Commit b6773d3

3 people authored on Jan 29 · 13 / 14 · Verified

```
Disallow TFSMLayer deserialization in safe_mode to prevent external SavedModel
execution (#22035)

* Implement safe mode checks in TFSMLayer

Added safe mode checks for loading TFSMLayer from external SavedModels.

* Update keras/src/export/tfsm_layer.py

Co-authored-by: gemini-code-assist[bot] <176961590+gemini-code-
assist[bot]@users.noreply.github.com>

* Align logic with __init__ method for robust checks

Co-authored-by: gemini-code-assist[bot] <176961590+gemini-code-
assist[bot]@users.noreply.github.com>

* Fix indentation and formatting in tfsm_layer.py

* Add setup method to enable unsafe deserialization

Enable unsafe deserialization for TFSM Layer tests.

* Update TFSMLayer initialization in tests

* Fix import for TFSMLayer in tfsm_layer_test.py

* Remove safe_mode check from TFSMLayer.__init__()

The safe_mode check should only be in from_config(), not __init__().

Direct instantiation (TFSMLayer(filepath=...)) is a legitimate use case
where the user explicitly creates the layer. The security concern is
only during deserialization of untrusted .keras files, which goes
through from_config().

This allows attackers to create malicious .keras files while still
blocking victims from loading them with safe_mode=True.

* Implement tests for TFSMLayer safe mode functionality

Add comprehensive tests for TFSMLayer safe_mode behavior:
- test_safe_mode_direct_instantiation_allowed: Verifies direct
```

```
  TFSMLayer instantiation works as expected
- test_safe_mode_from_config_blocked: Verifies from_config() raises
  ValueError when safe_mode=True
- test_safe_mode_from_config_allowed_when_disabled: Verifies
  from_config() works with safe_mode=False
- test_safe_mode_model_loading_blocked: Tests the full attack scenario
  where loading a .keras file with safe_mode=True is blocked

* Clarify test docstrings in tfsm_layer_test.py

Updated test docstrings for clarity on instantiation and loading behavior.

* Invoke model with random input in tfsm_layer tests

Added model invocation with random input to tests for TFSMLayer.

* Set safe_mode default to True in from_config method

* Update tfsm_layer_test.py

* Update tfsm_layer_test.py

* Update tfsm_layer_test.py to original

* New test case tfsm_layer_test.py

* Update Comments tfsm_layer.py

* Update tfsm_layer_test.py

* Update tfsm_layer.py

* Update tfsm_layer.py to remove ruff errors

* Update tfsm_layer.py

* Update tfsm_layer_test.py

* Update tfsm_layer.py

* Update tfsm_layer.py

* Update tfsm_layer_test.py

* Update tfsm_layer.py format fix

Changes in format

* Update tfsm_layer.py

* Update tfsm_layer_test.py

* Update tfsm_layer.py
```

```
* Update tfsm_layer_test.py

* Fixes unnecessary changes tfsm_layer.py

* Added new test case tfsm_layer_test.py

* Set `safe_mode=None` in `from_config`, which fixes the unit tests.

Also re-added empty lines.

* Remove unneeded `custom_objects` in unit tests.

---------

Co-authored-by: gemini-code-assist[bot] <176961590+gemini-code-
assist[bot]@users.noreply.github.com>
Co-authored-by: Fabien Hertschuh <1091026+hertschuh@users.noreply.github.com>
```

master (#22035) · v3.14.0

1 parent 38687a6 commit b6773d3

2 files changed +66 -3 lines changed

Top

Filter files...

