Security, Privacy, and User Expectations:

Case Studies in Browsers and Smartphones

Franziska Roesner

Department of Computer Science & Engineering
University of Washington

Security, Privacy, and User Expectations:

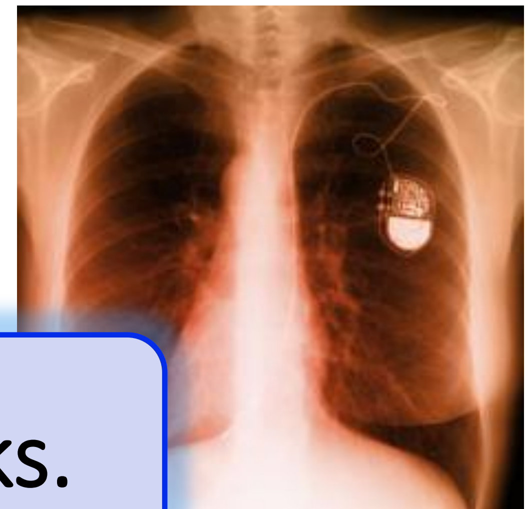
Case Studies in Browsers and Smartphones

Franziska Roesner

Department of Computer Science & Engineering
University of Washington

In collaboration with: *James Fogarty, Tadayoshi Kohno, David Molnar, Alexander Moshchuk, Bryan Parno, Chris Rovillos, Alisha Saxena, Helen Wang, David Wetherall, and others.*

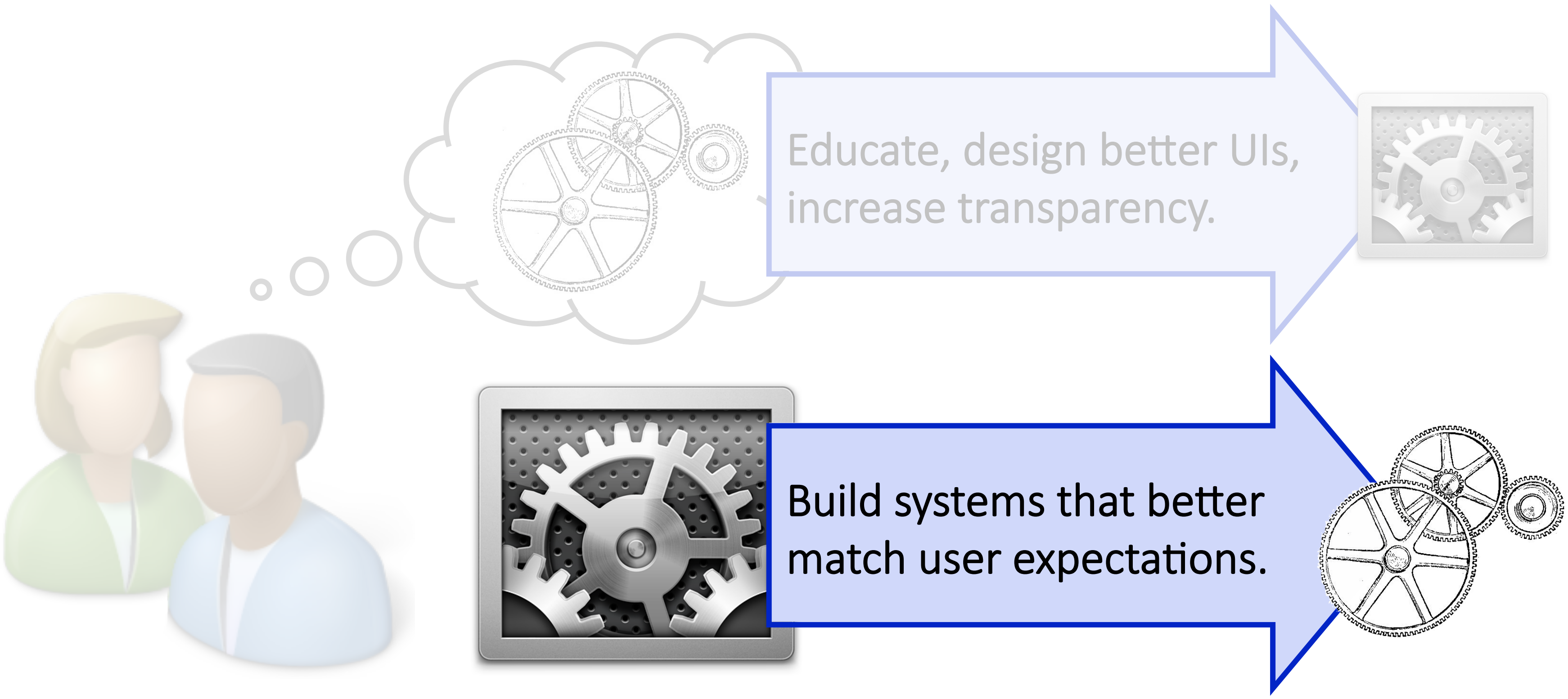
New technologies bring new benefits…



… but also new risks.

Improving Security & Privacy

Security and privacy challenges often arise when
user expectations don't match real system properties.



Outline

I. **Browsers:**
   Third-Party Tracking

II. **Smartphones:**
   Permission Granting

III. **Security & Privacy in Other Contexts**

Outline

I. **Browsers:**
Third-Party Tracking

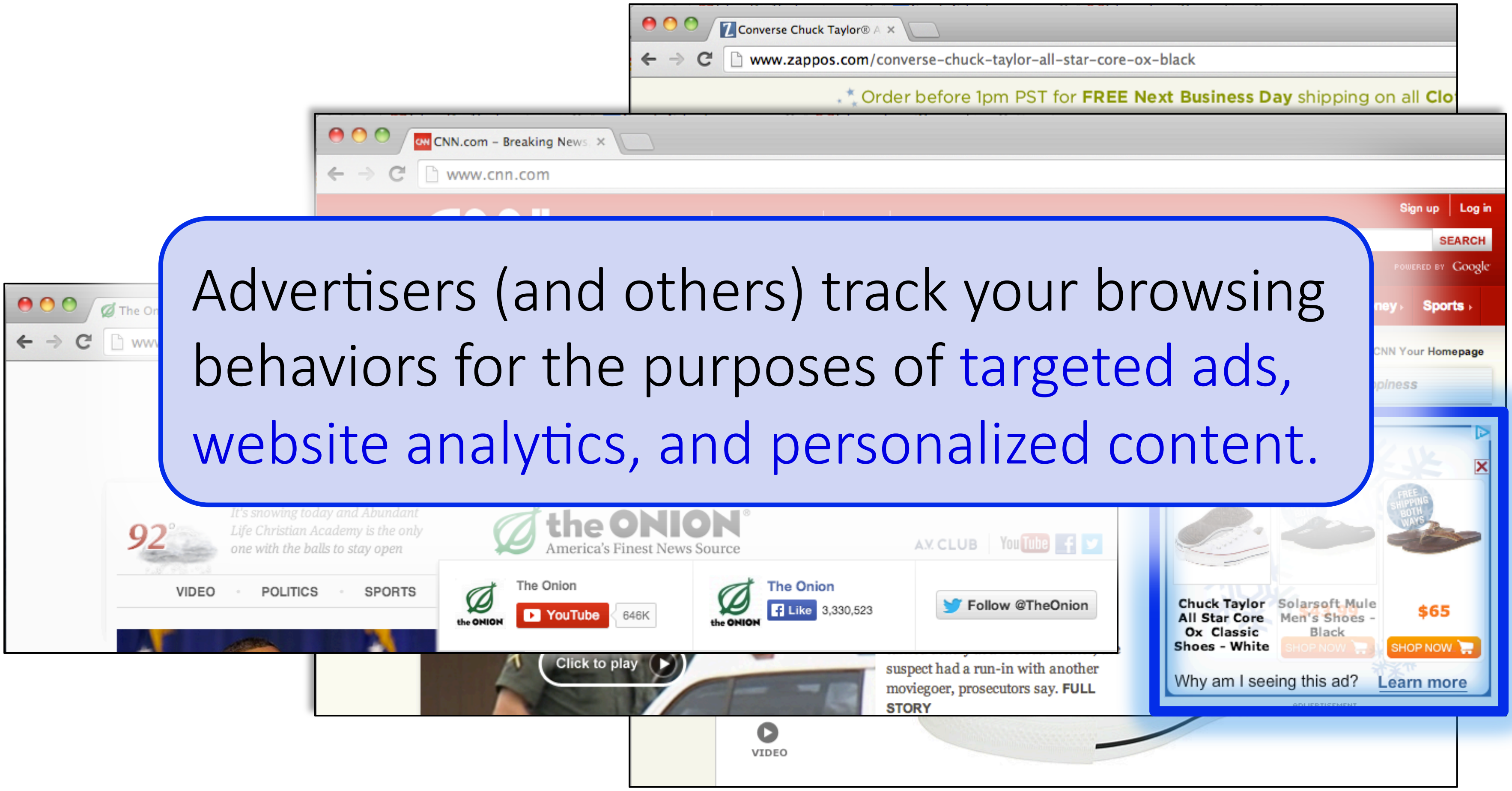
II. Smartphones:
Permission Granting

III. Security & Privacy in Other Contexts

F. Roesner, T. Kohno, D. Wetherall. "Detecting and Defending Against Third-Party Tracking on the Web." In USENIX Symposium on Networked Systems Design and Implementation (NSDI) 2012.

