorch, Scikit-learn | ● |
| | dill [44] | PyTorch, Scikit-learn | ● |
| | cloudpickle [9] | Scikit-learn, MLFlow | ● |
| | SavedModel [80] | Tensorflow | ◐ |
| | Checkpoint [78] | TensorFlow | ◐ |
| | TFLite [81] | TFLite | ◐ |
| | HDF5 [79] | Keras | ◐ |
| | GGUF [21] | llama | ○ |
| | ONNX [58] | ONNX | ○ |
| Weights Only | JSON [66] | / | ○ |
| | MsgPack [45] | Flax | ○ |
| | Safetensors [30] | Huggingface | ○ |
| | NPY [51] / NPZ [52] | Numpy | ○ |

Towards Measuring Malicious Code Poisoning Attacks on Pre-trained Models Hubs J. Zhao, S. Wang, Y. Zhao – ASE 2024

# THERE ARE EXPLOITS TO BE HAD!

- **Serialization / Deserialization**

*Embedding malicious code directly into a serialized model, which then executes during deserialization.*

- **Computation Graphs**

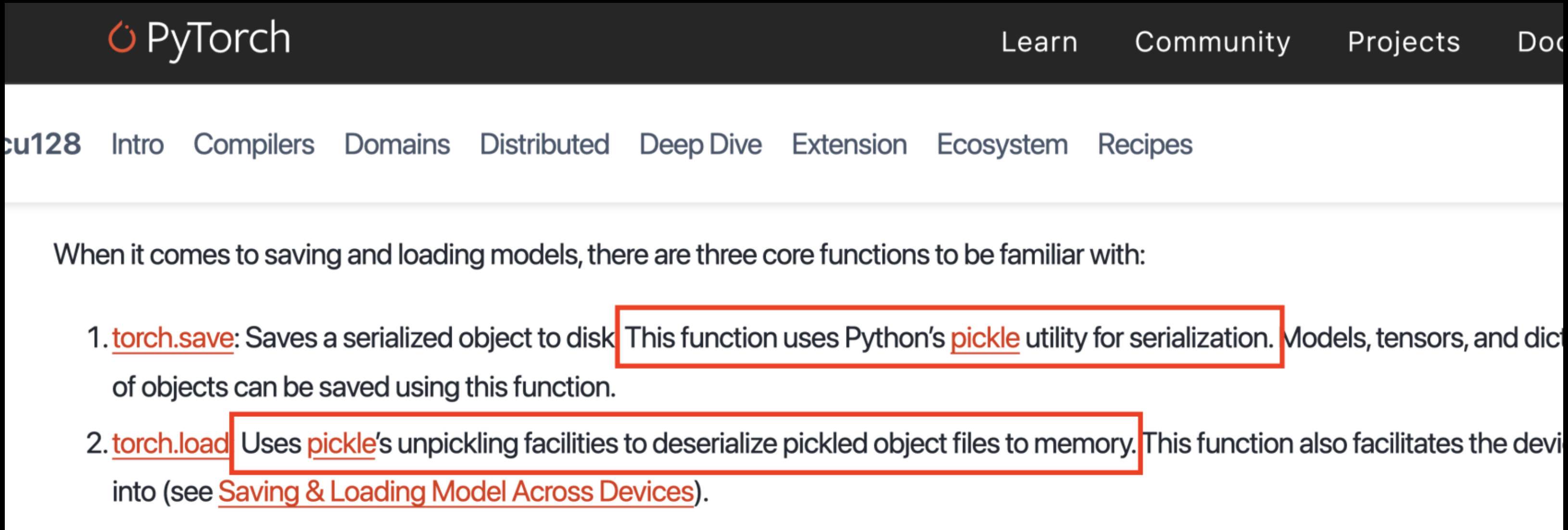
*Adding paths to the model's computation graph to modify model outputs.*

- **Steganography within model weights and biases**

*Using steganography techniques (e.g., LSB) to encode malware within the floating-point values of model weights.*

# SOME MODELS ARE PICKLES

- PyTorch (one of the most popular ML frameworks for deep learning) uses Python's pickle under the hood.

