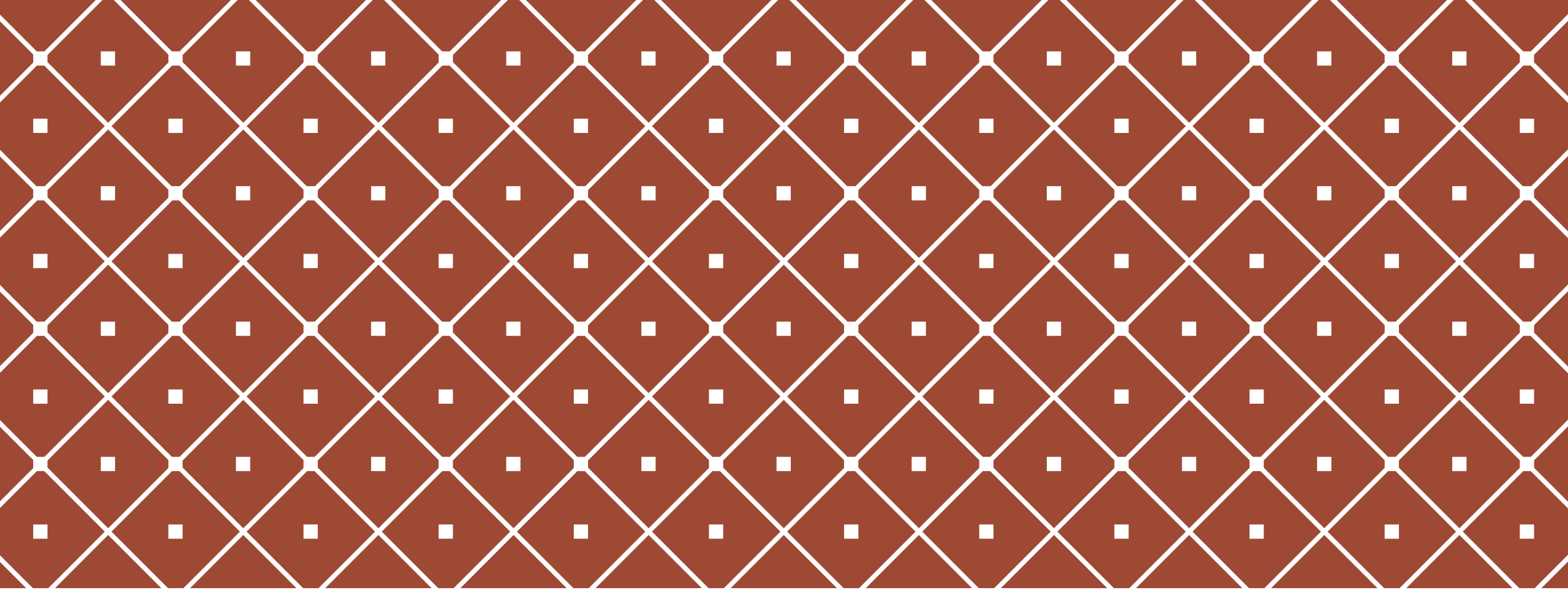


THE ONLY HUMAN FACTOR: FORMAL METHODS AND PASSWORD POLICIES

Saul Johnson

WHAT ARE FORMAL METHODS?

When we say “formal methods” we refer to a particular set of techniques focusing on the precise mathematical specification, development and verification of systems.

Sitting down and hacking away at building our software product straight away is a recipe for bugs! Our own intuition about development decisions that “feel right” begins to creep into the project, risking the product becoming unfit for purpose.

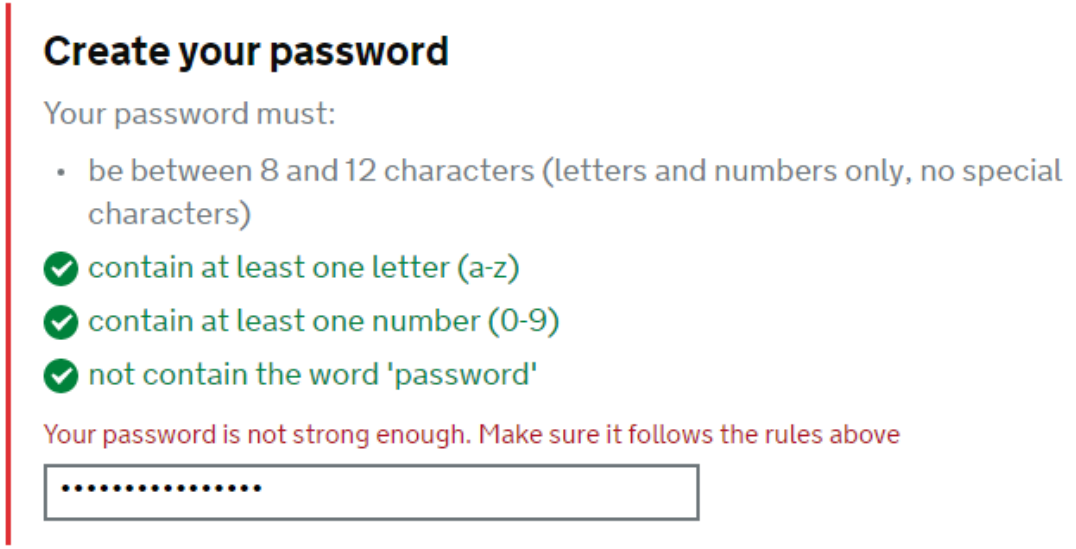
This is especially apparent in password security, particularly in the sets of rules system administrators enforce around password creation, known as **password composition policies**.

PASSWORD POLICIES: YOU KNOW THEM ALREADY!

Password composition policies have become a fact of life when creating/changing passwords, particularly to online accounts.

They are designed with the intention of making password guessing attacks less likely to succeed by encouraging users to choose passwords that are harder to guess, but the vast majority of these policies out there today are unfit for purpose.

The password composition policy behind the form on the right belongs/belonged to HMRC. Is there anything wrong with it?

