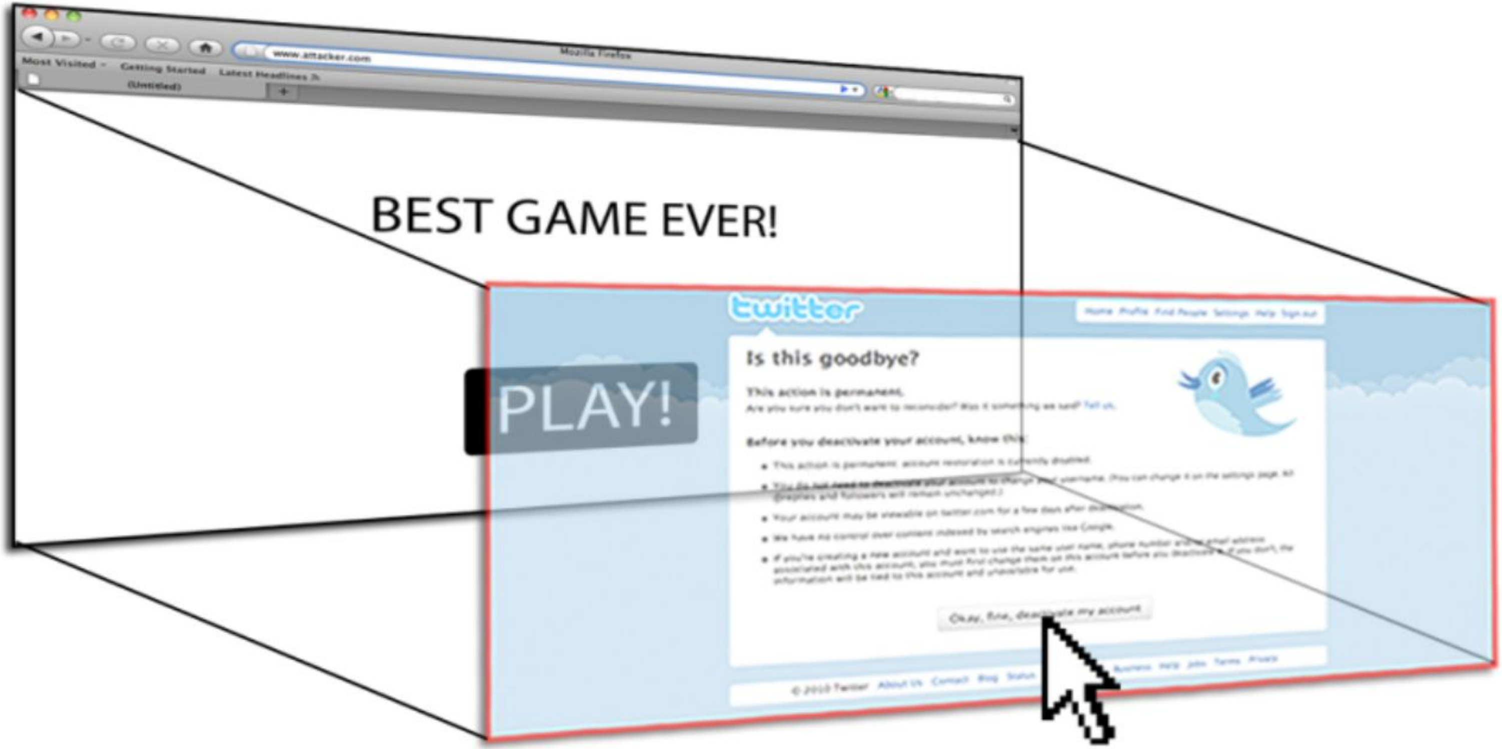
Clickjacking

Lecture 14

Software Security Engineering

Winter 2023
Thompson Rivers University



Is this goodbye?

This action is permanent.

Are you sure you don't want to reconsider? Was it something we said? Tell us.