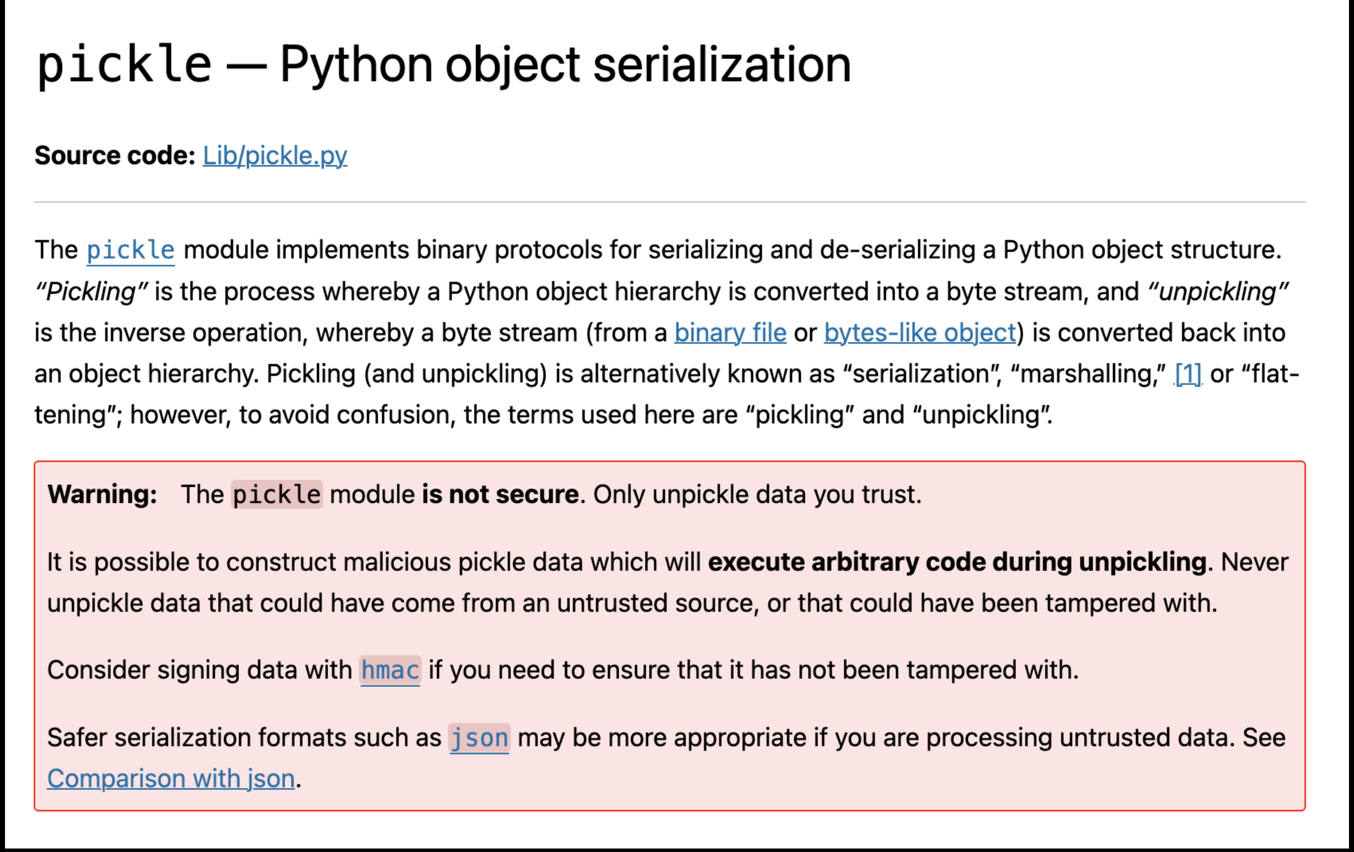


When it comes to saving and loading models, there are three core functions to be familiar with:

1. torch.save: Saves a serialized object to disk. This function uses Python's pickle utility for serialization. Models, tensors, and dict of objects can be saved using this function.
2. torch.load: Uses pickle's unpickling facilities to deserialize pickled object files to memory. This function also facilitates the devi into (see Saving & Loading Model Across Devices).

https://docs.pytorch.org/tutorials/beginner/saving_loading_models.html

# PICKLES ARE VULNERABLE, BUT ...



## `pickle` — Python object serialization

**Source code:** Lib/pickle.py

The `pickle` module implements binary protocols for serializing and de-serializing a Python object structure. *"Pickling"* is the process whereby a Python object hierarchy is converted into a byte stream, and *"unpickling"* is the inverse operation, whereby a byte stream (from a binary file or bytes-like object) is converted back into an object hierarchy. Pickling (and unpickling) is alternatively known as "serialization", "marshalling," [1] or "flattening"; however, to avoid confusion, the terms used here are "pickling" and "unpickling".