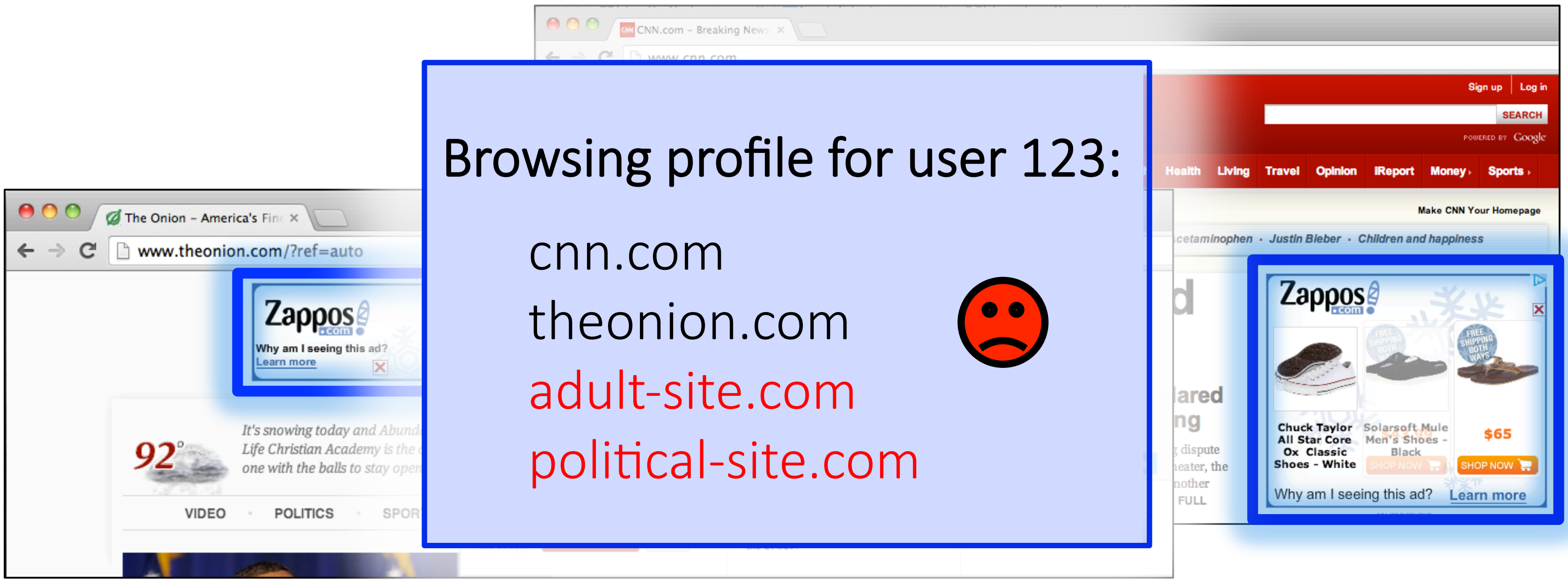
F. Roesner, C. Rovillos, T. Kohno, D. Wetherall. "ShareMeNot: Balancing Privacy and Functionality of Third-Party Social Widgets." In USENIX ;login: 2012.

Ads That Follow You



Advertisers (and others) track your browsing behaviors for the purposes of targeted ads, website analytics, and personalized content.

Third-Party Web Tracking



These ads allow **criteo.com** to link your visits between sites, even if you never click on the ads.

Concerns About Privacy (2010 – 2011)

THE WALL STREET JOURNAL.

WHAT THEY KNOW | JULY 30, 2010

The Web's New Gold Mine: Your Secrets

The New York Times

May 6, 2011, 5:01 pm | 3 Comments

'Do Not Track' Privacy Bill Appears in Congress

By TANZINA VEGA

And the privacy legislation just keeps on coming.

On Friday, two bills were introduced in Washington in support of a Do Not Track mechanism that would give users control over how much of their data was collected by advertisers and other online companies.

Understanding the Tracking Ecosystem

In 2011, much discussion about tracking, but limited understanding of how it actually works.

Our Goal: systematically study web tracking ecosystem to inform policy and defenses.

Challenges:

– No agreement on definition of tracking.

– No automated way to detect trackers. (State of the art: blacklists)

Our Approach

ANALYZE

(1) Reverse-engineer trackers' methods.
(2) Develop tracking taxonomy.

MEASURE

(3) Build automated detection tool.
(4) Measure prevalence in the wild.
(5) Evaluate existing defenses.

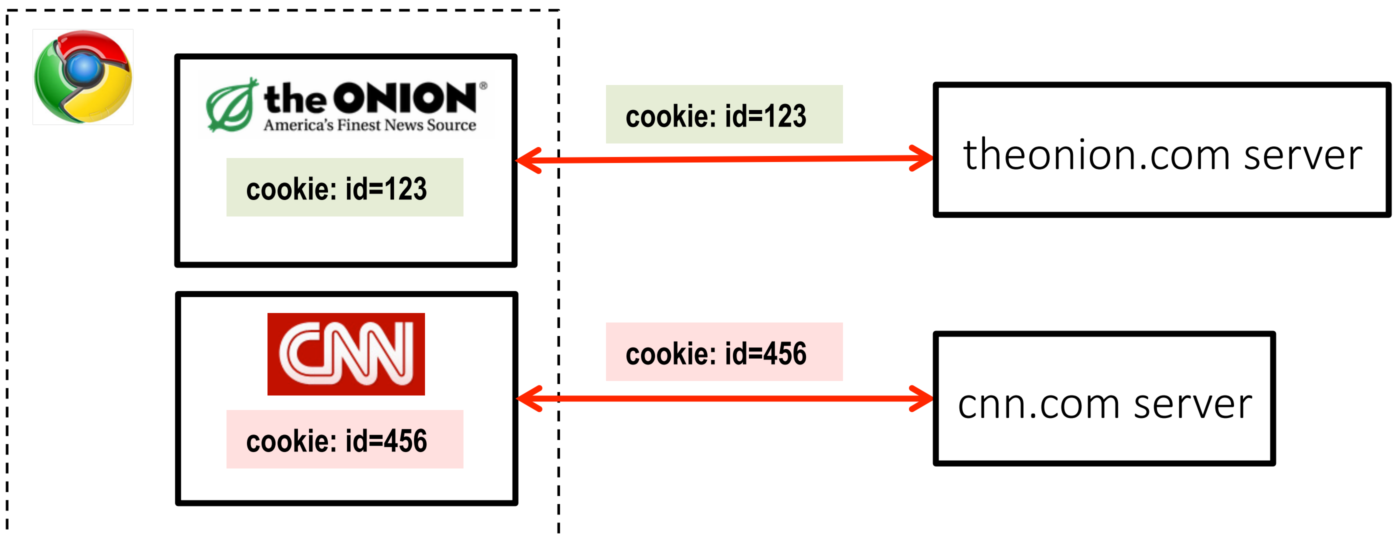
BUILD

(6) Develop new defenses.

