

User Authentication

Basic Concepts and Passwords

Budi Arief

b.arief@kent.ac.uk

Based on slides by Shujun Li

Overview of My Lectures

Please Note: The first lecture in Week 17 (which was originally scheduled for Monday 25 November 2024) has been rescheduled to Thursday 28 November 2024, 10:00-11:00 (which was originally the slot for the second lecture in Week 17). Subsequently, the second lecture in Week 17 will be held later in the afternoon of the same day (Thursday 28 November 2024, 14:00-15:00).

Week	Date	Lecture Topic
Week 16	18 Nov 2024 Monday	Lecture 15: User Authentication (Basic Concepts and Passwords)
	21 Nov 2024 Thursday	Lecture 16: User Authentication (Cracking Textual Passwords and Potential Improvement on Textual Password)
Week 17	28 Nov 2024 Thursday	Lecture 17: User Authentication – Beyond Passwords (1)
	28 Nov 2024 Thursday	Lecture 18: User Authentication – Beyond Passwords (2)
Week 18	02 Dec 2024 Monday	Lecture 19: Non-User Authentication
	05 Dec 2024 Thursday	Lecture 20: Access Control and Authorisation (1)
Week 19	09 Dec 2024 Monday	Lecture 21: Access Control and Authorisation (2)
	12 Dec 2024 Thursday	Lecture 22: Accountability

- Basic Concepts
- (Textual) Passwords
- Summary

What is (entity) authentication?

- Linguistic origin
 - From Greek: αὐθεντικός authentikos, “real, genuine”, from αὐθέντης authentes, “author” (source: [Wikipedia](#))
- Technical definition
 - “the act of proving an assertion, such as the identity of a computer system user” (source: [Wikipedia](#))
- Types (according to the entity type)
 - **User authentication**
 - **Physical product/object** authentication (1/2-D barcodes, RFID chips, security hologram labels, watermarks, ...)
 - **Message** authentication (what does MAC stand for?)
 - **Device / Server** authentication
 - **Humanness** authentication (CAPTCHA)
 - ...



Authentication: Entity vs Identity



- Entity vs Identity: a many-to-many relationship
 - One entity can have multiple identities.
 - One identity can be allocated/claimed/used by multiple entities.
 - An **identity management system** or an **identity provider** (IdP/IDP) is often used to create, maintain and manage identities, including mappings to entities.
- Identity vs Identifier (ID)
 - Identities are normally identified via a **unique identifier** to avoid ambiguity in the authentication process.
- Real vs Virtual
 - Example: a person's real name vs a person's pen name
- Physical vs Electronic
 - Example: a person's real name vs a person's email address

Authentication vs Identification

- Authentication

- An entity makes an explicit claim.
- A verifier checks if the claim is legitimate.
- An identity is often used to facilitate the authentication process. \Rightarrow authentication = verification of a claimed identity
- The claimant normally needs to provide some proof to allow the authentication check. \Rightarrow claimant = prover



- Identification

- Nobody makes a claim explicitly.
- An entity is present with an unknown identity.
- An identifier tries to uncover the present entity's identity.
- Some people (including some researchers) mix the two terms up, so be careful when reading.



User authentication



- Authentication \Rightarrow User authentication
 - When the claimant is a (human) user.
 - The verifier is normally a computer (at least in our context), but can be another (human) user.

- 3 main (traditional) user authentication factors
 1. **Knowledge**-based: “What you know” (e.g., passwords)
 2. **Possession**-based: “What you possess / have” (e.g., hardware security tokens and smart cards)
 3. **Inherence**-based: “Who you are” (including “How you behave”) = Biometrics

Context-based user authentication



- Another authentication factor
 - **Context**-based: “Where you are”
 - Here, “where” is broader than just geo-location, and can include other environmental factors such as information from other sensors.
 - Location information can be relative, e.g., indoors, on board of a train or a flight.
- Raghav V. Sampangi and Kirstie Hawkey, "[Who Are You? It Depends \(On What You Ask Me!\): Context-Dependent Dynamic User Authentication](#),” WAY 2016 (co-located at SOUPS 2016), USENIX Association
- Michael Kirkpatrick and Elisa Bertino, "[Context-Dependent Authentication and Access Control](#),” iNetSec 2009, Springer
- Nishanth Chandran, Vipul Goyal, Ryan Moriarty and Rafail Ostrovsky, "[Position Based Cryptography](#),” CRYPTO 2009, Springer