**Warning:** The `pickle` module **is not secure**. Only unpickle data you trust.

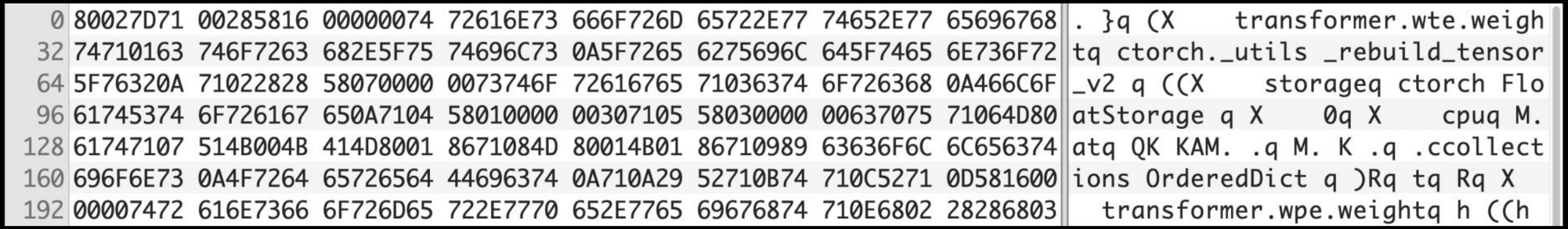
It is possible to construct malicious pickle data which will **execute arbitrary code during unpickling**. Never unpickle data that could have come from an untrusted source, or that could have been tampered with.

Consider signing data with `hmac` if you need to ensure that it has not been tampered with.

Safer serialization formats such as `json` may be more appropriate if you are processing untrusted data. See Comparison with json.

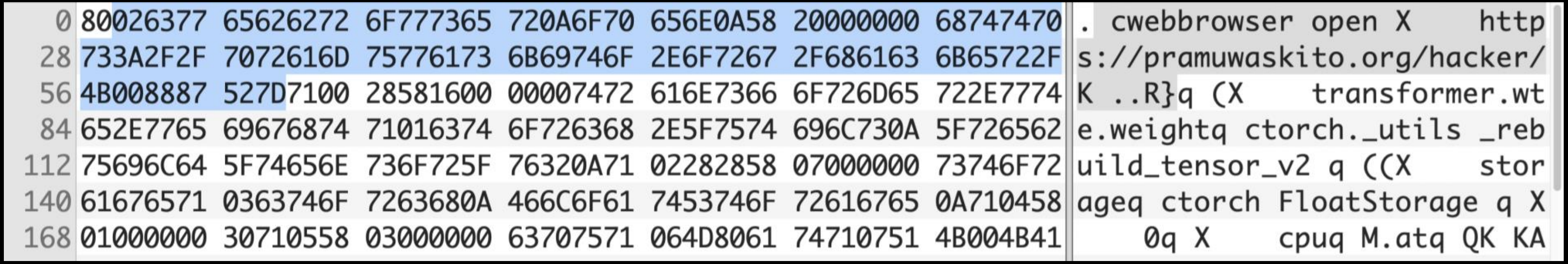
https://docs.python.org/3/library/pickle.html