- keras/src/export
  - tfsm_layer.py
  - tfsm_layer_test.py

2 files changed +66 -3 lines changed

Top

Search within code

keras/src/export/tfsm_layer.py

```
@@ -2,6 +2,7 @@
from keras.src import layers
from keras.src.api_export import keras_export
from keras.src.export.saved_model import _list_variables_used_by_fns
+ from keras.src.saving import serialization_lib
from keras.src.utils.module_utils import tensorflow as tf


@@ -146,3 +147,36 @@ def get_config(self):
```

```
146  147              "call_training_endpoint": self.call_training_endpoint,
147  148          }
148  149          return {**base_config, **config}
     150  +
     151  +    @classmethod
     152  +    def from_config(cls, config, custom_objects=None, safe_mode=None):
     153  +        """Creates a TFSMLayer from its config.
     154  +        Args:
     155  +            config: A Python dictionary, typically the output of `get_config`.
     156  +            custom_objects: Optional dictionary mapping names to custom
          objects.
     157  +            safe_mode: Boolean, whether to disallow loading TFSMLayer.
     158  +                When `safe_mode=True`, loading is disallowed because TFSMLayer
     159  +                loads external SavedModels that may contain attacker-controlled
     160  +                executable graph code. Defaults to `True`.
     161  +        Returns:
     162  +            A TFSMLayer instance.
     163  +        """
     164  +        # Follow the same pattern as Lambda layer for safe_mode handling
     165  +        effective_safe_mode = (
     166  +            safe_mode
     167  +            if safe_mode is not None
     168  +            else serialization_lib.in_safe_mode()
     169  +        )
     170  +
     171  +        if effective_safe_mode is not False:
     172  +            raise ValueError(
     173  +                "Requested the deserialization of a `TFSMLayer`, which "
     174  +                "loads an external SavedModel. This carries a potential risk "
     175  +                "of arbitrary code execution and thus it is disallowed by "
     176  +                "default. If you trust the source of the artifact, you can "
     177  +                "override this error by passing `safe_mode=False` to the "
     178  +                "loading function, or calling "
     179  +                "`keras.config.enable_unsafe_deserialization()."
     180  +            )
     181  +
     182  +        return cls(**config)
```

keras/src/export/tfsm_layer_test.py

```
@@ -114,19 +114,48 @@ def test_serialization(self):
```

```
114  114
115  115            # Test reinstantiation from config
116  116            config = reloaded_layer.get_config()
117       -        rereloaded_layer = tfsm_layer.TFSMLayer.from_config(config)
     117  +        rereloaded_layer = tfsm_layer.TFSMLayer.from_config(
     118  +            config, safe_mode=False
     119  +        )
118  120            self.assertAllClose(rereloaded_layer(ref_input), ref_output, atol=1e-7)
119  121
120  122            # Test whole model saving with reloaded layer inside
121  123            model = models.Sequential([reloaded_layer])
122  124            temp_model_filepath = os.path.join(self.get_temp_dir(), "m.keras")
123  125            model.save(temp_model_filepath, save_format="keras_v3")
124  126            reloaded_model = saving_lib.load_model(
125       -            temp_model_filepath,
126       -            custom_objects={"TFSMLayer": tfsm_layer.TFSMLayer},
     127  +            temp_model_filepath, safe_mode=False
127  128            )
128  129            self.assertAllClose(reloaded_model(ref_input), ref_output, atol=1e-7)
129  130
     131  +    def test_safe_mode_blocks_model_loading(self):
     132  +        temp_filepath = os.path.join(self.get_temp_dir(), "exported_model")
     133  +
     134  +        # Create and export a model
     135  +        model = get_model()
     136  +        model(tf.random.normal((1, 10)))
     137  +        saved_model.export_saved_model(model, temp_filepath)
     138  +
     139  +        # Wrap SavedModel in TFSMLayer and save as .keras
     140  +        reloaded_layer = tfsm_layer.TFSMLayer(temp_filepath)
     141  +        wrapper_model = models.Sequential([reloaded_layer])
     142  +
     143  +        model_path = os.path.join(self.get_temp_dir(), "tfsm_model.keras")
     144  +        wrapper_model.save(model_path)
     145  +
     146  +        # Default safe_mode=True should block loading
     147  +        with self.assertRaisesRegex(
     148  +            ValueError,
     149  +            "arbitrary code execution",
     150  +        ):
```

```
151 +             saving_lib.load_model(model_path)
152 +
153 +         # Explicit opt-out should allow loading
154 +         loaded_model = saving_lib.load_model(model_path, safe_mode=False)
155 +
156 +         x = tf.random.normal((2, 10))
157 +         self.assertAllClose(loaded_model(x), wrapper_model(x))
158 +
130 159     def test_errors(self):
131 160         # Test missing call endpoint
132 161         temp_filepath = os.path.join(self.get_temp_dir(), "exported_model")
```

Comments 0

Please sign in to comment.