Clickjacking Protection Under Non-trivial Circumstances

Sebastian Lekies, Martin Johns

OWASP
The Open Web Application Security Project

SAP

WebSand

Agenda

OWASP
The Open Web Application Security Project

- **What is Clickjacking?**
- **Current Defenses**
- **Open Issues and Limitations**
- **Protection Approach**

SAP

WebSand

Agenda

OWASP
The Open Web Application Security Project

- **What is Clickjacking?**
- Current Defenses
- Open Issues and Limitations
- Protection Approach

SAP

WebSand

Technical Background
What is Clickjacking?

OWASP
The Open Web Application Security Project

> *Clickjacking* (also called *UI redressing*) is an attack that lures an unsuspicious user into clicking on an element that is different to what the user perceives to click on.

SAP

WebSand

Technical Background
What is Clickjacking?

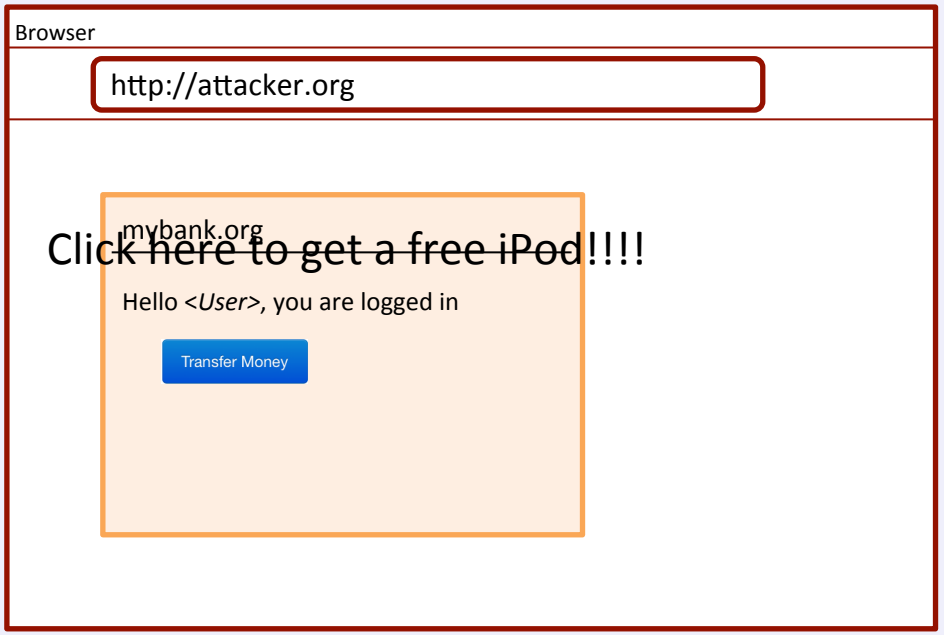
OWASP
The Open Web Application Security Project

Controllable Container

- Frames
- Object, Embed
- Popup windows

Disguising the UI

- Covering it with other elements
- Reducing its size
- Displaying it only for a very short amount of time
- Making it transparent



mybank.org

User

SAP

WebSand

Agenda

OWASP
The Open Web Application Security Project

- What is Clickjacking?
- **Current Defenses**
- Open Issues and Limitations
- Protection Approach

Current Defense Techniques

Overview

OWASP
The Open Web Application Security Project

Client-side approaches:

- Protect a single user against attacks on all web applications
  - E.g.: NoScript Clearclick
  - Alternative browser designs: Gazelle, OP, Secure Web Browser

Server-side approaches:

- Protect all users of single web application against attacks
- Deployed on the server-side, but enforcement happens on the client-side
  - Frame Busting
  - X-Frame-Options Header

Current Defense Techniques

Server-Side

OWASP
The Open Web Application Security Project

Frame Busting

- Basic Idea: Avoid unauthorized framing of a web page
- Implementation: A small snippet of JavaScript code checks if page is framed. If so, it navigates the top frame towards the framed page

```
<script>
  if (parent!= self)
    parent.location = self.location;
</script>
```

Several ways exist to circumvent this protection:

- Prevent JavaScript execution
  - Misusing modern XSS filters
  - Using sandboxed iframes
- Prevent redirect
  - 204 flushing
  - Double framing
  - By asking the user nicely (onbeforeunload event)

It is possible to build secure frame busters. However, the knowledge about it is not widely spread

OWASP
The Open Web Application Security Project

X-Frame-Options Header

- Approach introduced by Microsoft to counter Clickjacking attacks
- Idea is similar to frame busting: Avoid unauthorized framing of a page
- Implementation:
  - Non-JavaScript solution
  - Based on an HTTP Response header
  - Browser enforces the Web server's desired behavior

The X-Frame-Options header takes one of two values:

