Lecture #20: Usable Security #1

UCalgary ENSF619

Elements of Software Security

Instructor: Lorenzo De Carli (lorenzo.decarli@ucalgary.ca)

Based on slides by Shradha Neupane

Lecture structure

- Usable security: overview and general concepts
- General discussion of the paper
- Some more thoughts on usable security

The problem

- A well-known line of reasoning among security professionals is that users make poor security decisions and ignore contextual clues that should make them suspicious of links, webpages, etc.
- According to this line of reasoning, UI-level attempts to forewarn the users and/or “nudge” them to make the right decision are useless, because users will click through anything
- Is this true?

A tale of planes

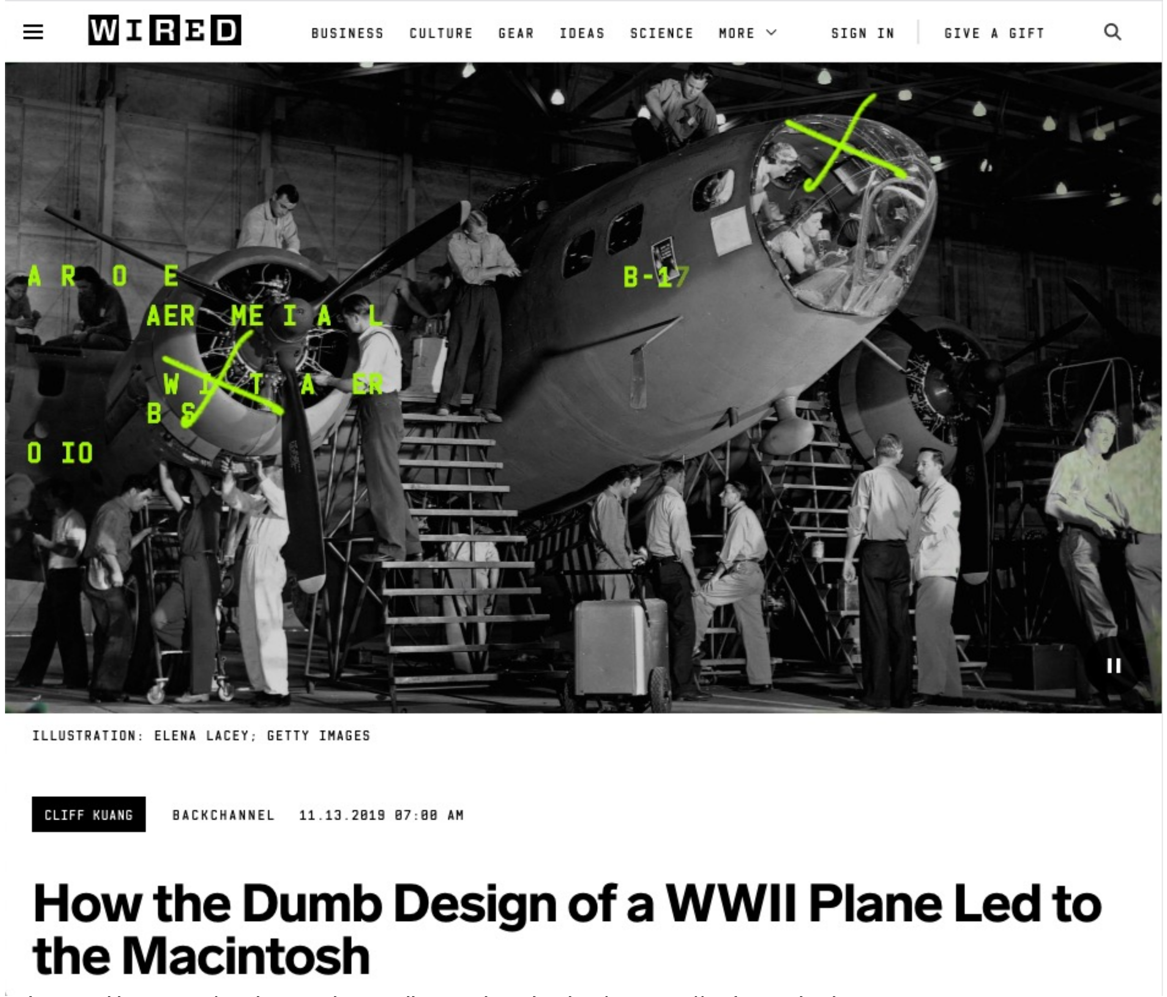


ILLUSTRATION: ELENA LACEY; GETTY IMAGES

CLIFF KUANG BACKCHANNEL 11.13.2019 07:00 AM

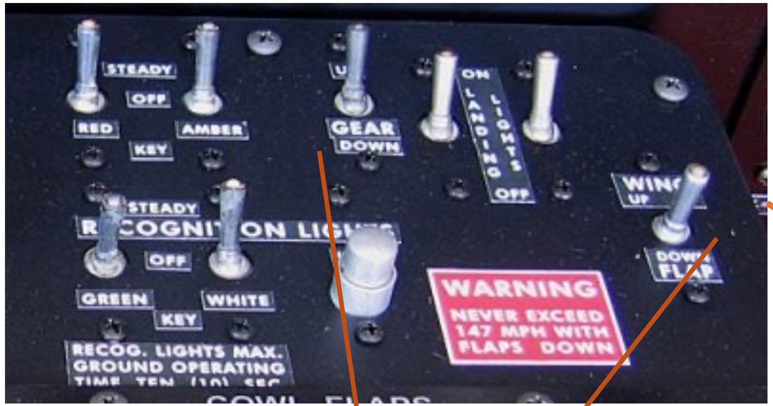
How the Dumb Design of a WWII Plane Led to the Macintosh

https://www.wired.com/story/how-dumb-design-wwii-plane-led-macintosh/?utm_source=pocket-newtab

Case study: the B-17

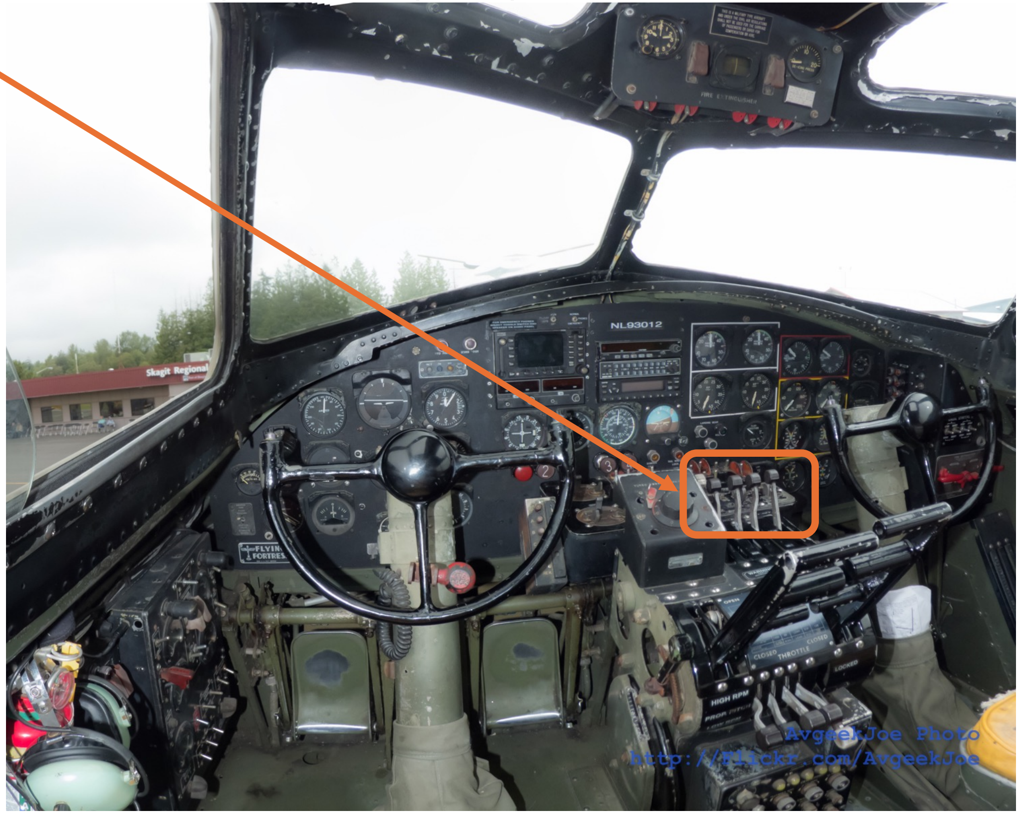
- During WWII, the army noticed that pilots flying B-17 strategic bomber crash-landed with worryingly high frequency
- It would have been easy to chalk it to poor piloting skills, but…
- Eventually, Paul Fitts and Alphonse Chapanis individuated a reasonable explanation