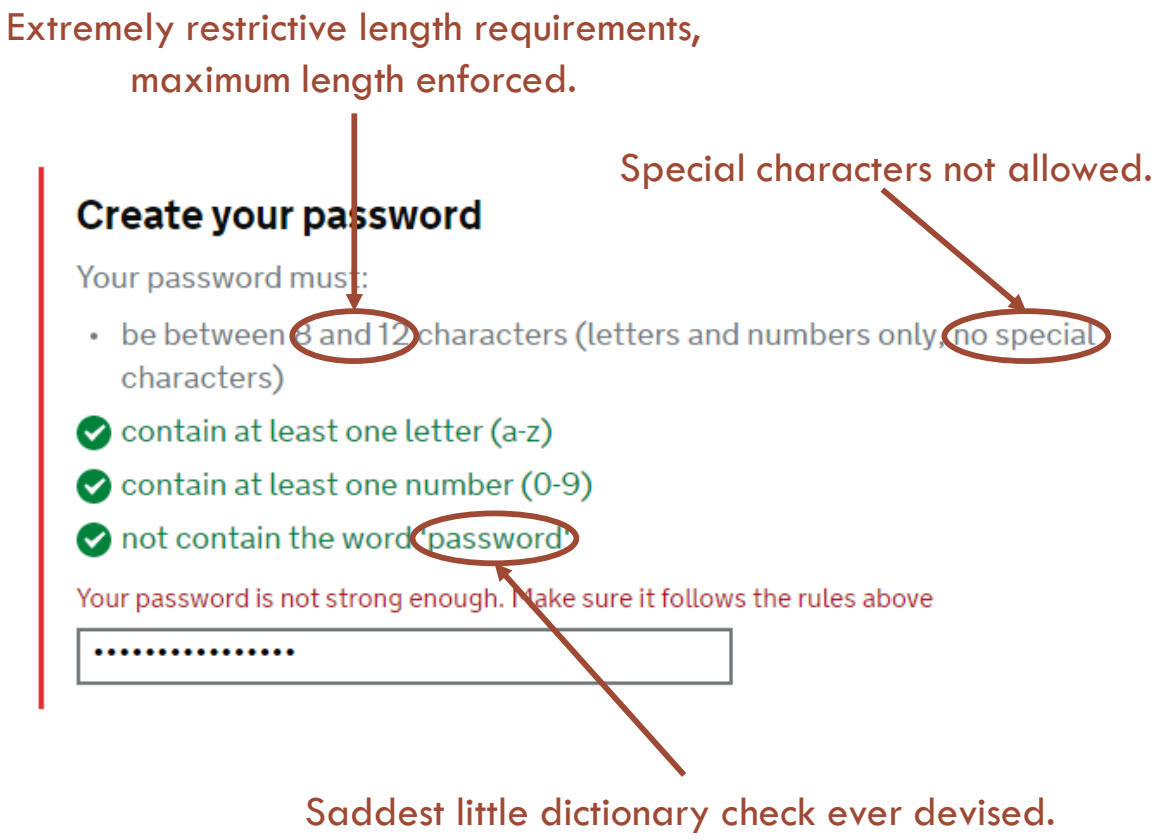


ANSWER: YES, YES THERE CERTAINLY IS

The password space is extremely restricted by the length constraints and limitations on character set. The users have mathematically fewer passwords to choose from.

The most common mistake users make when creating their passwords is to overuse dictionary words. This “dictionary” check prohibits one word.

There is never an excuse for prohibiting passwords containing certain characters or passwords that are “too long”. All passwords should be hashed to a fixed-length string anyway, so why should the website care?