Web Background

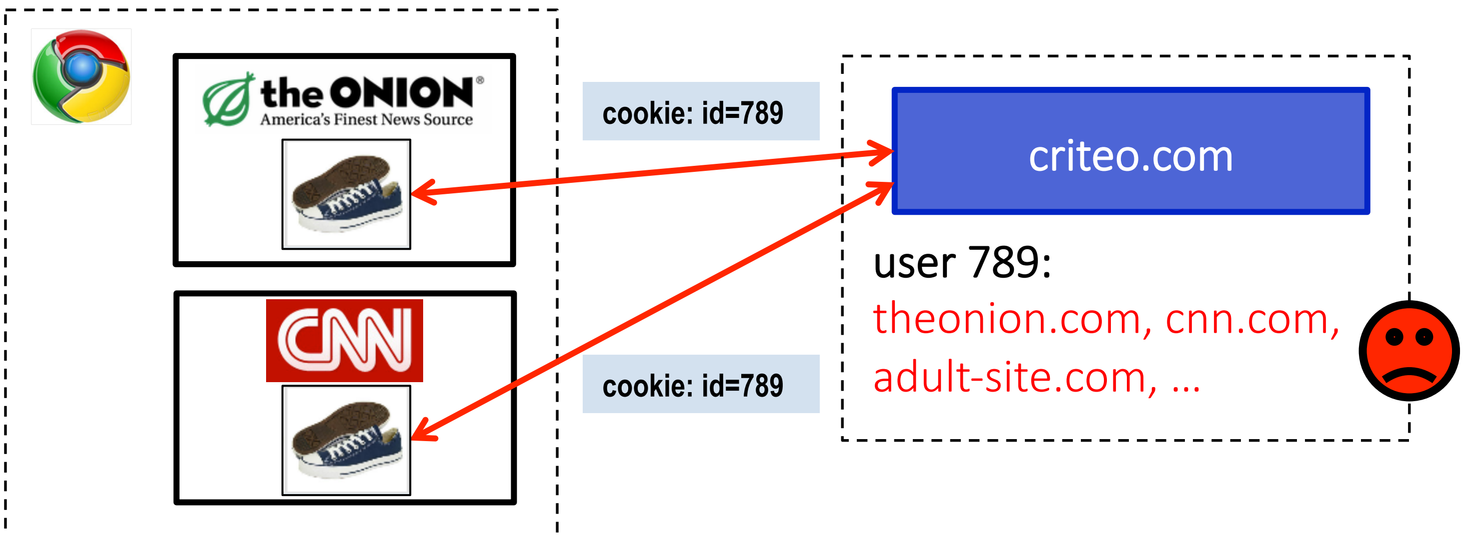
Websites store info in cookies in the browser.

- Only accessible to the site that set them.
- Automatically included with web requests.



Anonymous Tracking

Trackers included in other sites use cookies containing unique identifiers to create browsing profiles.



Our Tracking Taxonomy *[NSDI '12]*

In the wild, tracking is much more complicated.

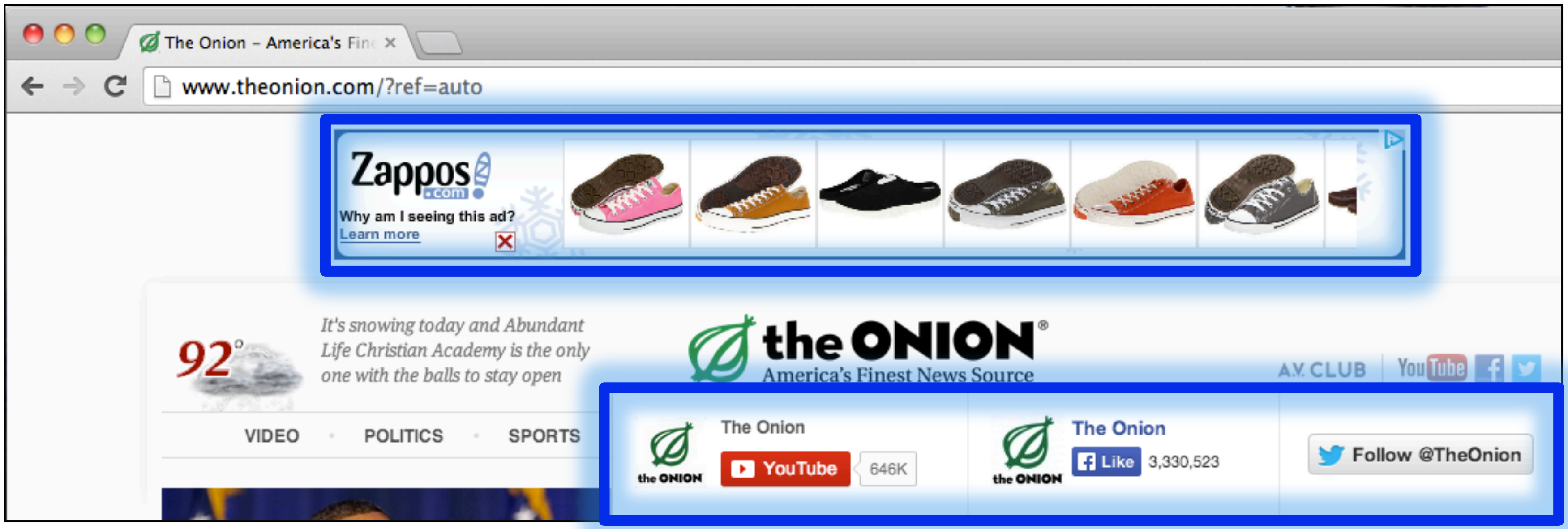
(1) Trackers don't just use cookies.

- Flash cookies, HTML5 LocalStorage, etc.

(2) Trackers exhibit different behaviors.

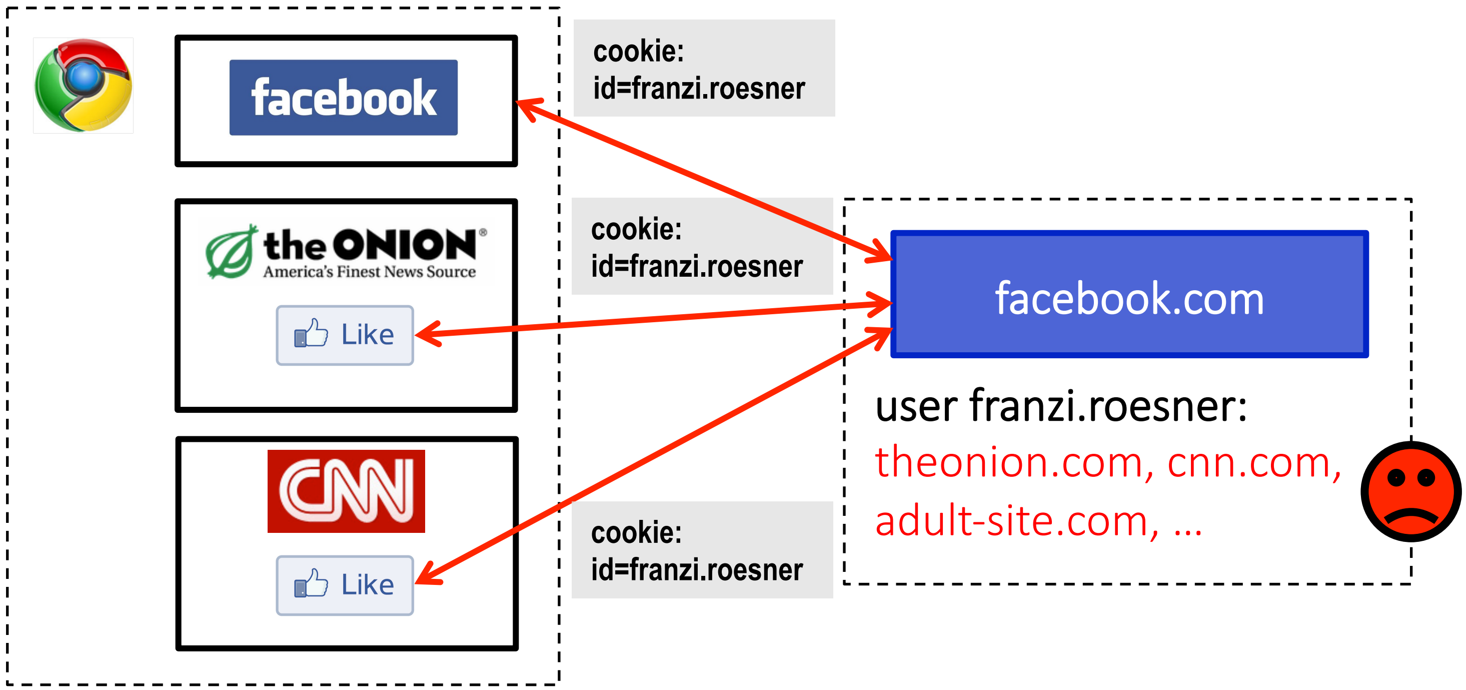
- Within-site vs. cross-site.
- Anonymous vs. non-anonymous.
- Specific behavior types:
  analytics, vanilla, forced, referred, personal.

Other Trackers?



"Personal" Trackers

Personal Tracking



- Tracking is not anonymous (linked to accounts).
- Users directly visit tracker's site → evades some defenses.