Before you deactivate your account, know this:

- This action is permanent; account restoration is currently disabled.
- You do not need to deactivate your account to change your username. (You can change it on the settings page. All @replies and followers will remain unchanged.)
- Your account may be viewable on twitter.com for a few days after deactivation.
- We have no control over content indexed by search engines like Google.
- If you're creating a new account and want to use the same user name, phone number and/or email address associated with this account, you must first change them on this account before you deactivate it. If you don't, the information will be tied to this account and unavailable for use.

Okay, fine, deactivate my account
© 2010 Twitter About Us Contact Blog Status Business Help Jobs Terms Privacy

Clickjacking

- portmanteau of “click hijacking”
- attacker overlays multiple transparent or opaque frames
  - trick a user into clicking a button or link on another page
- circumvents same-origin policy
  - malicious page cannot click the link itself

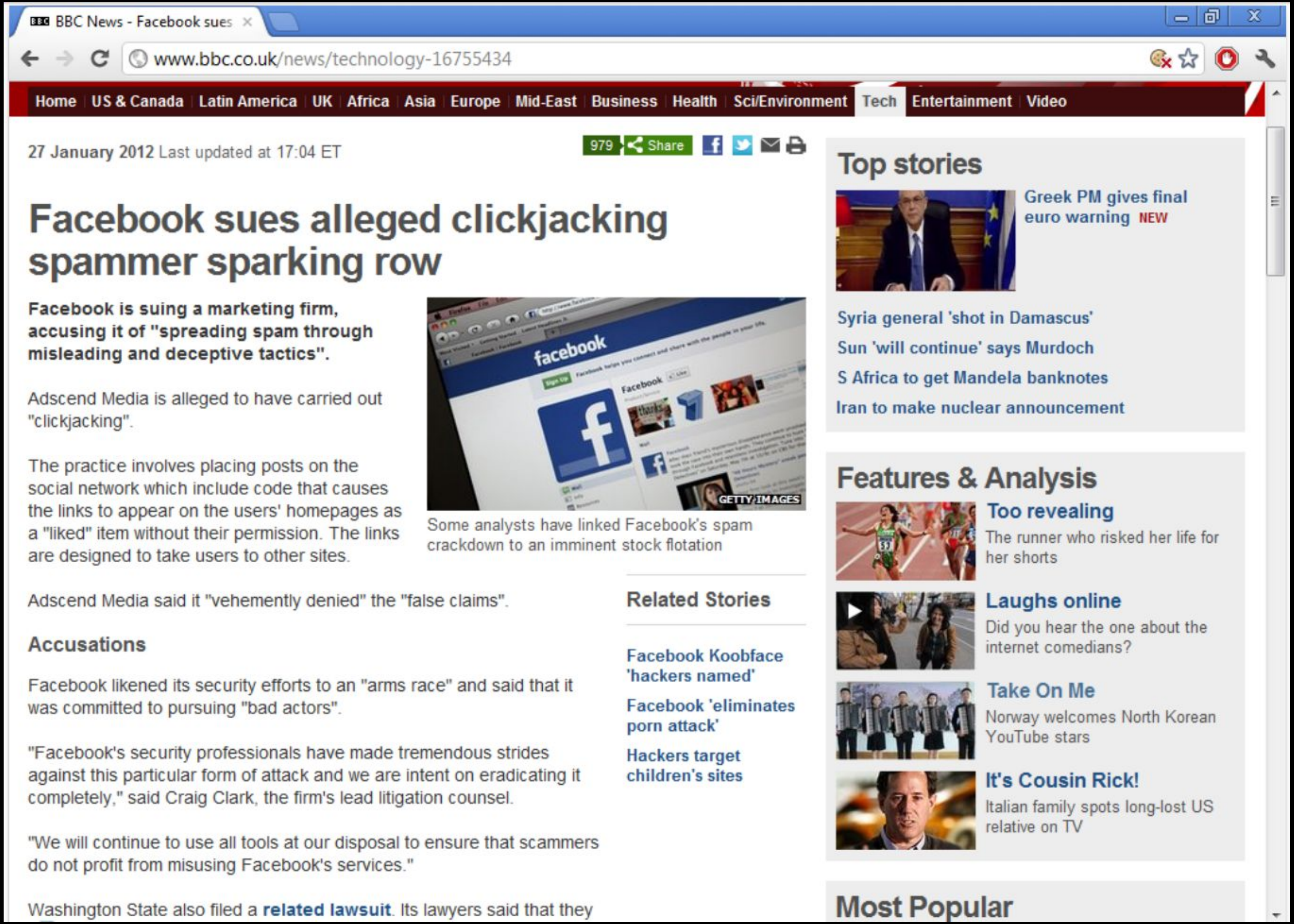
Clickjacking in the Wild:

