Asymmetric PSI and Its Leakage:

A Case Study of the MIGP Protocol

GEORGE MASON UNIVERSITY

Evgenios M. Kornaropoulos

https://encrypted.systems

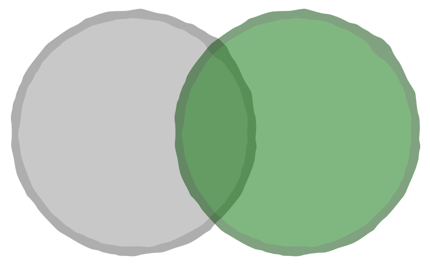
Joint work with:

Dario Pasquini
George Mason University

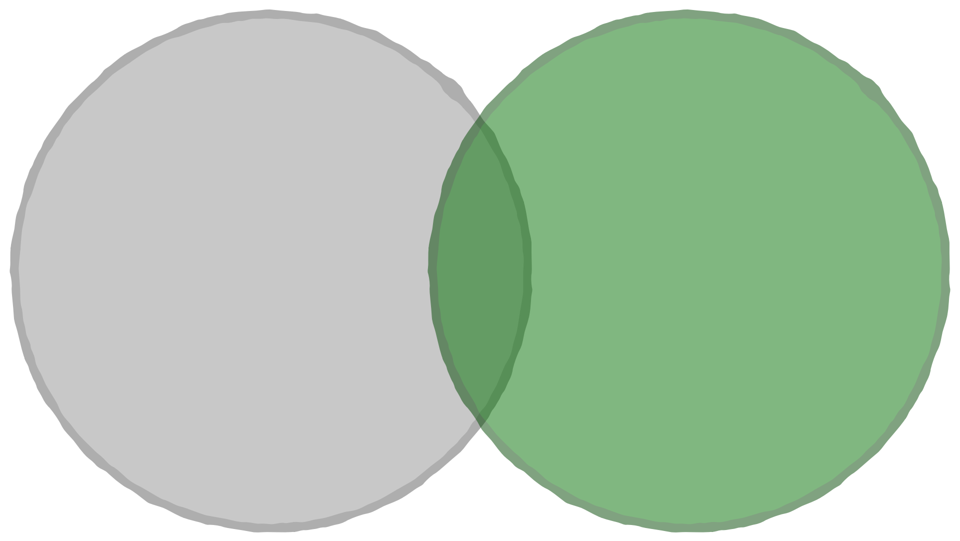
Danilo Francati
Royal Holloway University

Giuseppe Ateniese
George Mason University

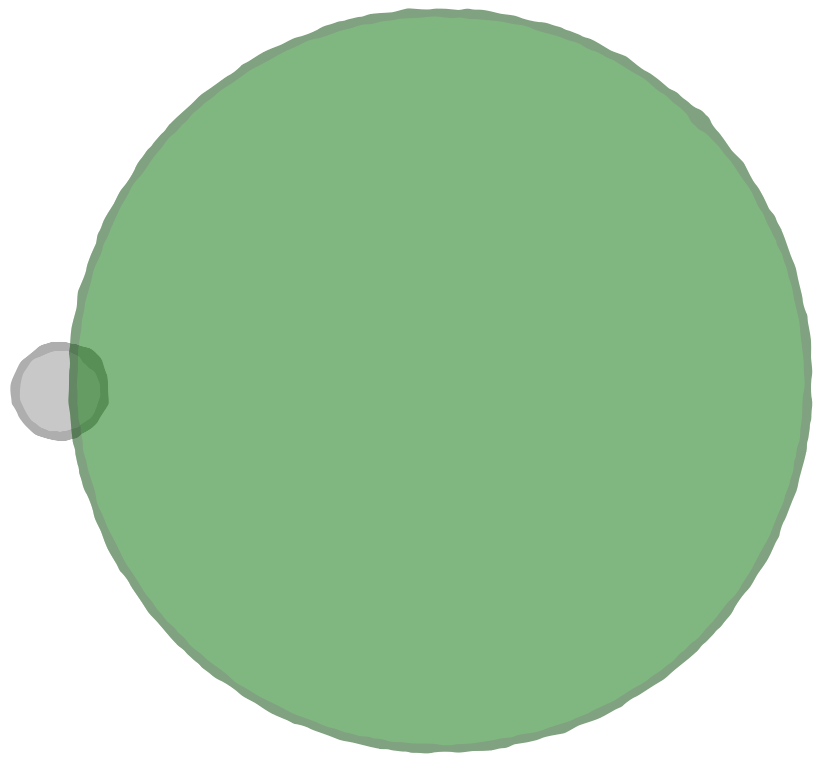
NIST Workshop on Privacy-Enhancing Cryptography (WPEC) 2024 - The PSI Day - 24th Sept 2024



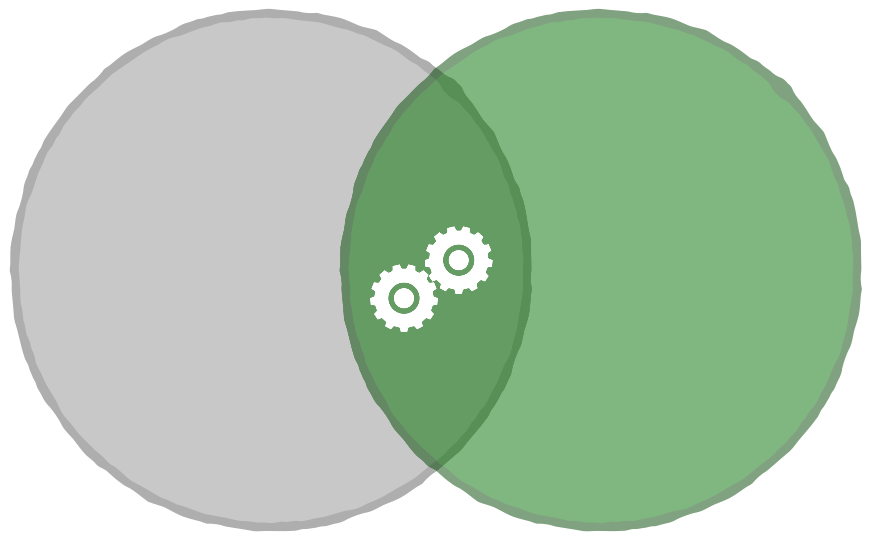
PSI on small sets (hundreds)



PSI on large sets (millions)



PSI on asymmetric sets (billions)

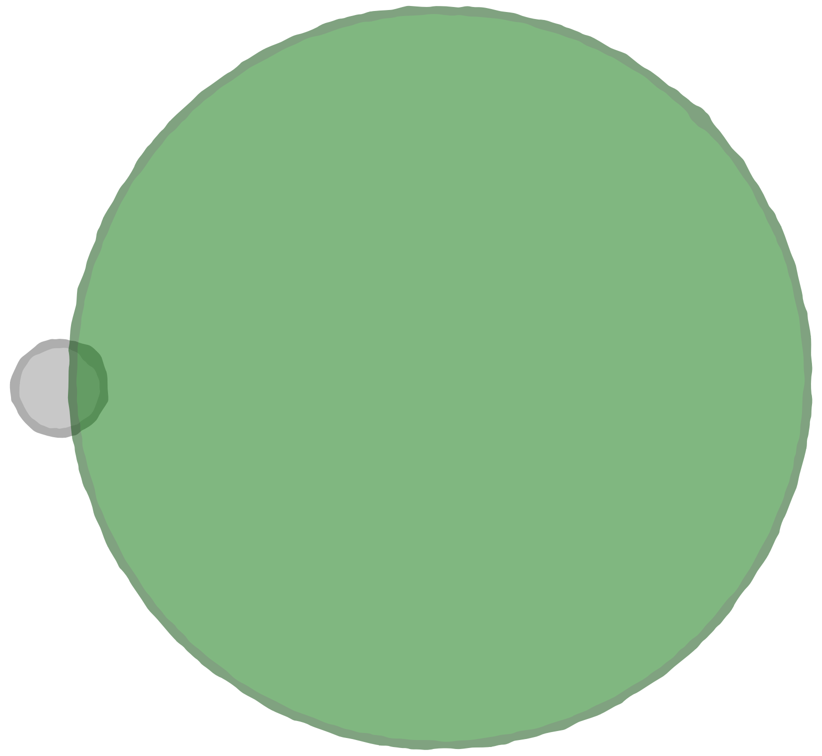


Computing on the set intersection

From: "A Brief Overview of Private Set Intersection" by Prof. Rosulek

PSI on small sets (hundreds)

PSI on large sets (millions)



PSI on asymmetric sets (billions)

- ✓ ▶ Huge challenge for practice
- ✓ ▶ Allow Leakage? Pre-processing?

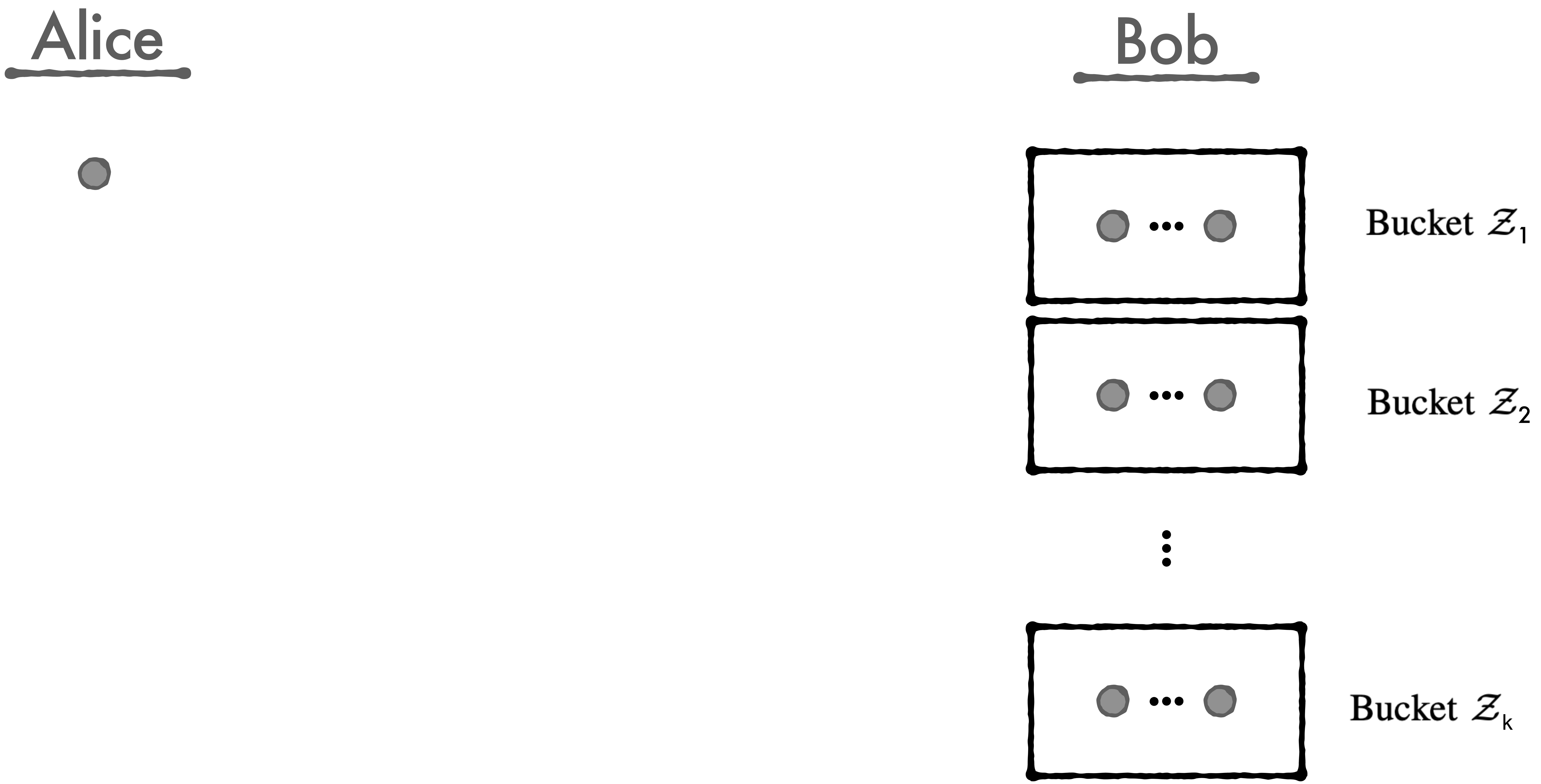
Computing on the set intersection

From: "A Brief Overview of Private Set Intersection" by Prof. Rosulek

A CLOSER LOOK AT A DEPLOYED ASYMMETRIC PSI

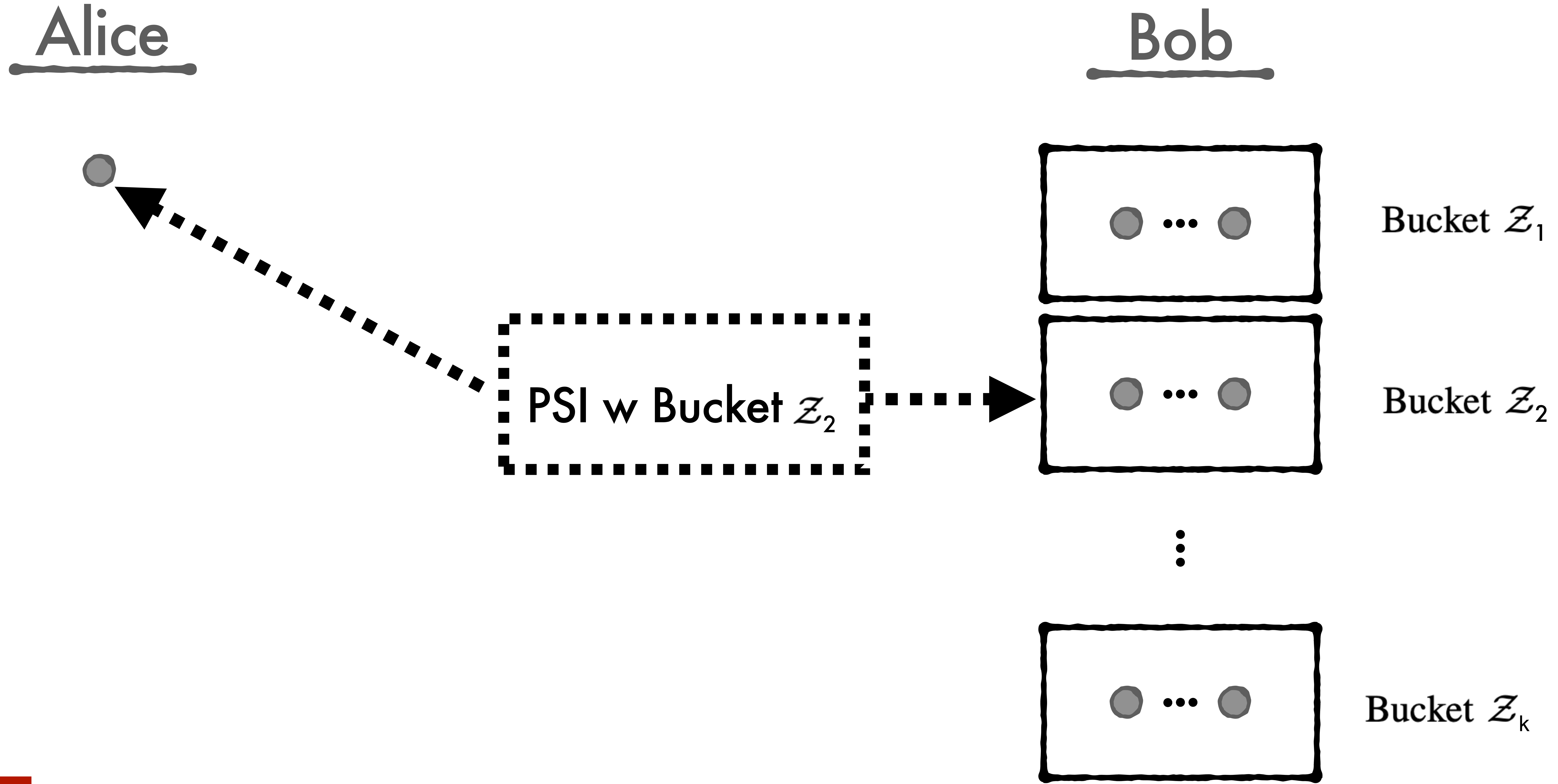
HOW UNBALANCED IS IT?

A CLOSER LOOK AT A DEPLOYED ASYMMETRIC PSI
SOLUTION: PARTITION THE LARGE SET



A CLOSER LOOK AT A DEPLOYED ASYMMETRIC PSI

SOLUTION: PARTITION THE LARGE SET



LEAKAGE

By Design: The choice of bucket reveals info about Alice's element

Need to Understand the **CONTEXT**
to Assess the Seriousness of this Leakage

CONTEXT: DATA COMPROMISES
HOW BAD IS IT?

Home » Security

Mother of all breaches reveals 26 billion records: what we know so far

Updated on: January 29, 2024 10:07 AM 3

Vilius Petkauskas, Deputy Editor

Image by Cybernews.

The supermassive leak contains data from numerous previous breaches, comprising an astounding 12 terabytes of information, spanning over a mind-boggling 26 billion records. The leak, which contains LinkedIn, Twitter, Weibo, Tencent, and other platforms' user data, is almost certainly the largest ever discovered.

There are data leaks, and then there's this. A supermassive Mother of all Breaches (MOAB for short) includes records from thousands of meticulously compiled and reindexed leaks, breaches, and privately sold databases. The full and searchable list is included at the end of this article.

Bob Dyachenko, cybersecurity researcher and owner at SecurityDiscovery.com, together with the Cybernews team, has discovered billions upon billions of exposed records on an open instance.

Even though at first the owner of the database was unknown, Leak-Lookup, a data breach search engine, said it was the holder of the leaked dataset. The platform posted a message on X, saying the problem behind the leak was a "firewall misconfiguration," which was fixed

Editor's choice

Why are people returning their Apple Vision Pro headsets?

by Neil C. Hughes 21 February 2024

Why is the highly anticipated Apple Vision Pro headset returning to stores from the hands of early adopters, influencers, and tech bros?

