# Clickjacking

An attack where a malicious website **tricks users into clicking** on something different from what they perceive — by **overlaying invisible iframes or UI elements** over legitimate content.  
**Example:**

* Attacker embeds a bank’s “transfer funds” button inside a transparent iframe on their site.
* User unknowingly performs a sensitive action (like transferring funds, liking posts, or confirming a transaction

## Impact

| Type | Description |
| --- | --- |
| Unauthorized Transactions | Trick users into initiating purchases, transfers |
| Change of Settings | Enable webcam, change privacy settings |
| Social Engineering + Spread | Automatically share malicious content |
| Session Exploitation | Works within active authenticated sessions (e.g., Gmail, banking, Facebook) |
| Brand Abuse | Third-party misuse of UI = trust loss |

## Types of Clickjacking

**1.Classic UI Redress**

* **Definition**: Invisible iframe over a fake button/image.
* **Example**: Victim thinks they're clicking "Watch Now", but it’s "Send Money".
* **Mitigation**: Use framebusting headers (X-Frame-Options, CSP).

**2. Likejacking**

* **Definition**: Trick user into "liking" content on social platforms.
* **Example**: Facebook Like button hidden under “Click Here to Win!”.
* **Mitigation**: Frame protection on social widgets, UI validation.

**3. Cursorjacking**

* **Definition**: Altering cursor's visual position using CSS to misalign UI and pointer.
* **Example**: Visual cursor is above one button, but real click target is elsewhere.
* **Mitigation**: Disable complex cursor effects, careful UI placement.

**4. File Upload Clickjacking**

* **Definition**: Trick user into uploading a file or granting camera/microphone access.
* **Example**: Hidden iframe under an "Upload Resume" button.
* **Mitigation**: Do not allow sensitive file or permission dialogs in iframes.

**Mitigation:**

| Strategy | Description |
| --- | --- |
| X-Frame-Options header | Prevents your site from being loaded in a frame/iframe |
| Content-Security-Policy: frame-ancestors | CSP header specifying allowed framing domains |
| UI Design Techniques | Verify user intent before sensitive actions (CAPTCHA, confirm dialogs) |
| Frame Busting Scripts (not reliable) | JS to break out of iframe — outdated method |
| SameSite Cookies | Prevent session misuse across origins (partial protection) |
| MFA | Reduces damage even if tricked into clicking |

* Use X-Frame-Options: DENY or SAMEORIGIN headers.
* Implement Content Security Policy (CSP) frame-ancestors directive.

X-Frame-Options: DENY  
X-Frame-Options: SAMEORIGIN  
Content-Security-Policy: frame-ancestors 'none';  
**Best Practice**: Prefer CSP frame-ancestors over X-Frame-Options, as CSP is more flexible and modern.

# Tools to Detect Clickjacking

| Tool | Use |
| --- | --- |
| Burp Suite (Manual) | Inspect headers & test frame embedding |
| Clickjacking PoC Generator | Generate PoC iframe page |
| XFO Test Tools (e.g., securityheaders.com) | Scan response headers |
| OWASP Clickjacking Test Page | Test if a site is frameable |
| Browser DevTools | Manually inject iframe, observe behavior |

# Points

Clickjacking is a **UI deception attack**, not a technical vulnerability per se — and it succeeds because **users trust the browser UI**.”

The **most effective mitigation is to use server-side headers like X-Frame-Options and CSP** to prevent framing.”

Test for clickjacking in every engagement by attempting to embed the target in an iframe and observing behavior.”

# Real-World Breaches

| Site | Vulnerability |
| --- | --- |
| Facebook (pre-2011) | Likejacking campaigns |
| Twitter (2010) | Tweetjacking via transparent layers |
| Bank of America (PoC) | Hidden fund transfer button in iframe |

# Testing Process

<iframe src="https://target.com/sensitive-action" width="800" height="600" style="opacity:0.01; position:absolute; top:0; left:0;"></iframe>

Overlay a fake button that matches the real button's position.  
Click and observe if the action is triggered.