Risk-based user authentication


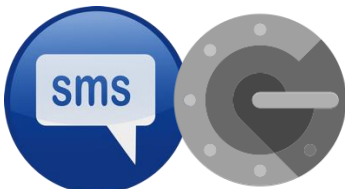


- Another authentication factor
 - **Risk**-based: “What your risk profile is”
 - Multiple risk factors around login behaviours, including contextual information!
 - Normally used to as the first layer of user authentication, e.g., the following rules may be used:
 - Low risk: no authentication or simpler authentication
 - Medium risk: standard authentication
 - High risk: multi-factor authentication (MFA, see Slide 10)
 - Less visible to end users, but widely adopted.
- Stephan Wiefeling, Luigi Lo Iacono and Markus Dürmuth, “Risk-based Authentication,” <https://riskbasedauthentication.org/>, last updated in 2020

- Another authentication factor
 - **Social authentication:** “Whom you know”
 - Authentication using your friends
 - Often used for **fallback/recovery/emergency authentication** when the password is forgotten.
 - Your friends provide endorsements to your request of recovering your identity.
 - Used by some online social media and instant messaging applications.
- John Brainard, Ari Juels, Ronald L. Rivest, Michael Szydlo and Moti Yung, [“Fourth-Factor Authentication: Somebody You Know,”](#) CCS 2006, ACM
- Stuart Schechter, Serge Egelman and Robert W. Reeder, [“It’s Not What You Know, But Who You Know: A social approach to last-resort authentication,”](#) CHI 2009, ACM



Multi-factor authentication (MFA)

- Two or more authentication factors
 - The [EU PSD2 \(European Payment Services Directive 2\)](#) prescribes two-factor authentication (2FA) for online payments (with a few exceptions). 
 - Many online services now support MFA, mostly by combining passwords and a mobile phone based factor (SMS, authenticator mobile app). 
 - Some UK universities start doing the same.
- Pros and cons
 - We will come to this later after we finish looking at all user authentication factors.

Continuous authentication vs De-authentication

- Continuous authentication

to be continued

- Once a user is authenticated, the verifier (system) can continuously check the behaviour of the user for authentication purposes.
- Often uses **inherence-based (behavioural biometrics)** and **risk-based** authentication techniques.

- De-authentication



- When it is detected that the “current” user does not seem to be the authenticated one (of a **high risk**), the user will be de-authenticated (logged out) and **re-authentication** is requested.
- Achieved via **continuous authentication**.

“In-class” exercises



- For the following scenarios, consider what authentication-related terms are applicable.
 - Type a secret word to log in.
 - Say a secret word to log in.
 - Say anything for 30 seconds to log in.
 - Type a one-time-password received from SMS to log in.
 - Your employer's system does not allow you to log in if you are not working at office.
 - The system automatically identify you as the regular user from your IP address and let you in directly.

- Basic Concepts
- (Textual) Passwords
- Summary

What is a password?



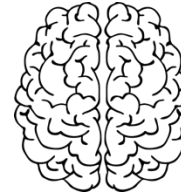
- Narrowly speaking
 - A **secret textual string** you present for accessing some protected resource(s).
- Broadly speaking: 1st authentication factor
 - “What you know” = “What you (have to) remember”
 - Examples
 - PIN (Personal Identification Number): digits only
 - Textual passwords: digits + letters + special chars
 - Pass-phrases
 - Graphical passwords: graphical elements
 - ...

How are passwords stored?



- At the user side

- In your brain
- On a piece of paper
- On your device (e.g., mobile phone)
- On a USB key
- In a password manager **LastPass ******
- Calculated from something you know
- ...



- What is stored may not be the password, but just a hint.

How are passwords stored?