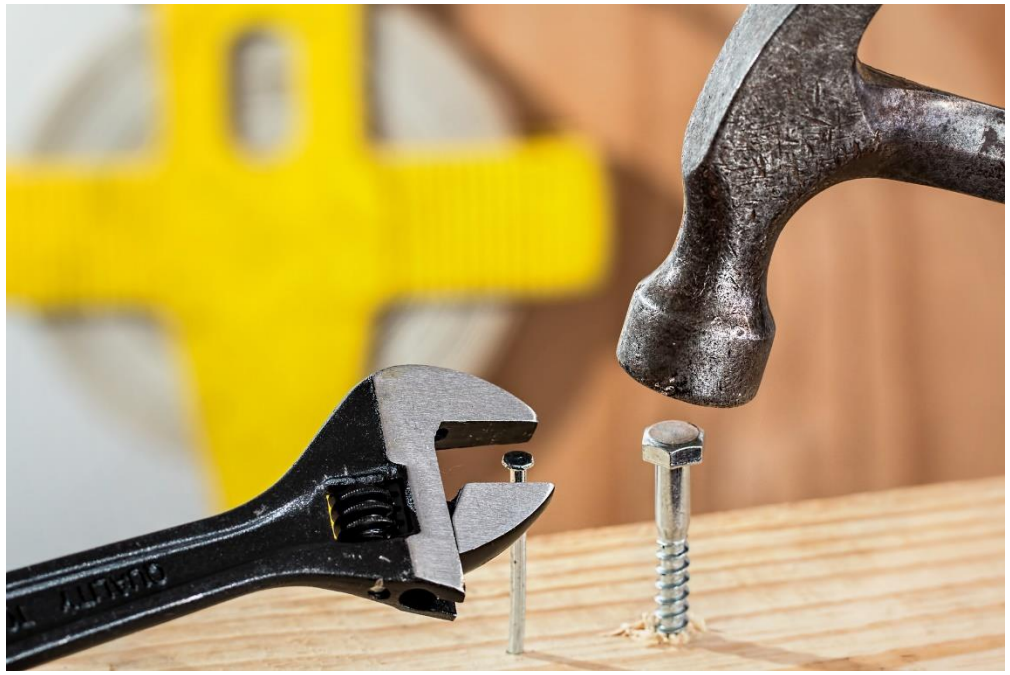


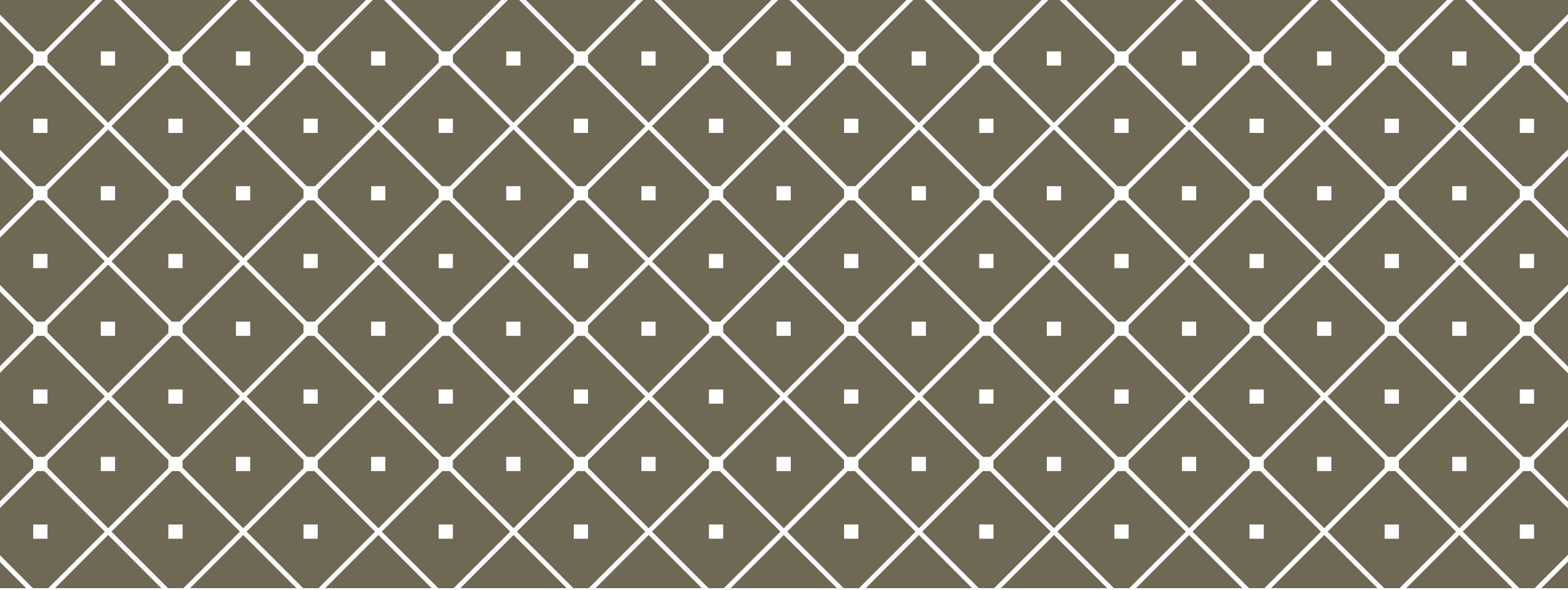
CHOICE AND ENFORCEMENT: RELATED BUT DISTINCT

To effectively employ a password composition policy on a system, we must first **choose** a policy in an informed way, and be able to justify that choice.

Then, we must ensure that this policy is **enforced** correctly on the system.

Our aim is the development of tools that put both of these things **within the reach of a system administrator** with little to no background in password security or formal methods.





VERIFIED PASSWORD COMPOSITION POLICY ENFORCEMENT

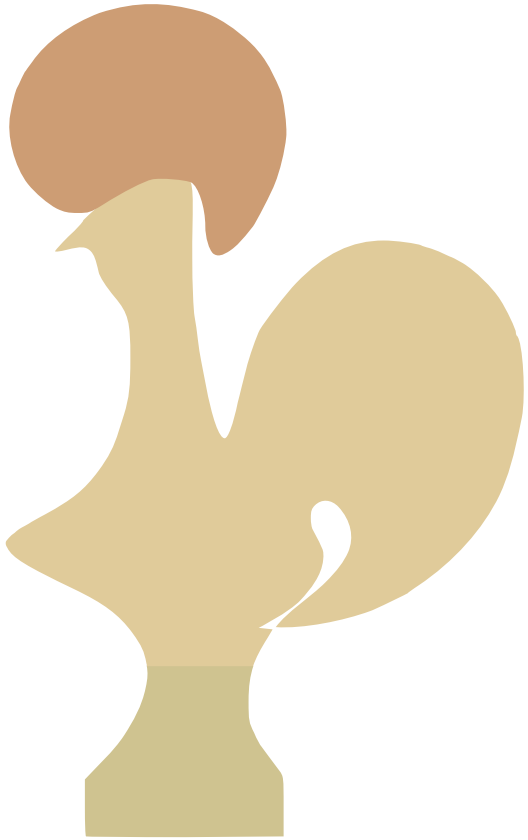
Developing formally verified, machine-checked password composition policy enforcement software.

