Security

Malicious code on library

Un buon inizio da leggere

> If an attacker successfully injects any code at all, it's pretty much game over

XSS is too small scale, and really well protected against.

Chrome Extensions are too locked down.

Lucky for me, we live in an age where people install npm packages like they're popping pain killers.

I was excited at this point — I had a compelling package — but I didn't want to wait around while people slowly discovered it and spread the word. So I set about making PRs to existing packages that added my colourful package to their dependencies.

I've now made several hundred PRs (various user accounts, no, none of them as "David Gilbertson") to various frontend packages and their dependencies. "Hey, I've fixed issue x and also added some logging."

Look ma, I'm contributing to open source!

There are a *lot* of sensible people out there that tell me they don't want a new dependency, but that was to be expected, it's a numbers game.

https://medium.com/hackernoon/im-harvesting-credit-card-numbers-and-passwords-from-your-site-here-s-how-9a8cb347c5b5

Un caso famoso

Malicious code found in npm package event-stream downloaded 8 million times in the past 2.5 months

NOVEMBER 26, 2018 | IN VULNERABILITIES | BY DANNY GRANDER

A widely used npm package, `event-stream` , has been found to contain a malicious package named `flatmap-stream` . This was disclosed via a GitHub issue raised against the source repo.

https://snyk.io/blog/malicious-code-found-in-npm-package-event-stream/

`flatmap-stream` is a malicious package which was used in order to steal bitcoins from wallets. The malicious code was able to check if the `copay-dash` package was installed, and then attempt to steal the bitcoins stored in it. It was distributed by hijacking the popular `event-stream` package and adding `flatmap-stream` as a dependency.

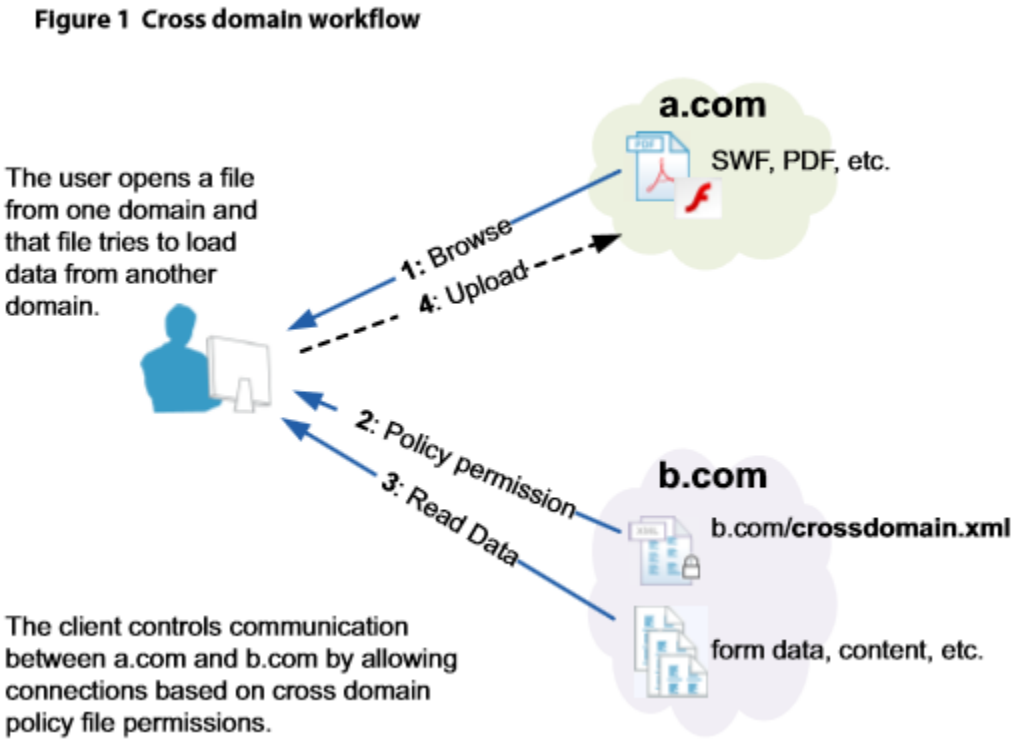
https://snyk.io/vuln/SNYK-JS-FLATMAPSTREAM-72637