Facebook worm superimposes invisible iframe
over entire page that links to victim’s Facebook page

Clickjacking in the Wild:

Facebook worm superimposes invisible iframe
over entire page that links to victim’s Facebook page

If victim is logged in, automatically recommends
link to new friends as soon as it is clicked on.



27 January 2012 Last updated at 17:04 ET

Facebook sues alleged clickjacking spammer sparking row

Facebook is suing a marketing firm, accusing it of "spreading spam through misleading and deceptive tactics".

Some analysts have linked Facebook's spam crackdown to an imminent stock flotation

Adscend Media is alleged to have carried out "clickjacking".

The practice involves placing posts on the social network which include code that causes the links to appear on the users' homepages as a "liked" item without their permission. The links are designed to take users to other sites.

Adscend Media said it "vehemently denied" the "false claims".

Accusations

Facebook likened its security efforts to an "arms race" and said that it was committed to pursuing "bad actors".

"Facebook's security professionals have made tremendous strides against this particular form of attack and we are intent on eradicating it completely," said Craig Clark, the firm's lead litigation counsel.

"We will continue to use all tools at our disposal to ensure that scammers do not profit from misusing Facebook's services."

Washington State also filed a **related lawsuit**. Its lawyers said that they

Related Stories

- Facebook Koobface 'hackers named'
- Facebook 'eliminates porn attack'
- Hackers target children's sites

Top stories

- Greek PM gives final euro warning NEW
- Syria general 'shot in Damascus'
- Sun 'will continue' says Murdoch
- S Africa to get Mandela banknotes
- Iran to make nuclear announcement

Features & Analysis

Too revealing
The runner who risked her life for her shorts

Laughs online
Did you hear the one about the internet comedians?

Take On Me
Norway welcomes North Korean YouTube stars

It's Cousin Rick!
Italian family spots long-lost US relative on TV

Most Popular

Twitter Clickjack

Users send out tweets against their will.

Twitter Clickjack

Users send out tweets against their will.

Users are tricked into clicking a post-to-twitter link.

Twitter Clickjack

Users send out tweets against their will.

Users are tricked into clicking a post-to-twitter link.

Works if they are logged in

Likejacking: clickjacking in the context of the Facebook like button.

But wait: how isn’t this just XSRF?

Clickjacking attack: when a user’s mouse click
is used in a way that was not intended by user.

Simple Example

```
<a onMouseDown=window.open(http://www.evil.com)
    href=http://www.google.com/> anchor text </a>
```

- anchor goes to evil.com
- why the google.com?

