Hochschule Wismar
University of Applied Sciences Technology, Business and Design
Fakultät für Ingenieurwissenschaften

Master's Thesis

Non-repudiable Password Authentication Using Mutually Authenticated Auditable aPAKE Protocols

date of submission:	24.11.2023
by:	Tobias Reich
	birth date: 13.03.1989
	birth place: Ulm
matriculation n°:	415059
supervisor:	Prof. Dr. Andreas Ahrens
co-supervisor:	Prof. Dr. Antje Raab-Düsterhöft

1 Abstract

English. In this comprehensive thesis, we scrutinize the critical issues and vulnerabilities inherent in password-based authentication. Acknowledging the shift towards alternative methods like Passwordless, which prioritize ownership over knowledge, our research delves into the underlying essences. This analysis dissects the transition from symmetric to asymmetric cryptography, the relocation of authentication processes from webpage source codes to the browser/OS, and the replacement of knowledge-based authentication with ownership-based approaches. These insights lead to innovative strategies for refining password-based web authentication. The exploration highlights the efficacy of asymmetric protocols such as OPAQUE, and A²PAKE. As a practical contribution, the thesis we propose an enhancement of A²PAKE with mutually-explicit authentication, thereby mitigating the prevalent vulnerability of PAKE protocols against one-time online password guessing attacks. Overall, the thesis provides a holistic view of the current state and potential future of password-based authentication, and highlight the potential for its cryptographic audibility.

German. In dieser umfassenden Arbeit untersuchen wir die kritischen Probleme und Schwachstellen, die in der passwortbasierten Authentifizierung inhärent zu sein scheinen . Unter Anerkennung des Trends hin zu alternativen Methoden wie Passwordless, die besitzbasierte über wissensbariete Authentisierung priorisieren, vertiefen wir uns in die zugrundeliegenden Essenzen dieser Methoden. Diese Analyse erkennt den Ablösung von symmetrischer durch asymmetrische Kryptographie, die Verlagerung der Authentifizierungslogik von Webseiten-Quellcodes in den Browser/das Betriebssystem und den Ersatz der wissensbasierten Authentifizierung durch eigentumsbasierte Ansätze. Diese Erkenntnisse führen zu innovativen Strategien zur Verfeinerung der passwortbasierten Webauthentifizierung. Die Untersuchung hebt die Wirksamkeit asymmetrischer Protokolle wie OPAQUE und A²PAKE hervor. Als praktischen Beitrag schlagen wir in dieser Arbeit eine Verbesserung von A²PAKE mit explizit gegenseitiger Authentifizierung vor, wodurch die verbreitete Schwachstelle von PAKE-Protokollen gegenüber einstweiligem Online-Passwort-Erraten mitigiert wird. Insgesamt bietet die Arbeit einen ganzheitlichen Überblick über den aktuellen Stand und die potenzielle Zukunft der passwortbasierten Authentifizierung und hebt das Potenzial für ihre kryptografische Überprüfbarkeit hervor.

Inhaltsverzeichnis

1 Introduction, Motivation and Objectives

In this chapter, we give a broad overview over the subject matter and main objectives of this thesis.

In order to motivate for both, we are starting off with two practical everyday scenarios that harbor potential for disputes as well as mutual accusations between service provider and their users, thereby illustrating the lack of password secrecy as well as non-repudiation in prevalent password authentication schemes. We will then define a problem statement followed by the corresponding research and developments objectives and enhance those by an *ideal solution scenario*, contrasting the preceding, motivational dispute scenarios. Thereafter, we note some contextual aspects of forensics, cybercrime, trust, and user awareness, to consider.

Closing the chapter, we are giving a comprehensive overview of the thesis outline and in which ways and means its chapters depend on and relate to each other.

1.1 Motivation

In this section, we aim to motivate the thesis through a vivid discussion of delicate real-world scenarios that are intimately familiar to most digital users. In an era marked by an ever-increasing reliance on digital platforms for a myriad of personal and professional activities, alongside the steady rise in sophistication of data breaches and cyber threats, the imperative to safeguard password integrity and ensure reliable user authentication has escalated dramatically.

This underscores the critical importance of secure password authentication and brings into focus the challenges and repercussions associated with password security and non-repudiation in online transactions.

We explore these issues through the inherently conflict-prone scenarios of:

- *Password security and compromise*, and
- *Repudiation of online transactions*

in the context of *usual* password authenticated services, and conclude with an examination of the significant impact that *non-repudiable* password authentication could have in resolving such disputes over password-compromise and online transactions, thereby enhancing online transaction security.

Behold. Before delving into the specific scenarios of password security (1.1.2) and non-repudiation (1.1.3) in online transactions, it is pertinent to distinguish between the terms 'Confidentiality' and 'Secrecy' within this context.

1.1.1 Password Security: Confidentiality vs. Secrecy

While 'confidentiality' and 'secrecy' are often used interchangeably in everyday language, these concepts hold distinct meanings in the realm of password security and have significant implications for the integrity and non-repudiation of password-authenticated services.

This brief discourse aims to elucidate these terms, setting the groundwork for a deeper understanding of the challenges and solutions presented in the subsequent scenarios.

Disambiguation. In the realm of cybersecurity, both terms often intersect yet serve distinct purposes, particularly in the context of password authentication and non-repudiation.

Confidentiality typically refers to the protection of information from unauthorized access or disclosure. In the case of password authentication, this is exemplified by practices such as securely transmitting *passwords over TLS* connections, safeguarding them from potential interception like man-in-the-middle attacks.

Secrecy on the other hand, implies a deeper level of concealment, where information is not only protected from unauthorized access but is kept undisclosed to any external party, including the authentication server. In terms of non-repudiation, secrecy takes on a critical role, as the user is committed to withholding the true secret (the password) from everyone, including the server they're authenticating with.

> "The fundamental password problem is simple to explain, but hard to solve: A password that leaves your possession is guaranteed to sacrifice security, no matter its complexity or how hard it may be to guess. Passwords are insecure by their very existence."
> — T. Bradley, in her Cloudflare blog post *OPAQUE:*
> ***The Best Passwords Never Leave your Device*** [**Bra20**]

This distinction underlines the nuanced application of these terms in different security scenarios and their implications in the broader context of digital trust and authentication integrity.

1.1.2 Dispute Scenario: Password Security and Compromise

Before we immerse ourselves in this delicate scenario, let us first think about how 'secure' password-based authentication is *usually* implemented.

When we think *password security*, as technically adept readers, there are typically three main bullet points or check-marks on our mind:

- Password *complexity* and *length*, as well as their *enforcement*,
- Encrypted transmission, i.e. via *TLS*, as previously discussed, and
- 'Encrypted' storage of passwords, i.e. storing only their *salted hashes* as verifiers.

Plain and simple. Even though long & complex, sent through a secure tunnel, resting on the server salted & hashed; the password is nevertheless being sent in *plain-text* - also known as 'PLAIN over TLS'.

Scenario. As an inescapable infliction by the prevalent PLAIN-over-TLS scheme, a malicious, compromised or accidentally mis-configured or mis-implemented server could potentially *compromise our password*—the one way or the other—and it may be hard if not impossible to tell which way it was compromised. Various *blame-games* might emerge between users, software vendors, site owners, and server administrators, after the fact. The latter can hypothetically accuse the users of *re-using their passwords*—and blaming another server where they have used the same password for the breach—or falling victim to a password harvesting or phishing campaign. As the blame-game ensues, the underlying issue often traces back to compromised passwords, a vulnerability that opens the door to a myriad of detrimental outcomes. Password compromise can lead to significant consequences, including *unauthorized access* to sensitive personal and corporate data, financial loss, and damage to the reputation of individuals and organizations.

Anecdote. Remember when in 2018, all of its 330 million users were asked by a popular micro-blogging platform to *change their passwords* — this was due to a 'bug' in a 'technology' that company used to 'masks'[1] the password, which caused those passwords to be stored "unmasked in an internal log," as described it in their official statement[aut18] .

Conclusion. Such breaches not only jeopardize *privacy* but also erode *trust* in digital services, underscoring the critical need for *robust* password protection mechanisms, that enable actual *secrecy* of the passwords.

1.1.3 Dispute Scenario: Repudiation of Online Transactions

Scenario. Imagine a scenario where a financial service provider faces a severe security breach. The server's integrity is *compromised*, or a subset of users falls prey to a sophisticated phishing campaign. Following this incident, a series of *unauthorized transactions* are conducted, leading to a contentious dispute between the users and the service provider. Users assert that the service has been hacked and *repudiate* their transactions, demanding

[1] salted hashing

compensation. Conversely, the service provider contends that the users must have succumbed to a phishing attack.
The crux of the issue lies in the inability to substantiate or refute either claim. Traditional password and multi-factor authentication systems lack the technical foundation to enable *non-repudiation* of—yet 'authenticated' and 'authorized'—online transactions; which in turn, can only be achieved through *cryptographic auditability*, leaving both parties entangled in a blame-game.

Conclusion. In scenarios where non-repudiation is absent, the resolution of disputes becomes a complex, often murky process. Without clear, *verifiable evidence* of each party's involvement, disputes can quickly devolve into a cycle of accusations and counter-accusations. The lack of a definitive way to prove *by whom* the repudiated transaction 'must have been'—or 'may not necessarily have been'—initiated or authorized leads to prolonged investigations, unresolved conflicts, and eroded trust among users and service providers alike.

1.1.4 Resolving Disputes: Signatures vs. Forensics

Unraveling the digital truth. While a thorough *digital forensics* investigation might eventually unravel the truth in both scenarios, such a process often faces *reluctance* from service providers, who may only comply under *legal compulsion*.

Investigation versus audit. The aforementioned standoff could be effectively circumvented if *digital signatures*, providing cryptographic non-repudiation, were integrated into the authentication and transaction processes. The use of digital signatures would enable straightforward *audits* of transactions against reliable verifiers, i.e. public keys to which users have committed via trusted registration authorities — in a legally binding way.

Delicate consequence. In such a framework, service providers would find it difficult to justify their reluctance, as the need for comprehensive forensic investigations could be replaced with audits focusing on the digital signatures in question.

Conclusion. This approach would not only streamline the process of uncovering the digital truth behind the repudiated *authorship* of an online transactions, but also put an end to the prevalent blame-game in dispute resolutions.

1.2 Problem Statement

1.2.1 Real-World Incidents and Password Vulnerabilities

In recent years, notable incidents have highlighted the critical vulnerabilities in password-based authentication systems.

Twitter. For instance, the 2018 Twitter bug inadvertently logged user passwords in plain text, exposing them to potential internal misuse.

Facebook. Similarly, Facebook, in 2019, faced a significant security lapse when it was discovered that millions of user passwords were stored in a readable format, accessible to numerous employees, raising serious questions about password 'confidentiality' and user trust [159†source].

Zuckerberg. Even more concerning was the 2004 case of Mark Zuckerberg, who exploited password failures to access Harvard email accounts, demonstrating a blatant disregard for 'secrecy' and the sanctity of user data [165†source].

Conclusion. These incidents not only underscore the susceptibility of password systems to both external and internal threats but also highlight the fundamental problem of password 'secrecy' versus 'confidentiality.'

1.2.2 Challenges in Web Authentication

As already pointed out in the previous scenarios and discussion thereof, the both convenient and conventional practice of having users entering their credentials—not necessarily limited to, but of critical relevance in the case of their passwords—into a webpage's login form and transmitting them to the web server as a 'Password-over-TLS' , is susceptible to various threats.

Salted hashes to the rescue. While by far and large almost all commonly used password-based authentication systems and especially websites— i.e. web servers in general—nowadays store passwords in the aforementioned salted-hash format, in order to prevent adversaries from the unobstructed extraction of plaintext passwords, in the event of a database breach through e.g. a SQL injection vulnerability, this is—though 'appropriate'—still *insufficient* to fully mitigate the risk, in the scenario of server compromise.

Database breach vs. server compromise. While servers usually comply with storing passwords only salted & hashed, a web server compromised through e.g. a Remote (RFI) or Local File Inclusion (LFI), other Remote Code Execution (RCE), or Upload vulnerability, could still be manipulated to exfiltrate[2] passwords sent for login authentication.

aPAKE to the rescue. Even the implementation of aPAKE—i.e. think of 'zero-knowledge'—protocols, which are designed almost entirely for the purpose of enhancing password security i.e. enabling actual *secrecy* of the password, is—though 'appropriate'—still *insufficient* to fully mitigate the risk, in the prevalent scenario of web-based applications with integrated authentication mechanisms.

[2] instruct the web server's code to copy the password from each received HTTP POST request (for PLAIN-over-TLS authentication), and send it to the attacker's infrastructure in a new—ideally accumulating and obfuscated/covert—outgoing message.

XSS, or 'client-compromise'. In the aforementioned scenario of code-execution related server compromise—and furthermore even in SQL injection, stored and reflected XSS, as well as client-targeted file upload attacks—the adversary might succeed in compromising—in addition to or instead of the server—the client, and thereby exfiltrating[3] the user's plaintext password as it is being entered into the infected webpage.

MITM and phishing. Even adversaries unable to successfully compromising the server, could still undertake a man-in-the-middle (MITM) attack against certain users of their significant interest — and yet succeed with injecting said malicious code. Furthermore, any webpage can be cloned and used by an adversary to launch social-engineered phishing attacks against either a dedicated user (or group) i.e. spear-phishing or whaling; or to vast masses of users i.e. usual phishing emails, leading to potential password harvesting.

Phishing aPAKE. The paper *Auditable Asymmetric Password Authenticated Public Key Establishment* by Faonio et al. [Fao+22], highlights the limitations of existing aPAKE protocols that—no matter how sophisticated their design, regardless of the security of their implementation, even without breaching neither server nor client—still allow for one online-guess of a candidate password against the client, during server impersonation i.e. phishing attacks.

Password-less to the rescue. These—supposedly inevitable and seemingly unsolvable—issues with passwords, have led leading tech firms to advocate for alternate authentication mechanisms under the banner of 'password-less.' In the interest of thoroughness, the most notable to mention are: 'Magic Links' (proof-of-possession of an email account), FIDO2 (see next chapter) and WebAuthn, as well as App- or Operating System-enabled, and Biometrically enabled Passkeys.

The password is dead; long live the password. Despite the persistently growing adoption and steady popularization of password-less i.e. proof-of-possession or SSO-based methods, it is undeniable that, there still remains a significant demand and plenty of preferred use-cases for secure password-based authentication, as to be discussed in the following subsection.

Conclusion. These scenarios underscore the challenges associated with password confidentiality, phishing mitigation, user authenticity and mechanisms for their proof-of-identity, as well as non-repudiation in password-based authentication schemes.

1.2.3 Risks and Trust in aPAKE Protocols

Amidst the advancements in web authentication, asymmetric PAKE (aPAKE) protocols have emerged as a potential cornerstone technology. However, they inherently grapple with

[3] instruct the malicious e.g. JavaScript code for key-logging, sending the logged key-strokes to the attacker's infrastructure.

a critical vulnerability: the possibility of online password guessing in each authentication attempt. This long-standing issue remains a significant challenge in the field.

Overdue. Despite three decades of extensive research in PAKE and aPAKE protocols, this vulnerability is often perceived as an 'intrinsic' or 'natural' flaw. The persistence of this risk, unchanged over years, raises questions about the evolution of aPAKE protocols and their ability to adapt to modern cybersecurity threats.

Try again. n practical scenarios, particularly phishing attacks, this vulnerability can be exploited by attackers, who can systematically attempt to guess passwords. Users, unaware of the deceit, might persist in trying to authenticate, each attempt providing the attacker with another chance to guess their password from a list of 'most common passwords' — or even more alarming, from an individually crafted list[4]. This way, a phishing server could iterate though such a password list, as long as the user keeps trying to login — while trusting and believing the remote site that keeps reporting a "network error", and inconspicuously prompting to "please reload and try again."
This creates a cycle of vulnerability, undermining the security of the authentication process.

Trust issues. The reality of cyber threats and the psychology of cyber criminals suggest that any exploitable vulnerability will eventually be targeted. This inevitability erodes the advanced trust users may potentially place in aPAKE protocols, especially among those who are less experienced, or would simply expect a prospective solution to be 'completely' secure.

Seeking Balance. Addressing this flaw requires a delicate balance between enhancing protocol security and maintaining user accessibility. The *ideal* implementation of an aPAKE protocol would eliminate these vulnerabilities without adding undue complexity; thereby preserving advanced trust in, and user satisfaction with, aPAKE-based authentication.

A Call for Innovation. The ongoing risks associated with aPAKE protocols underscore the urgent need for innovative solutions that can effectively counteract aforementioned online-guessing attacks and phishing attempts, setting a new benchmark in secure online authentication.

1.2.4 Digital Sovereignty and Authentication Choices

> Authentication is not only a matter of security, but also a question of *digital sovereignty*.

[4] to attack each victim individually, with a password candidate list, automatically (i.e. AI-assisted) generated from OSINT available information about the victim, or from a specific password-pattern linked to the victim's email address discovered is a data breach.

In the rapidly evolving landscape of digital security, the concept of 'digital sovereignty' has become increasingly significant. This notion extends beyond mere protection from unauthorized access, embodying the user's autonomy and right to make informed choices about their authentication methods; and in certain use-cases, passwords may simply remain the *preferred option*. Therefore, it is imperative to develop and implement password-based authentication in the most secure manner possible, addressing the aforementioned vulnerabilities and threats.

Proof-of-Identity Dilemma: Knowledge v.s Possession. Currently emerging and popularly promoted authentication mechanisms, such as those exemplified by FIDO2, rely on the 'exclusive possession' or at least 'physical control' of *cryptographic keys*—in the form of devices or tokens—serving as the proof-of-identity for their owners. Asserting identity with 'exclusive knowledge' on the other had, is typically exemplified by password-based methods, offering a familiar and direct form of authentication.
This distinction is pivotal; while possession can be rightfully transferred, illicitly stolen, or even legally confiscated, knowledge is inherently personal. This dichotomy at the same time constitutes a dilemma for both user autonomy and technological sovereignty.

User Autonomy in Digital Identity. As outlined above, the proof-of-identity contrast is not just a technical detail; it reflects a fundamental decision about how users interact with and control their digital identities. For instance, while Proof-of-Possession offers robust security against certain types of attacks, it may limit user flexibility and choice under certain circumstances. Conversely, Proof-of-Knowledge places control squarely in the hands of users but often at the expense of increased vulnerability to social engineering and phishing attacks.

Balancing Security and Sovereignty. Within the scope of our thesis, we delve into how these two paradigms can be balanced or even *reimagined* to enhance both security and sovereignty. The proposed solution aims to navigate these complexities, offering an approach that not only secures the user's digital identity but also empowers them with meaningful choices in how they authenticate.

Concluding, the journey towards robust digital authentication is not just about thwarting cyber threats; it's equally about upholding the user's right to choose and control their authentication methods. Our work seeks to contribute to this journey, paving the way for systems that respect and enhance digital sovereignty.

1.2.5 Limitations of Current Authentication Mechanisms

In addressing the limitations of current authentication schemes, demand for the proof-of-identity mechanism of choice, and corresponding challenges for password-based authentication and web protocols, our thesis advocates for a robust approach that upholds 'digital sovereignty' — the user's control over their digital identity and the freedom to choose

secure authentication methods, whether they may prefer emerging and promoted schemes based on proof-of-possession, or are still confident and content with equally secure i.e. secrecy-preserving proof-of-knowledge schemes.

This approach is essential in an era where traditional password systems are increasingly inadequate against sophisticated cyber threats.

In forthcoming chapters, including detailed discussions on—identified by us as such—'Essences' of FIDO2, we delve into how emerging technologies accomplish the fortification of user authentication, and derive applicable 'lessons learned' for our proposed solution approach, while aligning with the principles of digital sovereignty and offering users a much-needed 'freedom of choice' in securing their digital presence.

1.3 Objectives

The primary objective of this thesis is to develop a password authentication protocol that effectively counters aforementioned prevalent risks associated with online password authentication, and especially online guessing attempts; with appropriate i.e. proportionate measures.

To achieve this, the proposed solution emphasizes the following key aspects:

- **Elimination of Online Guessing Attacks**: The protocol is designed to completely negate any possibility of online guessing attacks, ensuring that attackers have no feasible means of exploiting password vulnerabilities.
- **Introduction of Individual Key-based Password Fingerprints**: A novel aspect of the protocol is the use of individual fingerprints for each user, and their password or user-selected pool of passwords. Hence, in contrast to all prior approaches[5], our key-based fingerprints are:
  - ***Unique per user***, and cryptographically independent of the passwords themselves, allowing for different passwords to be used per fingerprint.
  - ***Consistent across all servers***, where the user registers using the same password (pool) i.e. with the corresponding fingerprint, while each server maintains its own secret verifier key, that the client may commit to.

  Server Pubkey Fingerprint & Public Password versus Key-based Password Fingerprint: This contrasts for example, with the 'public password' approach outlined in *Public-key cryptography and password protocols* by Halevi and Krawczyk [HK99], which necessitates validation of distinct fingerprints *for each server*, even if the same password is being reused. A familiar example of this is seen in the use of public keys for SSH servers, where each server has its own unique public key fingerprint,

[5] to our best of knowledge.

which the user has to validate upon first-time SSH sessions with each server, with the difference, that a SSH server public key fingerprint is the same for all users, but a "public password" would constitute an individual fingerprint per user per server. Our protocol, in contrast, allows for the use of a consistent fingerprint across multiple servers, even when the same password is used; while still providing each server with its own corresponding verifier i.e. 'public key', enhancing usability without compromising security.

- **Commitment via Registration Authorities**: As described in more detail in the upcoming subsection, with our protocol, the user commits to his fingerprint – and optionally to each of his server-associated verifier key – through trusted registration authorities, ensuring a secure and verified user identity. As each server maintains its own secret verifier key, which it may require the user to commit to, this mechanism not only enhances security but also provides the technical requirements and lays the conceptual groundwork for legal and regulatory non-repudiation.
- **Use of External Credential Hardening Service**: To generate the corresponding secret signing keys from low-entropy and easily guessable password in a secure manner, the client needs to utilize an external credential hardening service. This approach contributes to the robustness of the password authentication process by denying the authentication server and attackers that may compromise it to offline-bruteforce user password, thus providing an additional layer of security. In 5 we undertake a comparison of viable approaches, provide arguments for our preference on Oblivious Pseudo Random Function (OPRF) services, and propose both a client and server prototype implementation.

These objectives are designed to address the inherent weaknesses in current password authentication systems and pave the way for a more secure, auditable, and non-repudiable method of online authentication. The proposed protocol seeks not only to elevate the security standards but also to align with real-world applications and regulatory requirements, as detailed in the subsequent ideal solution scenario.

1.3.1 Ideal Solution Scenario

In an optimal *non-repudiable* password authentication system, the protocol must not only provide robust cryptographic security but also facilitate real-world applications, particularly in resolving disputes over online transactions; from mere login activities, over communication and authorship, up to financial transactions and legally binding contracts. This scenario involves:

- **Auditability and Forensicability**: The protocol should enable detailed logging and tracing of authentication events, making it possible to conduct thorough forensic investigations in the event of a security breach or dispute. This capability is

pivotal for determining the authenticity and integrity of each transaction, ensuring that all parties involved in an online transaction can be held accountable for their actions. The key to implementing this capability rests in the adequate application and implementation of digital signatures and commitment to corresponding public keys i.e. their respective equivalent (see next item). As outlined in 1.5, this challenge and viable approaches will be discussed in the upcoming chapters, and implemented in 7.

- **User Commitment and Trusted Registration Authorities**: A significant aspect of the protocol involves the role of trusted registration authorities, where users commit to their unique digital identities — represented as public keys or fingerprints. This commitment is a binding agreement, establishing the user's identity and intent in a legally and regulatory compliant manner. It forms the basis for non-repudiation, ensuring that users cannot deny their participation in a transaction, for which they have authenticated i.e. which they have *signed* with their respective private key.
- **Legal and Regulatory Non-repudiation**: The integration of user commitment with trusted authorities provides a framework for legal and regulatory non-repudiation. It holds individuals or entities liable for their actions in digital transactions, thus fostering a secure and trustworthy digital environment.

Conclusion. In an ideal solution scenario, the design of a non-repudiable password authentication system must intricately balance the technical aspects of secure authentication with the practical implications of its implementation in real-world contexts. This involves ensuring absolute immunity to online guessing attacks, enabling detailed auditability and forensic investigations for dispute resolution, and incorporating user commitment through trusted registration authorities. Such a system should not only offer strong cryptographic foundations but also cater to legal and regulatory non-repudiation requirements. Moreover, the protocol should integrate seamlessly into existing infrastructures, striking a harmonious balance between robust security and user convenience, thereby fostering a secure, trustworthy, and legally compliant digital environment.

1.4 Contextual aspects to consider

In this section, we aim to provide contextual aspects of forensics, cyber crimes, trust, and user awareness, that may be consider, when envisioning novel authentication schemes for human users.

1.4.1 Enhancing Forensic Investigation

The 'forensicability' and 'auditability' of an authentication system are pivotal in establishing its credibility. These features enable detailed analysis and tracking of authentication

processes, contributing significantly to digital forensic investigations and bolstering overall system security.

1.4.2 Criminal Psychology and Advanced Trust

Understanding the mindset of cybercriminals is crucial in designing foolproof security systems. Achieving an advanced level of trust in authentication technologies requires making online guessing attacks an impossibility. This paradigm shift is essential for the widespread acceptance and adoption of new, more secure authentication methods.

1.4.3 User Education and Awareness

The efficacy of any authentication system partially hinges on user understanding and awareness. Educating users about the importance and correct validation of their authentication fingerprints is vital. This understanding is a cornerstone of maintaining the integrity and security of the authentication process.

1.5 Thesis Structure

This thesis is structured as follows:

- **Chapter II: Password-based Authentication, the Web, and Non-repudiation** discusses the historical and current state of password-based authentication methods on the web, including their vulnerabilities and limitations. This chapter also explores the concept of non-repudiation in the context of online authentication.
- **Chapter III: Password-Authentication Key Exchange (PAKE) Protocols and Non-repudiation** provides an overview of PAKE protocols, their evolution, and their role in enhancing online security. The chapter also critically analyzes the limitations of existing protocols in achieving non-repudiation.
- **Chapter IV: Security Analysis and Protocol Re-design: Towards (Explicit-)Mutually Authenticated A²PAKE** delves into various attack scenarios, security requirements, and goals. This chapter also outlines the design objectives of the proposed authentication protocol.
- **Chapter V: Oblivious Pseudo Random Functions (OPRF) and OPRF Services** discusses the implementation and challenges of OPRFaaS, highlighting its significance in password hardening and enhancing security against low-entropy challenges.

- **Chapter VI: Bi-Functionality, Post-Generation Independence, Generator-Base-Dependent Reproducibility: Unlocking RSA and the Unique Properties of its Key-pairs** focuses on the unique properties of certain asymmetric cryptosystems, such as RSA and bilinear pairings, and their application in the proposed authentication protocol.
- **Chapter VII: Design and Implementation of a Non-Repudiable Password Authentication Protocol** presents the core contribution of this thesis, detailing the novel protocol design and its mechanisms for ensuring non-repudiation and mutual authentication.
- **Chapter VIII: Conclusion** summarizes the thesis, highlighting the key contributions, the implications for IT security and digital forensics, and potential areas for future research.

Each chapter aims to build upon the knowledge and context established in the preceding sections, culminating in a comprehensive understanding of the challenges, solutions, and innovations presented in this thesis.

1.6 Challenges

There has been and currently still is a lot of research going on towards Strong- and Auditable aPAKE protocols and there are many prototypes and experimental libraries and public repositories available, however, assessing and choosing the most promising and applicable ones might be a challenging task. The novel and hence most unpredictable challenge will be the assessment of the applicability of appropriate aPAKE protocols to achieve the non-repudiation of password-based user authentication of a given user at a given time on a given service) and if necessary: protocol adaption & prototyping.

1.7 Quotes

We conclude this chapter with some thematically related quotes of various sources:

- *Most importantly, this paper illustrates, once again, the subtlety associated with designing password-based protocols.* – from the ”M-PAKE”paper[STW95]
- *Since authentication and key-exchange protocols which do not allow passive dictionary attacks against the user's password are possible—Encrypted Key Exchange [BM92, BM94] and its variants [Jab96, Jab97, Wu98], IPSec—it seems imprudent for Microsoft to continue to rely on the security of passwords.* – from the ”Cryptanalysis of [MS-CHAPv2]”paper [SMW99])

- *Remarkably, our analysis shows optimal resistance to off-line password guessing attacks under the choice of suitable public key encryption functions. In addition to user authentication, we describe ways to enhance these protocols to provide two-way authentication, authenticated key exchange, defense against server's compromise, and user anonymity.* – from the "PKC and password protocols"paper [HK99]
- *Any secure password authentication protocol requires some sort of secret key exchange.* – from the "PKC and password protocols: Multi-user"paper [Boy99]
- *Although good password selection should be encouraged, it remains true that, in practice, user-selected passwords are weak. This is to be expected since high-entropy passwords are difficult (if not impossible) to remember. It is preferable to recognize this and design protocols which remain secure despite this limitation.* – from J. Katz' PhD thesis [GK02]
- *Although PAKE-based web authentication offers some clear advantages over conventional in-page password authentication, it remains unclear whether these advantages are sufficient to overcome the hurdles it faces.* – from the Ïs it too late for PAKE?"paper [Eng+09]
- *The first rule of PAKE is: nobody ever wants to talk about PAKE. The second rule of PAKE is that this is a shame, because [it] is actually one of the most useful technologies that (almost) never gets used. It should be deployed everywhere, and yet it isn't.* – from blog.cryptographyengineering.com [Gre18]
- *The challenge of password-based schemes is obtaining strong security based on weak secrets.* – from the "PAPKE"paper [Bra+19]

2 Password-based Authentication, the Web, and Non-repudiation

> **Etymology[1]:** *pass* + *word*; the sentry-passing sense predates the cryptographic sense and was its inspiration.

2.1 Introduction

History. The domain of password-based authentication, a cornerstone in the edifice of digital security, presents a rich tapestry of historical evolution, technological advancements, and an unceasing quest for balance between *ease of use* and *impenetrable security.* This journey, originating from the most rudimentary impromptu-protection for text-terminals of the early computing era, to the sophisticated, multi-layered authentication mechanisms of today, encapsulates a relentless pursuit of safeguarding *digital identities* against an ever-evolving landscape of cyber threats.

Evolution. From its inception, the password has been a fundamental element in authenticating users. Its *simplicity*, rooted in the human ability to *remember* and use secret information, made it an intuitive choice for early computer systems. As technology advanced, the password evolved, adapting to new challenges and changing user environments. This evolution was not merely technological but also *conceptual*, reflecting a growing awareness of digital vulnerabilities and the complexities of cyber security.