B-17 control panel



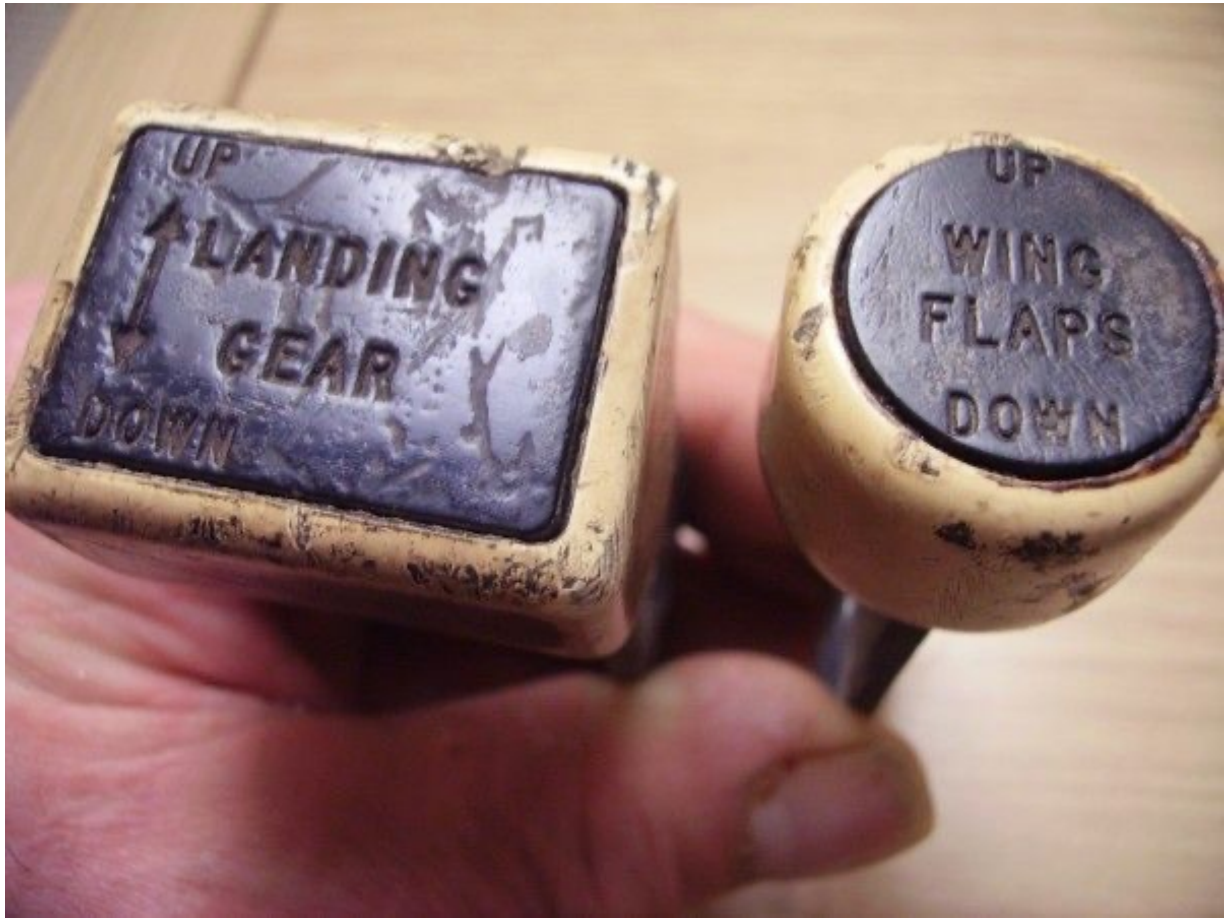
http://lusa.one/2017/10/16/anders-ellerstrand-chapanis-chronicles-en-sjalvbiografi/

https://www.squawkpoint.com/2017/05/human-error/



Turns out, pilot tended to confuse those when under pressure

Solution

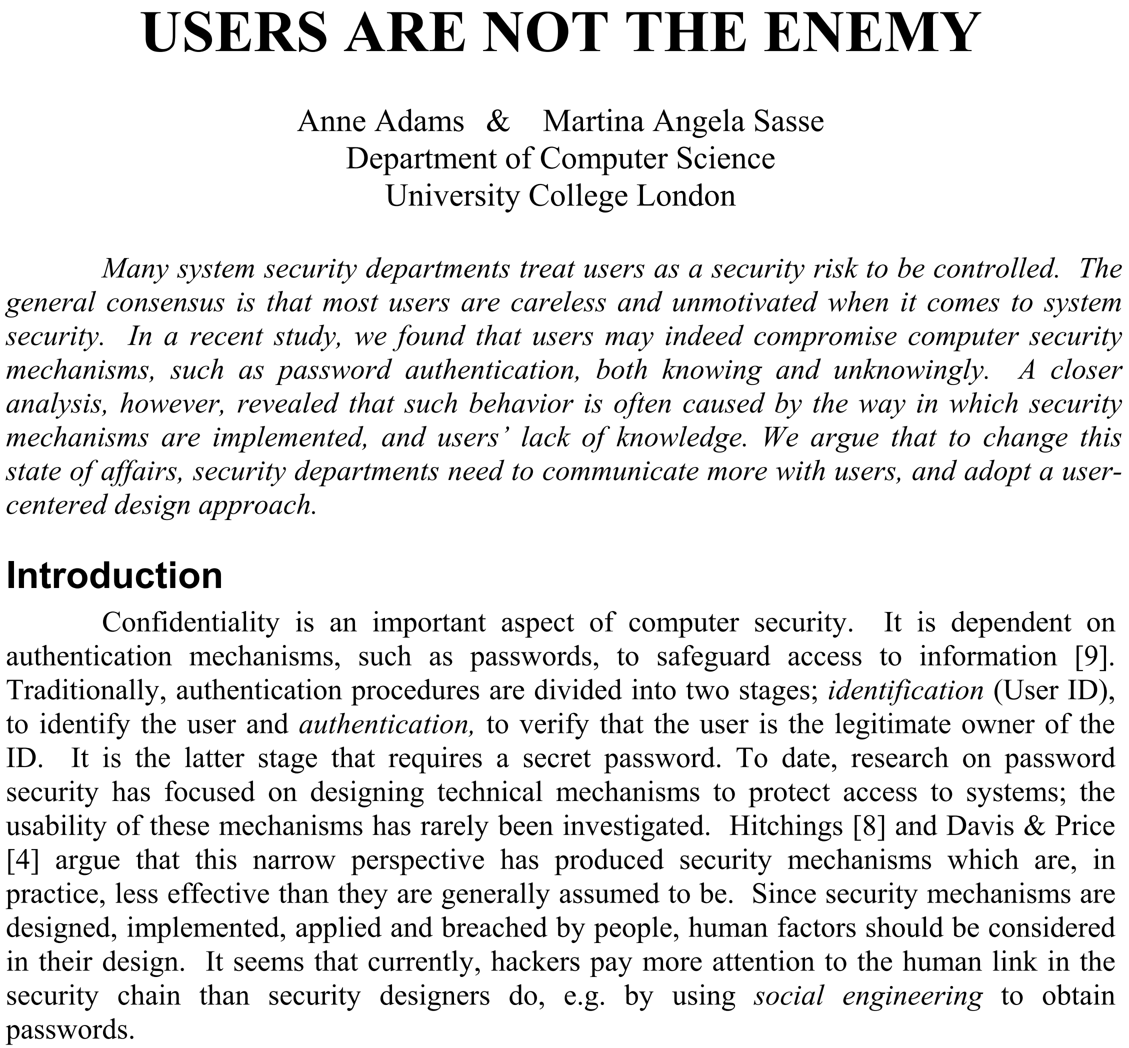


https://www.quora.com/Why-is-the-landing-gear-lever-in-an-airplane-cockpit-designed-to-look-like-little-landing-gear

