

CSE 484 / CSE M 584  
Computer Security:  
Clickjacking

Jared Moore

[jlcmoore@cs.washington.edu](mailto:jlcmoore@cs.washington.edu)

Thanks to **Franzi Roesner**, Dan Boneh, Dieter Gollmann, Dan Halperin, Yoshi Kohno, John Manferdelli, John Mitchell, **Vitaly Shmatikov**, Bennet Yee, and many others for sample slides and materials ...

# Lab 2

- Questions
  - What is the flow of the XSS attacks?
  - How kind of information do you need leaked for SQL injection attacks?
  - Others?

# Directing User Behavior

- Demo
  - <http://lcamtuf.coredump.cx/ffgeo2/>

The following slides originally made by Vitaly  
Shmatikov

[hNp://www.cs.utexas.edu/~shmat/courses/  
cs361s/clickjack.ppt](http://www.cs.utexas.edu/~shmat/courses/cs361s/clickjack.ppt)

# Clickjacking (UI Redressing)

[Hansen and Grossman 2008]

- Attacker overlays multiple transparent or opaque frames to trick a user into clicking on a button or link on another page



- Clicks meant for the visible page are hijacked and routed to another, invisible page

# Clickjacking in the Wild

- Google search for “clickjacking” returns 624,000 results... this is not a hypothetical threat!
- Summer 2010: Facebook worm superimposes an invisible iframe over the entire page that links back to the victim's Facebook page
  - If victim is logged in, automatically recommends link to new friends as soon as the page is clicked on
- Many clickjacking attacks against Twitter
  - Users send out tweets against their will

# It's All About iFrame

- Any site can frame any other site

`<iframe`

`src="http://www.google.com/...">`

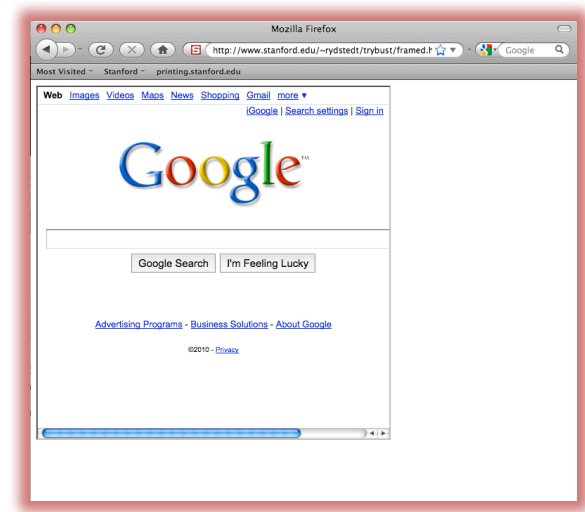
`</iframe>`

- HTML attributes

– Style

– **Opacity** defines visibility percentage of the iframe

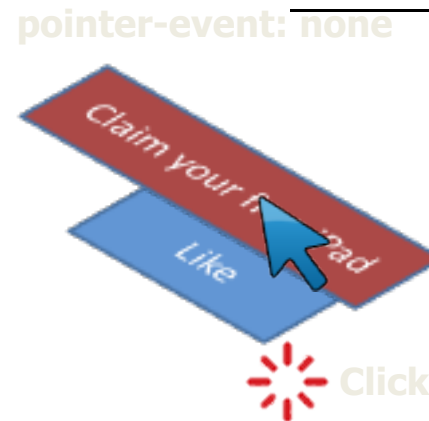
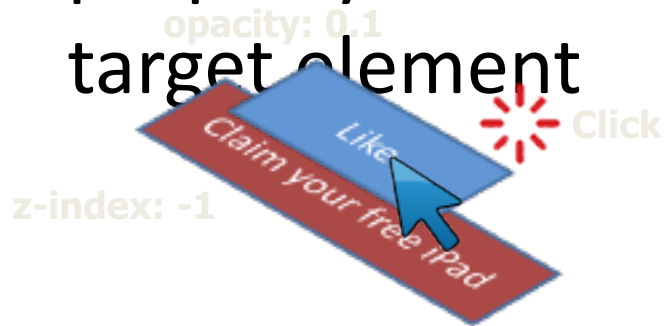
- 1.0: completely visible
- 0.0: completely invisible