Measurement Study (2011)

Questions:

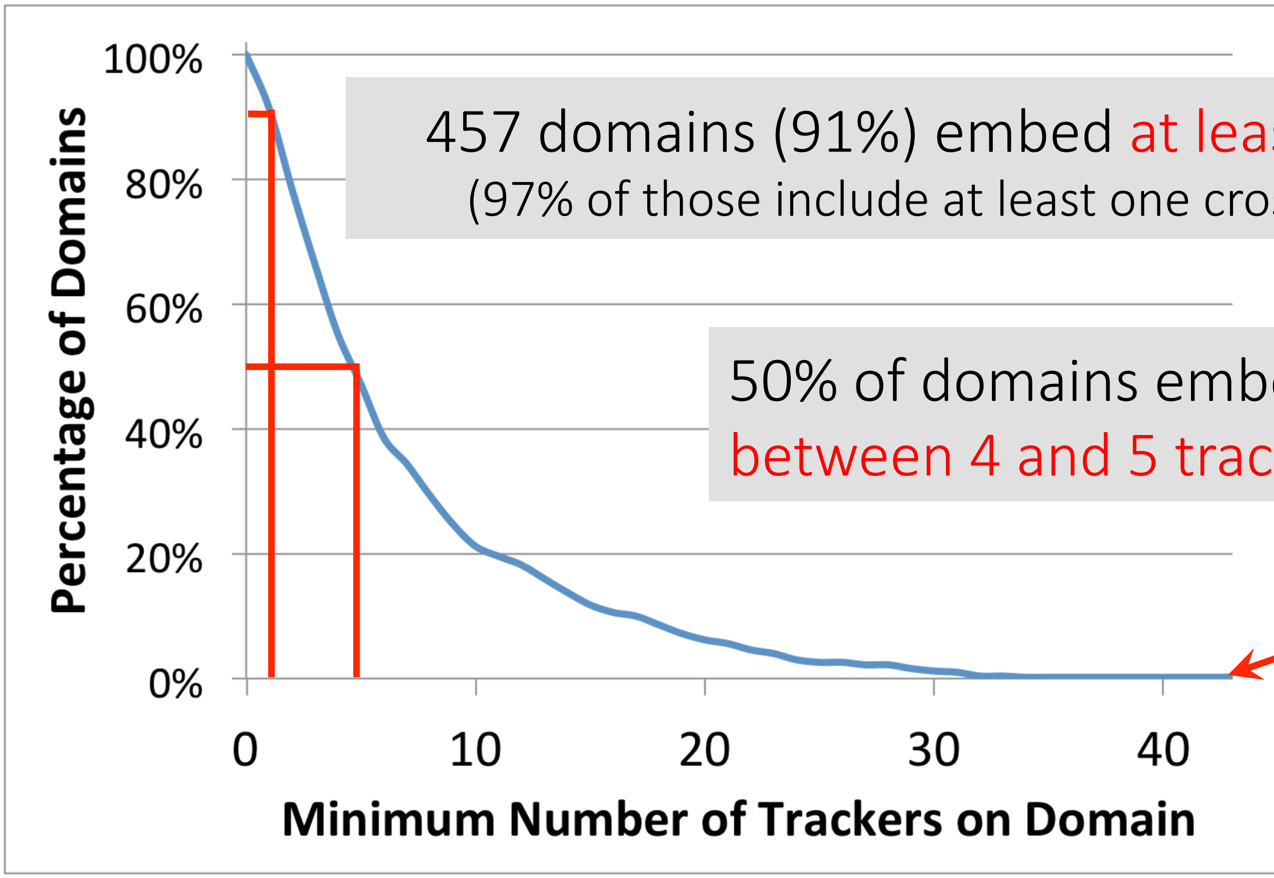
- How prevalent is tracking (of different types)?
- How much of a user's browsing history is captured?
- How effective are defenses?

Approach: Build tool to automatically crawl web, detect and categorize trackers based on our taxonomy.

Our longitudinal study in 2013 showed that the tracking ecosystem has not substantially changed since 2011.

How prevalent is tracking?

524 unique trackers on Alexa top 500 websites (2011).



457 domains (91%) embed at least one tracker.
(97% of those include at least one cross-site tracker.)

50% of domains embed between 4 and 5 trackers.

One domain includes 43 trackers.

How are users affected?

Question: How much of a real user's browsing history can top trackers capture?

Measurement challenges:

- Privacy concerns.
- Users may not browse realistically while monitored.

Insight: AOL search logs (released in 2006) represent real user behaviors.

How are users affected?

Idea: Use AOL search logs to create 30 hypothetical browsing histories.

- 300 unique queries per user → top search hits.

Trackers can capture a large fraction:

- Doubleclick: Avg 39% (Max 66%)
- Facebook: Avg 23% (Max 45%)
- Google: Avg 21% (Max 61%)

How are users affected?

POLICY & LAW | US & WORLD | NATIONAL SECURITY

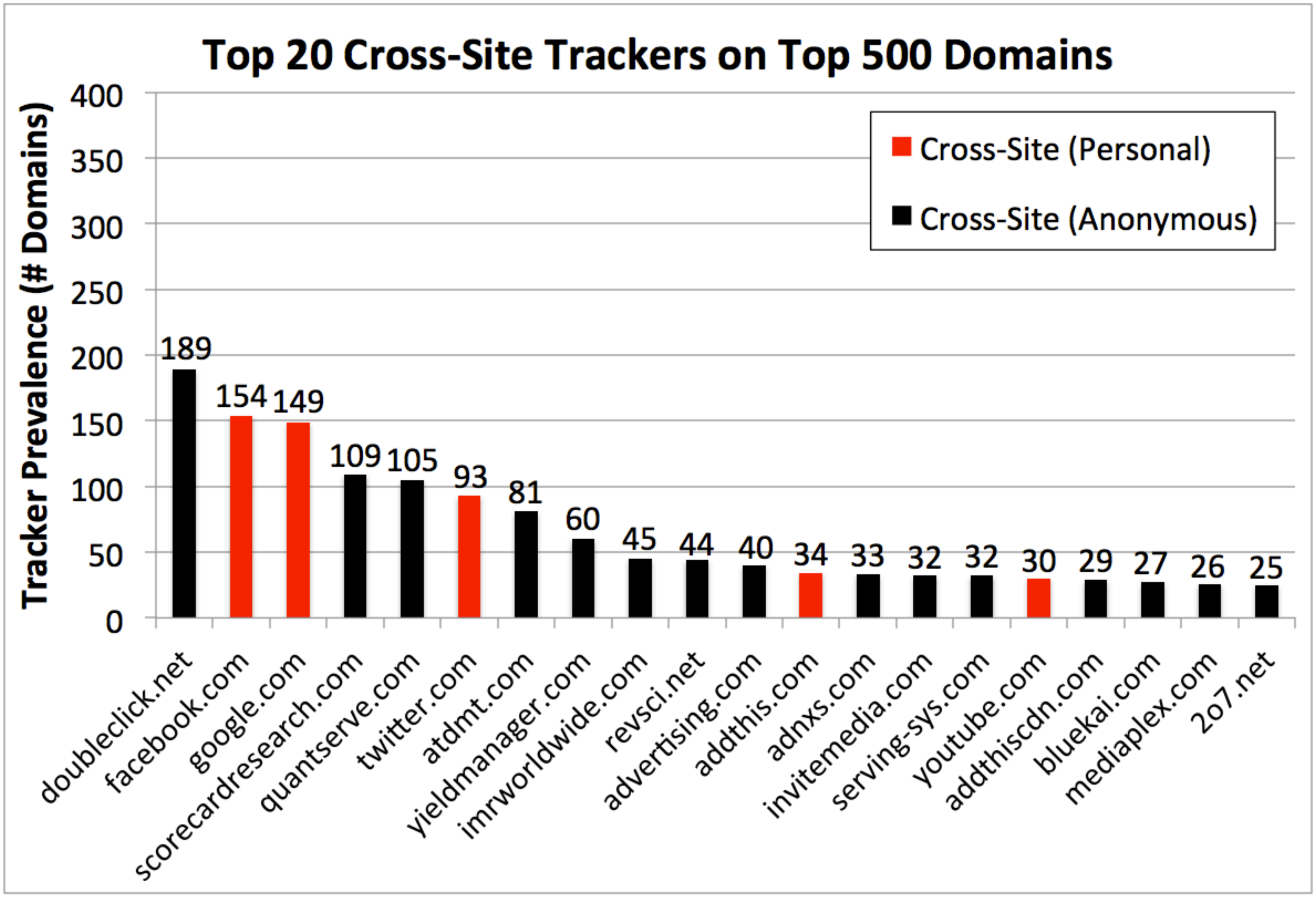
NSA reportedly 'piggybacking' on Google advertising cookies to home in on surveillance targets