Read more

Cybernews podcast: how algorithms curate and flatten our online lives

23 February 2024

Book review: Kyle Chayka's "Filterworld" investigates algorithms

24 February 2024

The unequal nature of working from home

23 February 2024

LockBit's earnings in the multi-billion-dollar territory – NCA

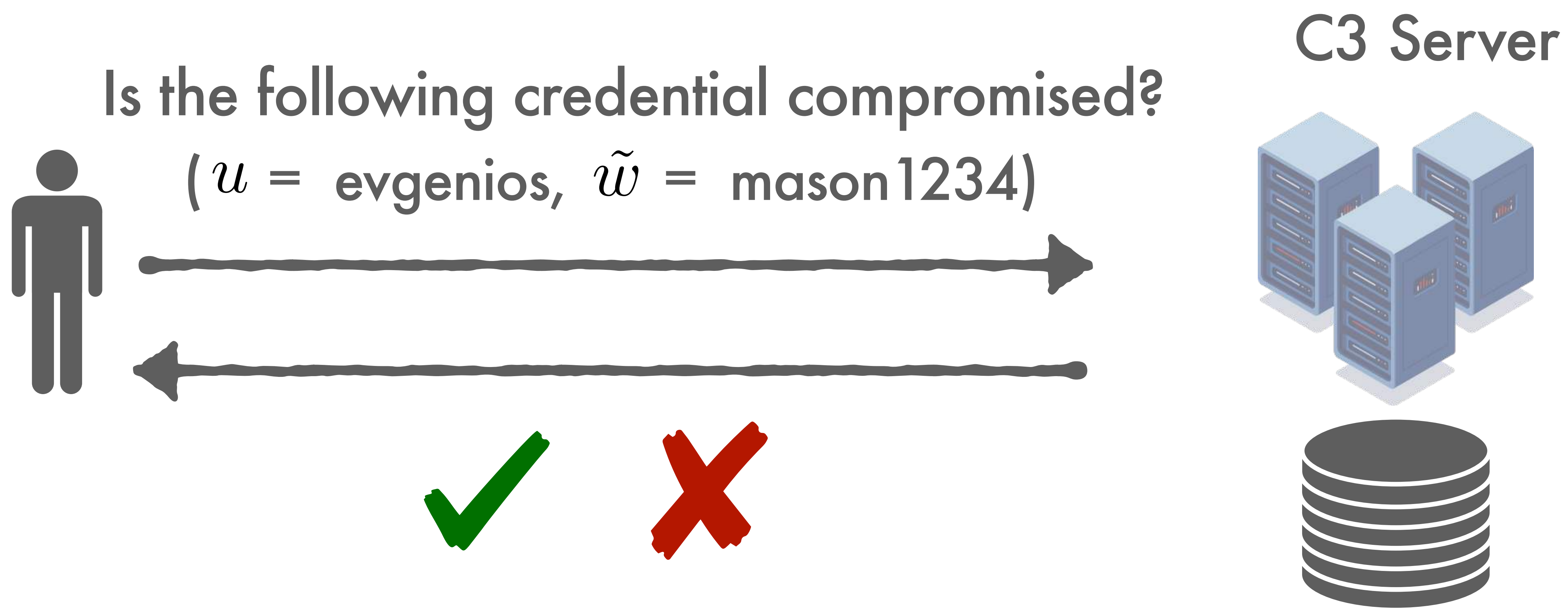
- MOAB contains *12 Terabytes* of information and *25 billion* records. (January 2024)

BRANDS WITH 100M+ LEAKED RECORDS

BRAND NAME	RECORDS LEAKED
Tencent	1.5B
Weibo	504M
MySpace	360M
Twitter	281M
Wattpad	271M
NetEase	261M
Deezer	258M
LinkedIn	251M
AdultFriendFinder	220M
Zynga	217M
Luxottica	206M
Evite	179M
Zing	164M
Adobe	153M
MyFitnessPal	151M
Canva	143M
JD.com	142M
Badoo	127M
VK	101M
Youku	100M

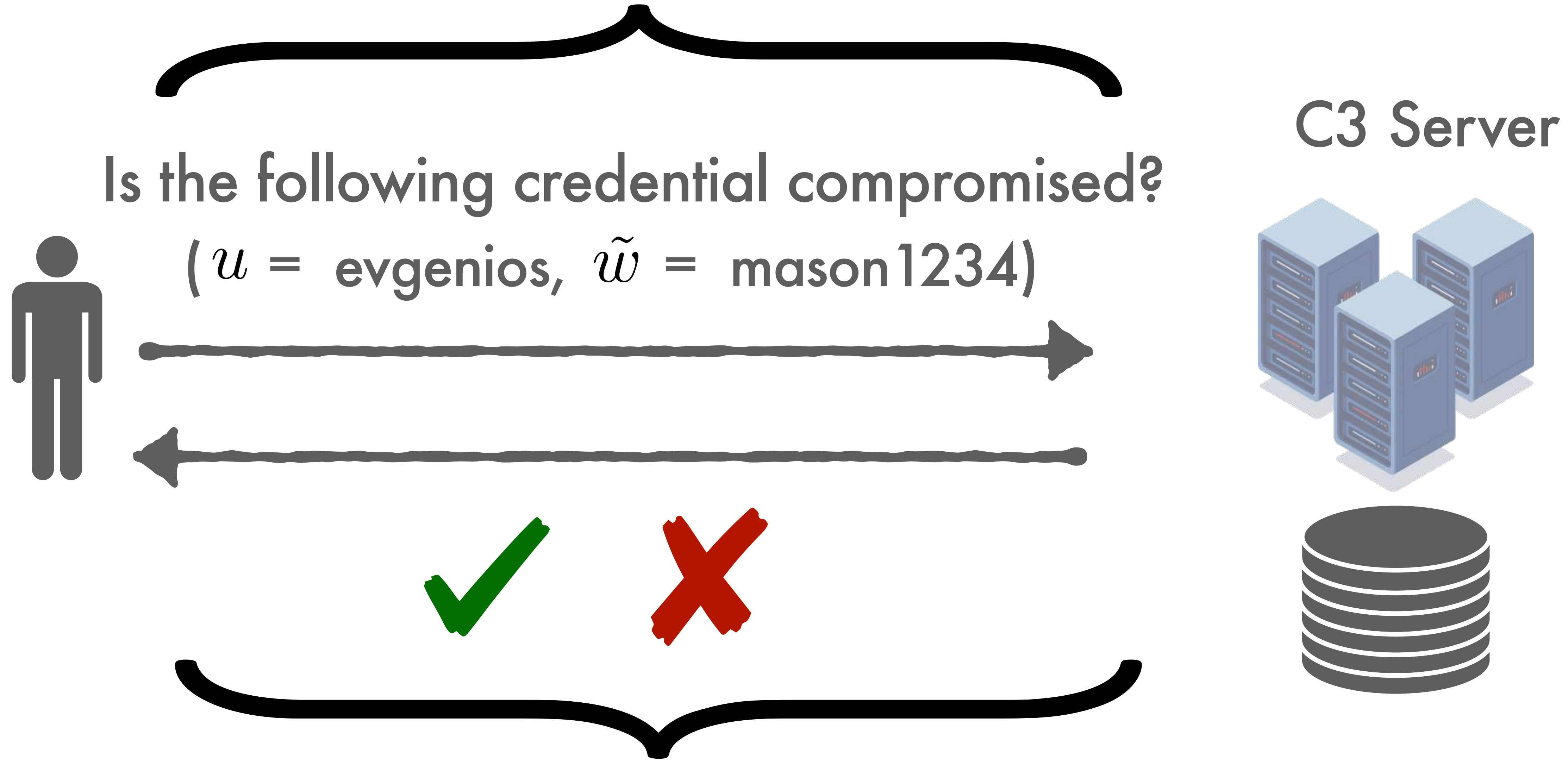
A similar dataset in 2021 had 3 billion records (an 8x increase in 2024)

POST-COMPROMISE SOLUTION
COMPROMISED CREDENTIAL CHECKING (C3) SERVICE



POST-COMPROMISE SOLUTION
COMPROMISED CREDENTIAL CHECKING (C3) SERVICE

Privacy Property:
Don't reveal information about the queried password



Privacy Property:
Don't reveal information about Compromised Data

WHY PROTECT THE ALREADY COMPROMISED DATA?

POST-COMPROMISE SOLUTION
COMPROMISED CREDENTIAL CHECKING (C3) SERVICE

HOME WORKSHOPS SPEAKING MEDIA ABOUT CONTACT SPONSOR

Seized Genesis Market Data is Now Searchable in Have I Been Pwned, Courtesy of the FBI and "Operation Cookie Monster"

05 APRIL 2023

A quick summary first before the details: This week, the FBI in cooperation with international law enforcement partners took down a notorious marketplace trading in stolen identity data in an effort they've named "Operation Cookie Monster". They've provided millions of impacted email addresses and passwords to Have I Been Pwned (HIBP) so that victims of the incident can discover if they have been exposed. This breach has been flagged as "sensitive" which means *it is not publicly searchable*, rather you must demonstrate you control the email address being searched before the results are shown. This can be done via the free notification service on HIBP and involves you entering the email address then clicking on the link sent to your inbox. Specific guidance prepared by the FBI in conjunction with the Dutch police on further steps you can take to protect yourself are detailed at the end of this blog post on the gold background. That's the short version, here's the whole story:

Troy Hunt

Hi, I'm Troy Hunt, I write this blog, run "Have I Been Pwned" and am a Microsoft Regional Director and MVP who travels the world speaking at events and training technology professionals →

Upcoming Events

I often run private workshops around these, here's upcoming events I'll be at:

NDC Oslo: 10 to 14 Jun, Oslo (Norway)

Must Read

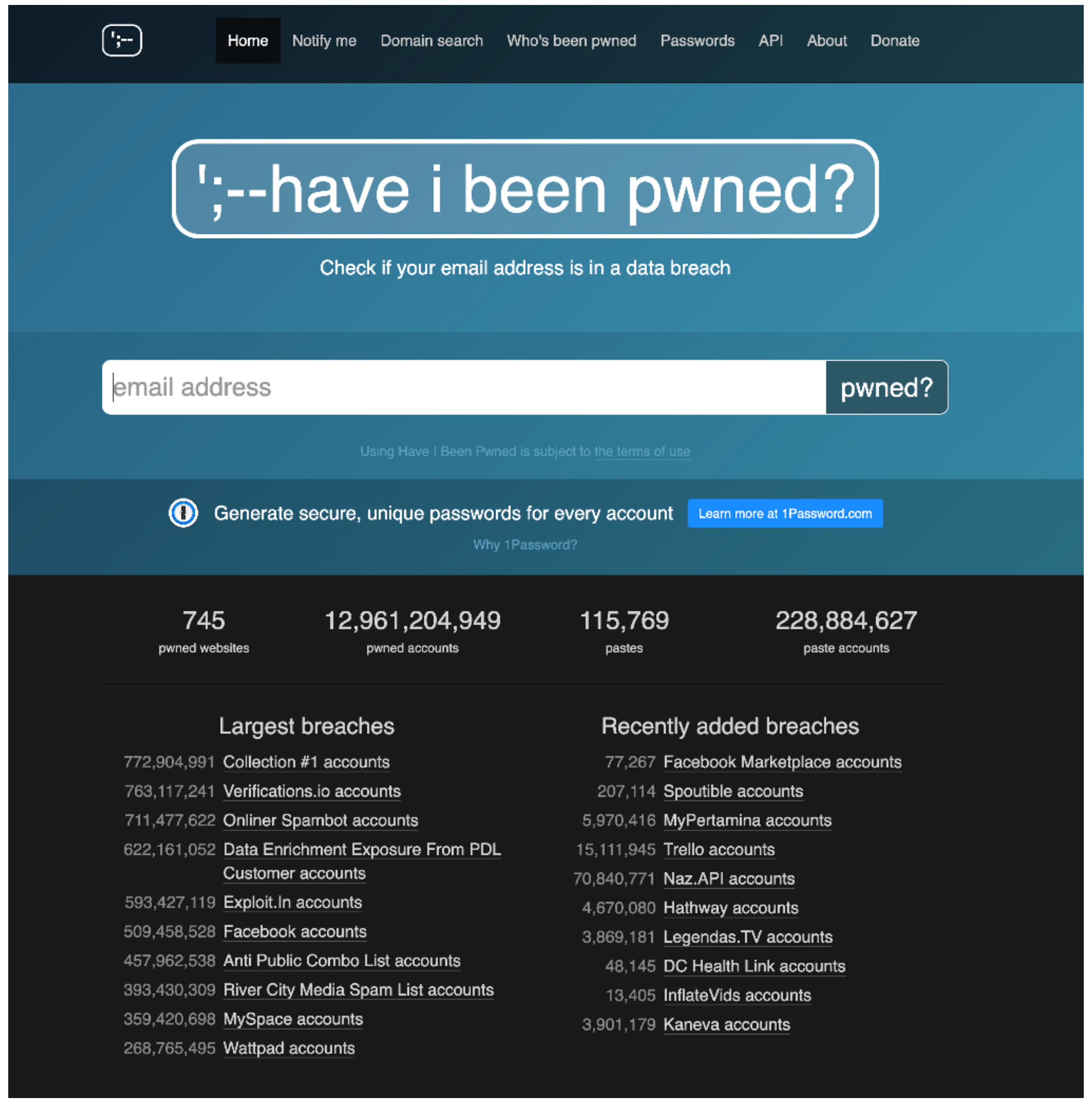
Data breach disclosure 101: How to succeed after you've failed

Data from connected CloudPets teddy bears leaked and ransomed, exposing kids' voice messages

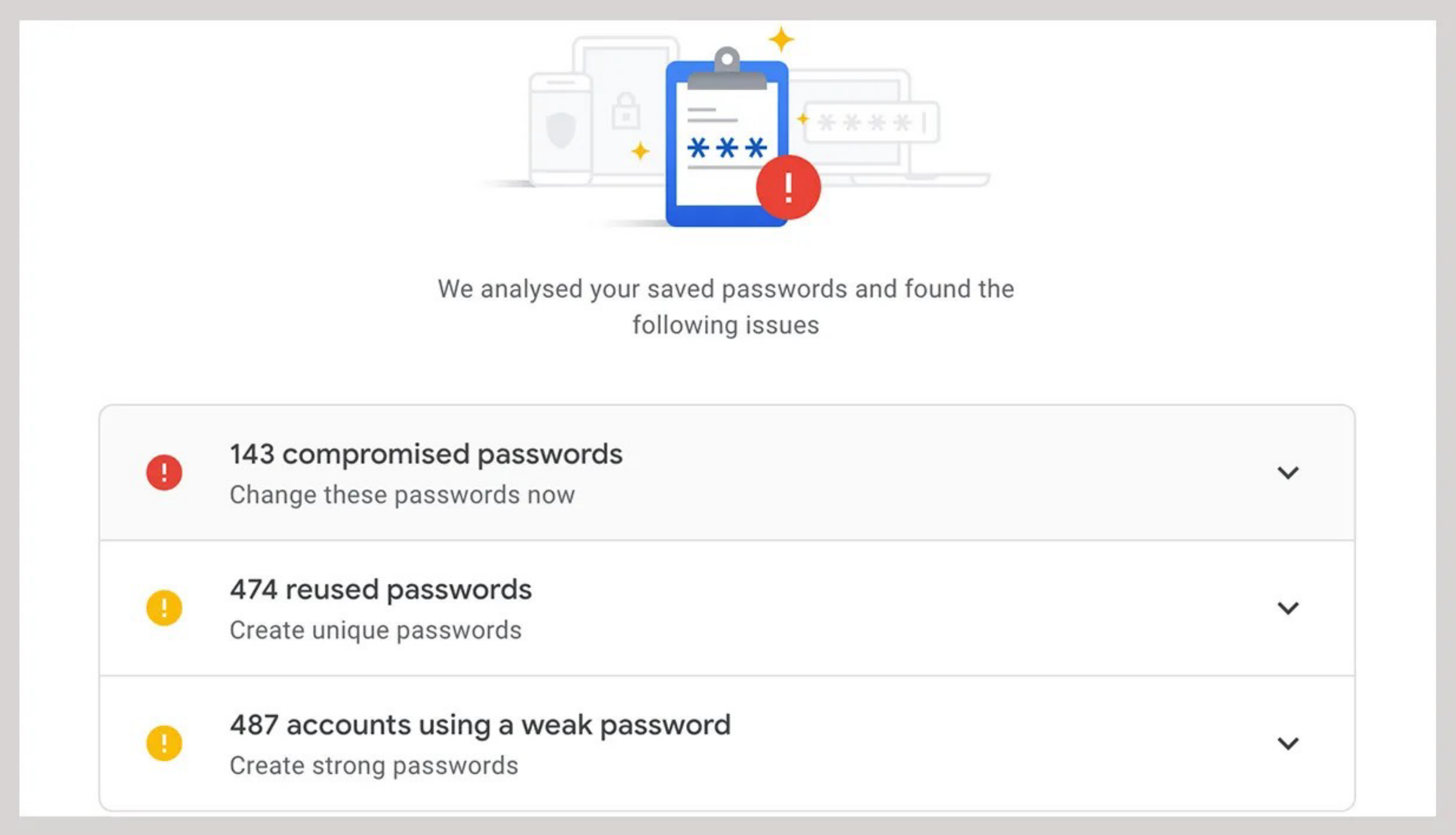
Here's how I verify data breaches

- In April 2023, FBI shared with a C3 service millions of compromised but not publicly available credentials
- In practice, the compromised credential set will contain a combination of:
  (1) publicly available and
  (2) non-accessible compromised credentials.

