Interpreter crash from `tf.io.decode_raw`

Critical **mihaimaruseac** published **GHSA-8pmx-p244-g88h** 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

The implementation of `tf.io.decode_raw` produces incorrect results and crashes the Python interpreter when combining `fixed_length` and wider datatypes.

```
import tensorflow as tf

tf.io.decode_raw(tf.constant(["1","2","3","4"]), tf.uint16, fixed_length=4)
```

The implementation of the padded version is buggy due to a confusion about pointer arithmetic rules.

First, the code computes the width of each output element by dividing the `fixed_length` value to the size of the type argument:

```
int width = fixed_length / sizeof(T);
```

The `fixed_length` argument is also used to determine the size needed for the output tensor:

```
TensorShape out_shape = input.shape();
out_shape.AddDim(width);
Tensor* output_tensor = nullptr;
OP_REQUIRES_OK(context, context->allocate_output("output", out_shape, &output_tensor));

auto out = output_tensor->flat_inner_dims<T>();
T* out_data = out.data();
memset(out_data, 0, fixed_length * flat_in.size());
```

This is followed by reencoding code:

```
for (int64 i = 0; i < flat_in.size(); ++i) {
  const T* in_data = reinterpret_cast<const T*>(flat_in(i).data());

  if (flat_in(i).size() > fixed_length) {
    memcpy(out_data, in_data, fixed_length);
  } else {
    memcpy(out_data, in_data, flat_in(i).size());
  }
  out_data += fixed_length;
}
```

The erroneous code is the last line above: it is moving the `out_data` pointer by `fixed_length * sizeof(T)` bytes whereas it only copied at most `fixed_length` bytes from the input. This results in parts of the input not being decoded into the output.

Furthermore, because the pointer advance is far wider than desired, this quickly leads to writing to outside the bounds of the backing data. This OOB write leads to interpreter crash in the reproducer mentioned here, but more severe attacks can be mounted too, given that this gadget allows writing to periodically placed locations in memory.

Patches

We have patched the issue in GitHub commit 698e01511f62a3c185754db78ebce0eee1f0184d.

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.

Severity

Critical

CVE ID

CVE-2021-29614

Weaknesses

No CWEs