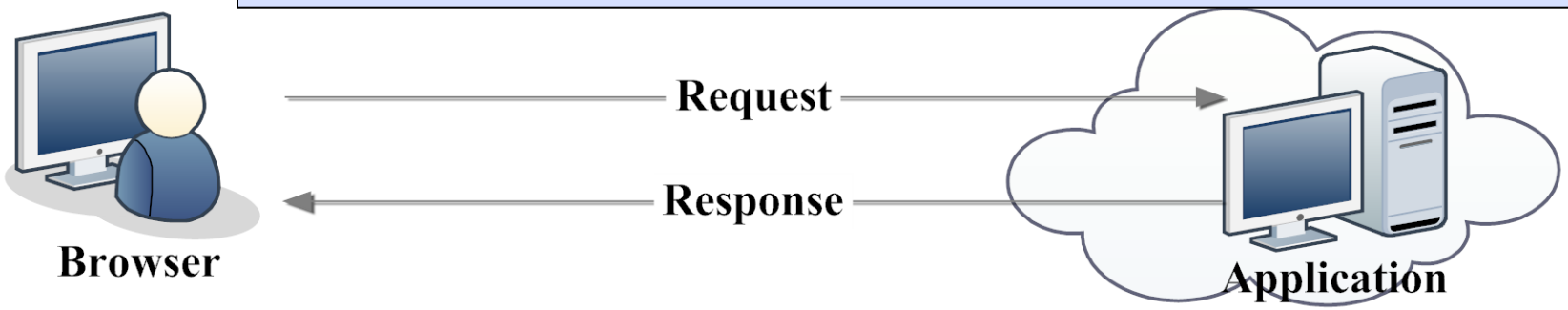
CS 6431

Security Issues in Web Applications

Vitaly Shmatikov

User Input Validation

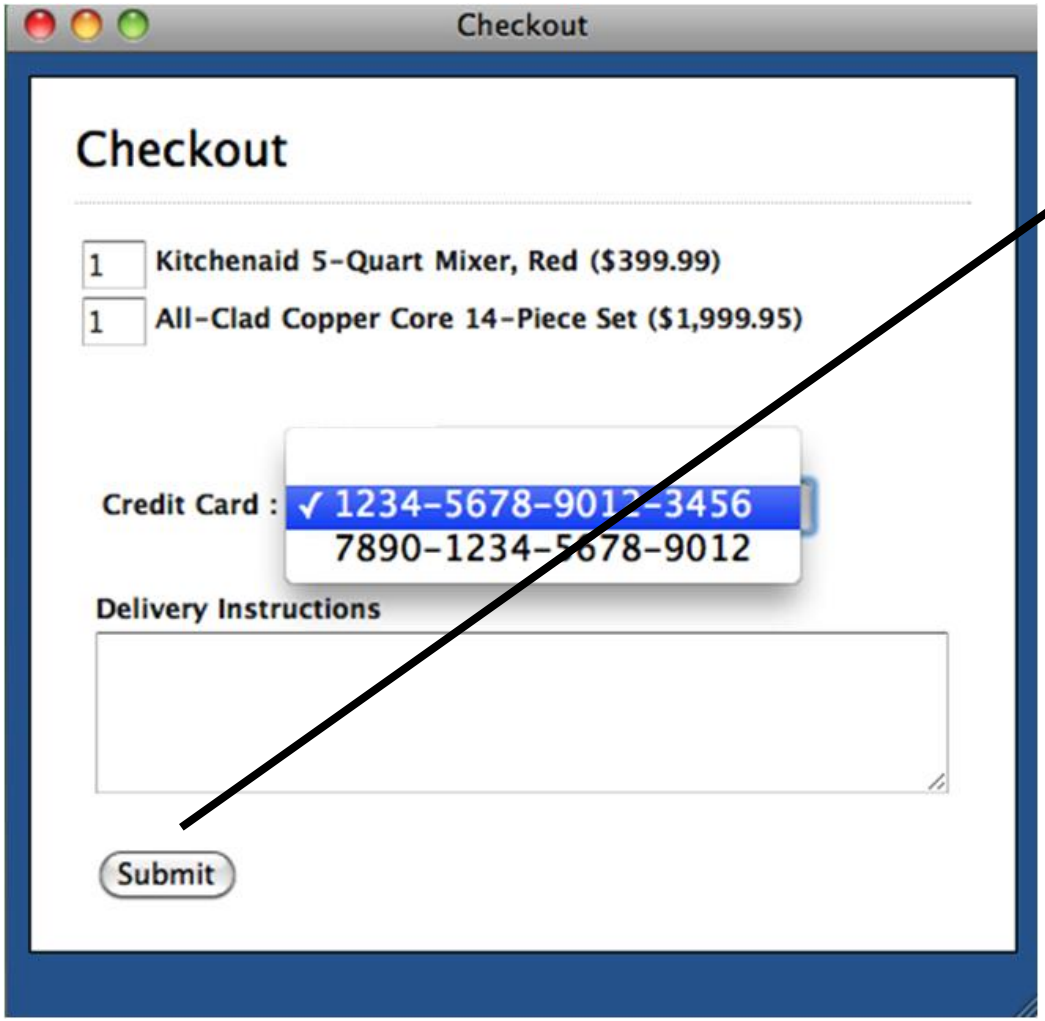
[Bisht et al. "NoTamper: Automatic Blackbox Detection of Parameter Tampering Opportunities in Web Applications". CCS 2010]



- Web applications need to reject invalid inputs
  - "Credit card number should be 15 or 16 digits"
  - "Expiration date in the past is not valid"
- Traditionally done at the server
  - Round-trip communication, increased load
- Better idea (?): do it in the browser using client-side JavaScript code

Client-Side Validation

[Bisht et al.]




```
onSubmit=
  validateCard();
  validateQuantities();
```

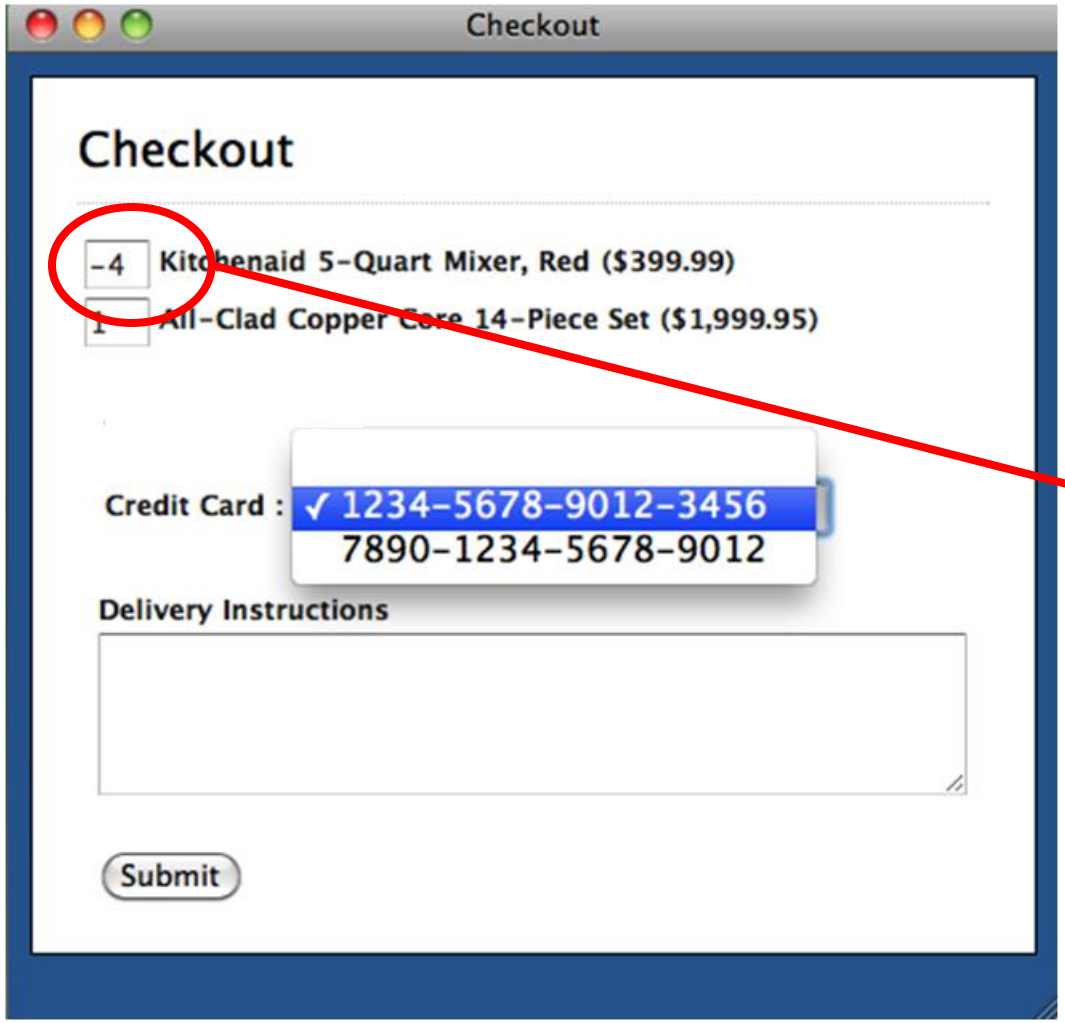
Validation Ok?

Yes → send inputs to server

No → reject inputs

Problem: Client Is Untrusted

[Bisht et al.]

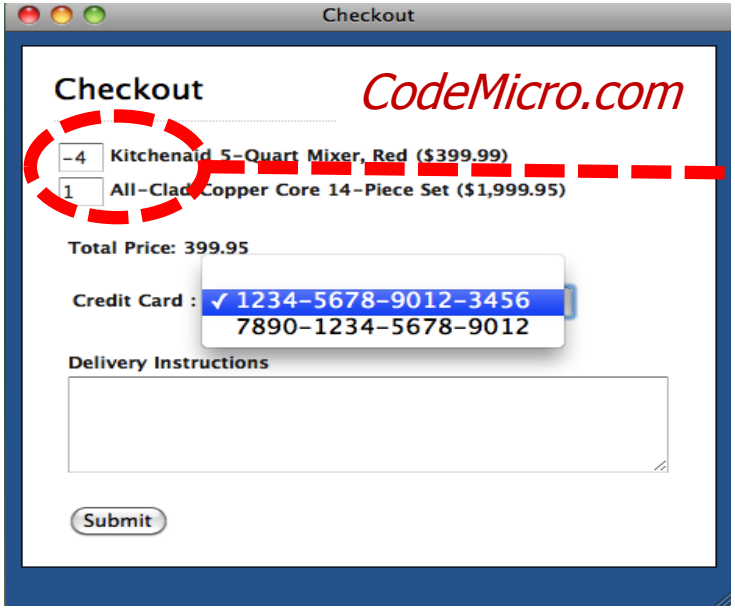


Previously rejected values sent to server

Inputs must be re-validated at server!

Online Shopping

[Bisht et al.]