- SAMEORIGIN: Only same-origin pages are allowed to frame the marked page
- DENY: Framing is forbidden
- (IE only: FROMORIGIN: Allows one specific origin to frame the marked page)

Agenda

OWASP
The Open Web Application Security Project

- What is Clickjacking?
- Current Defenses
- **Open Issues and Limitations**
- Protection Approach

SAP

WebSand

Open Issues

OWASP
The Open Web Application Security Project

Clickjacking is not limited to frames

- Double Clickjacking
- Clickjacking via History navigation

Open Issues

OWASP
The Open Web Application Security Project

Double Clickjacking

Technique developed by Huang and Jackson

- Based on popups instead of frames

- **Procedure**

1. Open target page as a popup window behind the current browser window
   - The opening page receives a window handle = navigational control
2. Lure the victim into double clicking on the current browser window
   - First click: hits current window + triggers focus to the popup
   - Second click: hits the popup + triggers state changing action
3. Close the popup shortly after the first click

E.g. Google OAuth authentication popup was attacked with this method

Open Issues

OWASP
The Open Web Application Security Project

Clickjacking via History Navigation (Caching)

Technique developed by Michal Zalewski

- Also based on popups instead of frames

- **Procedure**

1. Open target page as a popup window
   - Page is cached by the browser
2. Immediately navigate the popup to an attacker controlled site
3. Lure user into clicking on the page
4. Shortly before the click: call history.back()
   - Target page is loaded from cache: It is immediately rendered
   - Click hits the target page: Navigate away or close the window afterwards

Open Issues
Vulnerabilities

OWASP
The Open Web Application Security Project

Removing HTTP response headers in Safari

- Vulnerability within the HTML5 Offline Application Cache
- Offline App Cache allows to store HTML documents for offline usage
- Websites are always retrieved from Cache first (even if a connection is available)
- The App Cache does not store HTTP response headers
  - Hence web pages stored/read from cache cannot be protected via X-Frame-Options

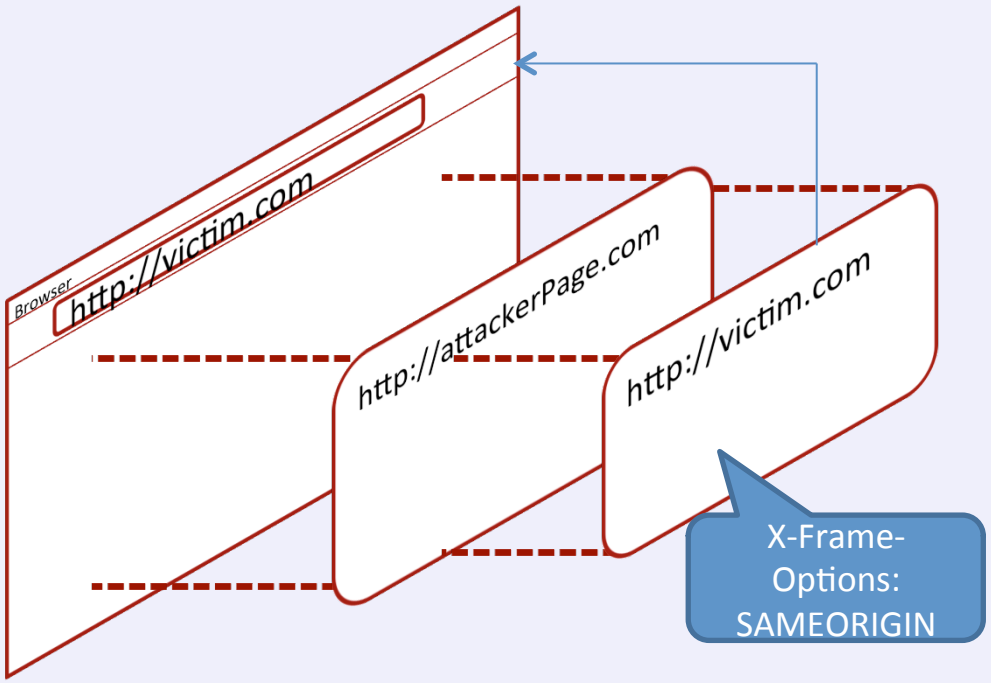
Risk Assessment

- HTML5 Offline Applications are often used in mobile environments
  - Mobile version of GMAIL
  - Mobile version of Hotmail
- Given Apple's (and therewith Safari's) market leadership this is a serious vulnerability for iOS users

Open Issues
Vulnerabilities

OWASP
The Open Web Application Security Project

Nested Clickjacking



SAP
WebSand

Open Issues
Vulnerabilities

OWASP
The Open Web Application Security Project

Nested Clickjacking

SAP
WebSand

Open Issues

OWASP
The Open Web Application Security Project

The current defense mechanisms have one thing in common

- Prevent unauthorized framing to protect against Clickjacking
- Authorized framing = same-domain framing
- Unauthorized framing = cross-domain framing

How to protect against Clickjacking if cross-domain framing is required?