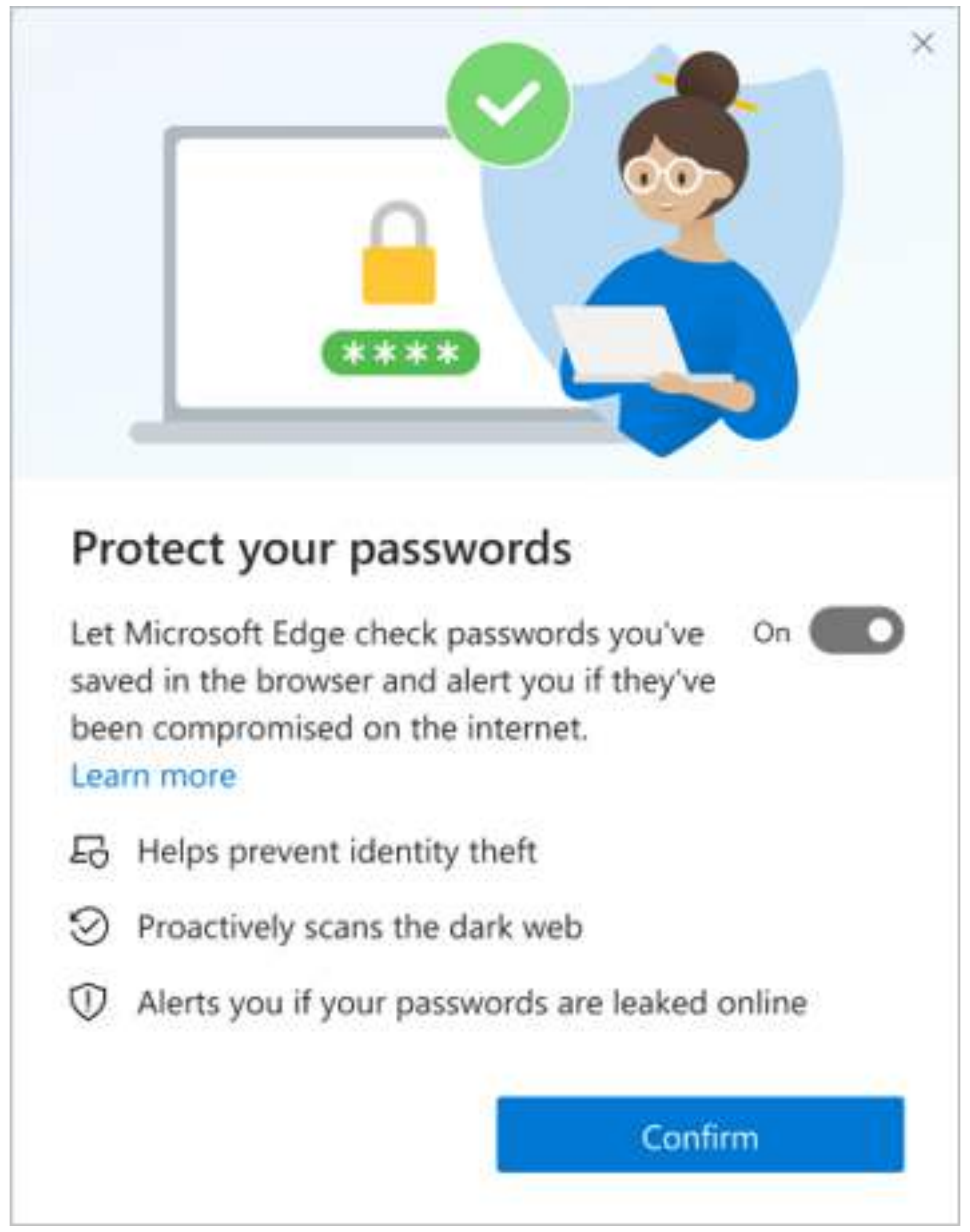
DEPLOYED TECHNOLOGY
COMPROMISED CREDENTIAL CHECKING (C3) SERVICE



https://haveibeenpwned.com/

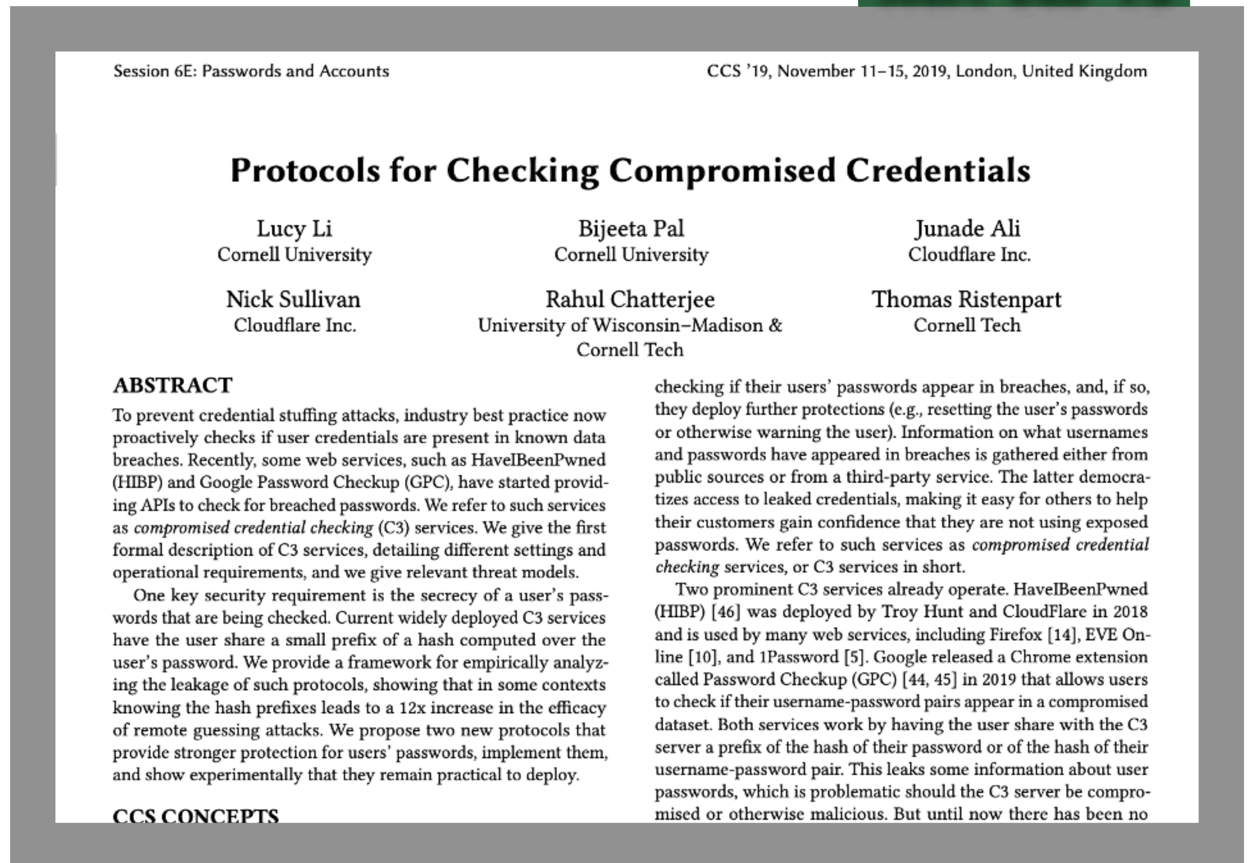


Password Checkup (Google)

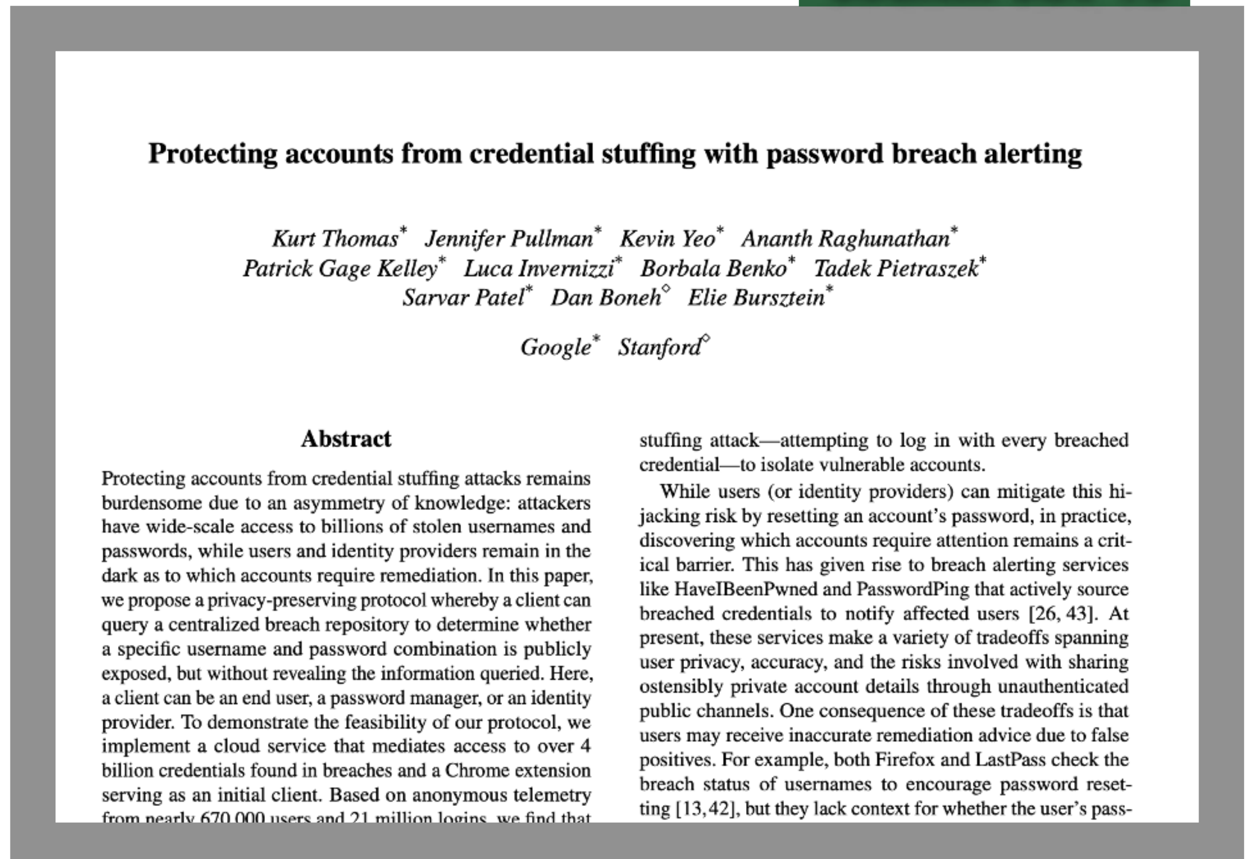


Password Monitor (Microsoft)

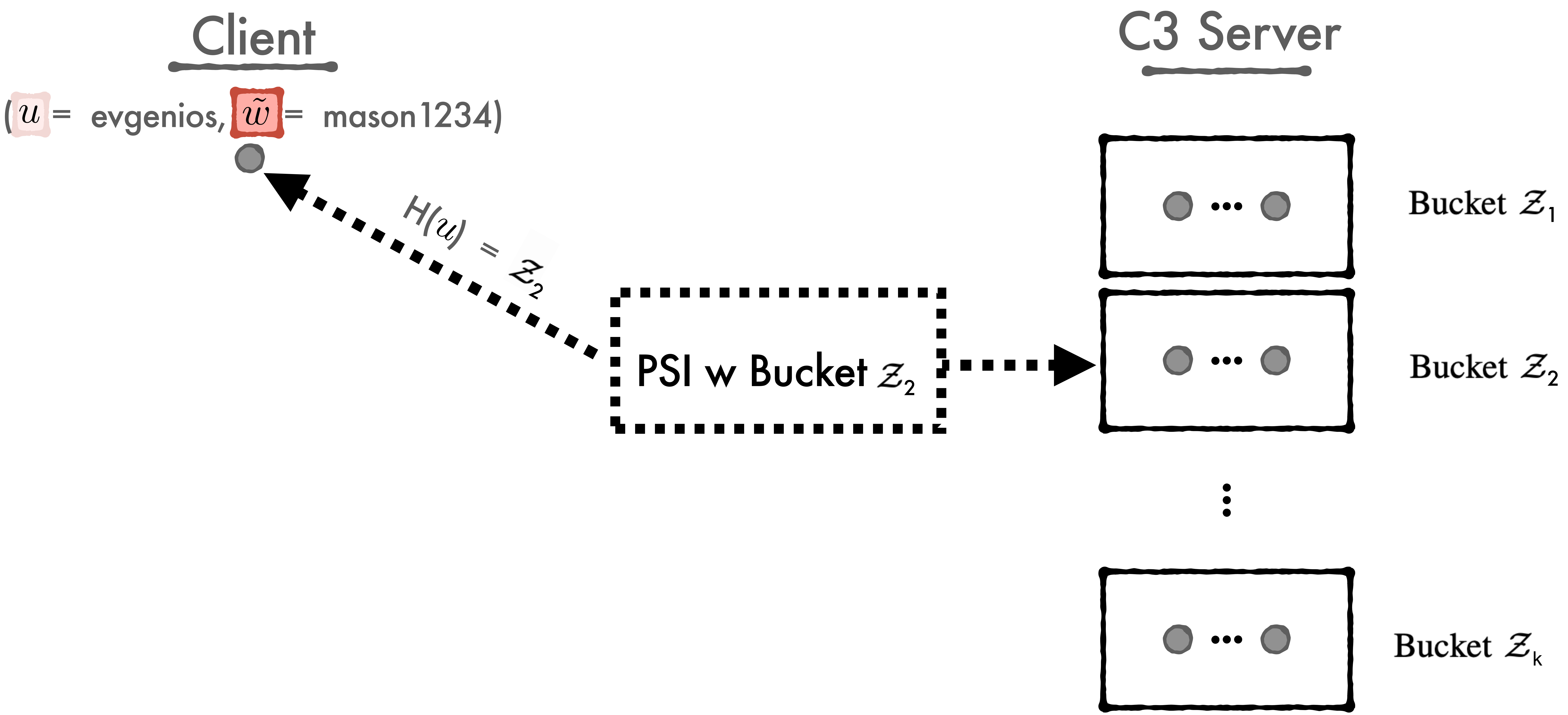
ACM CCS'19



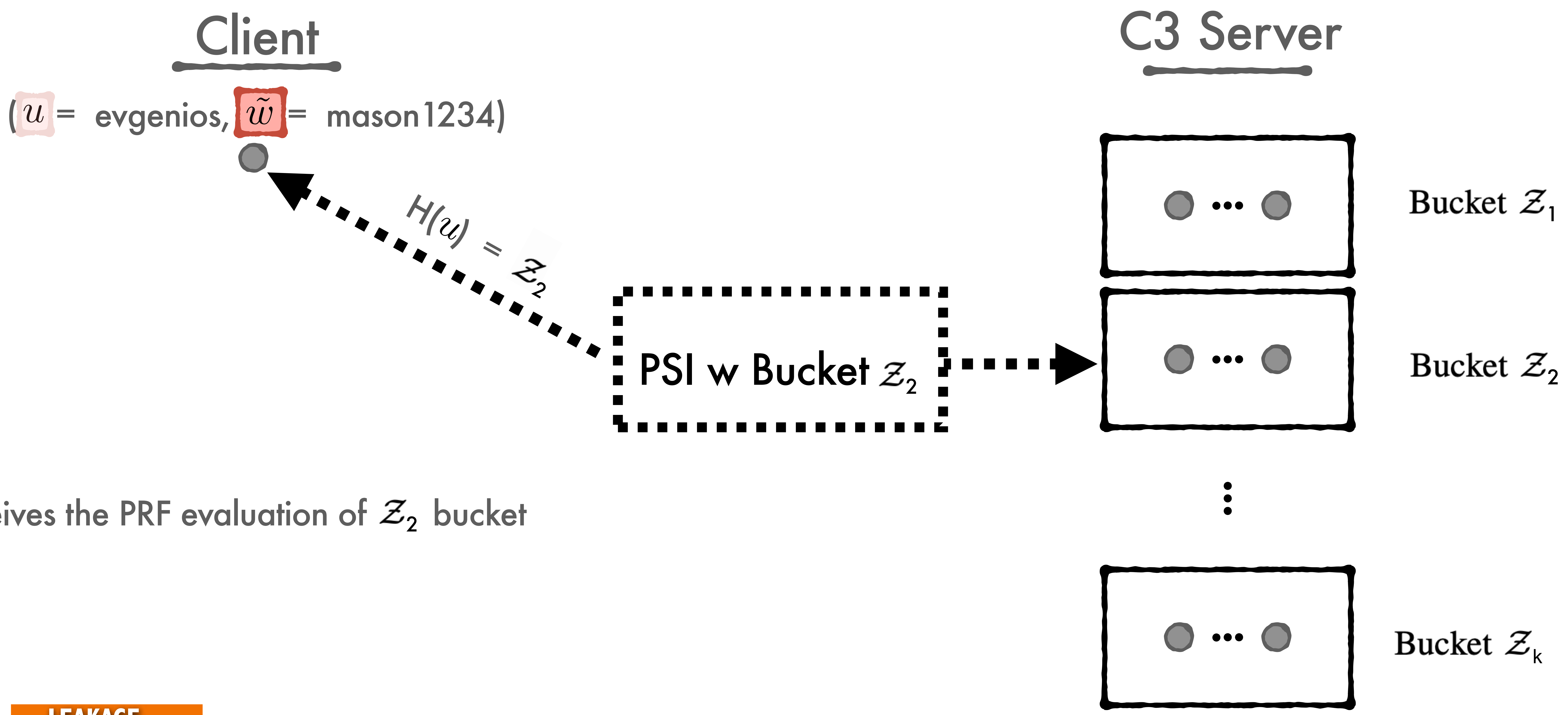
USENIX SEC'19



APPLICATION CONTEXT
IS THIS LEAKAGE HARMFUL?



APPLICATION CONTEXT
IS THIS LEAKAGE HARMFUL?