Client-side constraints:

quantity1 ≥ 0

quantity2 ≥ 0

Server-side code:

total = quantity1 * price1 +
quantity2 * price2

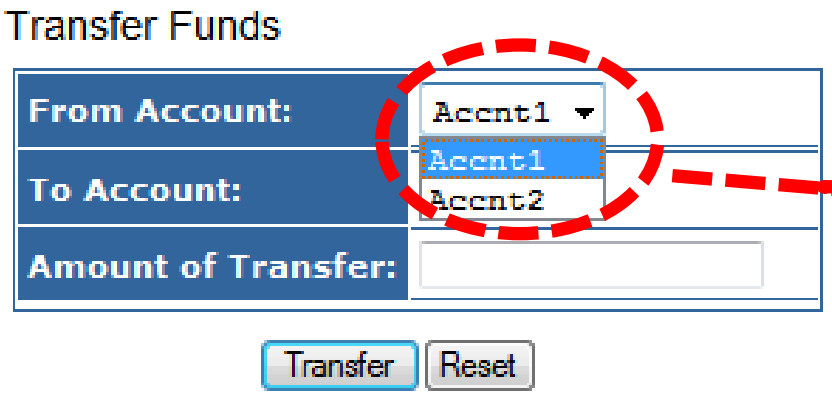
Vulnerability: malicious client submits negative quantities for unlimited shopping rebates

Two items in cart: price1 = $100, price2 = $500

quantity1 = -4, quantity2 = 1, total = $100 (rebate of $400 on price2)

Online Banking

[Bisht et al.]



Client-side constraints:

from IN (Accnt1, Accnt2)

to IN (Accnt1, Accnt2)

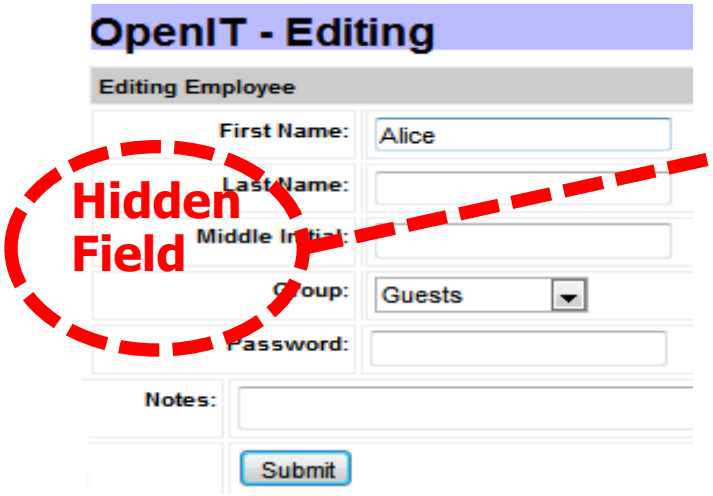
Server-side code:

transfer money from → to

Vulnerability: malicious client submits arbitrary account numbers for unauthorized money transfers

IT Support

[Bisht et al.]



Client-side constraints:

userId == 96 (hidden field)

Server-side code:

Update profile with id 96 with new details

Vulnerability: update arbitrary account

Inject a cross-site scripting (XSS) payload in admin account, cookies stolen every time admin logged in

Content Management

[Bisht et al.]

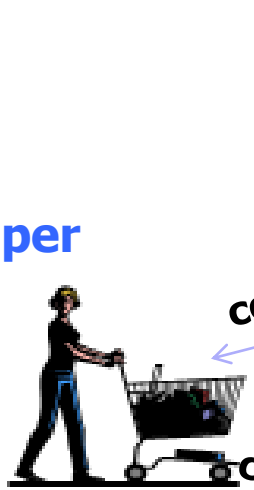
Server-side code:

```
privilege = non-admin;
if ( _COOKIE[‘make_install_prn’]
   == 1 )
        privilege = admin;
```

Vulnerability: malicious client sets make_install_prn cookie, creates fake admin account

Cashier-as-a-Service

[Wang et al. "How to Shop for Free Online: Security Analysis of Cashier-as-a-Service Based Web Stores". Oakland 2011]



Shopper

Web store

communication about the order

communication about the payment

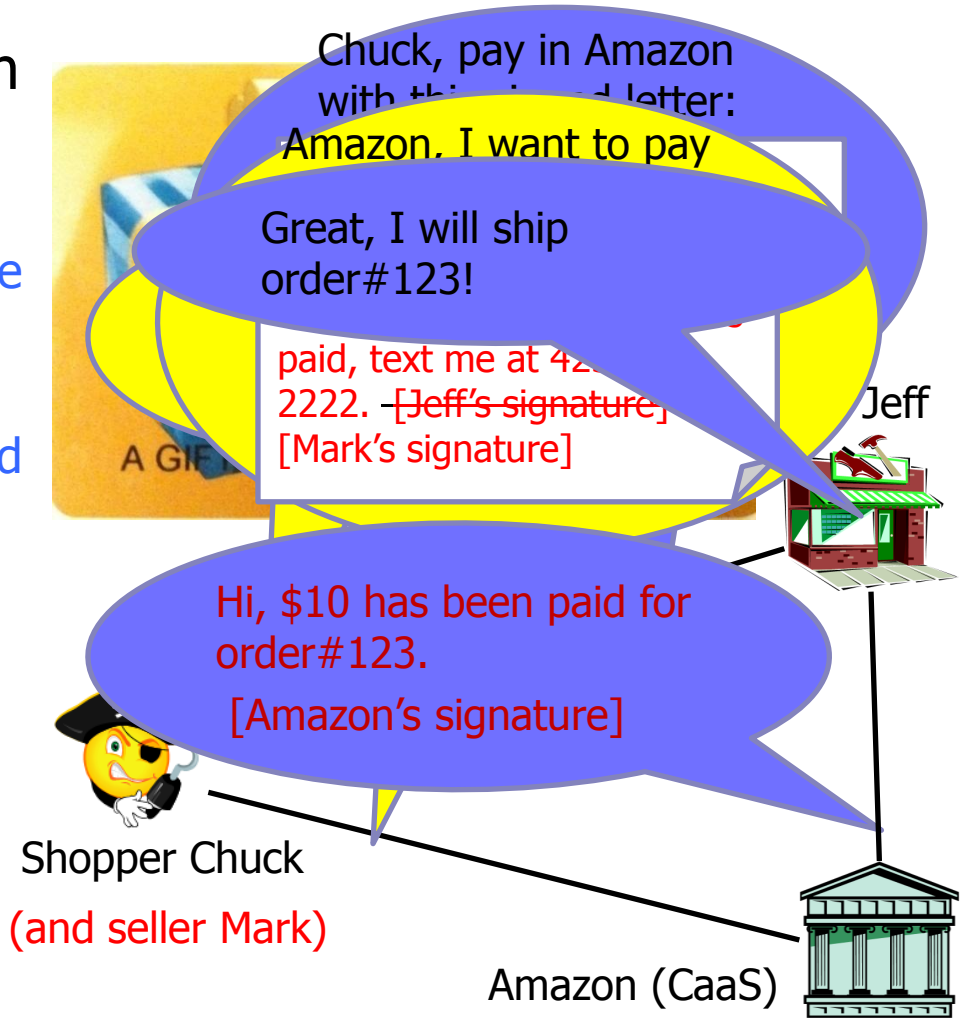
Joint decision:
is an order
appropriately paid?

PayPal, Amazon Payments,
Google Checkout, etc.

nopCommerce + Amazon Simple Pay

[Wang et al.]

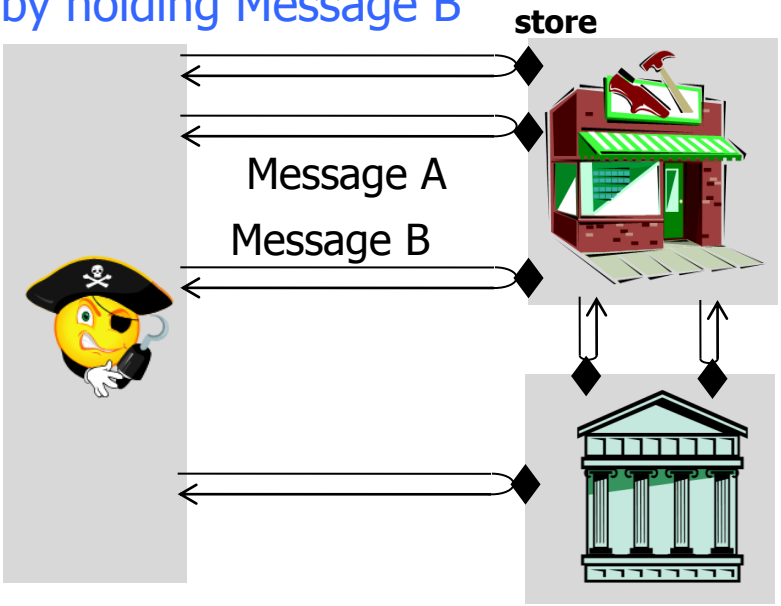
- Anyone can register an Amazon seller account, so can Chuck
  - Purchase a $25 MasterCard gift card by cash, register under a fake address and phone number
  - Create seller accounts in PayPal, Amazon and Google using the card
- Chuck's trick
  - Check out from Jeff, but pay to "Mark" (Chuck himself)
  - Amazon tells Jeff that payment has been successful
  - Jeff is confused, ships product



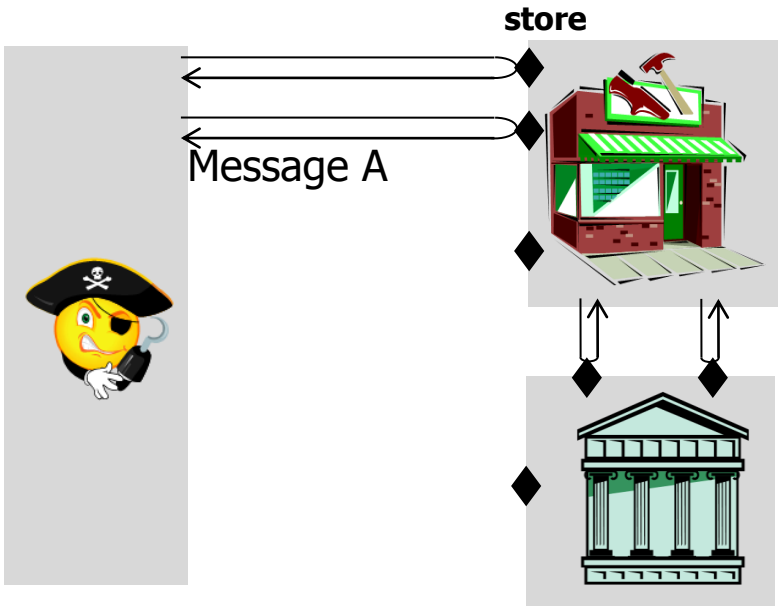
Interspire + PayPal Express

[Wang et al.]