Big Picture. This chapter embarks on a journey to explore the historical context and *significance* of password-based authentication. It delves into the traditional mechanisms and their *limitations*, unearthing the intrinsic vulnerabilities that have spurred the development of more robust authentication methods. It also examines the impact of emerging technologies and shifting paradigms in web authentication, highlighting the critical role of passwords in the broader narrative of digital security and user sovereignty. Particularly, it reflects on the 'lessons learned' from the transition from *symmetric* to *asymmetric* cryptography, the relocation of trusted *authentication logic* to the client/browser or operating system, and the imperative to preserve the user's *freedom of choice* between proof-of-knowledge and proof-of-possession in authentication methods.

Outlook. Our discourse navigates through the intricacies of multi-factor authentication, the subtleties of public key infrastructure, and the revolutionary potential of FIDO2 and

[1] en.wiktionary.org/wiki/password

WebAuthn. As we traverse this landscape, we underscore the enduring relevance of passwords, their entanglement with user autonomy, and the impetus for their reinvention in the face of modern cyber challenges.

2.2 Legacy Mechanisms of Password Authentication and Their Limitations

> **Etymology**[2]: *password*; "secret word appointed as a sign to distinguish friend from foe" (1798)

2.2.1 Foundational Password Mechanisms and Inherent Limitations

Early Beginnings. The inception of password-based authentication can be traced back to the dawn of computing. Initial password systems, employed in early mainframe computers and text-based user interfaces, were simplistic yet revolutionary. They introduced the concept of user-specific secrets (*passwords*) as a means to control access to resources.

Operational Simplicity. These early systems implemented passwords as static strings, checked against user entries for authentication. This simplicity, however, was a double-edged sword. While easy to implement and use, it lacked *robustness* against even basic attack vectors. Vulnerabilities to brute force attacks, where attackers try multiple combinations, and to social engineering techniques were inherent in these systems.

Evolution Under Threat. As computing systems evolved and interconnected, the threats magnified. The static nature of early passwords made them susceptible to *dictionary attacks*, where common passwords are systematically tested. Furthermore, the advent of the internet and the proliferation of *online services* increased the visibility and attractiveness of password-based systems to malicious actors.

The PKI Reliance in PLAIN-over-TLS. The reliance on Public Key Infrastructure (PKI) for securing passwords transmitted over TLS—i.e. *Transport*, not *Session*[3] Layer Security—protocols became a notable point of discussion in cryptographic circles[4]. This approach, while elevating the security level, introduced dependencies on the *integrity* of the PKI system, as well as the *legitimacy* of certain PKI-authenticated entities amongst many[5], underscoring the need for stronger password authentication mechanisms beyond just transmission security.

Inherent Limitations and Push for Innovation. The fundamental limitations of early password mechanisms – primarily their vulnerability to various forms of attacks and their

[2] etymonline.com/word/password
[3] note, that neither SSL stood for 'Secure *Session* Layer'
[4] "Did he really just say 'PKI'?!"
[5] note, that 'every' phishing site is TLS-enabled, and hence public-PKI-authenticated.

reliance on user memory – catalyzed the push towards more innovative and secure methods. This urgency laid the groundwork for the development of *multi-factor* authentication, *asymmetric* cryptography—as in *a*PAKE—-protocols, and other advancements that sought to address the inherent shortcomings of these legacy systems.

2.2.2 Security Aspects, Threats, and Attacks

Vulnerability Landscape. Traditional password authentication systems, while foundational in the realm of digital security, have long been plagued by an array of vulnerabilities. The simplicity of these systems, often their most significant advantage, has paradoxically rendered them susceptible to a spectrum of cyber threats.

Brute Force and Dictionary Attacks. Two of the most prevalent attack vectors are *brute force* and *dictionary* attacks. Brute force attacks involve systematically trying every possible combination of characters until the correct password is discovered. In contrast, dictionary attacks exploit the tendency of users to choose common, easily *guessable* passwords, testing combinations from curated lists of widely used passwords.

Social Engineering and Phishing. Beyond technical exploits, social engineering poses a significant threat to password security. These attacks manipulate users into revealing their passwords, often through deceptive means. Phishing attacks, a form of social engineering, trick users into *entering* their credentials *into*[6] fraudulent websites, thereby compromising their digital identity.

Data Breaches and Exposure. The rise in large-scale data breaches has further amplified the risks associated with password-based systems. Compromised databases often contain vast amounts of user credentials, which, if not adequately protected, can lead to widespread unauthorized access and identity theft.

Limitations of Password-only Systems. These vulnerabilities underscore the inherent limitations of password-only authentication systems. While passwords serve as a key to digital identity, their effectiveness is diminished in isolation. The lack of additional verification layers in these systems makes them an attractive target for attackers, seeking to exploit the weakest link in security – the *human factor*.

Impetus for Advanced Protocols. The persistent threat landscape surrounding traditional password mechanisms has been a driving force behind the development of more secure and sophisticated authentication protocols. This necessity has catalyzed innovations in multi-factor authentication and asymmetric cryptography, laying the foundation for protocols like '*Mutually Authenticated* Auditable aPAKE,' which address these security gaps and offer enhanced protection for digital identities.

[6] because—*where* is the 'authentication logic' in PLAIN-over-TLS?!

2.2.3 Enhancements: Multi-factor Authentication (MFA)

Beyond Passwords: The Advent of MFA. The recognition of inherent vulnerabilities in password-only systems led to the development of Multi-factor Authentication (MFA) as a pivotal enhancement. MFA introduces additional authentication factors, significantly fortifying security by requiring more than just the *knowledge*—or guessing—of a password.

Layers of Defense. MFA typically incorporates a combination of 'something you know' (like a password), 'something you have' (like a security token), and 'something you are' (like a biometric feature). This multi-layered approach drastically reduces the risk of unauthorized access, as it complicates attacks that rely solely on compromising passwords.

Misconceptions and Reality. Despite its effectiveness against brute-force and dictionary attacks, a common misconception prevails that MFA provides comprehensive protection against all forms of cyber threats, including *phishing*. In reality, MFA can be vulnerable in phishing scenarios, especially where attackers can relay MFA prompts in real-time through a compromised or fake website.

Phishing: The Achilles' Heel of MFA. The phishing attack vector, particularly in the context of sophisticated, *TLS-enabled* phishing sites, presents a notable challenge to MFA. Attackers can deceive users into entering their credentials and MFA responses on fraudulent sites, which are then relayed to the legitimate service. This method effectively bypasses MFA, undermining its perceived invulnerability to phishing attacks.

Implications for User Trust. This limitation of MFA, especially against phishing, has significant implications for user trust and the perceived security of authentication systems. It underscores the need for more advanced authentication methods that address not just the risk of password compromise but also the nuanced vulnerabilities presented by phishing and similar attacks.

2.2.4 PLAIN-over-TLS and PKI Reliance

Securing Passwords in Transit: PLAIN-over-TLS. The widespread adoption of PLAIN-over-TLS marked a significant advancement in protecting passwords during transmission over the internet. By encrypting the communication channel, TLS ensures that passwords, even if sent in plain text, remain *confidential* and secure from eavesdropping and man-in-the-middle attacks.

Dependence on Public Key Infrastructure. The effectiveness of PLAIN-over-TLS heavily relies on the robustness of Public Key Infrastructure (PKI). PKI provides the necessary framework for *digital certificates* and public-key encryption, establishing a chain of trust. However, this reliance introduces its own set of complexities and vulnerabilities, including the security of certificate authorities and potential weaknesses in the certificate issuance process.

Phishing: A Persistent Threat. Despite the security enhancements offered by TLS, it does not inherently protect against phishing attacks. Sophisticated phishing sites, often TLS-enabled themselves, can still deceive users into *submitting* their credentials. These sites exploit the trust users place in TLS-'secured' connections, illustrating a critical gap in this security model.

MFA's Limitation in TLS Context. This vulnerability also extends to scenarios where Multi-factor Authentication (MFA) is employed. In phishing attacks, attackers can relay MFA prompts along with password requests, thereby circumventing the additional security layer provided by MFA. This demonstrates how MFA, while effective against certain threats, fails to address the more nuanced risks posed by TLS-enabled phishing sites.

Rethinking Authentication Security. The limitations of PLAIN-over-TLS and the reliance on PKI underscore the need for more sophisticated authentication solutions. This necessity becomes particularly evident in the face of advanced phishing attacks, highlighting the importance of developing authentication protocols that are resilient not only to traditional threats but also to more subtle and sophisticated cyber-attacks.

2.2.5 Password Managers: Bridging the Knowledge Gap

The Advent of Password Managers. As digital ecosystems expanded and the number of online accounts per user surged, the limitations of *human memory* confronted the security imperative for *unique* and *complex* passwords. Password managers emerged as a practical solution to this conundrum, offering a secure way to *store* and manage an array of passwords.

Enhancing Password Security. These tools not only store passwords but often include features to strengthen—i.e. generate strong, randomized—passwords themselves, thus significantly reducing the risk associated with weak or *reused* passwords. By automating the process of creating and retrieving complex passwords, password managers alleviate the burden on users to remember each password, while enhancing overall security.

Centralization of Risk. However, the reliance on password managers introduces a new set of risks. They centralize the user's passwords into a *single repository*—and hence also 'single point of failure'—making them a lucrative target for cyber-attacks. The security of a password manager thus becomes critically important, as a breach could compromise all stored credentials.

User Convenience vs. Security Trade-off. While password managers offer a convenient way to handle multiple passwords, they also create a single point of failure. This underscores the need for robust security practices, such as the use of strong *master passwords* and ideally additional *MFA* to protect the password manager itself.

A Stepping Stone to Future Authentication Methods. As we progress towards more advanced authentication methods, password managers, while pivotal for password security, are though often thought of as a 'transitional' tool. They address some of the inherent flaws of traditional passwords while highlighting the ongoing challenges in balancing user convenience with robust security measures.

2.3 Uncertain Future: Password-based and Alternate Web Authentication

2.3.1 Smartcards: A Historical Step Beyond Passwords

Historical Context of Smartcards. The advent of smartcards marked a significant departure from the conventional password-based authentication systems. As one of the early forms of physical token-based security, smartcards introduced a tangible element in the authentication process, embodying the concept of 'something you have.'

From Passwords to Physical Tokens. Smartcards operate on the principle of *possession*-based security. They typically store *cryptographic keys* and personal identifiers, which are used to authenticate the bearer of the card. This shift from knowledge-based (passwords) to possession-based (smartcards) mechanisms represented a pivotal moment in the history of digital security.

Advantages and Adoption. Smartcards brought several advantages, such as the ability to store and use cryptographic keys securely, making them ideal for various applications, including access control in enterprise and government settings. Their physical nature offered a level of security that purely knowledge-based systems couldn't match.

Challenges and Limitations. Despite their security benefits, the widespread adoption of smartcards was hindered by factors like the cost of deployment, infrastructure requirements, and user convenience. The need for specialized readers and the management of physical tokens added layers of complexity to their implementation.

Role in Modern Authentication. Today, smartcards are seen as a crucial step in the evolution of authentication methods. They laid the groundwork for further advancements in possession-based security, paving the way for more sophisticated technologies like (mobile) client certificates and biometric authentication.

2.3.2 Transitioning from Legacy to Modern Authentication: Challenges and Innovations

Evolving Cybersecurity Landscape. The shift from legacy authentication methods, such as passwords and smartcards, to modern techniques is primarily driven by an evolving cybersecurity landscape. With the rise of sophisticated cyber threats, traditional methods

have shown vulnerabilities, prompting a need for more secure and resilient authentication mechanisms.

User Experience and Security: A Delicate Balance. As technology advanced, the challenge wasn't just about enhancing security but also about improving user experience. The goal has been to develop authentication methods that are both secure *and* user-friendly — a task 'easier said than done.' *Modern* authentication solutions strive to find this balance, offering ease of use without compromising security.

Innovative Approaches to Authentication. In response to these challenges, a variety of innovative authentication methods have emerged. These include *biometric* authentication, reaching from fingerprint-sensors on notebooks to front-camera face-recognition in smartphones; *mobile*-based authentication, from software i.e. app-based solutions like key-vaults, time-based 2FA-generators, camera-based i.e. QR-code scanners, and push- or pull-notification confirmation, over USB-tokens and NFC readers, to hardware-embedded client-certificate- and key-stores in mobile devices, offering convenience and portability; and—as previously discussed, the various combinations of possible selections from the aforementioned categories—*MFA*, integrating multiple layers of security. Each method and combination brings its strengths and addresses specific aspects of the security and usability conundrum.

The Role of Emerging Technologies. Technologies such as *machine learning* and *artificial intelligence* are also being integrated into authentication systems. They enhance security by detecting *unusual patterns* of access and can provide adaptive authentication mechanisms based on context and risk assessment.

Future Directions in Authentication. Looking ahead, the future of authentication is likely to be a blend of various methods, each serving different security needs and user preferences. The continuous innovation in this field aims to stay ahead of cyber threats while catering to the evolving demands of a digital society.

2.3.3 Passwordless: From Virtual Smartcards over FIDO2 & WebAuthn to Passkeys

A Journey towards Password Independence. The quest for a viable *password-less* option for user authentication has been a pivotal theme in the evolution of digital security. This journey commenced with the advent of *virtual smartcards*, pioneering the concept of software-based secure identity storage, leveraging the Trusted Platform Module (TPM) to emulate the functionality of physical smartcards. Virtual smartcards, by offering secure, two-factor authentication without the need for a physical token, marked a significant step towards more sophisticated passwordless solutions.

FIDO2 and WebAuthn: Setting New Standards. The introduction of FIDO2 and WebAuthn technologies represents a watershed moment in this journey. These technologies

have redefined the—now known under this established term—*passwordless* authentication by enabling users to authenticate to online services securely and conveniently without a password. FIDO2, a key advancement in this arena, allows users to leverage local biometric authentication or external security keys for website access. WebAuthn, an API for accessing public key credentials, works seamlessly with FIDO2 to enhance user experience and security.

Passkeys: The Next Frontier. Building on these advancements, Passkeys emerge as the latest innovation in passwordless authentication. They represent a novel approach, offering a seamless and secure method for users to access services across different devices and platforms. Passkeys utilize cryptographic techniques to create unique credentials *for each service*, eliminating the need for traditional passwords altogether. This cutting-edge technology symbolizes the ongoing commitment to advancing passwordless authentication, making it more accessible and user-friendly.

Conclusion. The evolution from virtual smartcards to FIDO2, WebAuthn, and now Passkeys exemplifies the dynamic nature of cybersecurity. Each step in this progression not only strengthens security but also enhances user autonomy, marking significant strides in our journey towards a truly passwordless future.

2.4 The Three Security Essences of Passwordless

Introducing the Core Pillars. As the digital world—driven by big tech companies and consumer electronics manufacturers—gravitates towards Passwordless authentication mechanisms, it becomes imperative to dissect and understand the foundational principles that make these technologies not just viable, but exemplary in enhancing web security. In this exploration, we delve into three core 'security essences' that form the inconspicuous backbone of Passwordless systems—cryptography, authentication logic, and proof of identity—that sets apart from legacy password authentication; and its weaknesses. Each essence represents a crucial aspect of the Passwordless paradigm, collectively revolutionizing the landscape of web authentication by addressing long-standing vulnerabilities and setting new standards in digital security and user experience.

2.4.1 Essence I - Cryptography: Asymmetric vs. Symmetric

Cryptography Foundations. In the realm of web authentication, cryptography serves as the bedrock for securing digital identities. The distinction between asymmetric and symmetric cryptography is pivotal in understanding the evolution of Passwordless technologies.

Asymmetric Cryptography in Passwordless. Passwordless authentication frameworks like FIDO2, WebAuthn, and Passkeys predominantly employ asymmetric cryptography. This approach utilizes a pair of keys - a public key that is shared openly and a private key that

remains secret. The public key, often incorporated into a user's digital certificate, is used for verifying digital signatures or encrypting messages to the user, while the private key is used for creating these signatures or decrypting received messages. This cryptographic model enhances security by ensuring that even if a public key is intercepted, without the corresponding private key, an attacker cannot impersonate the user or decrypt confidential information.

Digital Signatures and Non-Repudiation. A critical aspect of asymmetric cryptography is its capability to facilitate non-repudiation through digital signatures. In Passwordless systems, user actions, such as transactions or authentication requests, can be digitally signed using the private key. These signatures, verifiable by anyone with access to the public key, serve as a cryptographic proof that the transaction or request originated from the user, thereby eliminating any deniability. This attribute of asymmetric cryptography aligns well with the legal and regulatory frameworks that demand non-repudiation in digital interactions.

Symmetric Cryptography: The Traditional Approach. In contrast, symmetric cryptography, which uses the same key for both encryption and decryption, has been the cornerstone of traditional password authentication mechanisms. While effective in certain scenarios, symmetric cryptography lacks the robustness of asymmetric techniques in the context of modern web threats. The shared key model of symmetric cryptography can be a vulnerability, especially if the key is exposed during transmission or storage, or can be offline-brute forced based on protocol transcripts.

Conclusion. The shift from symmetric to asymmetric cryptography in Passwordless approaches marks a significant advancement in web authentication. By leveraging the strengths of asymmetric cryptography, Passwordless technologies not only enhance security but also address critical requirements like non-repudiation, making them a formidable choice for modern web security challenges.

2.4.2 Essence II - Authentication Logic: Client-Server vs. Server-only

Shifting the Paradigm. Traditional web authentication mechanisms often hinged on server-driven logic, predominantly hosted within the *webpage source code*—received by client i.e. web-browser—*from the server*. This approach, while functional, harbored inherent risks, particularly in scenarios where server integrity could be compromised. Passwordless authentication, as embodied in systems like FIDO2 and WebAuthn, marks a significant paradigm shift. It entrusts a substantial portion of the authentication logic to the client's browser or operating system. This shift not only fortifies the process against server-side vulnerabilities but also paves the way for enhanced mutual authentication.

Enhanced Security and Trust. By embedding the authentication logic within the client's domain, Passwordless technologies mitigate risks associated with phishing and man-in-the-

middle attacks. This model ensures that even if a phishing website mimics a legitimate one, it cannot access or manipulate the authentication process governed by the client's browser or OS. This advancement not only heightens security but also fosters a sense of trust among users, aware that their authentication process is securely managed by their local device.

Reimagining User Authentication. The implications of this transition are profound. It represents more than just a technical upgrade; it signifies a reimagining of how user authentication is conceptualized and executed. In the Passwordless era, the user is not just a passive participant in the authentication process; they are an active player, with *their device* playing a crucial role in securing *their digital identity*. This approach aligns perfectly with the modern user's expectations of security, convenience, and control over their digital interactions.

2.4.3 Essence III - Proof of Identity: Ownership vs. Knowledge

The Traditional Dichotomy. In the realm of authentication, the previously discussed dichotomy between ownership (proof-of-possession) and knowledge (proof-of-identity) has long been a subject of debate. The Passwordless initiative, around FIDO2 and WebAuthn, has been leaning towards *ownership-based* models from the earliest stages and onward, emphasizing hardware-based asymmetric keys as primary identifiers, locked by biometric user access control, as their idealised solution.

Revisiting Knowledge-Based Authentication. While Passwordless solutions offer robust protection, they shift the focus away from traditional knowledge-based methods like passwords. Our thesis aims to explore how the security advancements of Passwordless can be adapted to enhance password-based systems, maintaining the simplicity and user autonomy associated with proof-of-knowledge, while integrating the security benefits of asymmetric cryptography and client-driven authentication logic.

Balancing Security with User Autonomy. The essence of our approach lies in striking a balance between the security provided by ownership-based models and the user autonomy inherent in knowledge-based systems. By reimagining password authentication with advanced cryptographic principles and client-side authentication logic, we aim to elevate password-based systems to new levels of security, without sacrificing the fundamental principle of user control inherent in proof-of-knowledge.

2.5 Lessons Learned from Passwordless: Asymmetric Cryptography & Client-side Authentication Logic

Introduction. Reflecting on the advancements and principles of Passwordless authentication, we gain valuable insights into the future of secure web authentication. This chapter

synthesizes the key lessons learned from Passwordless technologies, emphasizing the potential of implementing these concepts within a Proof-of-Knowledge framework.

Embracing Asymmetric Cryptography. The shift towards asymmetric cryptography in Passwordless systems underscores the need for enhanced security in authentication protocols. Implementing similar cryptographic principles in a *password-based* framework can significantly increase security, offering robust protection against a range of cyber threats while maintaining the simplicity and familiarity of passwords.

Client-side Authentication Logic. Passwordless approaches advocate for harmonizing i.e. standardizing and sustaining trusted authentication logic in the client's browser or OS, rather than relying on server-provided mobile—i.e. webpage source—code — or the externalization of the authentication mechanism to trusted third parties. This shift towards client-side self-supply of authentication logic, enhances security against phishing and MITM attacks. Integrating this concept into password-based authentication could strengthen the protocol against such attacks, providing a more secure authentication experience for users.

Navigating Proof-of-Identity Paradigms. Passwordless systems predominantly utilize Proof-of-Possession, contrasting with traditional password-based Proof-of-Knowledge methods. Our analysis reveals that while Proof-of-Possession offers certain security advantages, it may not always align with user autonomy and digital sovereignty. This insight guides us towards developing password-based authentication methods that preserve user control and choice, without compromising on security.

Non-repudiation: An Unexpected Benefit. The exploration of Passwordless technologies brings to light the significance of *non-repudiation* in digital authentication. While traditionally not a focus of neither legacy password-based systems, nor in the research field of Asymmetric Password-Authenticated Key Exchange (aPAKE) schemes, our work suggests the potential for integrating non-repudiation features into password authentication, thus enhancing accountability and trust in digital transactions.

Conclusion. The lessons learned from Passwordless authentication offer a roadmap for enhancing password-based systems. By incorporating aspects of asymmetric cryptography, client-side authentication logic, and non-repudiation, we can envision a future where password authentication is not only secure but also empowers users with greater control over their digital identities.

3 Password-Authentication Key Exchange (PAKE) Protocols and Non-repudiation

3.1 Introduction

Exploration. Authentication stands as both a cornerstone of digital security, and as a sentinel in the safeguarding of the digital identities of humans, ensuring that the claimed identity of a communicating party is genuine. Among the repertoire of cryptographic authentication schemes, *Password-Authenticated Key Exchange (PAKE)* protocols occupy an exclusive niche. They have emerged as an influential and dynamic area of research, striving to bridge the gap between the convenience of passwords and the rigor of cryptographic security. This exploration delves into PAKE not just as an established construct but as a realm of endless possibilities for securing and *enabling* digital identities.

The More Essences. The following sections of this chapter distill—as previously those of Passwordless (in 2.4)—the 'essences' of PAKE protocols, charting their inception with (RSA-)EKE and the alleged necessity to depart from RSA due to the supposed offline dictionary attacks it enabled—a pertinent challenge aptly addressed in the influential work of Halevi and Krawczyk ([**HK98**]).

The Blind Spot. As our discourse is quickly escalating to include contemporary advancements and building blocks, we are also taking into sight the inconspicuous beacon of accountability that is leading our exploration towards the phenomenon of *non-repudiation* in the cryptographic realm; a desirable feature that fortifies trust but has historically been a blind spot in traditional PAKE research.

Asymmetric Promise. A particular emphasis is placed on categorization, delineating the structural differences between balanced, augmented, and fully asymmetric approaches. The goal is to engage the reader with the premise that only truly asymmetric PAKE protocols—those not relying on mere augmentation—can facilitate non-repudiation alongside password-based authentication. Within this discourse, the *Auditable Asymmetric PAKE (A²PAKE)* [Fao+22] stands out, not merely as an academic contribution but as a practical incursion into the realm of authenticated key establishment and *non-repudiable sessions*.

This chapter aims to be concise yet comprehensive, ensuring a focus on the salient aspects that underpin the field's continuous development. It is not the length of the story that mat-

ters, but the lasting impact discerned through the impeccable merit of *truly* asymmetric PAKE protocols and their inconspicuous potential for real-world applications.

3.2 Challenges

Entropy. These protocols allow two or more parties to generate a shared secret, typically a high-entropy session key, based solely on the knowledge of a weak, i.e. low-entropy password. In doing so, PAKE protocols eliminate the need for a public key infrastructure (PKI) — as is the case for TLS; while still providing confidentiality and integrity over secure channels i.e. authenticated key exchange sessions, in the face of eavesdropping and active impersonation attacks.

Vulnerability and Mitigation. The ability and freedom of choice to authenticate using passwords—as a low-entropy secret getting-by on human memorability and ease of use—is prone to introducing certain vulnerabilities. Low entropy secrets, especially 'words,' are inherently susceptible to brute-force, and *dictionary* or *guessing* attacks, respectively. Mitigating these vulnerabilities by solving the underlying 'entropy-problem' can be considered one of the main challenges and most intensively studied research matters in the field of PAKE protocols.

Hubris. This thesis' exploration towards non-repudiation in PAKE schemes is mesmerized by an *intellectual ordeal* that transcends encrypting mere channels to fostering a landscape where trust, confidentiality, and accountability are inherently engineered into the protocols. In Chapter 1, our highlighting narrative on the need for 'evolutionary' steps in authentication, was meant to set the stage for PAKE protocols as a transformative answer to this call, glancing toward the unchartered horizon of *non-repudiation.*

3.3 A Brief History of PAKE Protocols

The quest to secure passwords against the backdrop of evolving cyber threats has yielded a rich tapestry of cryptographic innovation. In the early days, protocols were rudimentary, often merely concealing passwords behind a veil of secrecy that was too easily pierced.

3.3.1 Seminal Works on Password Authentication

Needham-Schroeder Protocol

Advent of secure key-exchange & sessions. The Needham-Schroeder protocol, proposed in 1978 [**NS78**], provided a foundational framework for secure communication, and can be viewed as one of the foundational works in the realm of secure authentication. While it primarily focused on *key exchange* through a trusted third party, its principles have

influenced many subsequent authentication protocols. As such, it has informed PAKE development efforts by underlining the importance of securely establishing *session keys*. Furthermore, it is often regarded as the main source of inspiration for symmetrically authenticated key exchange and secure session protocols, such as *Kerberos*.

Lamport's Hash

One of the earliest methods proposed for secure *password-based* authentication was *Lamport's hash* [Lam81]. This form of hash-based one-time password represented in 1981 can be accounted as one of the early strides towards the development of robust password authentication systems. By leveraging hash functions, this protocol aimed at the perennial issue of transmitting *confidential* information—or 'a secret', for simplicity—across potentially *insecure channels*, without compromising the secret itself. This was in a time when remotely transmitted network traffic was mostly unencrypted; and hence it was still common, to also send send secrets like *passwords* over the network—entirely unprotected—in plaintext, as it is the case for the—back then without encrypting alternatives to them—*Telnet*, and *Remote Shell* protocols; as *SSH* was to be first introduced in 1995.

3.3.2 PAKE: Historical Milestones and Highlights

The narrative of PAKE research is punctuated by significant milestones – each an intellectual leap toward the realization of secure—i.e. *secrecy*-preserving—password authentication.

The First PAKE Protocol: EKE

In Bellovin and Merritt's 1992 paper [BM92], EKE addressed the vulnerabilities of both password exposure during transmission, and offline brute force i.e. *dictionary attacks* from recorded transmission transcripts. It ushered in a suite of variants, applying the concept of *password-based encryption* to the challenging design of secure schemes for *key transport* and *key agreement* protocols; thus minting the fixed term for *PAKE* as the 'password-based key exchange'.

The First Augmented PAKE (aPAKE): A-EKE

Along came, proposed by the same authors shortly after, the concept for an *augmented* variant of PAKE [BM93]; which inspired the design of authentication schemes and protocols striving to defend against the dire consequences of *server compromise*, i.e. of the breached *confidentiality* of compromised *password files*. As seminal works showed, these

protocols were pivotal in initiating a design philosophy that mitigated the risk of, not only insecure networks, but also potential server-side breaches.

The (True) Arrival of Asymmetric PAKE (APAKE)

The introduction of 'Asymmetric' PAKE—abbreviated as *A*PAKE with capitalized 'a', in this thesis—may be considered controversial; or at least uncertain, and open to subjective interpretations. Even though often accredited to Jablon for his 1997 paper (see next subsubsection) introducing a Diffie-Hellman-based, and hence *asymmetric*, variant of (augmented) PAKE; our literature review suggests that indeed *Yaksha*, an asymmetric variant of Kerberos, proposed by Ganesan as early as 1995 [Gan95], deserves that credit. By building on a password-based variant of *mRSA* (see below), the Yaksha protocol marked a pivotal—and still pioneering—shift towards leveraging the unique properties of *key-pairs* from asymmetric cryptosystems. His paper wen as far as explicitly suggesting the feasibility of non-repudiation in his protocol; however, no further research and development effort was put towards this functionality, since the sole purpose of Yaksha was defined as an asymmetric variant—and prospective replacement—of Kerberos; and as such simply did not require a mechanism for non-repudiation.

The First Diffie-Hellman-based PAKE: DH-PAKE

As already mentioned in the prior subsubsection, in 1997 [Jab97] Jablon proposed a *password-based* variant of the *Diffie-Hellman* key exchange. As such, it was the first scheme for key exchange to *authenticate* the—otherwise 'anonymous'—*public keys* based on a corresponding *password*; which constitutes a viable and ever-since inspiring combination of both password-based authentication and asymmetric cryptography. However, the 'asymmetric' aspects of DH-PAKE are limited to the '*authenticated* key exchange,' as opposed to the broader 'user authentication, key exchange, and session' scheme from password-based asymmetric—*secret*, not 'public'—key-pairs of Yaksha, as outlined in the preceding subsubsection (3.3.2).

The Secure Remote Password (SRP)

SRP, a pioneering augmented password-authenticated key exchange (aPAKE) protocol, emerged as a *historical highlight* in the realm of aPAKEs. Initiated in 1997 and first proposed in the following year [Wu+98], SRP has undergone several evolutionary steps, from SRP-1 to the widely adopted SRP-6a, addressing various security concerns like the 'two-for-on' guessing and messaging ordering attacks. SRP's approach to securing password authentication involves *zero-knowledge proofs*, allowing users to authenticate without exposing their passwords, thereby countering interception and replay attacks.

Withstanding the Proof of Time. Its adoption in standards like ISO/IEC 11770-4:2006, RFC 5054, and IEEE Std 1363.2-2008, and the inclusion of variants like SRP-5, demonstrates SRP's significant impact on password-based web authentication methods, making it *the* most adopted out of a myriad—most of them to remain 'flashes in the pan'—of PAKEs.

The First Strong aPAKE (SaPAKE): OPAQUE

OPAQUE, an advanced augmented PAKE (aPAKE) protocol proposed in 2018 [JKX18], addresses a critical vulnerability present in traditional aPAKEs: susceptibility to *pre-computation attacks*. These attacks enable instantaneous password compromise upon server breach. OPAQUE fortifies aPAKE's core security by resisting such pre-computation strategies, making it robust even in scenarios of server compromise.
aPAKE-Advent of OPRF. Furthermore—or precisely for this purpose—it was the first[1] PAKE to utilize *Oblivious Pseudo-random Functions (OPRF)* as its cornerstone, offering forward secrecy and mutual-implicit authentication in a PKI-free environment. This protocol represents a significant leap in aPAKE design, providing an efficient, 2-3 message exchange framework that's secure in the Universally Composable (UC) setting and supports user-side hash iterations and password-based storage/retrieval of secrets. Its design highlights the importance of anticipating and neutralizing emerging threats in password-based authentication systems.

