happy-re-web-game

分析题目，填写邀请码正确才能开始游戏，合成2048方格可以得到flag。

于是查看js源码。

邀请码检查片段。

```
checkInvited() {
    let args = [...arguments];

    let buf = new ArrayBuffer(24);
    const view = new DataView(buf);
    view.setUint8(0, 18);
    view.setUint8(1, 101);
    view.setUint8(2, 83);
    view.setUint8(3, 84);
    view.setUint16(4, 0x036d, true);
    view.setUint16(6, 0x0604, true);
    view.setUint16(8, 0x6b67, true);
    view.setUint16(10, 0x3e09, true);
    view.setUint32(12, 0x2f261100, true);
    view.setUint32(16, 0x063e5e52);
    view.setUint32(20, 0x120000);

    function check(code) {
      if (code.length !== 24) return false;
      let encode = [];
      for (let i = 0; i < code.length; i++) {
        if (~i % 2 === 0) {
          encode.push(code.charCodeAt(i) ^ code.charCodeAt(i - 2));
        } else {
          encode.push(code.charCodeAt(i) ^ code.charCodeAt(i + 1));
        }
      }
      for (let i = 0; i < code.length; i++) {
        if (view.getInt8(i) !== encode[i]) return false;
      }
      return true;
    }

    return function () {
      if (!!arguments.length) {
        [].push.apply(args, arguments);
        return arguments.callee;
      }
      return check(args.join(""));
    };
  }
// this.checkInvited(input)() 调用
```

合成2048成功代码片段。

```
  gameSuccess() {
```

```
      this.mask.style = "display: flex;";
      this.maskBtn.innerHTML = "Play Again";
      this.gameStute = "Over";
      this.content.innerHTML = this.successReward();
    }

    successReward() {
      const code = this.inviteCode;
      let successContent = atob(
"Iw0LQn9bQxRBMB0TUBYVPgBbTXJUU05TFh4VVD0Db1xZLxgYLxMRPgtQCF1GRAxGFggHXDJfXVlVfhU="
      );
      let len = code.length;
      let result = "";

      for (let i = 0; i < successContent.length; i++) {
        result += String.fromCharCode(
          successContent.charCodeAt(i) ^ code.charCodeAt(i % len)
        );
      }
      return result;
    }
```