- At the server side
 - In clear (should never happen, but ...): **password**
 - Hashed directly without any other additional data: **$H(\text{password})$**
 - Hashed with a random number (called a salt): **$H(\text{password} || \text{salt})$**
 - Stored using advanced key stretching (password strengthening) methods, e.g., **$H^n(\text{password} || \text{salt})$** , where n is large.
 - Honeytokens (**fake passwords**) may be used.
- Why **hashing**?
 - Avoid password leakage if the server is hacked.
- Why **salting** (on top of hashing)?
 - Fact: salt is random but stored in clear at server side
 - For resisting rainbow table (a pre-calculated table of possible passwords and their hash values) attacks
 - Play a role similar to cryptographic nonce



Password hashing: user registration

- User provides the following information to a system for registering an account:
 - Username (unique ID): u
 - Password (in clear): p
 - Other profile information: PI
- Server receives the above and does the following:
 - Generate a new and random salt s for this user.
 - Calculate the hash value as $h = H^n(p \parallel s)$.
 - Store (u, s, h, PI) as a new record in the user database.

Password hashing: login

- User provides the following information to a system for log into an account:
 - Username: u
 - Password (in clear): p^*
- Server receives the above and does the following:
 - Use u to retrieve the user's record (u, s, h, PI) in the user database
 - Re-calculate the hash value as $h^* = H^n(p^* || s)$.
 - Compare h and h^* : if they are equal then let the user in; otherwise reject the user.

An example

- Hash function: SHA-256
- Password: password1234 (chosen by the user)
- Salt: NYcb52ZP (a server-generated random string)
- $n=2$
- What is stored on the server side (hexadecimal format):
 1. $H(\text{password1234NYcb52ZP}) =$
 0xe585ae12d1ae68681d705f81f6ec5e0b1c012815b10db9e0f9a0
 0f6869051c0d
 2. $H(0xe585ae12d\dots1c0d) =$
 0xd680727884034387ab565b4b16225a39c39b707b2f12e78184
 0a4a9a47081826

\Rightarrow ("NYcb52ZP",
 0xd680727884034387ab565b4b16225a39c39b707b2f12e781840a4
 a9a47081826) is stored on the server side.

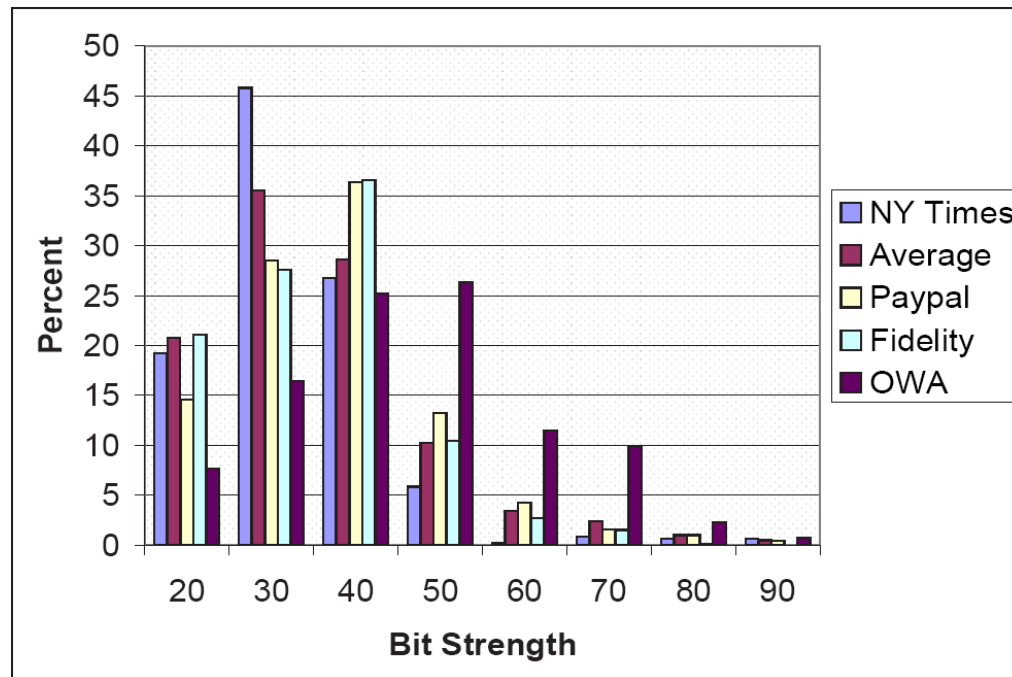
How many passwords are there?

