



Open in app

Get started



Published in Bits and Pieces



Piumi Liyana Gunawardhana

Follow

Sep 2, 2021 · 5 min read · Listen



Save



4 IFrame Security Concerns You Should Know

Facts to consider before using iframes in your web application

SAFETY CONCERNS IN IFRAMES



The iframe is one of the oldest and simple content embedding techniques used in web development, even used today. But, using them in practice comes with several security risks that could open doors for attackers.

So, in this article, I will discuss 5 security threats you need to be aware of before using iframes.





Open in app

Get started

iframe injection is a very common cross-site scripting attack.

iframes use multiple tags to display HTML documents on web pages and redirect users to different web addresses. This behavior allows 3rd parties to inject malicious executables, viruses, or worms into your application and execute them in user's devices.

We can find the iframe injections by scanning the HTML that your web server sends. All you need to do is open a page in your browser and then enable the “view source” feature to see the HTML. Since these iframes typically point to raw IP addresses, look for the `<iframe>` tags rather than domain names.

For example, let's take the below code:

```
[sourcecode]
++++%23wp+/+GPL%0A%3CScript+Language%3D%27Javascript%27%3E%0A++++%3C%21-
%0A++++document.write%28unescape%28%273c696672616d65207372633d276874
74703a2f2f696e666
f736563696e737469747574652e636f6d2f272077696474683d27312720686569676
8743d273127207374
796c653d277669736962696c6974793a2068696464656e3b273e3c2f696672616d65
3e%27%29%29%3B%0A
++++//-%3E%0A++++%3C/Script%3E
[/sourcecode]
```

It appears to be a common and relevant code for this site. However, it is the source of the issue. If you decode it using the JavaScript decoding function, the output will look like this:

```
[sourcecode]
#wp / GPL
<Script Language='Javascript'>
<!--
document.write(unescape(`3c696672616d65207372633d27687474703a2f2f696
```





Open in app

Get started

```
</Script>  
[/sourcecode]
```

Again, it appears to be legit because the attacker has used the terms “GPL” “wp” and the language type as “Javascript”. However, the digits and letters appear to be HEX. So next, we can use a hex decoder to decrypt it, and the final output will look like below.

```
<iframe src='https://www.infosecinstitute.com/' width='1' height='1'  
style='visibility: hidden;'></iframe>
```

So, suppose you find an iframe in your HTML and realize that it's something not put by you. In that case, it's important to investigate it and eliminate it from the website or database as soon as possible.

2. Cross- Frame Scripting

Cross-Frame Scripting (XFS) combines iframes with malicious JavaScript to steal data from users.

XFS attackers persuade a user to visit a web page regulated by the attacker and loads an iframe combined with malicious JavaScript referring to a legitimate site. The malicious JavaScript code keeps track of the user's keystrokes after inserting credentials into the legitimate site within the iframe.

XFS attacks can be prevented by including `Content-Security-Policy: frame-ancestors` and `x-Frame-Options` headers in web server configuration.