By Nathan Ingraham on December 10, 2013 10:41 pm Email @NateIngraham

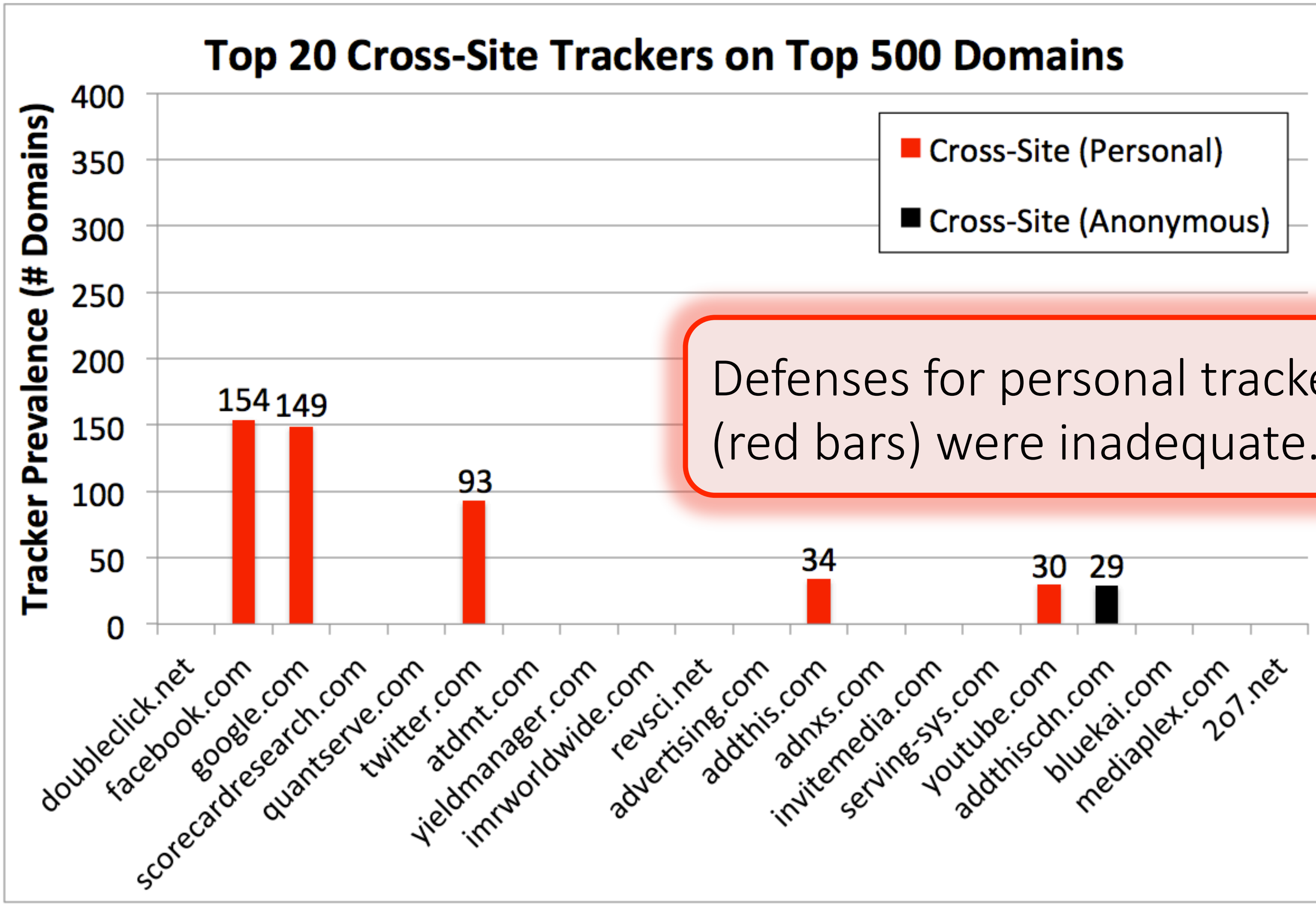
Trackers can capture a large fraction:

- Doubleclick: Avg 39% (Max 66%)
- Facebook: Avg 23% (Max 45%)
- Google: Avg 21% (Max 61%)

Who/what are the top trackers? (2011)



Who/what are the top trackers? (2011)



Defenses for personal trackers (red bars) were inadequate.

Defense: ShareMeNot

Prior defenses for personal trackers: ineffective or completely removed social media buttons.

Our defense:

- ShareMeNot (for Chrome/Firefox) protects against tracking without compromising button functionality.
- Blocks requests to load buttons, replaces with local versions. On click, shares to social media as expected.
- Techniques adopted by Ghostery & PrivacyBadger (EFF).

http://sharemenot.cs.washington.edu

Summary: Web Tracking

Pre-2011: Limited understanding of web tracking.

Our work:

- Comprehensive tracking taxonomy.
- Example results: >500 unique trackers, some able to capture up to 66% of a user's browsing history.
- New defense for "personal trackers" like Facebook, Google, Twitter: built into ShareMeNot, adopted by Ghostery and the EFF's PrivacyBadger.

Outline

I. Browsers:
   Third-Party Tracking

II. Smartphones:
   Permission Granting

III. Security & Privacy in Other Contexts

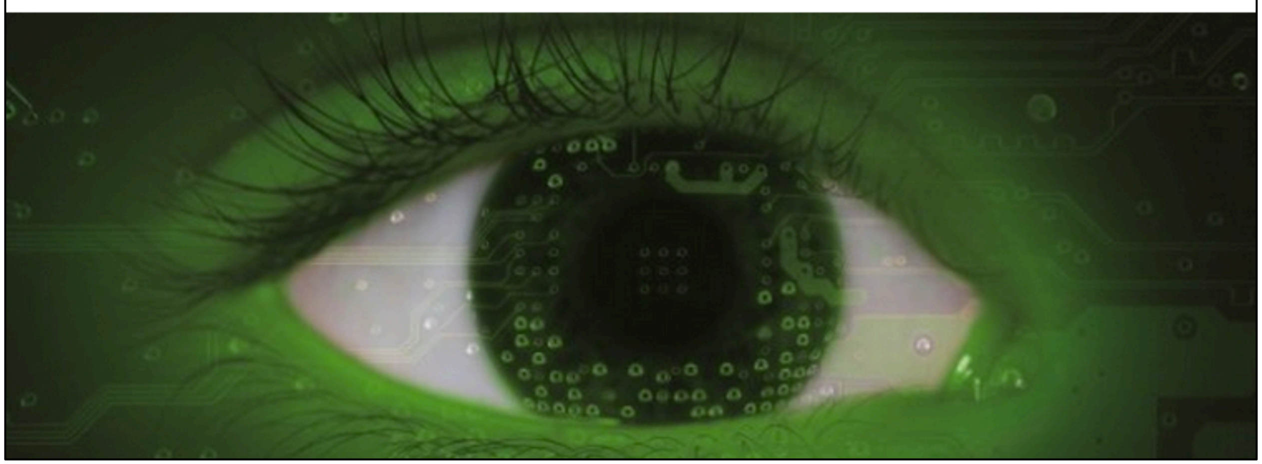
F. Roesner, T. Kohno, A. Moshchuk, B. Parno, H. J. Wang, C. Cowan. "User-Driven Access Control: Rethinking Permission Granting in Modern Operating Systems." In IEEE Symposium on Security & Privacy 2012 (Best Practical Paper Award).

Smartphone (In)Security

Users accidentally install malicious applications.

Over 60% of Android malware steals your money via premium SMS, hides in fake forms of popular apps

By Emil Protalinski, Friday, 5 Oct '12 , 05:50pm



Smartphone (In)Security

Users accidentally install malicious applications.

Even legitimate applications exhibit questionable behavior.


Top Mobile Apps Overwhelmingly Leak Private Data: Study

By Robert Lemos | Posted 2013-07-31 Email Print

according to a survey of the top-400 mobile applications conducted by application-analysis firm Appthority.

Hornyack et al.: 43 of 110 Android applications sent location or phone ID to third-party advertising/analytics servers.