- 4 digits (PINs): $10^4=10$ thousand $\approx 2^{13.3}$
- 6 digits (PINs): $10^6=1$ million $\approx 2^{20}$
- Lowercase letters only, 7 characters: $26^7 \approx 8$ million $\approx 2^{33}$
- Lowercase letters + digits, 7 characters: $36^7 \approx 78.4$ million $\approx 2^{36}$
- Lowercase & uppercase letters + digits, 7 characters: $62^7 \approx 10$ trillion $\approx 2^{42}$
- Lowercase & uppercase letters + digits, 11 characters: $62^{11} \approx 52$ quintillion $\approx 2^{65.5}$

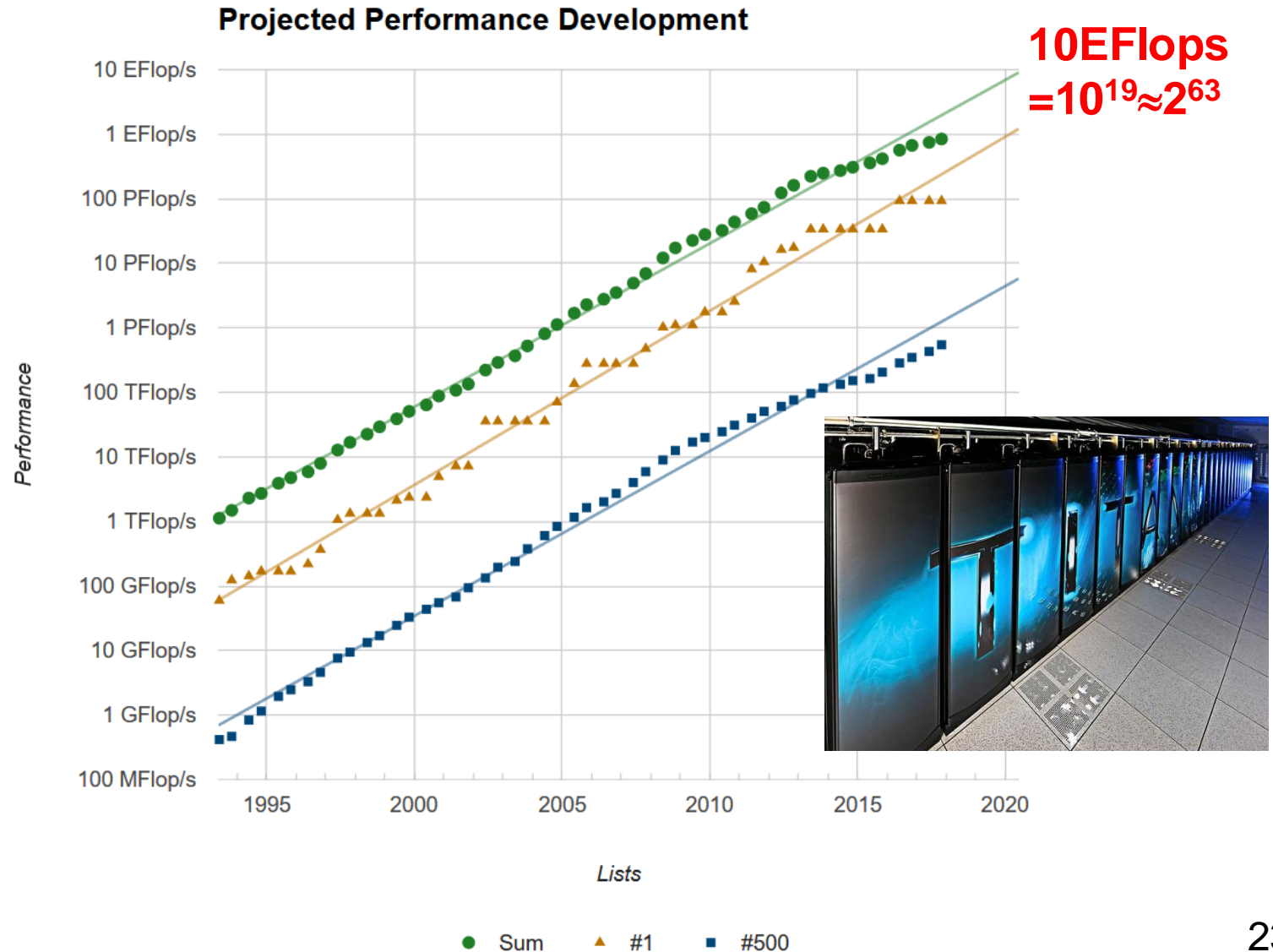


What passwords are/were being used?