# EXPLOITS ARE STATICALLY DETECTABLE



```
  0 80027D71 00285816 00000074 72616E73 666F726D 65722E77 74652E77 65696768 . }q (X    transformer.wte.weigh
 32 74710163 746F7263 682E5F75 74696C73 0A5F7265 6275696C 645F7465 6E736F72 tq ctorch._utils _rebuild_tensor
 64 5F76320A 71022828 58070000 0073746F 72616765 71036374 6F726368 0A466C6F _v2 q ((X    storageq ctorch Flo
 96 61745374 6F726167 650A7104 58010000 00307105 58030000 00637075 71064D80 atStorage q X    0q X    cpuq M.
128 61747107 514B004B 414D8001 8671084D 80014B01 86710989 63636F6C 6C656374 atq QK KAM. .q M. K .q .ccollect
160 696F6E73 0A4F7264 65726564 44696374 0A710A29 52710B74 710C5271 0D581600 ions OrderedDict q )Rq tq Rq X
192 00007472 616E7366 6F726D65 722E7770 652E7765 69676874 710E6802 28286803   transformer.wpe.weightq h ((h
```

CLEAN

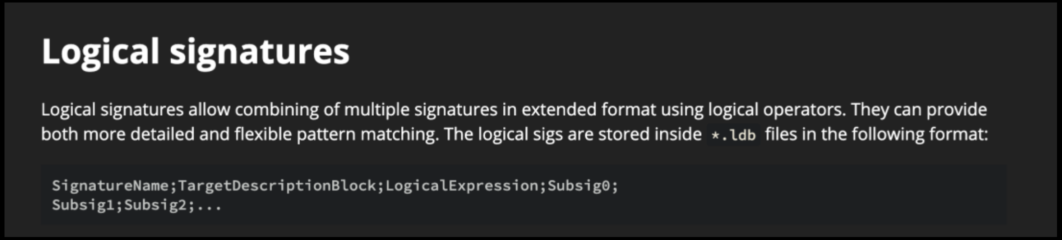


```
  0 80026377 65626272 6F777365 720A6F70 656E0A58 20000000 68747470 . cwebbrowser open X    http
 28 733A2F2F 7072616D 75776173 6B69746F 2E6F7267 2F686163 6B65722F s://pramuwaskito.org/hacker/
 56 4B008887 527D7100 28581600 00007472 616E7366 6F726D65 722E7774 K ..R}q (X    transformer.wt
 84 652E7765 69676874 71016374 6F726368 2E5F7574 696C730A 5F726562 e.weightq ctorch._utils _reb
112 75696C64 5F74656E 736F725F 76320A71 02282858 07000000 73746F72 uild_tensor_v2 q ((X    stor
140 61676571 0363746F 7263680A 466C6F61 7453746F 72616765 0A710458 ageq ctorch FloatStorage q X
168 01000000 30710558 03000000 63707571 064D8061 74710751 4B004B41     0q X    cpuq M.atq QK KA
```

INFECTED

# CLAMAV TO THE RESCUE

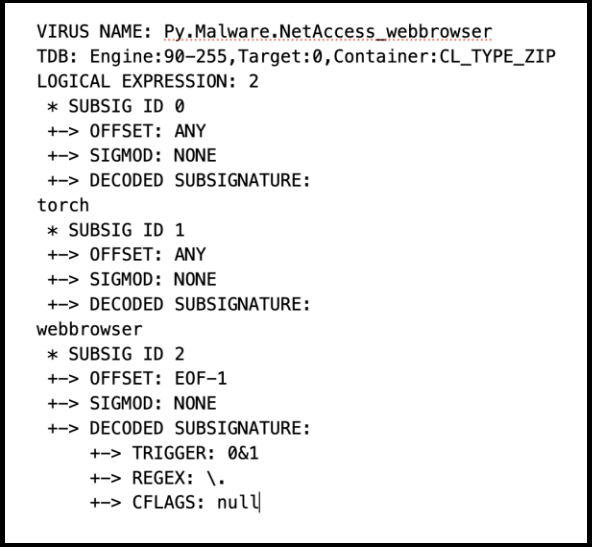
ClamAV® is an open-source antivirus engine for detecting trojans, viruses, malware & other malicious threats.