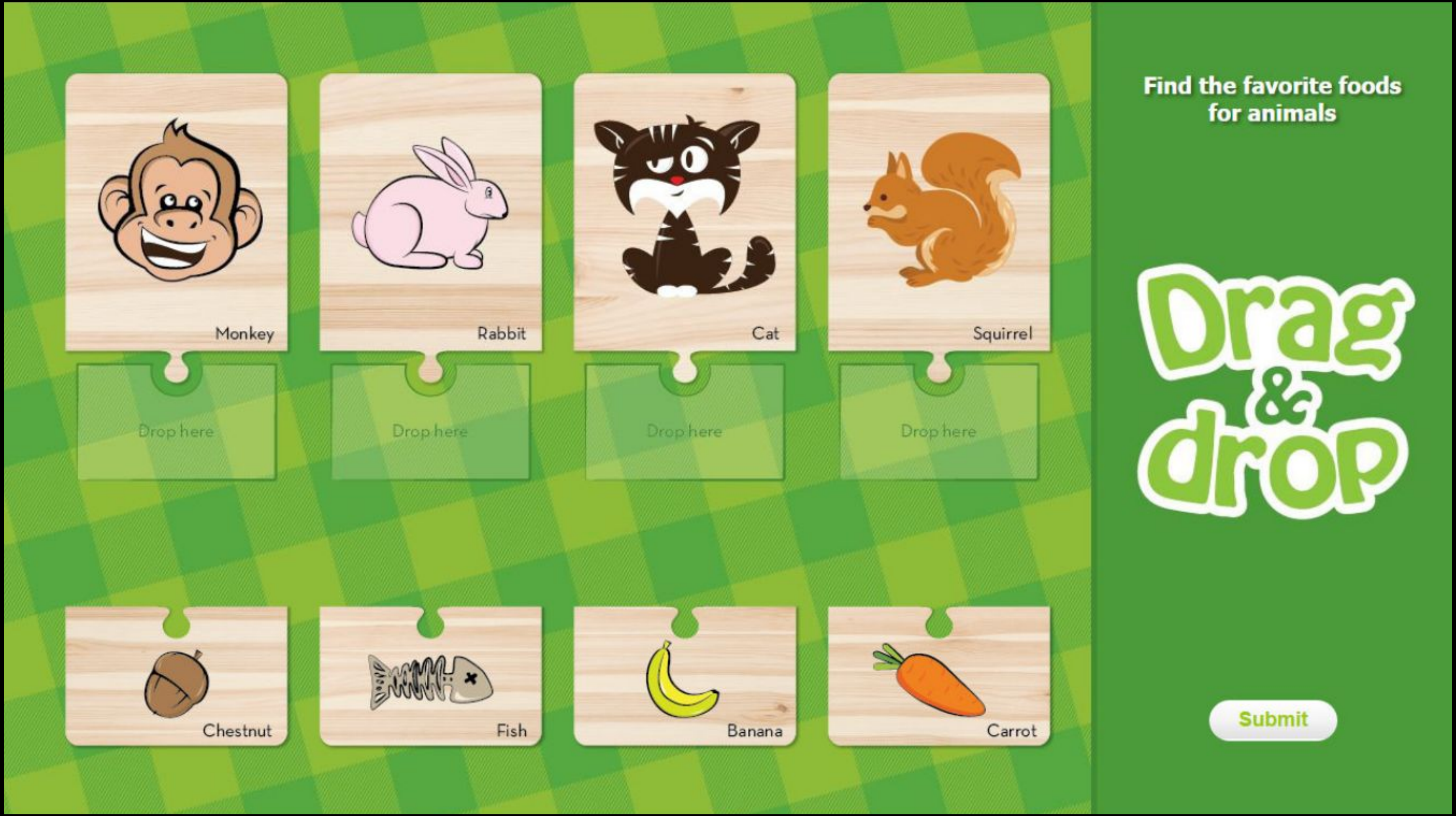
iframes

- any website can frame any other website
  - have a subwindow or such that shows its content
- main frame does not need to handle all the logic of managing two things
  - subframe can be its own session, links clicking, changing page, etc.
- `<iframe src=“http://www.google.com/...”> </iframe>`
  - HTML attributes include OPACITY (percentage visible)
    - 1.0: totally visible
    - 0.0: totally invisible
  - z-index: position on the stack (top gets clicks)
  - pointer-event: set to none to say ignore click (goes to next)

Drag-and-Drop Abuse

- same origin policy stops the html page to “see” what the user selects in an iframe
  - e.g., iframe_text_field.textContents throws an exception
- but selected text can be dragged into an object despite same origin
  - motive is that user does this deliberately
  - i.e., mouse events cannot be spoofed

How can this be exploited?