- Dinei Florêncio and Cormac Herley, [“A Large-Scale Study of Web Password Habits,”](#) WWW 2007, W3C/ACM
 - Real passwords collected from 544,960 web users in three months in 2006.



How fast are today's supercomputers?



What passwords are/were being used?

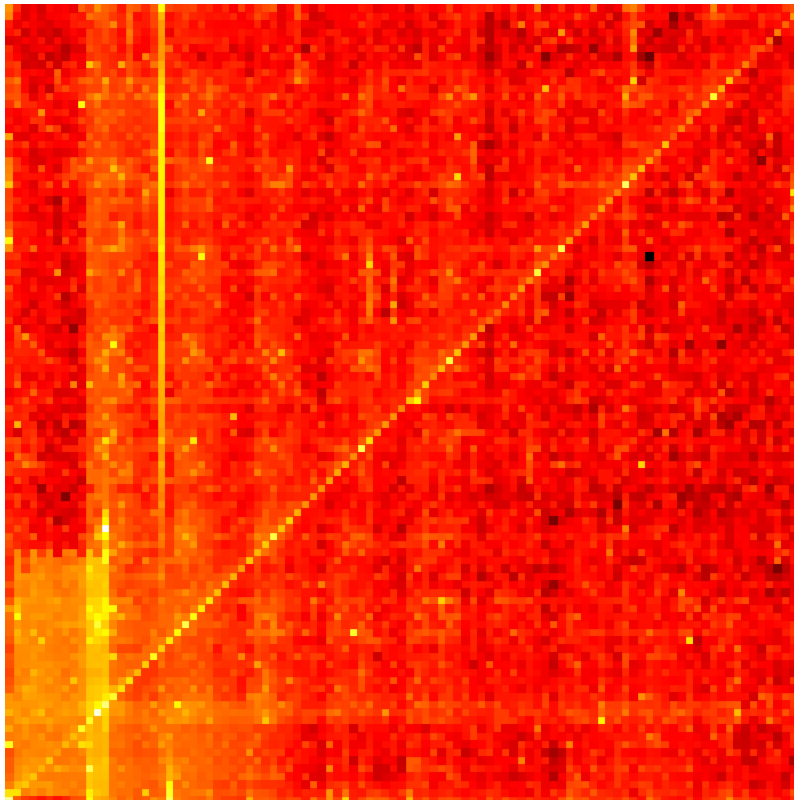
- Things we can remember. ([Link of the video](#))



What PINs are/were being used?

- A 2012 visual analysis of 3.4 million leaked passwords composed of 4 digits (PINs)

00xy



9999

“In-class” exercise

- Identify all obvious patterns you can visually identify
- Explain each identified pattern



xy00

Book now!