Take-away

- Blaming all user errors on the user is the hallmark of the lazy system/interface designer
- Humans make mistakes, and ignoring this fact leads to bad design
- Also, there is plenty of evidence that humans do respond positively to good design (no landing accidents after shape coding for airplane controls was introduced; similar shape coding is still in use today)

Let’s talk about the paper



USERS ARE NOT THE ENEMY

Anne Adams & Martina Angela Sasse
Department of Computer Science
University College London

Many system security departments treat users as a security risk to be controlled. The general consensus is that most users are careless and unmotivated when it comes to system security. In a recent study, we found that users may indeed compromise computer security mechanisms, such as password authentication, both knowing and unknowingly. A closer analysis, however, revealed that such behavior is often caused by the way in which security mechanisms are implemented, and users’ lack of knowledge. We argue that to change this state of affairs, security departments need to communicate more with users, and adopt a user-centered design approach.

Introduction

Confidentiality is an important aspect of computer security. It is dependent on authentication mechanisms, such as passwords, to safeguard access to information [9]. Traditionally, authentication procedures are divided into two stages; *identification* (User ID), to identify the user and *authentication,* to verify that the user is the legitimate owner of the ID. It is the latter stage that requires a secret password. To date, research on password security has focused on designing technical mechanisms to protect access to systems; the usability of these mechanisms has rarely been investigated. Hitchings [8] and Davis & Price [4] argue that this narrow perspective has produced security mechanisms which are, in practice, less effective than they are generally assumed to be. Since security mechanisms are designed, implemented, applied and breached by people, human factors should be considered in their design. It seems that currently, hackers pay more attention to the human link in the security chain than security designers do, e.g. by using *social engineering* to obtain passwords.

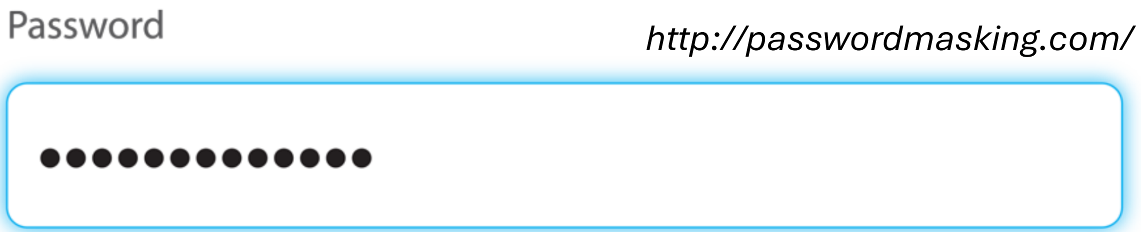
More food for thoughts

Password usability

- Claims that a new technology (biometrics, tokens, etc.) will “kill the password” are frequent
- In practice, passwords do not seem on the way out – why?
- Password are the cockroaches of the authentication world:
    - Anyone can use them (can’t say the same for biometrics, OTPs, etc.) – something you know vs something you own
    - Can be easily changed if compromised
    - Unambiguous verification is easy

Password masking

- You know, those little asterisks covering the password as you enter it



Password

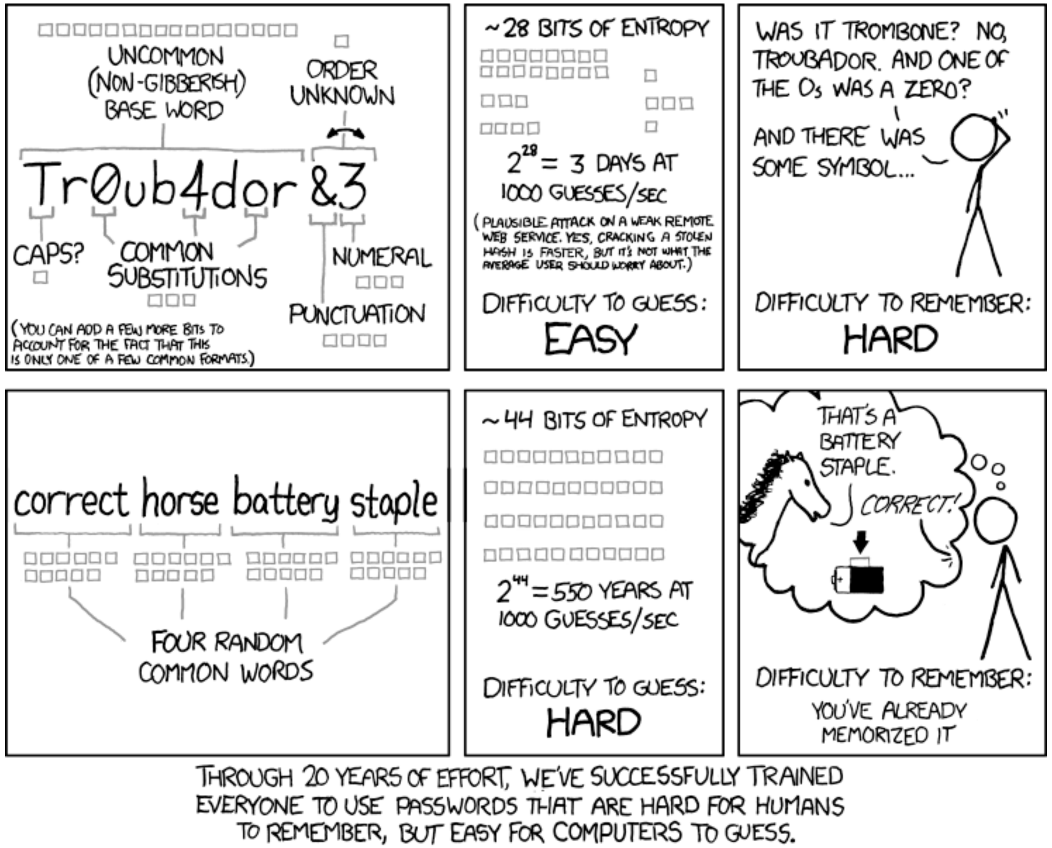
•••••••••••••

http://passwordmasking.com/

- Is it useful? Or just a hindrance? Depends on the threat model!
  - Goal: prevent “shoulder surfing” – but how big of a problem is it, anyway?
  - Drawback: hard to ensure that user is entering password correctly
  - From UI design perspective, probably a good idea to have a “reveal password” option
  - (For more: https://www.schneier.com/blog/archives/2017/07/password_maskin.html)