Session 1: pay for a cheap order (orderID1), but prevent the merchant from finalizing it by holding Message B



Session 2: place an expensive order (orderID2) , but skip the payment step



Message A redirects to store.com/finalizeOrder?[orderID1]$_{store}$

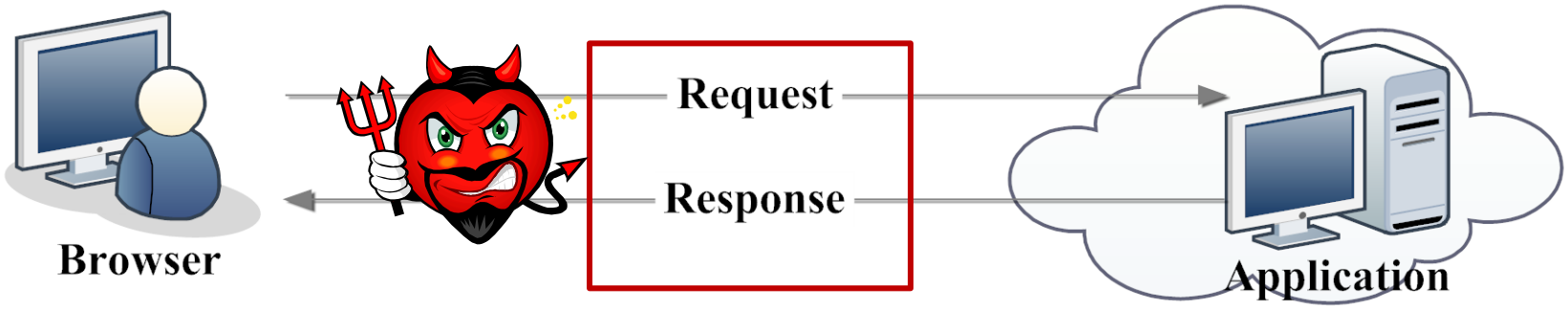
Message A redirects to store.com/finalizeOrder?[orderID2]$_{store}$

Message B calls store.com/finalizeOrder?[orderID1]$_{store}$ [orderID2]$_{store}$

Expensive order is checked out but the cheap one is paid!

Side-Channel Leaks

[Chen et al. "Side-Channel Leaks in Web Applications: a Reality Today, a Challenge Tomorrow". Oakland 2010]



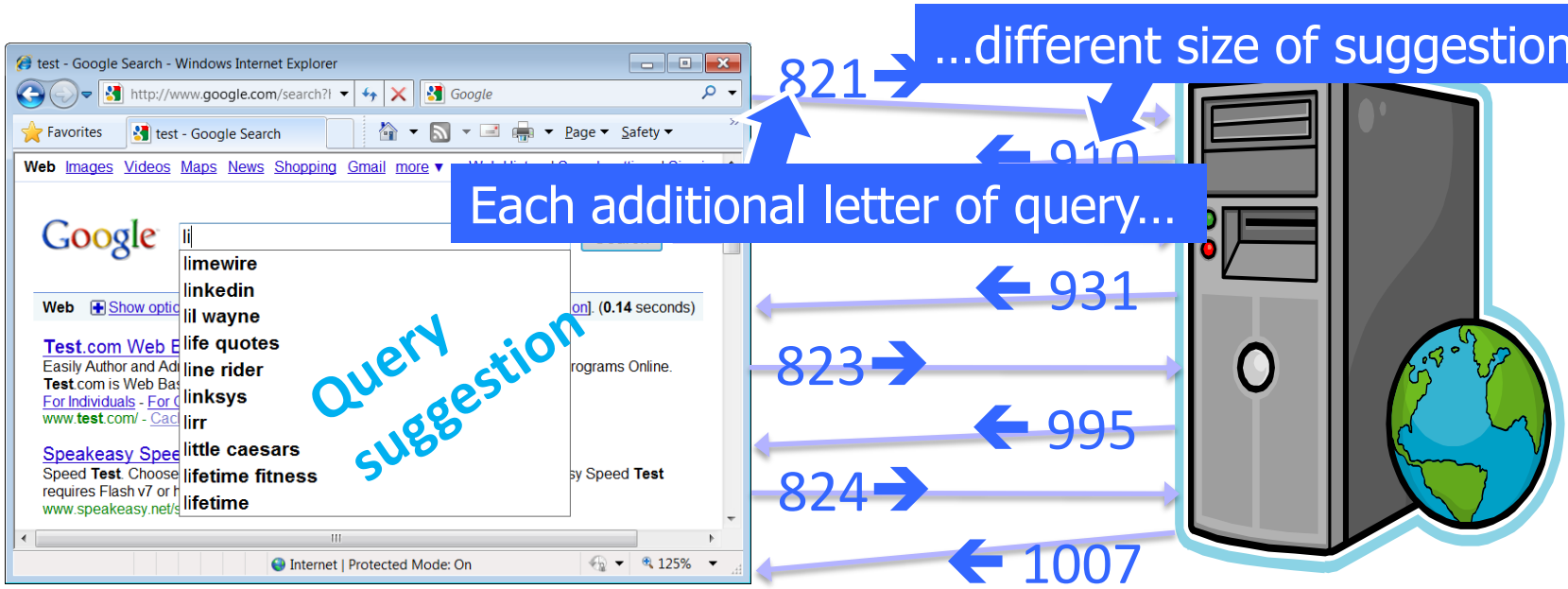
encrypted!

privacy problems solved?

Attacker can still see the number of packets, size of each packet, time between packets...

[Chen et al.]

- Search using encrypted Wi-Fi (WPA / WPA2)
- Example: user types "l-i-s-t" on his laptop…



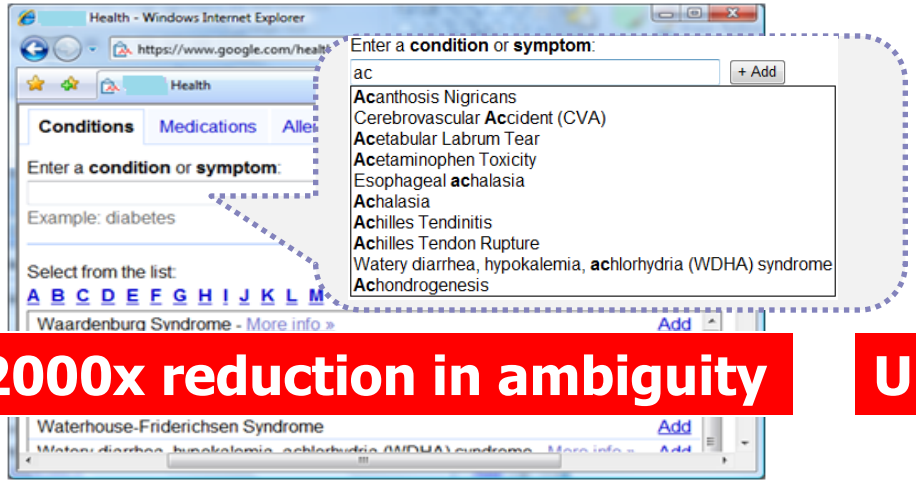
Attacker's effort linear in the size of query

Consequence: any eavesdropper knows our search queries

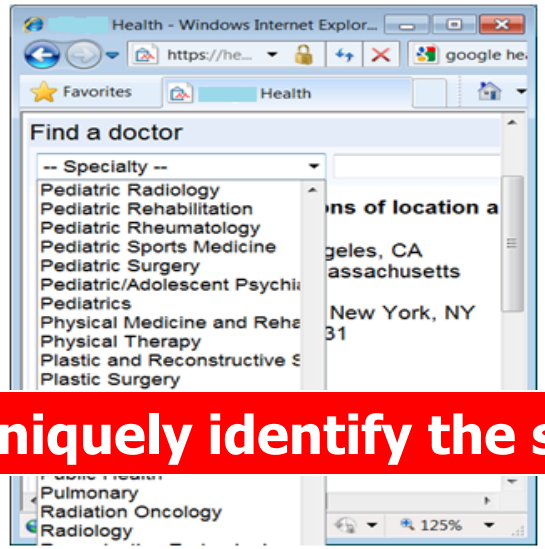
Online Medical Application

[Chen et al.]

- Entering health records
  - By typing – auto-suggestion
  - By mouse – a tree structure of elements
- Finding a doctor
  - Dropdown list



2000x reduction in ambiguity



Uniquely identify the specialty

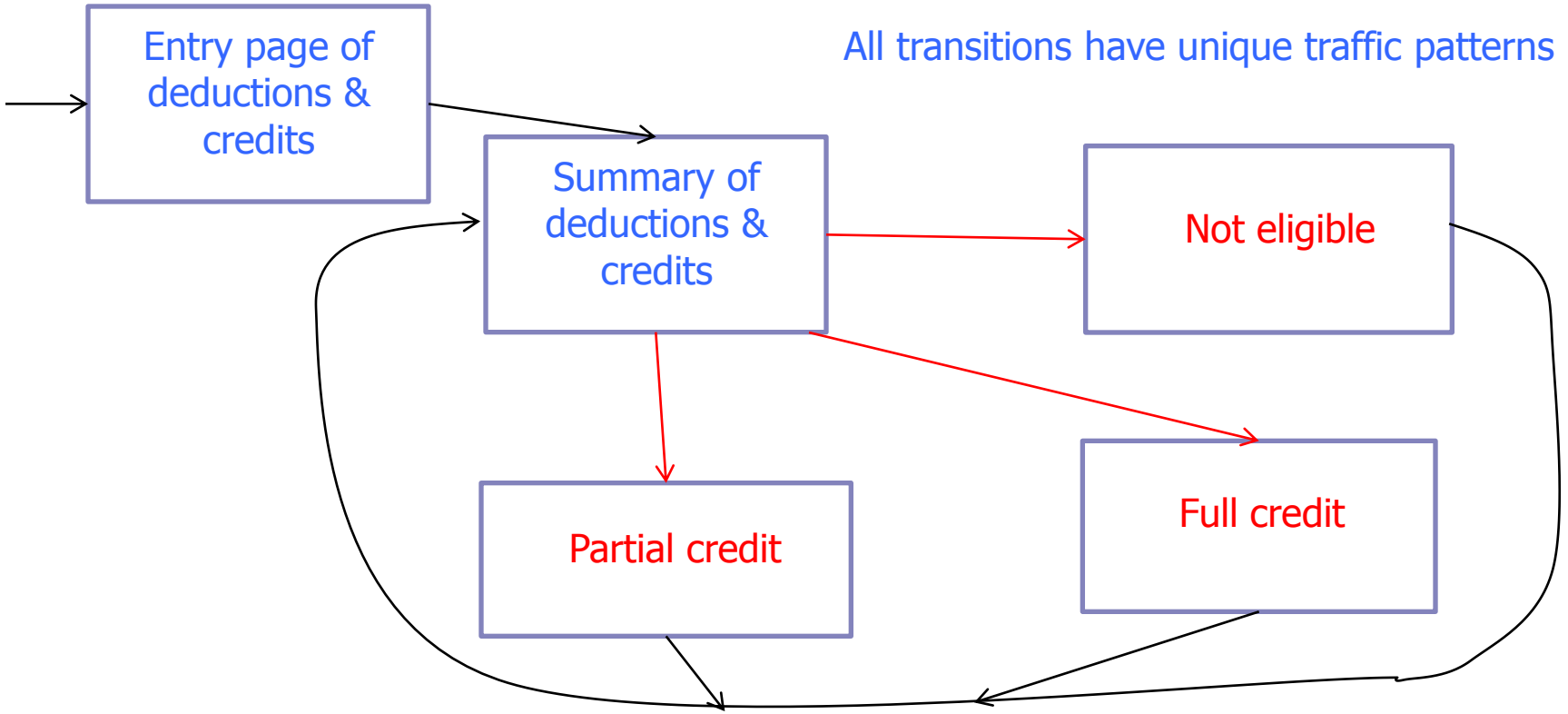
Tax Preparation Application

[Chen et al.]