The First Auditable APAKE: A²PAKE

The evolution PAKE protocols witnessed a significant milestone with the advent of A²PAKE, published in 2020 and first proposed in 2022[Fao+22], which constitutes a *truly* Asymmetric PAKE incorporating *built-in auditability*. This paradigm shift represents a fusion of traditional password-based methods and the rich possibilities of asymmetric cryptography. A²PAKE, emerging as a response to the limitations of existing PAKE models, particularly in client-server environments, marks a significant milestone in PAKE development.
A Glimpse into the Future. It is here, that PAKE's path crosses with non-repudiation, a juncture that sensed the need for protocols to have built-in auditing capabilities that extend their utility to accountable digital interactions; and the non-repudiable signing of legally binding transactions. This brief historical overview of PAKE, now culminating in A2PAKE, paves the way for a deeper exploration of its technicalities, applications, and implications in the broader context of online user i.e. web authentication. The upcoming detailed section on A²PAKE3.5 will delve into these aspect.

Conclusion. The landscape of password authentication has seen continuous evolution, with researchers striving to address the inherent vulnerabilities of passwords. From Lam-

[1] to our best of knowledge.

port's hash to modern protocols like OPAQUE and A2PAKE, the journey reflects the community's relentless pursuit of security. As this thesis delves into a novel approach to non-repudiable password authentication using OPRFs and special properties of certain asymmetric cryptosystems (foremost RSA), understanding the milestones and challenges faced by preceding works provides invaluable context. As we traced the lineage of PAKE, we not only acknowledge the historical significance of these protocols but also lay the groundwork for discussions on their further development into schemes, protocols, and tools, equipped to handle not just confidentiality and integrity—by authenticating secure channels—but also enable the binding assertion of cryptographic non-repudiation.

3.4 Categorization of PAKE

In the realm of Password Authenticated Key Exchange (PAKE) protocols, a nuanced understanding and clear categorization are imperative for both theoretical analysis and practical implementation. This section delineates the key categories of PAKE protocols, namely balanced, augmented, asymmetric, and auditable, and explains their unique characteristics and applications.

3.4.1 Balanced PAKE

Balanced PAKE protocols represent the foundational approach in PAKE development. In these protocols, both parties, typically a client and a server, *contribute equally* to the authentication process. This balance ensures that neither party has an inherent advantage, making these protocols ideal for environments and use-cases in which the 'shared secret' is—though *confidential*—not of exclusive *secrecy*; and where *mutual trust* is a given, yet mutual suspicion is warranted. Balanced PAKE schemes are characterized by their *symmetric nature*, where both parties share the 'same level of knowledge' and power in the authentication process.

3.4.2 Augmented PAKE (aPAKE)

Augmented PAKE protocols, commonly denoted as aPAKE, offer an advanced level of security, particularly in *client-server* models where the server is considered more vulnerable to attacks or data breaches. These protocols are designed to safeguard against scenarios where server-side data might be compromised. In aPAKE schemes, the client i.e. user possesses a *password*, while the server holds a *verifier*or or a *derivative* of the password, but not the password itself. This design ensures that even if the server data is compromised, the attacker cannot *directly* obtain the user's password, thereby providing an additional layer of security.

3.4.3 Asymmetric PAKE (APAKE)

Asymmetric PAKE protocols, denoted in this thesis as *A*PAKE with capitalized 'a' in order to enable for a clear disambiguation from augmented PAKE, mark a significant evolution in PAKE technology by incorporating asymmetric cryptographic principles. Unlike balanced or augmented PAKEs, or 'asymmetrically' key-exchanging aPAKEs, APAKE schemes leverage asymmetric i.e. 'public' key cryptography, where the server and client utilize—either each one key of a shared pair, or each their own pair out of two—pairs of cryptographically entangled keys, in the authentication process; while neither of which key needs to be 'public' or even 'publicly verifiable' as in the PKI setting, at all. It just so happened that this concept, or 'cryptographic primitive' has been assigned a term coined for its most common application i.e. the 'public key' setting. This *truly* asymmetric approach not only enhances on security notions but also bears the up-until-recent mostly *disregarded* implementability of *non-repudiation* and *auditability*; which are desirable properties, and essential e.g. in legal and regulatory contexts. APAKE protocols represent the groundwork towards a more secure and accountable digital interaction paradigm. However, for an APAKE protocol to become actually non-repudiable and practically auditable, it happens to be necessarily designed specifically for this purpose.

3.4.4 Strong aPAKE

Exemplified by OPAQUE, which maximizes password confidentiality by employing OPRFs [JKX18], the thereby introduced notion of 'Strong aPAKE' emerges in response to the vulnerabilities of traditional aPAKE protocols against pre-computation attacks. These attacks are significant as they lead to instantaneous compromise of user passwords upon server breach, undermining the intended aPAKE security. Strong aPAKE protocols are designed to be immune to such attacks by not revealing any for of 'public salt' value, derivative or otherwise vulnerable variable; maintaining security integrity even if server data is compromised. This enhancement crucially protects individual user passwords from being rapidly deciphered, thereby fortifying the entire authentication process against advanced cyber threats.

3.4.5 Auditable APAKE (A²PAKE)

Auditable PAKE, which seemingly are also asymmetric (APAKE) *necessarily*, are a recent innovation in the PAKE landscape. These protocols are designed with built-in auditing capabilities, which decisively extends the functionality of traditional PAKE and aPAKE protocols by providing mechanisms, that previously were deemed as 'another domain' of cryptography and also not inherently relevant to the field of PAKE research and development. Hence, we are exploring this further, in a dedicated section for *the* A²PAKE paper 3.5.

Conclusion. The categorization of PAKE protocols into balanced, augmented, asymmetric, and auditable types provide a clear framework for understanding their respective strengths and applications. This distinction is crucial in the context of evolving web authentication needs, where the choice of a PAKE protocol can significantly impact the security, functionality, and user experience. As the digital landscape continues to evolve, so too will the development and refinement of PAKE protocols, each category playing a pivotal role in the future of secure online interactions.

3.5 A²PAKE: Auditable Asymmetric PAKE

The Evolution of PAKE Protocols. A²PAKE, as the first explicitly *Auditable* Asymmetric PAKE [Fao+22]., marks a significant advancement in the realm of password-authenticated key exchange protocols. It represents a novel integration of asymmetric cryptography with the traditional password-based authentication, bringing a unique perspective to PAKE protocols, especially in client-server settings.

Addressing Limitations. Traditional PAKE protocols, while effective in establishing secure communication channels using either shared or augmented passwords, often fell short in scenarios requiring non-repudiation and auditability. A²PAKE addresses these gaps by incorporating mechanisms that allow for the auditing of authentication events and the signing of transactions, thus enhancing accountability in digital interactions.

Built-in Auditing Capabilities. What sets A²PAKE apart is its built-in auditing feature. This enables the protocol to facilitate not only secure password-based authentication but also the ability to record and verify authentication events and transactions. This feature is crucial for scenarios where legal and regulatory compliance is paramount, such as in financial transactions or in digital contract signing.

Implications for Online Authentication. The introduction of A²PAKE signifies a shift towards more accountable and verifiable online interactions. It opens up possibilities for integrating PAKE protocols into systems where user actions need to be both authenticated and audited, extending beyond the traditional scope of PAKE applications.

Conclusion. A²PAKE is a testament to the evolving landscape of PAKE protocols, where the focus is not only on ensuring secure authentication but also on providing mechanisms for accountability and non-repudiation. Its introduction is a stepping stone towards developing more sophisticated and legally compliant authentication systems in the digital domain.

3.6 Building Blocks: mRSA, OPRF, Designated Verifier Signatures

Introduction: In the intricate world of password-based user- i.e. web-authentication, various cryptographic building blocks play pivotal roles. This section delves into the nuances of mRSA, OPRFs, and Designated Verifier Signatures. While each of these components offers unique attributes and capabilities, only some of them are currently contributing to the development of advanced and secure authentication protocols. Their potential *synergy*, when combined in a well-structured authentication scheme, may open new avenues for enhancing both security and user experience in *non-repudiable* digital interactions.

3.6.1 RSA and Mediated RSA (mRSA)

RSA: Early Trials and Error in PAKE

RSA Cryptography. RSA, a cornerstone in public-key cryptography, has been pivotal in secure communications. However, its direct use in password-authenticated key exchange (PAKE) protocols, like RSA-EKE by Bellovin and Merritt [BM92], has been discouraged due to vulnerabilities to offline dictionary attacks, as mentioned by the authors, and highlighted in early works on EKE variants, as by Halevi and Krawczyk [**HK98**].

mRSA: Strategic Modifications for PAKE

Mediating Strategy. mRSA, a variant of RSA, ingeniously alters the traditional RSA scheme, particularly in key generation. This modification, independently introduced by Boyd [**Boy89**] and Ganesan and Yacobi [**GY94**], involves splitting the RSA private key into two distinct portions. In systems like Yaksha, one part becomes the user's 'password,' while the other serves as the server's counterpart. This division of the private key allows blending public key infrastructure with Kerberos-like systems, offering enhanced security with minimal protocol changes [Gan95]. Furthermore, Ganesan mentioned, as early as 1995 in his work, that Yaksha can potentially enable non-repudiation and 'password-based signatures' leveraging its built-in cryptographic mechanisms. However, this capability would be limited to the realm of the enterprise i.e. Kerberos domain or forest of domain trusts, in which Yaksha and its non-repudiation feature were to be set-up and configured, specifically for the corresponding set of domain users.

Implications in PAKE: The introduction of mRSA addressed the limitations of using standard RSA in PAKE scenarios, providing a more secure and adaptable framework. Its unique approach to key management, especially in the context of Yaksha, demonstrates how mRSA can bolster the security of password-based authentication systems.

3.6.2 Oblivious Pseudorandom Functions (OPRFs)

Introduction: Oblivious Pseudorandom Functions (OPRFs) have rapidly evolved from an inconspicuous cryptographic primitive, into both a robust building block of novel PAKE schemes, and a vital tool in the design and implementation of cryptographic protocols, thereby also finding diverse applications in privacy-preserving technologies. Originating from the concept of Pseudorandom Functions (PRFs), OPRFs enable a client-server interaction where the client computes a function on its input, without revealing the input or the function's output to the server.

Execution. In an OPRF setup, a client, holding an input x_c, interacts with a server possessing a secret key k_s. The client first 'blinds' the input using a randomly generated 'mask' $m_{c,r}$ and a suitable 'blinding function' b_m, transforming x_c it into $x'_c = b_{m_{c,r}}(x_c)$. This blinded input is then sent to the server. The server computes the function $F_{k_s}(x'_c)$ and sends the result y'_{c,k_s} back to the client. The client, in turn, 'unblinds' this output using the same mask $m_{c,r}$ as before and a corresponding 'unblinding function' b_m^{-1} to obtain the final result of $F_{k_s}(x_c)$ as $y_{c,k_s} = b^{-1}_{m_{c,r}}(y'_{c,k_s})$. This process ensures that the server learns neither the original input x_c nor the final output $F_{k_s}(x_c)$, while the client does not learn anything about F_{k_s}, thereby preserving both the secrecy of the client's input x_c and the server's secret key k_s.

Variants and Applications: OPRFs are categorized based on their underlying PRFs, such as Naor-Reingold and Hashed Diffie-Hellman. Their applications range from private set intersection (PSI) to PAKE. In the latter of which, the pivotal use of OPRFs is to remotely *and* securely evaluate a 'salted hash' of a password. [CHL22]

Significance in PAKE

In the context of PAKE protocols, OPRFs offer a robust mechanism to protect passwords during the authentication process. By using OPRFs, the client can securely (re)generate cryptographic materials, such as high-entropy seeds, based on 'low-entropy input'—such as the user's *password*—without revealing any information about that low-entropy input i.e. password to the OPRF server. This makes OPRFs particularly interesting for asymmetric mechanisms relying on *high-entropy secrets*.

3.6.3 Signatures: Schnorr and Designated Verifiers

In the cryptographic ecosystem, signatures play a critical role in authentication and non-repudiation. While Schnorr signatures are a common fixture in PAKE protocols, non-repudiation features are traditionally considered part of the 'independent domain' of *digital signatures*. This is an interesting contrast we are shedding a light on, in this subsection. Our special attention in this regard, is dedicated to the concept of 'Designated Verifier

Signatures' (DVS), which may bear significant potential for application in non-repudiable APAKE protocls.

Schnorr Signatures

Overview: Schnorr signatures, known for their simplicity and efficiency, have become a staple in cryptographic protocols, including PAKE. Their security relies on the intractability of discrete logarithm problems, making them an attractive choice for ensuring the integrity and authenticity of digital communications.
Role in PAKE: In the realm of PAKE protocols, Schnorr signatures are utilized to authenticate transactional elements, ensuring secure and verified exchanges. Their implementation within PAKE frameworks contributes significantly to the overall security and reliability of these protocols.

Designated Verifier Signatures

Overview: Designated Verifier Signatures (DVS) are a specialized form of digital signatures where verification is confined to a specific, pre-designated party. This exclusive verification process ensures a high level of confidentiality and security in digital communication [JSI96].
Distinctiveness and Uses: While not as prevalent as Schnorr Signatures in PAKEs, Designated Verifier Signatures provide a unique approach to secure communications, particularly useful in scenarios demanding robust protection of the identify and verifiability of the engaged signing and verifier keys of either side.
Potential in Authentication Schemes: In authentication contexts, Designated Verifier Signatures could be instrumental in ensuring that responses are confidential and verifiable solely by the intended recipient. This feature could offer an innovative layer of security and privacy in authentication systems, including Auditable APAKE protocols.

Conclusion: The exploration of these building blocks - mRSA, OPRF, and Designated Verifier Signatures - reveals a rich landscape of cryptographic techniques. Each element brings its unique strengths to the table, either already or prospectively contributing significantly to the evolution of password-based authentication. The potential amalgamation of these building blocks in a unified authentication protocol may harbor a promise to a robust, user-centric authentication and signing mechanism. Such an integration might not only fortify integrity, secrecy-preservation, and protection from various cyber threats, but also align with our vision of developing authentication systems that are both versatile and secure, and respectful of user autonomy and digital sovereignty.

3.7 Conclusion

This chapter embarked on an exploration of PAKE protocols, from their inception as simple password-guarded gateways to complex systems endowed with non-repudiation capabilities. The historical landscape has been marked by significant milestones, from the early advent of Lamport's hash through to the sophisticated protocols like A²PAKE, which portend a new era of accountable password-authenticated exchanges. These efforts set the stage for the contemporary landscape, where asymmetric paradigms reign supreme.

The PAKE field has been defined by a consistent effort to strike an elusive balance between ease of use and robust security. RSA-EKE, originally part of the suite of solutions proposed by Bellovin and Merritt [BM92], cast light on the vulnerability of public keys under password-based encryption, prompting a crucial pivot toward protocols less susceptible to offline dictionary attacks. This has led to a plethora of innovation, with PAKE protocols increasingly relying on discrete log-based constructions over RSA for their efficiency and security.

The maturation of PAKE protocols has seemingly culminated in the cautious incorporation of non-repudiation features, traditionally considered an independent domain of *digital signatures*. A²PAKE exemplifies this evolution, showcasing how PAKE can serve not only to verify the user's knowledge of a password while preserving its secrecy, but also to authenticate transactions—from logins, over authorized access to confidential information, to the signing of sensitive transactions—in a legally accountable manner [Fao+22].

As PAKE protocols continue to evolve, they demonstrate an inherent flexibility in adapting to the demands of a rapidly shifting cybersecurity landscape. They have carved a niche as versatile, user-friendly tools that significantly enhance digital security.

The insights drawn from this discourse underscore one prevailing notion: the true strength of PAKE protocols lies not in their ability to merely secure passwords but in their potential to facilitate secure, non-repudiable transactions that are vital in the age of digital transformation.

4 Security Analysis and Protocol Re-design: Towards Mutually Authenticated A²PAKE (mA³PAKE)

4.1 Introduction

This chapter delves into the intricacies of designing a robust Password-Authenticated Key Exchange (PAKE) protocol that not only ensures secure web authentication but also emphasizes user autonomy and non-repudiation. We explore the landscape of existing PAKE protocols, analyzing their security strengths and weaknesses, and identify areas where they could be improved, especially in the context of mutual authentication and accountability.

The crux of this chapter is the introduction of a proposed protocol, dubbed Mutually Authenticated A²PAKE (mA³PAKE), which integrates the principles of asymmetric cryptography and mutual authentication. The mA³PAKE protocol aims to address the challenges identified in existing PAKE systems while prioritizing user control and digital sovereignty.

As we embark on this journey, we critically analyze the current state of PAKE protocols and pave the way for a new era in password-based web authentication—one that is secure, efficient, and respects the user's digital identity and autonomy.

4.2 Security Analysis of Existing PAKE Protocols

4.2.1 Overview of Existing PAKE Protocols

Existing PAKE protocols, such as RSA-EKE, mRSA, and A2PAKE, have played crucial roles in the evolution of web authentication. RSA-EKE, a pioneer in PAKE protocols, laid the groundwork but faced vulnerabilities. mRSA, a strategic modification of RSA, addressed some of these issues by splitting the RSA private key, enhancing security. A2PAKE, representing the next evolution, integrated non-repudiation into PAKE protocols, expanding their scope beyond mere password authentication.

4.2.2 Security Strengths of Current Protocols

Each of these protocols has its unique strengths. RSA-EKE's simplicity made it an early favorite, while mRSA's key division improved resilience against certain attacks. A2PAKE's approach to non-repudiation added a layer of legal accountability to digital interactions, a significant advancement in PAKE protocol design.

4.2.3 Weaknesses and Vulnerabilities

Despite their strengths, these protocols exhibit vulnerabilities. RSA-EKE was susceptible to dictionary attacks, a significant drawback in password security. mRSA, while mitigating some of RSA-EKE's weaknesses, mRSA still relied on the integrity of the public key-share, which has its limitations. A2PAKE, though innovative, required careful implementation to avoid potential security loopholes.

4.2.4 Comparative Analysis

Comparing these protocols, it's evident that while they advanced PAKE technology, each had limitations in security, efficiency, or user experience. RSA-EKE's vulnerability to dictionary attacks, mRSA's dependency on key management, and A2PAKE's implementation complexities highlight the need for a more robust PAKE solution.

4.2.5 Implications for Web Authentication

The strengths and weaknesses of these PAKE protocols significantly impact web authentication. Their limitations underscore the need for a more secure, efficient, and user-centric PAKE protocol, capable of addressing the modern web's evolving security landscape and user demands.

4.3 Challenges in Designing Secure PAKE Protocols

Overview. In the realm of PAKE protocols, crafting a solution that is both secure and user-friendly presents an intricate challenge. This section explores key obstacles encountered in the design of secure PAKE protocols.

4.3.1 Security vs. Usability Trade-off

A primary challenge in PAKE design is balancing robust security with user convenience. Solutions must prevent a range of attacks, including offline dictionary attacks, without complicating the user experience. This balance is crucial for widespread adoption and effective security.

4.3.2 Vulnerability to Phishing Attacks

Despite advancements, all current PAKE protocols remain vulnerable to phishing attacks, granting an attacker one candidate password-guess per authentication attempt by the user. Devising mechanisms that safeguard against such threats, is a critical aspect of PAKE design.

4.3.3 Scalability and Efficiency Concerns

PAKE protocols must be scalable and efficient for deployment in diverse environments. They should accommodate a large number of users without significant performance degradation or undue resource demands.

4.3.4 Integrity of Public Keys

For protocols like mRSA, ensuring the integrity of the public key-share is vital, and hence a hindering burden. Our proposed mA³PAKE protocol aims to address this by using twin-secret keys, which decouple both server-side verification and authentication from client-side entropy.

4.3.5 Comprehensive Security Features

A well-rounded PAKE protocol should include features like forward secrecy and resistance to common cyber threats. It should also align with principles of user autonomy and digital sovereignty, as envisioned in our mA³PAKE protocol.

Conclusion. Designing a secure PAKE protocol is a multifaceted endeavor that requires a nuanced understanding of both cybersecurity and cryptographic protocols (and their design), user behavior, and technological constraints, i.e. in web-based use-bases. Our proposed mA³PAKE protocol aims to address these challenges, setting a new standard in secure web authentication.

4.4 Proposed Redesign: Explicit-Mutually Authenticated A2PAKE (mA3PAKE)

Overview. The proposed mA3PAKE protocol represents a significant redesign of traditional PAKE systems, integrating asymmetric cryptography and client-side authentication logic to create a robust, user-centric authentication solution.

4.4.1 Integrating OPRF with Asymmetric Cryptography

At the core of mA3PAKE is the integration of Oblivious Pseudorandom Functions (OPRFs) with an asymmetric cryptographic framework. This combination allows users to select a trusted OPRF service, which plays a pivotal role in the client-side derivation of a high-entropy seed from the low-entropy password, while maintaining user privacy and security.

4.4.2 Decoupling Server-Side Verification from Client-Side Entropy

mA3PAKE addresses the challenge of server-side verification by decoupling it from client-side entropy. This approach ensures that server-side authentication and verification processes are independent of client-generated entropy, enhancing security against phishing attacks.

4.4.3 Assuming Feasibility of Key Components

The design of mA3PAKE assumes the feasibility of certain key components, including the user's ability to easily verify the integrity of a consistent visualized value (fingerprint) and the authenticity of the OPRF server during registration. These assumptions are critical to the protocol's security and effectiveness. We will propose a RSA-based instantiating as well as implementation of this novel concept in Chapter 6, and 7 respectively.

4.4.4 Mutual Authentication through Twin-Secret Key-Pairs

mA3PAKE introduces a novel concept of twin-secret key-pairs for mutual authentication. The client possesses a signing key (ssk), enabling it to generate signatures that the server can verify with its corresponding secret verifier key (svk). Conversely, the server's signatures can be verified by the client using its ssk. This mechanism effectively creates a 'mirrored' variant of *Designated Verifier Signatures* (mDVS), thereby achieving (explicit) mutual authentication between client and server.

4.4.5 Use of Probabilistic Signatures

To further enhance the security of the mA3PAKE protocol, probabilistic signatures, such as RSA-PSS, are employed. This approach is designed to prevent attackers from validating key candidates against protocol transcripts, thus ensuring a higher level of security against replay and similar attacks. The use of probabilistic signatures adds an additional layer of uncoupling entropy and unpredictability to the authentication process, significantly bolstering its resistance to various forms of cyberattacks.

Conclusion. mA3PAKE represents a novel approach in PAKE protocol design, combining the strengths of asymmetric cryptography and OPRF with a focus on user autonomy and digital sovereignty. Its innovative design sets a new standard in secure web authentication, addressing key challenges and vulnerabilities of existing PAKE protocols.

4.5 Conclusion

Summary of Key Findings: This chapter has rigorously explored the landscape of PAKE protocols, highlighting their evolution and the emergent need for a protocol that harmonizes security with user autonomy. Our discourse culminated in the proposition of the mA3PAKE protocol, an innovative redesign which integrates mutual authentication and non-repudiation, while preserving user convenience and digital sovereignty.

Advancements in PAKE Protocols: The proposed mA3PAKE protocol represents a significant step forward in the realm of PAKE protocols. By incorporating advanced cryptographic techniques, it addresses the traditional vulnerabilities associated with password authentication, offering a robust and user-centric solution.

Impact on Web Authentication: mA3PAKE has the potential to revolutionize web authentication by providing a secure, efficient, and user-friendly authentication mechanism. This protocol could set new standards in digital security, enhancing the integrity and confidentiality of online interactions.

Future Exploration: The upcoming chapters will delve deeper into the critical building blocks of the mA3PAKE protocol, such as OPRF and our modified 'mutually authenticating' RSA variant (maRSA). These chapters will elucidate how these components contribute to the protocol's unique capabilities, paving the way for a comprehensive understanding of its operational and security features.

Closing Remarks: The exploration and development of the mA3PAKE protocol underscores our commitment to advancing the field of password-authenticated key exchange, with a keen focus on balancing security with user empowerment. As we move forward, we will continue to refine and optimize our approach.

5 Oblivious Pseudo Random Functions (OPRF) and OPRF Services

5.1 Introduction to OPRF

Common Understanding. Oblivious Pseudorandom Functions (OPRFs) have become a crucial component in modern cryptographic protocols, particularly in the realm of PAKE schemes. As outlined in 3.6.2, an OPRF Service [JKR18; JKR19][1] enables a client to remotely *and* securely compute a *secret function* on a *secret* input in such a way that the server learns neither the input nor the output, ensuring the privacy of the client's data, while neither neither the client, nor an adaptive adversary or exhaustive online brute force attack can learn anything about the underlying server-side secret to this function. This cryptographic construct has found increasing relevance in enhancing *secrecy* and *privacy*[2] in various applications.

5.1.1 Role in PAKE Schemes

In PAKE systems, OPRFs have quickly taken on an important role in securely handling secret, and especially low-entropy, authentication credentials. By transiently prior to or obscuring—i.e. 'blinding' or 'masking'—the actual password or key prior to or during (or both) the authentication process, OPRFs help in mitigating risks associated with various attack scenarios, such as password leaks, server compromises, and especially offline brute force and *online-guessing*. This is particularly vital in scenarios where password-based authentication remains prevalent, ensuring that the underlying credentials are never exposed to guessing or brute force attacks.

5.1.2 Externalizing or Internalizing OPRF Services

Externalization. By externalizing this crucial component to a trusted service provider[3], OPRF-evaluated secrets i.e. credentials, not matter how low their entropy may[4] be, become

[1] sometimes also contemporarily referred to as 'OPRF as a Service' (OPRFaaS) as in the cited references; however, not in the commercialized sense of Cloud-based 'XaaS' offerings.

[2] privacy enhancing mechanisms in the form of 'anonymization services.'

[3] who 'would never reveal a secret key `k` to any other party'; especially not to an authentication server.

[4] in ideal constructions, not even one single bit—in the sense that `secret.bitLength()` = 1 , or more commonly thought of as 'learning the equivalent of the state of one bit in the entropy of the `secret` '—will be leaked.

even immune to brute force attacks by the authenticating server itself; which constitutes precisely one of the declared main objective in the field of PAKE research.

Internalization. On the other hand, by internalizing the OPRF Service, with e.g. a self-hosted instance or self-owned trusted device as with 'DE-PAKE' [Jar+15] , not only the same security fortifying effect can be achieved, but also control and assurance of the secure handling and non-disclosure of the server-side secret keys can be established; thereby enabling user autonomy and digital sovereignty.

5.1.3 Application in Web Authentication

The integration of OPRFs into web authentication mechanisms marks a significant advancement in digital security. They allow web services to authenticate users without ever having direct access to their passwords, thereby enhancing the overall security posture against various forms of cyber threats, including server compromise, brute-force and eaves-dropping attacks.

Client-side Authentication Logic

Phishing. It is important to note, that especially in web-based scenarios, as outlined in 2.4.2, the mere *integration* of such an OPRF component into the existing framework—i.e. server-provided web-app and its mobile source code—is *insufficient*. In order to prevent phishing i.e. active man-in-the-middle, adaptive adversary, and server impersonation attacks, also the *relocation* of the client-side *authentication logic* for it to become 'trusted', is necessary.

Conclusion: The introduction to OPRFs sets the stage for their detailed exploration in the subsequent sections. We will delve into the specifics of implementing OPRFs in a server-client model, focusing on their role in secure web authentication and their integration into the proposed mA^{3}PAKE protocol. This will include a brief discussion on language and platform selection as well as early integration challenges, emphasizing the practical aspects of deploying OPRFs in real-world scenarios.

5.2 Considerations for Implementing OPRF Services

Options and Choices. Before beginning to implement an OPRF Service, it is imperative to first delve into a process of exploring and comparing various OPRF libraries, code repositories, and projects across programming languages and platforms. With the upcoming subsection, we are highlighting the importance of both a structured search for a well-maintained repository i.e. active development project, to provide a robust and reliable OPRF implementation, as well as careful considerations on the feasibility of its integration

into the client-server model; by our own example, that we experienced while implementing this crucial component for our mA³PAKE prototypes. As a result of such a structured search and well-considered choice, one may envision and draft a viable solution that, on the one hand, ensures both server- and client-side compatibility and security, as well as, on the other hand, is sufficiently simple to implement, maintain and further developed, and allows for both platform portability and scalability. *Easier said than done.*

5.2.1 Choosing JavaScript for Compatibility

After having compared and test a wide variety of available OPRF implementation across different programming languages and supposed platforms i.e. runtime environments, such as Python, Java, Go, and Javascript, our final choice of JavaScript as the programming language for our OPRF component was justified as follows.

Hurdles. Initially—for personal reasons—our early attention was drawn to *Python* i.e. `Flask`-Server based implementations like `nthparty/oprfs`, which seemily provide an 'out-of-the-box' solution, at least for the server-side. The server could indeed be setup and modified easily, yet the interaction with it from a browser-based client, which usually cannot readily execute python code, seemed cumbersome. Using a *JavaScript*-based client implementation to interact with a *Python*-based server, on the other hand, turned out to be technically almost unfeasible, or al least would require intense modification efforts on either or both sides. Similarly, even though both the corresponding server implementation `bytemare/voprf` in *Go*, and client/server implementation `cloudflare/voprf-ts` in *JavaScript* are based on Internet Draft `draft-irtf-cfrg-voprf-21` , eventually they turned out to be same as incompatible, without further adjustments.

Resolution. Even though, out of all available languages, *JavaScript* was the one we had the least experience with respect to server-side implementations, i.e. `Node.js`-based web-server development, we compare the two most promising implementation, `cloudflare/voprf-ts` and `multiparty/oprf` , which seem to suffice the requirements for code maturity and project sustainability, and made a preliminary decision for the latter, based on:

- the seamless compatibility of (*TypeScript*-to-be-transpiled to) *JavaScript* across both server and client environments, allowing for hurdleless integration in both, the popular `Node.js` web-server platform and regular web client i.e. browser applications, and especially browser-plugins like *Chrome Extensions;* and
- the quality and extent of the corresponding project and repository documentation, as well as the subjective developer experience and measurable coding effectiveness when working with their provided packages and functions, in order to adapt them to the web-based client-server deployment model; which we are covering in the following subsection.