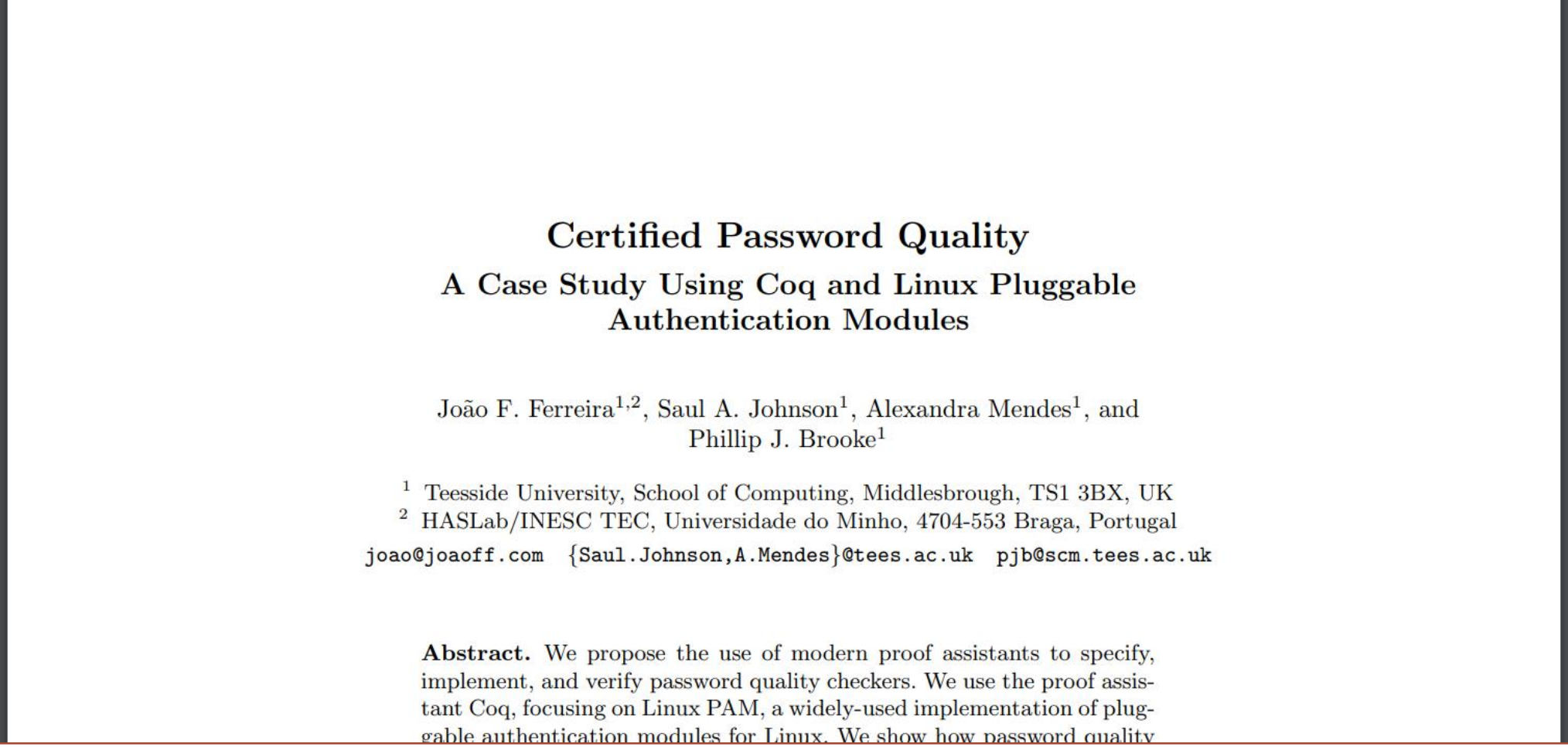
OUR TOOLING: THE COQ PROOF ASSISTANT

We used **the Coq proof assistant** to write software and mathematically prove its correctness. For the parts of our software artifacts that are of critical importance, this is extremely valuable to have!

Where performance or flexibility is more important, we have used Python/Java to develop our tooling (more of this later).

Getting all these tools to work together was **a real challenge**, but it was well worth it!





Certified Password Quality

A Case Study Using Coq and Linux Pluggable Authentication Modules

João F. Ferreira[1,2], Saul A. Johnson[1], Alexandra Mendes[1], and Phillip J. Brooke[1]

[1] Teesside University, School of Computing, Middlesbrough, TS1 3BX, UK
[2] HASLab/INESC TEC, Universidade do Minho, 4704-553 Braga, Portugal

joao@joaoff.com {Saul.Johnson,A.Mendes}@tees.ac.uk pjb@scm.tees.ac.uk

Abstract. We propose the use of modern proof assistants to specify, implement, and verify password quality checkers. We use the proof assistant Coq, focusing on Linux PAM, a widely-used implementation of pluggable authentication modules for Linux. We show how password quality

CERTIFIED PASSWORD QUALITY

Our first paper was presented at iFM 2017 in Turin, Italy. It covers verified password policy *enforcement*.