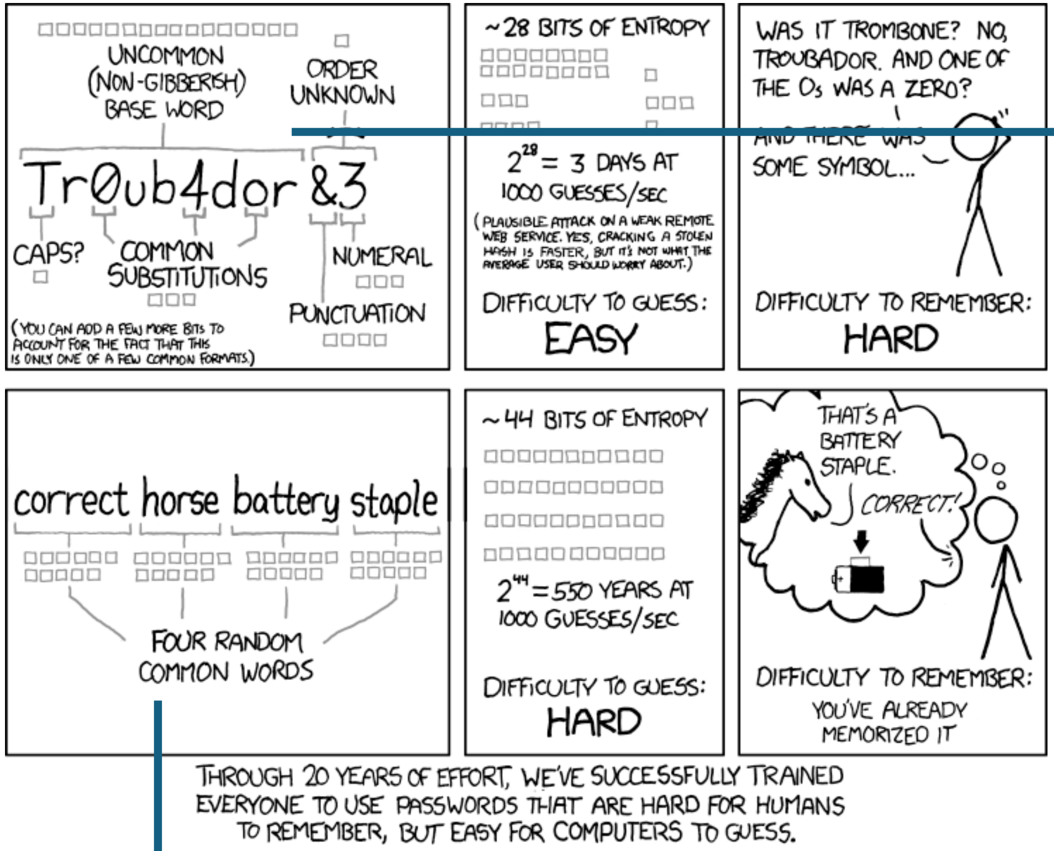
What about secure passwords?

https://xkcd.com/936/, 2011/8/10


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^{28} = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^{44} = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Note: probably not sound advice anymore

Secure password


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

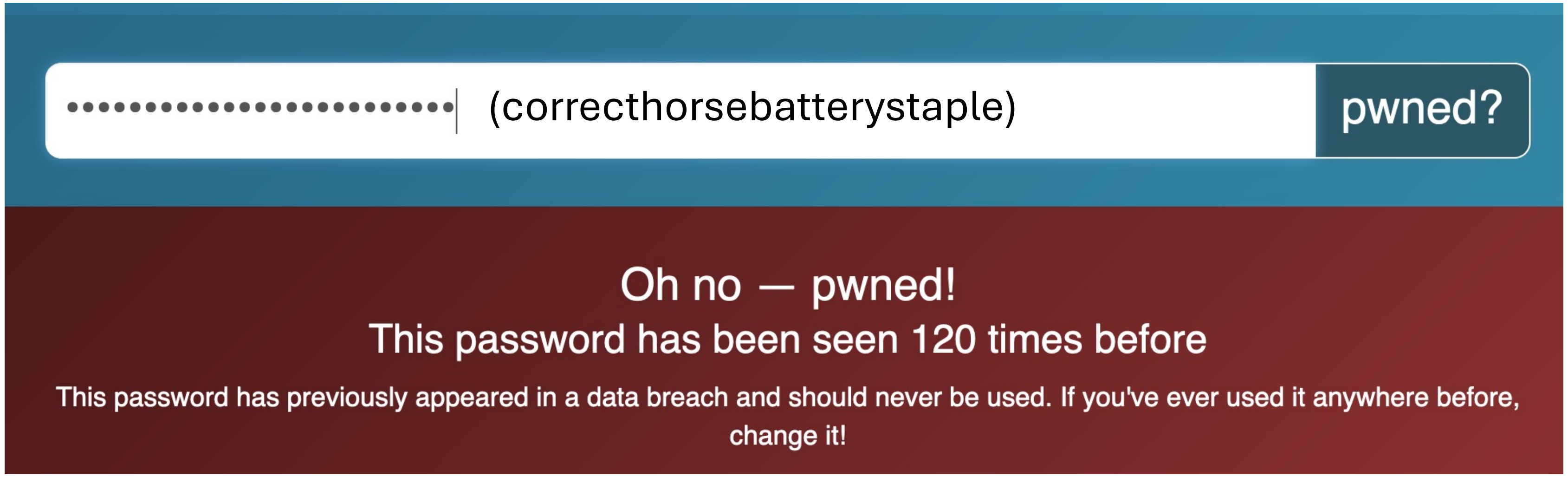
Password entropy describes the size of the space of possible guesses:

- Assuming a dictionary of ~65K base words, the word choices makes up for 16 bits of entropy
- Times 2 because word may or may not be uppercased (1 bit)
- Times 8 (3 bits) to account for a small set of common substitutions
- Times 16 (4 bits) since the word is expected to be followed to one of a small number of punctuation marks
- Times 8 (3 bits) since the punctuation mark is expected to be followed by a number
- Times 2 (1 bit) because punctuation

Assuming a dictionary of 2048 common words, each word adds 11 bits of randomness

Why is this not sound advice anymore?

Reason #1:



(correcthorsebatterystaple)

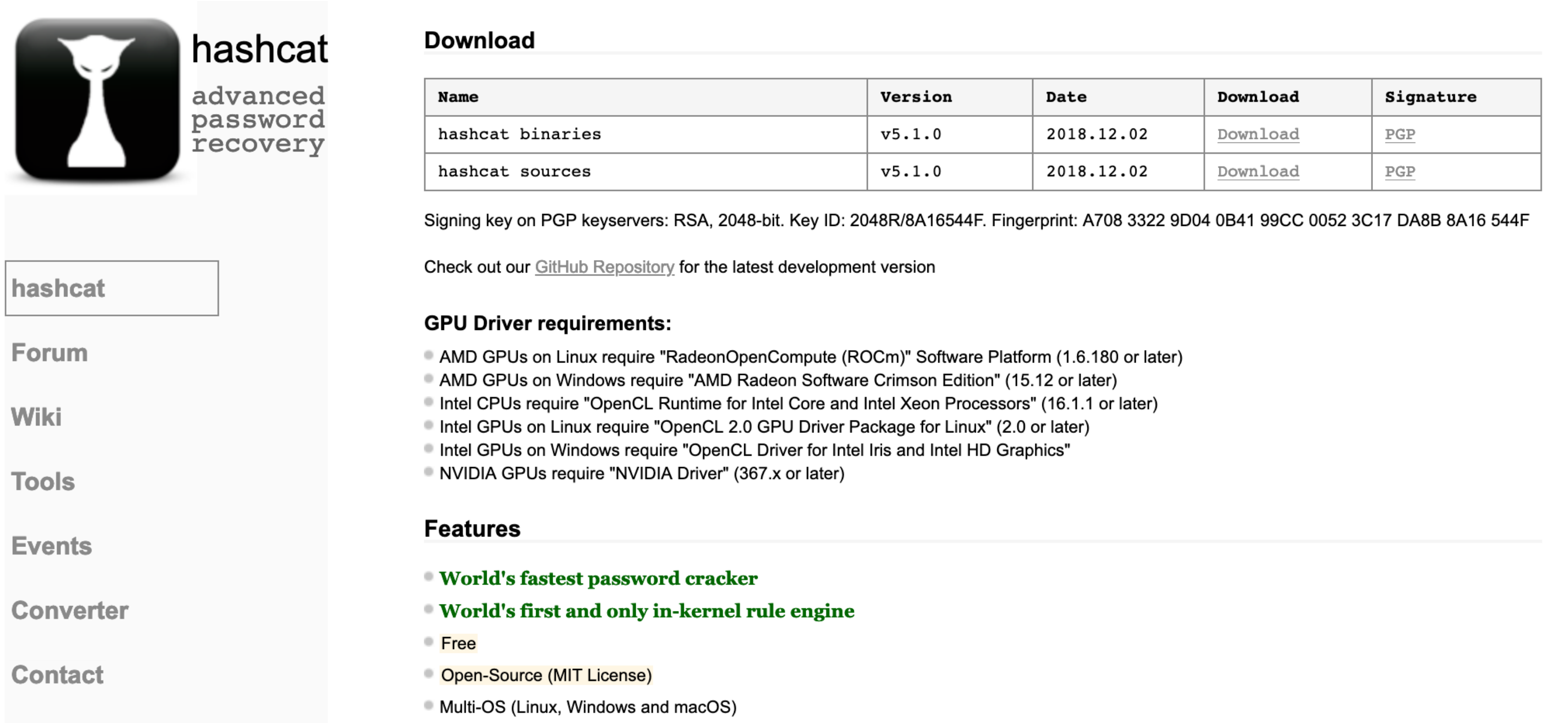
Oh no — pwned!