Language	Author/Project	Repository	Scheme/Standard
Java	Pieter-Jan Vrielynck	pvriel/oprf4j	Custom ECC
C	Stefan Marsiske	stef/liboprf	Ristretto255
Python	Styepanovich Klyoovtuokmshnik	stef/klutshnik (liboprf wrapper)	Ristretto255
Python	Magnite / Nth Party	nthparty/oprf	Ed25519
Go	Daniel Bourdrez	bytemare/voprf	VOPRFv21
TypeScript / JavaScript	Cloudflare, Inc.	cloudflare/voprf-ts	VOPRFv21
TypeScript / JavaScript	Boston University / multiparty.org	multiparty/oprf	Ristretto255

Tabelle 1: OPRF Packages and Projects on Github

Comparative Overview. table 1 presents a comparative analysis of various OPRF implementations available on GitHub, showcasing their diversity in terms of programming languages, cryptographic schemes, and originating projects or organizations. This comparison was instrumental in our decision-making process for selecting an appropriate OPRF package for our mA^{3}PAKE implementation. The table highlights the trade-offs between different implementations, including language compatibility, cryptographic robustness, and community support. It underscores our rationale for choosing the `multiparty/oprf` package, balancing the requirements of code maturity, ease of integration, and robust cryptographic foundations.

5.2.2 Adopting Node.js by Conviction

Learning Curve. Embarking on the implementation of an OPRF service using Node.js and the `multiparty/oprf` package, we anticipated a steep learning curve. To our delight, the process was more accessible than expected, owing to the well-documented project resources, comprehensive online tutorials, and a vibrant community on platforms like Stack Overflow. The use of Visual Studio Code as an Integrated Development Environment (IDE) significantly enhanced our efficiency, offering dynamic debugging and effective code refactoring tools.

Lines of Code. Remarkably, the core functionality of our OPRF service, *after* input sanitization and handling of the client-ID, encompassing input validation, OPRF evaluation, output validation and encoding, and HTTP response management, as well as basic error handling for each, was distilled into less than 50 lines of code, including comments. This

```
const OPRF = require('oprf');
import axios from 'axios';
async function oprfClient(  serverUrl: string, clientId: string,
                            password: string, encoding: string = 'UTF-8') {
    /* ... */
}
module.exports = oprfClient;
```

Listing 1: oprfClient() function in TypeScript (1)

concise yet powerful codebase could be neatly displayed on a single screen, reflecting the modular and well-architected nature of the `multiparty/oprf` package. In contrast, auxiliary functionalities like input sanitization, client ID checks and database interactions spanned over 150 lines, underscoring the importance of robust input handling and client management in ensuring a secure and efficient OPRF service.

5.3 Adapting to Client-Server Mode

In adapting our chosen OPRF package for the client-server architecture, critical considerations included crafting a robust communication flow and integrating error handling mechanisms. The `oprfClient.js` file on the attached data medium (also available from our Github repository, see 8.4) delineates the steps the client takes to securely generate a password-derived high-entropy seed, referred to as `passseed` in this work, that is being generated and used in a way that is resistant to various attacks, particularly phishing. In the following we will break down the client-side implementation into its main component.

5.3.1 The oprfClient() Function

One of the central design choices for the client is, how *exactly* the OPRF function is to be executed by its requiring code. A function like `oprfClient()` could be either static function of a module or package, or a static function or method or a Class of one of its Instances, respectively. Due to the stateless nature of an OPRF evaluation and its single occurrence for each password being used to authenticated as a certain user to a certain authentication server, we decided for a static module function, as listed in listing 1, considering it appropriate, and the most reasonable option. This function has only the OPRF server's URL, the client's ID, and the user's password, as mandatory parameter. The parameter for the encoding of the `input` payload set to the server, is optional, and `UTF-8` by default. As the function is `async`, it returns a so-called `Promise` which resolves *asynchronously* either to an `Unit8Array`—i.e. the `passseed`—or to `Null`, in case of failure.

The conversion of the user's password to the input, to be sent to the server, is given in listing 2. After instantiating and initializing a `new OPRF` object (provided by the `multiparty/oprf` package), `password` is hashed and mapped to a Ristretto group point

```
const oprf = new OPRF();
await oprf.ready;
const hashPoint = oprf.hashToPoint(password);
const maskedPoint = oprf.maskInput(hashPoint);
const encodedMaskedPoint = oprf.encodePoint(maskedPoint.point, 'UTF-8');
```

Listing 2: oprfClient() function in TypeScript (2)

```
const response = await axios.post(serverUrl, {
    id: clientId,
    input: input
});
```

)

Listing 3: oprfClient() function in TypeScript (3

using the `hashToPoint` method. This—still `secret`—`hashPoint` it then being 'blinded' using the `maskInput` method, which uses a randomly generated `mask` and stores it as the `.mask` property inside the resulting `maskedPoint` object for 'unblinding' the server's `output`, again. The `.point` property of the `maskedPoint` has now both high-entropy, and—without `.mask`—no information about the password. Now, `.point` only needs to be encoded using the corresponding `encodePoint` method, and is ready to be sent to the server for the actual OPRF-evaluation, to be covered in section 5.4.

Using the popular `Axios` library for HTTP requests, the OPRF Client sends the encoded `input`—either in the original `UTF-8` used by default, or if otherwise specified, in **hex** or `base64url` encoding—together with the `clientID` as a JSON sting to the specified `serverUrl` and `await`s its response; as shown in listing 3

As illustrated in listing 3, the received `output` (in the `response` body) will then—pending on the sent encoding—be transcoded back to an encoding supported by the `decodePoint` method (`UTF-8` in most cases), and as such mapped back to the Ristretto group point that resulted from the server's OPRF-evaluation of the clients `maskedPoint.point` using its also *high-entropy, secret* 'salt' key. Using the `unmaskPoint` method of the object and `maskedPoint.mask` , the still 'blinded' `maskedSaltedPoint` is then being 'unblinded,' in order to obtain the—still high-entropy—*secret* `saltedPoint`, representing the `passseed` in terms of our proposed mA^{3}PAKE protocol.

```
maskedSaltedPoint = oprf.decodePoint(encodedSaltedPoint, 'UTF-8');
saltedPoint = oprf.unmaskPoint(maskedSaltedPoint, maskedPoint.mask);
return saltedPoint;
```

Listing 4: oprfClient() function in TypeScript (4)

```
Received output (salted masked point) from server.
Salted masked point (hex-encoded):
e0b588ef8a97e5a588e9bdb2e181a5e28f94eeaf8fefbfbdeca489ecbf87eca499e596a5ed9fbde38888ea89b4e38384
Decoding salted masked point...
Salted masked point is invalid.
This can happen... Trying again with a newly masked point.
Masking hash point...
Masked Point is valid.
Input to OPRF (hex-encoded):
e3a1bae69882eca593e19b95efbfbde9ba98eeb68ee49b9a23ecbeaae0bea4e39692e1a28defa3bde48fade4aa98
Sending input (masked point) to server (4lphaNumT3stId1234 @ https://oprf.reich.org) ...
Error from OPRF server: Masked point is not a valid point on the curve
This can happen… Trying again with a newly masked point.
Masking hash point...
Masked Point is valid.
Input to OPRF (hex-encoded):
e49f80e2b4afe7a08fe28b98e9879fe38180eeb49d2ce99cbeef9c97eb9897efbeb2ee95aceb89ade18482e39da1
Sending input (masked point) to server (4lphaNumT3stId1234 @ https://oprf.reich.org) ...
Received output (salted masked point) from server.
```

Bild 1: OPRF Client Errors

5.3.2 Challenges and Solutions

The integration of the chosen OPRF package to a real-world application scenario posed several challenges. Among these were validating the input, error handling, and managing the asynchronous runtime of JavaScript to sustain consistent program state across imported packages and asynchronous web requests. To address such obstacles, auxiliary functions for the encoding were implemented, and preliminary debug messages integrated, allowing for a focus on the relevant code and straightforward debugging.

Quirks. The by far most notable 'bug,' we would like to share with the reader, are hard-to-explain encoding bug that are allegedly inherited from the encodings supposed by `multiparty/oprf` : `UTF-8` and `ASCII`. Even though our auxiliary functions use the standard method of `Buffer.from()` and `.toString()` , which in all others of our use-cases work just fine, and we tried alternate approaches as well, still en- or decoding errors happen regularly. A example of this is depicted in fig. 1, where not only the client receives an invalid `output` from the server—most probably due to the more error prone trans-coding to and from `hex` —but also on the next try, using a newly generated `mask` , it sends an `input` that was either erroneously encoded by itself, or erroneously decoded by the server.
Solution. Our solution to this it as simple as effective: We have implemented the client with a `do-while` loop, that tries again with a new `mask` , until a valid `output` i.e. `passseed` is obtained. An error-free (slightly shortened) example output of the OPRF Client Demo Script `oprfClientDemo.js` (in chapter C) is shown in listing 5.

```
$ node oprfClientDemo.js
Trying to execute:
oprfClient("https://oprf.reich.org", "4lphaNumT3stId1234", "password123", "hex");
Hashing password to curve point... Hash Point is valid.
Masking hash point... Masked Point is valid.
Input to OPRF (hex-encoded): e5b086e7bb9aeaa8bbeb899de1a781ec95b....
Sending input (masked point) to server (4lphaNumT3stId @ https://oprf.reich.org)
Received output (salted masked point) from server.
Salted masked point (hex-encoded): eb929ce894bfe4bf88e192a0e4a29eef85baeab...
Decoding salted masked point... Salted masked point is valid.
Unmasking salted point... Unmasked salted point is valid.
Unmasked salted point (ASCII): ä²¶á«é¸½á¡·ãï¿½à·¯ç§á§³ì§âá§î¿çâ¸â¿½¶á³§
OPRF complete. oprfClient() executed successfully.
Your OPRF-salted 256 bit password hash (hex) is: b64cc61a3d9e771844353bdfef0...
You can now use this secretly salted password hash as a seed (passseed) for whatever
↪ you want to do with it. oprf.reich.org will store your client ID (4lphaNumT3stId)
↪ together with the (randomly selected) secret salt key for future requests.
Note: same password + same client ID = same passseed (Well, most of the times).
```

Listing 5: OPRF client demo script (oprfClientDemo.js) output

5.4 OPRF Server(less) Implementation on AWS Lambda with DynamoDB

5.4.1 Server-side OPRF Evaluation with Secret Key

Before we dive into the implementation details of our server(less) implementation as a Lambda Function with DynamoDB, we are now completing the big picture of the OPRF equation by presenting its essential yet missing *server-side* part, that was assumed to have been executed during the client's HTTP request in listing 3: the OPRF evaluation with the server's secret key. Stripping our implementation from all request handling, sanitization, database interactions, control flow instructions and error handling, the distilled essence are basically the two lines of code in 6: the scalar multiplication of the client's masked point i.e. its `input`, with the server's secret key, either retrieved from the database by the client ID, or randomly generated for a new ID; and the encoding of the resulting `saltedPoint`.

```
const saltedPoint = oprfInstance.scalarMult(maskedPoint, secretKey);
const encodedSaltedPoint = oprfInstance.encodePoint(saltedPoint, 'UTF-8');
```

Listing 6: OPRF Evaluation on server-side — scalar multiplication with secret 'salt' key

This concludes the OPRF specific element of our client-server implementation, based on `multiparty/oprf`, in the cryptographic sense. In the following section, we are highlighting the aspects of the server's runtime integration and the persistent storage of its secret keys.

5.4.2 Deciding for AWS Lambda and DynamoDB

In the rest of this section we are focusing on select technical aspects and decision-making processes involved in our choice of AWS Lambda, for our OPRF service to be implement for and deployed on, using a DynamoDB for persistence.

Generally speaking, we have come to value the combination of the stateless, on-demand compute models in AWS Lambda, with the high-performance of NoSQL provided by DynamoDB; and specifically speaking, because this combination fits well to the following properties of OPRF client-server models:

- the statelessness of the service itself,
- the one-flow communication that suffices an OPRF request, evaluation, and response,
- the low cost of an occasionally used Lambda function, and
- the significant scalability and reliability of the platform;

latter of which constitutes an essential requisite for cryptographic services that may experience variable and potentially vast workloads in the future.

5.4.3 AWS Lambda for Server-Side Logic

Serverless. AWS Lambda is a 'serverless' compute service that allows code execution in response to various events on a pay-as-you-go basis, scaling automatically with the incoming request volume. This makes it an ideal candidate for deploying the server-side logic of OPRF, which involves the computationally non-negligible operation of performing scalar multiplication on elliptic curve points, while incurring no fixed e.g. monthly costs. In several month of occasionally using our Lambda function, the accumulated costs for both Lambda and DynamoDB did never reach an amount not to be rounded off to zero.

Node.js in AWS Lambda

Serverless Building Blocks. AWS Lambda's compatibility with Node.js serves as a cornerstone for deploying serverless applications. By embracing Node.js, Lambda allows humbly equipped developers like us, to deploy our cartographic building blocks in a highly scalable, event-driven environment. This also perfectly aligns with the stateless nature of OPRF services, as outlined in the previous subsection. The adoption of Lambda-based Node.js functions as self-contained building blocks for cryptographic protocols not only taps into its non-blocking I/O model, ideal for handling cryptographic computations like those required in OPRF, but also leverages the vast ecosystem of Node.js libraries, such as the `multiparty/oprf` package. This integration facilitates the seamless execution of

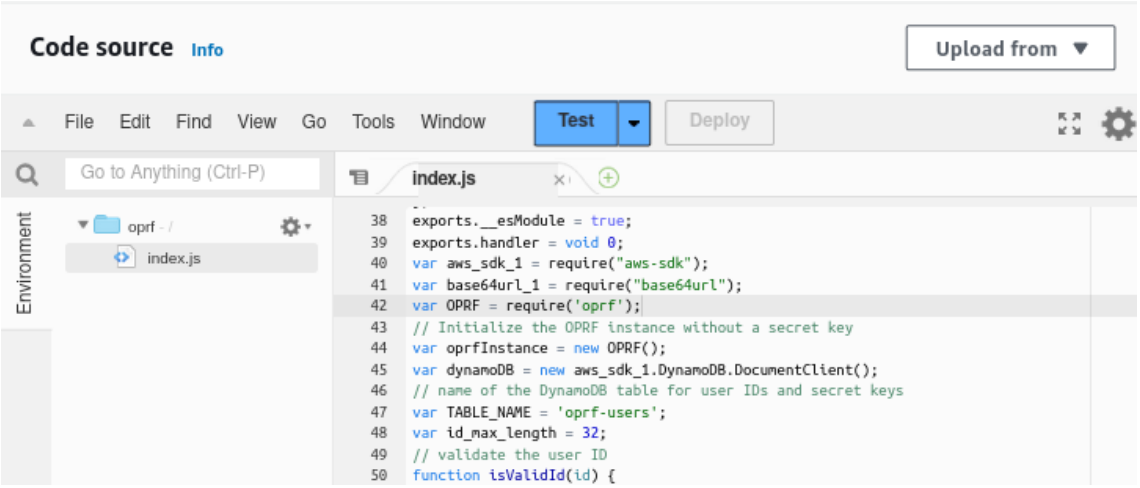


Bild 2: Lambda Dashboard: Node.js Code Source Panel

server-side logic, from scalar multiplication on elliptic curves to encoding results, while maintaining a cost-effective and scalable infrastructure.

The Lambda Platform

Motivating Insights. For both illustrative and inspirational i.e. motivational purposes, we are including screenshots from the AWS Lambda Dashboard, to showcase the aforementioned to be further explored integration aspects with Lambda. Both fig. 2 and fig. 3 are taken from the same page i.e. the 'Lambda Functions' Dashboard. The 'Code source' Panel is being displayed directly below the 'Function overview' when clicking on the 'Code' tab in between them. The code source panel not only let's us recognize the JavaScript-transpiled TypeScript source code of our server implementation, but many more, also offers functionalities and tools to seamless edit, test and re-deploy our function; or update it by uploading a new index.js file, or pulling it from a S3 bucket.

API Gateways

Seamless Integration: The AWS Lambda dashboard depicted in fig. 3 offers a clear visualization of the seamless integration between our `Node.js` Lambda function and the `API Gateway`. This setup allows the OPRF service to be triggered via HTTP requests, with the API Gateway acting as the interface for incoming client calls. Illustrated well by the attached 'plugin' in the bottom left corner, the dashboard indicates a streamlined connection, where the `API Gateway` (2 triggers) directly invoke our `oprf` Lambda function. We deem such an effortless means of integration as *pivotal* for the practical implementation and real-world deployment of cryptographic building block and services, as it not only simplifies the *request-routing* process but also enhances the scalability and manageability

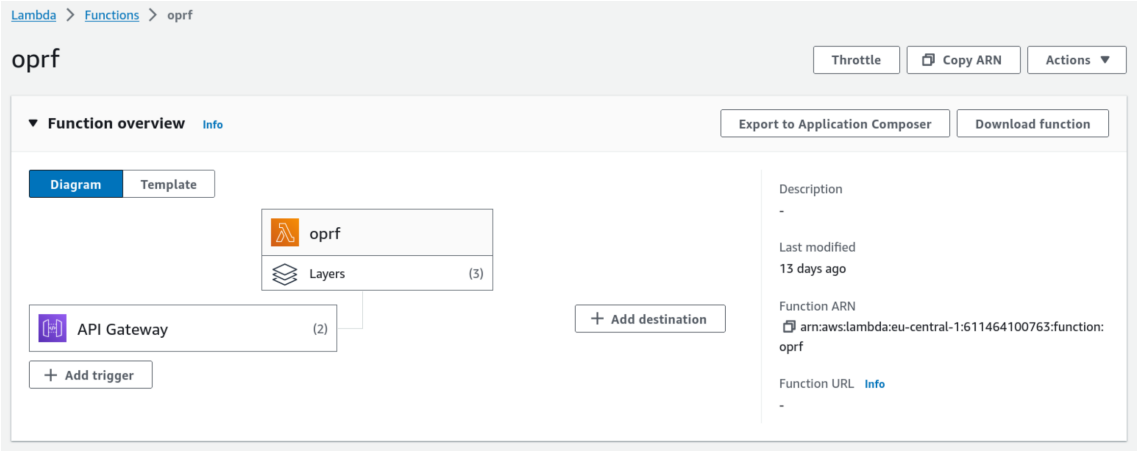


Bild 3: Lambda Dashboard: OPRF Function Overview

of the service, enabling smooth and secure interactions between the client and server, as it is desirable in the case of our OPRF-based PAKE protocol.

Lastly, the almost intuitive accessibility of platforms like AWS Lambda, also aligns with our envisioned aspect of digital sovereignty, due to the interoperability and generalized compatibility being adhered to, preventing vendor lock-ins, while handing over sufficient control of the integration and configuration management that these deployments are based on.

Lambda OPRF Service Handler

Seamless Execution and Scalability. The AWS Lambda function, meticulously architected as an OPRF handler, is a paragon of serverless efficiency. Its implementation, as depicted in listing 8, leverages the robust multiparty/oprf package within a TypeScript environment. This handler is the linchpin of the service, deftly managing incoming requests through the API Gateway, validating inputs with precision, and interfacing with DynamoDB for secure key management. This workflow culminates in the OPRF computation, transforming client inputs into cryptographically secure outputs. The handler embodies the principles of serverless design, ensuring high throughput and scalability while maintaining a steadfast commitment to the atomicity and security of cryptographic operations. It epitomizes the modern approach to deploying resilient cryptographic services, engineered to adapt to evolving AWS features and maintain a consistently swift, low-latency interface.

5.4.4 DynamoDB for Persistent Storage

A closer look. Having started with Lambda, the choice for DynamoDB as our OPRF server's persistent storage, was nearby. Beginning to understand its blazing efficiency and

```
import { APIGatewayProxyHandler } from 'aws-lambda';
import { DynamoDB } from 'aws-sdk';
const dynamoDB = new DynamoDB.DocumentClient();
const TABLE_NAME = 'oprf-users';
const OPRF = require('oprf');
let oprfInstance = new OPRF();
// Additional functions (isValidId, isValidUtf8Input, etc.) ...
export const handler: APIGatewayProxyHandler = async (event) => {
    const { id, input } = event.body ? JSON.parse(event.body) : \{};
    // OPRF Service Lambda function implementation ...
    return {  // Return the salted point as the response
        statusCode: 200,
        body: JSON.stringify({ id: id, output: output }),
    };
};
```

Listing 7: OPRF Service Handler Implementation for AWS Lambda with DynamoDB

native integration into the Lambda platform as an ideal means for managing client IDs and corresponding secret keys, we began to look deeper into it.

Serverless cryptographic functions

Architecting Serverless. Especially worth mentioning is DynamoDB's support for scalable and secure data storage in the serverless architecture, by offering a flexible and fast NoSQL database service that seamlessly scales to accommodate the fluctuating needs of internet-scale applications. Its integration into our OPRF function ensures the efficient handling and retrieval of client IDs and their corresponding secret keys, while unfolding an even broader set of attributes that are highly desirable when implementing *serverless cryptographic functions*.

```
// In case an ID is provided, try to fetch its secret key from DynamoDB
    let providedIdExists = false;
    let secretKey = Uint8Array.from([]);
    if (id) {
        // Check if the provided ID already exists in DynamoDB
        const result = await dynamoDB.get({
            TableName: TABLE_NAME, Key: { id }
        }).promise();
        // If the provided ID already exists, use the secret key of that ID
        if (result.Item) {
            providedIdExists = true;
            // Convert the secret key from a binary attribute to a Uint8Array
            secretKey = new Uint8Array(result.Item.secretKey);
        }
    }
```

Listing 8: OPRF Server secret key retrieved from DynamoDB by client ID

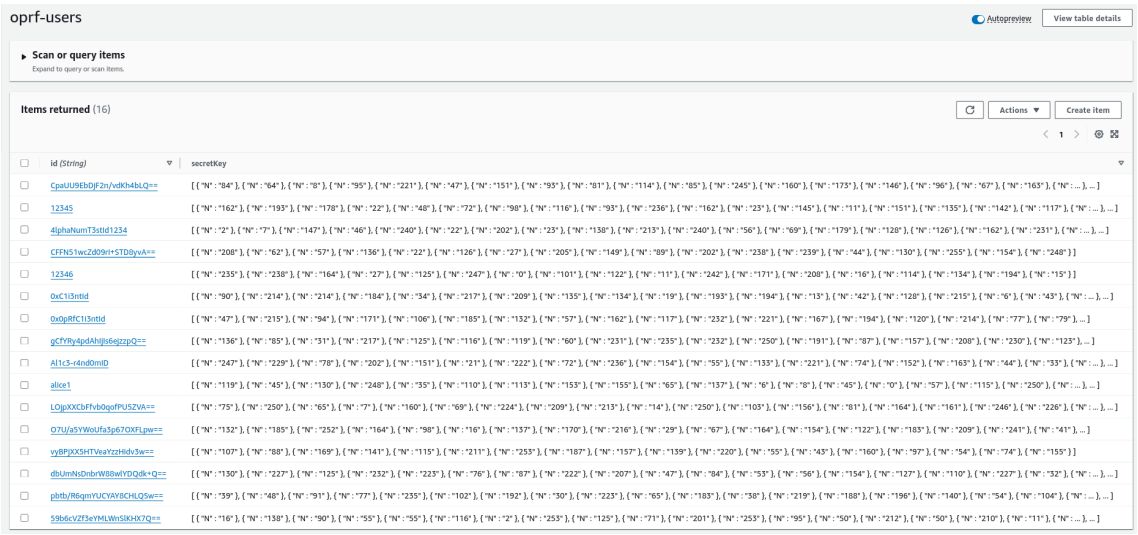


Bild 4: OPRF Client IDs & Server Secret Keys in DynamoDB

Secure Database Retrieval

Retrieving Secret Keys. The snippet in 8 illustrates the process of securely retrieving a secret key associated with a client ID from DynamoDB within the AWS Lambda function. When the OPRF server receives a request, it checks if the client ID is already registered in the database. If so, the existing secret key—stored in a serialization format—is fetched and transformed back into its original `UTF-8` format for `multiparty/oprf` to operate on, as shown in listing 6. This mechanism ensures that each client retains a unique, persistent secret key for consistent OPRF evaluations. DynamoDB's efficiency together with both AWS Lambda's access control management capabilities, and its seamless integration of the DynamoDB service client, make this retrieval both secure and swift, providing the backbone for a robust serverless OPRF service.

Optimized Database Interactions

Confidential Insights. The DynamoDB screenshot, as shown in fig. 4, showcases our `oprf-users` table where each `clientID` and its corresponding server `secretKey` is meticulously stored. Each entry is serialized in a format optimized for performance, allowing for rapid access and manipulation. The data retrieval, demonstrated in listing 8, is performed through a service client that interacts with DynamoDB in a *serverless context*, bypassing traditional database connection overheads. This not only accelerates the process but also enhances performance significantly. Furthermore, AWS's robust access control mechanisms enable us to rigorously assert, that our corresponding OPRF function has the exclusive permissions to access the table, thereby safeguarding the data and ensuring that only authorized service clients can retrieve existing secret keys, and insert new entries. This seamless and secure interaction between AWS Lambda and DynamoDB underpins the high efficiency

and security of our serverless approach to an OPRF service, contributing to its agility and reliability in cryptographic operations.

5.4.5 Conclusion

Synchronizing Cryptography and Cloud Services. The fusion of AWS Lambda and DynamoDB culminates in an OPRF service that inconspicuously orchestrates a 'symphony of serverless.' Their meticulous service begins when the client-side authentication logic invokes the `oprfClient()` function, necessitating an OPRF evaluation, in our case of the user's password. This consequent request, carrying the blinded password representation, is seamlessly transmitted via the API Gateway to a primed Lambda function. The Lambda function—acting as a vigilant custodian—verifies the request, liaisons with DynamoDB to secure the requisite secret key, and performs the OPRF evaluation. The service's denouement is the encoding and return of the salted masked point, which traverses back to the client to the `oprfClient()` function. Finally unblinded again, it returns to the authentication logic—for our purpose in shape of a `passseed`— to fulfill its cryptographic destiny. This inspiring choreography of request handling, data retrieval, cryptographic processing, and response delivery exemplifies the prowess of serverless architectures in the realm of cryptographic services, ensuring a workflow that is as reliable as it is resilient.

Through this exemplification, also rather salient attributes of the serverless architecture—especially AWS Lambda—were allowed to unfold and propose their potential by offering both scalability and cost-effectiveness. Concurrently, DynamoDB, with its potent and swift empowerment for serverless data retrieval and persistence, reinforces the architecture's attractive promises to developers as well as its inherent resilience, as an essential requisite for cryptographic services. This exploration may serve as a testimony to the practicality and implementability of OPRF within a serverless framework—underscoring its viability for practitioners and scholars alike, who may tread this path in their pursuit of cryptographic achievements.

5.5 Concluding Review on Security Aspects of OPRF Services

OPRF ensures that even if a server is compromised, the attacker gains no knowledge about clients' inputs or the corresponding outputs. The robustness of this approach lies in its resistance to replay, pre-computation, and server-side brute-force attacks, safeguarding the secrecy even if inputs are low-entropy or reused.

In a well-designed client-server model, OPRF services operate without revealing the server's secret keys, crucial for consistent cryptographic operations. This is achieved by employing a server that performs the OPRF evaluation without learning the client's input

or the evaluated output, thereby preserving privacy and thwarting potential adaptive adversaries.

The security considerations of our OPRF service implementation are underscored by a meticulous choice of cryptographic primitives, a serverless architecture that leverages AWS Lambda and DynamoDB for efficiency and secure data handling, and a rigorous workflow that assures the integrity and secrecy of the operations. By utilizing high-entropy seeds derived from user inputs, our service fortifies the generation of keys, ensuring that even with repeated usage or input reuse, the cryptographic strength remains uncompromised.

In conclusion, the security aspects of our OPRF service implementation are solidified by a blend of cutting-edge serverless technologies and established cryptographic assurances, thereby ensuring a secure, robust and efficient deployment of services for sensitive cryptographic operations.

6 Bi-Functionality, Post-Generation Independence, Generator-Base-Dependent Reproducibility: Unlocking RSA and the Unique Properties of its Key-pairs

6.1 Introduction

RSA (Rivest–Shamir–Adleman) is one of the first public-key cryptosystems and is widely used for secure data transmission. Its security is based on the practical difficulty of factoring the product of two large prime numbers, the factoring problem. In the realm of non-repudiable password authentication and *Mutually Authenticated* Auditable APAKE (mA³PAKE), we found RSA to present unique properties that can be harnessed for *novel* PAKE functionalities.

6.1.1 Outlook

This chapter aims to delve into three pivotal properties of RSA Key-pairs:

- **Bi-Functionality** (of each Asymmetric Key), to be explore in section 6.4,
- **Post-Generation Independence** (of both Asymmetric Keys), defined and explored in section 6.5, as well as
- **Generator-Base-Dependent Reproducibility** (of Key-pairs), proposed and justified in section 6.6,

while exploring both their definitions and mathematical underpinnings, as well as their practical applications in cryptographic protocols. Our special interest and attention thereby, is obviously dedicated to their contribution to *mutual authentication* (see section 6.1.2, and *auditability* functionalities of PAKE, while ensuring usability and user autonomy; for the latter of which aspects, we are proposing and justifying a fourth property, which the practical application of *Generator-Base-Dependent Reproducibility* relies on, and therefor is posed to as an implicit necessity:

- **Generator-Base Recoverability** (from Key-pair), to be defined and harnessed in section 6.7.

While not directly a 'property' in the narrow sense, we will encompass the four mutually related terminologies in an accompanying concept we coin the **'Key-pair Generator**

Base' or optionally 'Key-pair Generator *Configuration*' (for occasions where both 'generator' and 'base' might be otherwise confused with group generators in the sense of exponentiation bases); therefor starting the chapter with the corresponding section.

6.1.2 PAKE-MEA: The 'Mutually' Stronger Notion of PAKE-EA

Refraining from keeping the reader in suspense, we want to note that our contribution proposes an enhancement to the prevalent notion of PAKE-EA [Poi22] by fortifying its 'entity authentication' (or 'explicit authentication') with a concept for *explicit*-mutual i.e. 'mutually-explicit' authentication, or thereby strikingly 'mutual entity authentication', denoted as PAKE-MEA. We are achieving PAKE-MEA through a sophisticated combination of:

- the *Post-Generation-Independence*, leverage for
- a *Designated Verifier Signature* scheme (see section 3.6.3), and
- the *Bi-Functionality* of the designated *Secret Verifier Key*, denoted as *svk*;

which thereby also enables non-repudiable commitment of the designated-verifier-issuing party, i.e. the user in our case, to the issued *svk*, which in turn allows the user to create *auditable* signatures. — That was an intentious spoiler to this chapter, admittedly.

6.1.3 Background

The RSA algorithm, named after its inventors Ron Rivest, Adi Shamir, and Leonard Adleman, was proposed in 1977 and published the year after [RSA78], as a breakthrough in public key cryptography (PKC) [Bon99]. The algorithm leverages the mathematical challenge of factoring large numbers into their prime components, a problem for which no polynomial-time solution is currently known [Poi22]. The RSA algorithm has found widespread application in the secure transmission of data, digital signatures, and more, owing to its robust security underpinnings and extensive scrutiny in the cryptographic community [MVV96].