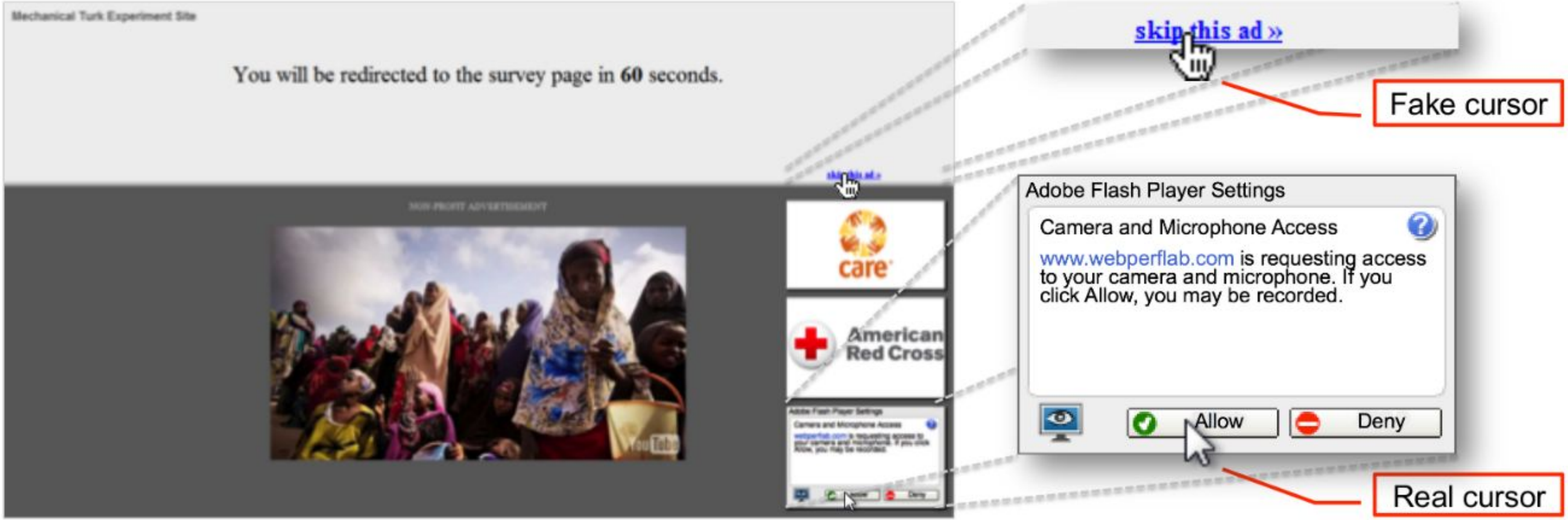
Find the favorite foods for animals

Abusing Drag-n-Drop

- only need to get the user to drag and drop for any reason
- hidden iframes will load the data that the evil site wants
- destination will be an HTML object within the evil site’s control
- user is tricked into circumventing same origin policy

Cursorjacking

- mouse cursor can be turned off in the web browser
  - CSS CURSOR property supports “none”
- then create another cursor in javascript that follows the mouse movement
  - different looking cursors won’t necessary be suspicious
  - though different cursor physics will be noticable



Mechanical Turk Experiment Site

You will be redirected to the survey page in **60** seconds.

NON-PROFIT ADVERTISEMENT

Adobe Flash Player Settings

Camera and Microphone Access

www.webperflab.com is requesting access to your camera and microphone. If you click Allow, you may be recorded.

Fake cursor

Real cursor

Strokejacking: suppose that bank.com needed the user to enter in numbers for an amount to do a bank transfer.

Strokejacking: suppose that bank.com needed the user to enter in numbers for an amount to do a bank transfer.

That is, clicks aren’t enough: the user has to hit keys.