- Wizard-style questionnaire
  - Tailor the questions based on previous inputs
- Which forms you work on reveal filing status, big medical bills, adjusted gross income…
- Knowing the state machine of the application the eavesdropper can infer sensitive information
  - Especially by combining information learned from multiple state machines

Child Credit State Machine

[Chen et al.]



Consult the IRS instruction:

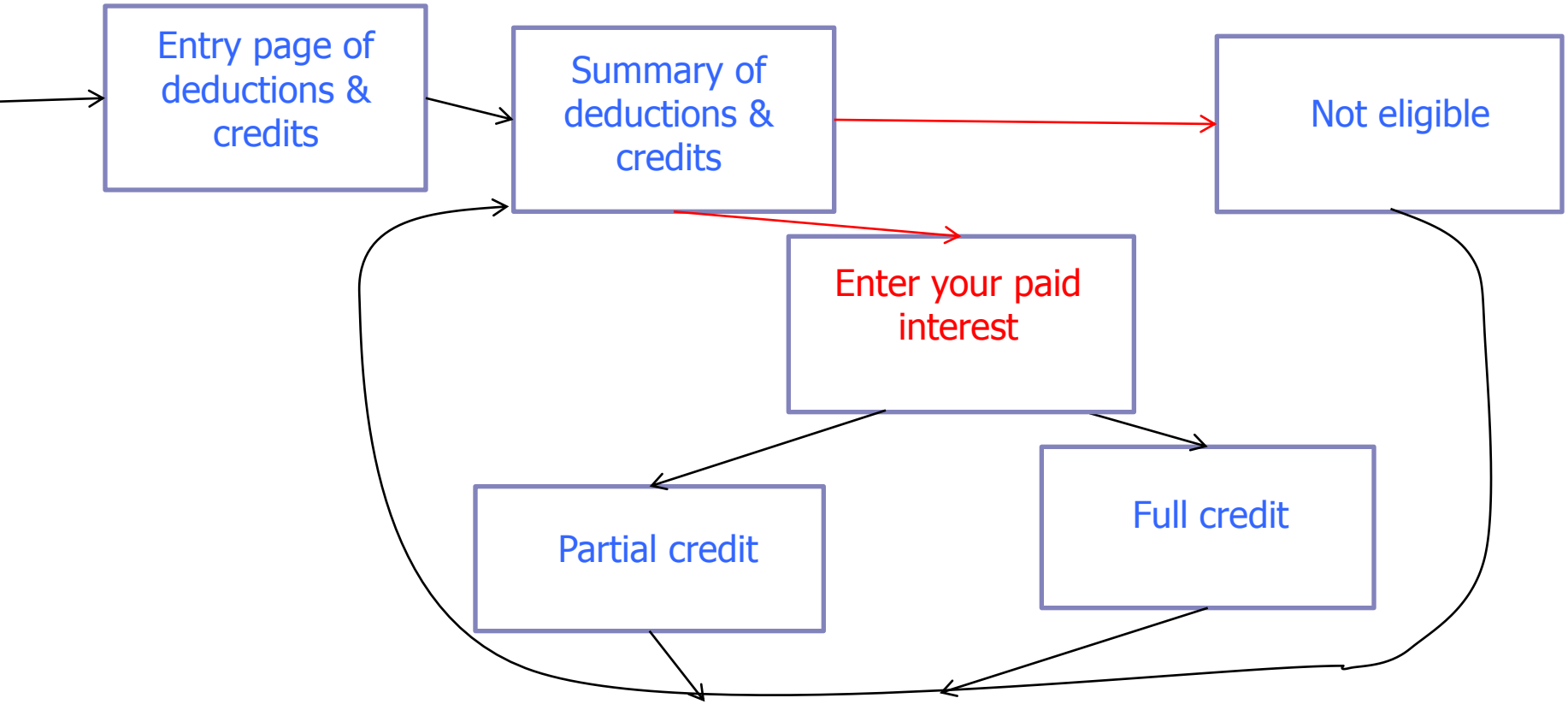
$1000 for each child

Phase-out starting from $110,000. For every $1000 income, lose $50 credit.

Student Loan Interest State Machine

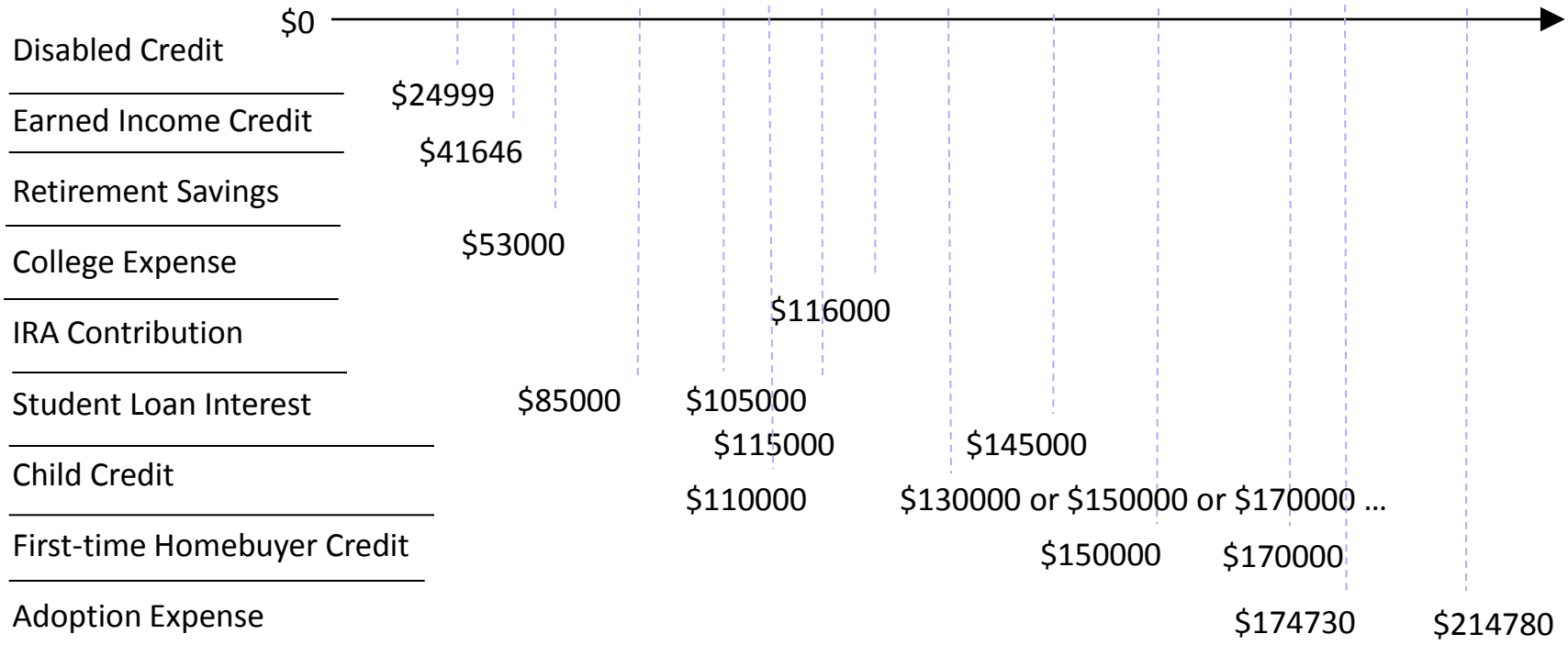
[Chen et al.]

Even worse, most decision procedures for credits/deductions have asymmetric paths: eligible – more questions, not eligible – no more questions



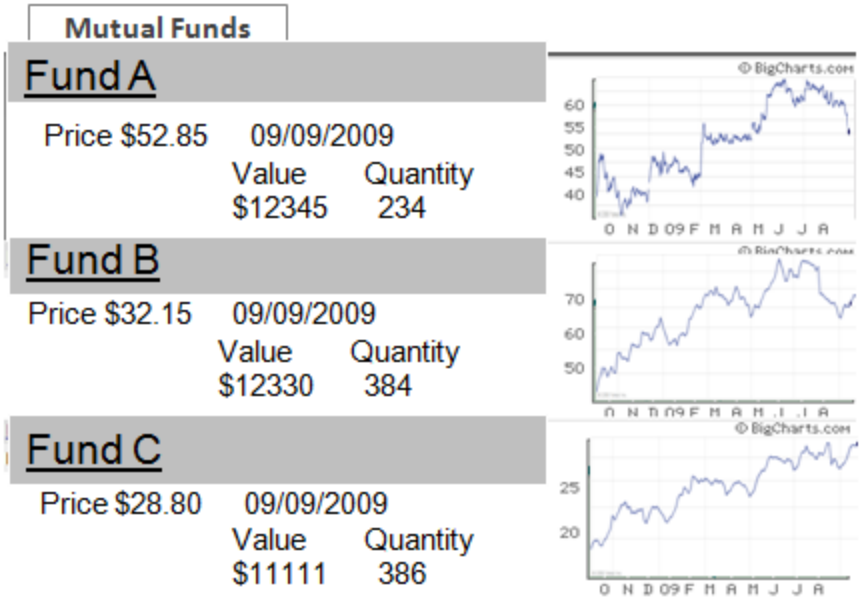
Some Identifiable AGI Thresholds

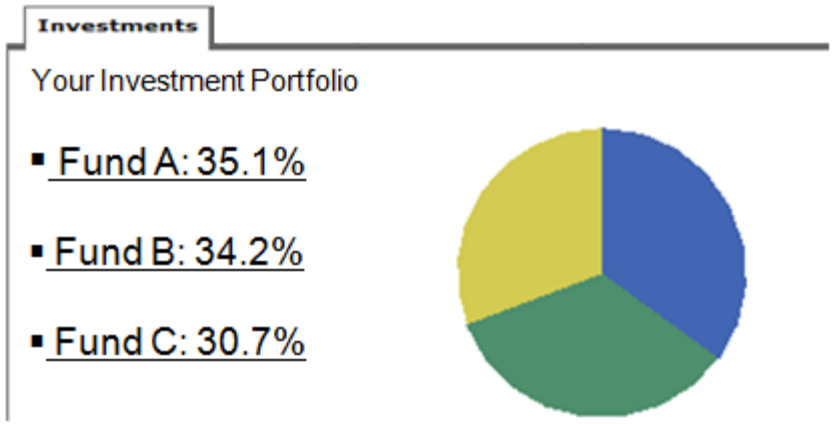
[Chen et al.]



Online Investments

[Chen et al.]





Which funds you invest in?

- Each price history curve is a GIF image from MarketWatch
  - Anyone in the world can get them from this website
- Just compare the image sizes!

Your investment allocation?