# Hiding the Target Element

[“Clickjacking: Attacks and Defenses”]

- Use CSS `opacity` property and `z-index` property to hide target element and make other element float under the target element
- Using CSS `pointer-events: none` property to cover other element over the target element



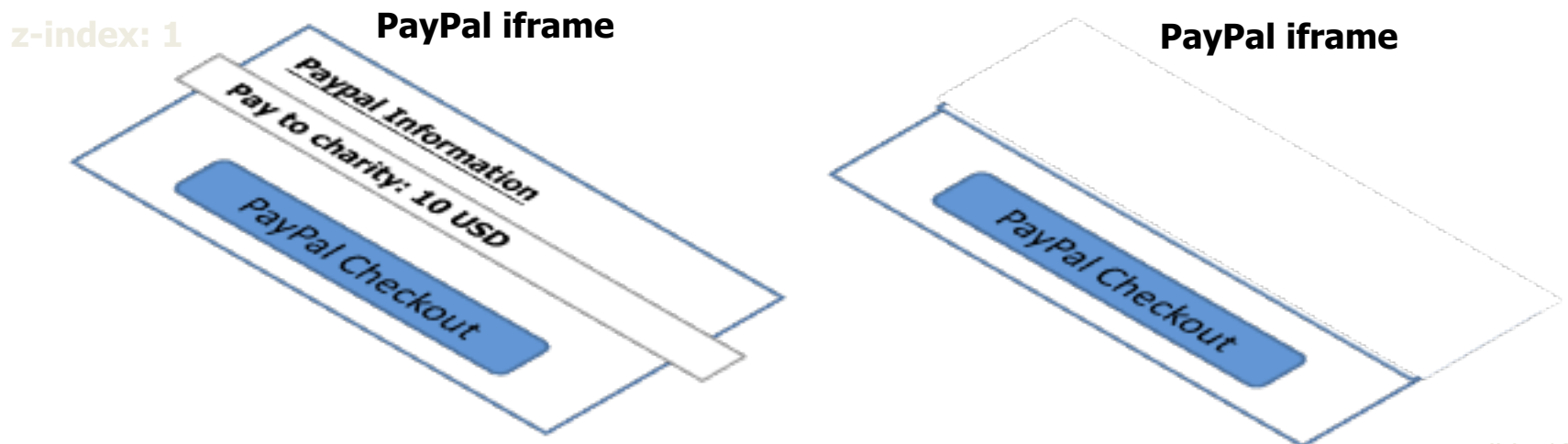


What other types of user-manipulative attacks might be possible using properties of css?

# Partial Overlays and Cropping

[“Clickjacking: Attacks and Defenses”]

- Overlay other elements onto an iframe using CSS `z-index` property or Flash Window Mode `wmode=direct` property
- Wrap target element in a new iframe and choose CSS position offset properties