6.2 Key-pair Generator Base (KPGB)

In *asymmetric* cryptosystems, the generation of Key-pairs often relies on certain foundational elements or parameters, also referred to as the [Poi22]. In *distinguished (use-)cases* that we are going to explore, these parameters may be described as a *Key-pair Generator Base*, due to their essential role and facilitation as a cryptographic tool.

6.2.1 Randomizing Asymmetric Key-pair Generation

Commonly, asymmetric cryptosystems rely on *randomized parameters* as foundational elements as well as additional randomized input or *seed* for their Key-pair generation process, from which thereby *randomized* public-private key pairs are derived. These initially randomized parameters i.e. foundational element are sometimes also referred to as the *internal state* of a randomized and commonly thought of as *ephemeral* 'Key-pair generator' [MVV96; Poi22]. From its supposedly ephemeral internal state, however, potentially 'arbitrarily many' such Key-pairs can be *generated*.

Disclaimer

This in turn is generally and understandably, *strongly discouraged*, since this randomization is essential to the Key-pair's—especially to the private key's—security, in terms of *exclusive secrecy* and entropy-based *unguessability*.

6.2.2 Definition

A **Key-pair Generator Base**, abbreviated as **KPGB**, or simply referred to as the 'Generator Base' or 'Key Base', constitutes the randomized internal state of an asymmetric Key-pair generation process, which is being redefined as the 'secret configuration' of an intentionally reused **Key-pair Generator (KPG)** which compensated its sacrificed randomized ephemerality by an equivalently randomized **Key-pair Generation Seed (KPGS)** for each Key-pair it generates.

6.2.3 Exemplifying Instantiation

For `RSA` a *Key-pair Generator Base* consists of the randomized prime factors p and q, used to derive the—thereby equivalently randomized—*secret* private-key exponent d from the—usually 'not-so-random'—public exponent e. Respectively for `ECC`, we might assume a randomized base point $G_r = rG$, and for `NTRU` the random polynomial g , their prospective *Generator Bases*.

6.2.4 Challenge

The question remains, how *exactly* each of these cryptosystems' Key-pair Generators is supposed to compensate its sacrificed ephemeral randomness.

6.3 Defining Properties of KPGB, and their Generated Key-pairs

Behold. Before we however accept the aforementioned challenge, we are following the imperative to further conceptualize this newly introduced *Key-pair Generator Base*, and define its crucial properties and implications for its generated Key-pairs; *before* we are building on top of it, as an encompassing concept for the initially promised novel application of those Key-pairs' spacial properties to the field of PAKE—or even take on the challenge towards specific instantiations or practical implementations of such.

6.3.1 Generator Base Intractability (GB-I)

Since the prospective generation of potentially 'arbitrarily many' Key-pairs and their corresponding security guarantees will depend on it, the secrecy in terms of *intractability* of the Generator Base is also pivotal for the security of any cryptographic schemes and protocols built from them. If an attacker can guess, deduce or recover the Generator Base, they might gain an undue advantage in attacking such a system or protocol, eventually allowing them to *validate* their illicit key *candidates*, or even to generate *forged* private i.e. signing keys; and thereby to *impersonate* the rightful owner of the—then compromised—Generator Base.

6.3.2 Key-pair Unforgeability (KP-U)

Even though it is obvious, we are defining the term of Key-pair Unforgeability (KP-U) intentionally, denoting the existential unforgeability of both the 'private' *Secret Signing Keys* and their corresponding 'designated' *Secret Verifier Keys*. Without the Generator Base, neither of which is forgeable solely base on publicly disclosed or mutually shared information. The latter includes parties that have been issued a Designated Verifier Key; which must not allow them to forge any signing keys, nor any 'additional' verifier keys, belonging to the Generator Base that their designated *svk* was generated from.

6.3.3 Generator Base Dependent Key-pair Reproducibility (KP-R_GB-D)

The aforementioned notions of candidate validation and key forging attacks, based on an otherwise to be considered *compromised* Generator Base, already anticipated its potential abuse in forging otherwise unforgeable keys or Key-pairs. By the definition of 'Generator Base Dependent (GB-D) Key-pair Reproducibility (KP-R)' i.e. KP-R_GB-D, we are intentionally 'bridging the gap' between both notions, by allowing the Generator Base to *exclusively re-produce* the 'same key pair', again, based on the randomized seed that was meant to compensate for the Generator Base's sacrificed, *ephemeral* randomness; without trading-off any of the aforementioned security notions. This KP-R_GB-D property

is prospectively essential for schemes where users might need to recover, renew or validate their keys i.e. Key-pairs, *without* the ability to publicly validate any of their integrities (as opposed to the PKI setting), or simply storing them on a trusted device, remote storage or self-contained vault.

6.3.4 Post-Key-pair-Generation Key Independence (K-I_pKG)

As already noted in section 6.3.2, after i.e. *post*-Key-pair Generation (pKG), neither the *Designated Verifier Key*, nor the *Secret Signing Key* must be deducible from the other without the (secret) Generator Base. This notion constitutes a tightening of *KP-U* property (section 6.3.2, by the following additional notions:

- explicitly mandating *GB-I* from both the *Designated Verifier Key* and the *Secret Signing Key*, and thereby
- implying the absence of the Generator Base itself from any of the Key-pair material, and therefor
- only permitting such information to be contained in each of the keys, that is derived from the Generator Base via a One-Way (OWF) or Trapdoor Function (TDF);

ensuring that even if either one of the keys is potentially compromised, i.e. guessable or breachable, the other one—and most importantly the Generator Base—still remains secure, i.e. *intractable*.
Example. As an exemplification with RSA, such 'TDF-derived Key-pair material' would be constituted by the modulus N with the multiplicative product of p and q as the *trapdoor function*, considering factorization a hard i.e. intractable problem, as we have been assuming since section 6.1 and section 6.1.3.

6.3.5 Generator Base Recoverability (GB-R)

As already mandated by the property of GB-I (section 6.3.1, the Generator Base must not be deducible i.e. must be *non-recoverable* from either the Signing or the Verifier Key alone. It may however, be yet recoverable from *both* keys of the same Key pair, explicitly allowing for a *Generator Base Recovery Function* to—be it deterministically or probabilistically—recover the Generator Base from a 'pair of matching keys'; while also requiring a prospective Recovery Function to be *efficient*, for practical utilization. In sophisticated combination with the *KP-R_GB-D* property, this *GB-R* allows the users to *exclusively* recover or validate their Key-pairs, and to continue generating new ones from the very same Generator Base, without additionally *storing* any Key or Generator Base.
Requisite. The only requisite to *GB-R* for its practical utilization, is however, not only an efficient Generator Base Recovery Function, but also the users to be 'at some point' *able to recognize* their own Generator Base i.e. the portion or component of the Key-pair material

that was OWF- or TDF-derived from their corresponding Generator Base, 'somehow'—which we are covering in upcoming sections. Not to mention of course, that users also need to be able to both derive their Signing Key as well as retrieve the corresponding Verifier Key *securely*; which we are also covering in section yet to come.

6.3.6 Relation to Cryptosystems

While the Generator Base plays a crucial role in Key-pair Generation, the properties like Bi-Functionality emerge from the nature and inner workings of the cryptosystem itself and not directly from the Generator Base. However, properties like K-I_pKG, KP-R_GB-D, and GB-R are directly influenced by the degree of secrecy of the Generator Base in both the Key-pair Generation process, and its information theoretic appearance i.e. *(in)tractability* as components or portions inside the key materials and their further usage.

For instance, in NTRU, even though there exists a confidential Generator Base g, any random g' can be used to generate valid public keys h' matching a certain private key (candidate) f. This means NTRU lacks Generator-Base-*Dependent* Key-pair Reproducibility.

6.4 Pair-Key Bi-Functionality (PK-BF)

Bi-functionality refers to the capability of Key-pairs i.e. both of their 'Pair-keys' to be utilized for both encryption and signing purposes, a property that is not universally applicable across all asymmetric cryptographic systems, however it *is* the case in e.g. RSA.

This section will explore the mathematical and algorithmic aspects of Bi-functionality in RSA, elucidating how a single Key-pair can be employed for dual-purposes without compromising security. Furthermore, the implications of this property in the context of non-repudiable password authentication and mA^{3}PAKE will be discussed, providing insights into its practical applications and potential challenges.

6.4.1 Definition and Explanation

Bi-functionality in the context of RSA refers to the unique property where both the public key (e, N) and the private key (d, N) can perform encryption and decryption operations interchangeably due to their mathematical relationship. Specifically, given a message m, ciphertext c, public exponent e, private exponent d, and modulus N, the following holds:

$$c \equiv m^e \mod N \tag{6.1}$$

$$m \equiv c^d \mod N \tag{6.2}$$

This property emerges from the fact that the public exponent e and the private exponent d are multiplicative inverses modulo $\phi(N)$, where ϕ is Euler's totient function. Thus, they can reverse each other's operations in the asymmetric encryption and decryption scenario:

$$c^d \equiv (m^e)^d \equiv m^{ed} \equiv m \mod N, \tag{6.3}$$

as well as in the scenario of creating and verifying of a digital signature s:

$$s^e \equiv (m^d)^e \equiv m^{de} \equiv m \mod N \tag{6.4}$$

Comparing [**rsa_signature**] with [**rsa_decryption_deails**], we can clearly see, that the difference between encryption & decryption and signature creation & verification lays solely in the application of the underlying operations and handling of their inputs and outputs. Mathematically, signature creation in RSA is equivalent to encryption with the private exponent d (instead of the public exponent e), and signature verification equivalent to decryption with the public exponent e (instead of the private exponent d). From this fact, we can deduce, that both the public and private key are in principle capable of the exact same functionality, which we refer to as *bi-functional*.

6.4.2 Key-pairs and their Use-cases: What *Public* and *Private* really means

The so-called *public* key is usually being used exclusively for encryption and signature verification, and the so-called *private* key for decryption and signing. This naming convention however, is merely a consequence of the corresponding and undoubtedly prevalent key-management use-case called *public-key*.

Key management. In a scenario, where every participant would like to be able to establish secure communication channels with any other participant, asymmetric cryptosystems offer a much more efficient key-management solution than symmetric cryptosystems. Instead of potentially up to $n(n-1)$ symmetric keys, only n asymmetric Key-pairs have to be managed. For this purpose, every participant creates their own Key-pair, *publishes* one of them and keeps the other *private*. For efficiency, the published keys can be optimized in a way to make encryption and signature verification operations computationally inexpensive – as long as it doesn't weaken security. In the case of RSA, this means, the exponent of the public key can be set as a small prime number. This makes the modular exponentiation operation with it rather cheap, as compared to exponents with a size in the order of the modulus, taking computational burden off the clients and thereby improving user experience. At the same time, this won't weaken security, as modular exponentiation is a trapdoor function that is irreversible in polynomial time, even for small exponents – without the corresponding multiplicative inverse exponent at hand. As of 2012, the most commonly used exponents e were $\{3, 5, 17, 19, 35, 41, 65537\}$, which accounted for 99,8%[Len+12] of published RSA keys.

Conclusion: 'Public' are not inherently guessable or necessarily public, they are only being treated accordingly to a specific use-case, commonly referred to as the 'public-key' setting. We are still free to decide how we want to treat *our* asymmetric keys, especially Secret Signing and Verifier Keys; and most importantly we can control their degree of secrecy, entropy, and hence their unguessability and intractability—even, or *especially* for use-cases in which supposedly 'public' keys are being utilized as designated Secret Verifier Keys.

6.4.3 Application in Non-Repudiable Password Authentication

To date, use-cases for asymmetric cryptosystems in which neither of the keys is being published, are rare. So rare, that there isn't even a proper alternate naming for *public-key*, besides perhaps the seldomly mentioned, mostly disregarded concept of 'Designated Verifier Signatures' (see section 3.6.3).

As we will be exploring in depth in the next chapter, the scenario of client-server authentication can turn out as a use-case for strictly *unpublished-key*, or *dual-private key*, where one of them becomes the 'designated' verifier for the other. As a preliminary outlook, for the context of non-repudiable password authentication protocols like mA³PAKE, the reader is invited to imagine how the Bi-functionality of RSA keys may provide a robust foundation for secure and non-repudiable client-server communication; at this point, also reconsidering our PAKE-MEA notion introduced in section 6.1.2.

As for now, we leave it with the observation, that the ability of both keys to both encrypt and decrypt i.e. sign and verify messages should allow for a flexible and secure exchange of cryptographic material and authentication messages between two recurrently communicating parties *sharing* such a Key-pair in an appropriate i.e. secure fashion, that is compliant to the *KP-U* property delineated in section 6.3.2.

We conclude this subsection by the anticipation for the Bi-functionality property to eventually allow its facilitating participant to mutually authenticate each other and establish a secure communication channel, without having to rely on the availability and integrity of a *public* key; or of a *key-share*, as with mRSA it would usually be the case.

Inclined readers may ask themselves now, which role a low-entropy secret, like a readily guessable password could ever play for establishing high-entropy cryptographic keys, necessary for facilitating Bi-functionality. At this point, we would only like to gently and discreetly remind them of our preceding chapter, where we covered Oblivious Pseudo-random Function (OPRF) and especially OPRF Services — of the latter of which constitutes an implementation of the prior its prospective facilitation as a 'Password-to-Random' (P2R or PTR) service. As for now, we will leave it with the hint, that OPRF Services plays a crucial role for the final design and implementation of the mA³PAKE protocol, covered in the next chapter.

6.4.4 Counter Example: Elliptic Curve Cryptography (ECC)

In contrast to the RSA cryptosystem, Elliptic Curve Cryptography (ECC) does not exhibit Bi-functionality in its Key-pairs. In ECC, the public key Q is generated by multiplying the private key d with the base point G on the elliptic curve:

$$Q = d \cdot G \tag{6.5}$$

Here, the private key cannot perform the same operations as the public key and vice versa. The public key Q is used to verify signatures and encrypt messages, while the private key d is used to generate signatures and decrypt messages. This distinction in functionality between the key pairs in ECC and RSA highlights the presumed (at least partial) exclusivity of this property.

6.4.5 Implications and Utilization in Protocols

As already mentioned, the Bi-functionality of RSA keys not only facilitates versatile cryptographic operations, but may also introduce potential avenues for innovative protocol designs as in password-authenticated key exchange mechanisms. This property could allow protocols to leverage the interchangeable encryption and decryption capabilities of one of both Pair-keys to enhance security, provide non-repudiation, and ensure mutual authentication, while becoming independent of any public key infrastructure (PKI).

In the subsequent sections, we will explore the remaining proposed properties of Post-Generation Independence and Generator-Base-Dependent Regenerability by exemplifications with RSA, and delve deeper into their implications and applications in cryptographic protocols.

6.5 Post-Key-pair-Generation Key Independence (K-I_pKG)

Post-Generation Independence, as a presumably unique property of RSA, introduces a layer of security and flexibility in cryptographic protocols, particularly in the context of non-repudiable password authentication and Mutually Authenticated Auditable Asymmetric PAKE (mA³PAKE). This property implies that once an RSA Key-pair, consisting of a signing key (ssk) and a verifier key (svk), is generated, neither key can be derived or regenerated from the other without additional information, namely the original primes (p, q) used in the generation process.

6.5.1 Defining Post-Generation Independence

Mathematically, given an RSA Key-pair comprising of a public key (e, N) and a private key (d, N), where $N = p \times q$ and $e \times d \equiv 1 \mod \phi(N)$, the property of Post-Generation Independence asserts that:

$$\nexists \mathcal{F}(e, N) \to d \quad \text{and} \quad \nexists \mathcal{G}(d, N) \to e \tag{6.6}$$

where $\mathcal{F}$ and $\mathcal{G}$ represent functions that could derive the private exponent d from the public exponent e and vice versa, without knowledge of the original primes p and q. This property ensures that even if an adversary has knowledge or candidates of one key, they cannot derive the pair-key without additional information, thereby enhancing the security of the cryptosystem.

6.5.2 Cryptologic Significance

In the context of cryptographic protocols, particularly those discussed in chapter 4 concerning various attacker scenarios like phishing, server impersonation, Man-In-The-Middle (MITM), online-bruteforce, and online-guessing attacks, Post-Generation Independence emerges as a pivotal property.

Given an attacker has managed to generate or guess candidates for the secret signing key (ssk), the inability to validate these candidates without the corresponding secret verifier key (svk) introduces a significant barrier to unauthorized access or illicit message signing. This means that even with potential ssk candidates, without the actual svk (which is kept secret by the authentication server), the attacker cannot validate the probabilistic ciphers or signatures created, thereby thwarting unauthorized access or message forgery

$$\nexists \mathcal{H}(ssk) \to svk \tag{6.7}$$

where $\mathcal{H}$ represents a function that attempts to derive the svk from an ssk. This non-derivation property, ensures that the attacker cannot bruteforce nor try to guess the password, even when derived (among false ones) the valid ssk from the true password, without the corresponding svk. The only way for him to validate it, would be an actually 'normal' authentication attempt with the authentication server – or breaching the server to obtain the svk.

6.5.3 Contrast with Other Cryptosystems: ECC as a Counter Example

In contrast, Elliptic Curve Cryptography (ECC) does not exhibit Post-Generation Independence. In ECC, given a private key d and a base point G on the elliptic curve, the public key Q can be directly derived as:

$$Q = d \cdot G \tag{6.8}$$

This direct derivation of Q from d without requiring additional information presents a stark contrast to RSA's Post-Generation Independence, highlighting the unique security attributes offered by RSA in specific cryptographic applications and protocols.

6.5.4 Implications in Protocol Design

The implications of Post-Generation Independence in protocol design, especially in the realm of non-repudiable password authentication and mA^{3}PAKE, are profound. The property ensures that cryptographic operations, authentication processes, and message exchanges can be conducted securely, even in scenarios where one of the keys might be potentially compromised through an (yet unknowingly) right guess.

In the subsequent sections, we will delve deeper into the third property, which currently seems as unique to RSA, namely Generator-Base-Dependent Regenerability, and explore its implications, applications, and potential challenges in cryptographic schemes and protocol designs.

6.6 Generator Base Dependent Key-pair Reproducibility (KP-R_GB-D)

In the realm of cryptographic systems, the concept of regenerability is pivotal, especially when it pertains to the generation of Key-pairs. The term 'Generator Base Dependent Key-pair Reproducibility' (KP-R_GB-D) encapsulates the principle that the generation and re-generation i.e. *reproducibility* of cryptographic Key-pairs are inherently tied to a specific Generator Base, which is utilized during the Key-pair Generation process.

6.6.1 The Key-pair Generator Base (KPGB) of RSA

In RSA, the KPGB is typically constituted by two prime numbers, denoted as p and q. These primes are crucial in the generation of the public key (e, N) and the private key (d, N), where N is the product of p and q, and e and d are related in such a way that $e \cdot d \equiv 1 \mod \phi(N)$. The Generator Base (p, q) is fundamental to the RSA algorithm, providing the foundation upon which the security and functionality of the RSA Key-pairs are built.

In contrast, Elliptic Curve Cryptography (ECC) employs a different kind of base, represented by a base point G on the elliptic curve, which is used to generate Key-pairs (d, Q), where d is a private scalar and Q is the public point obtained by performing scalar multiplication $Q = d \cdot G$.

6.6.2 The Essence of Reproducibility in RSA

KP-R_GB-D in RSA implies that if the Generator Base (p, q) and either the public key (e, N) or private they (d, N) are known, the corresponding private/public key can be reproduced, by leveraging the mathematical relationships of d and e regarding (p, q). This property is particularly vital in scenarios where the user needs to establish multiple Key-pairs for continued secure communication or data retrieval, but can only manage one Generator Base and can only use one random seed for each Key-pair generation.

6.6.3 Cryptologic Interest and Implications

We regard and propose the KP-R_GB-D property as of significant cryptologic interest as it may introduce a layer of resilience and recovery into cryptographic schemes. Especially in scenarios where self-contained key management as described above and potentially also key recovery are critical, KP-R_GB-D provides a mechanism to reproduce keys, ensuring the continuity of secure operations without necessitating the re-establishment of new keys across all communicating entities.

Moreover, in the context of authentication protocols, KP-R_GB-D in combination with K-I_pKG ensures that even if a private key candidate has been guessed by an attacker correctly, the attacker cannot derive the corresponding public key, given that the Generator Base is undisclosed and the public key is not guessable i.e. pseudo-random.

6.6.4 Forward-Looking: A Glimpse into PAKE/mA³PAKE

As we delve deeper into the intricacies of PAKE and its herein proposed mutually-explicit authenticated variants in the subsequent chapter, the concept of KP-R_GB-D will surface as a pivotal element, especially in the design and implementation of secure, mutually-authenticated, and non-repudiable PAKE protocols like mA³PAKE. The ability to reproduce Key-pairs, particularly in adversarial environments, will play a crucial role in maintaining the integrity and security of communication channels, even in the face of partial compromise.

6.7 Generator Base Recoverability (GB-R)

In the realm of RSA cryptography, the security of the system fundamentally relies on the difficulty of factoring the product of two large prime numbers, p and q. The public and private keys, represented as (e, N) and (d, N) respectively, are generated based on these primes. However, a pertinent question arises:

> If both keys of a Key-pair are compromised, is the system still secure?

6.7.1 Disclaimer

It is imperative to understand that if both the public and private keys of an RSA Key-pair are breached, the integrity of the system is jeopardized. With possession of the public key, malicious actors can validate (guess, dictionary, or brute-force attack) private key candidates, and together with the valid private key potentially recover the original prime numbers p and q i.e. the Generator Base of RSA-type Key-pair Generators. A recovery of the Key Base would be fatal.

6.7.2 The Higher Purpose

We only accept this risk in case of PAKE i.e. mA³PAKE, because the 'only thing', that the secrecy and intractability of the Key Base shall protect, is 'the mere password itself.' This *higher purpose* however, *cannot*, and especially *not* by the abstract risk imposed by—usually discouraged—modulus reuse i.e. Generator Base utilization, *defeat itself.*

The security of RSA is fundamentally tied to the difficulty of factoring the product of p and q (denoted as N), especially when these primes are large. The process of recovering p and q from (e, d, N) is non-trivial but feasible under common conditions. While the specifics of this algorithm will be elaborated in the subsequent chapter, it is essential to recognize its existence and implications for the overarching security model.

In certain scenarios, especially in the context of mA³PAKE, GB-R in combination with K-I_pKG turns out as worth the otherwise unorthodox conduct of reusing the Generator Base.

6.7.3 Algorithmic Recovery of Primes Factors

The algorithmic recovery of p and q from the public and private keys (e, d, N) is a topic that has been explored in cryptographic literature, notably in the Handbook of Applied Cryptography (1996) [MVV96] and by Boneh (1999) [Bon99]. While the algorithms and methods for this recovery are efficient and well-documented, their application and implications in practical cryptographic protocols, especially those that intentionally violate certain cryptographic norms, are of significant importance.

6.7.4 Implications in mA³PAKE: Mitigation and Protocol Design

In the context of mA³PAKE schemes, where certain cryptographic norms will be consciously violated in the pursuit of higher purposes i.e. for achieving specific protocol objectives, the GB-R property introduces unique challenges and considerations. The protocol, while leveraging these presumably unique properties of RSA, or cryptosystems with equivalent

properties, must also contend with the potential vulnerabilities introduced by the recoverability of p and q. Ensuring the security and robustness of the mA^{3}PAKE scheme and corresponding protocol instantiations, while also navigating the nuanced vulnerabilities introduced by GB-R, necessitates a meticulous design of these protocols. In this respect, we will revisit the findings from Chapter 4 (section 2.2.2 at given time during design and implementation in the next chapter.

6.7.5 Conclusion

The exploration of Generator Base Recoverability, (GB-R) especially in the context of an authentication protocol that navigates the delicate balance of leveraging and violating cryptographic norms, provides a rich avenue for research and exploration. Ensuring the security, integrity, and robustness of such protocols, while also safeguarding against the vulnerabilities introduced by GB-R, is a complex yet vital component of their design and implementation.

6.8 Exemplifications: NTRUSign & Type-3 Pairings (T3P)

In the following two section we are exemplifying our proposed properties further with corresponding explorations of NTRUSign and Type-3 Pairings (T3P).

6.8.1 NTRUSign: Lattice-Based Digital Signature Scheme

Introduction to NTRUSign

NTRUSign is a lattice-based digital signature scheme that emerged as an alternative to traditional cryptosystems like RSA and ECC, offering potential resistance to quantum computer attacks. Unlike RSA, which is based on the factorization problem, NTRUSign relies on the hardness of lattice problems, making it an intriguing subject for cryptographic research and application.

Key Generation in NTRUSign

The key generation process in NTRUSign involves selecting two polynomials, typically denoted as f and g, from a specific polynomial ring. These polynomials form the basis for the private and public keys in the NTRUSign cryptosystem. The private key is typically a combination of f and its inverse modulo a specific polynomial, while the public key h is derived from the product of f and g. [Hof+03]

Exploration of our Proposed Property Definitions

By the following notes, we are briefly exploring NTRUSigns properties and assessing them by our proposed definitions.

Key-pair Bi-Functionality (KP-BF) Unlike RSA, NTRUSign does not exhibit KP-BF. In RSA, both the public and private keys can encrypt and decrypt (or sign and verify signatures), but in NTRUSign, the private key structure and operation are distinct from those of the public key. The NTRUSign private key cannot perform the same operations as the public key, underscoring a fundamental difference from RSA's KP-BF property.

Post-Generation Independence (K-I_pKG) In NTRUSign, once a Key-pair is generated, the public key h cannot be derived from the private key without additional information, such as the polynomial g. This property presumably aligns with the K-I_pKG property, where deriving one key from another post-generation is infeasible without knowing the Generator Base, which in case of NTRUSign may be constituted by g. However, it turns out, that instead of the original Generator Base g, actual any suitable polynomial g' can be used instead, in order to derive, i.e. *forge* a valid public key h' from the private key f, which not only violates the property of KP-U, as outlined in section 6.3.2, but retroactively also invalidates its presumed K-I_pKG property.

Generator Base Dependent Key-pair Reproducibility (KP-R_GB-D) NTRUSign allows for the reproducibility of key pairs given the generator polynomial g and initial parameters. This property would be crucial for scenarios where users might need to regenerate their keys. However, without fulfilling KP-U, in fact not only the users could regenerate keys, but also attackers could derive verifier key candidates h' from their illicit private key candidates f' in order to validate the latter against intercepted or phished signatures created with the true private key f, which in turn, renders NTRUSign inappropriate for the Designated Verifier Signature setting.

Generator Base Recoverability (GB-R) In NTRUSign, recovering the generator polynomial g from the public key h alone is not straightforward, which adds a layer of security. However, if both pair-keys f and h are given, g can in fact be recovered by the corresponding inverse of f, that can directly be calculated given the public parameters. This aligns NTRUSign with the GB-R property.

Algorithms and their Adoption

The mathematics of NTRUSign revolves around polynomial algebra in a lattice framework. The signature generation and verification involve complex operations in this polynomial

space, distinct from RSA's number-theoretic approach. Even if NTRUSign was to fulfill the required properties, due to its complexity and unfamiliarity, its broader adoption remain unlikely.

Conclusions and Outlook on Post-Quantum Instantiations of mA³PAKE

The unique properties of NTRUSign, particularly its approach to Key-pair generation and the lattice-based security model, make it an interesting candidate for integration into advanced cryptographic protocols. However, the lack of KP-BF and KP-R_GB-D render it useless for mA³PAKE. However, exploring vast field of alternate asymmetric cryptosystems in the same way as we just did with NTRUSign, could eventually lead to a quantum resilient instantiation of mA³PAKE. This brief exploration into NTRUSign not only adds an initial understanding of lattice-based systems, but also exemplified the property-based assessment of Key-pair Generation and usage across various cryptosystems; and how these compare to more familiar ones, like RSA, particularly with regards to property definitions as requisites for specific schemes and applications.

6.8.2 Type-3 Pairings (T3P)

Introduction to Type-3 Pairings (T3P)

Type-3 Pairings, often referenced as asymmetric or non-degenerate pairings [GPS06], present a unique structure in the world of cryptography. They involve operations over two distinct groups, providing a bilinear mapping between them. This capability lends them a significant edge in cryptographic protocols, particularly those requiring enhanced efficiency and security.

Key Generation in T3P

In T3P, the key generation involves selecting generator points P and Q from two different groups, G_1 and G_2. The private key is a scalar s, while the public key is the point sP derived through scalar multiplication.

$$\text{Private Key: } s \in \mathbb{Z}_p \tag{6.9}$$

$$\text{Public Key: } sP \in G_1 \tag{6.10}$$

Exploration of Proposed Property Definitions in T3P

In the following, we are briefly exploring the properties of T3P keys and assessing them by our proposed definitions.

Key-pair Bi-Functionality (KP-BF) T3P inherently support KP-BF, enabling interactions between elements of two distinct groups. This property is crucial for cryptographic operations like identity-based encryption and signature schemes. The bi-functionality in T3P is fundamentally different from RSA's. In T3P, it arises from the ability to perform pairings between elements of G_1 and G_2, by using one's own private and public key together with the other participant's public key, as opposed to the trapdoor-permutation scheme of RSA, where already one of the two pair-keys is sufficient to perform the permutation.

Post-Generation Independence (K-I_pKG) In T3P, the K-I_pKG property is not directly applicable. While the scalar multiplication result in a public key that is in fact solely dependent on the private key, T3P use two of these private key, one for each participant. This in turn means, that not only would an adversary have to guess the private key of one of the participant in order to perform its candidate validation against intercepted of phished signatures, but also the attacker would have to breach the other participants private key. This security property highlights a contrast with RSA, where K-I_pKG is based on the mathematical structure of the cryptosystem and its resulting Key-pair generation process, rather than on the innerworkings of its trapdoor permutations of RSA; and the pairing operation or T3P, respectively.

Generator Base Dependent Key-pair Reproducibility (KP-R_GB-D) For similar reasons also KP-R_GB-D is not directly applicable in T3P, as the Key-pairs are not derived from a secret Generator Base or otherwise ephemerally randomized Key-pair generation process like in RSA or NTRUSign. Hence, the concept of a Generator Base need to be fundamentally altered in the context of T3P due to its reliance on the properties of bilinear pairings and the distinct group structures.

Generator Base Recoverability (GB-R) For this very reason (re-definition of KP-R_GB-D) also GB-R is not directly applicable in T3P. The mutual independence of its Key-pairs in combination with its lack of a traditional Key-pair generation process i.e. Generator Base, also spares the requirement of GB-R. Even though not relying on a traditional Key-pair generation process, the security of T3P and its Key-pairs is equivalent i.e. in a specific sense even stronger, while it is rooted in the computational complexity of the bilinear pairing operation and the discrete logarithm problem in the respective groups.