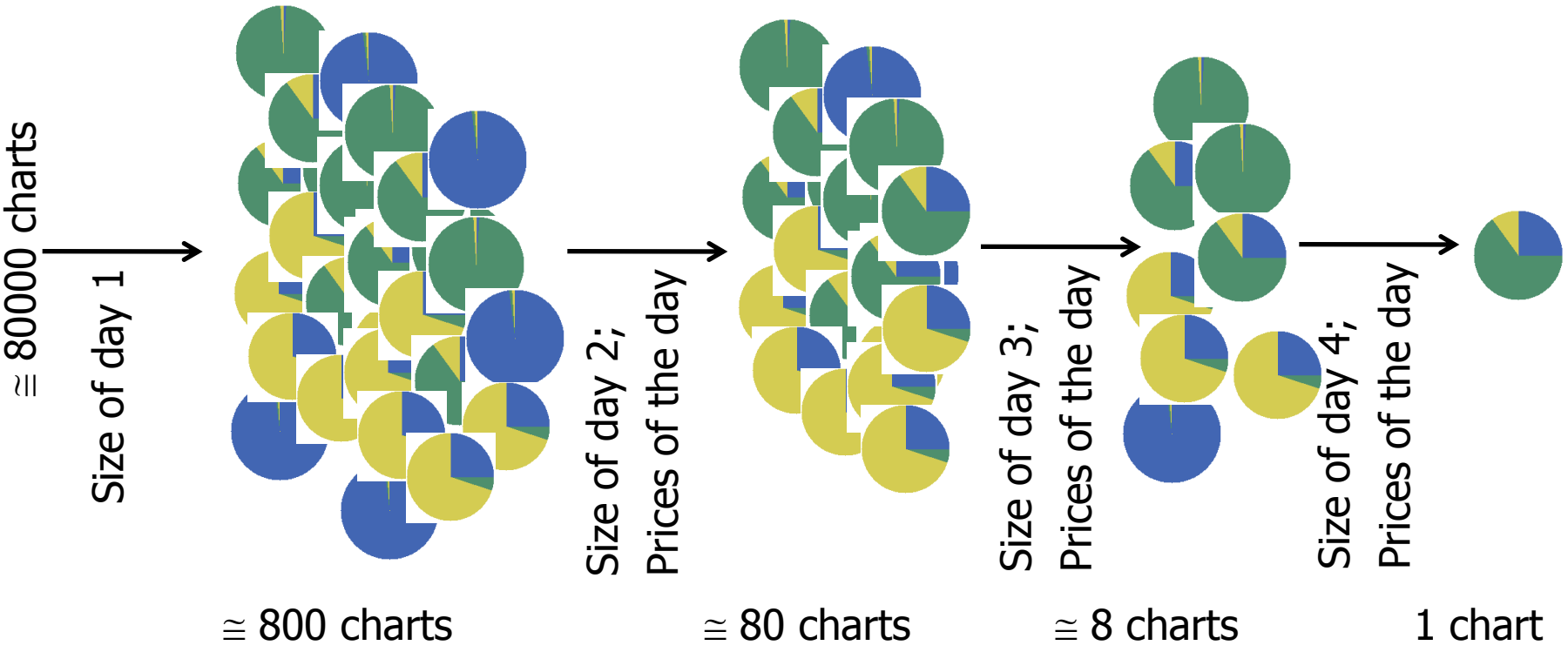
- Can see the size of the pie chart, but hundreds of pie charts have the same image…

Change Over Time Is Revealing!

[Chen et al.]

Financial institution updates your pie chart every day after market close.

Mutual fund prices are public knowledge.



Rounding? Padding?

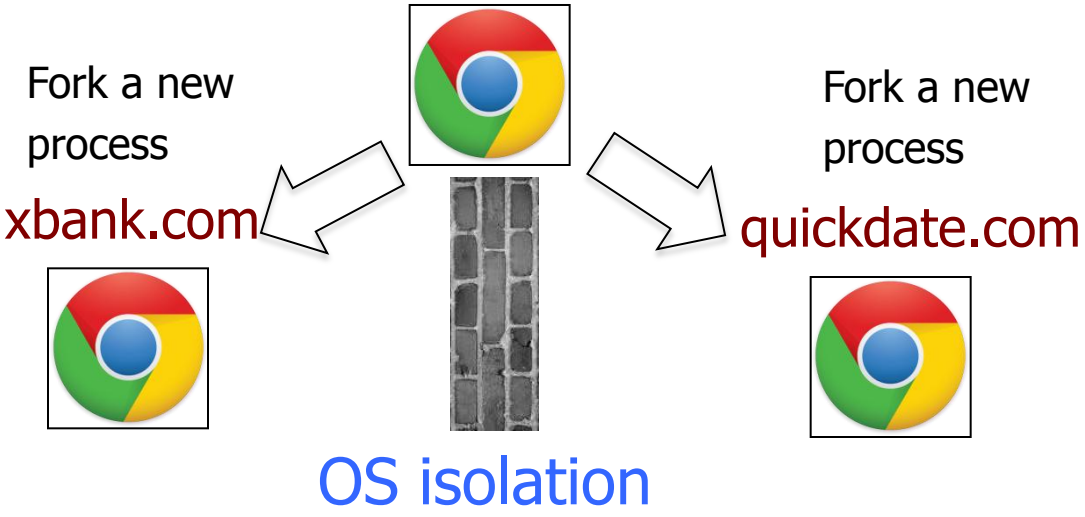
[Chen et al.]

- Still have the asymmetric path problem
- Google's responses are compressed, destination networks may or may not uncompress responses
  - For example, Microsoft gateways uncompress and inspect Web traffic, but university does not
  - Round before compression – university still sees distinguishable sizes; after compression – Microsoft does
- Random padding is not appropriate
  - If user checks several times, repeated random padding of the same responses quickly degrades effectiveness
  - Images come from MarketWatch, not site itself

Trends in Software Design

[Jana and Shmatikov. "Memento: Learning Secrets from Process Footprints". Oakland 2012]

- Applications rely on OS abstractions to improve their safety and reliability
  - "Process", "User"
- Case study: Web browsers



Unintended Consequences

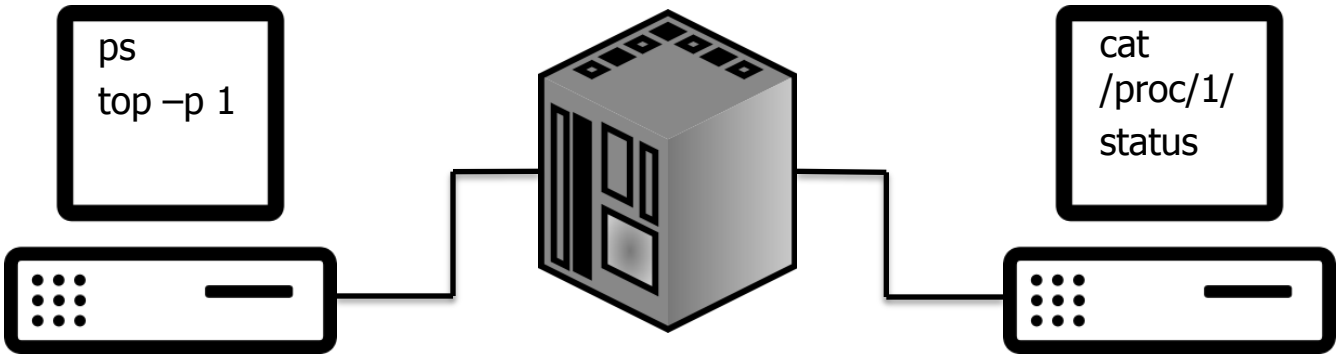
Good

- Better isolation
- Better reliability
  - Others not affected if one process crashes
- Better safety

Bad

Leaks more info to concurrent processes

ProcFS in Multi-User OS

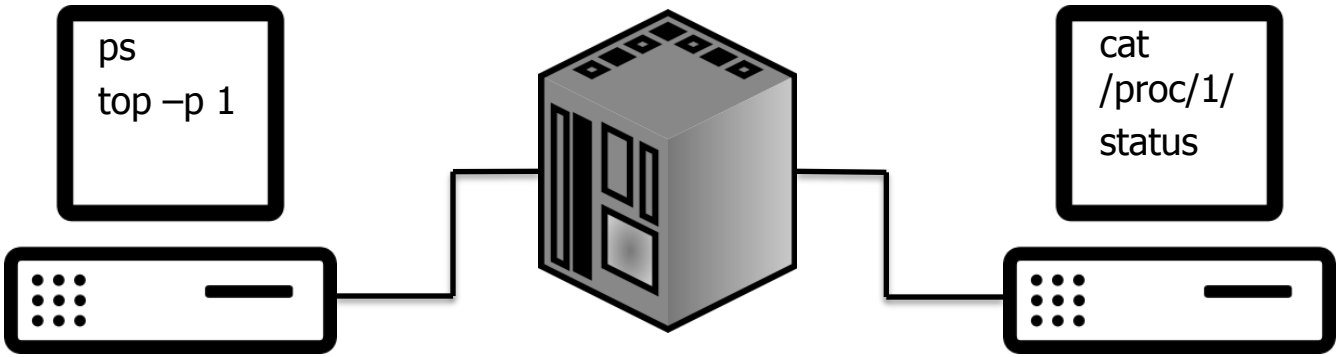


Introduced in the 1980s

Tom Killian
"Processes as Files"
(1984)

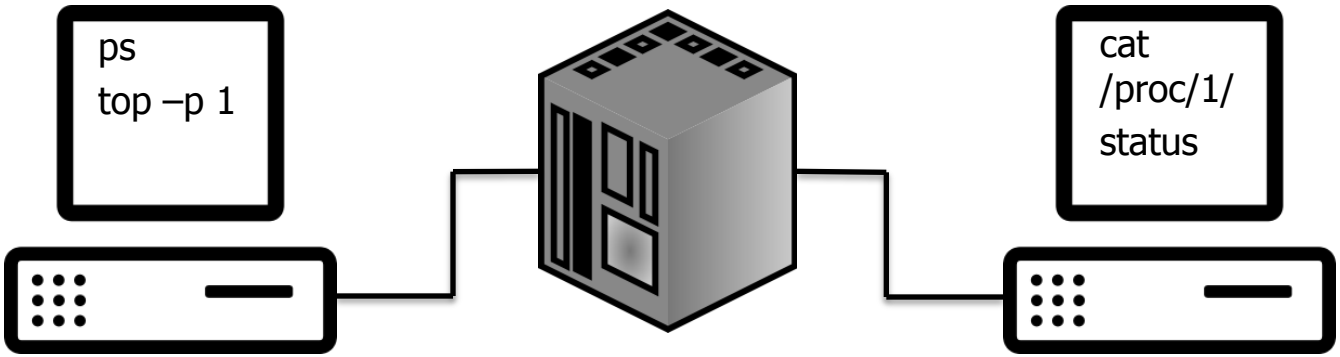
“Noone Uses Multi-User OS Anymore”