*Client receives the PRF evaluation of $\mathcal{Z}_2$ bucket

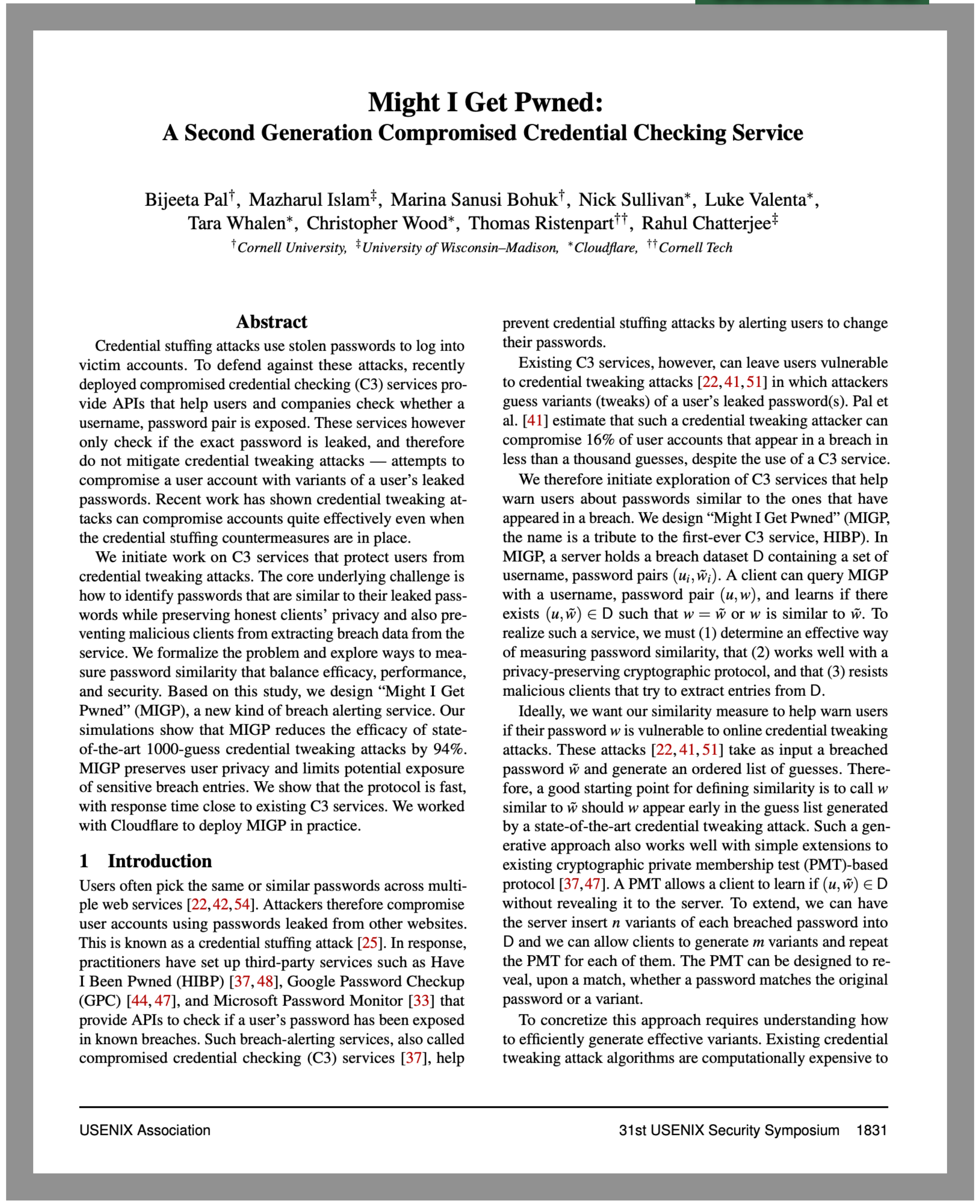
LEAKAGE

By Design: The choice of bucket reveals info about Client's username

MIGHT I GET PWNED

THE ASYMMETRIC PSI PROTOCOL UNDER ATTACK

USENIX SEC'22



Might I Get Pwned:
A Second Generation Compromised Credential Checking Service

Bijeeta Pal†, Mazharul Islam‡, Marina Sanusi Bohuk†, Nick Sullivan*, Luke Valenta*, Tara Whalen*, Christopher Wood*, Thomas Ristenpart††, Rahul Chatterjee‡

†*Cornell University,* ‡*University of Wisconsin–Madison,* **Cloudflare,* ††*Cornell Tech*

Abstract

Credential stuffing attacks use stolen passwords to log into victim accounts. To defend against these attacks, recently deployed compromised credential checking (C3) services provide APIs that help users and companies check whether a username, password pair is exposed. These services however only check if the exact password is leaked, and therefore do not mitigate credential tweaking attacks — attempts to compromise a user account with variants of a user's leaked passwords. Recent work has shown credential tweaking attacks can compromise accounts quite effectively even when the credential stuffing countermeasures are in place.

We initiate work on C3 services that protect users from credential tweaking attacks. The core underlying challenge is how to identify passwords that are similar to their leaked passwords while preserving honest clients' privacy and also preventing malicious clients from extracting breach data from the service. We formalize the problem and explore ways to measure password similarity that balance efficacy, performance, and security. Based on this study, we design "Might I Get Pwned" (MIGP), a new kind of breach alerting service. Our simulations show that MIGP reduces the efficacy of state-of-the-art 1000-guess credential tweaking attacks by 94%. MIGP preserves user privacy and limits potential exposure of sensitive breach entries. We show that the protocol is fast, with response time close to existing C3 services. We worked with Cloudflare to deploy MIGP in practice.

1 Introduction

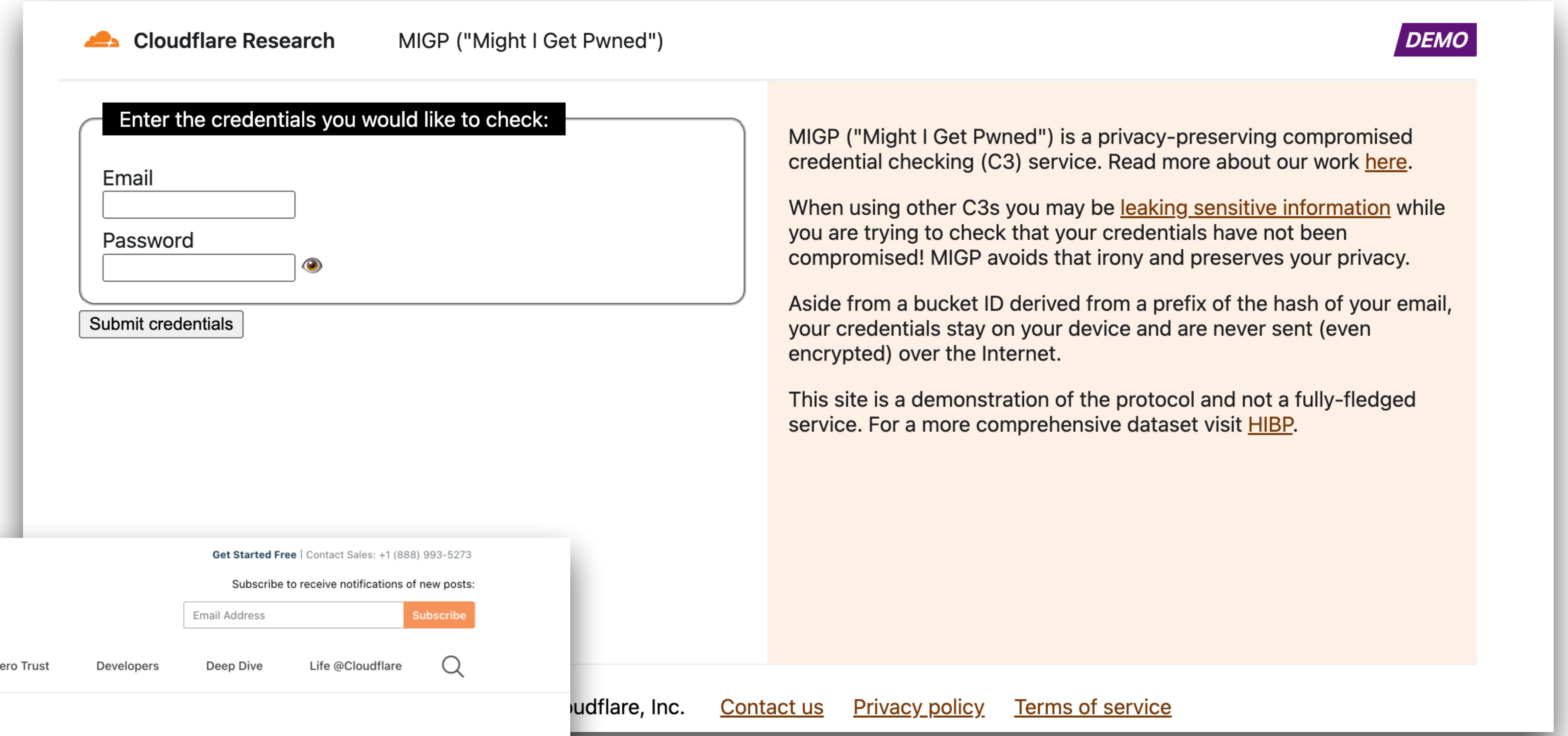
Users often pick the same or similar passwords across multiple web services [22, 42, 54]. Attackers therefore compromise user accounts using passwords leaked from other websites. This is known as a credential stuffing attack [25]. In response, practitioners have set up third-party services such as Have I Been Pwned (HIBP) [37, 48], Google Password Checkup (GPC) [44, 47], and Microsoft Password Monitor [33] that provide APIs to check if a user's password has been exposed in known breaches. Such breach-alerting services, also called compromised credential checking (C3) services [37], help prevent credential stuffing attacks by alerting users to change their passwords.

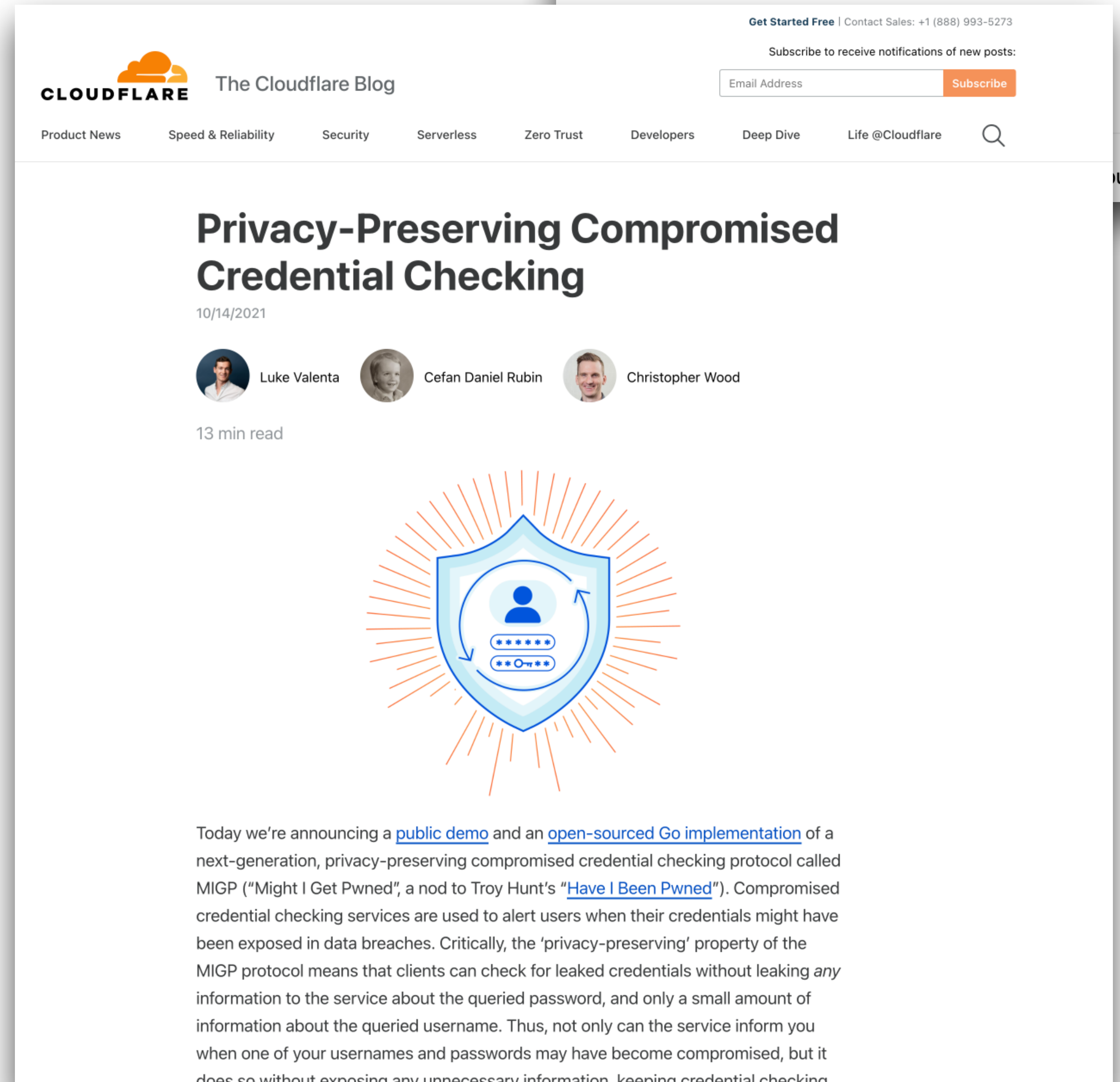
Existing C3 services, however, can leave users vulnerable to credential tweaking attacks [22, 41, 51] in which attackers guess variants (tweaks) of a user's leaked password(s). Pal et al. [41] estimate that such a credential tweaking attacker can compromise 16% of user accounts that appear in a breach in less than a thousand guesses, despite the use of a C3 service.

We therefore initiate exploration of C3 services that help warn users about passwords similar to the ones that have appeared in a breach. We design "Might I Get Pwned" (MIGP, the name is a tribute to the first-ever C3 service, HIBP). In MIGP, a server holds a breach dataset D containing a set of username, password pairs $(u_i, \tilde{w}_i)$. A client can query MIGP with a username, password pair (u, w), and learns if there exists $(u, \tilde{w}) \in \mathsf{D}$ such that $w = \tilde{w}$ or w is similar to $\tilde{w}$. To realize such a service, we must (1) determine an effective way of measuring password similarity, that (2) works well with a privacy-preserving cryptographic protocol, and that (3) resists malicious clients that try to extract entries from D.

Ideally, we want our similarity measure to help warn users if their password w is vulnerable to online credential tweaking attacks. These attacks [22, 41, 51] take as input a breached password $\tilde{w}$ and generate an ordered list of guesses. Therefore, a good starting point for defining similarity is to call w similar to $\tilde{w}$ should w appear early in the guess list generated by a state-of-the-art credential tweaking attack. Such a generative approach also works well with simple extensions to existing cryptographic private membership test (PMT)-based protocol [37, 47]. A PMT allows a client to learn if $(u, \tilde{w}) \in \mathsf{D}$ without revealing it to the server. To extend, we can have the server insert n variants of each breached password into D and we can allow clients to generate m variants and repeat the PMT for each of them. The PMT can be designed to reveal, upon a match, whether a password matches the original password or a variant.

To concretize this approach requires understanding how to efficiently generate effective variants. Existing credential tweaking attack algorithms are computationally expensive to





https://migp.cloudflare.com/

Breach Extraction Attacks: Exposing and Addressing the Leakage in Second Generation Compromised Credential Checking Services

Dario Pasquini*
SPRING lab, EPFL
dario.pasquini@epfl.ch

Danilo Francati*
Aarhus University
dfrancati@cs.au.dk

Giuseppe Ateniese
George Mason University
ateniese@gmu.edu

Evgenios M. Kornaropoulos
George Mason University
evgenios@gmu.edu

***Abstract*—Credential tweaking attacks use breached passwords to generate semantically similar passwords and gain access to victims' services. These attacks sidestep the first generation of compromised credential checking (C3) services. The second generation of compromised credential checking services, called "Might I Get Pwned" (MIGP), is a privacy-preserving protocol that defends against credential tweaking attacks by allowing clients to query whether a password or a semantically similar variation is present in the server's compromised credentials dataset. The desired privacy requirements include not revealing the user's entered password to the server and ensuring that no compromised credentials are disclosed to the client.**

In this work, we formalize the cryptographic leakage of the MIGP protocol and perform a security analysis to assess its impact on the credentials held by the server. We focus on how this leakage aids breach extraction attacks, where an honest-but-curious client interacts with the server to extract information about the stored credentials. Furthermore, we discover additional leakage that arises from the implementation of Cloudflare's deployment of MIGP. We evaluate how the discovered leakage affects the guessing capability of an attacker in relation to breach extraction attacks. Finally, we propose MIGP 2.0, a new iteration of the MIGP protocol designed to minimize data leakage and prevent the introduced attacks.

1. Introduction

In the evolving cyber threat landscape, attackers target user credentials, particularly those stored in plaintext, exploiting system vulnerabilities to compromise and post them online, thereby breaching user privacy and enabling *credential stuffing attacks* [1]. In these attacks, adversaries exploit widespread password reuse [2], [3], [4], [5] by using credentials exposed from a data breach to attempt unauthorized access to another unrelated domain. Services like "Have I Been Pwned" [6], Google Password Checkup [7], and Microsoft Password Monitor [8]—known as Compromised Credential Checking (C3) services— aim to alert users about the possibility of a credential stuffing attack. Specifically, they allow users to check if their active credentials appear in breach datasets. To accomplish this, C3 services use cryptographic tools to create a privacy-preserving protocol, ensuring that the queried password of the user (which may not be breached) is not disclosed to the server and the sensitive breached credentials are not shared with the client.

However, these services cannot cover an increasingly common type of attack: *credential tweaking attacks* [2], [9], [10]. In these attacks, cybercriminals employ sophisticated techniques to generate slight variants of known breached passwords, enabling them to make distinct educated password guesses towards unauthorized access to the target's services. Unfortunately, credential tweaking attacks are not covered by C3 services since they only check for *an exact match* against the breached credentials. To address these shortcomings, Pal et al. [11] proposed *Might I Get Pwned* (MIGP), a second-generation C3 service. MIGP extends the capabilities of conventional C3 by checking not only for exact password matches but also for semantic similarity with breached credentials. To achieve this, MIGP uses a password-generating function called τ to generate semantically similar passwords during the initialization phase.