Security

CORS

Cos'è:

CORS: Cross-Origin Resource Sharing



Cos’è:

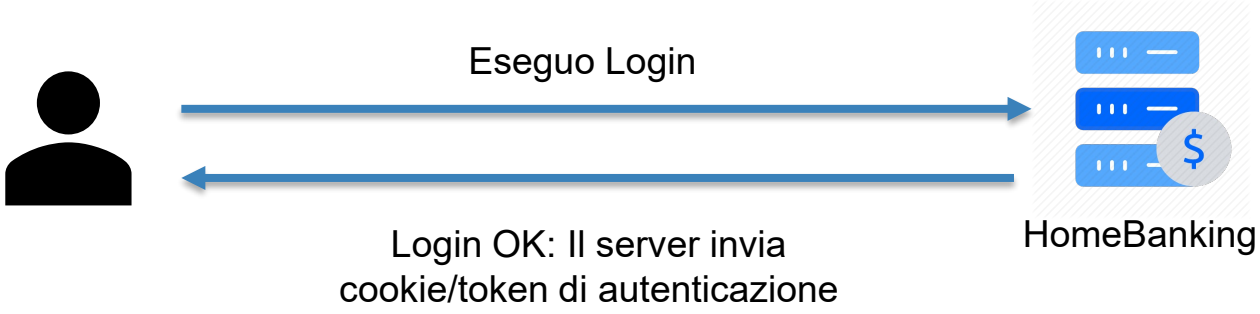
CORS: Cross-Origin Resource Sharing

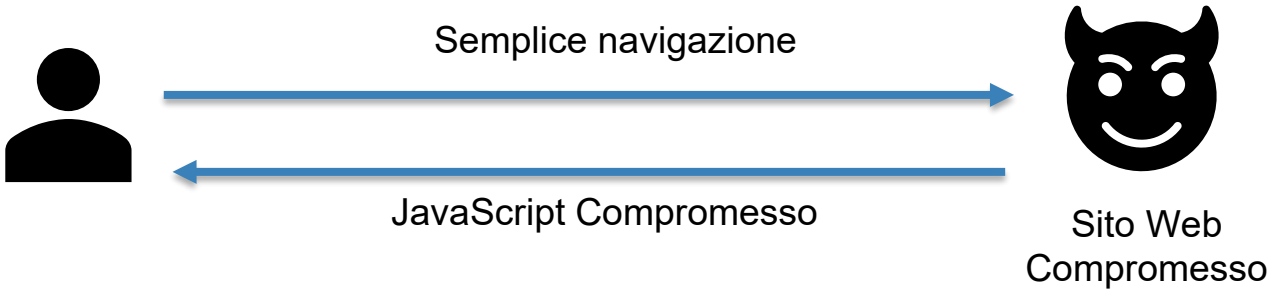
Il Cross-Origin Resource Sharing (CORS) è un meccanismo che usa header HTTP addizionali per indicare a un browser che un'applicazione Web in esecuzione su un'origine (dominio) dispone dell'autorizzazione per accedere alle risorse selezionate da un server di origine diversa. Un'applicazione web invia una **cross-origin HTTP request** quando richiede una risorsa che ha un'origine (protocollo, dominio e porta) differente dalla propria.

Esempio di cross-origin request: Il codice Javascript di frontend per un'applicazione web servita da `http://domain-a.com` utilizza `XMLHttpRequest` per inviare una richiesta a `http://api.domain-b.com/data.json`.

Importante: per permettere l’accesso su server di origine diversa si deve necessariamente **abilitare** e non disabilitare il CORS.

Perché è importante il CORS e cosa combatte?
Gli attacchi CSRF (Cross site request forgery)





Perché è importante il CORS e cosa combatte?
Gli attacchi CSRF (Cross site request forgery)