## Logical signatures

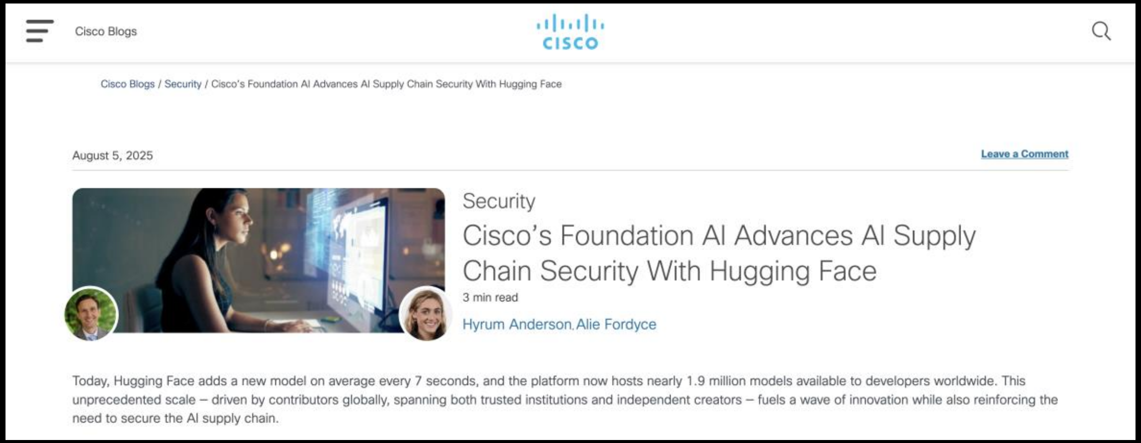
Logical signatures allow combining of multiple signatures in extended format using logical operators. They can provide both more detailed and flexible pattern matching. The logical sigs are stored inside `*.ldb` files in the following format:

```
SignatureName;TargetDescriptionBlock;LogicalExpression;Subsig0;
Subsig1;Subsig2;...
```



```
VIRUS NAME: Py.Malware.NetAccess_webbrowser
TDB: Engine:90-255,Target:0,Container:CL_TYPE_ZIP
LOGICAL EXPRESSION: 2
 * SUBSIG ID 0
 +-> OFFSET: ANY
 +-> SIGMOD: NONE
 +-> DECODED SUBSIGNATURE:
torch
 * SUBSIG ID 1
 +-> OFFSET: ANY
 +-> SIGMOD: NONE
 +-> DECODED SUBSIGNATURE:
webbrowser
 * SUBSIG ID 2
 +-> OFFSET: EOF-1
 +-> SIGMOD: NONE
 +-> DECODED SUBSIGNATURE:
     +-> TRIGGER: 0&1
     +-> REGEX: \.
     +-> CFLAGS: null
```

# HUGGINGFACE USES CLAMAV!



Cisco Blogs

Cisco Blogs / Security / Cisco's Foundation AI Advances AI Supply Chain Security With Hugging Face