The Role of Cryptographic Leakage in MIGP. It is important to note that the privacy-preserving design of MIGP serves (in part) the purpose of safeguarding the collection of breached credential data from being exposed to MIGP query issuers. Paradoxically, despite the MIGP server's data collection being labeled as "breached credentials," it can contain credentials that have been *breached but are not publicly available*. In April 2023 [12], the FBI took down a stolen identity marketplace that was selling non-publicly available breached credentials. To combat credential stuffing attacks, the FBI shared in confidence millions of non-publicly available compromised credentials with HIBP. Thus, real-world C3 services work with breached credentials that should not be exposed under any circumstances.[1]

Our work sheds light on an unexplored aspect of the MIGP protocol: the existence of *cryptographic leakage* over the stored credentials. This leakage is a controlled disclosure intentionally designed into the protocol. The term *breach extraction attack* describes the attack vector in which

* *Equal contribution.*

1. We note that if the breached credentials of the server are all considered public, then there is no point in deploying a privacy-preserving C3; the server can simply return a subset (i.e., a bucket) of the credentials in plaintext. This change enhances efficiency by forgoing cryptographic operations for non-interactive queries. Additionally, it fortifies defenses against tweaking attacks, allowing users to apply arbitrary similarity functions to leaked passwords.

https://eprint.iacr.org/2023/1848

BREACH EXTRACTION ATTACKS: Exposing and Addressing the Leakage in Second-Generation Compromised Credential Checking Services

PASQUINI, FRANCATI, ATENIESE, **E.M.K.**

Proc. IEEE SECURITY & PRIVACY (Oakland), 2024

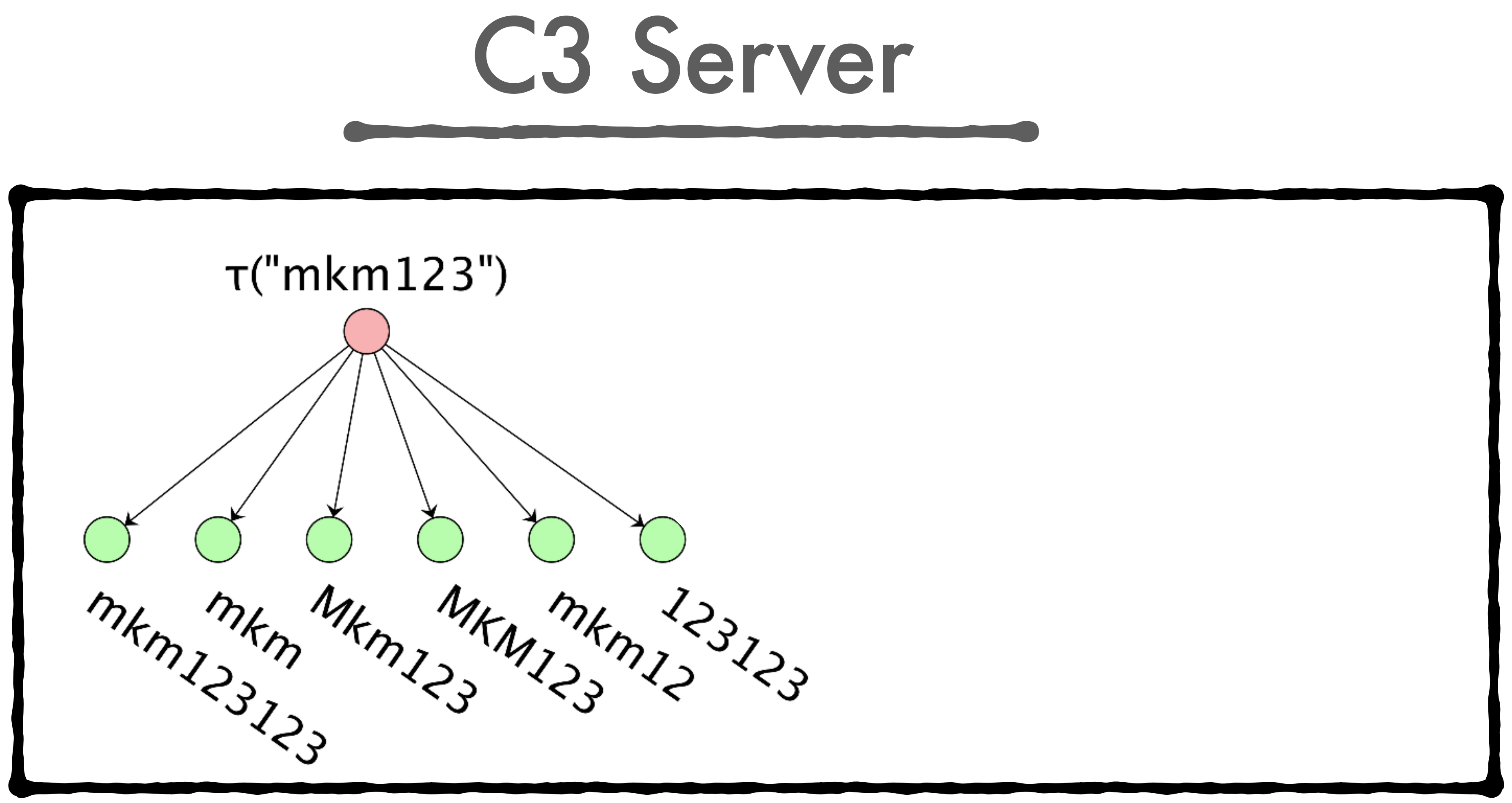
Our Contributions:

1. Formalizing the Leakage in MIGP
2. Taxonomy of τ-collisions
3. Breach Extraction Attacks via Leakage
4. MIGP 2.0

Pwnie Award Finalist
"Best Cryptographic Attack"
2024

UNINTENDED LEAKAGE OF MIGP
ZOOMING IN THE BUCKETS OF ASYMMETRIC PSI

- MIGP needs to check whether a client **MIGHT** get exposed
- Given a password "mkm123", MIGP generates **PROBABLE PASSWORDS** the client might be using now or in the future



Bucket $\mathcal{Z}_2$

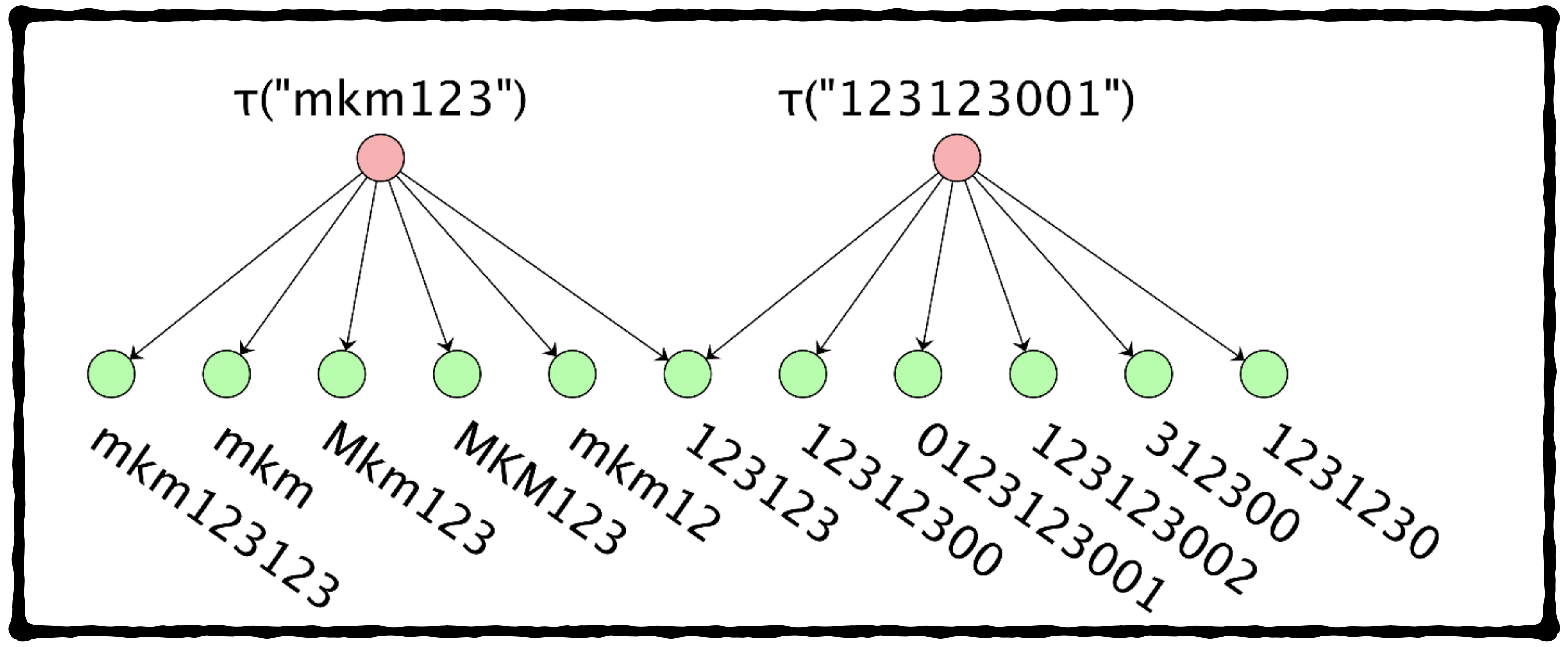
UNINTENDED LEAKAGE OF MIGP

ZOOMING IN THE BUCKETS OF ASYMMETRIC PSI

- MIGP needs to check whether a client **MIGHT** get exposed
- Given a password "mkm123", MIGP generates **PROBABLE PASSWORDS** the client might be using now or in the future

If both passwords "mkm123" and "123123001" are breached, then duplicate-insertion creates a **hole**.

C3 Server

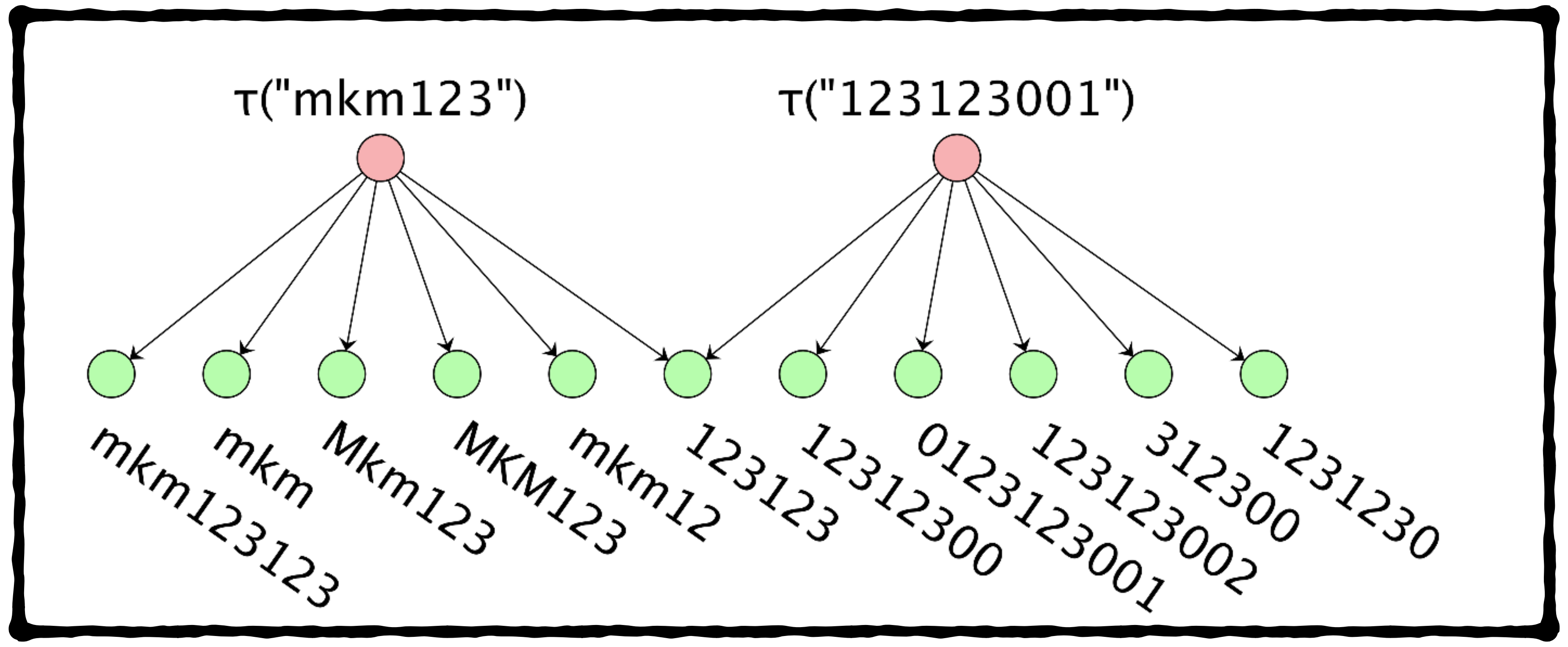


Bucket $\mathcal{Z}_2$

UNINTENDED LEAKAGE OF MIGP
ZOOMING IN THE BUCKETS OF ASYMMETRIC PSI

- MIGP needs to check whether a client **MIGHT** get exposed
- Given a password "mkm123", MIGP generates **PROBABLE PASSWORDS** the client might be using now or in the future

C3 Server



Bucket $\mathcal{Z}_2$

If both passwords "mkm123" and "123123001" are breached, then duplicate-insertion creates a **hole**.