# Drag-and-Drop API

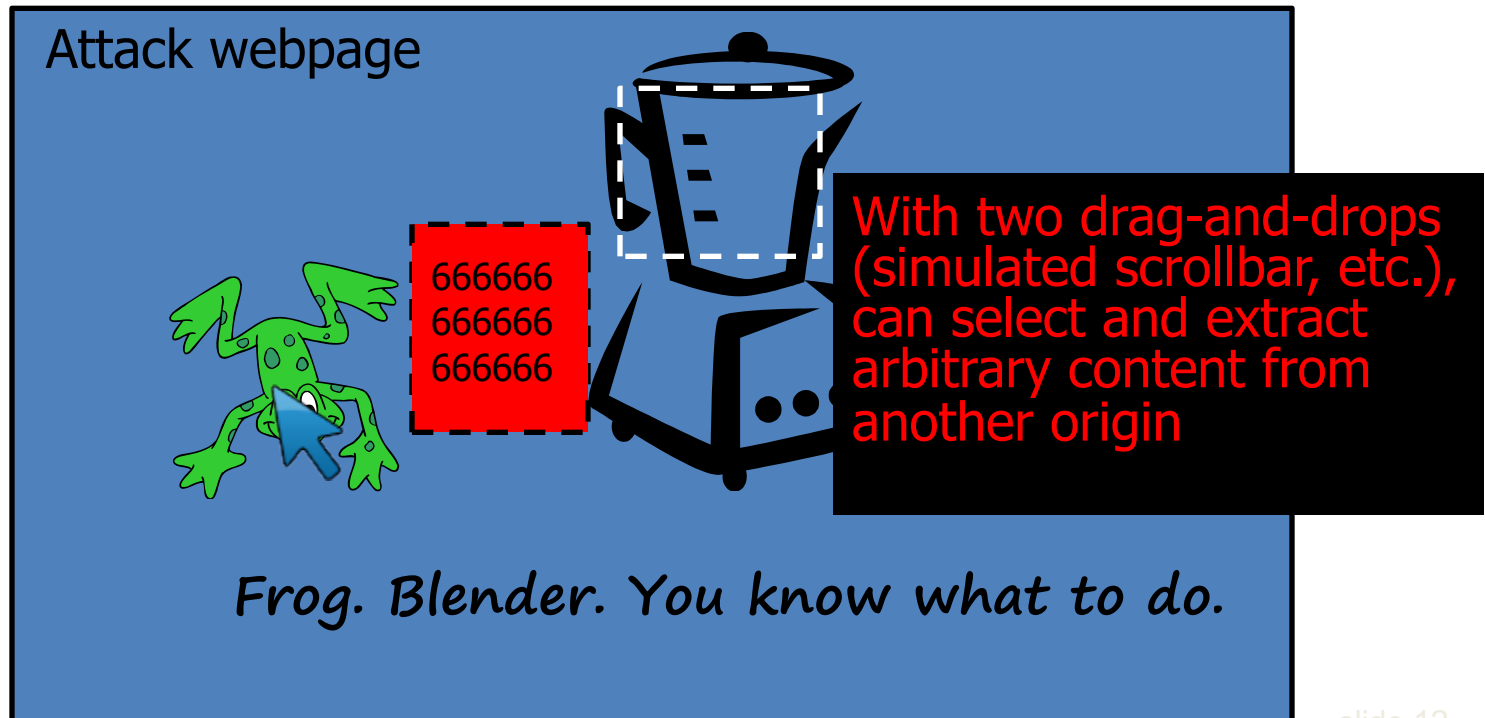
[“Next Generation Clickjacking”]

- Modern browsers support drag-and-drop API
- JavaScript can use it to set data being dragged and read it when it's dropped
- Not restricted by the same origin policy: data from one origin can be dragged to a frame of another origin
  - Reason: drag-and-drop can only be initiated by user's mouse gesture, not by JavaScript on its own

# Abusing Drag-and-Drop API

[“Next Generation Clickjacking”]

1. Bait the user to click and start dragging
2. Invisible iframe with attacker’s text field under mouse cursor, use API to set data being dragged
3. Invisible iframe from another origin with a form field



# Fake Cursors

[“Clickjacking: Attacks and Defenses”]

- Use CSS `cursor` property and JavaScript to simulate a fake cursor icon on the screen