Permission Granting Problem

Smartphones (and other modern OSes) try to prevent such attacks by limiting applications' access to:

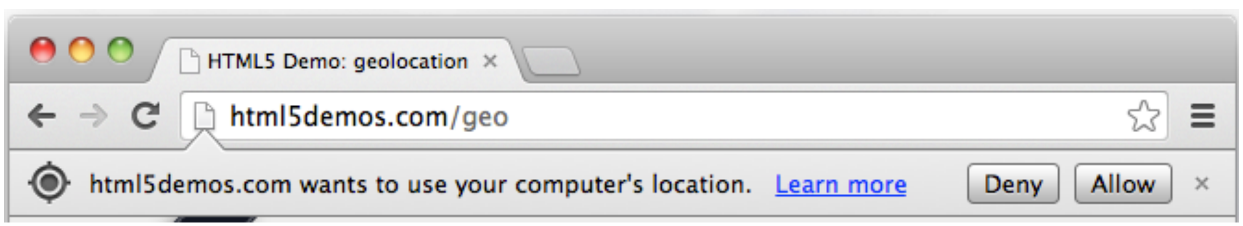
- System Resources (clipboard, file system).
- Devices (camera, GPS, phone, …).

How should operating system grant permissions to applications?

Standard approach: Ask the user.

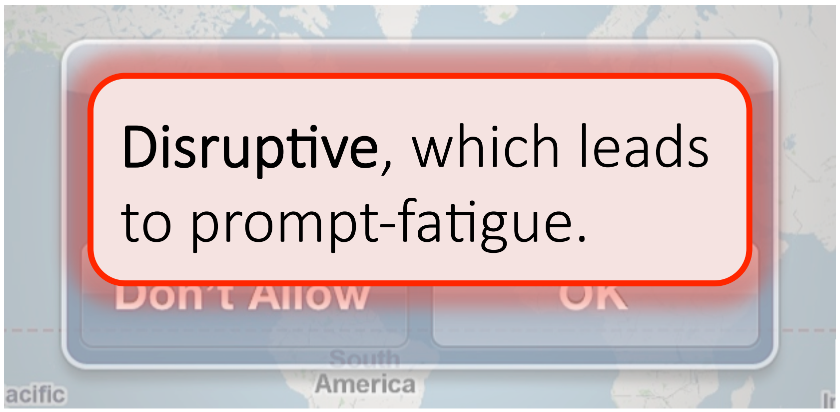
State of the Art

Prompts (time-of-use)

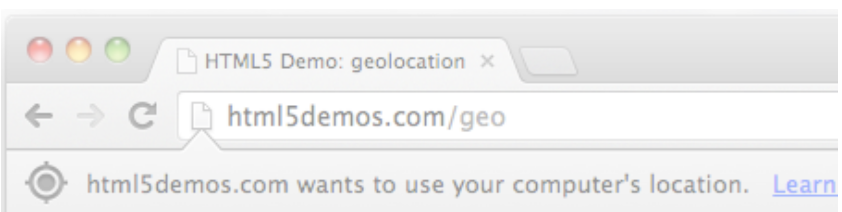


State of the Art

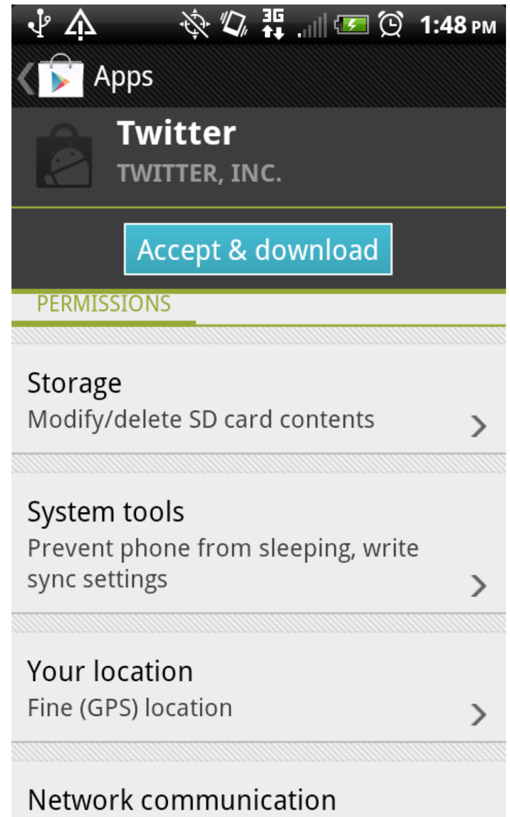
Prompts (time-of-use)



Disruptive, which leads to prompt-fatigue.

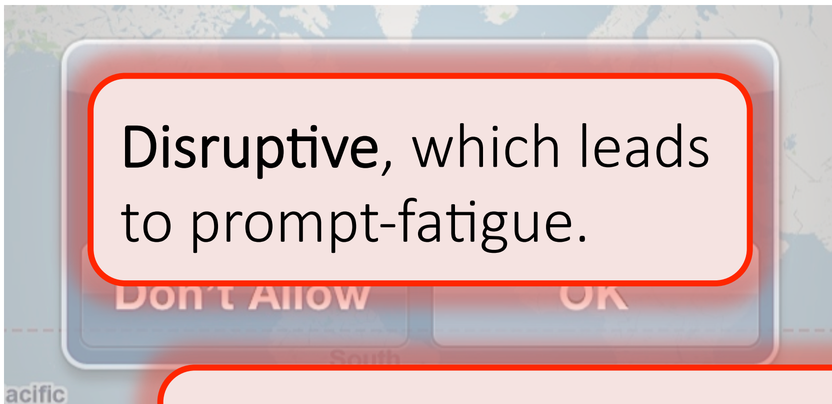


Manifests (install-time)

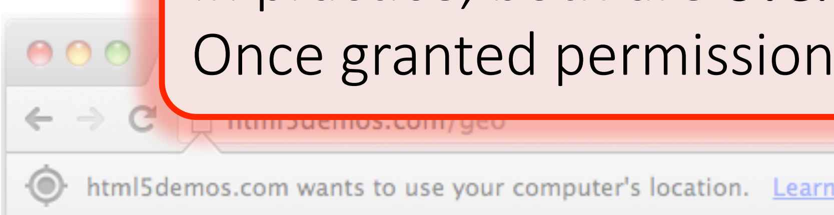


State of the Art

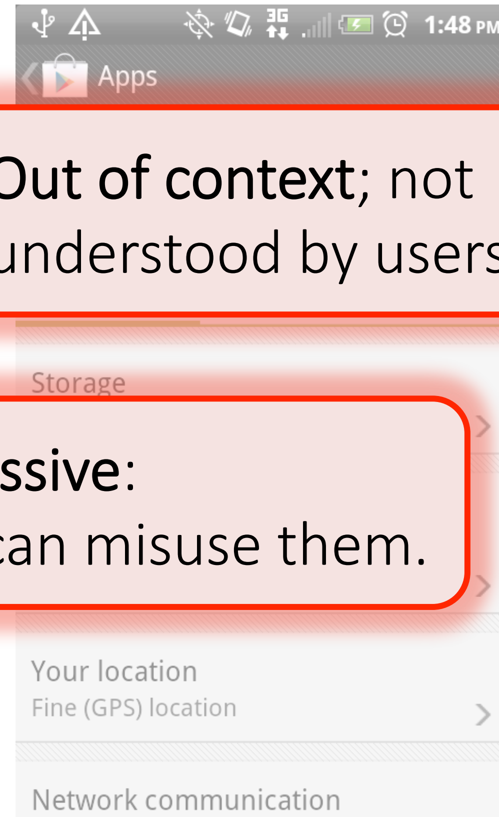
Prompts (time-of-use)



Disruptive, which leads to prompt-fatigue.



Manifests (install-time)



Out of context; not understood by users.

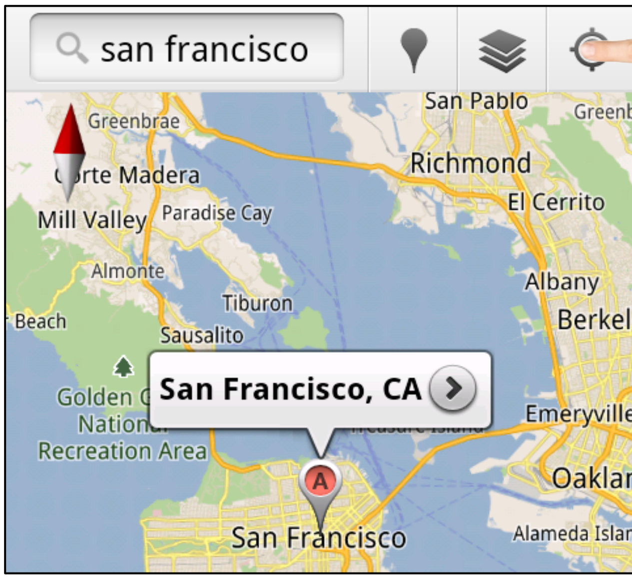
In practice, both are **overly permissive**:
Once granted permissions, apps can misuse them.