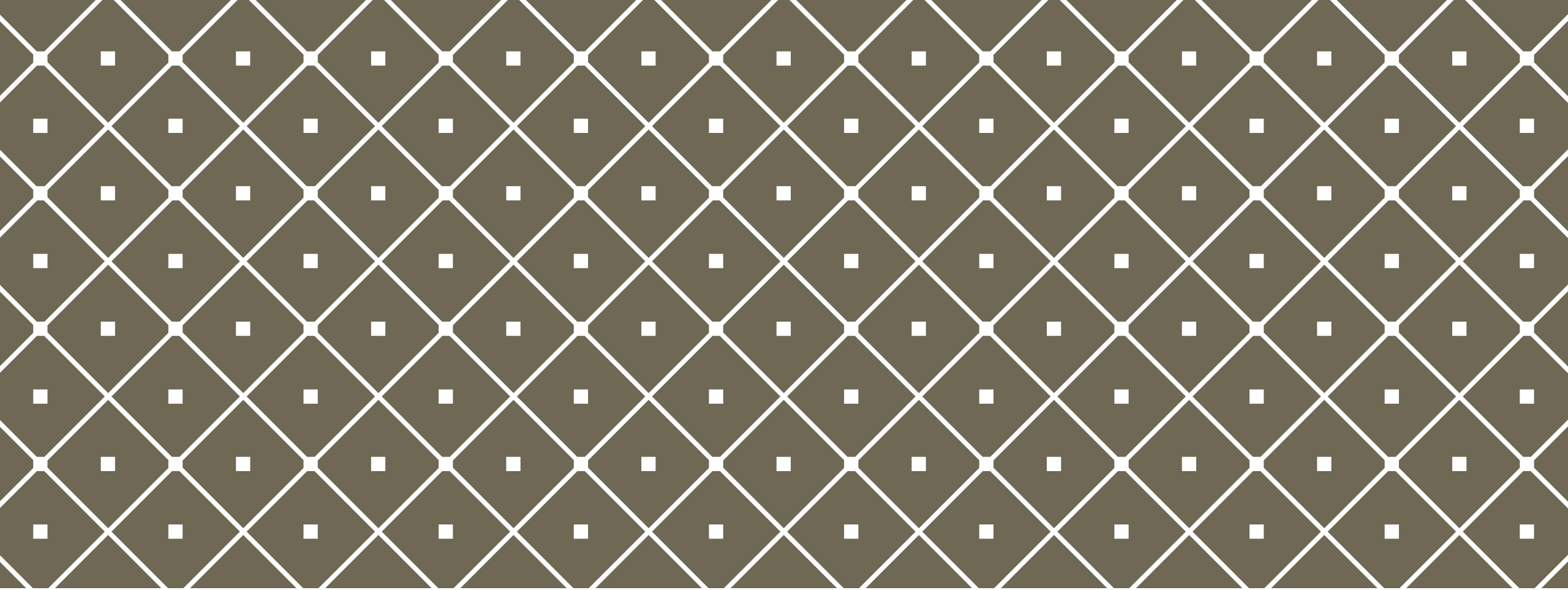
SERENITY: EXTENDING OUR FIRST WORK

Our domain-specific language (DSL) **Serenity** has been in the works for a while now to permit system administrates to build password composition policy enforcement software that is *correct by construction*.

The language is embedded within the Coq proof assistant, and its building blocks are formally verified. It's also intuitive enough that system administrators can straightforwardly express their intended policy with minimal training.

SERENITY: AN EXAMPLE POLICY

```
Definition comprehensive8 :=
  (enforce new_pwd (min length 8)
    "New password must be at least 8 characters long!")
  /*\ (enforce new_pwd (min count_upper 1)
    "New password must contain an uppercase letter!")
  /*\ (enforce new_pwd (min count_lower 1)
    "New password must contain a lowercase letter!")
  /*\ (enforce new_pwd (min count_digit 1)
    "New password must contain a digit!")
  /*\ (enforce new_pwd (min count_other 1)
    "New password must contain a symbol!").
```



JUSTIFIABLE PASSWORD COMPOSITION POLICY SELECTION

Building a system for automatic, justified and privacy-preserving password composition policy choice.

SOURCING QUALITY DATA FROM PASSWORD DATA DUMPS

Large sets of password data, breached by cybercriminals, are available online, with **some containing hundreds of millions of passwords**.

It's **vital** that researchers look at these in order to understand user password choice and advance password security research. **Only by examining real passwords can we understand how to secure real systems.**

These aren't always clean though, some non-password tokens are usually present that aren't compliant with the password policy in place on the system at the time the passwords were stolen.

Dump	Policy	Size	Invalid
RockYou	Length > 5	≈32.6m	0.24%
000webhost	Length > 6 Digits > 1	≈15.2m	2.19%
Yahoo	Length > 6	≈453.5k	1.89%
LinkedIn	Length > 6	≈172.4m	0.01%

Lost in Disclosure: On The Inference of Password Composition Policies

Saul Johnson*, João F. Ferreira†, Alexandra Mendes‡ and Julien Cordry*
*Software and Systems Research Group, Teesside University, Middlesbrough, UK
†Instituto Superior Técnico, University of Lisbon and INESC-ID, Lisbon, Portugal
‡HASLab, INESC TEC and Computer Science Department, University of Beira Interior, Covilhã, Portugal
Email: *{saul.johnson,j.cordry}@tees.ac.uk, †joao@joaoff.com, ‡alexandra@archimendes.com

***Abstract*—Large-scale password data breaches are becoming increasingly commonplace, which has enabled researchers to produce a substantial body of password security research utilising real-world password datasets, which often contain numbers of records in the tens or even hundreds of millions. While much study has been conducted on how password composition policies—sets of rules that a user must abide by when creating a password—influence the distribution of user-chosen passwords on a system, much less research has been done on inferring**

unwilling to disclose any information regarding their security practices. Reasons for this might include, for example:

- The organisation may have ceased to exist entirely, prior to the time at which the research in question is being conducted. There are several examples of this happening in the real world, for example the now-defunct Christian dating site *singles.org* [5] which ceased to exist sometime after 2009 when their entire user credential database was

ON THE INFERENCE OF PASSWORD COMPOSITION POLICIES

This work arose as a consequence of our pursuit of clean password data collected under a known password composition policy. Accepted at RSDA 2019.

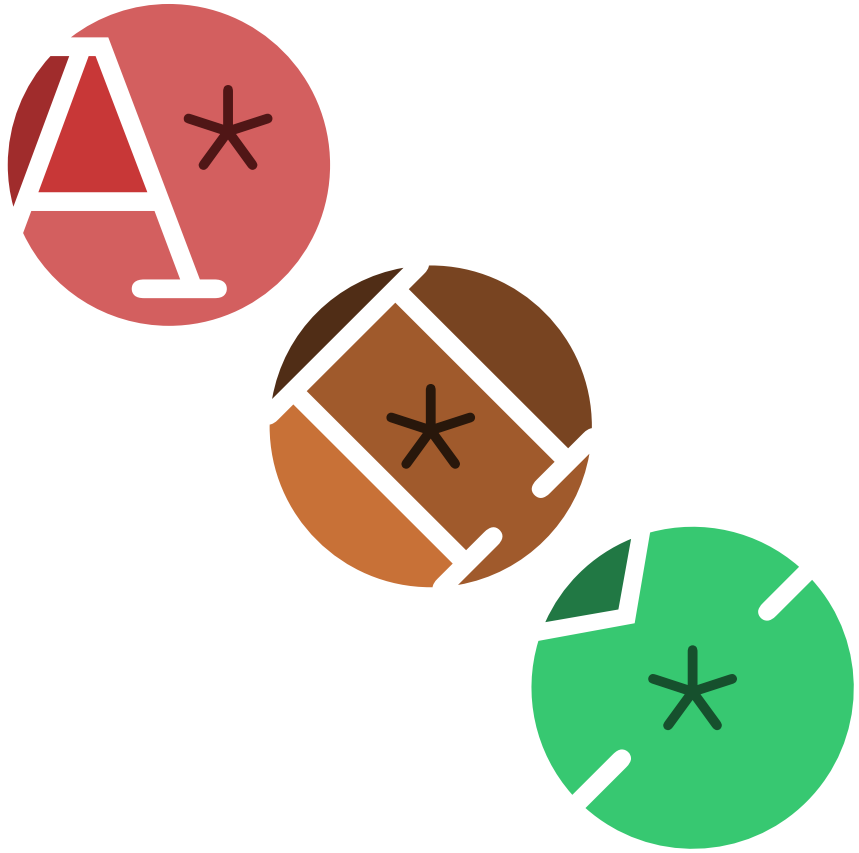
SKEPTIC: AUTOMATIC, JUSTIFIED AND PRIVACY-PRESERVING PASSWORD POLICY CHOICE

Our project culminates in **Skeptic**, a 3-part system for automatically choosing a password composition policy.