Real cursor icon

`cursor: none`

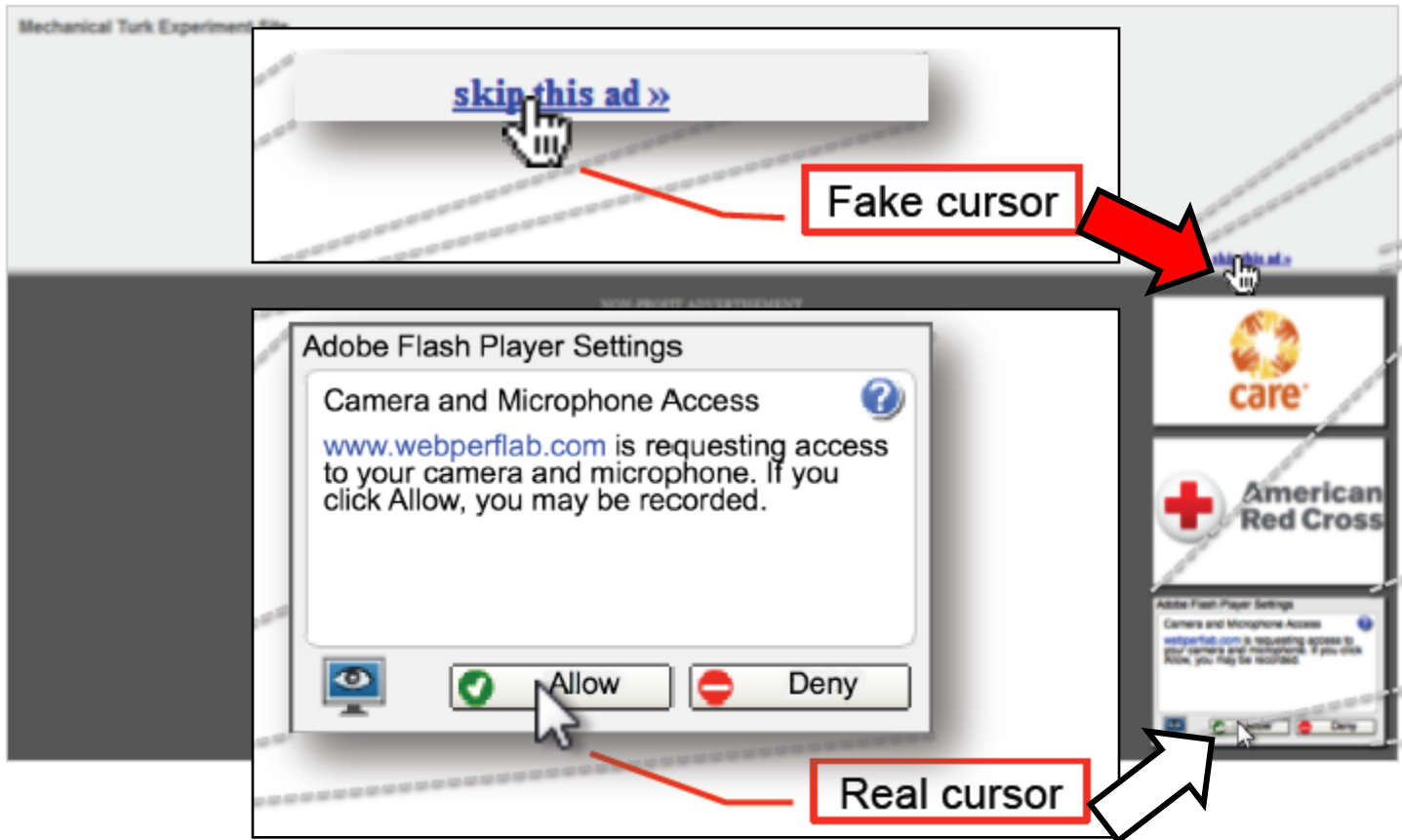


Fake cursor icon



# Cursor Spoofing

[“Clickjacking: Attacks and Defenses”]



# Keyboard “Strokejacking”

[“Clickjacking: Attacks and Defenses”]

- Simulate an input field getting focus, but actually the keyboard focus is on target element, forcing user to type some unwanted information into target element

**Attacker’s page**

Typing Game  
Type whatever screen shows to you


Xfpog95403poigr06=2kfpX

[  ]



**Hidden iframe within attacker’s page**

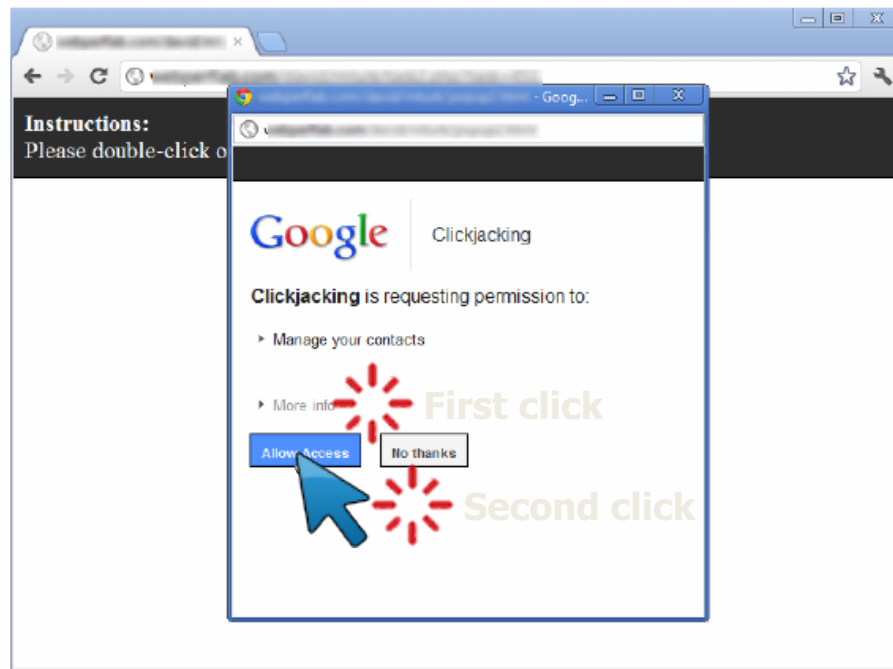
Bank Transfer  
Bank Account: 9540  
Amount: 3062 USD