3. Clickjacking



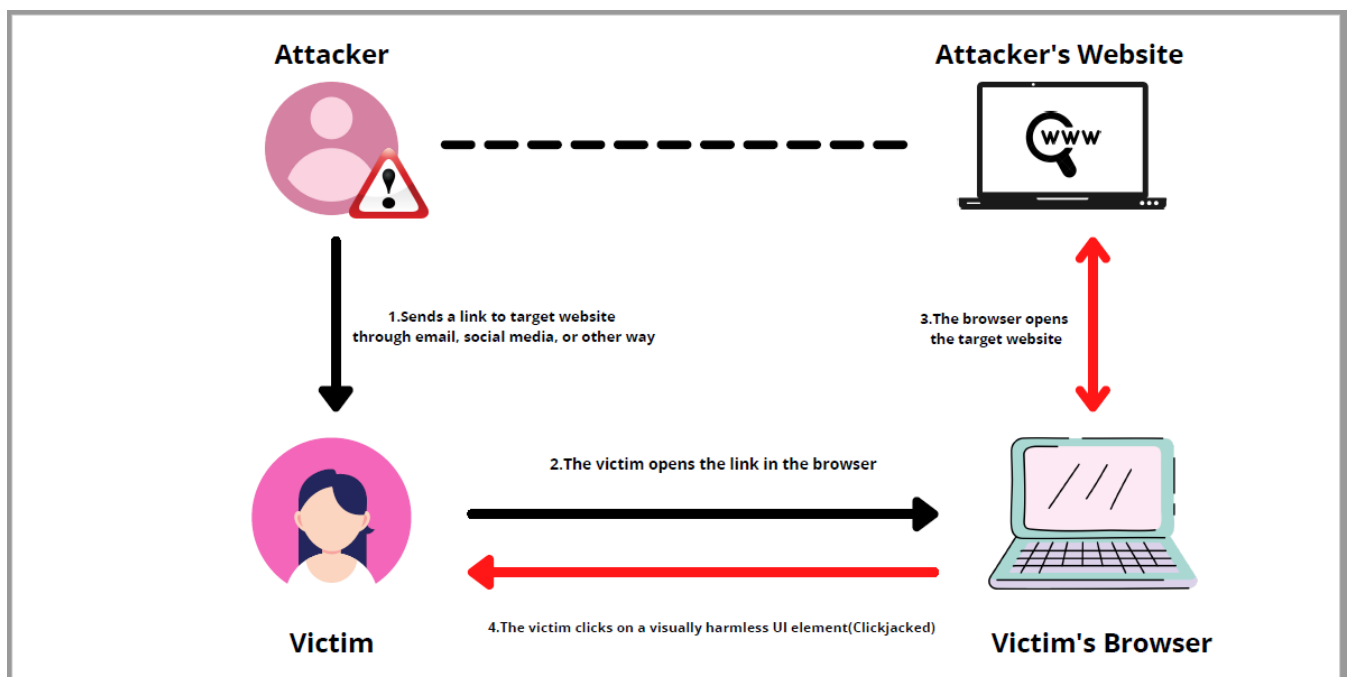


Open in app

Get started

Attackers typically perform clickjacking by placing an invisible page or HTML element inside an iframe on top of the user's page.

Users think that they click on the visible page, but they're clicking on a hidden element on the additional page overlaid on top of it.



Author's Work

There are two main strategies to protect yourself from clickjacking:

- Client-side methods include Frame Busting, which is the most prevalent. But Client-side methods are not the best solution because they are simply disregarded.
- X-Frame-Options is the most popular server-side method. Security experts strongly suggest server-side methods as a good way to prevent clickjacking.

4. Iframe Phishing

If you consider the social networking platforms, there are many users and developers who



[Open in app](#)[Get started](#)

Attackers often exploit this feature by using these incorporated iframes for phishing attacks.

By default, content from an iframe can trigger top-level navigation. So, an attacker might leverage cross-site scripting (XSS) vulnerability on a web application to insert phishing code as an iframe to lead the user into a phishing website.

Let's consider the below code for an example:

```
[sourcecode]
<html>
  <head>
    <title>Infosec Institute iFrame by Irfan</title>
  </head>
  <body>
    <iframe src="/user/piumi/" width="1450" height="300"
frameborder="0"></iframe>
    <iframe src="http://phishing.com/wp-login" width="1450"
height="250" frameborder="0"></iframe>
  </body>
</html>
[/sourcecode]
```

In the above code, there is a phishing site embedded using an iframe. The user will be redirected there, and if the user is not attentive to the address bar, the attacker will easily obtain the user's credentials.

iframe phishing attackers can't mimic the URL bar, but they can cause a redirect and then manipulate all of the content that users perceive after that.

This problem can be avoided by replacing `allow-top-navigation` from the `sandbox` attribute value.



[Open in app](#)[Get started](#)

iframes are a great option to keep your users more engaged. But, when you use an iframe, you're handling content from a third-party source that you have no control over. As a result, iframes often pose security threats to the applications.