Authority: A verified core written in Coq that filters a password data dump according to a policy.

Pyrrho: A user behaviour model that simulates users choosing different passwords in response.

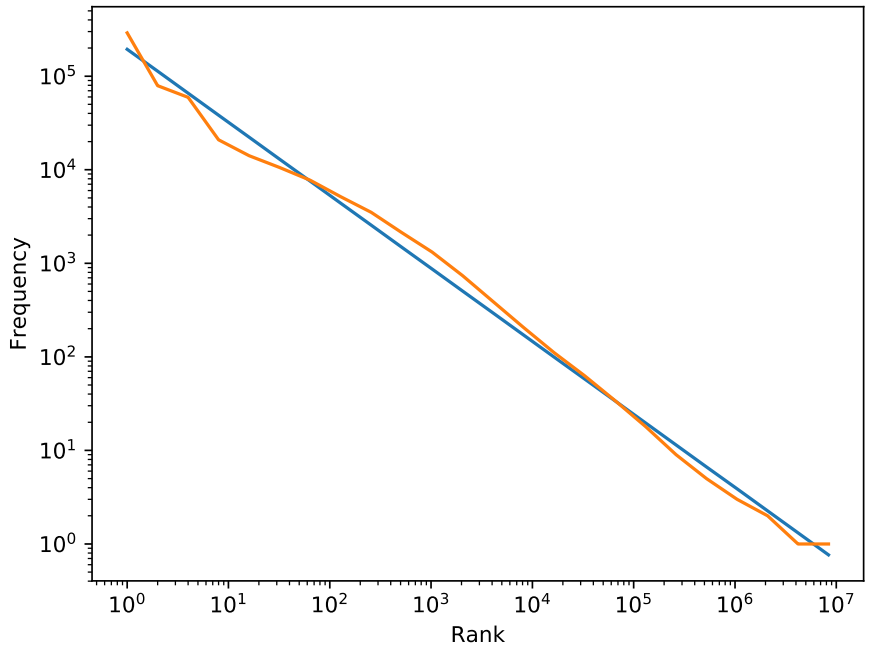
PaCPAL: **Pa**ssword **c**omposition **p**olicy **a**ssertion **l**anguage. A DSL that allows system administrators to easily extract results from this data automatically.



WE DON'T NEED TO SEE USER PASSWORDS!

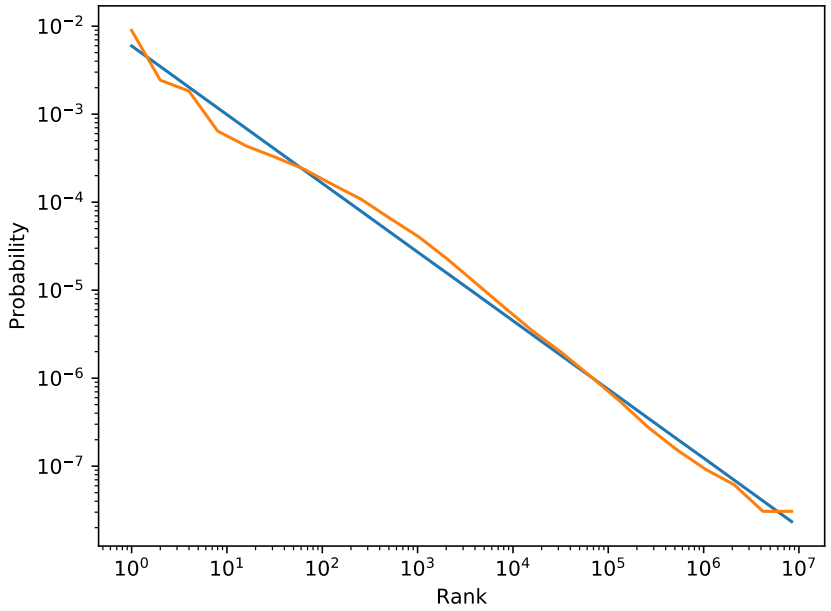
A key finding so far in our work is that we don't actually need to have access to user passwords in order to justify our password policy choice. This is great for avoiding the propagation/sharing of password data and preserving user privacy.

Password distributions tend to follow Zipf's law. A few passwords are chosen very often, and many password are chosen rarely, with an exponential fall-off. This means all we need to rank password policies is the **equations that fit the distributions they induce**. See the blue line on the right.