- E.g. in Corporate-Portal environments
- E.g. Ad providers
- E.g. Web widgets
- No possibility to protect against Clickjacking in this case
  - At least not until allow-from is widely supported

SAP

WebSand

Agenda

OWASP
The Open Web Application Security Project

- What is Clickjacking?
- Current Defenses
- Open Issues and Limitations
- **Protection Approach**

SAP

WebSand

Protection Approach

OWASP
The Open Web Application Security Project

Design Goals

1. Based on standard browser features
   - To ensure immediate availability and cross browser compatibility
2. Protection within different environments
   - Frames
   - Popups
   - Protection against the described vulnerabilities
3. Configurability
   - Allow a whitelisting approach
   - With a potentially unlimited number of entries

Protection Approach

OWASP
The Open Web Application Security Project

High-level Overview

- Idea: Whenever a page tries to render(frame/popup) a page outfitted with our countermeasure, force the embedding page to prove it's authenticity to the embedded page. Rendering of the embedded webpage is postponed until a prove is given.
- Challenges:

1. Prevent rendering of the page

```
<style>
  body { display:none; }
</style>
```

2. Detect untrusted view-port constellations (framing/popup)

```
parent !== self || opener !== null
```

3. Prove the authenticity of the embedding page

```
function handlePostMessage(event){
  if(event.origin === 'http://example.net'){
    bodyElement.style.display='block ';
  }
}
```

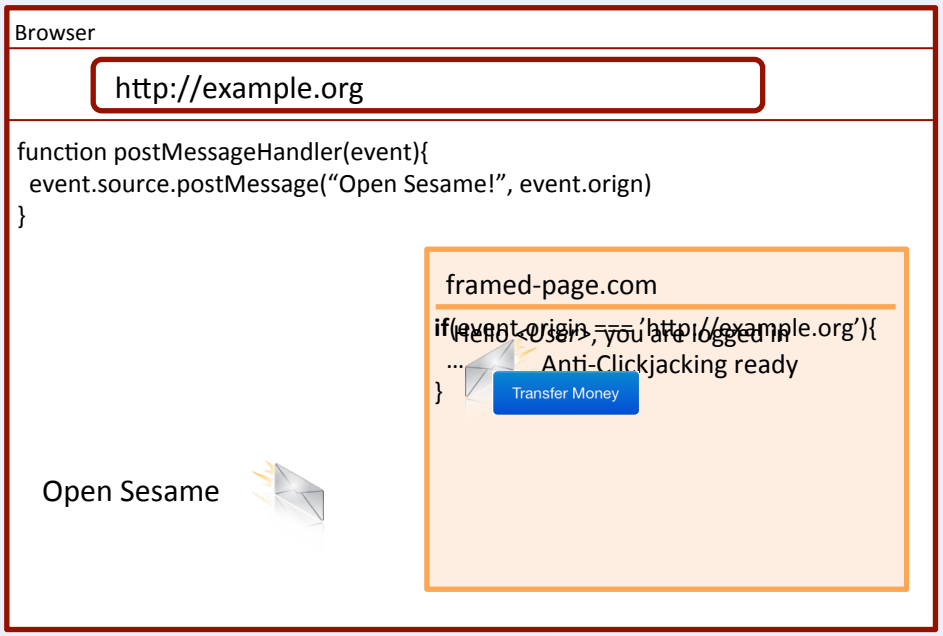
4. Inform the framing about the registration of the PostMessage handler within the frame

```
parent.postMessage("Anti-Clickjacking ready", "*")
```

Protection Approach

Putting the pieces together

OWASP
The Open Web Application Security Project



Embeding Page

- Register a PostMessage handler
- Insert the iframe element
- Reply PostMessage

Framed Page

- Load & send PostMessage
- Receive reply
- Check origin of reply
- If origin is appropriate, reveal the content

Conclusion

OWASP
The Open Web Application Security Project

Clickjacking

- Attack that hijacks clicks from an unsuspicious user

Protection Approaches

- Client-side: Clearclick or alternative browser designs
- Server-side: X-Frame-Options, Frame busting, CSP
- All techniques focus on preventing framing
- Techniques are not applicable in many cases
- Vulnerabilities: Nested Clickjacking, Clearclick circumvention

New Protection Approach

- Based on standard features: JavaScript (PostMessage API) & CSS
- Protects against all the outlined cases
- Flexible and configurable

OWASP
The Open Web Application Security Project

Thank you

Contact Information:
Sebastian Lekies
SAP Security Research
sebastian.lekies@sap.com
Twitter: @sebastianlekies

Martin Johns
SAP Security Research
martin.johns@sap.com
@datenkeller