However, we can't stop using iframe due to security threats. We need to be aware of them and take preventive actions to secure the applications.

So, I think this article would have helped you identify the safety concerns of using iframes, and let me know your thoughts in the comments section.

Thank you for Reading !!!


Build Great Design Systems and Micro Frontends

Take frontend development to the next level with independent components. Build and collaborate on component-driven apps to easily unlocks Micro Frontends, and to share components.

OSS Tools like [Bit](#) offer a great dev experience for building and composing independent components, and build solo or together with your team.

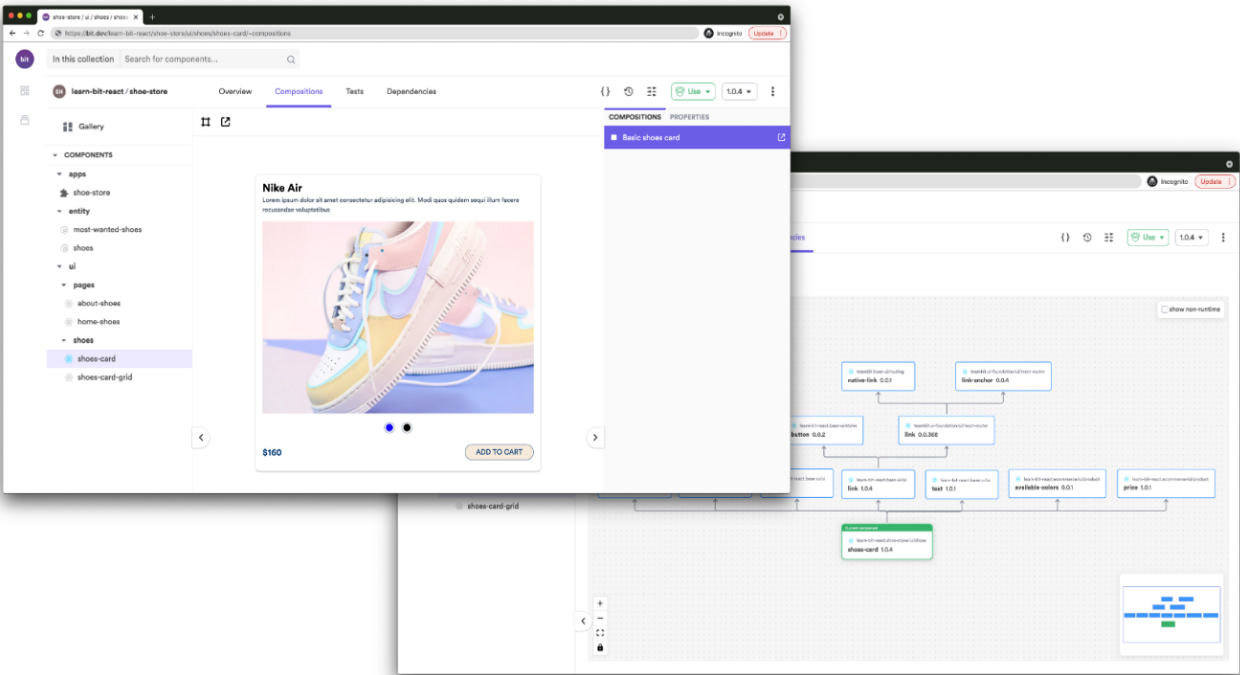
[Give it a try →](#)





Open in app

Get started



An independently source-controlled and shared “card” component. On the right => its dependency graph, auto-generated by Bit.

Learn More

Top Libraries for Iframes in React

Using Iframes with React is simple but needs support for advanced use cases. These Four libraries exactly do that.

blog.bitsrc.io

Best Practices in Using Iframes with React

Learn how to use Iframes with React following the best practices for Security and Performance

blog.bitsrc.io



Open in app

Get started

Iframe or script, what is the better option?

blog.bitsrc.io

Get an email whenever Piumi Liyana Gunawardhana publishes.

Subscribe