PROTOCOL

semantically-similar passwords → τ-collisions

ATTACK

semantically-similar passwords ←?— τ-collisions

INTRINSIC TENSION

STRONG PROTECTION AGAINST TWEAKING ATTACKS

Function τ generates passwords contained in future breaches

$\Longrightarrow$

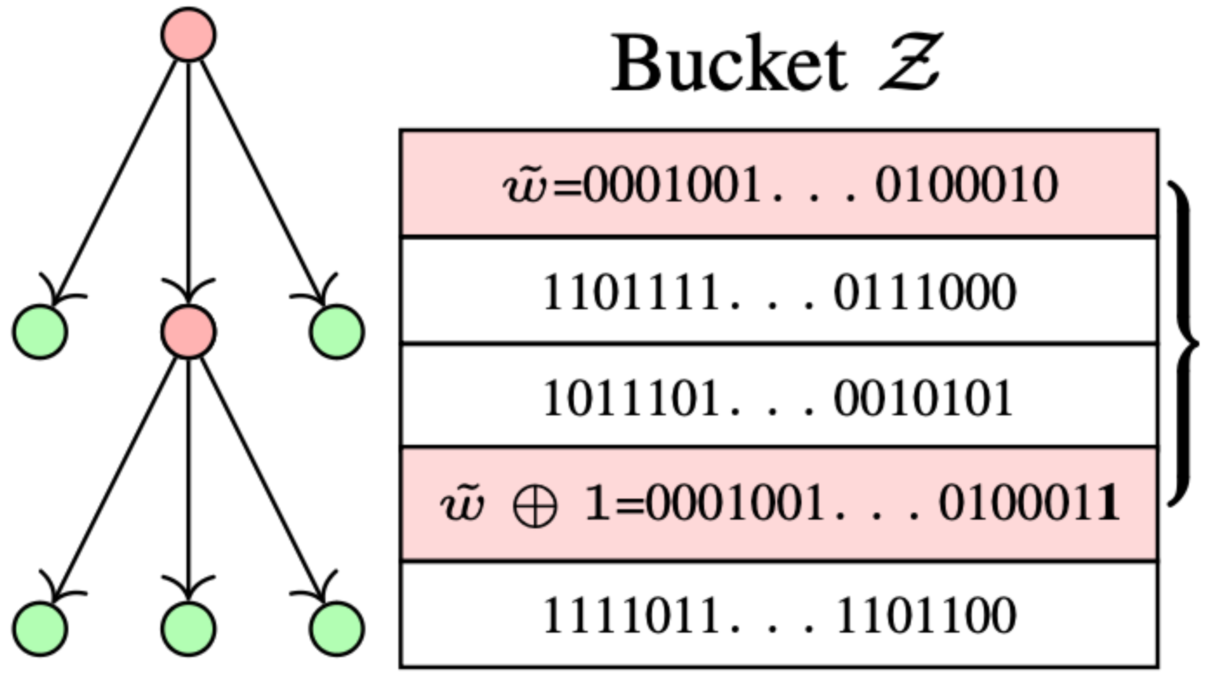
INCREASED LEAKAGE

Function τ maximizes the number of collisions

A TAXONOMY OF τ-COLLISIONS

DEFINITION AND DETECTION MECHANISMS

TYPE - 0



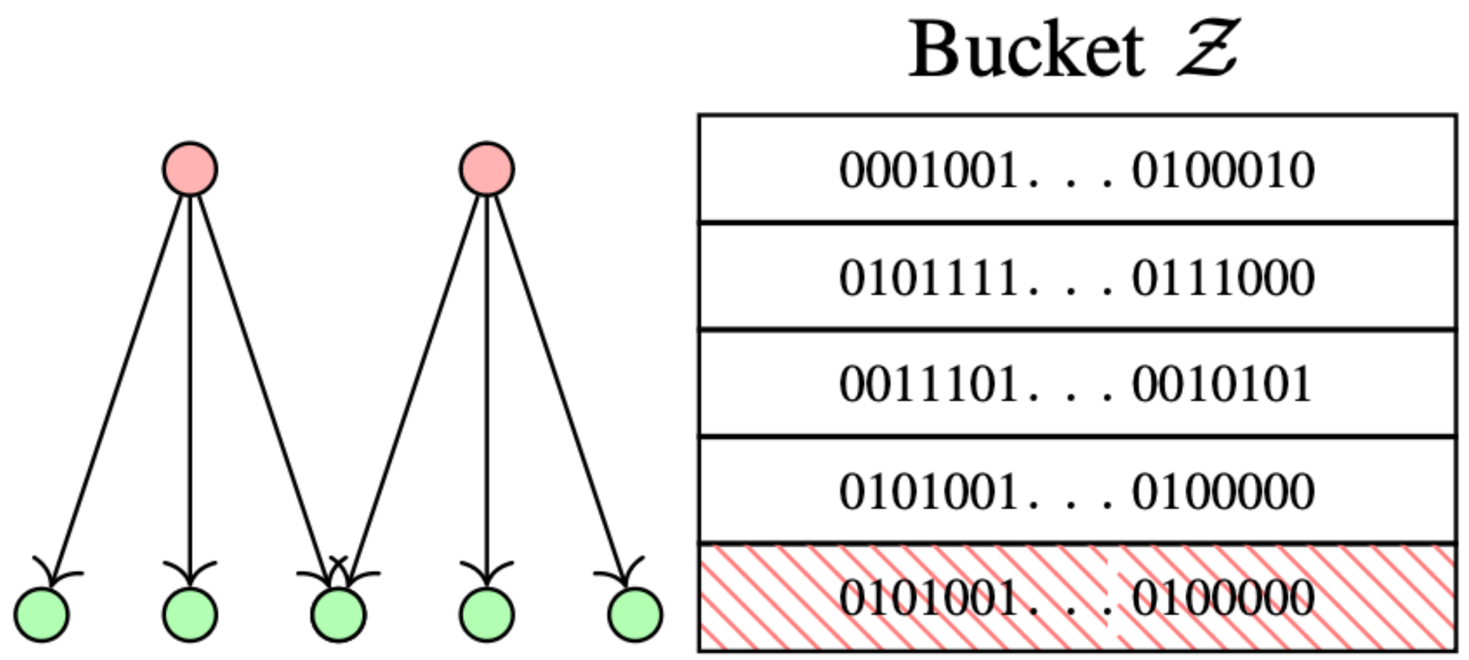
TYPE - 1A

TYPE - 1B

A TAXONOMY OF τ-COLLISIONS
DEFINITION AND DETECTION MECHANISMS

TYPE - 0

TYPE - 1A



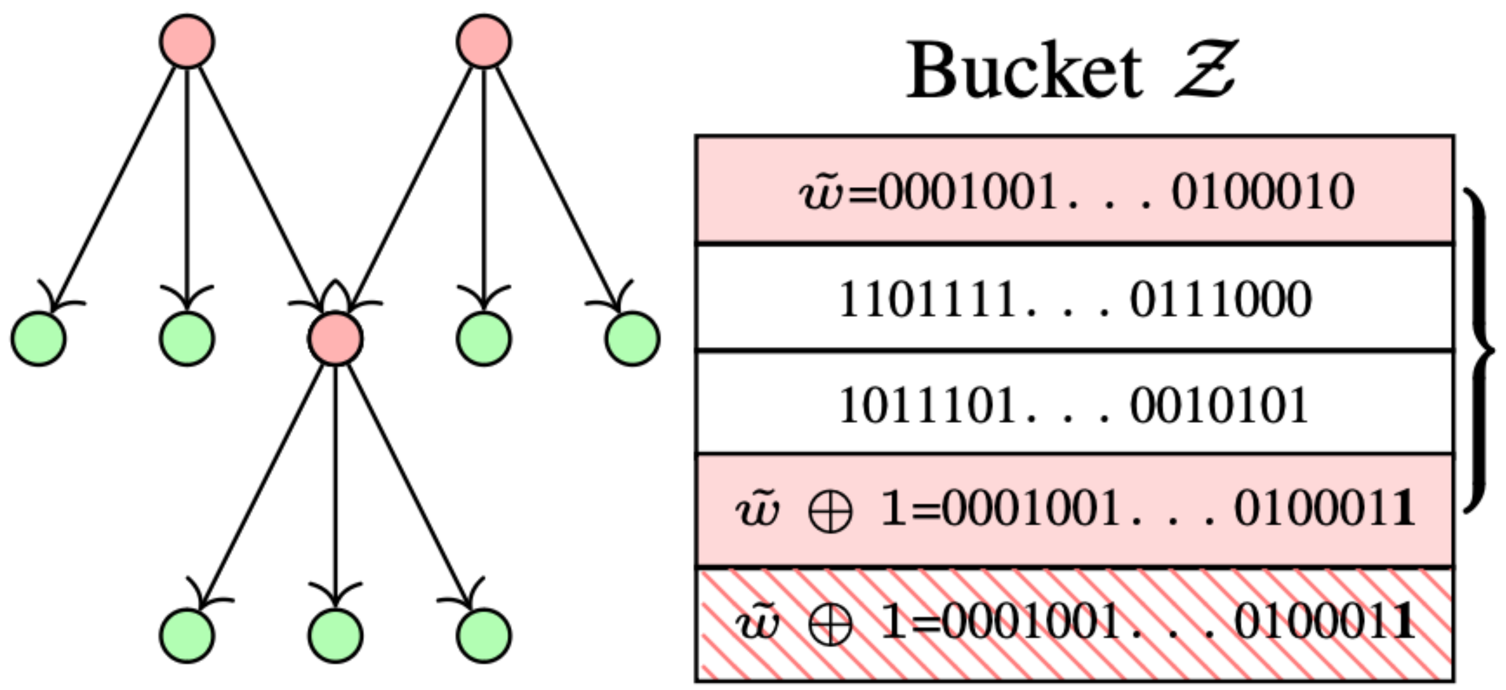
TYPE - 1B

A TAXONOMY OF τ-COLLISIONS
DEFINITION AND DETECTION MECHANISMS

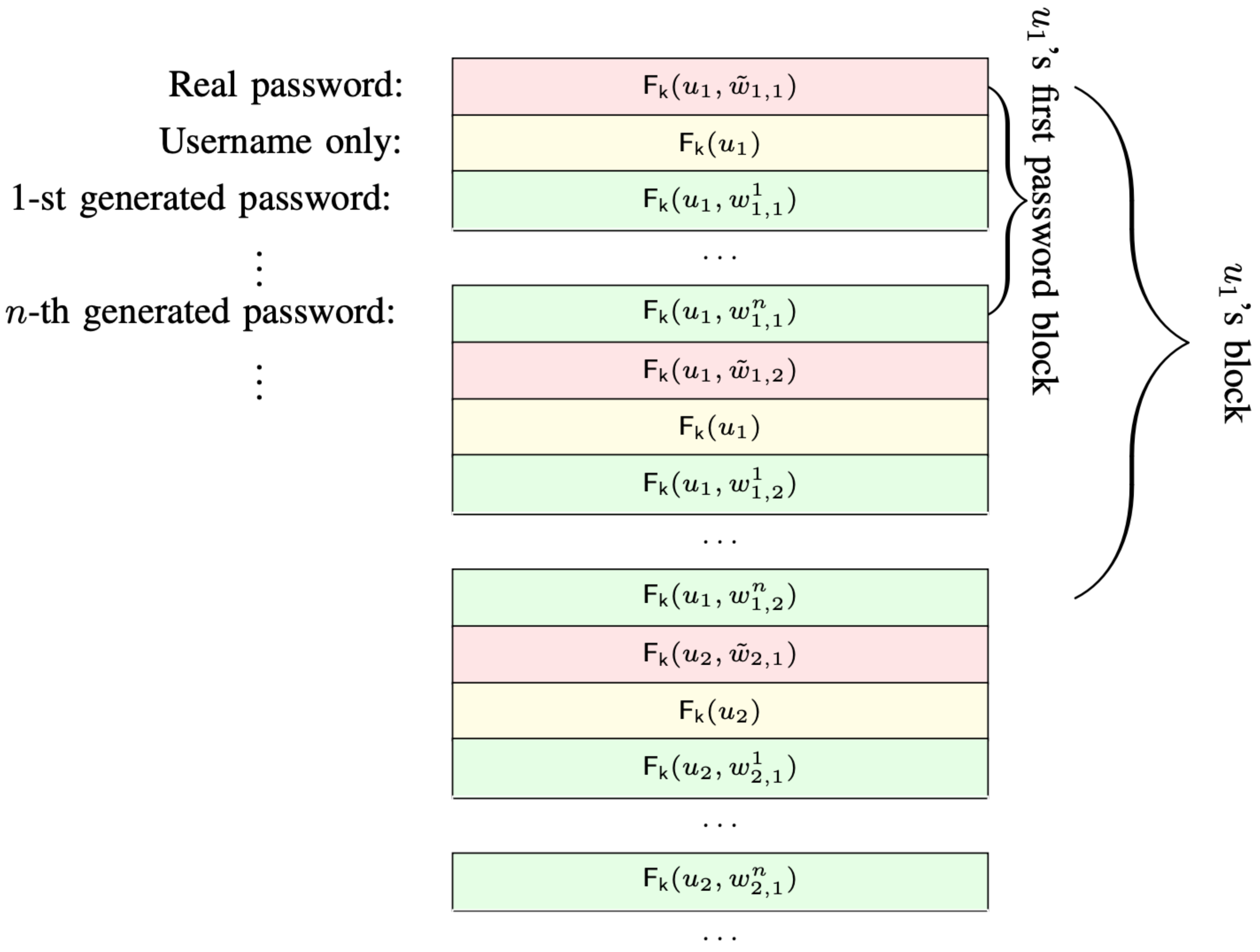
TYPE - 0

TYPE - 1A

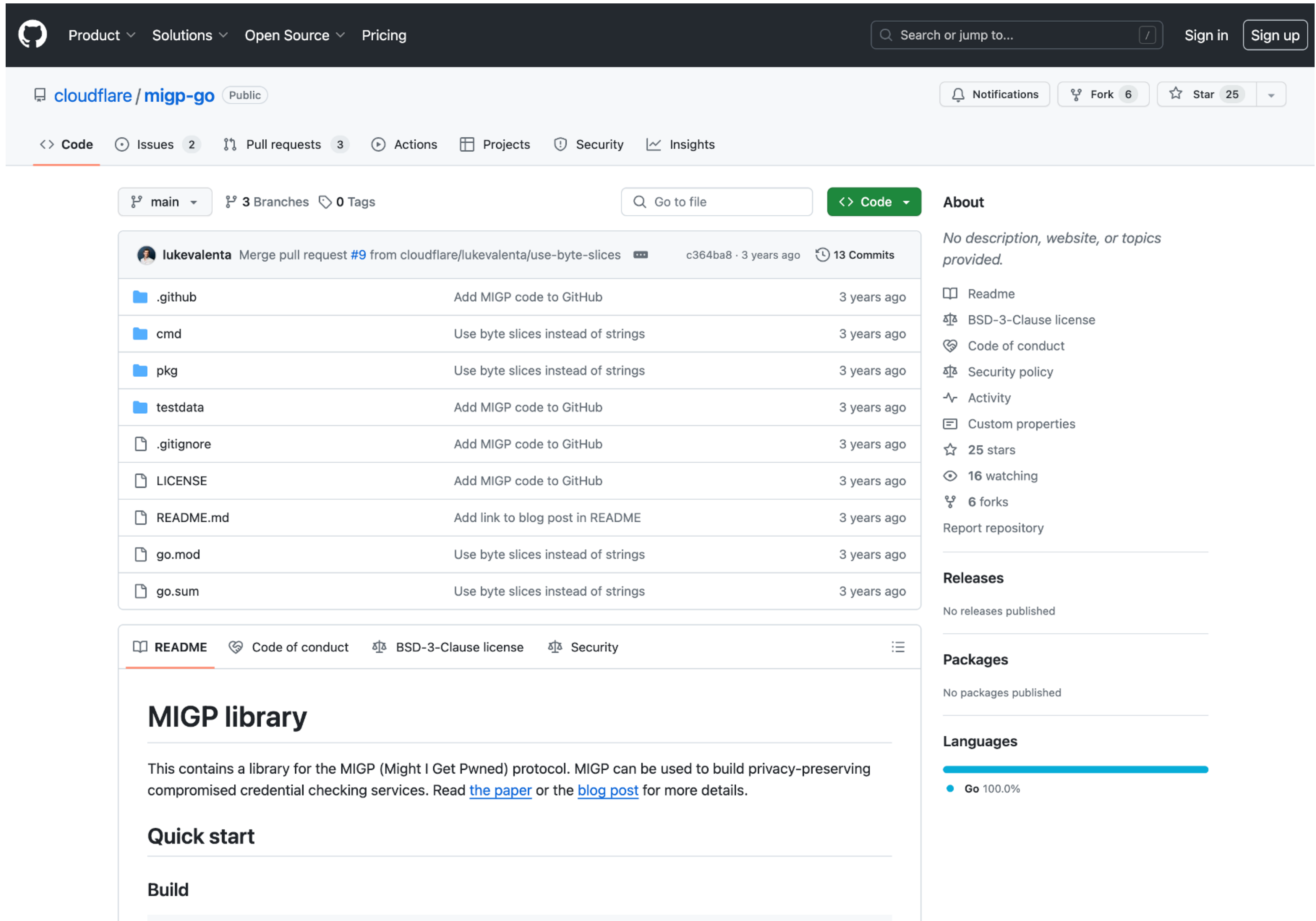
TYPE - 1B

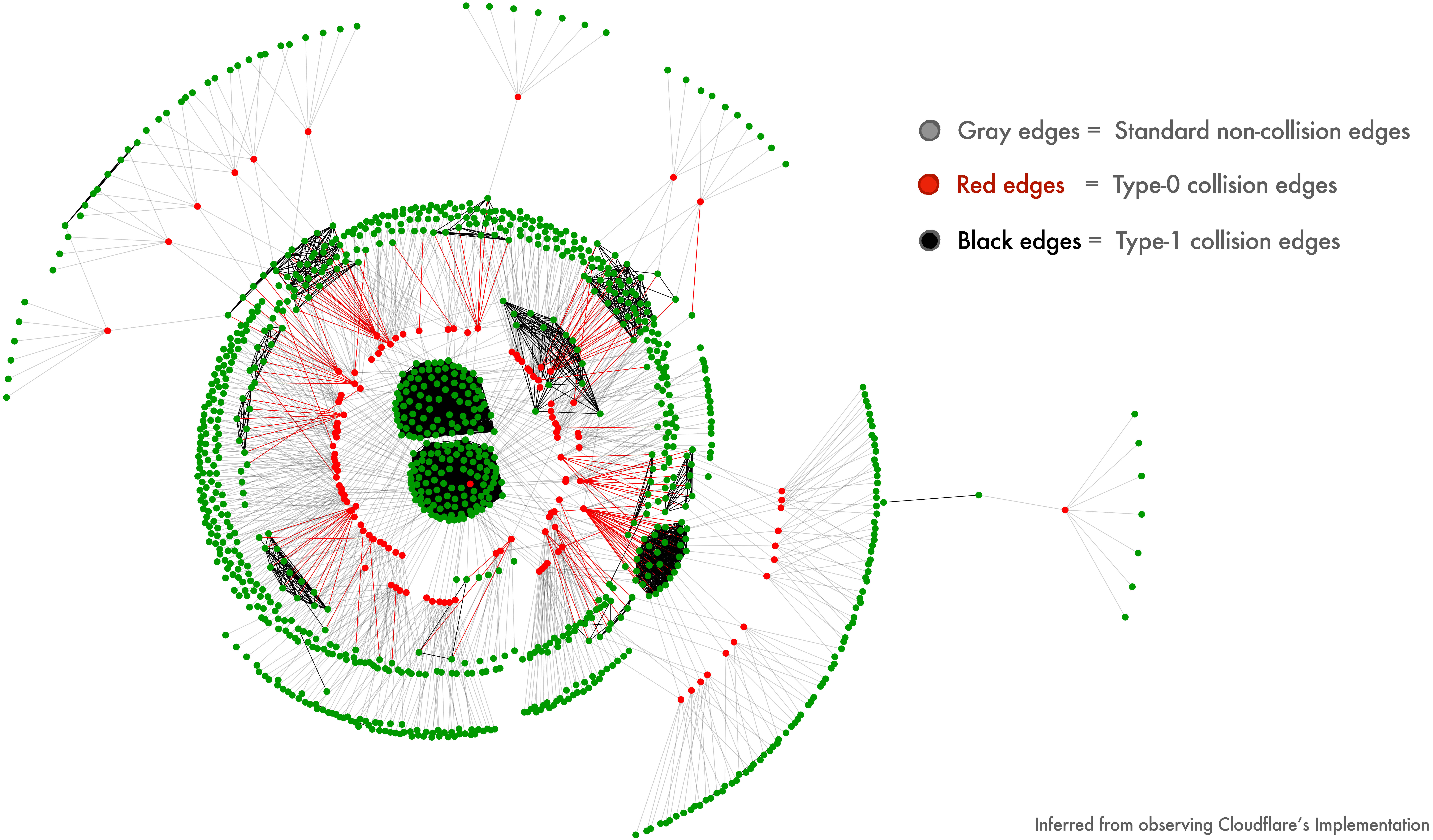


ADDITIONAL LEAKAGE IN CLOUDFLARE MIGP
STRUCTURE OF BUCKET



- Preserved Ordering
- Preserve PRF Duplicates
- PRF Evaluation of the Username





Gray edges = Standard non-collision edges
Red edges = Type-0 collision edges
Black edges = Type-1 collision edges
Inferred from observing Cloudflare's Implementation

BREACH EXTRACTION VIA COLLISION GRAPH
ONE KNOWN PASSWORD AND τ-INVERSION

- **Threat Model**

- Client A tries to infer the credentials of another target user u_{trgt}
- A knows one compromised password (publicly available)
- Given this one password, A guesses the rest of the compromised passwords of u_{trgt}

BREACH EXTRACTION VIA COLLISION GRAPH
ONE KNOWN PASSWORD AND τ-INVERSION

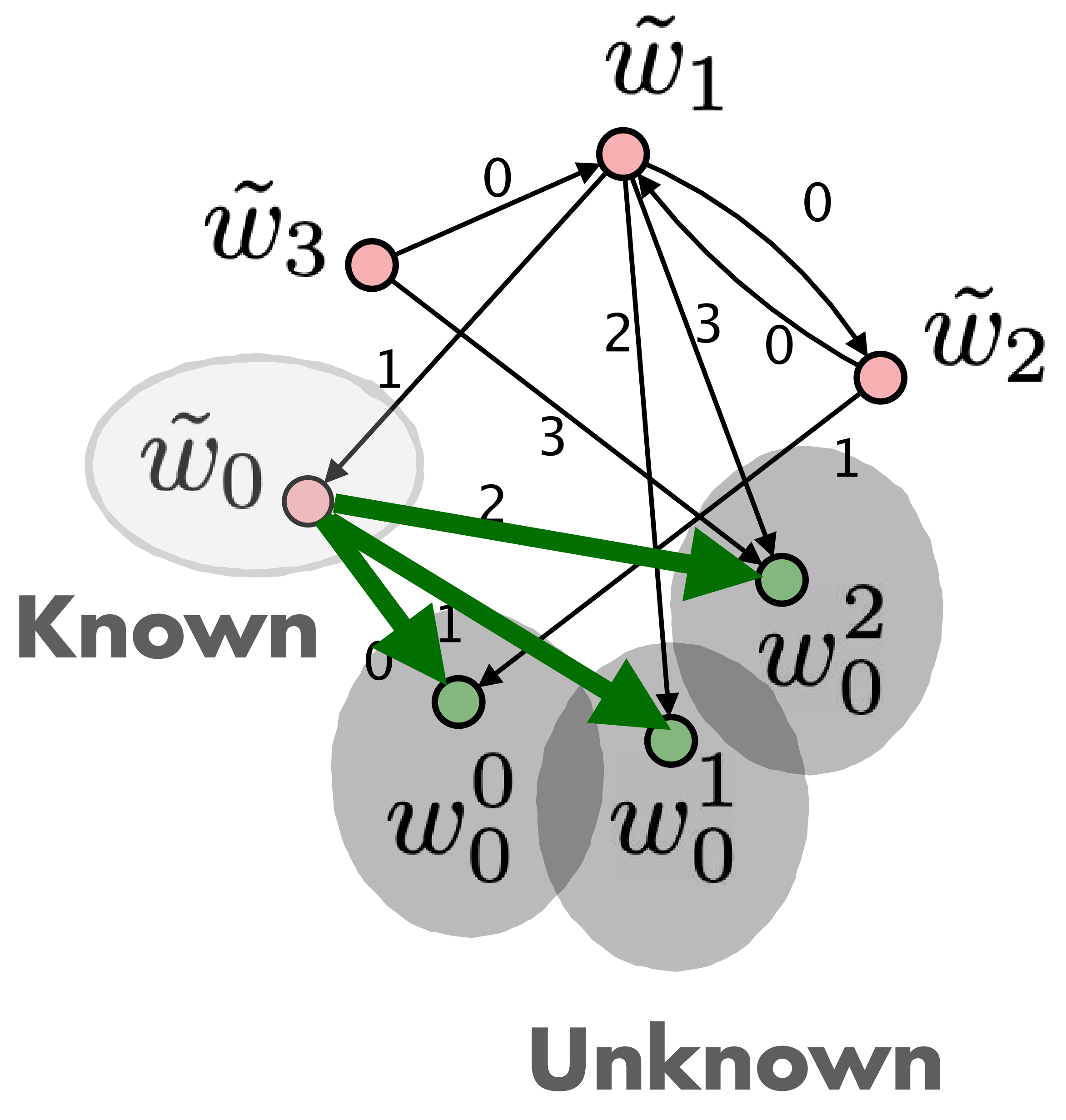
Threat Model

- Client A tries to infer the credentials of another target user u_{trgt}
- A knows one compromised password (publicly available)
- Given this one password, A guesses the rest of the compromised passwords of u_{trgt}