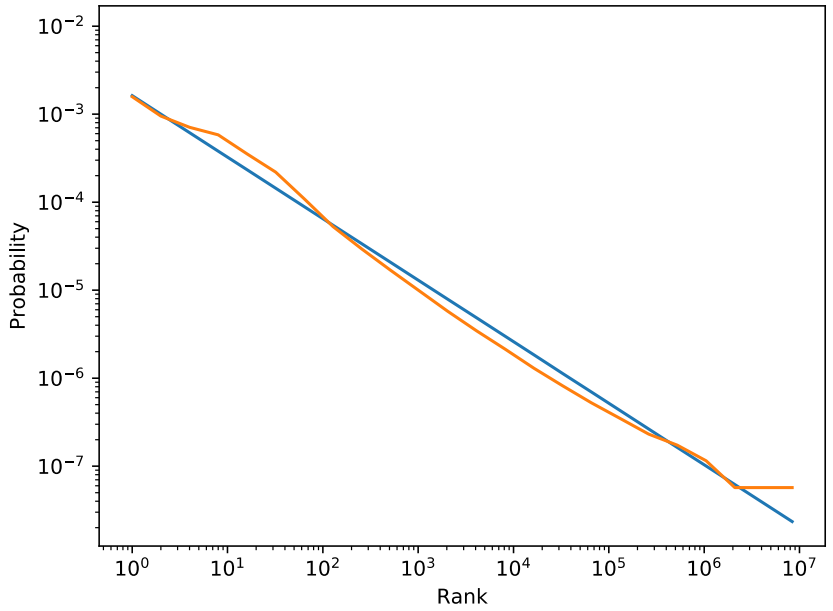


MORE DIVERSE PASSWORDS, MORE SECURE SYSTEM!

Weaker Policy: Steeper Curve/Less Uniform Distribution (Length 5)



Stronger Policy: Shallower Curve/More Uniform Distribution (Length 6, 1 Digit)



Skeptic: Automatic, Justified and Privacy-Preserving Password Composition Policy Selection

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

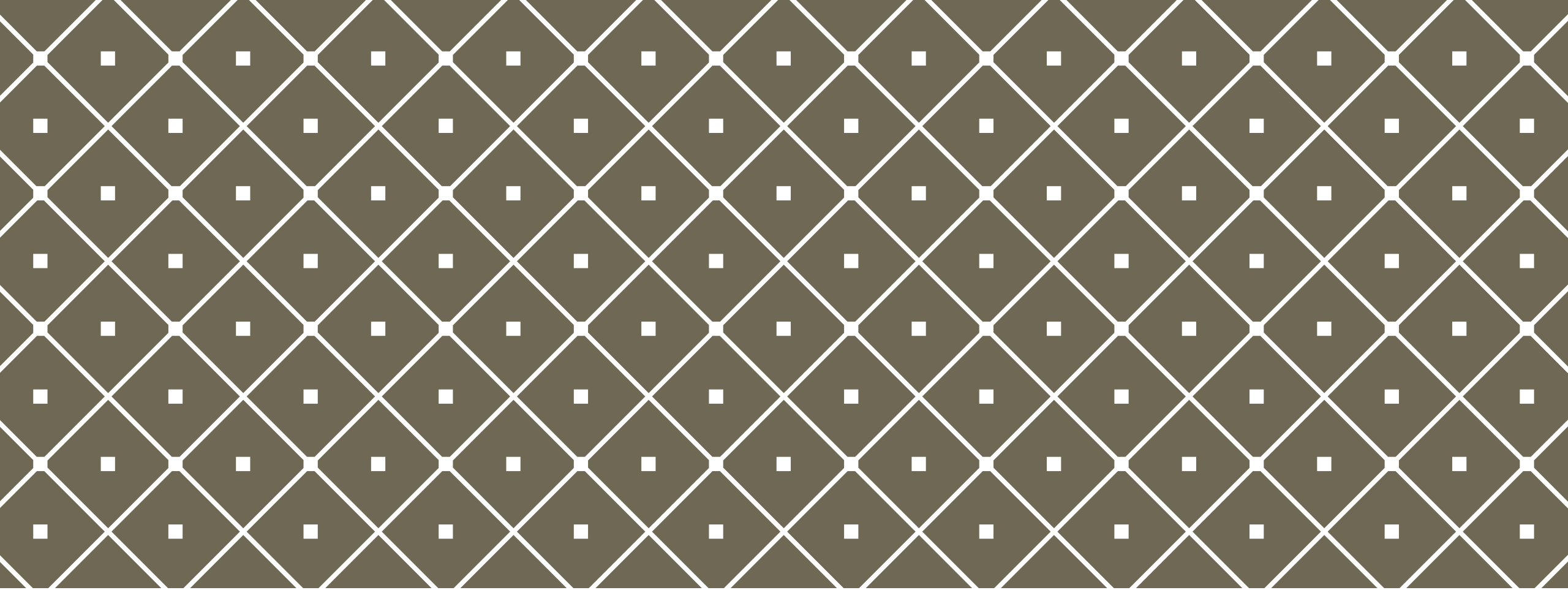
***Abstract*—The choice of password composition policy to enforce on a password-protected system represents a critical security decision, and has been shown to significantly affect the security of user-chosen passwords. In practice however, this**

the value of the assets it protects is somewhat alarming [2], and makes a strong case for a more rigorous method of selection.

While much study to date has been conducted on how password composition policies affect the security of the distri-