# Double-Click Attack

[“Clickjacking: Attacks and Defenses”]

- Bait the user to perform a double-click, switch focus to a popup window under the cursor right between the two clicks





# Whack-A-Mole Attack

[“Clickjacking: Attacks and Defenses”]

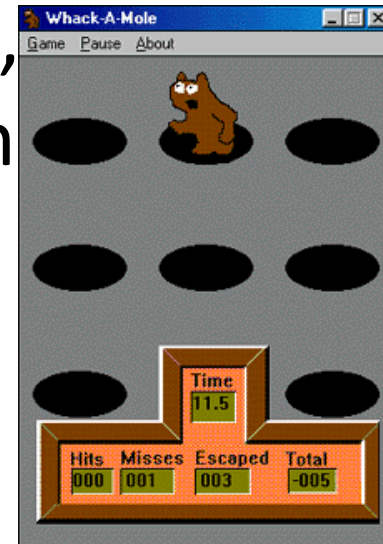
- Ask the user to click as fast as possible, suddenly switch Facebook Like button

**Instructions:**  
Please click on blue buttons *as fast as possible*. The faster you complete this game, the greater your chances to win a \$100 prize! If you don't click on a button, the game will skip it in 10 seconds.

Buttons clicked: 17/20  
Time elapsed: 27.6 sec

[CLICK ME](#)

[Like](#) 1



**SOLUTIONS?**

# Solution: Frame Busting

- I am a page owner
- All I need to do is make sure that my web page is not loaded in an enclosing frame ...

Clickjacking: solved!

- Does not work for FB “Like” buttons and such, but  
Ok

- How 

```
if (top != self)
  top.location.href = location.href
```

# Frame Busting in the Wild

- ◆ Survey by Gustav Rydstedt, Elie Burzstein, Dan Boneh, Collin Jackson



Following slides shamelessly jacked from Rydstedt

# If My Frame Is Not On Top ...

## Conditional Statements

```
if (top != self)
```

```
if (top.location != self.location)
```

```
if (top.location != location)
```

```
if (parent.frames.length > 0)
```

```
if (window != top)
```

```
if (window.top !== window.self)
```

```
if (window.self != window.top)
```

```
if (parent && parent != window)
```

```
if (parent &&  
    parent.frames &&  
    parent.frames.length>0)
```

```
if((self.parent&&  
    !(self.parent===self))&&  
    (self.parent.frames.length!=
```

# ... Move It To Top

## Counter-Action Statements

```
top.location = self.location
```

```
top.location.href = document.location.href
```

```
top.location.href = self.location.href
```

```
top.location.replace(self.location)
```

```
top.location.href = window.location.href
```

```
top.location.replace(document.location)
```

```
top.location.href = window.location.href
```

```
top.location.href = "URL"
```

```
document.write('')
```

```
top.location = location
```

```
top.location.replace(document.location)
```

```
top.location.replace('URL')
```

```
top.location.href = document.location
```

```
top.location.replace(window.location.href)
```

```
top.location.href = location.href
```

```
self.parent.location = document.location
```

```
parent.location.href = self.document.location
```

```
top.location.href = self.location
```

```
top.location = window.location
```

```
top.location.replace(window.location.pathname)
```

# What About My Own iFrames?

- Check: **is the enclosing frame one of my own?**
- How hard can this be?
- Survey of several hundred top websites ...  
... **all** frame busting code is broken!