Applicability to mA³PAKE

A Generator Base of T3P, if applicable at all, would be trivially constituted by the Key-pair of one of the two participants. In case of mA³PAKE, this would mean the every server a user registers with, would be issued the same server-side private-public Key-pair,

however at the same time, each server would be issue an individual public i.e. Designated Verifier Key from the client-side.

Implications and Utilization in Protocols

T3P's unique structure and properties make it a valuable tool in cryptographic protocols, particularly those requiring identity-based encryption or signature schemes. The bi-functionality and post-generation independence in T3P offer innovative opportunities for protocol designs that leverage the distinct characteristics of bilinear pairings. Understanding how T3P can be integrated into cryptographic protocols, given its unique properties, can lead to the development of more efficient and secure cryptographic systems. This section on Type-3 Pairings (T3P) adds a detailed analysis of our proposed properties, highlighting yet again, their prospective potential for the advanced exploration of cryptographic primitives as well as the design and instantiation of novel authentication schemes. This in turn, provides a contrasting perspective to traditional schemes like RSA and ECC, and even enables the comparison with rather unfamiliar asymmetric cryptosystems like NTRU i.e. NTRUSign, thereby demonstrating the diversity and complexity inherent in modern cryptographic systems.

6.9 Evaluating Common Signature Schemes for mA³PAKE: The Necessity of Post-Generation Independence

6.9.1 Criteria for mA³PAKE Signature Schemes

For the mA³PAKE scheme, the designated verifier signature (DVS) scheme must facilitate a secret verifier key (svk) that is post-generation independent (K-I_pKG) from the secret signing key (ssk). This independence is critical to ensure that an adversary cannot validate prospective key candidates against intercepted signatures, thereby enhancing the security of the mA³PAKE protocol.

6.9.2 Analysis of Signature Schemes

In the following we are briefly checking this notion against other common signature schemes to conclude this chapter with a comparative and insightful overview.

ECDSA and Schnorr Signatures

In both ECDSA and Schnorr signatures, the public key (i.e., verifier key) does not possess K-I_pKG from the private key. Thus, these schemes do not meet the necessary criteria for mA³PAKE. The absence of K-I_pKG implies that if an adversary guesses a signing key

candidate correctly, they can also compute the true corresponding verifier key, and thereby in turn validate their signing key candidate against intercepted or phished signatures.

Ring Signatures

Ring Signatures, while offering signer ambiguity, require the integrity of additional keys. This requirement is similar to the public key-share in most mRSA variants, making them insufficient for mA³PAKE. The reliance on additional keys undermines the desired independence and security properties needed for mA³PAKE.

BLS Signatures

Applying the proposed recognizably of the KPGB by e.g. a visualized fingerprint, to BLS Signatures would lead to a scenario where all servers would necessarily share the same private key, hence making it a global variable. While BLS could theoretically be adapted for mA³PAKE, it would require careful custom implementations and cannot leverage existing library support, unlike RSA-based signatures. [BLS01]

Homomorphic Signature Schemes

Homomorphic Signature Schemes might offer potential for mA³PAKE, however their applicability and implementation would need to be thoroughly investigated, considering the specific requirements of the protocol. Also their contribution and prospectively broad adoption to especially user authentication schemes remains questionable.

Commitment Schemes

While commitment schemes can enhance existing signature schemes, they are not self-contained signature schemes with specific cryptographic properties. Their role in mA³PAKE would be ancillary, providing additional security layers to the primary signature mechanism, and prospectively to the registration and commitment process (with e.g. a Registration Authority).

6.10 Conclusion

The choice of a suitable signature scheme for mA³PAKE is pivotal. The scheme must ensure that each of a user's svk's remains secure and independent even if one of them or its candidate is compromised or correctly guessed. This analysis underscores the challenge in

selecting an appropriate signature scheme that not only meets the cryptographic requirements but also aligns with practical implementation considerations for the mA^{3}PAKE protocol.

7 Design and Implementation of a Non-Repudiable Password Authentication Protocol

Introduction

In the rapidly evolving digital landscape, the security and integrity of password-based authentication remain paramount concerns. This chapter delves into the intricate design and practical implementation of a novel Non-Repudiable Password Authentication Protocol (mA³PAKE). Anchored in the latest cryptographic advancements, this protocol aims to address the pivotal challenges plaguing conventional password-based systems, namely vulnerability to online password guessing attacks and the lack of non-repudiation.

The chapter begins by exploring the nuanced aspects of safe prime generation, crucial for cryptographic systems like RSA, and its implications on performance and implementation, especially when leveraging the Web Cryptography API's crypto.subtle interface in mobile applications. A comprehensive discussion on the mathematical and computational complexities behind generating primes and safe primes sets the stage for understanding the protocol's foundational elements.

The subsequent sections focus on the innovative use of the Drunken Bishop algorithm for fingerprint generation, offering an ASCII-based visualization of RSA modulus hashes, thus enhancing user authentication in phishing and MITM attack scenarios. This leads to a detailed exploration of the Boneh99 algorithm, a cornerstone in the RSA cryptosystem, enabling the recovery of prime factors from public and private exponents. The application of Boneh99 in password-based authentication marks a significant shift in traditional PAKE schemes, providing users with a more streamlined and secure experience.

The core of the chapter, "mA³PAKE: Protocol Design,"delves into the intricacies of developing a non-repudiable password authentication system. It highlights the necessity of externalizing the OPRF, employing distributed trust through multi-party computation, and the unconventional yet effective use of RSA, particularly the RSA-PSS. The protocol sequence diagram and its accompanying description outline the meticulous steps involved in the authentication process, emphasizing user-centric design and robust security mechanisms.

Concluding with the practical implementation of the mA³PAKE protocol, the chapter presents a detailed walkthrough of the authentication phase, showcasing the protocol's resilience against prevalent security threats and its adherence to principles of non-repudiation

and user sovereignty. This comprehensive exposition not only reinforces the protocol's theoretical strengths but also demonstrates its practical applicability in enhancing the security landscape of password-based authentication in the digital world.

7.1 Performance and Implementation Subtleties in Safe Prime Generation

Prime number generation is a fundamental aspect of many cryptographic systems, including RSA. The performance of prime generation algorithms can significantly impact the overall performance of these systems. This section discusses the performance characteristics of prime generation, implementation subtleties when using the Web Cryptography API's 'crypto.subtle' interface, in the case of mobile apps, and the mathematical and computational complexity theories behind generating primes and safe primes.

7.1.1 Mathematical and Computational Complexity Theories

The generation of prime numbers is a probabilistic process. The algorithm randomly selects a number and tests it for primality. The probability of a randomly selected number being prime is inversely proportional to its size, so the expected time to find a prime number increases exponentially with the bit size.

Safe primes are primes p such that $(p-1)/2$ is also prime. Generating safe primes is even more computationally intensive than generating regular primes. The expected time to find a safe prime is roughly the square of the expected time to find a regular prime of the same size. This is because, for each candidate prime, we must also test whether $(p-1)/2$ is prime.

7.1.2 Performance Characteristics

The time complexity of generating prime numbers is not linear, but exponential. This means that doubling the bit size of the primes to be generated will more than double the time it takes to generate them. Our experiments confirm this behavior. For example, generating two 1024-bit safe primes took approximately 1-5 seconds, while generating two 2047-bit safe primes took between 20 seconds and 2 minutes, more than a tenfold increase in average.

7.1.3 Implementation Subtleties

When using the 'crypto.subtle' interface for key generation, we observed a significant increase in generation time when the key size reached 2048 bits. This "magic wallät 2048 bits suggests that the underlying cryptographic library may change its behavior at this

threshold. Many cryptographic libraries use different algorithms or optimizations depending on the key size, switching to a more efficient but complex algorithm for larger key sizes. We did not look into the specific implementation of the ‘crypto.subtle‘ interface to find out what is causing the 2048 bits barrier. With respect to existing and upcoming recommendations for minimum RSA key lengths from recognized standardization bodies, it is already clear, that efficient browser- or mobile app-based implementations of safe prime generators would require adequate hardware support; and hence either bindings to native binaries like OpenSSL or an equivalent crypto API functionality that gives direct access to hardware-supported prime generation.

7.1.4 Implementing an Efficient Safe Prime Generator App on iOS and Android

Implementing an efficient safe prime generator app on iOS and Android would greatly benefit from leveraging the cryptographic libraries provided by the platforms and optimizing the prime generation algorithm for mobile devices. However, similar as with the browser implementation, also mobile platforms offer little native support for prime generation.

iOS Implementation

On iOS, the Common Crypto library provides a function ‘CCRandomGenerateBytes‘ to generate random numbers. However, it does not provide a direct function to generate prime numbers. Therefore, we would need to implement a prime number generation algorithm ourselves.

A common approach is to generate a random number of the desired bit length, and then test it for primality. If the number is not prime, generate a new random number and repeat the process, or simply iterate the generated random number until until a prime number is found. For safe prime generation, we also need to check that $(p-1)/2$ is prime.

To improve efficiency, we can use the Miller-Rabin primality test, which is a probabilistic test that is much faster than deterministic tests for large numbers. We can also use multithreading to generate and test multiple numbers in parallel, taking advantage of the multicore processors in modern iOS devices.

Android Implementation

On Android, the Java Cryptography Architecture (JCA) provides a class ‘java.security.SecureRandom‘ that can be used to generate random numbers. The ‘java.math.BigInteger‘ class provides a method ‘probablePrime‘ that generates a probable prime number of the specified bit length.

To generate safe primes, we can directly generate a prime number p, and then check whether $(p-1)/2$ is also prime. If not, generate a new prime number and repeat the process until a safe prime is found.

To improve efficiency, we can use the Miller-Rabin primality test implemented in the 'BigInteger.isProbablePrime' method. We can also use multi-threading to generate and test multiple numbers in parallel.

In both iOS and Android implementations, it's important to note that generating large safe primes is a computationally intensive task and may take a significant amount of time. Therefore, the app should be designed to handle long-running operations, for example by using background threads or asynchronous tasks.

In conclusion, the performance characteristics and implementation subtleties of prime generation algorithms can significantly impact the performance of cryptographic systems. Understanding these factors is crucial for the efficient implementation of protocols and applications that rely on safe primes.

7.2 Fingerprint Generation Using the Drunken Bishop Algorithm

In the mA3PAKE protocol proposed in this work, it is crucial for the phishing and MITM mitigation for users to identify and validate their personal RSA modulus. In order to provides an ASCII-based visualization method for their modulus' hash value, the Drunken Bishop algorithm is used as a fingerprint generator.

7.2.1 Origin

This type of hash visualization algorithm originates from the paper *Hash visualization: a new technique to improve real-world security* (1999) [PS99] by Perrig and Song, who were later credited for their inspirational idea behind common implementations.

7.2.2 Introduction of the Drunken Bishop Algorithm in OpenSSH

The Drunken Bishop algorithm was "devised and implemented by Alexander von Gernler during OpenBSD Hackathon 2008" and introduced in OpenSSH 5.1 thereafter. The goal was to provide the SSH user with a method for visualizing SSH fingerprints they previously had to compare in clear-text hexadecimal strings. According to von Gernler this technique was also inspired by "random art" graphical hash visualization schemes and by Dan Kaminsky's discussions on the subject at the 23C3 in Berlin [**Ger08**]. In his commit comment he briefly described the algorithm as simulating "a worm crawling over a discrete plane, leaving a trace [...] everywhere it goes". The movement of the worm is determined by the raw digest of the SSH server's public key fingerprint (hash).

7.2.3 Security Analysis

The first work to analyse security properties of the algorithm was the paper T*he drunken bishop: An analysis of the OpenSSH fingerprint visualization algorithm* (2009) [LLG09] by Loss et al., with von Gernler as co-author. The authors were the first to call the algorithm the "Drunken Bishop" and described the same as "moving randomly on a chessboard". In chess the bishop makes only diagonal steps and can only move on the same tile color, however, it the algorithm hits a wall, it now moves to the side, switching from one tiles color to the other. The pattern of his moves are drawn by placing a "coin on the floor" at each new position "to mark that he has been there before". The number of coins left on each board tile forms the fingerprint of the public key. The algorithm has several interesting properties. For example, it has been shown that the bishop is more likely to visit certain positions on the board than others, leading to a non-uniform distribution of coins. This property can be analyzed using a Markov model. However, the security of the Drunken Bishop algorithm is not fully understood. While some initial brute-force attacks have been conducted, further research is needed to determine whether the algorithm is secure enough for its intended purpose.

7.3 Recovery of *keybase* (p, q) from (d, e, N) using Boneh99

The RSA cryptosystem, one of the foundational pillars of modern cryptography, is based on the mathematical properties of large prime numbers. Specifically, the security of RSA relies on the difficulty of factoring the product of two large prime numbers, p and q. However, under certain conditions, it is possible to efficiently recover these primes given both the public exponent e and the private exponent d, and the modulus N.

In this work the exponents are usually denoted as s and v to account for the deviating key generation process as well as their special properties and usage. Thus, the heading should state (s, v, N), however, to stay consistent with the literature underlying literature, the author decided to stick with (d, e, N) for this section.

7.3.1 The Boneh Recovery Algorithm

Boneh's method, introduced in 1999, referred to as "Boneh99ör "Boneh's (p, q) Recovery", leverages the properties of RSA's private exponent d and its inherent relation to e via Euler's totient $(p-1)(q-1)$. Form the very design of the RSA cryptosystem, it was clear that with the factorization of N, the private key d can be efficiently recovered. Twenty years later, Boneh could show that the converse holds also true, and given some RSA public key (N, e) and its corresponding private key d, one can efficiently factor the modulus $N = p \cdot q$. Boneh99 is a testament to the power of number theory in cryptography, as it ingeniously reverses the RSA key generation process.

The core idea behind Boneh's algorithm is based on the special relation between the exponents e and d in the very modulus that allowed to derive d from e to begin with. Namely, that modulus is the Euler's totient function of N, or $\phi(N)$ for short, which in the case of $N = p \cdot q$ evaluates to $(p-1)(q-1)$. Mind, we only need to find a valid candidate for either p or q, and can then easily calculate the other. The aforementioned special relation between e and d is probably best-known as modular inverse (often implemented as functions called e.g. `modInv()`), meaning that d is calculated as the multiplicative inverse of e in modulus $\phi(N)$. By definition, the product of a any group element with its multiplicative inverse (if it exists) equates to the multiplicative neutral element, i.e. 1. This can be written as $e \cdot d \equiv 1 \mod \phi(N)$. From this point onward, Boneh applies a series of number-theoretic implications, algorithmic techniques, and clever complexity reduction in order to reverse engineer the quasi *forgotten* modulus that must have been used when calculating the *modInv* equation for the RSA key-pair generation.

Boneh starts with the implicit assumption of $e \cdot d - 1 \equiv 0 \mod \phi(N)$ and makes the observation that given e and d we can calculate the value $k = e \cdot d - 1$, which must be a multiple of $\phi(N)$, because $k \equiv 0 \mod \phi(N)$. What follows next can be seen as the clever preparation of a search for unity square roots. From the obvious primality property it follows that p, q are odd, i.e. both $(p-1)$ and $(q-1)$, and hence also $\phi(N)$ are even. From there, he goes on to deduce the expression of $k = 2^t \cdot r$, where r is odd and $t \geq 1$, thereby enabling an efficient unity square root search in the range of $2^1, 2^2, , 2^t$, as we will shortly see. Boneh now stages the unity square root search by exploiting both

- a) the fact that every element g in $\mathbb{Z}_N^*$ satisfies the property $g^k \equiv 1 \mod N$ and
- b) his direct conclusion that $g^{k/2}$ must be a square root of unity modulo N.

From here, Boneh must have been delighted when he discovered, that the Chinese Remainder Theorem (CRT) can provide a significant shortcut. The CRT indicates that 1 has four square roots modulo $N = p \cdot q$. While two of them are trivially ± 1, Boneh found that the other two, denoted as $\pm x$, satisfy the interesting congruences $x \equiv 1 \mod p$ and $x \equiv -1 \mod q$, which reveal the factorization of N as $\gcd(x-1, N)$. Looking back at his "clever preparationöf t with $k = 2^t \cdot r$, as well as a) and b), we only now begin to understand Boneh's crucial observation:

> If g is chosen randomly from $\mathbb{Z}_N^*$, then with a probability of at least $1/2$, one of the elements in the sequence $g^{k/2}, g^{k/4}, \ldots, g^{k/2^t} \mod N$ is a square root of unity that reveals the factorization of N.

He concludes with an assurance that all elements in sequence $g^{k/2}, g^{k/4}, \ldots, g^{k/2^t} \mod N$ can be efficiently computed in cubic time $O(n^3)$ with n as bit-length of N.

1: Compute $k = e \cdot d - 1$
2: Factor k to get $k = 2^t \cdot r$ where r is odd
3: **for** each iteration **do**
4: Choose random g in range $[2, N-2]$
5: Compute $y = g^r \mod N$
6: **if** $y = 1$ or $y = N - 1$ **then**
7: Continue to next iteration
8: **end if**
9: **for** $j = 1$ to $t - 1$ **do**
10: $x = y$
11: $y = y^2 \mod N$
12: **if** $y = 1$ **then**
13: Compute $p = \gcd(x - 1, N)$
14: $q = N/p$
15: **return** p, q
16: **end if**
17: **if** $y = N - 1$ **then**
18: Break and continue to next iteration
19: **end if**
20: **end for**
21: $y = y^2 \mod N$
22: **if** $y = 1$ **then**
23: Compute $p = \gcd(x - 1, N)$
24: $q = N/p$
25: **return** p, q
26: **end if**
27: **end for**

Algorithm 7.1: Boneh's (p, q) Recovery

7.3.2 Application in Password-Based Authentication

The Boneh recovery algorithm presents a novel approach in the realm of password-based authentication. By allowing the recovery of the user's *keybase* (i.e., p, q) from any server's registered verifier key *svk*, it eliminates the need for users to store their *keybase* in a key vault. This is a significant departure from traditional PAKE schemes, where the emphasis is on securely storing and retrieving user credentials by either multi-party computation, or hidden credential retrieval (HCR), or trusted devices (see e.g. DE-PAKE and SPA), or any combination of those.

In the proposed scheme, upon successful login, the server sends its *svk* (encrypted) to the user. The client, armed with the *svk* and the user's secret signing key *ssk*, and hence (v, s, N), can then recover the *keybase* (p, q). This recovered *keybase* can subsequently be used to re-register with the same password but a different OPRF-derived signing exponent s i.e. separate *ssk* and corresponding *svk* on other services.

This approach is groundbreaking for several reasons:

- It simplifies the user experience by removing the need for external trusted devices or services.
- It enhances security by ensuring that the *keybase* can only be recovered if a svk is leaked and the attacker manages to online-bruteforce the correct *ssk* candidate from the used OPRF.
- It acknowledges the inherent risks of password-only authentication while providing a robust mechanism to mitigate them.

A Note on RSA Modulus Reuse: In general, reusing the same RSA modulus across different contexts is considered a bad practice and is discouraged. However, in this specific use-case, the only secret the RSA modulus protects is the user's password. If an attacker can recover the *keybase*, they have already compromised the user's password. Thus, the emphasis shifts from protecting the *keybase* to ensuring the password's security.

A Note on complexity: With the cubic time $O(n^3)$ Boneh referred to the calculation of elements in sequence $g^{k/2}, g^{k/4}, \ldots, g^{k/2^t} \mod N$, however one might ask, how many such elements there may be. It turns out, that in our special case, the actual execution time of this algorithm is most significantly determined by the safe prime property of our p, q. With $p = 2 \cdot p' + 1$ and $q = 2 \cdot q' + 1$ our value for $k = l \cdot (p-1)(q-1)$ breaks down to $k = l \cdot 4 \cdot p' \cdot q'$. As p', q' are prime i.e. odd they will be factors of r in $k = 2^t \cdot r$ and hence, we can expect our t to be significantly below average.

7.3.3 Conclusion

The integration of Boneh's recovery algorithm into password-based authentication schemes represents a paradigm shift in how we perceive and handle user credentials. By allowing users to recover their keybase from any registered *svk*, it offers a seamless and secure authentication experience. As with all cryptographic innovations, it's essential to understand the underlying principles and potential risks, but with proper implementation, this approach holds great promise for the future of online authentication.

For the JavaScript implementation of the Boneh recovery algorithm being used in the client prototype, refer to Appendix A "recoverPQ.js".

7.4 mA³PAKE: Protocol Design

7.4.1 Introduction

The design of a non-repudiable password authentication protocol requires a deep understanding of cryptographic principles, the nuances of existing authentication mechanisms, and the potential vulnerabilities they might harbor. This chapter delves into the design considerations and decisions that led to the development of the proposed protocol.

7.4.2 The Need for Externalizing the OPRF

OPRF as a Service

One of the initial challenges was the integration of the Oblivious Pseudo Random Function (OPRF) within the protocol. It became evident that for flexibility and enhanced security, the OPRF needed to be externalized, allowing it to be optionally used äs a Service". This design decision not only decentralizes the trust but also offers the possibility of leveraging multi-party computation across multiple OPRF services.

7.4.3 Distributed Trust through Multi-Party Computation

By distributing the OPRF computation across multiple services, the protocol inherently becomes more resilient. An adversary would need to compromise multiple services to gain any significant advantage, making the system inherently more secure.

7.4.4 The Role of Symmetric Key Encryption

A pivotal realization was the necessity of a symmetric key to encrypt the server's signature. This step ensures mutual authentication while protecting the signature from potential private key validation attacks by malicious entities.

7.4.5 Unconventional Use of RSA

Unique Properties of RSA-PSS

The RSA-PSS (Probabilistic Signature Scheme) was chosen due to its property that a signature can only be verified by its corresponding public key. This property is crucial for ensuring the non-repudiation aspect of the protocol.

Inverting the RSA Key Generation Process

Traditionally, the RSA public key (with a typically low-entropy public exponent) is determined first, followed by the derivation of the private key. However, for the protocol's requirements, this process had to be inverted. The private key is determined first, making the subsequent public key essentially a byproduct of the process. This inversion was necessary to ensure the independence of the keys once the prime numbers p and q are discarded.

Encrypting with the Private Key

Another unconventional approach in the protocol is the encryption of the symmetric key using the user's private RSA key. While this is not a common practice, it was a necessary step to ensure the integrity and confidentiality of the symmetric key in the protocol.

7.4.6 Splitting User Authentication

Pre-Authentication

Before the actual authentication process, a pre-authentication step was introduced. In this phase, the user authenticates themselves and signs the encrypted symmetric key.This step ensures that the user has the necessary credentials and is ready for the next phase.

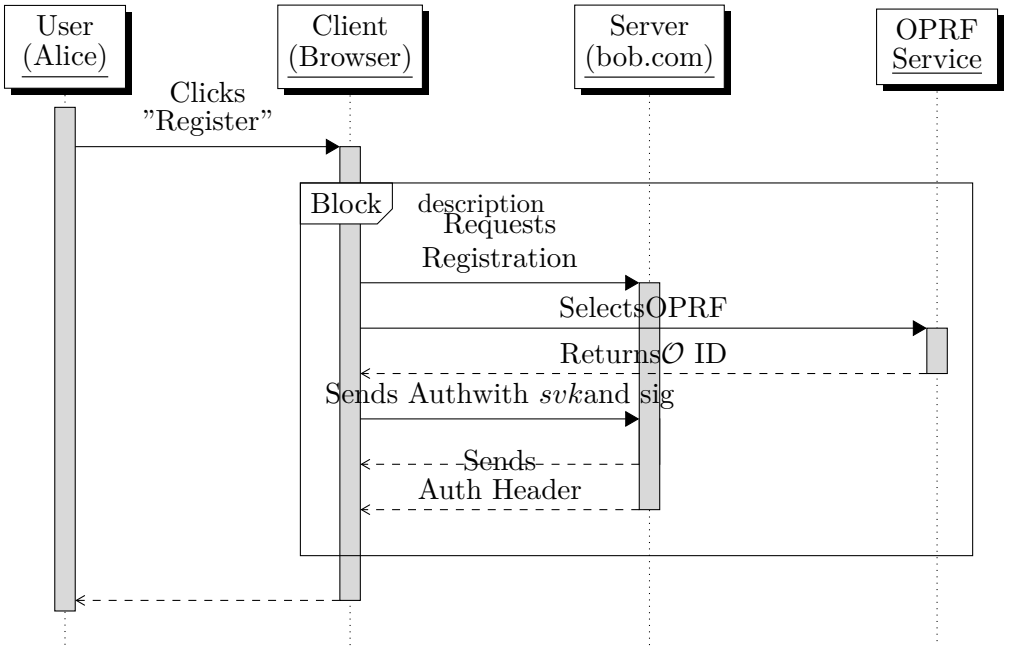
Authentication

Only after the server has authenticated itself to the user does the actual authentication process commence. At this stage, the user is willing to sign the actual "(trans)action", such as a login, ensuring mutual trust between both parties.

Conclusion

The design of the non-repudiable password authentication protocol presented numerous challenges, each requiring a deep understanding of cryptographic principles and a willingness to challenge conventional methodologies. By addressing each pitfall meticulously, the protocol ensures robust security, mutual trust, and non-repudiation, making it a significant contribution to the realm of password authentication.

7.4.7 Protocol Sequence Diagram



1. **Initialization**: The user, Bob, navigates to `alice.com` and initiates a new session. The server responds with a session cookie.
2. **Login Request**: Bob clicks on "My Alice-Account", prompting the server to generate a login action with a timestamp.
3. **User Input**: Bob is prompted to enter his username and password. The browser then blinds the password using the OPRF blind function.

4. **Server Processing**: The server processes the login request, associates the session with the user, and generates a challenge. Additionally, the server provides Bob with his OPRF ID and the address of the OPRF service to use. The server also shares the RSA modulus that will be mutually used for subsequent cryptographic operations.
5. **OPRF as a Service (OPRFaaS)**: The browser sends the blinded password to the OPRFaaS, which returns the evaluated OPRF value. The browser then unblinds this value to derive the private key.
6. **Pre-Authentication**: The browser sends a pre-authentication header to the server. In this step, Bob doesn't sign the action (login) yet. Instead, he demonstrates to Alice that he possesses the private key (i.e., knows the password). This step ensures Alice that Bob's encrypted symmetric key can be trusted. If an attacker were to obtain Alice's signature, they wouldn't be able to test private key candidates against it without this assurance.
7. **Authentication**: The browser decrypts the server's signature and verifies it. Upon successful verification, Bob signs the äction"that Alice prompted in the Login Request step, thereby completing the actual authentication process.
8. **Successful Login**: Upon successful verification by the server, it sends the content of "My Alice-Account"to the browser, indicating a successful login.

7.5 Implementation of mA3PAKE Protocol

In this section, we will outline the implementation of the mA3PAKE protocol. The protocol is implemented in JavaScript and uses the Forge library for cryptographic operations.

7.5.1 Authentication Phase

The authentication phase is the core part of the mA3PAKE protocol. It involves a series of interactions between the client (Alice) and the server (Bob). The function 'authenticationPhase(client, server)' simulates this process.

7.5.2 Login Invitation

The process begins when Alice sends a GET request to Bob's server to log into her account. Bob's server responds by initiating a login/authentication process. It generates a nonce and a timestamp, and sends a login invitation to Alice's browser.

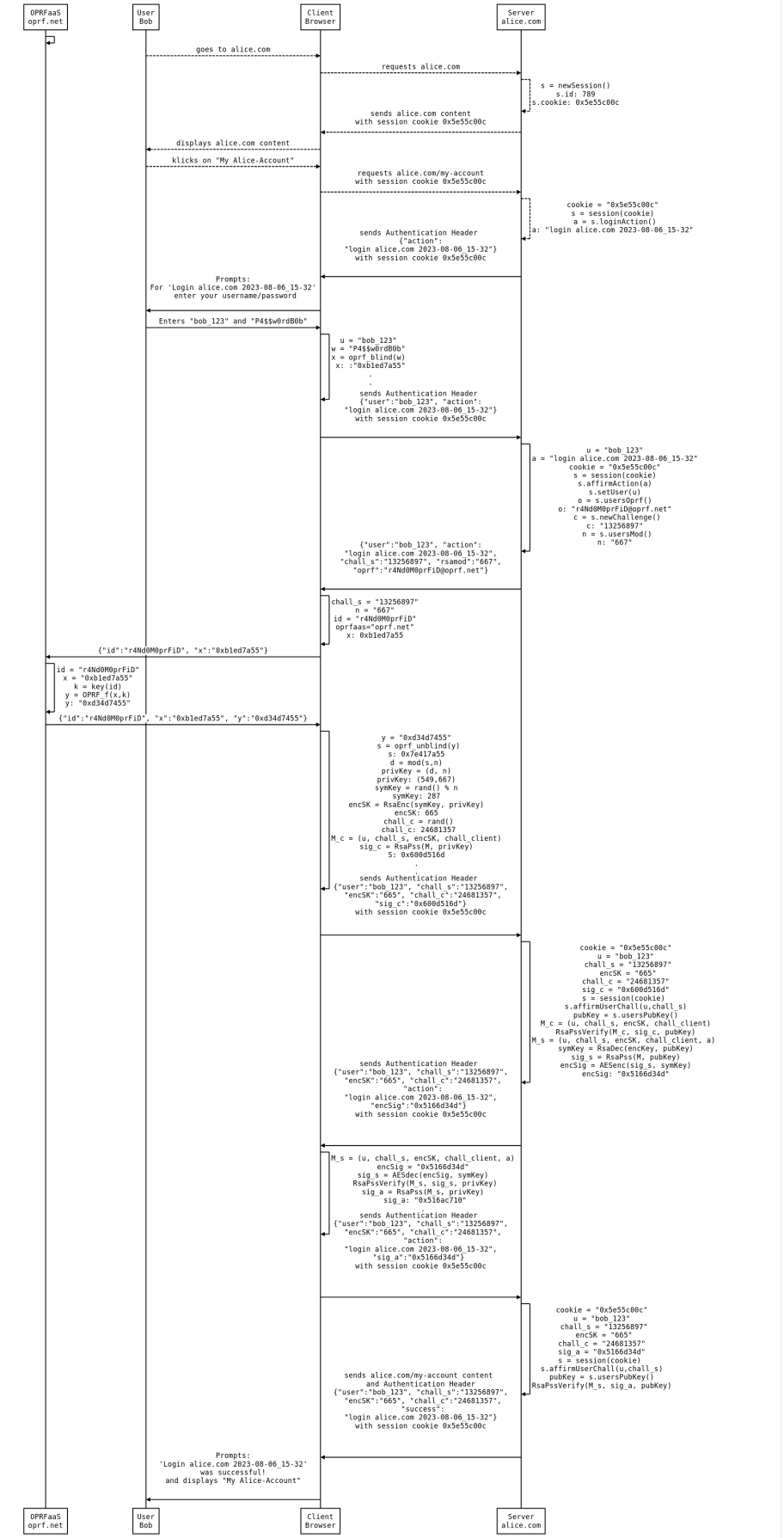


Bild 5: Protocol Sequence Diagram

7.5.3 Login Request

Upon receiving the login invitation, Alice's browser prompts her to enter her username and password. It then updates the login message with the username and a nonce, and sends it back to Bob's server.