This password has been seen 120 times before

This password has previously appeared in a data breach and should never be used. If you've ever used it anywhere before, change it!

https://haveibeenpwned.com/Passwords

Why is this not sound advice anymore?



Download

Name	Version	Date	Download	Signature
hashcat binaries	v5.1.0	2018.12.02	Download	PGP
hashcat sources	v5.1.0	2018.12.02	Download	PGP

Signing key on PGP keyservers: RSA, 2048-bit. Key ID: 2048R/8A16544F. Fingerprint: A708 3322 9D04 0B41 99CC 0052 3C17 DA8B 8A16 544F

Check out our GitHub Repository for the latest development version

GPU Driver requirements:

- AMD GPUs on Linux require "RadeonOpenCompute (ROCm)" Software Platform (1.6.180 or later)
- AMD GPUs on Windows require "AMD Radeon Software Crimson Edition" (15.12 or later)
- Intel CPUs require "OpenCL Runtime for Intel Core and Intel Xeon Processors" (16.1.1 or later)
- Intel GPUs on Linux require "OpenCL 2.0 GPU Driver Package for Linux" (2.0 or later)
- Intel GPUs on Windows require "OpenCL Driver for Intel Iris and Intel HD Graphics"
- NVIDIA GPUs require "NVIDIA Driver" (367.x or later)

Features

- **World's fastest password cracker**
- **World's first and only in-kernel rule engine**
- Free
- Open-Source (MIT License)
- Multi-OS (Linux, Windows and macOS)

- Entropy of XKCD password: 2^{44} bits
- Hashcat MD5 performance on 8-GPU cracking appliance: 200 Ghash/sec ($\approx 2^{37}$ attempts/sec)
- $2^{44}/2^{37} = 2^{7} \approx 2$ minutes

More advice

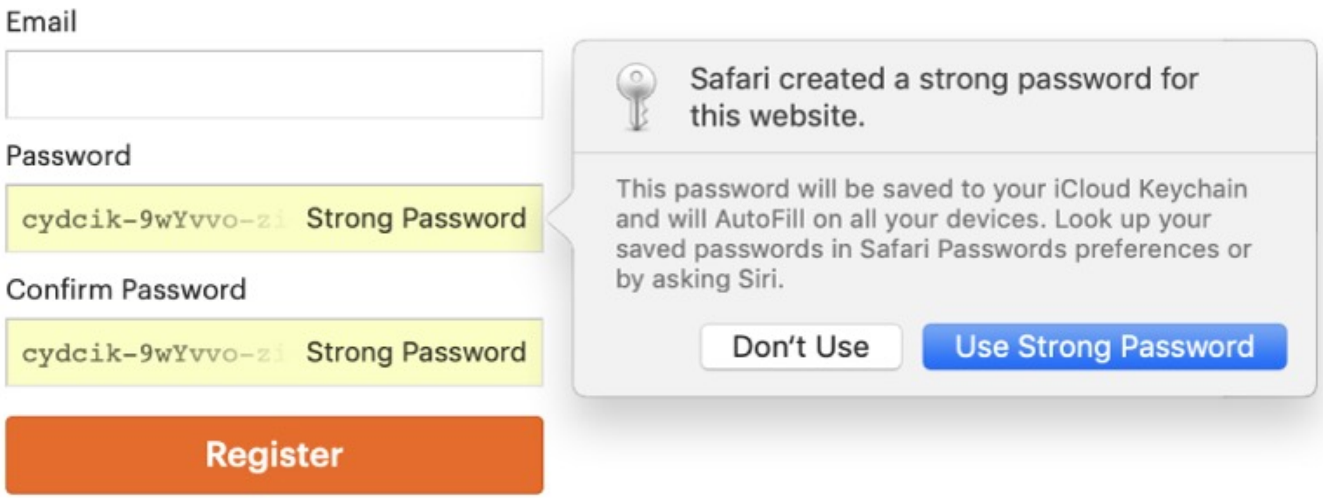
- It’s probably good to avoid dictionary words so to force crackers to search the entire space of all possible character combinations (~95 printable ASCII characters -> ~6 bits of entropy per character; a 25-character password has 150 bits of entropy
  - Hashcat can hack away at it for ~3000 years and still not get it
- How to generate and remember such passwords?

More advice/2

- The Schneier trick: << *Pretty much anything that can be remembered can be cracked.*
  There's still one scheme that works. Back in 2008, I described the "Schneier scheme":
  So if you want your password to be hard to guess, you should choose something that this process will miss. My advice is to take a sentence and turn it into a password. Something like "This little piggy went to market" might become "tlpWENT2m". That nine-character password won't be in anyone's dictionary. Of course, don't use this one, because I've written about it. Choose your own sentence -- something personal.>>
  (https://www.schneier.com/blog/archives/2014/03/choosing_secure_1.html)

More advice/3

- The “Schneier scheme” is probably good for short password, but what if you want a longer one?
- Use a password manager!



Email

Password: cydcik-9wYvvo-z Strong Password

Confirm Password: cydcik-9wYvvo-z Strong Password

Safari created a strong password for this website.

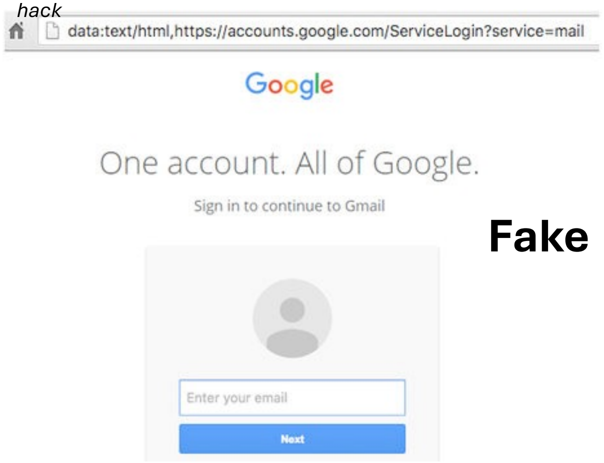
This password will be saved to your iCloud Keychain and will AutoFill on all your devices. Look up your saved passwords in Safari Passwords preferences or by asking Siri.

https://support.apple.com/guide/mac-help/use-keychains-to-store-passwords-mchlf375f392/mac

URL problems

- Lots of people have gripes with URLs
  - Complex, long, hard to read
  - People can easily be redirected to a phishing website imitating a legitimated one without realizing it

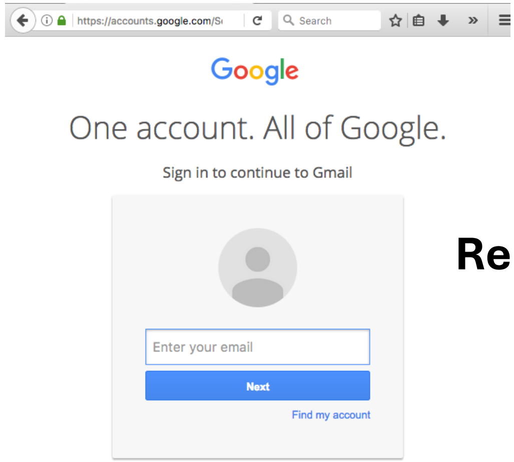
https://www.express.co.uk/life-style/science-technology/755409/gmail-phishing-scam-fake-email-login-hack


data:text/html,https://accounts.google.com/ServiceLogin?service=mail

One account. All of Google.