# Courtesy of Walmart

```
if (top.location != location) {  
  if(document.referrer &&  
    document.referrer.indexOf("walmart.com") == -1)  
  {  
    top.location.replace(document.location.href);  
  }  
}
```





# Error in Referrer Checking



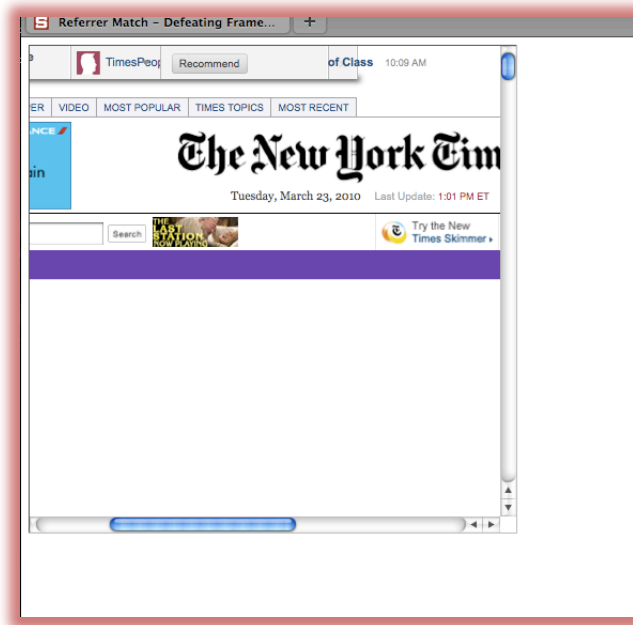
From <http://www.attacker.com/walmart.com.html>

```
<iframe src="http://www.walmart.com">
```

# The New York Times

```
if (window.self !== window.top &&  
    !document.referrer.match(  
    /https?:\/\/[^\?\/]+\\.nytimes\.com\/\//))  
{  
    self.location = top.location;  
}
```

# Error in Referrer Checking

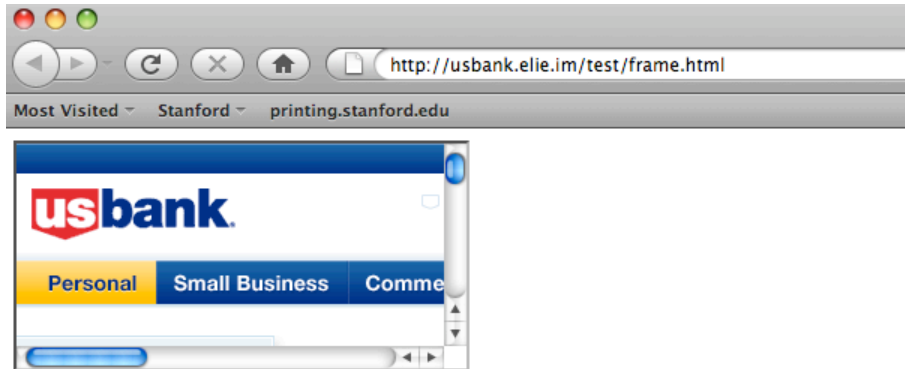


From <http://www.attacker.com/a.html?b=https://www.nytimes.com/>  
<iframe src="http://www.nytimes.com">



```
if (self != top) {  
    var domain = getDomain(document.referrer);  
    var okDomains = /usbank|localhost|usbnet/;  
    var matchDomain = domain.search(okDomains);  
  
    if (matchDomain == -1) {  
        // frame bust  
    }  
}
```