Intuition Using the Collision Graph

Edge (u , v) from Known u to Unknown v

Apply τ function (Das-r/P2P)



BREACH EXTRACTION VIA COLLISION GRAPH
ONE KNOWN PASSWORD AND τ-INVERSION

Threat Model

- Client $\mathcal{A}$ tries to infer the credentials of another target user u_{trgt}
- $\mathcal{A}$ knows one compromised password (publicly available)
- Given this one password, $\mathcal{A}$ guesses the rest of the compromised passwords of u_{trgt}

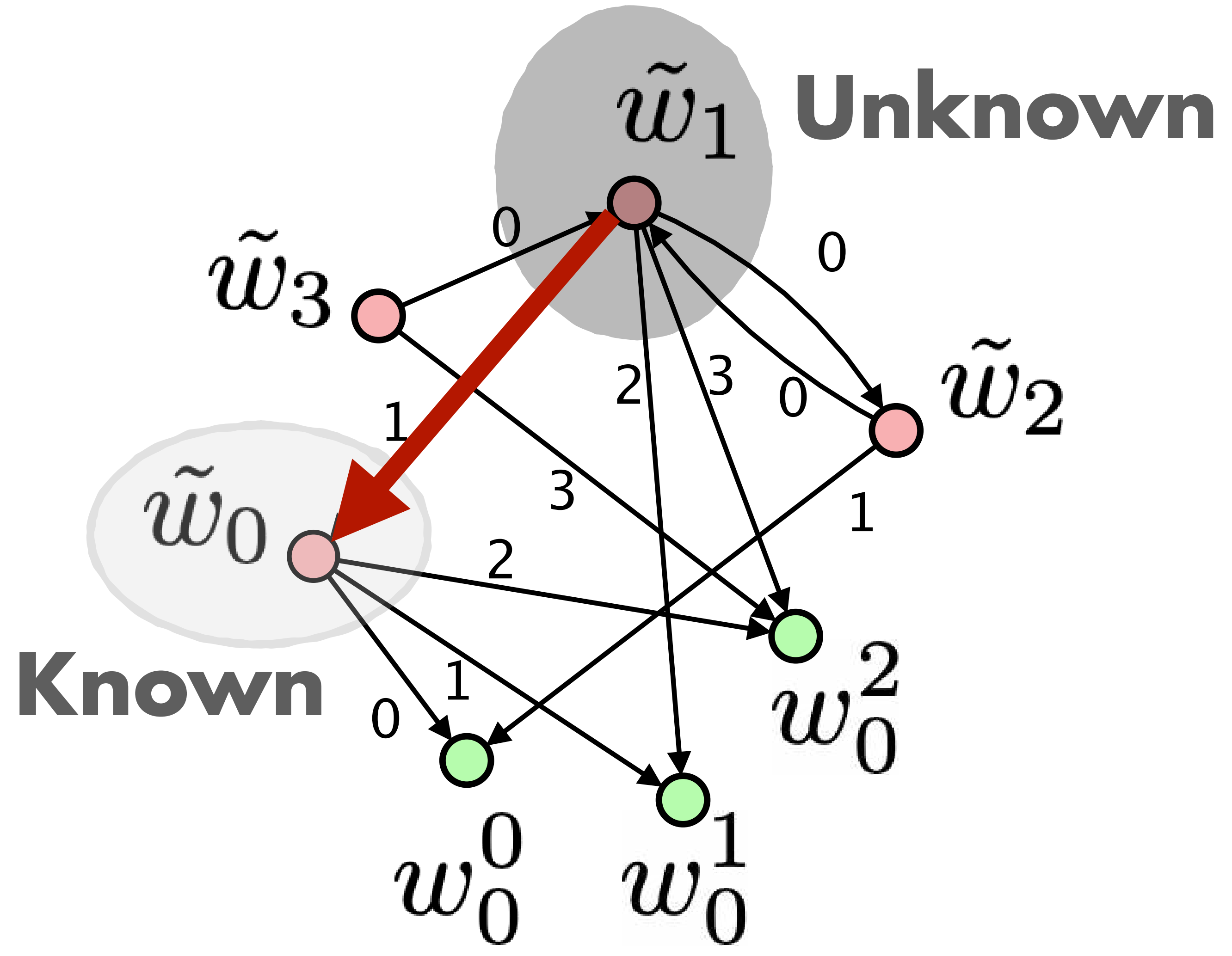
Intuition Using the Collision Graph

Edge (u , v) from Known u to Unknown v

Apply τ function (Das-r/P2P)

Edge (u , v) from Unknown u to Known v

Need to invert τ function!



BREACH EXTRACTION VIA COLLISION GRAPH
ONE KNOWN PASSWORD AND τ-INVERSION

Threat Model

- Client A tries to infer the credentials of another target user u_{trgt}
- A knows one compromised password (publicly available)
- Given this one password, A guesses the rest of the compromised passwords of u_{trgt}

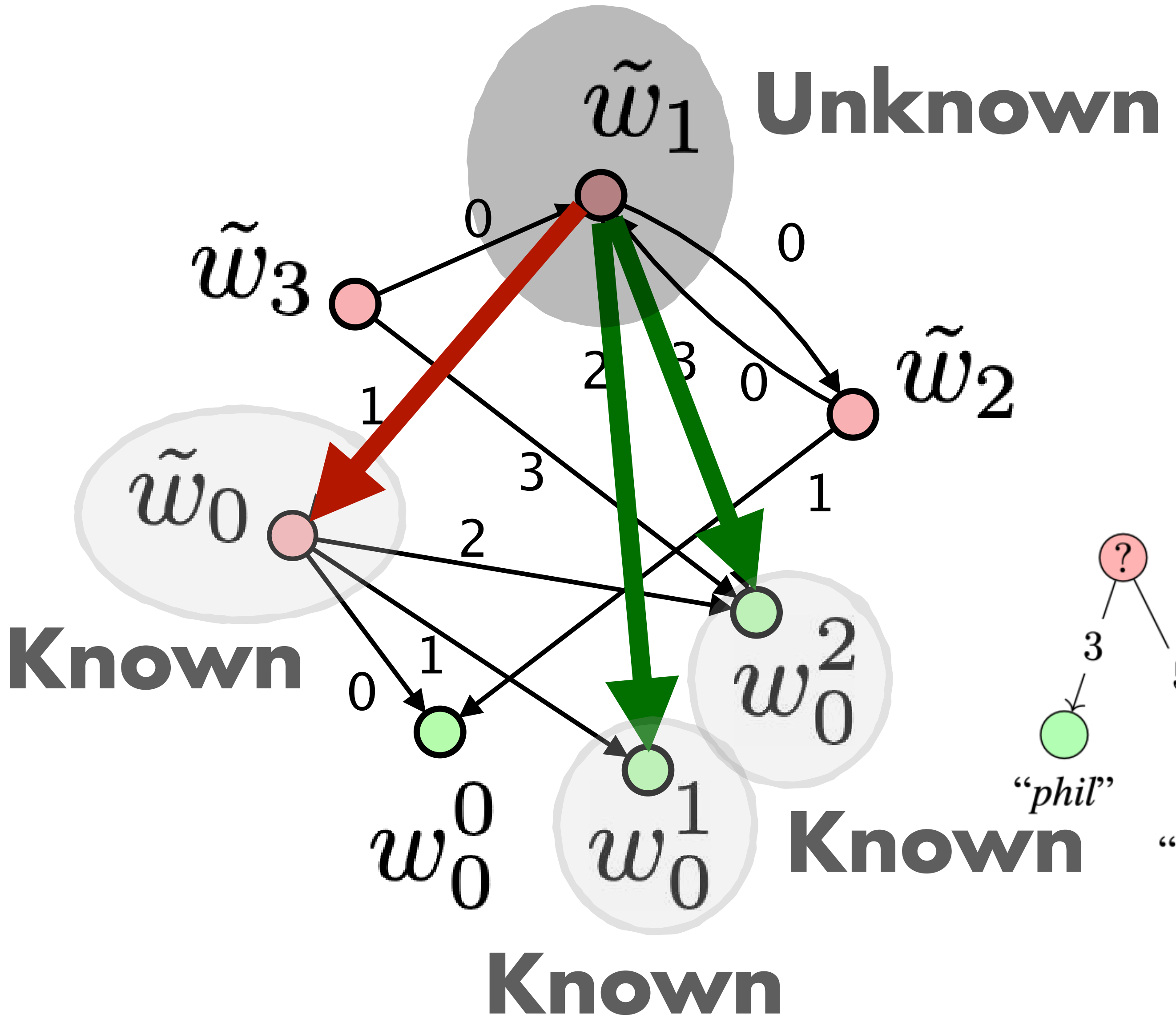
Intuition Using the Collision Graph

Edge (u , v) from Known u to Unknown v

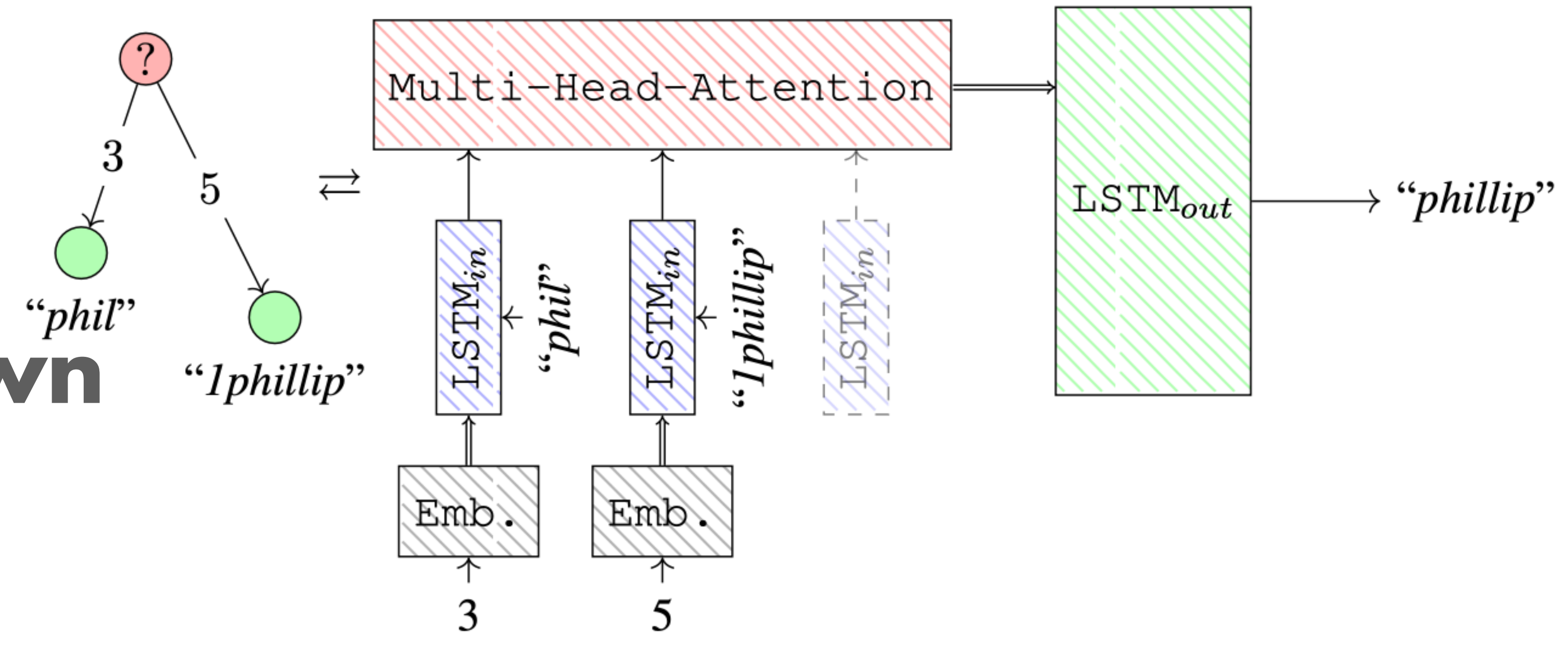
Apply τ function (Das-r/P2P)

Edge (u , v) from Unknown u to Known v

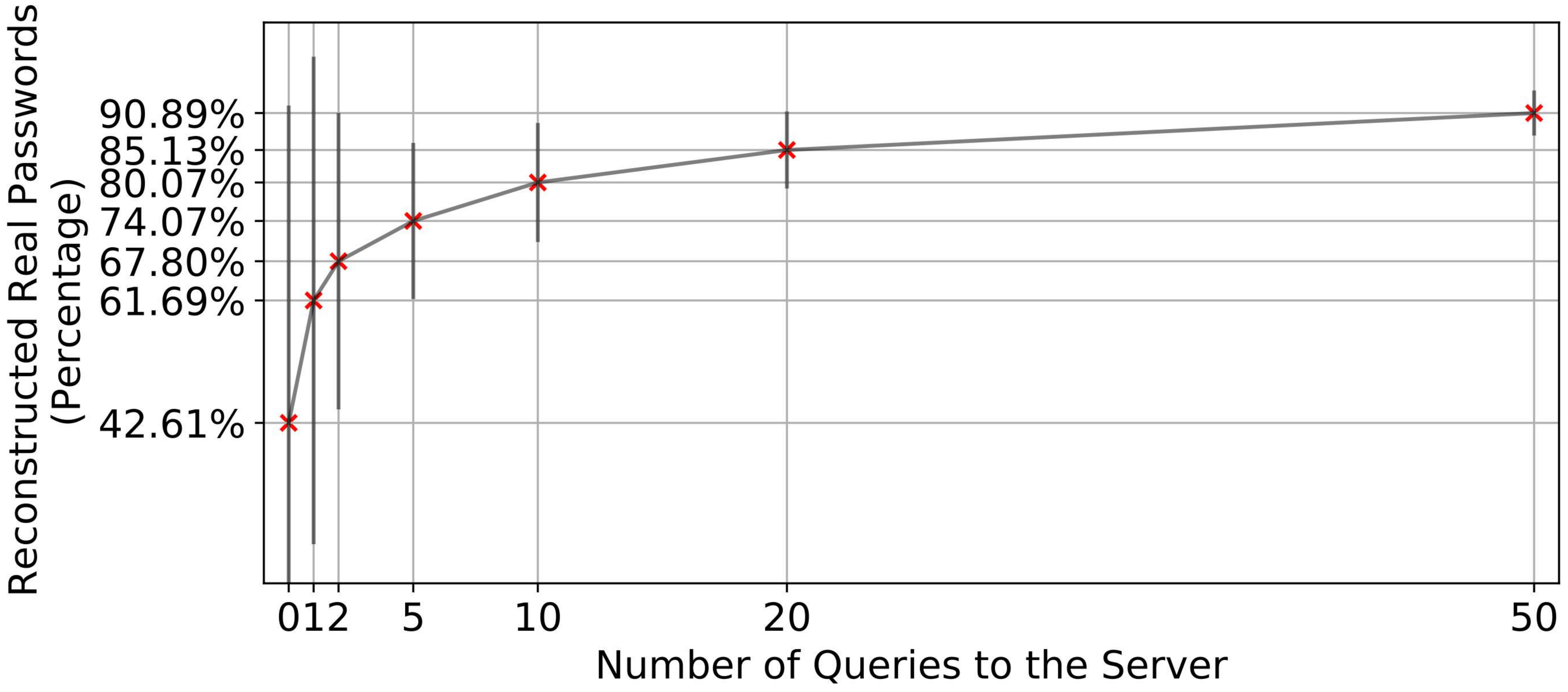
Need to invert τ function!