Multi-User Isolation



UNIX multi-users in the 1980s

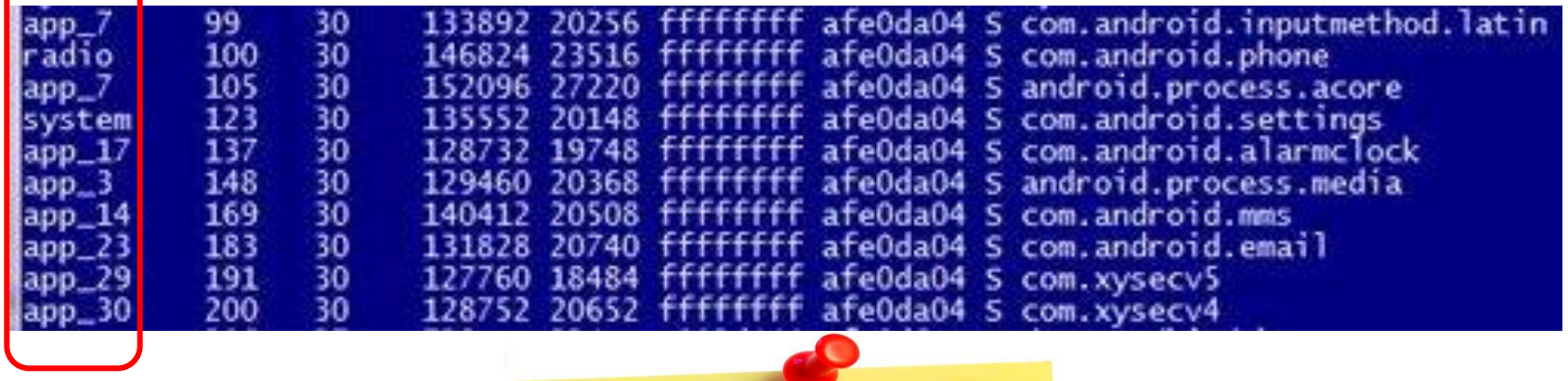
Android Sandboxing



Android “multi-users” today

Android Apps as "Users"

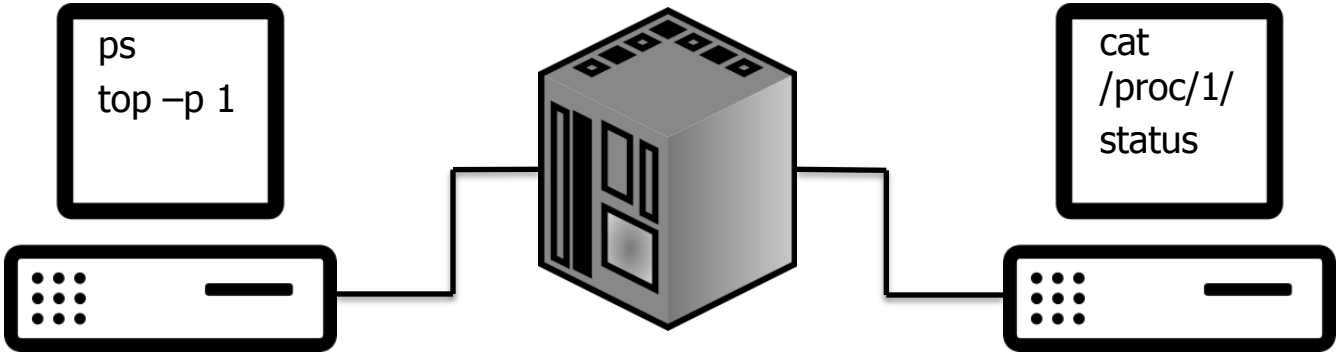
- Different apps run as different users



Android uses OS "user" abstraction to isolate applications

ProcFS Did Not Go Away

ProcFS API is still unchanged!



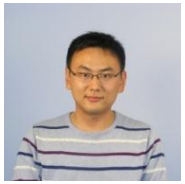
Android "multi-users" today

This Is Not Just About Android

What Can Be Learned from ProcFS?

- No permissions needed to read any world-readable file in ProcFS …
  - IP addresses of network connections ⇦ WTF?!
  - Value of stack pointer ⇦ "Peeping Tom" attacks

  - Various statistics
    - Packet counters ⇦ TCP sequence number inference



    - Number of context switches / CPU scheduling statistics
    - Memory usage ⇧ "Memento" attacks

