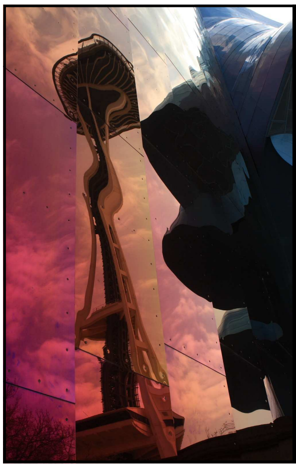


Thwarting Malware and UI Redressing Attacks with Verifiable User Actions

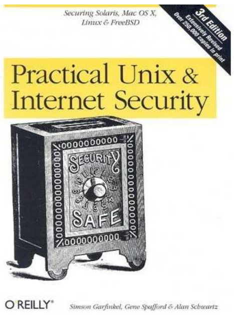
University of Washington
1 May 2009

Jeff Shirley and David Evans
University of Virginia
Department of Computer Science

Security is all about User Intentions

*A computer is **secure** if you can depend on it and its software to behave as you expect.*

(Garfinkel, Spafford & Schwartz)



Securing Solaris, Mac OS X, Linux & FreeBSD
3rd Edition
Practical Unix & Internet Security
SECURITY SAFE
O'REILLY
Simson Garfinkel, Gene Spafford & Alan Schwartz

Examples

Malware

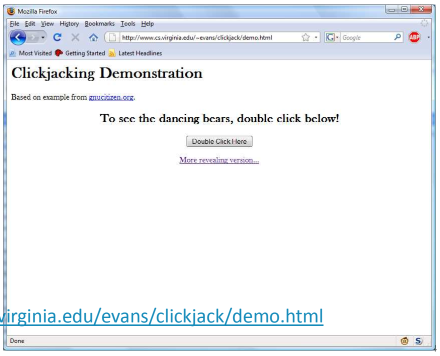
Phishing

CAPTCHAs

Next: Clickjacking

Images: HowStuffWorks.com; DevCentral.com; www.virtualblight.com

Clickjacking (UI Redressing)



Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.cs.virginia.edu/~evans/clickjack/demo.html
Google
Most Visited
Getting Started
Latest Headlines
Done

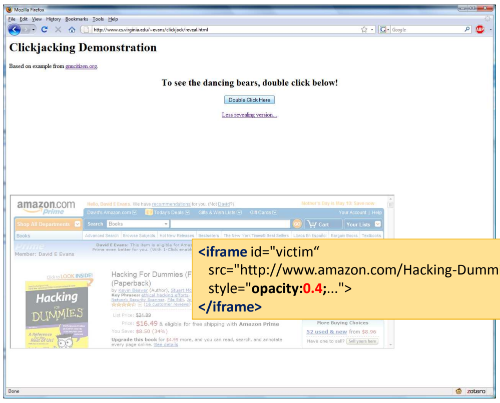
Clickjacking Demonstration

Based on example from gnucitizen.org.

To see the dancing bears, double click below!

More revealing version...

http://www.cs.virginia.edu/evans/clickjack/demo.html



Clickjacking Demonstration

Based on example from gnucitizen.org.

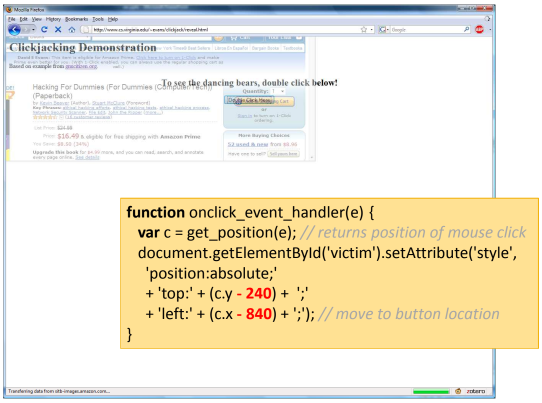
To see the dancing bears, double click below!

Less revealing version...