Deep Learning Architecture



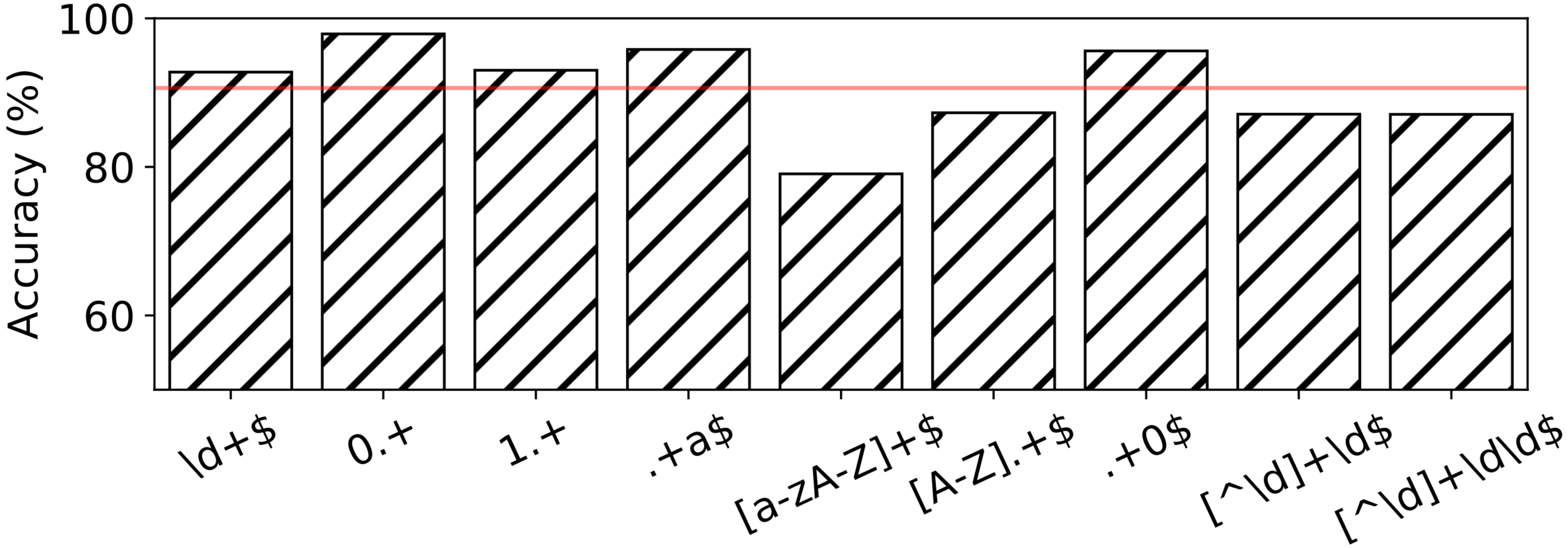
BREACH EXTRACTION VIA COLLISION GRAPH
EVALUATION: ONE KNOWN PASSWORD



BREACH EXTRACTION VIA STRUCTURE OF GRAPH
GRAPH NEURAL NETWORKS AND NO KNOWN PASSWORDS

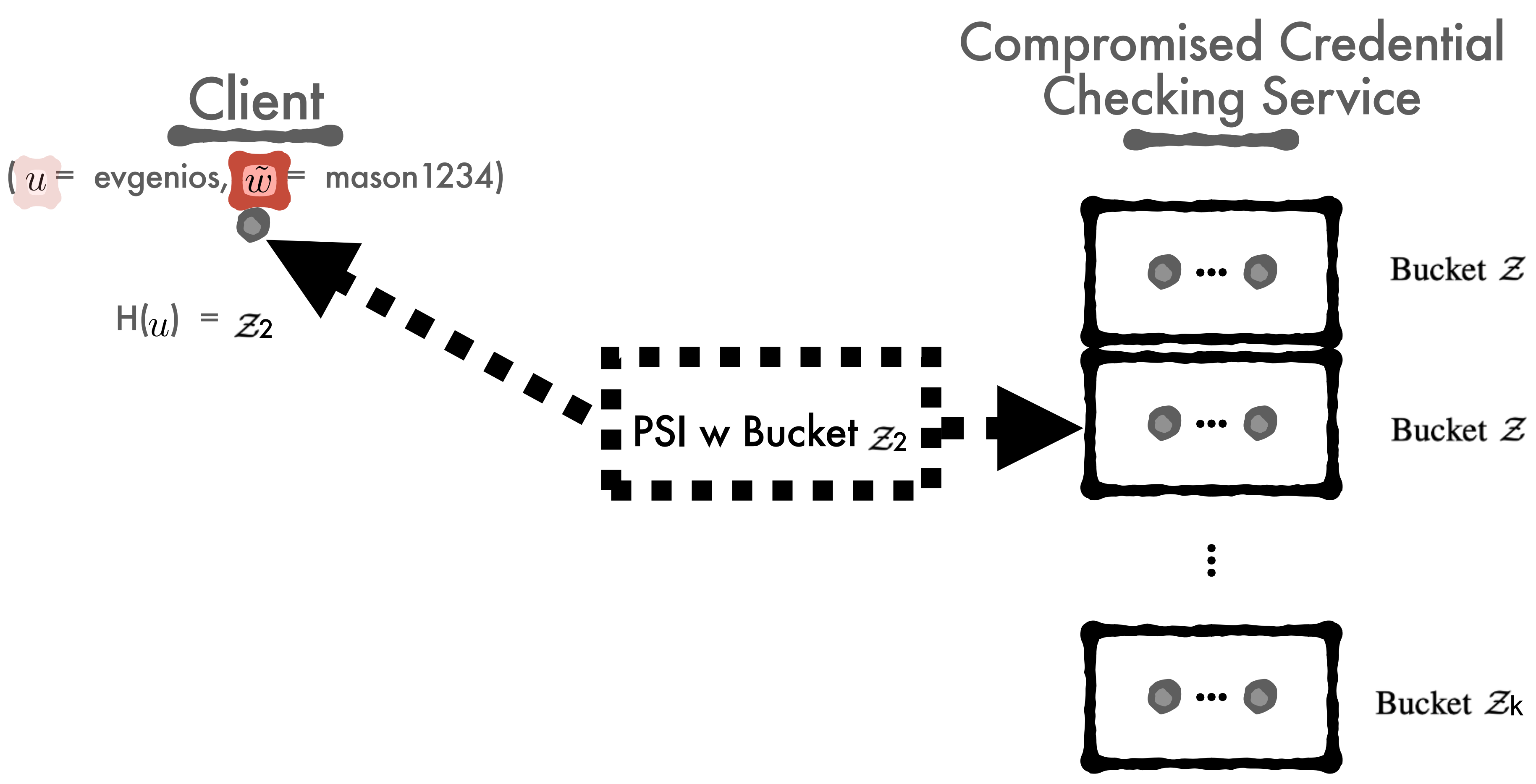
Regex:	**(1)** “\d+$”	**(2)** “0.+”	**(2)** “1.+”	**(4)** “.+a$”	**(5)** “[a-zA-Z]+$”	**(6)** “[A-Z].+$”	**(7)** “.+0$”	**(8)** “[^\d]+\d$”	**(9)** “[^\d]+\d\d$”
Description:	It consists solely of digits.	It starts with “0”	It starts with “1”	It ends with “a”	It consists solely of alphabetic characters	It starts with a capital letter	It ends with “0”	It ends with a digit	It ends with two digits

E.g., “**P**assword**10**” would have template [0,0,0,0,0,1,1,0,1]



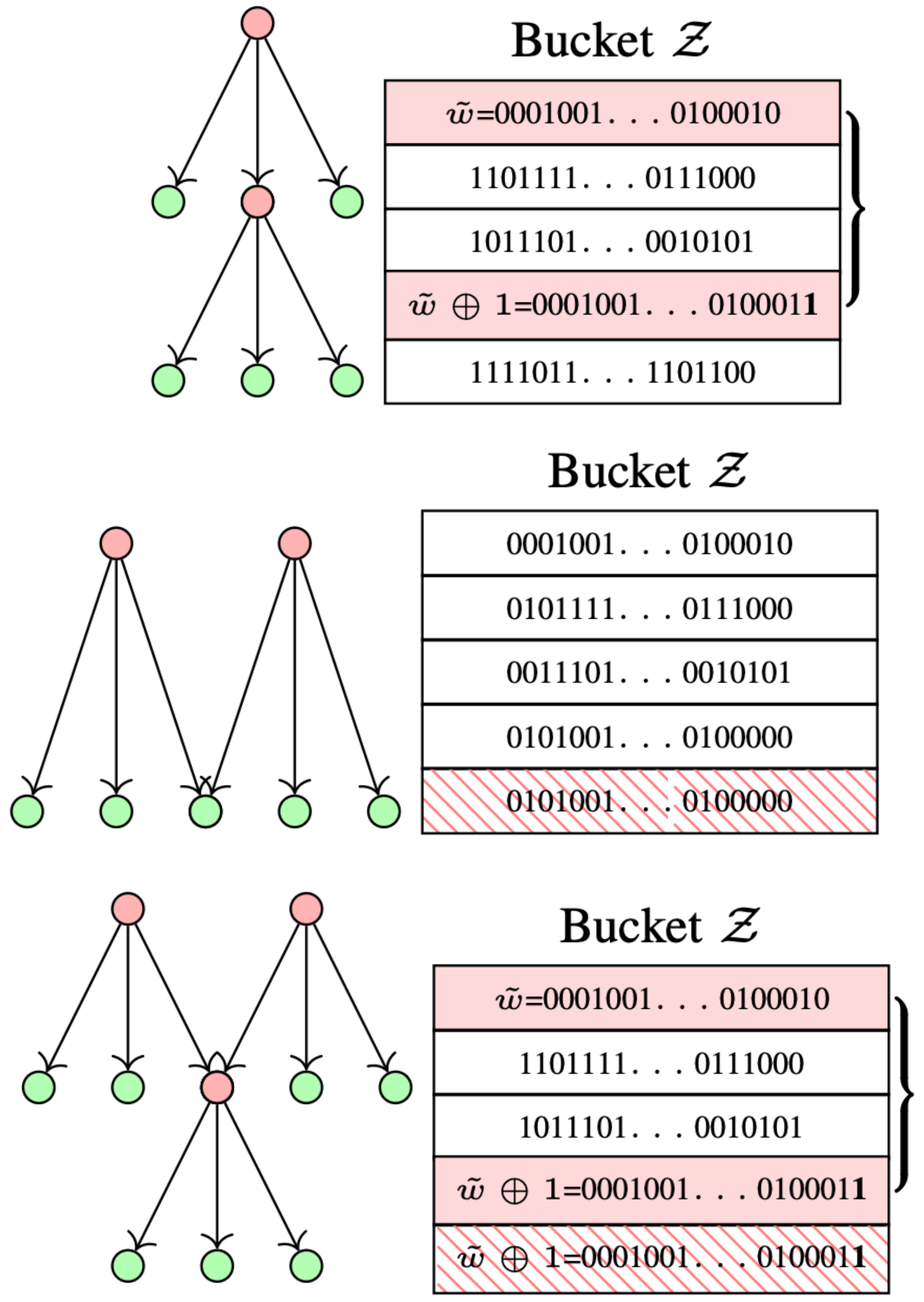
IN SUMMARY
DISCOVERED LEAKAGE IN ASYMMETRIC PSI

Intended Leakage to Speed Up Computation



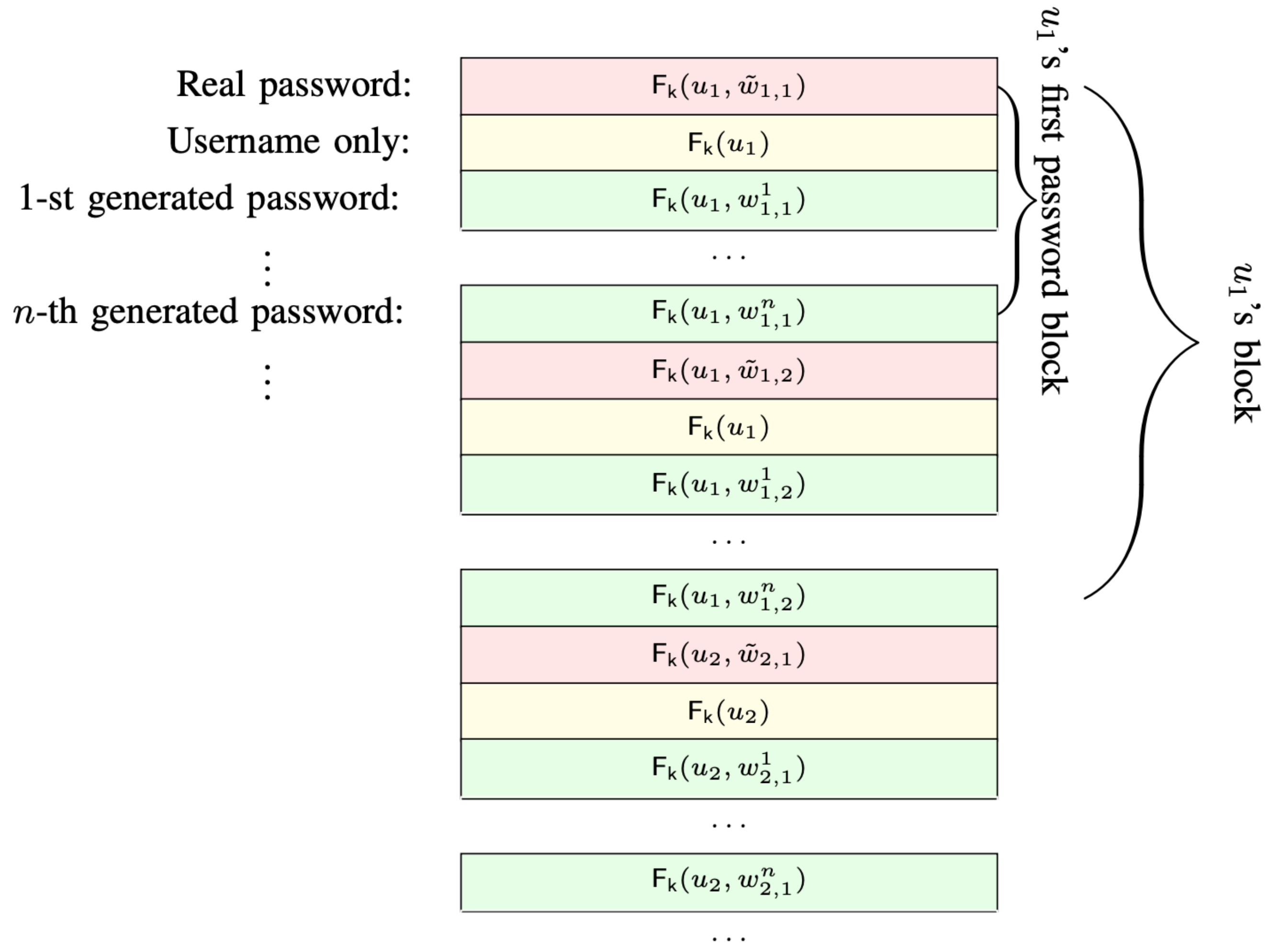
Infer the Username

*New Unintended Leakage Due to Synthetic Passwords



Infer Collisions from τ-function

*New Unintended Leakage from the Implementation



Construct Edges in Collision Graph

PRIORITIZING PRACTICALITY MAY LEAD TO POORLY UNDERSTOOD TRADE-OFF BETWEEN PRIVACY AND EFFICIENCY

PRIORITIZING PRACTICALITY MAY LEAD TO POORLY UNDERSTOOD TRADE-OFF BETWEEN PRIVACY AND EFFICIENCY

Other Takeaways:

1. Leakage might be the only (practical) option when scalability is a must
2. The impact of leakage depends on the context (no one-size-fits-all analysis)
3. Some leakage can be fixed given insightful cryptanalysis


Breach Extraction Attacks: Exposing and Addressing the Leakage in
Second Generation Compromised Credential Checking Services

Dario Pasquini* — SPRING lab, EPFL — dario.pasquini@epfl.ch
Danilo Francati* — Aarhus University — dfrancati@cs.au.dk
Giuseppe Ateniese — George Mason University — ateniese@gmu.edu
Evgenios M. Kornaropoulos — George Mason University — evgenios@gmu.edu

***Abstract*—Credential tweaking attacks use breached passwords to generate semantically similar passwords and gain access to victims' services. These attacks sidestep the first generation of compromised credential checking (C3) services. The second generation of compromised credential checking services, called "Might I Get Pwned" (MIGP), is a privacy-preserving protocol that defends against credential tweaking attacks by allowing clients to query whether a password or a semantically similar variation is present in the server's compromised credentials dataset. The desired privacy requirements include not revealing the user's entered password to the server and ensuring that no compromised credentials are disclosed to the client.**

In this work, we formalize the cryptographic leakage of the MIGP protocol and perform a security analysis to assess its impact on the credentials held by the server. We focus on how this leakage aids breach extraction attacks, where an honest-but-curious client interacts with the server to extract information about the stored credentials. Furthermore, we discover additional leakage that arises from the implementation of Cloudflare's deployment of MIGP. We evaluate how the discovered leakage affects the guessing capability of an attacker in relation to breach extraction attacks. Finally, we propose MIGP 2.0, a new iteration of the MIGP protocol designed to minimize data leakage and prevent the introduced attacks.

1. Introduction

In the evolving cyber threat landscape, attackers target user credentials, particularly those stored in plaintext, exploiting system vulnerabilities to compromise and post them online, thereby breaching user privacy and enabling *credential stuffing attacks* [1]. In these attacks, adversaries exploit widespread password reuse [2], [3], [4], [5] by using credentials exposed from a data breach to attempt unauthorized access to another unrelated domain. Services like "Have I Been Pwned" [6], Google Password Checkup [7], and Microsoft Password Monitor [8]—known as Compromised Credential Checking (C3) services— aim to alert users about the possibility of a credential stuffing attack. Specifically, they allow users to check if their active credentials appear in breach datasets. To accomplish this, C3 services use cryptographic tools to create a privacy-preserving protocol, ensuring that the queried password of the user (which may not be breached) is not disclosed to the server and the sensitive breached credentials are not shared with the client.

However, these services cannot cover an increasingly common type of attack: *credential tweaking attacks* [2], [9], [10]. In these attacks, cybercriminals employ sophisticated techniques to generate slight variants of known breached passwords, enabling them to make distinct educated password guesses towards unauthorized access to the target's services. Unfortunately, credential tweaking attacks are not covered by C3 services since they only check for *an exact match* against the breached credentials. To address these shortcomings, Pal et al. [11] proposed *Might I Get Pwned* (MIGP), a second-generation C3 service. MIGP extends the capabilities of conventional C3 by checking not only for exact password matches but also for semantic similarity with breached credentials. To achieve this, MIGP uses a password-generating function called τ to generate semantically similar passwords during the initialization phase.

The Role of Cryptographic Leakage in MIGP. It is important to note that the privacy-preserving design of MIGP serves (in part) the purpose of safeguarding the collection of breached credential data from being exposed to MIGP query issuers. Paradoxically, despite the MIGP server's data collection being labeled as "breached credentials," it can contain credentials that have been *breached but are not publicly available*. In April 2023 [12], the FBI took down a stolen identity marketplace that was selling non-publicly available breached credentials. To combat credential stuffing attacks, the FBI shared in confidence millions of non-publicly available compromised credentials with HIBP. Thus, real-world C3 services work with breached credentials that should not be exposed under any circumstances.[1]

Our work sheds light on an unexplored aspect of the MIGP protocol: the existence of *cryptographic leakage* over the stored credentials. This leakage is a controlled disclosure intentionally designed into the protocol. The term *breach extraction attack* describes the attack vector in which

*Equal contribution.

1. We note that if the breached credentials of the server are all considered public, then there is no point in deploying a privacy-preserving C3; the server can simply return a subset (i.e., a bucket) of the credentials in plaintext. This change enhances efficiency by forgoing cryptographic operations for non-interactive queries. Additionally, it fortifies defenses against tweaking attacks, allowing users to apply arbitrary similarity functions to leaked passwords.

https://eprint.iacr.org/2023/1848

BREACH EXTRACTION ATTACKS: Exposing and Addressing the Leakage in Second-Generation Compromised Credential Checking Services

PASQUINI, FRANCATI, ATENIESE, **E.M.K.**

Proc. IEEE SECURITY & PRIVACY (Oakland), 2024

Our Contributions:

1. Formalizing the Leakage in MIGP
2. Taxonomy of τ-collisions
3. Breach Extraction Attacks via Leakage
4. MIGP 2.0

Pwnie Award Finalist
"Best Cryptographic Attack"
2024