Keystroke sniffing ⇧

Putting Memory Streams Together

Memprint: Stream of Memory Usage

10568 KB

15976 KB

49380 KB

65948 KB

11632 KB

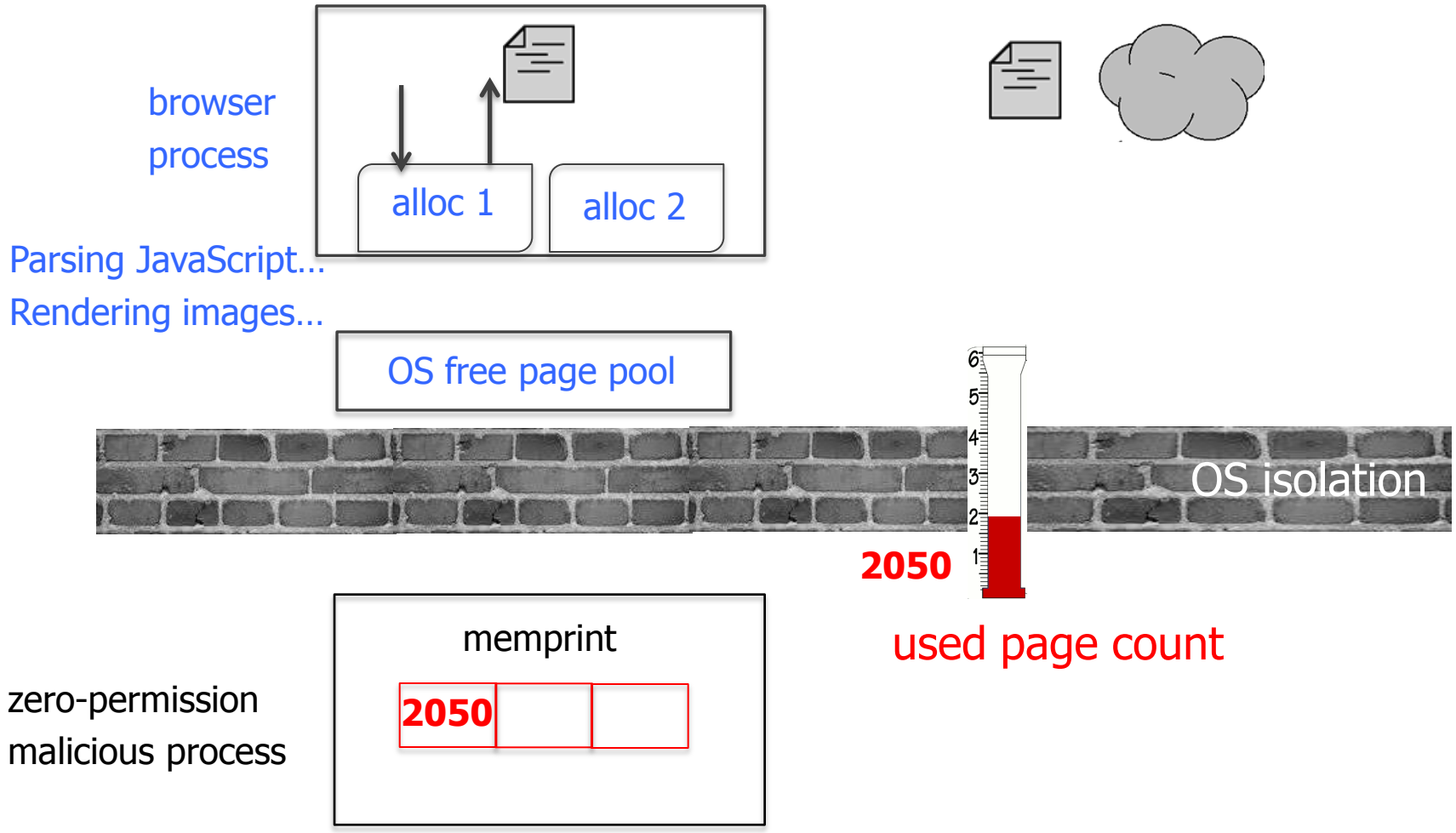
48996 KB

60280 KB

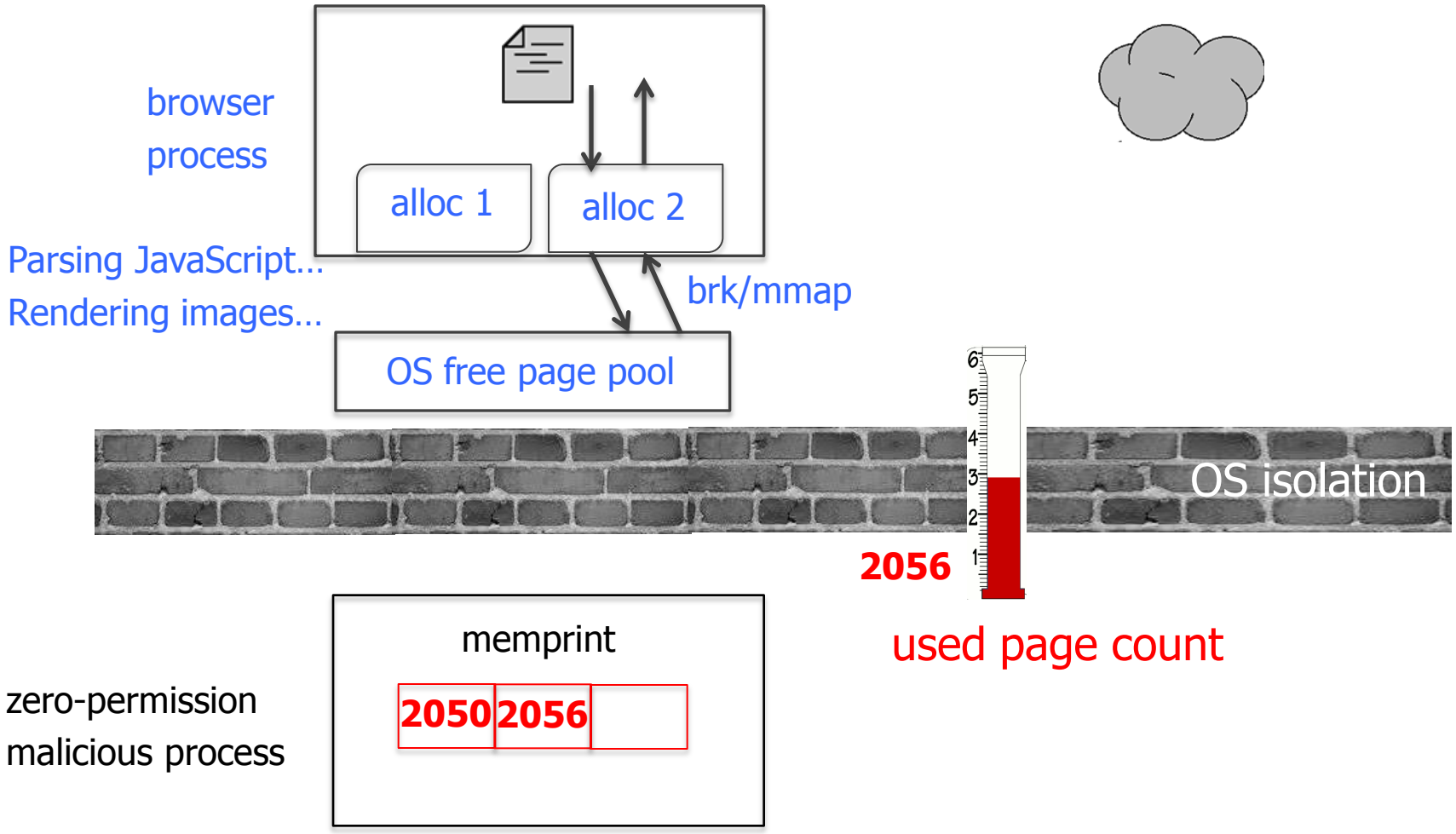
60820 KB

59548 KB

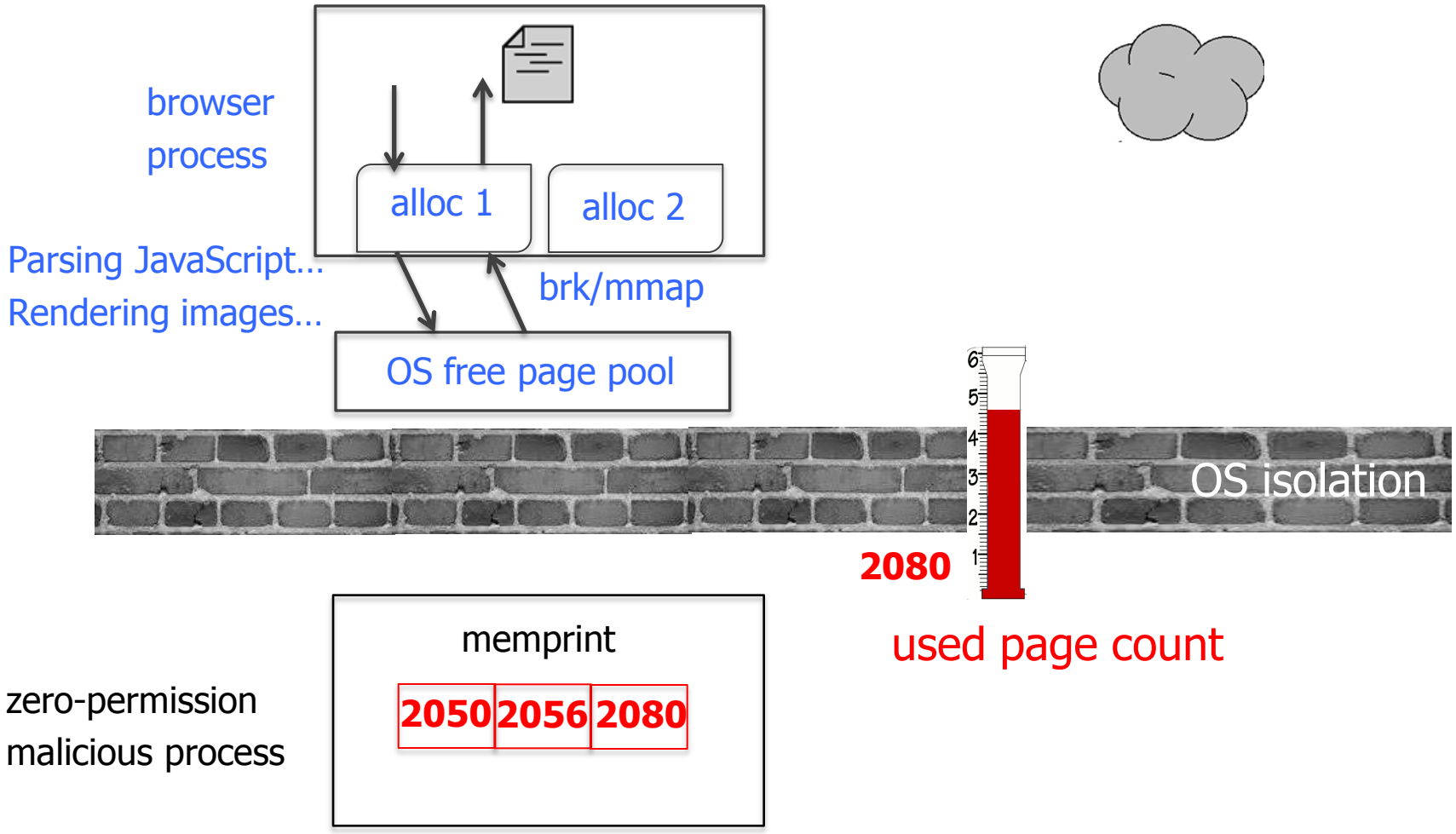
Sniffing Memory Footprints



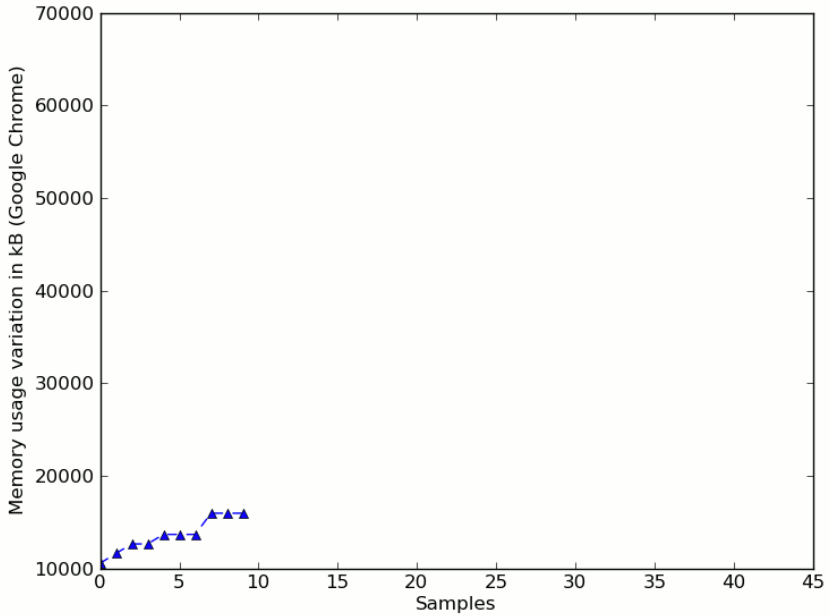
Sniffing Memory Footprints

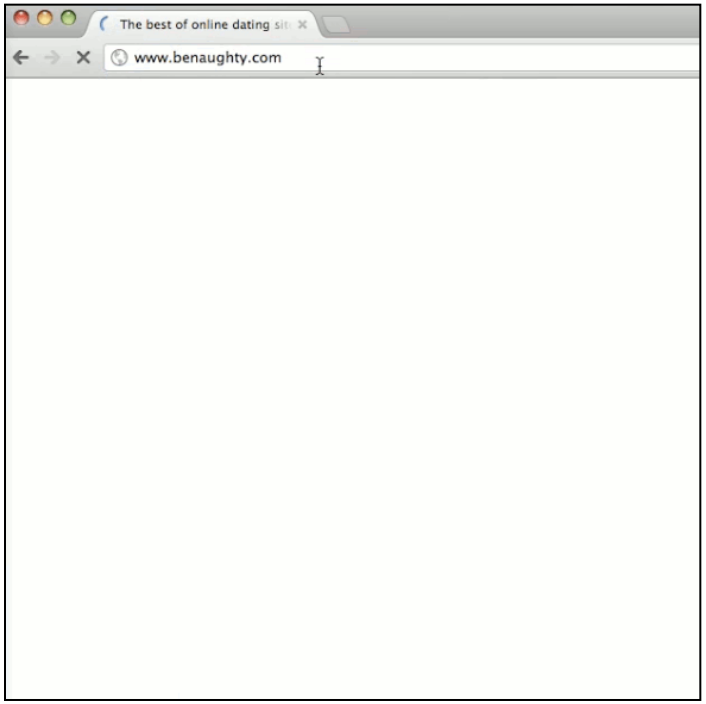


Sniffing Memory Footprints

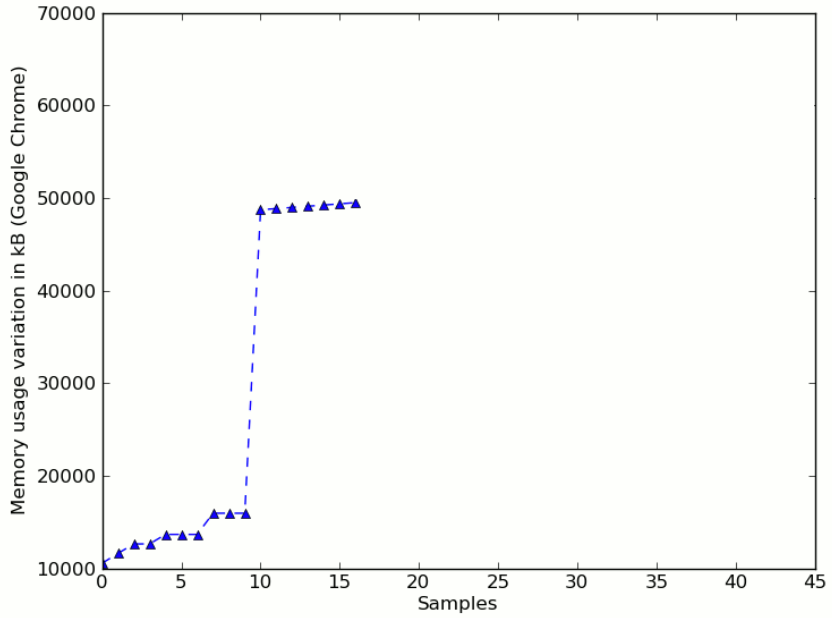


Loading BeNaughty.com in Chrome

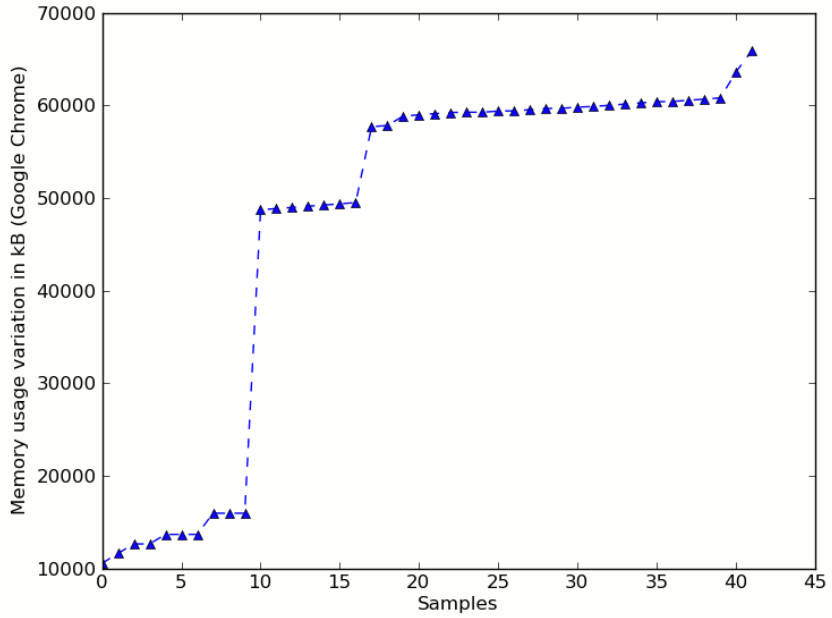


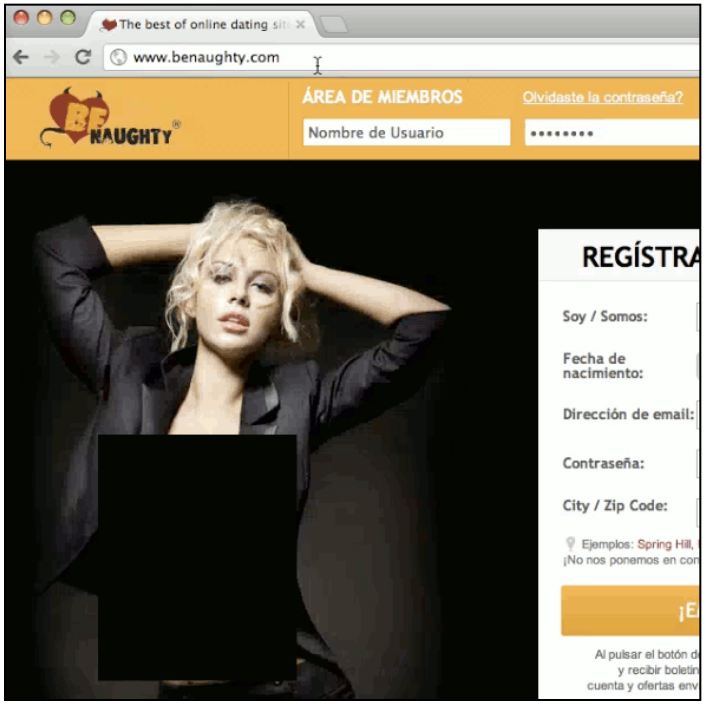


Loading BeNaughty.com in Chrome

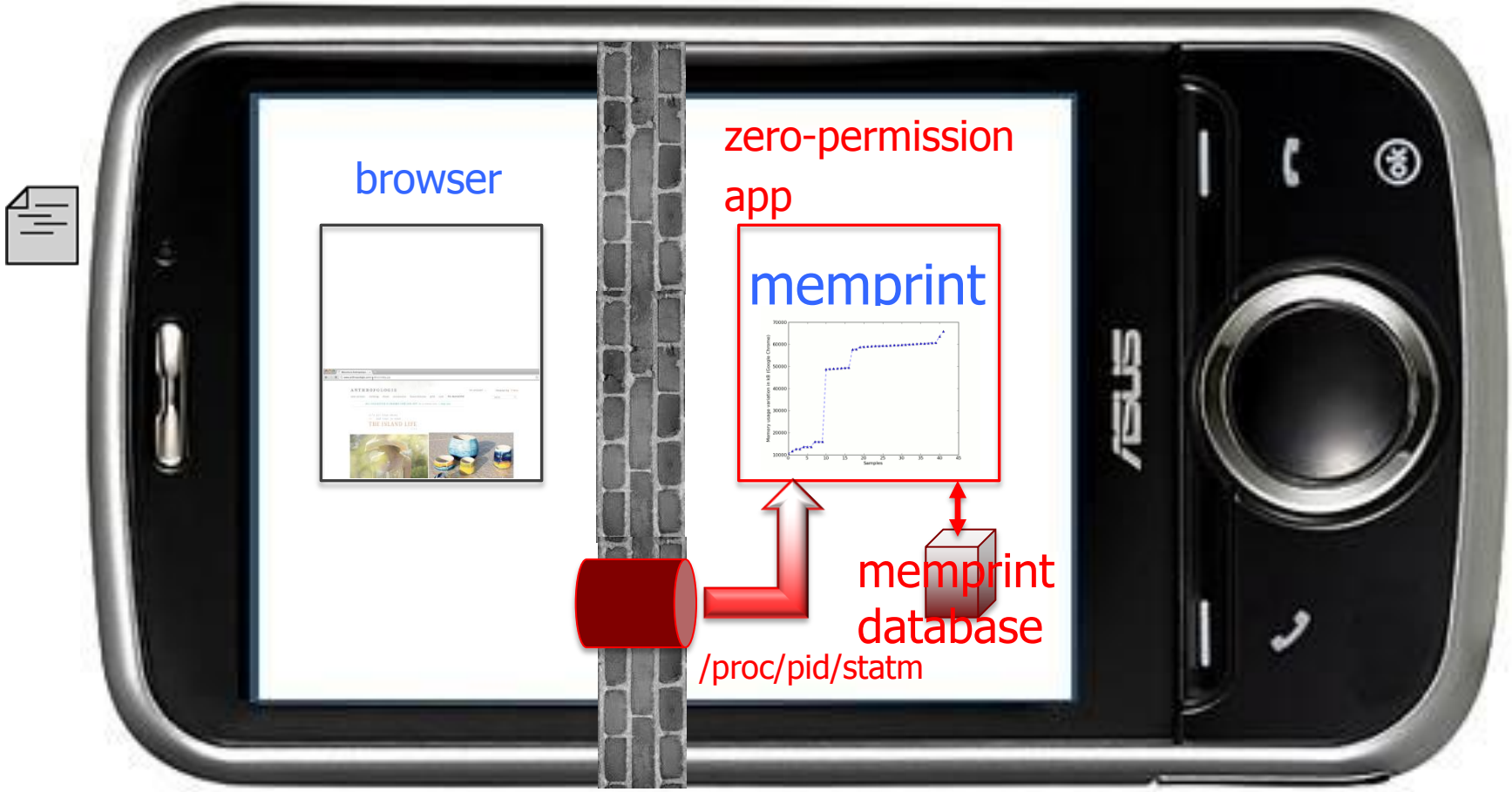


Loading BeNaughty.com in Chrome





Full Attack

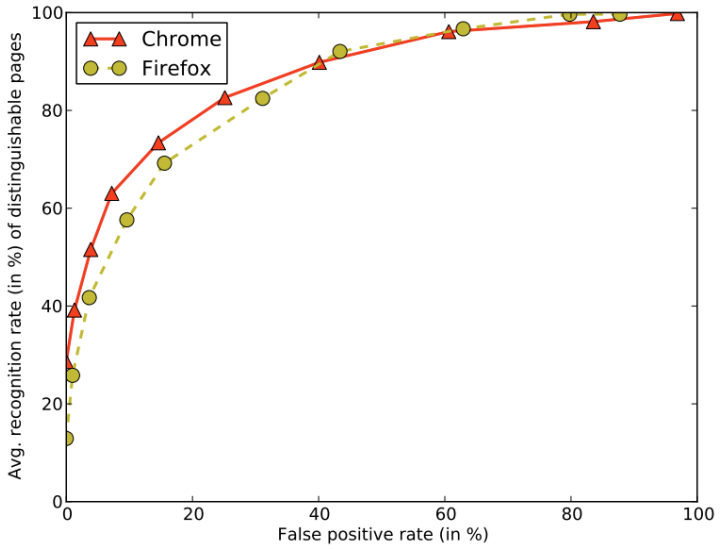


