1. Данный пример – HTML-страница с JS, который перенаправляет пользователя на URL, указанный в параметре «r» (как пример, ?r=example.com)
2. В коде есть две части (HTML и JS), где:

Часть HTML: «if you do not redirect click [here]» и переносит на /home.html

Часть JS: filter заменяет некоторые символы. Скрипт берёт параметр r, если он задан, то пользователь перенаправляется на URL, обработанный фильтром.

1. Отсюда есть несколько уязвимостей:

* Недостаточная фильтрация функцией filter
* Open Redirect. Злоумышленник может использовать параметр r для перенаправления на произвольный URL
* Так же нет никакой проверки на достоверность домена в параметре r

**КАК ВОСПРОИЗВЕСТИ:**

1. Использовав параметр r, заключаем туда вредоносный URL, по которому перейдёт пользователь:

*http://127.0.0.1:8080/redir.html?r=http://spydomain.com*



1. При переходе по ссылке с параметром r, пользователь сразу перенаправляется на вредоносный URL

**Как защитить:**

1. Можно ограничить перенаправление только на доверенные домены:

*var trustedDomains = ['example.com', 'trusted-site.com'];*

*var urlParams = new URLSearchParams(window.location.search);*

*var path = urlParams.get('r') ?? '';*

*if (path !== '') {*

*try {*

*var url = new URL(path);*

*if (trustedDomains.includes(url.hostname)) {*

*location = path; // Разрешаем перенаправление*

*} else {*

*console.error('Redirect to untrusted domain blocked');*

*location = '/home.html'; // Перенаправляем на безопасную страницу*

*}*

*} catch (e) {*

*console.error('Invalid URL');*

*location = '/home.html'; // Перенаправляем на безопасную страницу*

*}*

*}*

1. Улучшить фильтрацию:

*function filter(s) {*

*return s.replace(/[^a-zA-Z0-9-\_]/g, "");*

*}*

1. Можно использовать белый список доверенных URL:

*const allowedUrls = ["/home.html", "/about.html", "/contact.html"];*

*if (!allowedUrls.includes(path)) {*

*console.error("Access denied");*

*return;*

*}*

*location = path;*