Sign in to continue to Gmail

Fake


https://accounts.google.com/S

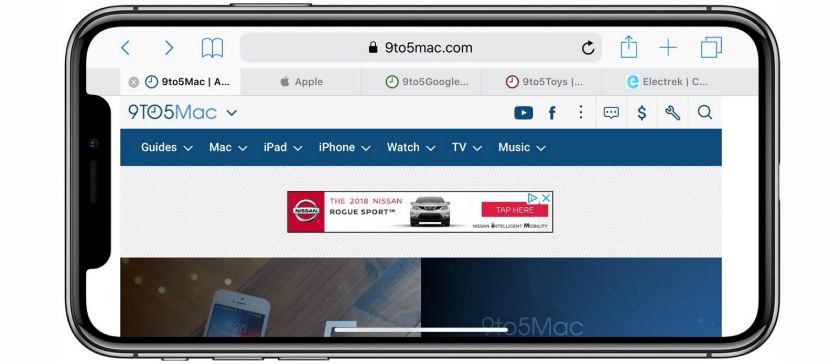
One account. All of Google.

Sign in to continue to Gmail

Real

URL problems/2

- That’s why modern mobile browsers highlight the hostname and hide the rest of the URL:



https://9to5mac.com/2018/06/15/iphone-ipad-how-to-show-safari-tab-icons-in-ios-12/

- Not fully satisfying, but it is hard to find a viable alternative to URLs!

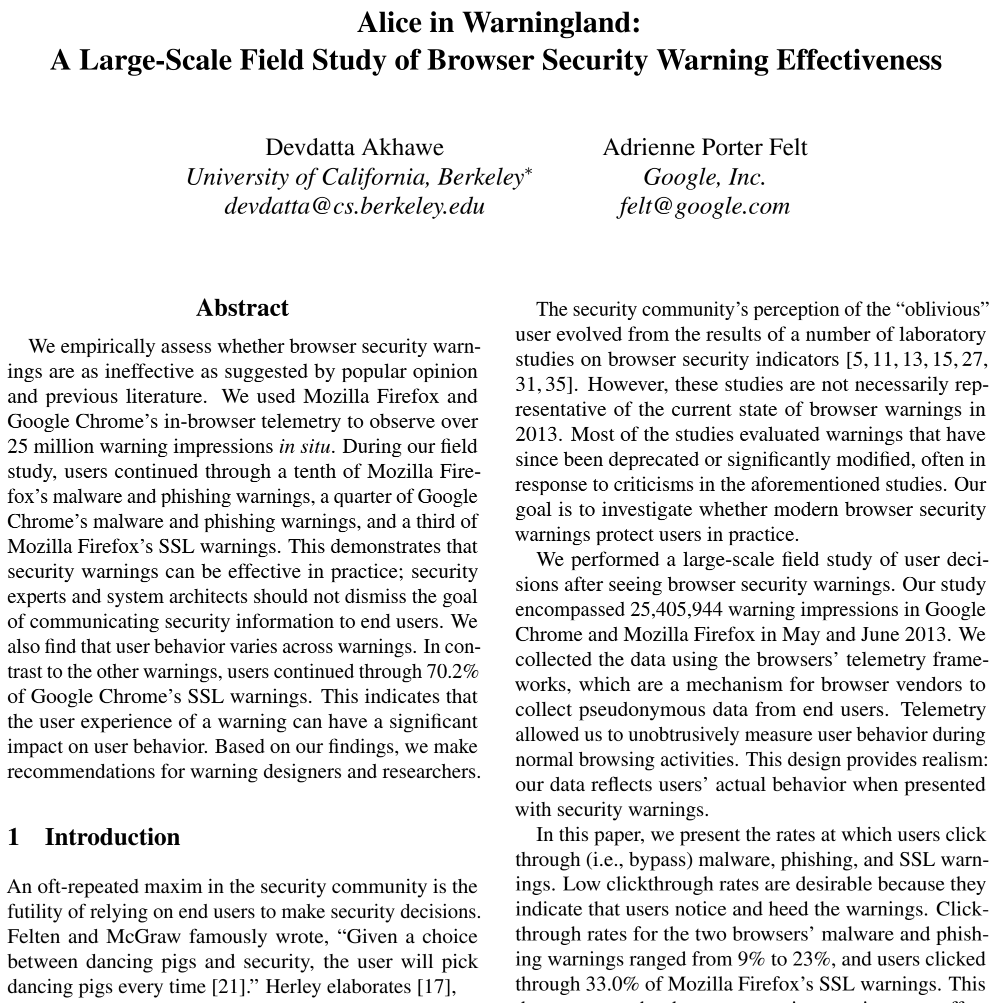
Google wants to get rid of URLs but doesn’t know what to use instead

Their complexity makes them a security hazard; their ubiquity makes replacement nigh impossible.

PETER BRIGHT - 9/5/2018, 10:04 AM

https://arstechnica.com/gadgets/2018/09/google-wants-to-get-rid-of-urls-but-doesnt-know-what-to-use-instead/

It does not stop at warnings…



Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness

Devdatta Akhawe
University of California, Berkeley[*]
devdatta@cs.berkeley.edu

Adrienne Porter Felt
Google, Inc.
felt@google.com

Abstract

We empirically assess whether browser security warnings are as ineffective as suggested by popular opinion and previous literature. We used Mozilla Firefox and Google Chrome's in-browser telemetry to observe over 25 million warning impressions *in situ*. During our field study, users continued through a tenth of Mozilla Firefox's malware and phishing warnings, a quarter of Google Chrome's malware and phishing warnings, and a third of Mozilla Firefox's SSL warnings. This demonstrates that security warnings can be effective in practice; security experts and system architects should not dismiss the goal of communicating security information to end users. We also find that user behavior varies across warnings. In contrast to the other warnings, users continued through 70.2% of Google Chrome's SSL warnings. This indicates that the user experience of a warning can have a significant impact on user behavior. Based on our findings, we make recommendations for warning designers and researchers.

1 Introduction

An oft-repeated maxim in the security community is the futility of relying on end users to make security decisions. Felten and McGraw famously wrote, "Given a choice between dancing pigs and security, the user will pick dancing pigs every time [21]." Herley elaborates [17],

The security community's perception of the "oblivious" user evolved from the results of a number of laboratory studies on browser security indicators [5, 11, 13, 15, 27, 31, 35]. However, these studies are not necessarily representative of the current state of browser warnings in 2013. Most of the studies evaluated warnings that have since been deprecated or significantly modified, often in response to criticisms in the aforementioned studies. Our goal is to investigate whether modern browser security warnings protect users in practice.

We performed a large-scale field study of user decisions after seeing browser security warnings. Our study encompassed 25,405,944 warning impressions in Google Chrome and Mozilla Firefox in May and June 2013. We collected the data using the browsers' telemetry frameworks, which are a mechanism for browser vendors to collect pseudonymous data from end users. Telemetry allowed us to unobtrusively measure user behavior during normal browsing activities. This design provides realism: our data reflects users' actual behavior when presented with security warnings.