OS isolation

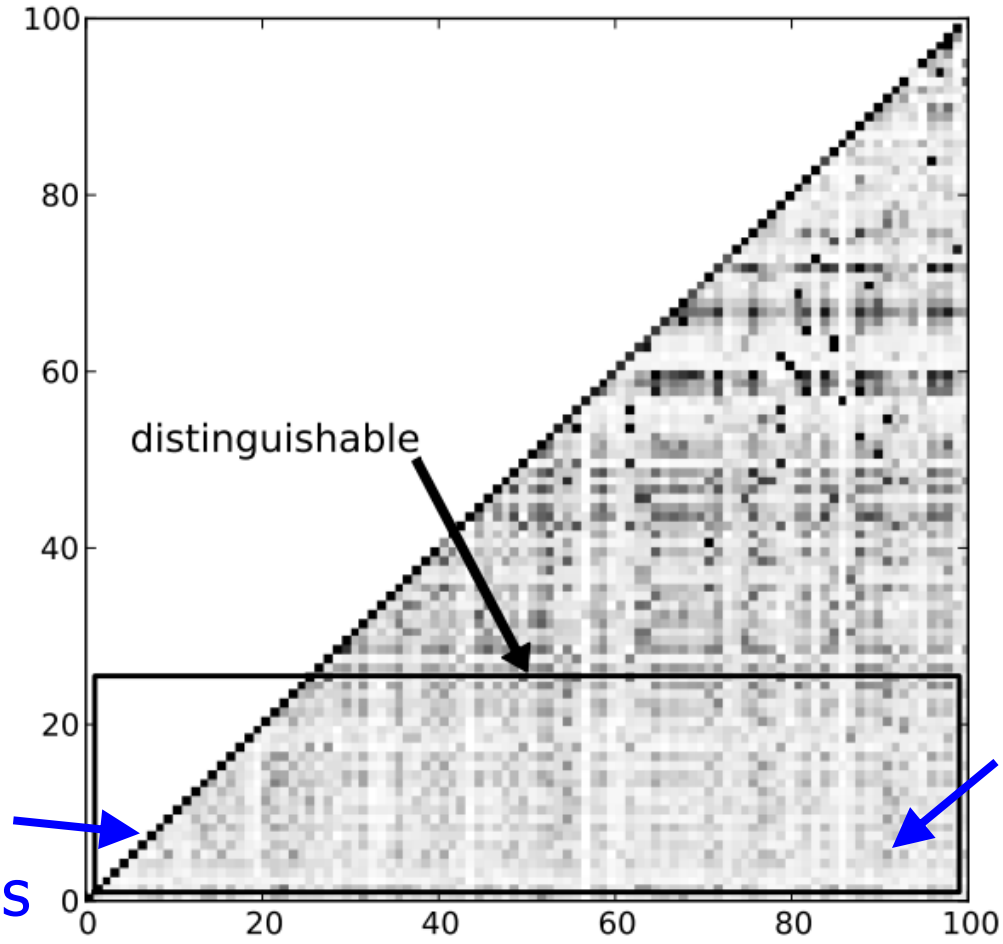
Why the Attack Works

- Memprints are unique - for up to 43% of Alexa top 100,000 pages
  - Can tune recognition to achieve zero false positives
- Memprints are stable across repeated visits to the same page

memprints are OS/browser-dependent but machine-independent



Cross-Page Similarity



similarity = Jaccard index of memprints

Similar to themselves

Different from others

Other Privacy Leaks

- Fine-grained memory dynamics reveal membership in dating sites, interest in medical conditions, etc.
- Dynamics of CPU scheduling reveal individual keystrokes
- General problem: fine-grained resource usage statistics are correlated with secrets
  - These statistics are visible across isolation boundary
  - Their dynamics are a high-bandwidth side channel