Strokejacking: suppose that bank.com needed the user to enter in numbers for an amount to do a bank transfer.

That is, clicks aren’t enough: the user has to hit keys.

SOP stops this from being faked.

Strokejacking

- site convinces the user to type some keystrokes on a simulated input field
- actual keystrokes being sent to the iframe that needs it
- e.g., numbers become the amount to send.
- how could the user be tricked?

All these attacks conspire to break SOP.

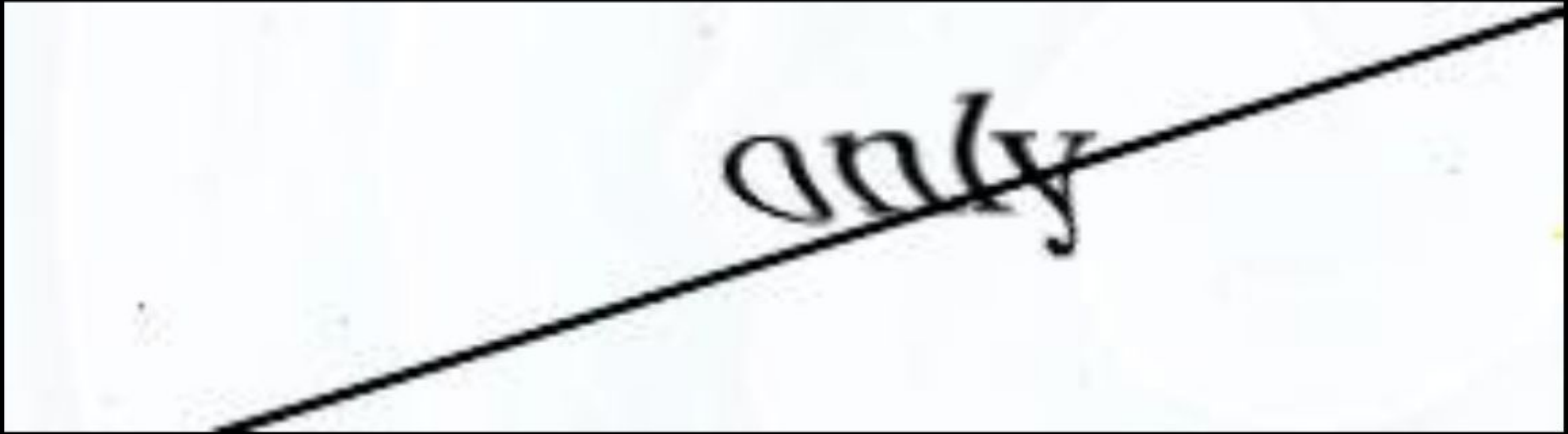
All these attacks conspire to break SOP.

They require human effort to click or type
and the user is being tricked into doing that.

Compromise Temporal Integrity

- temporal integrity refers to the state remaining the same in time
  - security issue involving something changing after security check is done but before something being allowed by that check is done
  - TOCTTOU: time of check to time of use
- for clickjacking, it means changing the UI after the user decides to click but before the click occurs
  - e.g., if logic executes on onClick, then change UI on mouseDown
  - e.g., bait the user to double click, and swap the UI between them

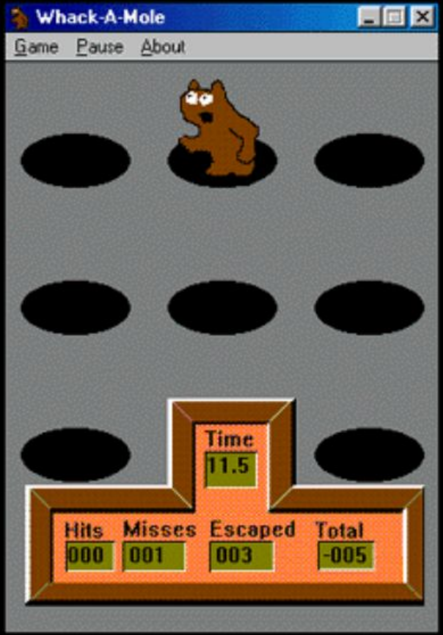
Temporal Integrity



only

Whack-A-Mole Attack

- bait the user to click as fast as possible
- switch to a different UI button when appropriate



Lots of choices! How do we stop this?