# Error in Referer Checking



From <http://usbank.attacker.com/>  
<iframe src="http://www.usbank.com">

# Strategic Relationship?

Norwegian State House Bank

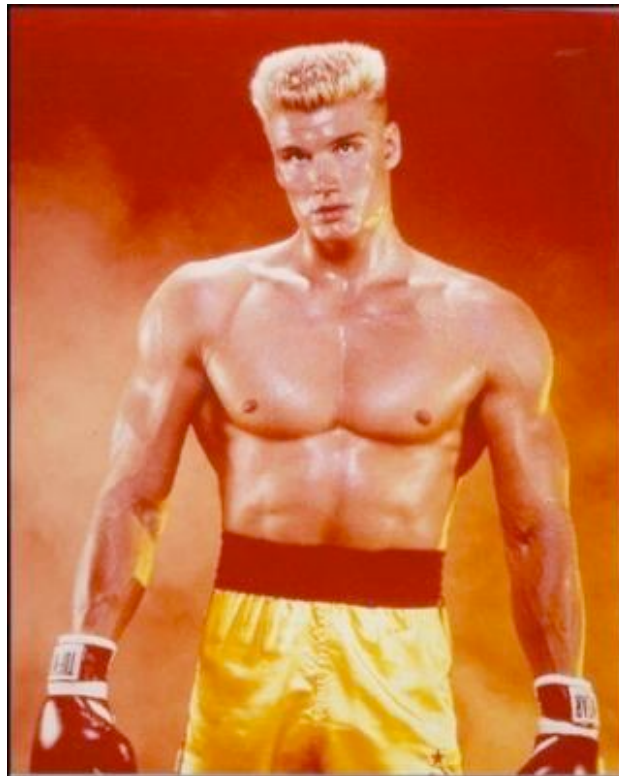
<http://www.husbanken.no>



# Strategic Relationship?

Bank of Moscow

<http://www.rusbank.org>



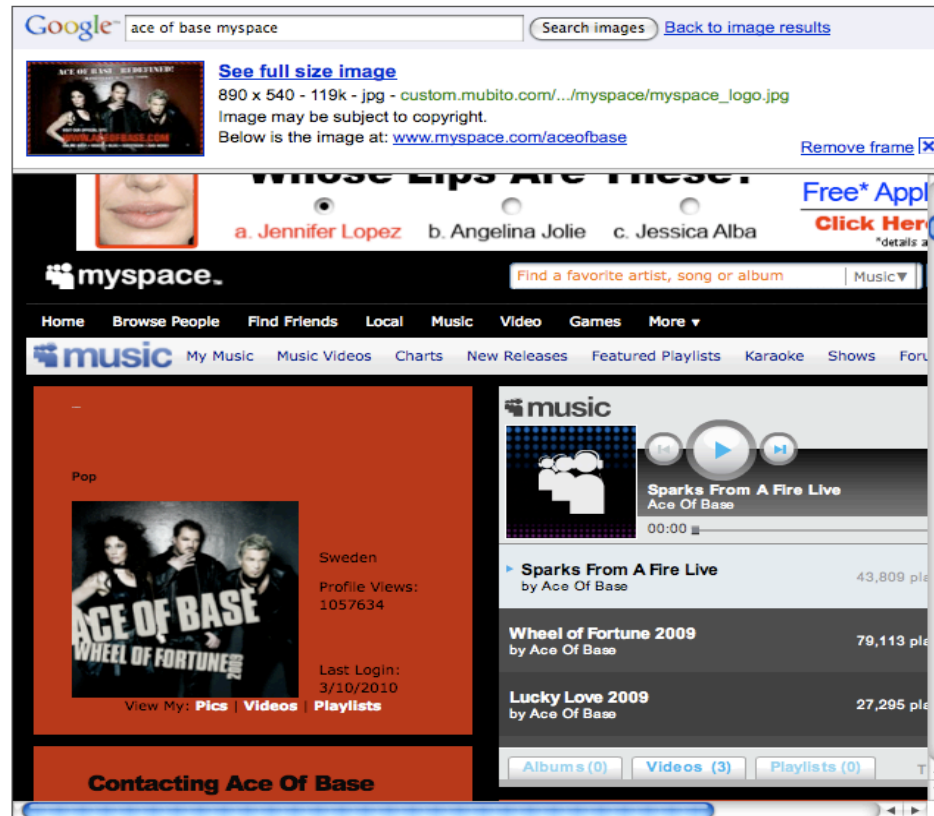


```
try{
  A=!top.location.href
} catch(B){}
A=A&&
!(document.referrer.match(/^https?:\\V\\[-az09.]
*.google\\.(co\\. | com\\. )? [a-z] +\\imgres/i))&&
!(document.referrer.match(/^https?:\\V\\([\\^\\V]*\\. )?
(myspace\\.com |
myspace\\.cn |
simsidekick\\.com |
levisawards\\.com |
digg\\.com)\\//i));

if(A){ // Frame bust }
```



