# Division by 0 in `QuantizedAdd`

Low **mihaimaruseac** published **GHSA-x83m-p7pv-ch8v** 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 cause a runtime division by zero error and denial of service in `tf.raw_ops.QuantizedAdd`:

```
import tensorflow as tf

x = tf.constant([68, 228], shape=[2, 1], dtype=tf.quint8)
y = tf.constant([], shape=[2, 0], dtype=tf.quint8)

min_x = tf.constant(10.723421015884028)
max_x = tf.constant(15.19578006631113)
min_y = tf.constant(-5.539003866682977)
max_y = tf.constant(42.18819949559947)

tf.raw_ops.QuantizedAdd(x=x, y=y, min_x=min_x, max_x=max_x, min_y=min_y, max_y=max_y)
```

This is because the implementation computes a modulo operation without validating that the divisor is not zero.

```
void VectorTensorAddition(const T* vector_data, float min_vector,
                          float max_vector, int64 vector_num_elements,
                          const T* tensor_data, float min_tensor,
                          float max_tensor, int64 tensor_num_elements,
                          float output_min, float output_max, Toutput* output) {
  for (int i = 0; i < tensor_num_elements; ++i) {
    const int64 vector_i = i % vector_num_elements;
    ...
  }
}
```

Since `vector_num_elements` is determined based on input shapes, a user can trigger scenarios where this quantity is 0.

### Patches

We have patched the issue in GitHub commit 744009c9e5cc5d0447f0dc39d055f917e1fd9e16.

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 Yakun Zhang and Ying Wang of Baidu X-Team.

**Severity**

Low

**CVE ID**

CVE-2021-29549

**Weaknesses**

No CWEs