Goals for Permission Granting

1. **Least-Privilege:** Applications should receive the minimum necessary access.
2. **Usable:**
   - Not disruptive to users.
   - Matches user expectations.
   - Doesn't require constant comprehension/management.

   ("magically" grants exactly those permissions expected by the user)
3. **Generalizable:** Easily extended to new resources.

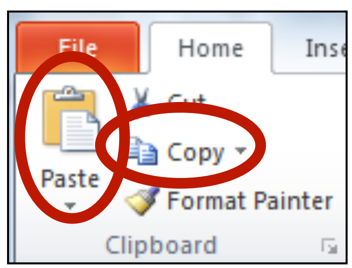
Our Work: User-Driven Access Control

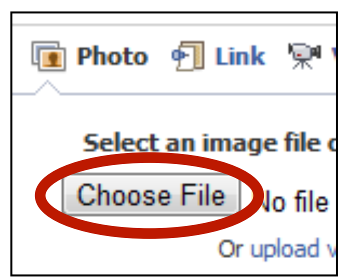


Let this application access my location **now**.

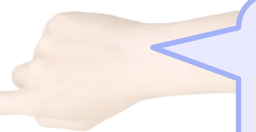
Insight:

A user's natural UI actions within an application implicitly carry permission-granting semantics.





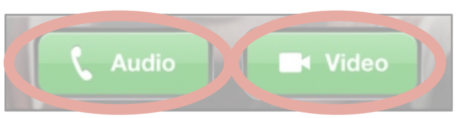
Our Work: User-Driven Access Control

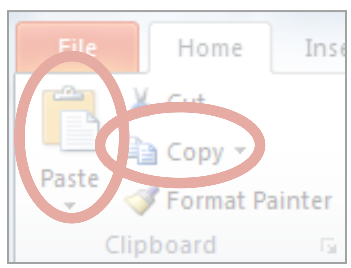


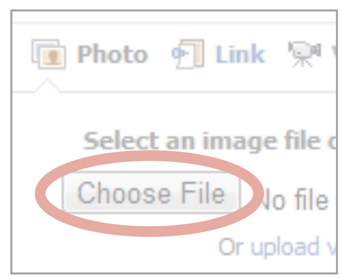
Let this application access my location **now**.

Our study shows:
Many users already believe (52% of 186) – and/or desire (68%) – that resource access follows the user-driven access control model.

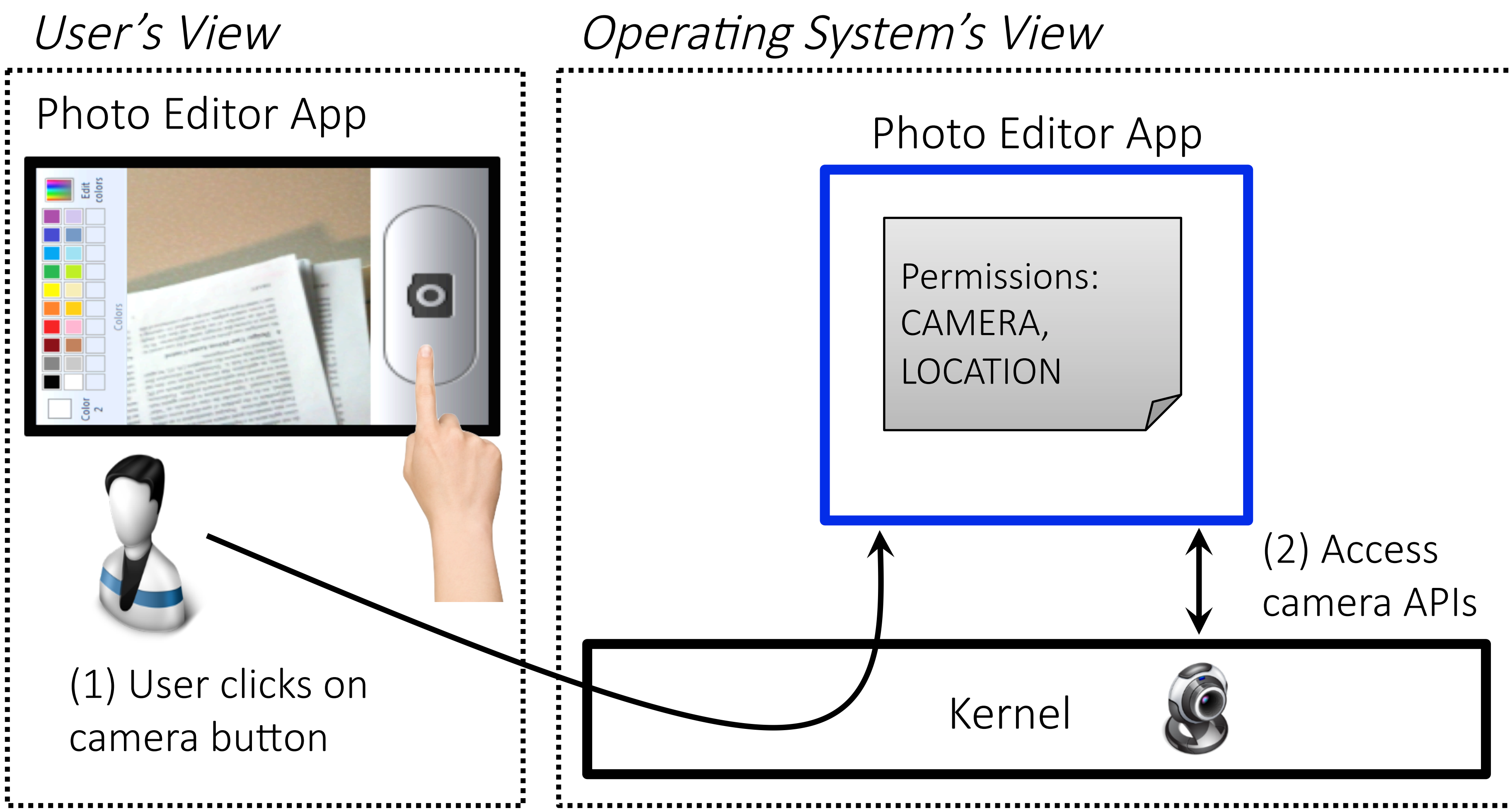
...s within
...carry
...antics.







Resource-Related UIs Today



Resource-Related UIs Today

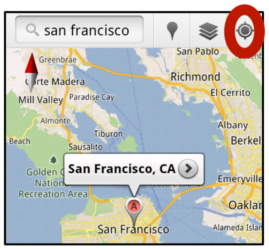
User's View *Operating System's View*

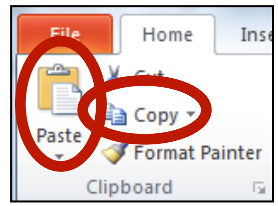
Problem: OS can't understand user's interaction with application → can't link permission use to user intent.

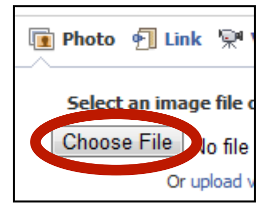
Challenge:
Can the system extract access control decisions from user actions in a **general, application-agnostic way**?