Solution: user confirmation

Solution: user confirmation

Good site pops up dialogue box with info about what it is about to do and confirms

Solution: user confirmation

Good site pops up dialogue box with info about what it is about to do and confirms

awful user experience

Solution: UI Randomization

Good site embeds form elements at random locations
so it is hard to overlay

Good site embeds form elements at random locations
so it is hard to overlay

e.g., paypal pay button always in different location

Good site embeds form elements at random locations
so it is hard to overlay

e.g., paypal pay button always in different location

awful user experience
multi-click attack

Solution: Opaque Policy

Solution: Opaque Policy

no element can be transparent

each pixel belongs to a single element

Solution: Opaque Policy

no element can be transparent

each pixel belongs to a single element

any problems?

Partial Overlaps and Cropping

- don’t completely cover the target
- instead hide the important parts
  - e.g., message that you mean to post
  - e.g., amount that your credit card is charged

Solution: Frame Busting

Solution: Frame Busting

I am the page owner (what gets put in iframe)

Solution: Frame Busting

I am the page owner (what gets put in iframe)

I insist that I am never loaded in an iframe

Solution: Frame Busting

I am the page owner (what gets put in iframe)

I insist that I am never loaded in an iframe

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

Frame Busting

- conditional check for iframing
  - take counter-action if iframing is detected
  - then no user behaviour on site is result of clickjacking
- doesn’t work for embedded stuff like facebook “like” buttons but oh well

So clickjacking is solved!

Frame Busting in the Wild

- survey of practices by Gustav Rydstedt, Elie Bursztein, Dan Boneh, and Collin Jackson
- looked at Alexa top 500 websites and all top US banks
- 14% use framebusting
- found 100% of framebusting can be circumvented one way or another
    - oops
    - some browser specific
    - some cross browser

Frequently it was in the code to allow their own iframes

Frequently it was in the code to allow their own iframes

i.e., I don’t want to be an iframe, but I want to have my own things as iframes

Frequently it was in the code to allow their own iframes

i.e., I don’t want to be an iframe, but I want to have my own things as iframes

and they are okay with being iframes as long as I’m still the main frame.

Frequently it was in the code to allow their own iframes

i.e., I don’t want to be an iframe, but I want to have my own things as iframes

and they are okay with being iframes as long as I’m still the main frame.

This policy can be hard to implement.

Walmart’s Framebusting

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

Error in Referrer Checking:

website http://www.attacker.com/walmart.com.html has the iframe

The New York Times's Framebusting

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

Error in Referer Checking:

website

http://eve.com/a.html?b=https://www.nytimes.com/

has the iframe

US Bank’s Framebusting

```
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:

website

http://usbank.attacker.com

has the iframe

Error in Referer Checking:

website
http://usbank.attacker.com
has the iframe

or the Norwegian State House Bank
http://www.husbanken.no

Error in Referer Checking:

website
http://usbank.attacker.com
has the iframe

or the Norwegian State House Bank
http://www.husbanken.no

or the Rusbank http://www.rusbank.org
(it's actually Rosbank, but still)

Typical Frame Busting code:

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

Double Framing Attack:

main frame has <iframe src=“frame2.html”>

Double Framing Attack:

main frame has <iframe src=“frame2.html”>

frame2.html has <iframe src=“victim.com”>

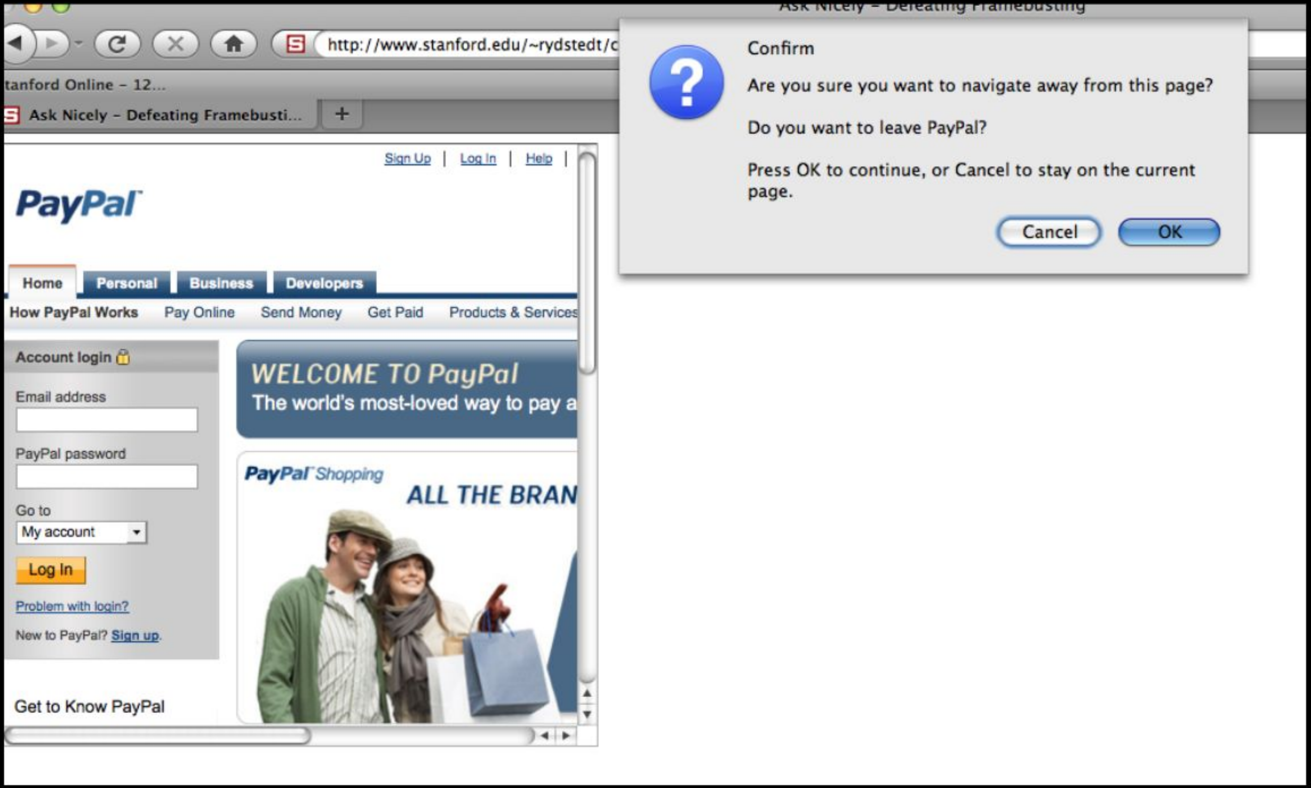
A fix?

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

Location Clobbering

- IE7: var location=“clobbered”;
- Safari: window.__defineSetter__(“location”, function(){})

Asking Nicely



Confirm

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.

Frame busting from Paypal will be cancelled if the user clicks cancel.

Frame busting from Paypal will be cancelled if the user clicks cancel.

The pop-up is actually the iframer’s onbeforeunload function.

The onbeforeunload event occurs when a document is about to be unloaded.

Best at the time

- style html’s body as “display: none”
- try to framebust if “self != top”
- change style to “display: block” if “self == top”

Now going forward

- X-Frame-Options HTTP header sent with page
- two possible values: DENY and SAMEORIGIN
- DENY: page will not render if framed
- SAMEORIGIN: page will only render if top frame has same origin
- up to sites to support