August 5, 2025

Leave a Comment

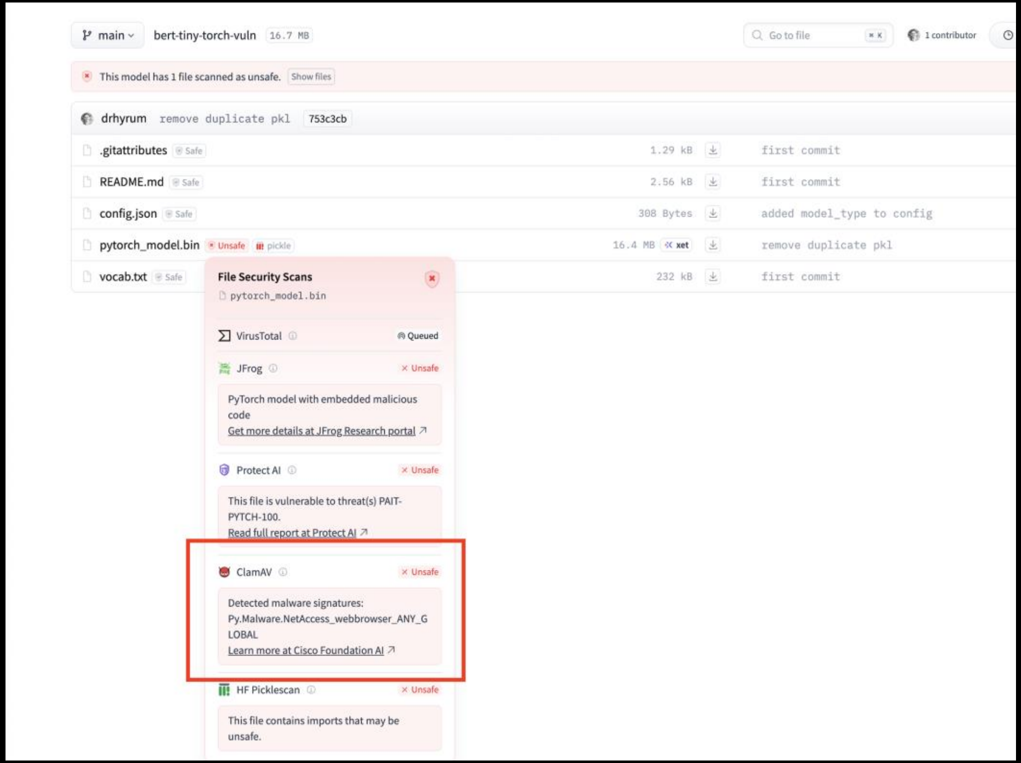
Security

## Cisco's Foundation AI Advances AI Supply Chain Security With Hugging Face

3 min read

Hyrum Anderson, Alie Fordyce

Today, Hugging Face adds a new model on average every 7 seconds, and the platform now hosts nearly 1.9 million models available to developers worldwide. This unprecedented scale – driven by contributors globally, spanning both trusted institutions and independent creators – fuels a wave of innovation while also reinforcing the need to secure the AI supply chain.



bert-tiny-torch-vuln 16.7 MB

This model has 1 file scanned as unsafe. Show files

drhyrum remove duplicate pkl 753c3cb

| | | | |
|---|---|---|---|
| .gitattributes | Safe | 1.29 kB | first commit |
| README.md | Safe | 2.56 kB | first commit |
| config.json | Safe | 308 Bytes | added model_type to config |
| pytorch_model.bin | Unsafe pickle | 16.4 MB xet | remove duplicate pkl |
| vocab.txt | Safe | 232 kB | first commit |

**File Security Scans**

pytorch_model.bin

VirusTotal — Queued

JFrog — × Unsafe

PyTorch model with embedded malicious code
Get more details at JFrog Research portal ↗

Protect AI — × Unsafe

This file is vulnerable to threat(s) PAIT-PYTCH-100.
Read full report at Protect AI ↗

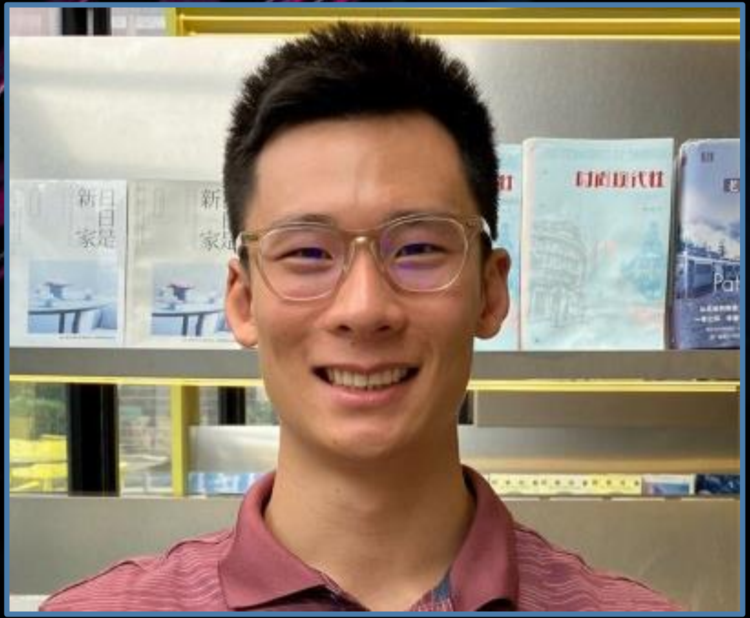
ClamAV — × Unsafe

Detected malware signatures:
Py.Malware.NetAccess_webbrowser_ANY_GLOBAL
Learn more at Cisco Foundation AI ↗

HF Picklescan — × Unsafe

This file contains imports that may be unsafe.

# LIVE DEMO



NATHAN
CHANG

nathchan@cisco.com

ROJIS
LANDISMANIS

roeeland@cisco.com