Hacking For Dummies (Paperback)

List Price: $24.99
Price: $16.49 & eligible for free shipping with Amazon Prime
You Save: $8.50 (34%)

```
<iframe id="victim“
  src="http://www.amazon.com/Hacking-Dummies-..."
  style="opacity:0.4;...">
</iframe>
```



Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.cs.virginia.edu/~evans/clickjack/reveal.html
Google
Double Click Here
Quantity: 1
Sign in to turn on 1-Click ordering.
More Buying Choices
52 used & new from $8.96
Have one to sell? Sell yours here
Transferring data from sitb-images.amazon.com...
zotero

Clickjacking Demonstration

Based on example from gnucitizen.org.

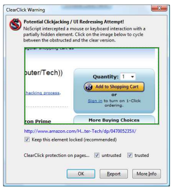
To see the dancing bears, double click below!

Hacking For Dummies (For Dummies (Computer/Tech)) (Paperback)

```
function onclick_event_handler(e) {
  var c = get_position(e); // returns position of mouse click
  document.getElementById('victim').setAttribute('style',
   'position:absolute;'
   + 'top:' + (c.y - 240) +  ';'
   + 'left:' + (c.x - 840) + ';'); // move to button location
}
```

UI Redressing Claims

- No good server-side defense
  - Server sees two perfectly normal requests
- Client-side defenses
  - Change browser to prevent attack page
    - e.g., no transparent frames, better display-sharing policy
    - Need to break backwards compatibility
  - NoScript's approach: warn when clicks reach hidden elements
  - **General defense: only allow actions that are consistent with user intentions**


ClearClick Warning

Potential Clickjacking / UI Redressing Attempt!
NoScript intercepted a mouse or keyboard interaction with a partially hidden element. Click on the image below to cycle between the obstructed and the clear version.

http://www.amazon.com/H...ter-Tech/dp/047005235X/

☑ Keep this element locked (recommended)

ClearClick protection on pages... ☑ untrusted ☑ trusted

Related Work: Using User Intentions

[Chou, Ledesma, Teraguchi, Boneh, Mitchell, *NDSS* 2004]

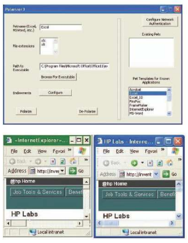
"BINDER exploits a unique characteristic of personal computers, that most network activities are directly or indirectly triggered by user input."

BINDER

[Cui, Katz & Tan, *USENIX Tech* 2005]

Not-a-Bot

[Gummadi, Balakrishnan, Maniatis, Ratnasamy, *NSDI* 2009]



Polaris

[Stiegler, Karp, Yee, Close & Miller, *CACM* 2006]

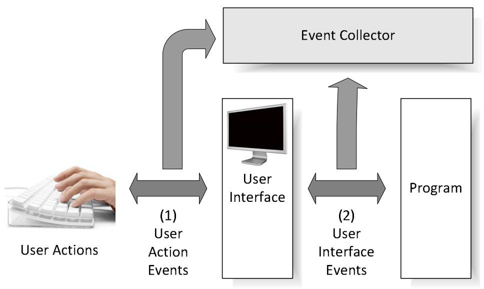
CapDesk [HP; Google]

Our Goal

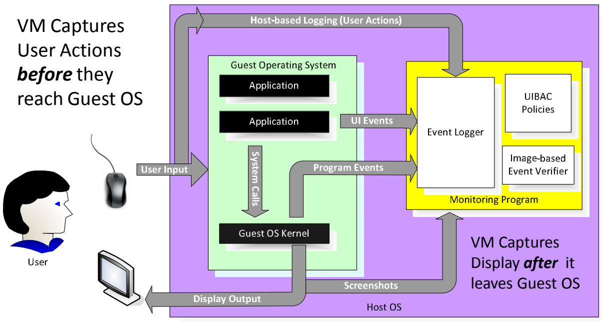
- **Systematically incorporate user intentions in security policies**
- Outline:
  - **Securely capture user actions**
  - Robustly infer user intentions from those actions
  - Express and enforce policies that incorporate user intentions

How can we securely capture user intentions?