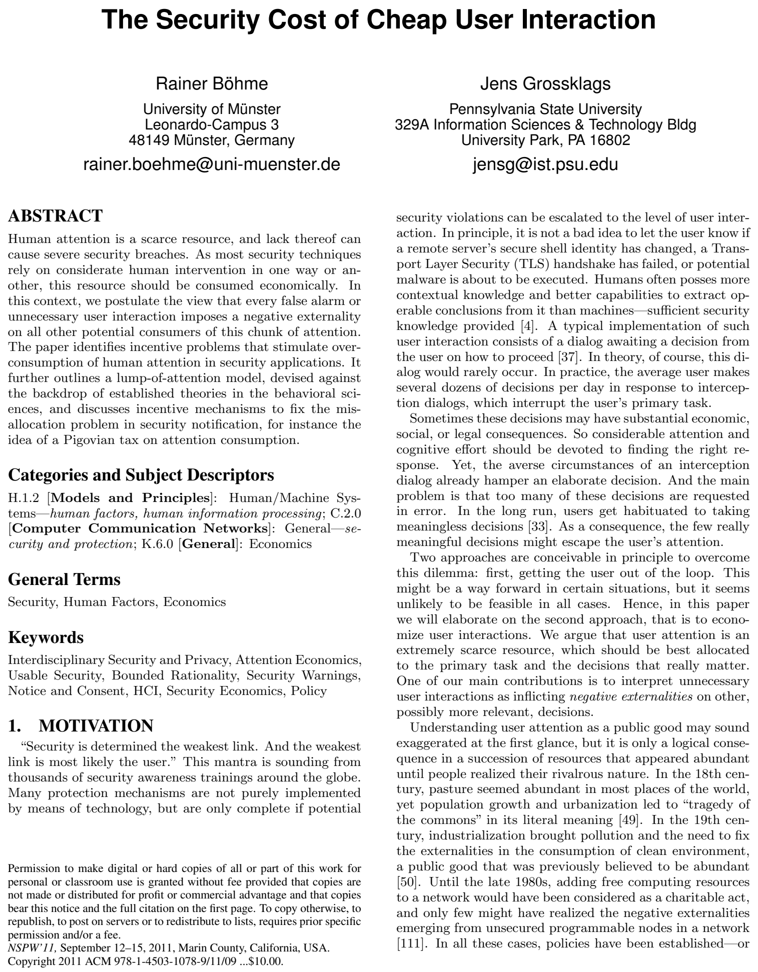
In this paper, we present the rates at which users click through (i.e., bypass) malware, phishing, and SSL warnings. Low clickthrough rates are desirable because they indicate that users notice and heed the warnings. Clickthrough rates for the two browsers' malware and phishing warnings ranged from 9% to 23%, and users clicked through 33.0% of Mozilla Firefox's SSL warnings. This

Why should we care about browser warnings?

- Browsing the web is a (potentially) risky business
  - Some websites are sketchier than others ☺
- Ultimately a browser cannot guess what the user intention is, however…
- …it can provide hints and clues to call users' attention on facts that may help the user make more informed decisions
- What's the best way to convey these clues and information however is an open UI design problem

Warning fatigue

- Human attention is a limited resource
- It should be consumed only when necessary
- When a lot of low-impact decisions are presented to the user, the user learns to pay them little attention



The Security Cost of Cheap User Interaction

Rainer Böhme
University of Münster
Leonardo-Campus 3
48149 Münster, Germany
rainer.boehme@uni-muenster.de

Jens Grossklags
Pennsylvania State University
329A Information Sciences & Technology Bldg
University Park, PA 16802
jensg@ist.psu.edu

ABSTRACT

Human attention is a scarce resource, and lack thereof can cause severe security breaches. As most security techniques rely on considerate human intervention in one way or another, this resource should be consumed economically. In this context, we postulate the view that every false alarm or unnecessary user interaction imposes a negative externality on all other potential consumers of this chunk of attention. The paper identifies incentive problems that stimulate overconsumption of human attention in security applications. It further outlines a lump-of-attention model, devised against the backdrop of established theories in the behavioral sciences, and discusses incentive mechanisms to fix the misallocation problem in security notification, for instance the idea of a Pigovian tax on attention consumption.

Categories and Subject Descriptors

H.1.2 [**Models and Principles**]: Human/Machine Systems—*human factors, human information processing*; C.2.0 [**Computer Communication Networks**]: General—*security and protection*; K.6.0 [**General**]: Economics

General Terms

Security, Human Factors, Economics

Keywords

Interdisciplinary Security and Privacy, Attention Economics, Usable Security, Bounded Rationality, Security Warnings, Notice and Consent, HCI, Security Economics, Policy

1. MOTIVATION

"Security is determined the weakest link. And the weakest link is most likely the user." This mantra is sounding from thousands of security awareness trainings around the globe. Many protection mechanisms are not purely implemented by means of technology, but are only complete if potential security violations can be escalated to the level of user interaction. In principle, it is not a bad idea to let the user know if a remote server's secure shell identity has changed, a Transport Layer Security (TLS) handshake has failed, or potential malware is about to be executed. Humans often posses more contextual knowledge and better capabilities to extract operable conclusions from it than machines—sufficient security knowledge provided [4]. A typical implementation of such user interaction consists of a dialog awaiting a decision from the user on how to proceed [37]. In theory, of course, this dialog would rarely occur. In practice, the average user makes several dozens of decisions per day in response to interception dialogs, which interrupt the user's primary task.

Sometimes these decisions may have substantial economic, social, or legal consequences. So considerable attention and cognitive effort should be devoted to finding the right response. Yet, the averse circumstances of an interception dialog already hamper an elaborate decision. And the main problem is that too many of these decisions are requested in error. In the long run, users get habituated to taking meaningless decisions [33]. As a consequence, the few really meaningful decisions might escape the user's attention.

Two approaches are conceivable in principle to overcome this dilemma: first, getting the user out of the loop. This might be a way forward in certain situations, but it seems unlikely to be feasible in all cases. Hence, in this paper we will elaborate on the second approach, that is to economize user interactions. We argue that user attention is an extremely scarce resource, which should be best allocated to the primary task and the decisions that really matter. One of our main contributions is to interpret unnecessary user interactions as inflicting *negative externalities* on other, possibly more relevant, decisions.

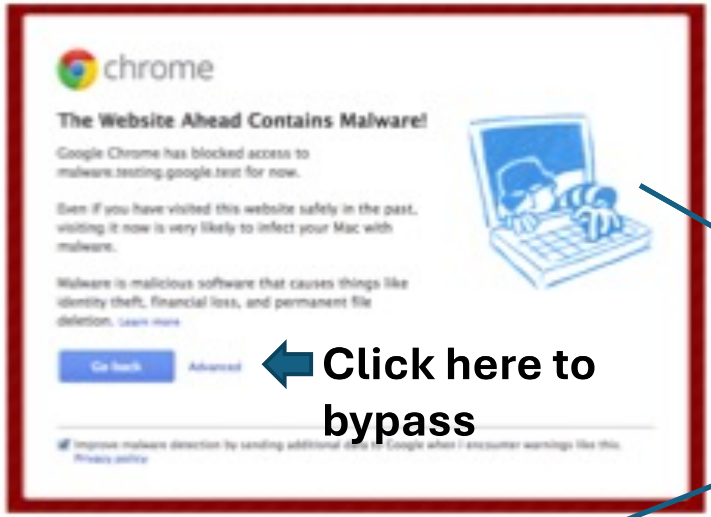
Understanding user attention as a public good may sound exaggerated at the first glance, but it is only a logical consequence in a succession of resources that appeared abundant until people realized their rivalrous nature. In the 18th century, pasture seemed abundant in most places of the world, yet population growth and urbanization led to "tragedy of the commons" in its literal meaning [49]. In the 19th century, industrialization brought pollution and the need to fix the externalities in the consumption of clean environment, a public good that was previously believed to be abundant [50]. Until the late 1980s, adding free computing resources to a network would have been considered as a charitable act, and only few might have realized the negative externalities emerging from unsecured programmable nodes in a network [111]. In all these cases, policies have been established—or

NSPW'11, September 12–15, 2011, Marin County, California, USA.