分析得，flag在那串base64中，但是经过了与我们输入的邀请码的逐位异或。

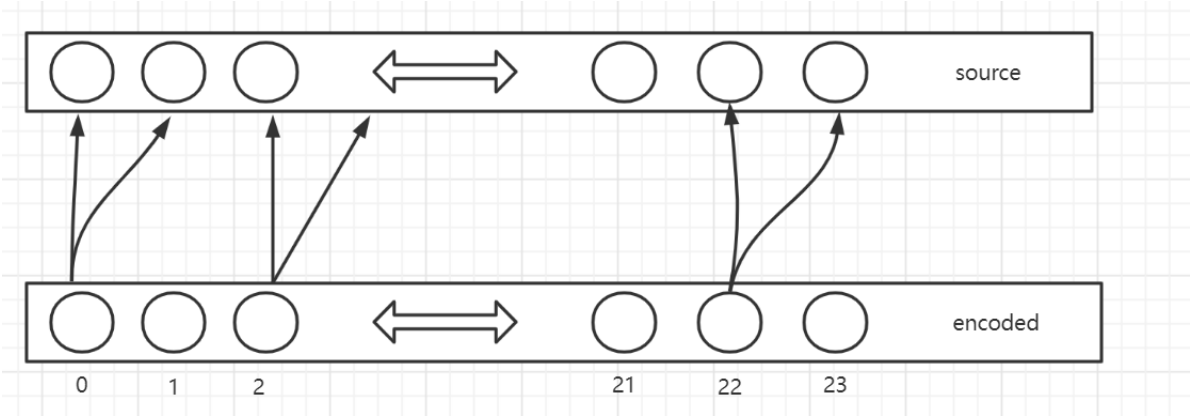
而我们是不知道邀请码的，所以必须逆前面的邀请码验证部分的代码。

0x01 邀请码

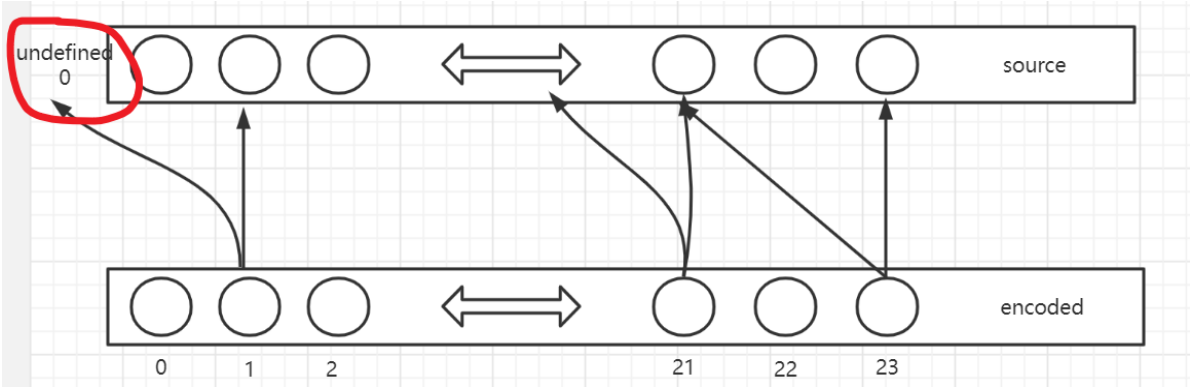
这一部分本着怎么麻烦怎么出的来着。简单整理：

```
1. ArrayBuffer 内部存储着加密后的邀请码
   view.setUint16(4, xxx, true); // 此处表示数据小端存储
   加密后：
   const code = [
      18, 101, 83, 84, 109, 3, 4, 6, 103, 107, 9, 62, 0, 17, 38, 47, 6, 62, 94,
82,
      0, 18, 0, 0
    ];
2. ~i % 2 === 0 意味着奇变偶，偶变奇
   ~1 => -2
   ~2 => -3
3. checkInvited() {  // 柯里化，这里其实只是用来看起来难看的，还是相对于直接调用了
checkInvited(input)
    let args = [...arguments];
    return function () {
      if (!!arguments.length) {
        [].push.apply(args, arguments);
        return arguments.callee;
      }
      return check(args.join(""));
    };
  }
4. 加密流程 ⇩
```

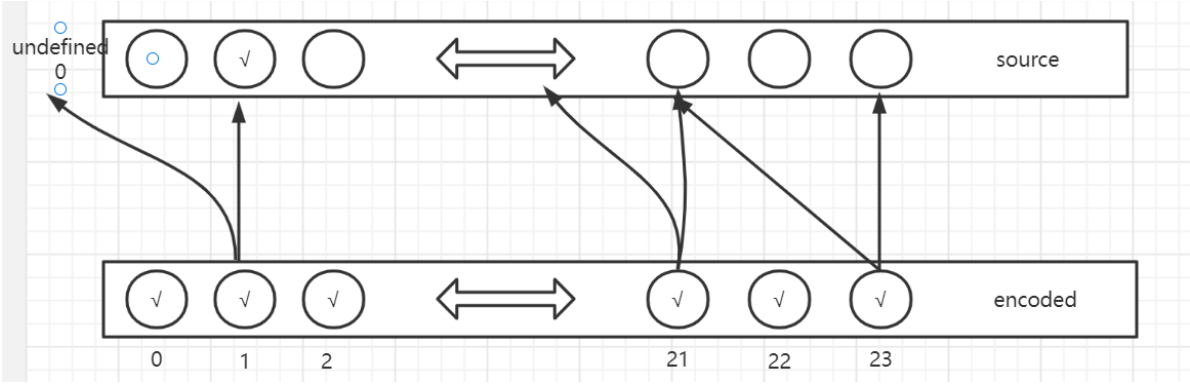
可以看到，偶数项加密无法找到突破口。



而由奇数序列，发现：encoded[1] = source[1] ^ source[-1]，js中数组越界为undefined，在异或操作中实则为0。



由此，找到突破口encoded[1] = source[1]。（√为已知）



而由奇数序列，encoded[3] = source[1] ^ source[3]，可得source[3]，依次类推可得所有奇数项。由上图偶数序列encoded[i] = source[i]^source[i+1]，可得所有偶数项source[i]。

解密脚本：

```
const code = [
  18, 101, 83, 84, 109, 3, 4, 6, 103, 107, 9, 62, 0, 17, 38, 47, 6, 62, 94, 82,
  0, 18, 0, 0
];
function decode() {
  let encode = [];
  for (let i = 1; i < code.length; i += 2) {
    encode.push(
      String.fromCharCode(code[i] ^ encode[(i - 3) / 2]?.charCodeAt(0))
    );
  }
  for (let i = 0; i < code.length; i += 2) {
    encode.splice(
      i,
```

```
        0,
        String.fromCharCode(code[i] ^ encode[i]?.charCodeAt(0))
      );
    }
    console.log(encode.join(""));
  }
  decode();
  // 邀请码：web1_2048_happy_gam3!!!!
```

0x02 flag

得到邀请码后，可以直接复制邀请码invitedCode到js源码处，控制台运行模拟游戏成功flag输出。

```
> function successReward() {
    const code = "web1_2048_happy_gam3!!!!";      改为邀请码
    let successContent = atob(
      "Iw0LQn9bQxRBMB0TUBYVPgBbTXJUU05TFh4VVD0Db1xZLxgYLxMRPgtQCF1GRAxGFggHXDJfXVlVfhU="
    );
    let len = code.length;
    let result = "";

    for (let i = 0; i < successContent.length; i++) {
      result += String.fromCharCode(
        successContent.charCodeAt(i) ^ code.charCodeAt(i % len)
      );
    }
    return result;
  }
< undefined
> successReward()
< 'This is your flag: Aurora{web1_happy_chal1enge-gamemmmmmm!}'
>
```

也可以自己把流程逆一遍。