Capturing User Actions



VM Captures User Actions ***before*** they reach Guest OS

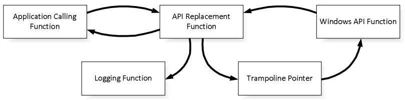


VM Captures Display ***after*** it leaves Guest OS

User-Intent Based Access Control

Main Challenges

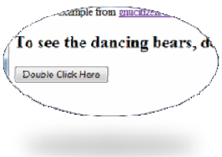
- **Inferring User Intent:** depends on what user **does** and **sees**
- **Designing UIBAC Policies:** grant permissions based on the history of user intentions and program actions
- Intercepting actions, enforcing policies



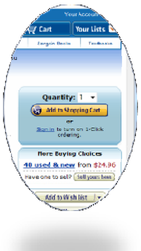
Galen Hunt and Doug Brubacher. *Detours: Binary Interception of Win32 Functions.* 1999.

Protecting User Interfaces

Intent of user action depends on **apparent** UI element user is interacting with

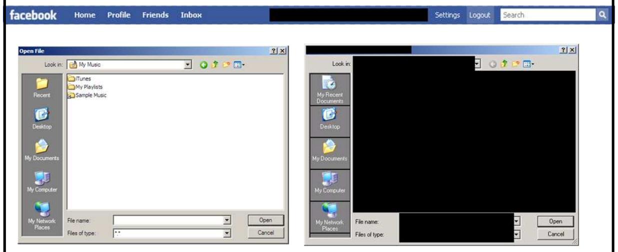


vs.



Your Account
Cart
Your Lists
Quantity: 1
Add to Shopping Cart
or
Sign in to turn on 1-Click ordering.
More Buying Choices
40 used & new from $24.96
Have one to sell? Sell yours here
Add to Wish List

Visual Templates

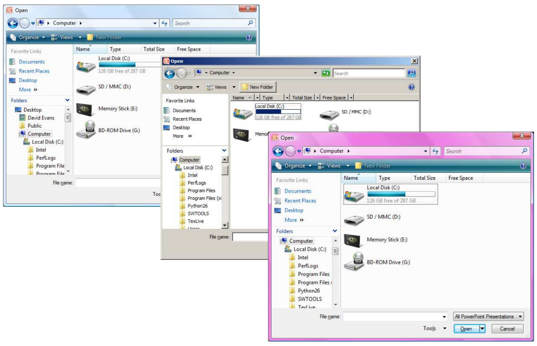


Prototype compares visual output from a virtual machine to "visual templates" that specify look of user interface elements

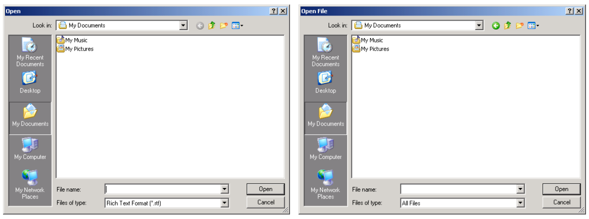
Template Matching

- Templates consist of a bitmap image plus a set of regions that are ignored during comparison
  - Compare screenshot with image template
  - Use precomputed SHA-1 hash for speed
- Ignored regions generalize visual templates
  - Tradeoff between generality and exactness of UI matching

Template Challenges



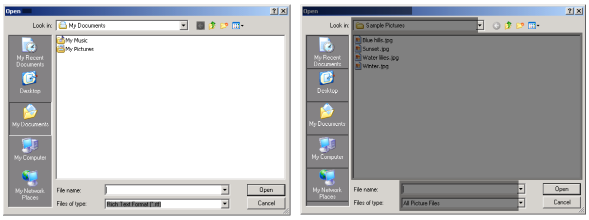
Learning Templates



99.75% of pixels identical

Collect screenshots of related dialogs by running trusted applications

Generalizing Templates



Generalize by clustering mismatched pixels, find minimal bounding boxes of varying regions, exclude from template

Inferring Intentions

- Many ways to express same intention
  - Mouse click sequences, keyboard shortcuts, etc.
- Sets of rules of inferring particular abstract intentions
  - e.g., intent to open file *f*