7.5.4 Login Challenge

Bob's server receives the login request and checks if the username exists in its database. If it does, the server creates a login message containing Alice's username, her OPRF server and client ID, her modulus, and her fingerprint. This message is sent back to Alice's browser.

7.5.5 Login Response

Alice's browser receives the login challenge, validates it, and starts the OPRF evaluation of her password. It then derives the secret exponent s from the passseed, constructs the secret signing key ssk from s and n, and signs the authentication message with ssk. The browser also creates a random AES key and OAEP-encrypts it with ssk. The login message is updated with the signature and the encrypted AES key, and sent back to Bob's server.

7.5.6 Login Confirmation

Bob's server receives the login response, verifies the signature with Alice's verifier key svk, and sends a login confirmation message to Alice's browser. If Alice has registered with 'svkRetrieval' and requests it now, the server includes the encrypted svk.

7.5.7 Key Recovery

Finally, Alice's browser receives the login confirmation, decrypts the encrypted signature and the encrypted verifier key with her AES key, and verifies the signature with her signing key ssk. The browser then checks the modulus of the decrypted verifier key against the one in the login message. If they match, the browser recovers her key-pair generator base (p, q) from s and v

Conclusion. The mA^{3}PAKE protocol provides a secure method for Alice to authenticate herself to Bob's server and recover her key-pair generator base. The protocol is resistant to online guessing and key candidate validation attacks and provides perfect forward secrecy. The implementation in JavaScript demonstrates the practicality of the protocol.

7.6 Conclusion

The culmination of this chapter's exploration into the design and implementation of the mA³PAKE protocol represents a significant stride in enhancing password-based authentication systems. By meticulously addressing the complex facets of cryptographic security, particularly focusing on non-repudiability and resistance to online guessing attacks, mA³PAKE emerges as a pioneering solution in the realm of digital authentication.

The protocol's foundation in safe prime generation, coupled with the innovative use of the Drunken Bishop algorithm for RSA modulus hash visualization, not only bolsters security but also improves user experience in identifying and validating personal cryptographic elements. The incorporation of Boneh99's algorithm for the recovery of prime factors from RSA exponents further exemplifies the protocol's commitment to advanced cryptographic techniques, ensuring robust protection against a wide array of cyber threats.

Throughout the chapter, the narrative seamlessly integrates complex theoretical concepts with practical implementation strategies, demonstrating the protocol's viability in real-world applications. The protocol design, with its emphasis on externalizing the OPRF and leveraging distributed trust through multi-party computation, marks a paradigm shift in traditional password authentication approaches. The sequence diagram and detailed implementation steps provide a clear roadmap for the protocol's operation, highlighting its user-centric approach and its capability to establish a secure and trustworthy digital environment.

In essence, the mA³PAKE protocol stands as a testament to innovative cryptographic research and application, bridging the gap between theoretical robustness and practical usability. Its comprehensive design not only addresses current security challenges but also lays the groundwork for future advancements in password-based authentication systems. As digital security continues to be a paramount concern in our increasingly connected world, protocols like mA³PAKE are pivotal in shaping a more secure, reliable, and user-friendly digital landscape.

8 Conclusion

8.1 Recapitulation

This thesis embarked on a comprehensive exploration of password-based authentication, delving deep into its intricacies, vulnerabilities, and potential enhancements. From the conventional mechanisms employed on the web to the avant-garde approaches like Passwordless, we traversed a spectrum of methodologies, each with its unique set of advantages and challenges.

8.2 Key Contributions

Our primary contribution lies in the design of the non-repudiable password authentication scheme and protocol mA^{3}PAKE, leveraging Oblivious Pseudo Random Functions (OPRF) as a OPRF Services. This design not only addresses the inherent vulnerabilities of traditional password-based systems but also introduces a paradigm shift in how authentication can be both secure and user-friendly. The protocol's resilience against prevalent attack vectors, especially (one-time) online guessing during server impersonation, is proposed as a new benchmark for the field PAKE protocols.

8.3 Future Implications

The potential of OPRF Services, especially when combined with asymmetric cryptosystems that are immune against private key candidate validation attacks, opens up avenues for further research and application. The protocol's dependency on RSA-PSS, while currently a strength, also underscores the need for future-proofing against quantum threats. The quest for a quantum-resistant asymmetric cryptographic system that retains the independence of public and private keys remains a challenge and an opportunity for future work. For this however, we have proposed a rigorous catalogue of well-defined properties, that prospective cryptosystems and signature schemes may fulfill, to be suitable for their application in prospective mA^{3}PAKE post-quantum variants.

8.4 Final Thoughts

In an era where data breaches and cyber threats are rampant, the significance of robust authentication mechanisms cannot be overstated. This thesis not only sheds light on the current state of affairs but also charts a path forward, pushing the boundaries of what's possible in password-based authentication. As technology continues to evolve, so will the threats we face. But with innovative approaches like the one proposed in this thesis, we can stay a step ahead, ensuring a safer digital future for all.

A Drunken Bishop (OpenSSH Fingerprint Visualization) Algorithm Implementation in JavaScript

JavaScript implementation of the generateFingerprintRandomart(hash) function

```
/**
 * This is a randomart generator that gets a hash value as input
 * and generates a "random" looking ascii art image,
 * that can be used as a "fingerprint",
 * e.g. for validating server public keys, as for initial SSH
     sessions.
 * @param {string} input - The input hash value to generate the
     fingerprint from.
 * @returns {string} - The generated fingerprint as an ASCII art
     image.
 */
const XLIM = 17; // X-axis limit
const YLIM = 9; // Y-axis limit
const ARSZ = XLIM * YLIM; // Array size
// Array of symbols used to generate the fingerprint
const symbols = [' ', '.', 'o', '+', '=', '*', 'B', 'O', 'X',
                       '@', '%', '&', '#', '/', '^', 'S', 'E'];
let array = new Array(ARSZ).fill(0); // Initialize an array of size
    ARSZ with 0s
// Generates the ASCII art image by printing the symbols in the array
function printGraph() {
    let result = "+--[ RandomArt ]--+\n";
    for (let i = 0; i < YLIM; i++) {
        result += "|";
        for (let j = 0; j < XLIM; j++) {
            result += symbols[array[j + XLIM * i]];
        }
        result += "|\n";
    }
    result += "+-----------------+\n";
    return result;
}
// Checks if a character is a hexadecimal character
function isHex(c) {
    const hexRegex = /^[0-9a-fA-F]$/;
    return hexRegex.test(c);
}
// Convert hexadecimal character to its decimal value
```

```
35 function hexVal(c) {
36     return parseInt(c, 16);
37 }
38 // Convert hex string to an array of decimal values
39 function convertString(arg) {
40     let string = [];
41     for (let i = 0; i < arg.length; i += 2) {
42         if (!isHex(arg[i]) || !isHex(arg[i + 1])) {
43             throw new Error("Unrecognized character");
44         }
45         string.push((hexVal(arg[i]) << 4) | hexVal(arg[i + 1]));
46     }
47     return string;
48 }
49 // Calculate the new position of the drunken walk
50 // depending on the current position and direction
51 function new_position(pos, direction) {
52     let x0 = pos % XLIM;
53     let y0 = Math.floor(pos / XLIM);
54     let x1, y1;
55     switch (direction) {
56         case 0: x1 = x0 - 1; y1 = y0 - 1; break; // NW
57         case 1: x1 = x0 + 1; y1 = y0 - 1; break; // NE
58         case 2: x1 = x0 - 1; y1 = y0 + 1; break; // SW
59         case 3: x1 = x0 + 1; y1 = y0 + 1; break; // SE
60         default: x1 = x0; y1 = y0; break;
61     }
62     x1 = Math.max(0, Math.min(XLIM - 1, x1));
63     y1 = Math.max(0, Math.min(YLIM - 1, y1));
64     return y1 * XLIM + x1;
65 }
66 // Perform the drunken walk guided by the input (hex) string
67 // and increment the array at the new position
68 function drunken_walk(inputString) {
69     let pos = 76; // Starting position
70     const string = convertString(inputString);
71     for (let idx = 0; idx < string.length; idx++) {
72         let temp = string[idx];
73         for (let i = 0; i < 4; i++) {
74             const newPos = new_position(pos, temp & 3);
75             if (newPos !== pos) {
76                 array[newPos]++;
77             }
78             pos = newPos;
79             temp >>= 2;
80         }
81     }
82     array[pos] = 16; // End ('E')
83     array[76] = 15; // Start ('S')
```

```
84 }
85 // generate the fingerprint of a hash value (hex string)
86 // and print it as a random ASCII art image
87 function generateFingerprintRandomart(hash) {
88     drunken_walk(hash);
89     return printGraph();
90 }
91
92 module.exports = generateFingerprintRandomart;
```

B Probabilistic Prime Factor Recovery Algorithm Implementation in JavaScript

The Prime Factor Recovery algorithm implemented in this work is based on the pseudo-code algorithm described in the NIST SP 800-56B: Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography (August 2009), specifically in Appendix C: Prime Factor Recovery (Normative) [Bar+09].

Given both the public and the private exponent of a RSA key-pair, this probabilistic algorithm recovers the prime factors of their modulus.

In NIST SP 800-56B, the algorithm is being attributed to Fact 1 in Boneh's *Twenty years of attacks on the RSA cryptosystem* paper (1999) [Bon99], which provides a verbal explanation of the algorithm. However, it turns out, that a similarly detailed verbal description of the algorithm could already be found in the *Handbook of Applied Cryptography* (1996) [MVV96] and implied knowledge of it could already be observed in Rabin's "Oblivious Transfer" paper (1981) [Rab81], in which the ϕ-function derived square root of a random cyclic group member in the RSA modulus is being used to recover its prime factors with an oblivious probability of 1/2.

Even though, the NIST authors mention that their algorithm "bears some resemblance to the Miller-Rabin primality testing algorithm" published in [Rab80], out of the aforementioned works, only their publication provides a complete algorithm in pseudo-code, which can be implemented almost effortlessly in any common programming language supporting basic modular arithmetic operations on integers of RSA-sufficient length.

The following JavaScript code is an almost exact implementation of their Prime Factor Recovery algorithm, and is being used as such in the RSA *key-pair generator recovery* module for the re-registration functionality of the proposed mA3PAKE protocol. Like the other JavaScript modules and scripts implemented for this work, also this module uses the `node-forge` package and its `BigInteger` type for the necessary modular arithmetic operations on RSA-size integers.

```javascript
const forge = require('node-forge');
const BigInteger = forge.jsbn.BigInteger;
const ZERO = BigInteger.ZERO;
const ONE = BigInteger.ONE;
const TWO = new BigInteger("2");

function recoverPQ(n, e, d) {
    // Step 1: Let k = de - 1.
    // If k is odd, then go to the error handling at the end
    const k = d.multiply(e).subtract(ONE);
    if (k.isEven()) {
        let r = k.clone();
        let t = ZERO;
        // Step 2: Write k as k = 2^t r,
        // where r is the largest odd integer dividing k, and t >= 1.
        while (r.isEven()) {
            r = r.divide(TWO);
            t = t.add(ONE);
        }
        // Step 3: For i = 1 to 100 do:
        for (let i = 1; i <= 100; i++) {
            // 3a: Generate a random integer g in the range [0, n-1].
            let randomBytes = forge.random.getBytesSync(n.bitLength() / 8);
            let hexString = forge.util.bytesToHex(randomBytes);
            let g = new BigInteger(hexString, 16);
            // 3b: Let y = g^r mod n.
            let y = g.modPow(r, n);
            // 3c: If y = 1 or y = n-1, then continue to the next iteration.
            if (y.equals(ONE) || y.equals(n.subtract(ONE))) {
                continue;
            }
            // 3d: For j = 1 to t-1 do:
            for (let j = 0; j < t; j++) {
                let x = y.modPow(TWO, n); // 3d I / 3e
                // 3d II / 3f:
                // If x = 1, then the prime factors have been found.
                if (x.equals(ONE)) {
                    let p = y.subtract(ONE).gcd(n);
                    let q = n.divide(p);
                    return [p, q];
                }
                // 3d III:
                // If x = n-1, then break and go to the next iteration.
                if (x.equals(n.subtract(ONE))) {
                    break;
                }
                y = x; // 3d IV
            }
        }
    }
    // If the function hasn't returned by this point, throw an error.
    throw new Error("Cannot compute P and Q");
}

module.exports = recoverPQ;
```

Listing 9: JavaScript implementation of the recoverPQ(n, e, d) function

C OPRF Client Demo Script

```
1  const oprfClient = require('./oprfClient.js');
2
3  const oprfServer = "oprf.reich.org";
4  const oprfID = "4lphaNumT3stId1234";
5  const password = "password123";
6  const encoding = "UTF-8";
7
8  const oprfURL = "https://" + oprfServer;
9
10 let passseed;
11
12 async function run() {
13   try {
14     console.log(`Trying to execute oprfClient("${oprfURL}",
15       "${oprfID}", "${password}", "${encoding}") ...`);
16     passseed = await oprfClient(oprfURL, oprfID, password, encoding);
17     console.log("oprfClient() executed successfully.");
18     console.log("Your OPRF-salted 256 bit password hash (hex) is:");
19     console.log(Buffer.from(passseed).toString('hex'));
20     console.log("You can now use this secretly salted password hash as a seed","
21       (passseed) for whatever you want to do with it.");
22     console.log(`${oprfServer} will store your client ID (${oprfID}) `
23       + "together with the (randomly selected) secret salt key for future
         ↪  requests.");
24     console.log("Note: same password + same client ID = same passseed",
25       "(Well, most of the times).");
26   } catch (error) { console.error("An error occurred:", error); }
27 }
28
29 run();
```

Listing 10: oprfClientDemo.js

References for Books and Standards

[MVV96] Alfred J Menezes, Paul C Van Oorschot und Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996.

[Bar+09] Elaine Barker u. a. *Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography*. NIST Special Publication 800-56B. National Institute of Standards und Technology, Aug. 2009.

[BMS20] Colin Boyd, Anish Mathuria und Douglas Stebila. *Protocols for Authentication and Key Establishment*. 2. Aufl. Information Security and Cryptography. Originally published under: Boyd C. and Mathuria A. Springer Berlin, Heidelberg, 2020. ISBN: 978-3-662-58146-9. DOI: 10.1007/978-3-662-58146-9.

[Won21] David Wong. *Real-World Cryptography*. Manning, Sep. 2021. ISBN: 978-1-617-29671-0.

[Poi22] David Pointcheval. *Asymmetric Cryptography: Primitives and Protocols*. Wiley-ISTE, 2022. ISBN: 978-1-394-18835-2.

[BS23] Dan Boneh und Victor Shoup. *A Graduate Course in Applied Cryptography*. Version 0.6. Jan. 2023. URL: https://toc.cryptobook.us/ (besucht am 14. 11. 2023).

References for Online Resources

[Ros+05] B. Ross u. a. *Stronger Password Authentication Using Browser Extensions*. Online; accessed 2023-01-31. Stanford University. 2005. URL: http://crypto.stanford.edu/~dabo/pubs/abstracts/pwdhash.html.

[On15] Christophe Liou Kee On. *pwdhash extension for google chrome browser*. Online; accessed 2023-01-31. Github. 2015. URL: https://github.com/infocris/pwdhash.

[16] *Passwords 2016*. Online; accessed 2023-01-31. Ruhr-Universität Bochum. 2016. URL: https://passwords2016.rub.de/.

[Jar+17a] Stanislaw Jarecki u. a. *How to Strengthen Password and Multi-Factor Authentication; Is password Insecurity Inevitable?* Online; accessed 2023-01-31. Stronger Security for Password Authentication - Webinar Series. 2017. URL: https://hdl.handle.net/2022/21639.

[Kra17] Hugo Krawczyk. *Hugo Krawczyk RWC 2017*. Online; accessed 2023-01-31. Youtube. 2017. URL: https://www.youtube.com/watch?v=px8hiyf81iM.

[aut18] unknown author. *Keeping your account secure*. Online; accessed 2023-11-08. Twitter Inc. 3. Mai 2018. URL: https://blog.twitter.com/official/en_us/topics/company/2018/keeping-your-account-secure.html.

[Gar18] Chaim Gartenberg. *Twitter advising all 330 million users to change passwords after bug exposed them in plain text*. Online; accessed 2023-11-08. The Verge. 3. Mai 2018. URL: https://www.theverge.com/2018/5/3/17316684/twitter-password-bug-security-flaw-exposed-change-now.

[Gre18] Matthew Green. *Let's talk about PAKE*. Online; accessed 2023-11-08. A Few Thoughts on Cryptographic Engineering (Blog). 19. Okt. 2018. URL: https://blog.cryptographyengineering.com/2018/10/19/lets-talk-about-pake/.

[Mar18] Stefan Marsiske. *WebSphinx chrome/opera addon: Chrom* Addon for Sphinx-based Password Storage*. [Accessed 31-Jan-2023]. Github. 2018. URL: https://github.com/stef/websphinx-chrom.

[Pal18] Wladimir Palant. *Should your next web-based login form avoid sending passwords in clear text?* Okt. 2018. URL: https://palant.info/2018/10/25/should-your-next-web-based-login-form-avoid-sending-passwords-in-clear-text/ (besucht am 08. 11. 2023).

[Jar21] Stanislaw Jarecki. *KHAPE: Asymmetric PAKE from Key-Hiding Key Exchange*. 2021. URL: https://www.youtube.com/watch?v=qAVEaNnFlNU.

[Mad21] Neil Madden. *From KEMs to protocols*. Apr. 2021. URL: https://neilmadden.blog/2021/04/08/from-kems-to-protocols/ (besucht am 16. 07. 2023).

[Mar22] Stefan Marsiske. *The Extended Sphinx Protocol*. [Accessed 31-Jan-2023]. Github. 2022. URL: https://github.com/stef/pwdsphinx/blob/master/whitepaper.org.

[Arg23] Ionut Arghire. *Popular WordPress Security Plugin Caught Logging Plaintext Passwords*. 2023. URL: https://www.securityweek.com/popular-wordpress-security-plugin-caught-logging-plaintext-passwords/ (besucht am 16. 07. 2023).

[Wu23] Tom Wu. *The Stanford SRP Homepage*. Online; accessed 2023-01-31. Stanford University. 2023. URL: http://srp.stanford.edu.

[mul] multiparty.org. *multiparty/oprf - Oblivious pseudorandom function over an elliptic curve*. [Accessed 31-Jan-2023]. Github. URL: https://github.com/multiparty/oprf.

References for Academic Publications and Journal Articles

[Mil75] Gary L. Miller. „Riemann's Hypothesis and Tests for Primality". In: STOC '75. Albuquerque, New Mexico, USA: Association for Computing Machinery, 1975, S. 234–239. ISBN: 9781450374194. DOI: 10.1145/800116.803773. URL: https://doi.org/10.1145/800116.803773.

[DH76] W. Diffie und M. Hellman. „New directions in cryptography". In: *IEEE Trans. Inform. Theory IT-22* 6 (1976), S. 472–492.

[Mer78] Ralph Merkle. „Secure Communication Over Insecure Channels". In: *Commun. ACM* 21 (Apr. 1978), S. 294–299. DOI: 10.1145/359460.359473.

[RSA78] Ronald L. Rivest, Adi Shamir und Leonard M. Adleman. „A method for obtaining digital signatures and public-key cryptosystems". In: *Commun. ACM* 21 (1978), S. 120–126. URL: https://api.semanticscholar.org/CorpusID:2873616.

[Lam79] Leslie Lamport. „Constructing Digital Signatures from a One Way Function". In: CSL-98. This paper was published by IEEE in the Proceedings of HICSS-43 in January, 2010. 1979. URL: https://www.microsoft.com/en-us/research/publication/constructing-digital-signatures-one-way-function/.

[Mer79] Ralph Charles Merkle. „Secrecy, Authentication, and Public Key Systems." AAI8001972. Diss. Stanford, CA, USA, 1979. DOI: 10.5555/909000.

[Rab79] M. O. Rabin. *DIGITALIZED SIGNATURES AND PUBLIC-KEY FUNCTIONS AS INTRACTABLE AS FACTORIZATION*. Techn. Ber. USA, 1979.

[Rab80] Michael O Rabin. „Probabilistic algorithm for testing primality". In: *Journal of Number Theory* 12.1 (1980), S. 128–138. ISSN: 0022-314X. DOI: https://doi.org/10.1016/0022-314X(80)90084-0. URL: https://www.sciencedirect.com/science/article/pii/0022314X80900840.

[Lam81] Leslie Lamport. „Password authentication with insecure communication". In: *Communications of the ACM* 24.11 (1981), S. 770–772.

[Rab81] Michael O. Rabin. *How to exchange secrets with oblivious transfer*. Techn. Ber. Aiken Computation Laboratory, Harvard University, 1981.

[Yao82] Andrew C. Yao. „Theory and application of trapdoor functions". In: *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*. 1982, S. 80–91. DOI: 10.1109/SFCS.1982.45.

[VK83] Victor L. Voydock und Stephen T. Kent. „Security Mechanisms in High-Level Network Protocols". In: *ACM Comput. Surv.* 15.2 (Juni 1983), S. 135–171. ISSN: 0360-0300. DOI: 10.1145/356909.356913. URL: https://doi.org/10.1145/356909.356913.

[GM84] Shafi Goldwasser und Silvio Micali. „Probabilistic Encryption". In: *J. Comput. Syst. Sci.* 28 (1984), S. 270–299. URL: https://api.semanticscholar.org/CorpusID:19616020.

[Sha84] Adi Shamir. „Identity-Based Cryptosystems and Signature Schemes“. In: *Annual International Cryptology Conference*. 1984. URL: https://api.semanticscholar.org/CorpusID:1402295.

[ElG85] Taher ElGamal. „A public key cryptosystem and a signature scheme based on discrete logarithms“. In: *IEEE transactions on information theory* 31.4 (1985), S. 469–472.

[FS87] Amos Fiat und Adi Shamir. „How to Prove Yourself: Practical Solutions to Identification and Signature Problems“. In: *Proceedings on Advances in Cryptology—CRYPTO '86*. Santa Barbara, California, USA: Springer-Verlag, 1987, S. 186–194. ISBN: 0387180478.

[LR87] Michael Luby und Charles Rackoff. „A study of password security“. In: *Journal of Cryptology* 1 (1987), S. 151–158. URL: https://api.semanticscholar.org/CorpusID:31435251.

[Mer87] Ralph Merkle. „A Digital Signature Based on a Conventional Encryption Function“. In: Bd. 293. Aug. 1987, S. 369–378. ISBN: 978-3-540-18796-7. DOI: 10.1007/3-540-48184-2_32.

[BM88] Mihir Bellare und Silvio Micali. „How to sign given any trapdoor function“. In: *Symposium on the Theory of Computing*. 1988. URL: https://api.semanticscholar.org/CorpusID:12212891.

[GMR88] Shafi Goldwasser, Silvio Micali und Ronald L. Rivest. „A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks“. In: *SIAM J. Comput.* 17 (1988), S. 281–308. URL: https://api.semanticscholar.org/CorpusID:1715998.

[Lom+89a] T. Lomas u. a. „Reducing Risks from Poorly Chosen Keys“. In: *Proceedings of the Twelfth ACM Symposium on Operating Systems Principles*. SOSP '89. New York, NY, USA: Association for Computing Machinery, 1989, S. 14–18. ISBN: 0897913388. DOI: 10.1145/74850.74853. URL: https://doi.org/10.1145/74850.74853.

[Lom+89b] T. Lomas u. a. „Reducing Risks from Poorly Chosen Keys“. In: *SIGOPS Oper. Syst. Rev.* 23.5 (Nov. 1989), S. 14–18. ISSN: 0163-5980. DOI: 10.1145/74851.74853. URL: https://doi.org/10.1145/74851.74853.

[Dif90] W. Diffie. „Authenticated key exchange and secure interactive communications“. In: *Securicom 90: 8th Worldwide Congress on Computer and Communications Security and Protection : Papers*. SEDEP, 1990. URL: https://books.google.de/books?id=NWkGzQEACAAJ.

[BM92] Steven Michael Bellovin und Michael Merritt. „Encrypted key exchange: Password-based protocols secure against dictionary attacks“. In: (1992).

[Sim92] William A. Simpson. *PPP Authentication Protocols*. RFC 1334. Okt. 1992. DOI: 10.17487/RFC1334. URL: https://www.rfc-editor.org/info/rfc1334.

[BM93] Steven M. Bellovin und Michael Merritt. „Augmented Encrypted Key Exchange: A Password-Based Protocol Secure against Dictionary Attacks and Password File Compromise“. In: *Proceedings of the 1st ACM Conference on Computer and Communications Security*. CCS '93. Fairfax, Virginia, USA: Association for Computing Machinery, 1993, S. 244–250. ISBN: 0897916298.

DOI: 10.1145/168588.168618. URL: https://doi.org/10.1145/168588.168618.

[Gon+93] L. Gong u. a. „Protecting poorly chosen secrets from guessing attacks“. In: *IEEE Journal on Selected Areas in Communications* 11.5 (1993), S. 648–656. DOI: 10.1109/49.223865.

[AND94] T. M. A ANDERSON R. J; LOMAS. „Fortifying key negotiation schemes with poorly chosen passwords“. In: *Electronics Letters* (1994). ISSN: 0013-5194.

[BR95] Mihir Bellare und Phillip Rogaway. „Optimal asymmetric encryption“. In: *Advances in Cryptology — EUROCRYPT'94*. Hrsg. von Alfredo De Santis. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, S. 92–111. ISBN: 978-3-540-44717-7. DOI: 10.1007/BFb0053428.

[Gan95] Ravi Ganesan. „Yaksha: augmenting Kerberos with public key cryptography“. In: *Proceedings of the Symposium on Network and Distributed System Security* (1995), S. 132–143. URL: https://api.semanticscholar.org/CorpusID:16807796.

[Gon95] Li Gong. „Optimal Authentication Protocols Resistant to Password Guessing Attacks“. In: *Proceedings of the 8th IEEE Workshop on Computer Security Foundations*. CSFW '95. USA: IEEE Computer Society, 1995, S. 24. ISBN: 0818670339.

[STW95] Michael Steiner, Gene Tsudik und Michael Waidner. „Refinement and Extension of Encrypted Key Exchange“. In: *SIGOPS Oper. Syst. Rev.* 29.3 (Juli 1995), S. 22–30. ISSN: 0163-5980. DOI: 10.1145/206826.206834. URL: https://doi.org/10.1145/206826.206834.

[BR96] Mihir Bellare und Phillip Rogaway. „The Exact Security of Digital Signatures - HOw to Sign with RSA and Rabin“. In: *Advances in Cryptology - EUROCRYPT '96*. Bd. 1070. Lecture Notes in Computer Science. Springer, 1996, S. 399–416. DOI: 10.1007/3-540-68339-9_34.

[FMR96] Michael J. Fischer, Silvio Micali und Charles Rackoff. „A Secure Protocol for the Oblivious Transfer (Extended Abstract)“. In: *J. Cryptology* 9 (1996). written & presented in 1984, S. 191–195.

[Gan96] Ravi Ganesan. „The Yaksha security system“. In: *Commun. ACM* 39 (1996), S. 55–60.

[Jab96] David P. Jablon. „Strong Password-Only Authenticated Key Exchange“. In: *SIGCOMM Comput. Commun. Rev.* 26.5 (Okt. 1996), S. 5–26. ISSN: 0146-4833. DOI: 10.1145/242896.242897. URL: https://doi.org/10.1145/242896.242897.

[Jab97] David P. Jablon. „Extended password key exchange protocols immune to dictionary attack“. In: *Proceedings of IEEE 6th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises* (1997), S. 248–255. URL: https://api.semanticscholar.org/CorpusID:10568917.

[BCK98] Mihir Bellare, Ran Canetti und Hugo Krawczyk. *A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols*. Cryptology ePrint Archive, Paper 1998/009. https://eprint.iacr.org/1998/009. 1998. URL: https://eprint.iacr.org/1998/009.

[Bel+98] Mihir Bellare u. a. „Relations among notions of security for public-key encryption schemes“. In: *Advances in Cryptology — CRYPTO '98*. Hrsg. von Hugo Krawczyk. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, S. 26–45. ISBN: 978-3-540-68462-6.

[CRW98] B. Christianson, M. Roe und D. Wheeler. *Secure Sessions from Weak Secrets*. Techn. Ber. Cambridge, UK: University of Cambridge und University of Hertfordshire, 1998. DOI: 10.1007/11542322_25.

[HZI98] Goichiro Hanaoka, Yuliang Zheng und Hideki Imai. „LITESET: A lightweight secure electronic transaction protocol“. In: Bd. 1438. Juli 1998, S. 215–226. ISBN: 978-3-540-64732-4. DOI: 10.1007/BFb0053735.

[Wu+98] Thomas D Wu u. a. „The Secure Remote Password Protocol.“ In: *NDSS*. Bd. 98. Citeseer. 1998, S. 97–111.

[ZI98] Yuliang Zheng und Hideki Imai. „How to Construct Efficient Signcryption Schemes on Elliptic Curves“. In: *Inf. Process. Lett.* 68 (1998), S. 227–233. URL: https://api.semanticscholar.org/CorpusID:14146129.

[Bon99] D. Boneh. „Twenty Years of Attacks on the RSA Cryptosystem“. In: *Notices of the AMS* 46.2 (1999), S. 203–213.

[Boy99] Maurizio Kliban Boyarsky. „Public-Key Cryptography and Password Protocols: The Multi-User Case“. In: *Proceedings of the 6th ACM Conference on Computer and Communications Security*. CCS '99. Kent Ridge Digital Labs, Singapore: Association for Computing Machinery, 1999, S. 63–72. ISBN: 1581131488. DOI: 10.1145/319709.319719. URL: https://doi.org/10.1145/319709.319719.

[HK99] Shai Halevi und Hugo Krawczyk. „Public-Key Cryptography and Password Protocols“. In: *ACM Trans. Inf. Syst. Secur.* 2.3 (Aug. 1999), S. 230–268. ISSN: 1094-9224. DOI: 10.1145/322510.322514. URL: https://doi.org/10.1145/322510.322514.

[MS99] Philip D. MacKenzie und R. P. Swaminathan. „Secure Network Authentication with Password Identification“. In: *submission to IEEE P1363*. 1999. URL: https://api.semanticscholar.org/CorpusID:17307854.

[PS99] Adrian Perrig und Dawn Xiaodong Song. „Hash Visualization: a New Technique to improve Real-World Security“. In: 1999. URL: https://api.semanticscholar.org/CorpusID:18240557.

[SMW99] Bruce Schneier, Mudge und David A. Wagner. „Cryptanalysis of Microsoft's PPTP Authentication Extensions (MS-CHAPv2)“. In: *International Exhibition and Congress on Network Security*. 1999. URL: https://api.semanticscholar.org/CorpusID:247846.