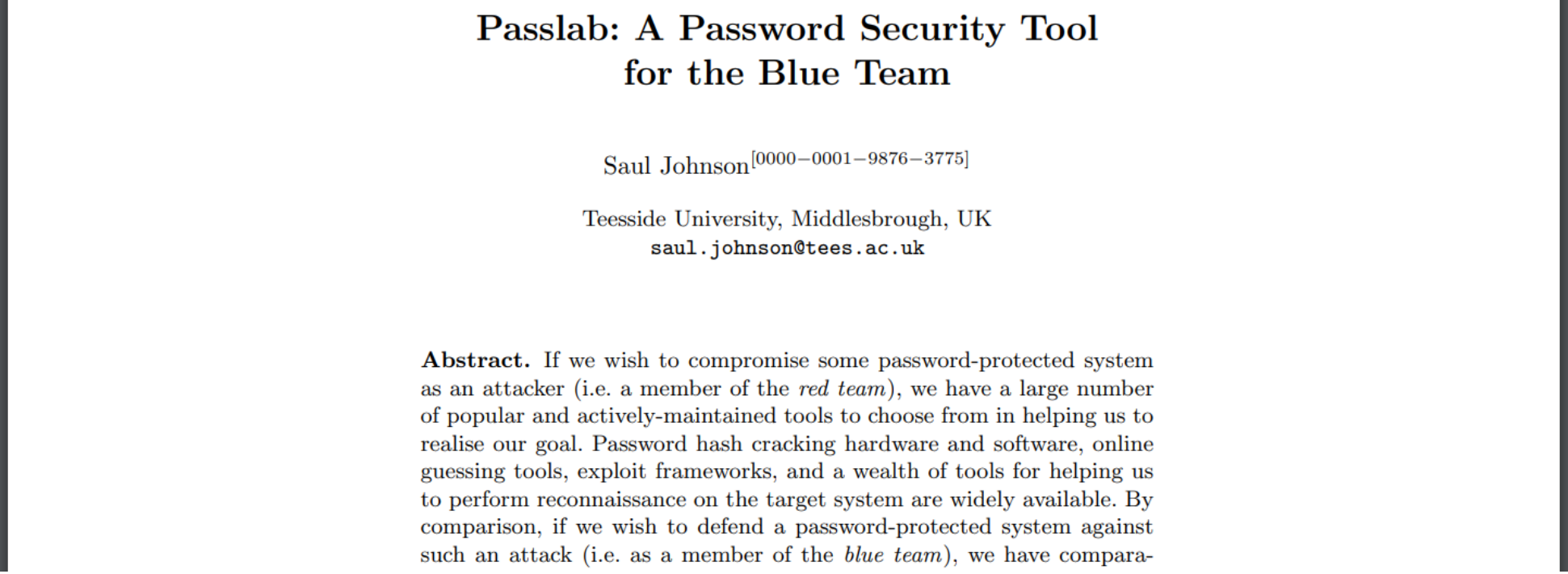
SKEPTIC: AUTOMATIC, JUSTIFIED AND PRIVACY-PRESERVING PASSWORD COMPOSITION POLICY SELECTION

This work has yet to be submitted, and represents the culmination of almost all of our work so far.



PULLING IT ALL TOGETHER: A USER-FRIENDLY TOOL

Making all this accessible to system administrators. With a live demo, **if we have time!**



Passlab: A Password Security Tool for the Blue Team

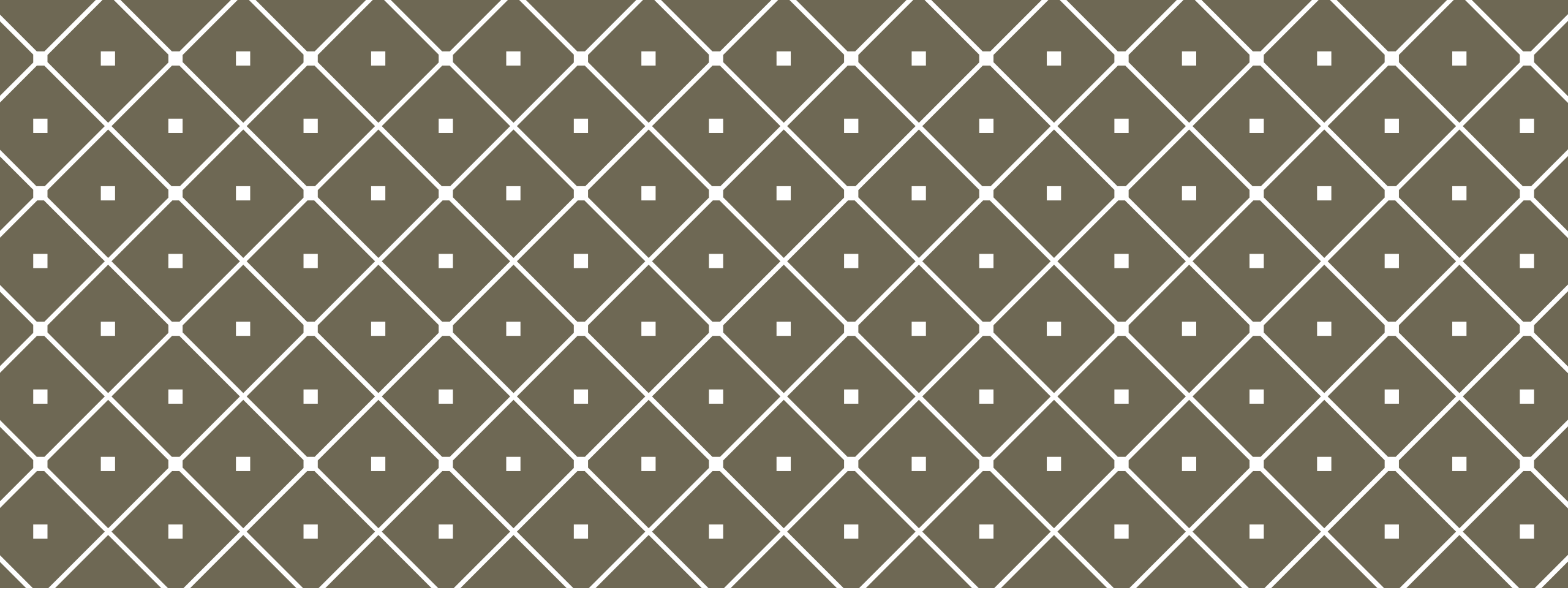
Saul Johnson[0000−0001−9876−3775]

Teesside University, Middlesbrough, UK
saul.johnson@tees.ac.uk

Abstract. If we wish to compromise some password-protected system as an attacker (i.e. a member of the *red team*), we have a large number of popular and actively-maintained tools to choose from in helping us to realise our goal. Password hash cracking hardware and software, online guessing tools, exploit frameworks, and a wealth of tools for helping us to perform reconnaissance on the target system are widely available. By comparison, if we wish to defend a password-protected system against such an attack (i.e. as a member of the *blue team*), we have compara-

PASSLAB: A PASSWORD SECURITY TOOL FOR THE BLUE TEAM

Passlab is a proposed piece of software for pulling together the Skeptic framework into an intuitive UI. This extended abstract was accepted at DSFM 2019.



THANK YOU! WHAT QUESTIONS DO YOU HAVE?

I'm also happy to talk offline if you would like to know more about my work in detail.