interesting
story

Select losses greater than 30,000 records
Last updated: 11th June 2020



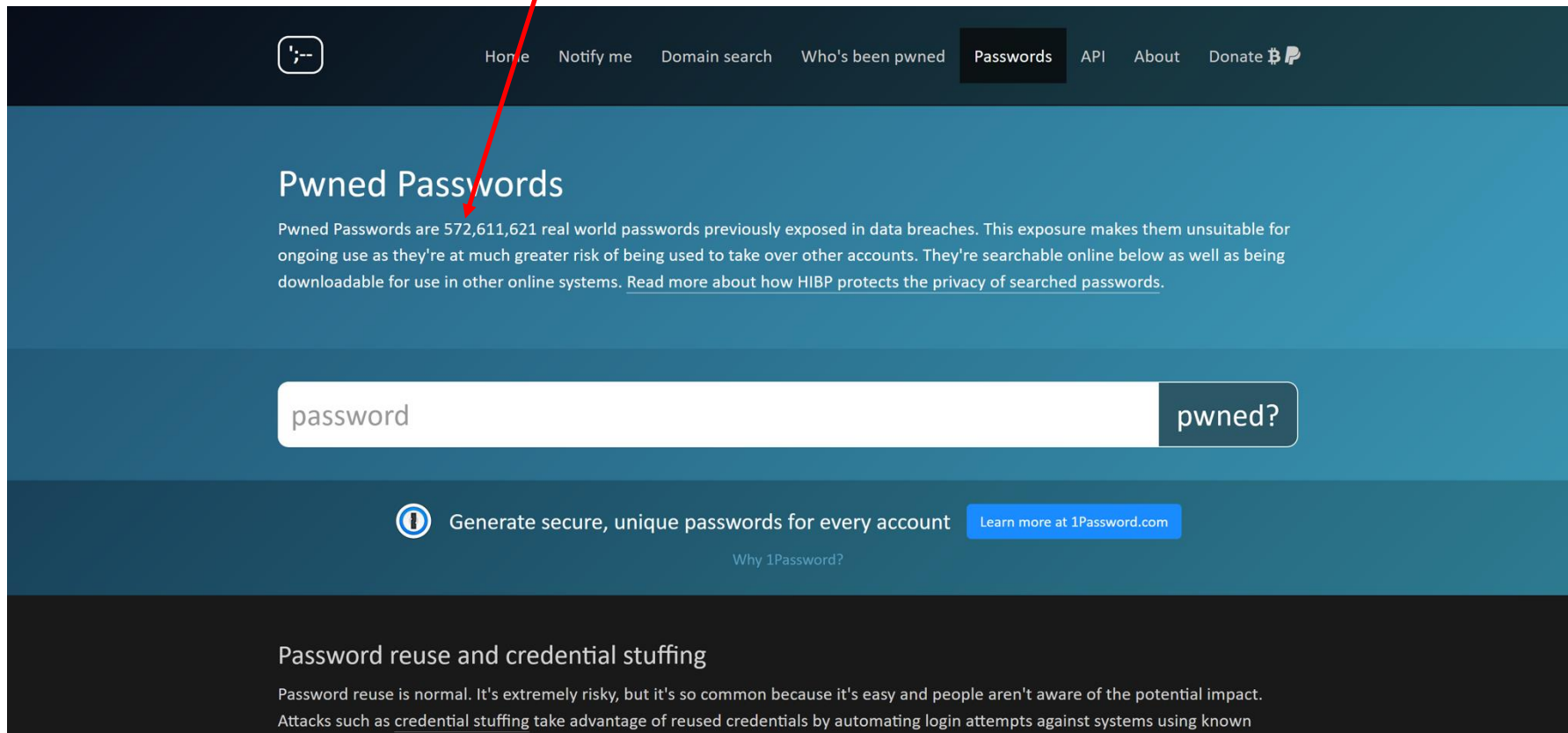
haveibeenpwned.com (2013-)




University of
Kent

[Troy Hunt](#)

572,611,621 real world passwords




Home Notify me Domain search Who's been pwned **Passwords** API About Donate 

Pwned Passwords

Pwned Passwords are 572,611,621 real world passwords previously exposed in data breaches. This exposure makes them unsuitable for ongoing use as they're at much greater risk of being used to take over other accounts. They're searchable online below as well as being downloadable for use in other online systems. [Read more about how HIBP protects the privacy of searched passwords.](#)

password pwned?

 Generate secure, unique passwords for every account [Learn more at 1Password.com](#)

[Why 1Password?](#)

Password reuse and credential stuffing

Password reuse is normal. It's extremely risky, but it's so common because it's easy and people aren't aware of the potential impact. Attacks such as [credential stuffing](#) take advantage of reused credentials by automating login attempts against systems using known

- Basic Concepts
- (Textual) Passwords
- Summary

Summary

- We looked at some basic concepts
 - Identification, authentication, various authentication techniques and authentication factors
- Textual passwords
 - How they are created and stored
 - Textual passwords weaknesses
- Next lecture
 - Cracking textual passwords
 - Potential improvement on textual password