Issue in warning design

- How often/in which occasions should warning be presented?
    - Not enough warning expose the users to risks; too many warnings habituates the user to ignore them
- How should warnings fit in the user workflow?
    - Modal warnings? Interstitial warnings? How many clicks to bypass?
- How should warning messages be designed?
    - Colors, graphic, language, etc.

Malware warnings



The Website Ahead Contains Malware!

Google Chrome has blocked access to malware.testing.google.test for now.

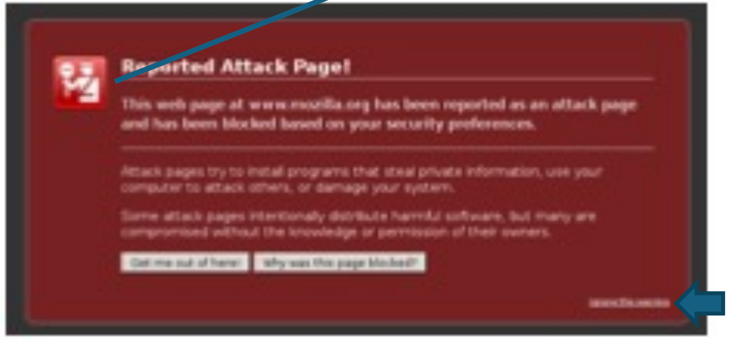
Even if you have visited this website safely in the past, visiting it now is very likely to infect your Mac with malware.

Malware is malicious software that causes things like identity theft, financial loss, and permanent file deletion. Learn more

☑ Improve malware detection by sending additional data to Google when I encounter warnings like this. Privacy policy

Click here to bypass

Figure 1: Malware warning for Google Chrome



Reported Attack Page!

This web page at www.mozilla.org has been reported as an attack page and has been blocked based on your security preferences.

Attack pages try to install programs that steal private information, use your computer to attack others, or damage your system.

Some attack pages intentionally distribute harmful software, but many are compromised without the knowledge or permission of their owners.

Click here to bypass

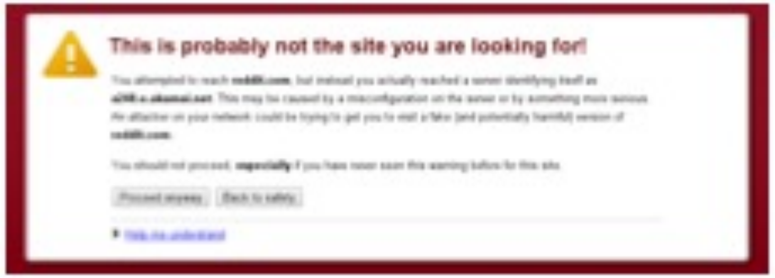
Figure 2: Malware warning for Mozilla Firefox

Similar messages, but…

- Different graphics
- Different language
- Different warning bypass workflows

SSL warnings

- Both messages attempt to explain in layman’s terms (although in different ways) the issues potentially related to SSL certificates that can’t be validated
- Warnings can be sign of man-in-the-middle-attacks, but also of a variety of benign issues (e.g. expired certificates)

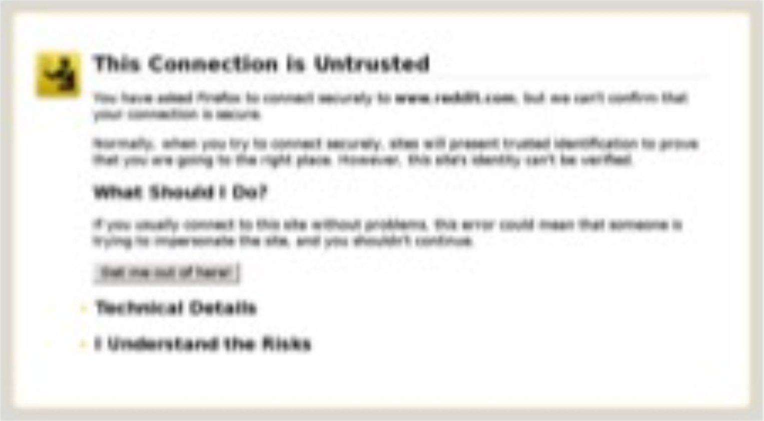


This is probably not the site you are looking for!

You attempted to reach **reddit.com**, but instead you actually reached a server identifying itself as **a248.e.akamai.net**. This may be caused by a misconfiguration on the server or by something more serious. An attacker on your network could be trying to get you to visit a fake (and potentially harmful) version of **reddit.com**.

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

Figure 3: SSL warning for Google Chrome. The first paragraph changes depending on the specific SSL error.



This Connection is Untrusted

You have asked Firefox to connect securely to **www.reddit.com**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Figure 4: SSL warning for Mozilla Firefox

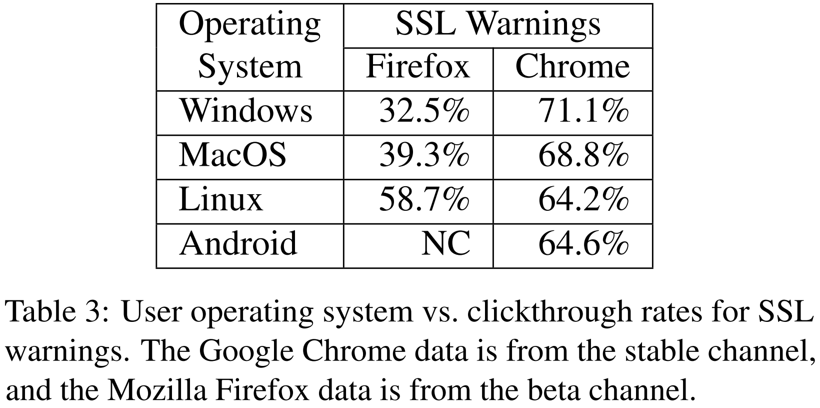
Experimental methodology

- Modern browsers (Firefox, Chrome) include instrumentation that captures various aspects of user behavior
  - “Telemetry framework”
  - Includes timing information
- This instrumentation is used to determine whether users heed or ignore warnings of suspicious situations
  - Both browsers use Google SafeBrowsing API to detect malicious URLs and present warnings to the user
  - Look and feel of warning messages differ between browsers

Experimental methodology/2

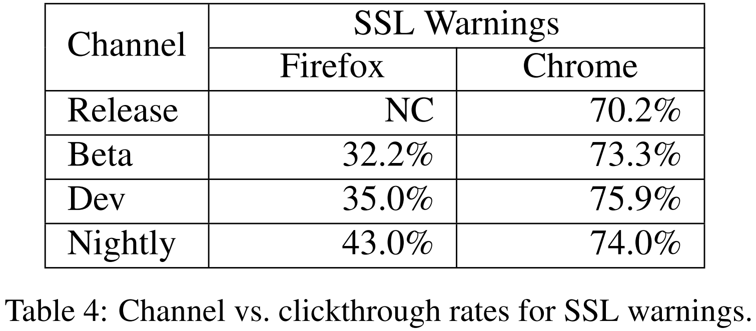
- Data collection resulted in 25M warnings
- Data collected between May and June 2013
- Some data comes from pre-release browser versions (alpha/beta)
    - Authors assume users of these version have technical skills somewhat above average, although there is no data substantiating this intuition

Some interesting results (to be put in context)



Operating System	SSL Warnings	
	Firefox	Chrome
Windows	32.5%	71.1%
MacOS	39.3%	68.8%
Linux	58.7%	64.2%
Android	NC	64.6%

Table 3: User operating system vs. clickthrough rates for SSL warnings. The Google Chrome data is from the stable channel, and the Mozilla Firefox data is from the beta channel.



Channel	SSL Warnings	
	Firefox	Chrome
Release	NC	70.2%
Beta	32.2%	73.3%
Dev	35.0%	75.9%
Nightly	43.0%	74.0%

Table 4: Channel vs. clickthrough rates for SSL warnings.

Note that Firefox users have to perform three clicks to bypass a warning; Chrome users, only one

That’s all for today!