Prior approaches are hard to generalize:
EWS [SVNC '04], NitPicker [FH '05], CapDesk [M '06], Qubes, Polaris [SKYCM '06], UIBAC [SE '08], BLADE [LYPL '10]

New OS Primitive:
Access Control Gadgets (ACGs)





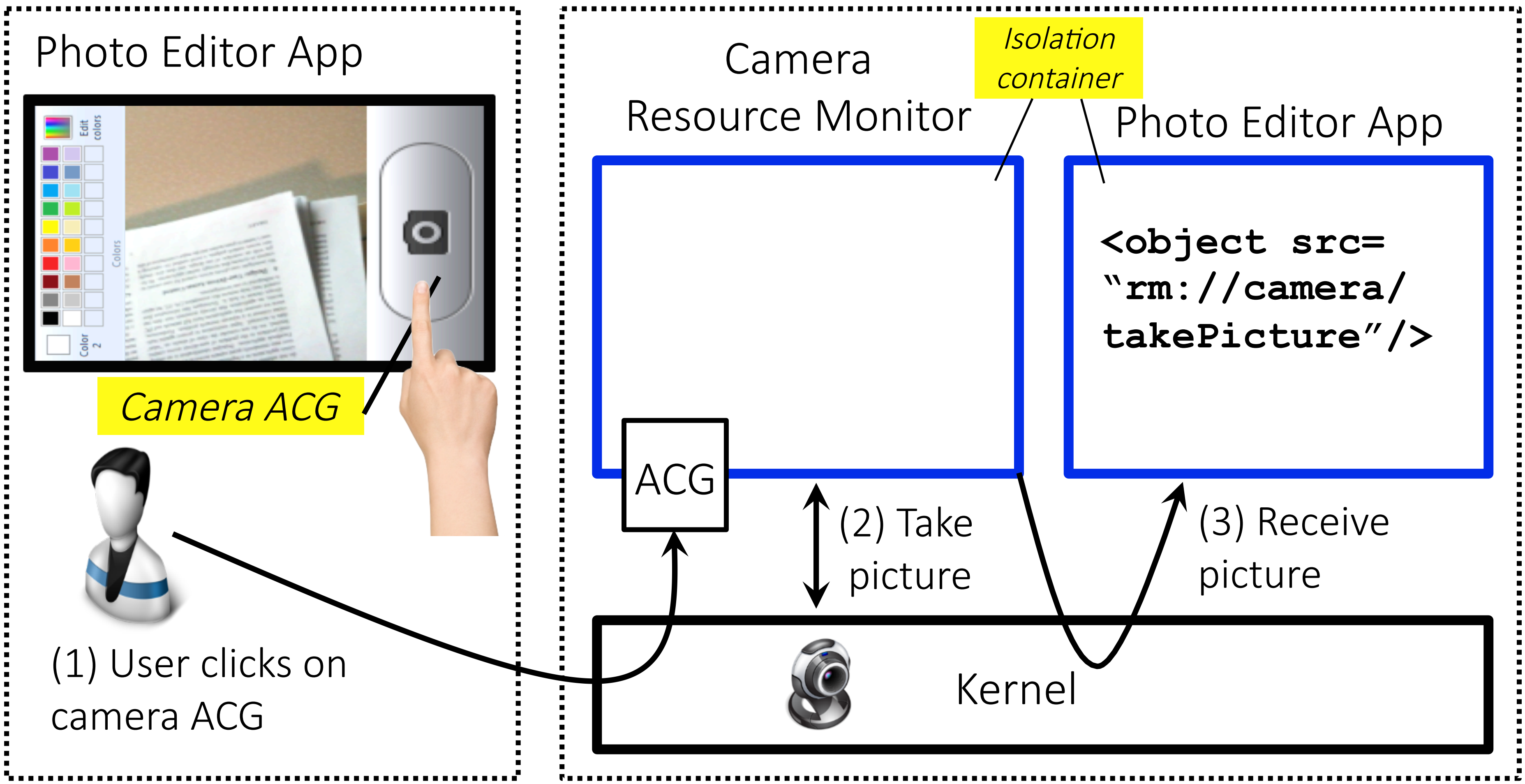


Approach: Make resource-related UI elements first-class operating system objects (access control gadgets).

- To receive resource access, applications must embed a system-provided ACG.
- ACGs allow the OS to capture the user's permission granting intent in application-agnostic way.

Access Control Gadgets (ACGs) in Action

User's View *Operating System's View*



Challenges with ACGs

Impact on applications:

- What about application customization?
- How to design system/resource APIs to support necessary application functionality?

Attacks on ACGs by malicious applications:

- How can system be sure that the user intent it captures is authentic?

Attacks on Access Control Gadgets

Malicious applications want to gain access without authentic user intent.

Example: Clickjacking attack.

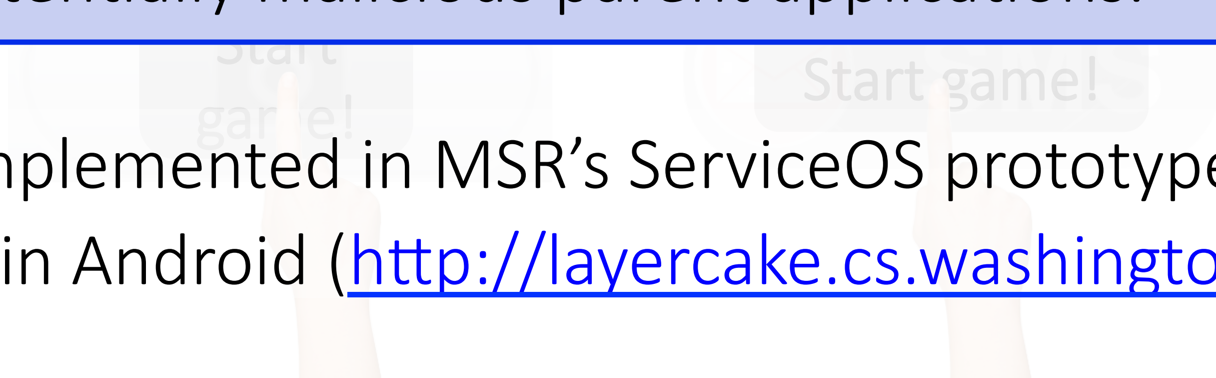
Trick users into clicking on ACG by making it transparent.

Attacks on Access Control Gadgets

Malicious applications want to gain access without authentic user intent.

Example: Clickjacking attack.

The operating system must protect ACGs from potentially malicious parent applications.



First implemented in MSR's ServiceOS prototype system, later in Android (http://layercake.cs.washington.edu).

Evaluation Highlights

User-driven access control matches user expectations.

Many users already believe (52% of 186) – and/or desire (68%) – that resource access follows the UDAC model.

User-driven access control improves security.

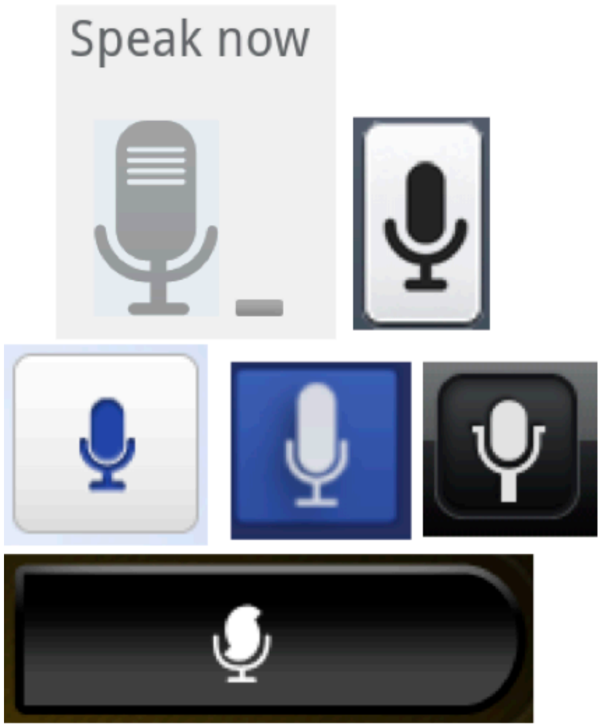
Addresses most published vulnerabilities related to resource access: 36 of 44 in Chrome (82%), 25 of 26 in Firefox (96%).

ACGs have minimal impact on user interface.

73% of top Android apps need only limited customization for resource-related UIs.

Evaluation Highlights

Limited Customization



Arbitrary Customization

73% of top Android apps need only limited customization for resource-related UIs.

Summary: Permission Granting

Prior approaches grant too much access, are too disruptive, or are not understood by users.

Our approach: user driven access control.

- OS extracts permissions from user actions.
- Enabled by new OS primitive: access control gadgets (must protect from malicious apps).
- Application-agnostic, improves security, and matches user expectations.

Outline

I. Browsers:
   Third-Party Tracking

II. Smartphones:
   Permission Granting

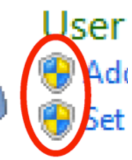
III. Security & Privacy in Other Contexts

My Research

franzi@cs.washington.edu


Analyze existing systems:
The Web *[NSDI '12]*, Automobiles *[IEEE S&P '10, USENIX Security '11]*.





Build new systems:
The Web, Smartphones *[IEEE S&P '12]*, UI Toolkits *[UIST '12, USENIX Security '13]*.

Anticipate future technologies:
Wearables, Augmented reality *[HotOS '13, CACM '14, CCS '14]*.

Understand mental models:
Permissions, Journalists, Snapchat *[FC '14]*.