# Do Your Trusted Sites Frame Bust?



Google Images does not frame bust

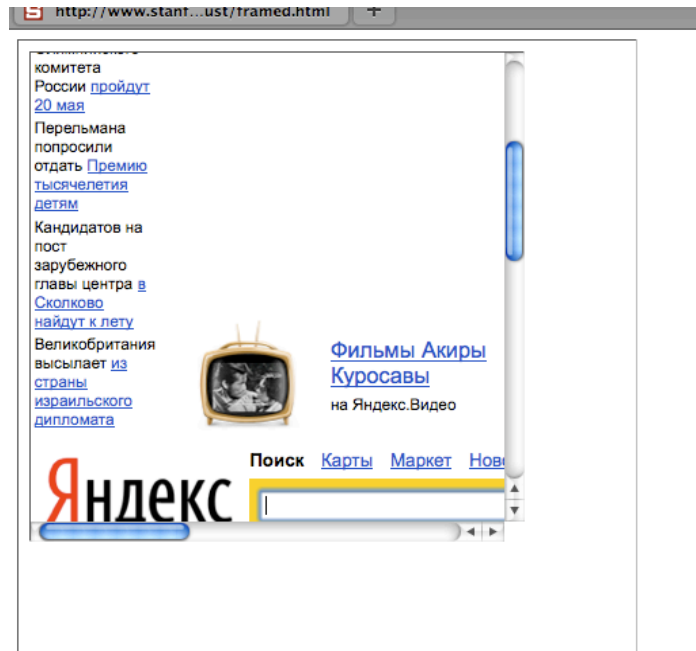
# Many Attacks on Referer Header

- Open redirect referer changer
- HTTPS->HTTP redirect changes the header
- Apparently, hard to get regular expression right
- Trust other sites to frame your pages, but what if those trusted sites can be framed themselves?

# Typical Frame Busting Code

```
if(top.location != self.location) {  
    parent.location = self.location;  
}
```

# Who Is Your ~~Daddy~~ Parent?



Double framing!!

```
framed1.html
<iframe
src="framed2.html">
```

```
framed2.html
<iframe
src="victim.com">
```

# Who Is On Top?

```
if (top.location != self.location)
  top.location = self.location
```

If **top.location** can be changed or disabled,  
this code is useless

# Location Clobbering

- IE 7

```
var location="clobbered";
```

- Safari

```
window.__defineSetter__("location", function(){});
```

– top.location now undefined

# User Can Stop Frame Busting

- User can manually cancel any redirection attempt made by frame busting code
- Attacker just needs to ask...

```
<script>
```

```
  window.onbeforeunload = function() {  
    return "Do you want to leave PayPal?";  
  }
```

```
</script>
```

```
<iframe src="http://www.paypal.com">
```

# Ask Nicely

The image shows a screenshot of a web browser window displaying the PayPal homepage. The browser's address bar shows the URL `http://www.stanford.edu/~rydstedt/c`. The page title is "Ask Nicely - Defeating Framebusting". The PayPal logo is visible at the top left, along with navigation links for "Sign Up", "Log In", and "Help". The main content area features a "WELCOME TO PayPal" banner and a "PayPal Shopping" section with the text "ALL THE BRAND". A confirmation dialog box is overlaid on the right side of the browser window, titled "Confirm". The dialog box contains the following text: "Are you sure you want to navigate away from this page? Do you want to leave PayPal? Press OK to continue, or Cancel to stay on the current page." The dialog box has two buttons: "Cancel" and "OK".



# ... Or Don't Even Ask

- Most browsers let **attacker cancel the relocation programmatically**

```
var prevent_bust = 0
window.onbeforeunload = function() {kill_bust++ }
setInterval(function() {
    if (kill_bust > 0) {
        kill_bust -= 2;
        window.top.location = 'http://no-content-204.com'
    }
}, 1);
<iframe src="http://www.victim.com">
```

# X-Frame-Options

- HTTP header sent with the page
- Two possible values: **DENY** and **SAMEORIGIN**
- DENY: page will not render if framed
- SAMEORIGIN: page will only render if top frame has the same origin

# Adoption of X-Frame-Options

- Good adoption by browsers
- Poor adoption by sites
- Limitations
  - Per-page policy
  - No whitelisting of origins
  - Proxy problems

# Content Security Policy (Firefox 4)

- Another HTTP header: **frame-ancestors** directive can specify allowed framers
- Allows specific restrictions and abilities per site

# Best For Now (Still Not Good)

```
<style>html { visibility: hidden }</style>
<script>
if (self == top) {
  document.documentElement.style.visibility = 'visible';
} else {
  top.location = self.location;
}
</script>
```