User-Intent Based Policies

Policies

- Universal Policies
  - **Anti-Malware**: apply to all processes
    - access files **selected by user** (e.g., File open/save dialog)
    - access files and directories **installed** with application
    - access files **created** by application
  - Take advantage of user intentions to enable default strict policy that is relaxed based on user actions
- Application-Specific Policies
  - Resource: amazon.com/add-to-cart
  - Granted by: user click on template

Anti-Malware Policy

file_save(Filename *f*, Process *p*)
grants (Process *p*) Write (Filename *f*)
file_save(Filename *f*, Process *p*)
grants (Process *p*) Create (Filename *f*)
file_open(Filename *f*, Process *p*)
grants (Process *p*) Write (Filename *f*)

program_creates_file(Filename *f*, Executable *e*)
grants (Executable *e*) Write (Filename *f*)

program_installer(Process *p*)
grants (Process *p*) Create (Filename *f*)
program_installer_creates(Process *p*, Executable *e*, Directory *d*)
grants (Executable *e*) Create (Directory *d*)
program_installer_creates(Process *p*, Executable *e*, Filename *f*)
grants (Executable *e*) Write (Filename *f*)

Mandatory Access Control
- Default deny
- Permissions granted based on history of all user interactions

Malware Preliminary Results

Prevention: 30 effective malware samples: all malicious behaviors prevented (except for limits in intercepting actions)

False Positives

Program	Actions	Policy Violations	Dialogs Validated
Firefox 3.0.5	4739	2	✓
iTunes 8.0	13382	0	✓
Windows Media Player 11.0	8217	8	✓
Wordpad 5.1	2897	0	✓
Word 2007	6303	5	No

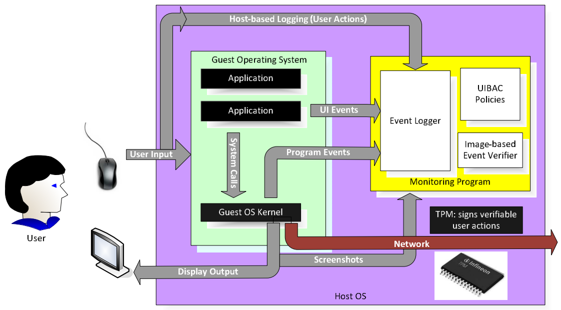
Flash component files

Uses IE as embedded component, accesses history and cookies

Uses non-standard UI elements

So, what about those CAPTCHAs?

"Spooky Actions at a Distance"



Externally-Verifiable User Actions

- Network messages include TPM-signed tokens
  - **Option 1:** attest to filter that collects and signs screenshots and inputs
  - **Option 2:** attest to analyzer that signs verified events
- Sample Applications:
  - No more CAPTCHAs!
  - Eliminate click fraud (only pay for signed clicks?)
  - Prevent worm propagation
  - Non-repudiatable transactions

Summary

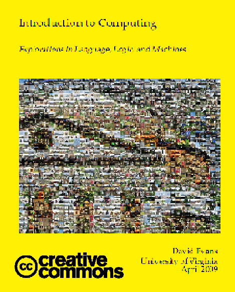
- Security is all about user intentions
  - Expressed through normal interactions, not security dialogs
- Understanding them is hard: interpreting intentions depends on understanding what users **do** and **see**
- Lots of opportunities to use collected user intentions
  - Desktop User-Intent Based Access Control
    - Universal policies to thwart malware
    - Application-specific policies
  - Externally-verifiable user actions
    - Verifiable, non-repudiatable user transactions

Shameless Book Plug

http://www.computingbook.org/

1. Defining Procedures
2. Analyzing Procedures
3. Improving Expressiveness
4. Limits of Computing
5. Programming the Web

Main underlying themes:
- Recursive definitions
- Abstraction
- Universality (Programs/Data)


Introduction to Computing
Explorations in Language, Logic, and Machines
creative commons
David Evans
University of Virginia
April 2009

Jeff Shirley and David Evans
University of Virginia
evans@cs.virginia.edu
http://www.cs.virginia.edu/evans