[Sho99] Victor Shoup. *On Formal Models for Secure Key Exchange*. Cryptology ePrint Archive, Paper 1999/012. https://eprint.iacr.org/1999/012. 1999. URL: https://eprint.iacr.org/1999/012.

[BPR00] Mihir Bellare, David Pointcheval und Phillip Rogaway. „Authenticated Key Exchange Secure against Dictionary Attacks“. In: *IACR Cryptol. ePrint Arch.* 2000 (2000), S. 14. URL: https://api.semanticscholar.org/CorpusID:1044118.

[BMP00] Victor Boyko, Philip Mackenzie und Sarvar Patel. „Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman“. In: Bd. 1807. Juli 2000. ISBN: 978-3-540-67517-4. DOI: 10.1007/3-540-45539-6_12.

[GL00] Oded Goldreich und Yehuda Lindell. *Session-Key Generation using Human Passwords Only*. Cryptology ePrint Archive, Paper 2000/057. https://eprint.iacr.org/2000/057. 2000. URL: https://eprint.iacr.org/2000/057.

[MPS00] Philip D. MacKenzie, Sarvar Patel und R. P. Swaminathan. „Password-authenticated key exchange based on RSA“. In: *International Journal of Information Security* 9 (2000), S. 387–410. URL: https://api.semanticscholar.org/CorpusID:12481879.

[BF01] Dan Boneh und Matt Franklin. „Identity-Based Encryption from the Weil Pairing“. In: Bd. 32. Aug. 2001, S. 213–229. ISBN: 978-3-540-42456-7. DOI: 10.1007/3-540-44647-8_13.

[BLS01] Dan Boneh, Ben Lynn und Hovav Shacham. „Short Signatures from the Weil Pairing“. In: *Journal of Cryptology* 17 (2001), S. 297–319. URL: https://api.semanticscholar.org/CorpusID:929219.

[BMN01] Colin Boyd, Paul Montague und Khanh Nguyen. „Elliptic Curve Based Password Authenticated Key Exchange Protocols“. In: *LNCS*. Bd. 2119. Springer-Verlag. 2001, S. 487–501.

[Can01] R. Canetti. „Universally Composable Security: A New Paradigm for Cryptographic Protocols“. In: Bd. 2000. Nov. 2001, S. 136–145. ISBN: 0-7695-1116-3. DOI: 10.1109/SFCS.2001.959888.

[CK01] Ran Canetti und Hugo Krawczyk. „Analysis of key-exchange protocols and their use for building secure channels“. In: *International conference on the theory and applications of cryptographic techniques*. Springer. 2001, S. 453–474.

[KOY01] Jonathan Katz, Rafail Ostrovsky und Moti Yung. *Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords*. Cryptology ePrint Archive, Paper 2001/031. https://eprint.iacr.org/2001/031. 2001. URL: https://eprint.iacr.org/2001/031.

[KP01] Charlie Kaufman und Radia Perlman. „PDM: A New Strong Password-Based Protocol“. In: *Proceedings of the 10th Conference on USENIX Security Symposium - Volume 10*. SSYM'01. Washington, D.C.: USENIX Association, 2001, S. 23.

[Kwo01] Taekyoung Kwon. „Authentication and Key Agreement Via Memorable Passwords“. In: *Network and Distributed System Security Symposium*. 2001. URL: https://api.semanticscholar.org/CorpusID:45673908.

[Zho01] Jianying Zhou. „Achieving Fair Nonrepudiation in Electronic Transactions“. In: *Journal of Organizational Computing and Electronic Commerce* 11.4 (2001), S. 253–267. DOI: 10.1207/S15327744JOCE1104_03.

[GK02] Zvi Galil und Jonathan Katz. „Efficient cryptographic protocols preventing man-in-the-middle attacks“. In: 2002. URL: https://api.semanticscholar.org/CorpusID:1170933.

[Mac02] Philip Mackenzie. „The PAK suite: Protocols for Password-Authenticated Key Exchange“. In: (Dez. 2002).

[Can+03] Brice Canvel u. a. „Password interception in a SSL/TLS channel“. In: *Lecture Notes in Computer Science* 2729 (Aug. 2003), S. 583–599. DOI: 10.1007/b11817.

[CS03] Claude Crépeau und Alain Slakmon. „Simple Backdoors for RSA Key Generation“. In: *Proceedings of the 2003 RSA Conference on The Cryptographers' Track*. CT-RSA'03. San Francisco, CA, USA: Springer-Verlag, 2003, S. 403–416. ISBN: 3540008470.

[GL03] Rosario Gennaro und Yehuda Lindell. *A Framework for Password-Based Authenticated Key Exchange*. Cryptology ePrint Archive, Paper 2003/032. https://eprint.iacr.org/2003/032. 2003. URL: https://eprint.iacr.org/2003/032.

[Hit+03] Yvonne Hitchcock u. a. „A Password-Based Authenticator: Security Proof and Applications“. In: *Progress in Cryptology - INDOCRYPT 2003*. Hrsg. von Thomas Johansson und Subhamoy Maitra. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, S. 388–401. ISBN: 978-3-540-24582-7.

[MM03] John Malone-Lee und Wenbo Mao. „Two Birds One Stone: Signcryption Using RSA“. In: *The Cryptographer's Track at RSA Conference*. 2003. URL: https://api.semanticscholar.org/CorpusID:18139837.

[AFP04] Michel Abdalla, Pierre-Alain Fouque und David Pointcheval. *Password-Based Authenticated Key Exchange in the Three-Party Setting*. Cryptology ePrint Archive, Paper 2004/233. https://eprint.iacr.org/2004/233. 2004. URL: https://eprint.iacr.org/2004/233.

[JG04] Shaoquan Jiang und Guang Gong. *Password Based Key Exchange with Mutual Authentication*. Cryptology ePrint Archive, Paper 2004/196. https://eprint.iacr.org/2004/196. 2004. URL: https://eprint.iacr.org/2004/196.

[ZSS04] Fangguo Zhang, Reihaneh Safavi-Naini und Willy Susilo. „An Efficient Signature Scheme from Bilinear Pairings and Its Applications“. In: *International Conference on Theory and Practice of Public Key Cryptography*. 2004. URL: https://api.semanticscholar.org/CorpusID:12304471.

[AP05] Michel Abdalla und David Pointcheval. „Simple Password-Based Encrypted Key Exchange Protocols“. In: Bd. 3376. Feb. 2005, S. 191–208. ISBN: 978-3-540-24399-1. DOI: 10.1007/978-3-540-30574-3_14.

[Can+05] Ran Canetti u. a. *Universally Composable Password-Based Key Exchange*. Cryptology ePrint Archive, Paper 2005/196. https://eprint.iacr.org/2005/196. 2005. URL: https://eprint.iacr.org/2005/196.

[Das+06] Manik Lal Das u. a. „A novel remote user authentication scheme using bilinear pairings“. In: *Computers & Security* 25.3 (2006), S. 184–189. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2005.09.002.

[GL06] Rosario Gennaro und Yehuda Lindell. „A Framework for Password-Based Authenticated Key Exchange“. In: *ACM Trans. Inf. Syst. Secur.* 9.2 (Mai 2006), S. 181–234. ISSN: 1094-9224. DOI: 10.1145/1151414.1151418. URL: https://doi.org/10.1145/1151414.1151418.

[GMR06] Craig Gentry, Philip MacKenzie und Zulfikar Ramzan. „A method for making password-based key exchange resilient to server compromise“. In: *Annual International Cryptology Conference*. Springer. 2006, S. 142–159.

[MSJ06] Philip Mackenzie, Thomas Shrimpton und Markus Jakobsson. „Threshold Password-Authenticated Key Exchange“. In: *Journal of Cryptology* 19 (Aug. 2006), S. 27–66. DOI: 10.1007/s00145-005-0232-5.

[Abd+08] Michel Abdalla u. a. „Efficient Two-Party Password-Based Key Exchange Protocols in the UC Framework“. In: (Jan. 2008). DOI: 10.1007/978-3-540-79263-5_22.

[CKS08] David Cash, Eike Kiltz und Victor Shoup. „The Twin Diffie-Hellman Problem and Applications“. In: *Advances in Cryptology – EUROCRYPT 2008*. Hrsg. von Nigel Smart. Berlin, Heidelberg: Springer-Verlag, 2008, S. 127–145. ISBN: 978-3-540-78967-3.

[Boy09] Xavier Boyen. „Hidden Credential Retrieval from a Reusable Password“. In: *ACM Symposium on Information, Computer & Communication Security—ASIACCS 2009*. New-York: ACM Press, 2009, S. 228–238. URL: http://www.cs.stanford.edu/~xb/asiaccs09/.

[Eng+09] Jack Engler u. a. „Is it too late for PAKE ?“ In: 2009. URL: https://api.semanticscholar.org/CorpusID:12990078.

[LLG09] Dirk Loss, Tobias Limmer und Alexander von Gernler. „The drunken bishop: An analysis of the OpenSSH fingerprint visualization algorithm“. In: (2009).

[OWT09] Yutaka Oiwa, Hajime Watanabe und Hiromitsu Takagi. „PAKE-based mutual HTTP authentication for preventing phishing attacks“. In: *The Web Conference*. 2009. URL: https://api.semanticscholar.org/CorpusID:18724238.

[LW10] Yi-Pin Liao und Shuenn-Shyang Wang. „A robust password-based remote user authentication scheme using bilinear pairings without using smart cards“. In: *2010 International Computer Symposium (ICS2010)*. 2010, S. 215–221. DOI: 10.1109/COMPSYM.2010.5685514.

[Bag+11] Ali Bagherzandi u. a. „Password-protected secret sharing“. In: *Proceedings of the 18th ACM conference on Computer and Communications Security*. 2011, S. 433–444.

[Fle+11] Nils Fleischhacker u. a. *Pseudorandom Signatures*. Cryptology ePrint Archive, Paper 2011/673. 2011. URL: https://eprint.iacr.org/2011/673.

[Yao11] Gang Yao. „A Three-Party Password Authenticated Key Exchange Protocol with Key Confirmation“. In: (2011).

[Bar12] Alexandru G. Bardas. „A Pake – Srp6 Browser Extension“. In: *Journal of Information Systems and Operations Management* 6 (2012), S. 244–257. URL: https://api.semanticscholar.org/CorpusID:50217856.

[GT12] Kristian Gjøsteen und Øystein Thuen. „Password-Based Signatures“. In: *Public Key Infrastructures, Services and Applications*. Hrsg. von Svetla Petkova-Nikova, Andreas Pashalidis und Günther Pernul. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, S. 17–33. ISBN: 978-3-642-29804-2.

[Len+12] Arjen Lenstra u. a. „Public Keys“. In: (Aug. 2012). DOI: 10.1007/978-3-642-32009-5_37.

[ABK13] Tolga Acar, Mira Belenkiy und Alptekin Küpçü. *Single Password Authentication*. Cryptology ePrint Archive, Paper 2013/167. https://eprint.iacr.org/2013/167. 2013. URL: https://eprint.iacr.org/2013/167.

[Gjø13] Kristian Gjøsteen. *Partially blind password-based signatures using elliptic curves*. Cryptology ePrint Archive, Paper 2013/472. 2013. URL: https://eprint.iacr.org/2013/472.

[Abd14] Michel Abdalla. „Password-Based Authenticated Key Exchange: An Overview“. In: *Provable Security*. Hrsg. von Sherman S. M. Chow u. a. Cham: Springer International Publishing, 2014, S. 1–9. ISBN: 978-3-319-12475-9.

[ABP14] Michel Abdalla, Fabrice Benhamouda und David Pointcheval. *Public-Key Encryption Indistinguishable Under Plaintext-Checkable Attacks*. Cryptology ePrint Archive, Paper 2014/609. A major revision of an IACR publication in PKC 2015. 2014. URL: https://eprint.iacr.org/2014/609.

[FR14] Roy T. Fielding und Julian Reschke. *Hypertext Transfer Protocol (HTTP/1.1): Authentication*. RFC 7235. Juni 2014. DOI: 10.17487/RFC7235. URL: https://www.rfc-editor.org/info/rfc7235.

[KM14] Franziskus Kiefer und Mark Manulis. „Distributed smooth projective hashing and its application to two-server password authenticated key exchange“. In: *Applied Cryptography and Network Security: 12th International Conference, ACNS 2014. Proceedings 12*. Springer. 2014, S. 199–216.

[MSD14] Mark Manulis, Douglas Stebila und Nick Denham. „Secure modular password authentication for the web using channel bindings“. In: *International Journal of Information Security* 15 (2014), S. 597–620. URL: https://api.semanticscholar.org/CorpusID:8392490.

[Cam+15] Jan Camenisch u. a. *Virtual Smart Cards: How to Sign with a Password and a Server*. Cryptology ePrint Archive, Paper 2015/1101. 2015. URL: https://eprint.iacr.org/2015/1101.

[Eve+15] Adam Everspaugh u. a. „The Pythia {PRF} Service“. In: *24th USENIX Security Symposium (USENIX Security 15)*. 2015, S. 547–562.

[Muf15] Alec Muffet. „Facebook: Password hashing & authentication“. In: *Presentation at Real World Crypto* (2015).

[SAB15] Rifaat Shekh-Yusef, David Ahrens und Sophie Bremer. *HTTP Digest Access Authentication*. RFC 7616. Sep. 2015. DOI: 10.17487/RFC7616. URL: https://www.rfc-editor.org/info/rfc7616.

[Din+16] Jintai Ding u. a. *Provably Secure Password Authenticated Key Exchange Based on RLWE for the Post-Quantum World*. Cryptology ePrint Archive, Paper 2016/552. 2016. URL: https://eprint.iacr.org/2016/552.

[Kie16] Franziskus Kiefer. „Advancements in password-based cryptography.“ In: 2016. URL: https://api.semanticscholar.org/CorpusID:31245715.

[KM16] Franziskus Kiefer und Mark Manulis. „Blind Password Registration for Verifier-Based PAKE“. In: *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography*. AsiaPKC '16. Xi'an, China: Association for Computing Machinery, 2016, S. 39–48. ISBN: 9781450342865. DOI: 10.1145/2898420.2898424. URL: https://doi.org/10.1145/2898420.2898424.

[RO16] P. Ranjan und H. Om. „An Efficient Remote User Password Authentication Scheme based on Rabin's Cryptosystem“. In: *Wireless Personal Communications 90* (2016), S. 217–244. URL: https://doi.org/10.1007/s11277-016-3342-5.

[BH17] Shaghayegh Bakhtiari-Chehelcheshmeh und Mehdi Hosseinzadeh. „A New Certificateless and Secure Authentication Scheme for Ad Hoc Networks“. In: *Wirel. Pers. Commun.* 94.4 (Juni 2017), S. 2833–2851. ISSN: 0929-6212. DOI: 10.1007/s11277-016-3721-y. URL: https://doi.org/10.1007/s11277-016-3721-y.

[Dup+17] Pierre-Alain Dupont u. a. *Fuzzy Password-Authenticated Key Exchange*. Cryptology ePrint Archive, Paper 2017/1111. 2017. URL: https://eprint.iacr.org/2017/1111.

[HL17] Björn Haase und Benoît Labrique. *Making Password Authenticated Key Exchange Suitable For Resource-Constrained Industrial Control Devices*. Cryptology ePrint Archive, Paper 2017/562. 2017. URL: https://eprint.iacr.org/2017/562.

[Hao17] Feng Hao. „J-PAKE: Password-Authenticated Key Exchange by Juggling“. In: *RFC* 8236 (2017), S. 1–15. URL: https://api.semanticscholar.org/CorpusID:1105374.

[İK17] Devriş İşler und Alptekin Küpçü. „Threshold Single Password Authentication“. In: *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Hrsg. von Joaquin Garcia-Alfaro u. a. Cham: Springer International Publishing, 2017, S. 143–162. ISBN: 978-3-319-67816-0.

[Jar+17b] Stanisław Jarecki u. a. „TOPPSS: Cost-Minimal Password-Protected Secret Sharing Based on Threshold OPRF“. In: *Applied Cryptography and Network Security*. Hrsg. von Dieter Gollmann, Atsuko Miyaji und Hiroaki Kikuchi. Cham: Springer International Publishing, 2017, S. 39–58.

[Lai+17] Russell WF Lai u. a. „Phoenix: Rebirth of a Cryptographic {Password-Hardening} Service“. In: *26th USENIX Security Symposium (USENIX Security 17)*. 2017, S. 899–916.

[Oiw+17] Yutaka Oiwa u. a. *Mutual Authentication Protocol for HTTP*. RFC 8120. Apr. 2017. DOI: 10.17487/RFC8120. URL: https://www.rfc-editor.org/info/rfc8120.

[Sch17] Jörn-Marc Schmidt. „Requirements for Password-Authenticated Key Agreement (PAKE) Schemes“. In: *RFC* 8125 (2017), S. 1–10. URL: https://api.semanticscholar.org/CorpusID:2216216.

[Agr+18] Shashank Agrawal u. a. „PASTA: password-based threshold authentication“. In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 2018, S. 2042–2059.

[Gao+18] Xinwei Gao u. a. „Post-Quantum Secure Remote Password Protocol from RLWE Problem“. In: Jan. 2018, S. 99–116. ISBN: 978-3-319-75159-7. DOI: 10.1007/978-3-319-75160-3_8.

[HL18] Björn Haase und Benoît Labrique. *AuCPace: Efficient verifier-based PAKE protocol tailored for the IIoT*. Cryptology ePrint Archive, Paper 2018/286. 2018. URL: https://eprint.iacr.org/2018/286.

[İK18] Devriş İşler und Alptekin Küpçü. *Distributed Single Password Protocol Framework*. Cryptology ePrint Archive, Paper 2018/976. 2018. URL: https://eprint.iacr.org/2018/976.

[İKC18] Devriş İşler, Alptekin Küpçü und Aykut Coskun. *User Study on Single Password Authentication*. Cryptology ePrint Archive, Paper 2018/975. https://eprint.iacr.org/2018/975. 2018. URL: https://eprint.iacr.org/2018/975.

[JKX18] Stanislaw Jarecki, Hugo Krawczyk und Jiayu Xu. *OPAQUE: An Asymmetric PAKE Protocol Secure Against Pre-Computation Attacks*. Cryptology ePrint Archive, Paper 2018/163. 2018. URL: https://eprint.iacr.org/2018/163.

[Lai+18] Russell WF Lai u. a. „Simple {Password-Hardened} Encryption Services". In: *27th USENIX Security Symposium (USENIX Security 18)*. 2018, S. 1405–1421.

[SL18] Marjan Skrobot und Jean Lancrenon. „On Composability of Game-Based Password Authenticated Key Exchange". In: *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. 2018, S. 443–457. DOI: 10.1109/EuroSP.2018.00038.

[Ata19] Latschesar Atanassov. „Online SPHINX: An Online Password Store that Is Resistant Against Offline Attacks". Master's Thesis. FH Campus Wien, 2019. URL: https://pub.fh-campuswien.ac.at/obvfcwhsacc/content/titleinfo/3431934/.

[Bec+19] Jose Becerra u. a. „An Offline Dictionary Attack Against zkPAKE Protocol". In: *ICT Systems Security and Privacy Protection*. Juni 2019, S. 81–90. ISBN: 978-3-030-22311-3. DOI: 10.1007/978-3-030-22312-0_6.

[BJX19] Tatiana Bradley, Stanislaw Jarecki und Jiayu Xu. *Strong Asymmetric PAKE based on Trapdoor CKEM*. Cryptology ePrint Archive, Paper 2019/647. 2019. URL: https://eprint.iacr.org/2019/647.

[Bra+19] Tatiana Bradley u. a. *Password-Authenticated Public-Key Encryption*. Cryptology ePrint Archive, Paper 2019/199. 2019. URL: https://eprint.iacr.org/2019/199.

[Liu+19] Chao Liu u. a. „Provably Secure Three-Party Password-Based Authenticated Key Exchange from RLWE". In: *Information Security Practice and Experience*. Hrsg. von Swee-Huay Heng und Javier Lopez. Cham: Springer International Publishing, 2019, S. 56–72. ISBN: 978-3-030-34339-2.

[Abd+20] Michel Abdalla u. a. *Universally Composable Relaxed Password Authenticated Key Exchange*. Cryptology ePrint Archive, Paper 2020/320. 2020. URL: https://eprint.iacr.org/2020/320.

[Bro+20a] Julian Brost u. a. *Threshold Password-Hardened Encryption Services*. Cryptology ePrint Archive, Paper 2020/1552. https://eprint.iacr.org/2020/1552. 2020. DOI: 10.1145/3372297.3417266. URL: https://eprint.iacr.org/2020/1552.

[Bro+20b] Julian Brost u. a. „Threshold Password-Hardened Encryption Services". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. CCS '20. Virtual Event, USA: Association for Computing Machinery, 2020, S. 409–424. ISBN: 9781450370899. DOI: 10.1145/3372297.3417266. URL: https://doi.org/10.1145/3372297.3417266.

[Erw+20] Andreas Erwig u. a. *Fuzzy Asymmetric Password-Authenticated Key Exchange*. Cryptology ePrint Archive, Paper 2020/987. 2020. URL: https://eprint.iacr.org/2020/987.

[LMR20] Kevin Lewi, Payman Mohassel und Arnab Roy. *Single-Message Credential-Hiding Login*. Cryptology ePrint Archive, Paper 2020/1509. https://eprint.iacr.org/2020/1509. 2020. URL: https://eprint.iacr.org/2020/1509.

[RJ20] Rachit Rawat und Mahabir Jhanwar. „PAS-TA-U: PASsword-Based Threshold Authentication with Password Update“. In: Dez. 2020, S. 25–45. ISBN: 978-3-030-66625-5. DOI: 10.1007/978-3-030-66626-2_2.

[Tar+20] Oleg Taraskin u. a. „Towards Isogeny-Based Password-Authenticated Key Establishment“. In: *Journal of Mathematical Cryptology* 15 (Nov. 2020), S. 18–30. DOI: 10.1515/jmc-2020-0071.

[BFS21] Daniel De Almeida Braga, Pierre-Alain Fouque und Mohamed Sabt. *PARASITE: PAssword Recovery Attack against Srp Implementations in ThE wild*. Cryptology ePrint Archive, Paper 2021/553. 2021. DOI: 10.1145/3460120.3484563. URL: https://eprint.iacr.org/2021/553.

[ES21] Edward Eaton und Douglas Stebila. *The "quantum annoying" property of password-authenticated key exchange protocols*. Cryptology ePrint Archive, Paper 2021/696. https://eprint.iacr.org/2021/696. 2021. URL: https://eprint.iacr.org/2021/696.

[GJK21] Yanqi Gu, Stanislaw Jarecki und Hugo Krawczyk. *KHAPE: Asymmetric PAKE from Key-Hiding Key Exchange*. Cryptology ePrint Archive, Paper 2021/873. https://eprint.iacr.org/2021/873. 2021. URL: https://eprint.iacr.org/2021/873.

[HO21] Feng Hao und Paul C. van Oorschot. *SoK: Password-Authenticated Key Exchange – Theory, Practice, Standardization and Real-World Lessons*. Cryptology ePrint Archive, Paper 2021/1492. https://eprint.iacr.org/2021/1492. 2021. URL: https://eprint.iacr.org/2021/1492.

[JKX21] Stanislaw Jarecki, Hugo Krawczyk und Jiayu Xu. *On the (In)Security of the Diffie-Hellman Oblivious PRF with Multiplicative Blinding*. Cryptology ePrint Archive, Paper 2021/273. 2021. URL: https://eprint.iacr.org/2021/273.

[Abd+22] Michel Abdalla u. a. *Password-Authenticated Key Exchange from Group Actions*. Cryptology ePrint Archive, Paper 2022/770. 2022. URL: https://eprint.iacr.org/2022/770.

[Ber+22] Csanád Bertók u. a. *Provably Secure Identity-Based Remote Password Registration*. Cryptology ePrint Archive, Paper 2022/286. https://eprint.iacr.org/2022/286. 2022. URL: https://eprint.iacr.org/2022/286.

[CHL22] Sílvia Casacuberta, Julia Hesse und Anja Lehmann. *SoK: Oblivious Pseudorandom Functions*. Cryptology ePrint Archive, Paper 2022/302. https://eprint.iacr.org/2022/302. 2022. URL: https://eprint.iacr.org/2022/302.

[Fao+22] Antonio Faonio u. a. „Auditable Asymmetric Password Authenticated Public Key Establishment“. In: *Cryptology and Network Security*. Hrsg. von Alastair R. Beresford, Arpita Patra und Emanuele Bellini. Cham: Springer International Publishing, 2022, S. 122–142.

[Jar+22] Stanislaw Jarecki u. a. „Asymmetric PAKE with low computation and communication“. In: Springer-Verlag, 2022.

[Jia+22] Jingwei Jiang u. a. *Quantum-Resistant Password-Based Threshold Single-Sign-On Authentication with Updatable Server Private Key*. Cryptology ePrint Archive, Paper 2022/989. 2022. URL: https://eprint.iacr.org/2022/989.

[RX22] Lawrence Roy und Jiayu Xu. „A Universally Composable PAKE with Zero Communication Cost (And Why It Shouldn't Be Considered UC-Secure)". In: *IACR Cryptol. ePrint Arch.* 2022 (2022), S. 1607. URL: https://api.semanticscholar.org/CorpusID:253857278.

[Beg+23] Hugo Beguinet u. a. *GeT a CAKE: Generic Transformations from Key Encaspulation Mechanisms to Password Authenticated Key Exchanges*. Cryptology ePrint Archive, Paper 2023/470. https://eprint.iacr.org/2023/470. 2023. URL: https://eprint.iacr.org/2023/470.

[DL23a] Dennis Dayanikli und Anja Lehmann. *Password-Based Credentials with Security against Server Compromise*. Cryptology ePrint Archive, Paper 2023/809. https://eprint.iacr.org/2023/809. 2023. URL: https://eprint.iacr.org/2023/809.

[DL23b] Dennis Dayanikli und Anja Lehmann. *Provable Security Analysis of the Secure Remote Password Protocol*. Cryptology ePrint Archive, Paper 2023/1457. https://eprint.iacr.org/2023/1457. 2023. URL: https://eprint.iacr.org/2023/1457.

[Haa23] Björn Haase. *(strong) AuCPace, an augmented PAKE*. Internet-Draft draft-haase-aucpace-07. Work in Progress. Internet Engineering Task Force, Jan. 2023. URL: https://datatracker.ietf.org/doc/draft-haase-aucpace/07/.

[Hao+23] Feng Hao u. a. *Owl: An Augmented Password-Authenticated Key Exchange Scheme*. Cryptology ePrint Archive, Paper 2023/768. https://eprint.iacr.org/2023/768. 2023. URL: https://eprint.iacr.org/2023/768.

[Hes+23] Julia Hesse u. a. *Password-Authenticated TLS via OPAQUE and Post-Handshake Authentication*. Cryptology ePrint Archive, Paper 2023/220. https://eprint.iacr.org/2023/220. 2023. URL: https://eprint.iacr.org/2023/220.

[PZ23] Jiaxin Pan und Runzhi Zeng. *A Generic Construction of Tightly Secure Password-based Authenticated Key Exchange*. Cryptology ePrint Archive, Paper 2023/1334. https://eprint.iacr.org/2023/1334. 2023. URL: https://eprint.iacr.org/2023/1334.

[SOR+23] Claudio SORIENTE u. a. „PASSWORD-AUTHENTICATED PUBLIC KEY ESTABLISHMENT". EP1234567. European Patent Office. 2023. URL: https://data.epo.org/publication-server/document?iDocId=7196780.

List of Figures

Tabellenverzeichnis

List of Listings

List of Algorithms

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.
Die Stellen der Arbeit, die anderen Quellen im Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

Ich erkläre ferner, dass ich die vorliegende Arbeit in keinem anderen Prüfungsverfahren als Prüfungsarbeit eingereicht habe oder einreichen werde.

Die eingereichte schriftliche Fassung entspricht der auf dem Medium gespeicherten Fassung.

Ort, Datum Unterschrift

Data medium

To the hard-copy of this thesis, a physical data medium is attached,
containing the files listed below. Readers of the PDF may download these files from
github.com/tobjasr/mastersthesis.

- `/` — Root directory
  - `thesis.pdf` — PDF-file of this Masters Thesis
  - `utils` — Utility functions
  - `NTRU` — NTRU Project directory
    - `recoverGfromHandF_NTRU.py` — NTRU Generator Base Recovery
  - `mA3PAKEkeyPairGen` — mA3PAKE Key-pair Generator Project Directory
    - `mA3PAKEkeyPairGen.js` — mA3PAKE Key-pair Generator
  - `OPRFserverless` — OPRF Server for Node.js on AWS Lambda Project Directory
    - `oprfServerless.ts` — OPRF Server for Node.js on AWS Lambda in TypeScript
  - `PrimeFactorRecovery` — RSA Prime Factor Recovery Project Directory
    - `recover_pq.cpp` — RSA Prime Factor Recovery in C++
    - `recover_pq.cs` — RSA Prime Factor Recovery C
  - `KeyBaseVault` — Generator Base Vault Project Directory
    - `gensafeprime.js` — Safe Prime Generator in JavaScript
    - `vault.js` — Generator Base Vault in JavaScript
      - `js` — Generator Base Vault Client-side JavaScript
        - `randomart.js` — Fingerprint Randomart Client-side in JavaScript
        - `deriveVerifierKey.js` — Derive Verifier Key in JavaScript
        - `generateKeyBase.js` — Generate RSA Key Base in JavaScript
  - `oprfClient` — OPRF Client in TypeScript Project Directory
    - `oprfClientDemo.js` — OPRF Client Demo Script in JavaScript
    - `oprfClient.ts` — OPRF Client in TypeScript
  - `GenerateKeyBase` — RSA Generator Base Project Directory
    - `generateRSAKeyBase.js` — RSA Safe Prime Generator
    - `worker_node-forge.js` — RSA Safe Prime Generator Worker Node-forge
    - `generateKeyPairGenBaseRSA_node-forge.js` — Generate Key Base RSA Node-forge
    - `worker_webcrypto-api.js` — RSA Safe Prime Generator Worker WebCryptoAPI

- `worker.js` ..... RSA Safe Prime Generator Worker
- `generateKeyPairGenBaseRSA.js` ..... Generate Key Base in JavaScript
- `fingerprintRandomart` ..... Fingerprint Randomart Project Directory
  - `fingerprintRandomart.js` ..... Fingerprint Randomart in JavaScript
  - `fingerprintRandomart.c` ..... Fingerprint Randomart in C
- `mA3PAKE-Client-Server` ..... mA3PAKE Client-Server Project Directory
  - `client-server-prototypes.js` ..... mA3PAKE Client-Server Prototyping
  - `client-simulation.js` ..... mA3PAKE Client-Server Simulation
  - `mA3PAKE_first-draft.png` ..... First draft of a mA3PAKE flow chart
  - `terminal-output-2.txt` ..... Terminal output from client-server.js