Spring 5-16-2020

# The Prom Problem: Fair and Privacy-Enhanced Matchmaking with Identity Linked Wishes

Dwight Horne
*Southern Methodist University*, dwight.horne@hotmail.com

THE PROM PROBLEM: FAIR AND PRIVACY-ENHANCED MATCHMAKING

WITH IDENTITY LINKED WISHES

Approved by:

_____
Dr. Suku Nair
Professor

_____
Dr. Amit Basu
Professor

_____
Dr. Jennifer Dworak
Associate Professor

_____
Dr. Daniel Engels
Professor of Practice

_____
Dr. Eric C. Larson
Associate Professor

# THE PROM PROBLEM: FAIR AND PRIVACY-ENHANCED MATCHMAKING WITH IDENTITY LINKED WISHES

A Dissertation Presented to the Graduate Faculty of

Lyle School of Engineering

Southern Methodist University

in

Partial Fulfillment of the Requirements

for the degree of

Doctor of Philosophy

with a

Major in Computer Science

by

R. Dwight Horne Jr.

(B.S. Computer Science, Baylor University, 2002)
(M.S. Computer Science, Baylor University, 2004)
(M.S. Security Engineering, Southern Methodist University, 2010)

May 16, 2020

ACKNOWLEDGEMENTS

First of all, I am grateful to my advisor Dr. Suku Nair for his support, encouragement, and wise counsel through the course of this research endeavor. I also thank the members of my PhD committee for their valuable contributions in the form of challenging questions, thoughtful feedback, ideas for future work, and generally giving of their time (a very limited resource) to further enhance these research pursuits. Finally, I am extremely thankful to my family for making significant sacrifices, while also being a source of continuous encouragement in support of my educational pursuits.

R. Dwight Horne, Jr.

B.S., Baylor University, 2002
M.S., Baylor University, 2004
M.S., Southern Methodist University, 2010

The Prom Problem: Fair and Privacy-Enhanced Matchmaking with Identity Linked Wishes

Advisor: Dr. Suku Nair

Doctor of Philosophy conferred May 16, 2020

Dissertation completed April 27, 2020

In *the Prom Problem* (TPP), Alice would like to attend the prom, a formal dance in high school, with Bob. She would like a risk-free, privacy preserving way to find out whether Bob shares that same wish. However, if Bob does not feel the same way, she does not want anyone to learn that she inquired about the wish, not even Bob. TPP represents a special class of matchmaking challenges that augment the properties of *privacy-enhanced matchmaking* further requiring fairness as well as support for *identity linked wishes* (ILW) – wishes that involve specific identities and are valid if and only if all involved parties have those same wishes. Prior protocols, and solutions to related problems, are vulnerable to a variety of practical attack threats such as impersonation, inference, database compromise, and early termination in the context of TPP.

The Horne-Nair (HN) protocol was put forth as a solution to TPP along with a sample embodiment in pseudo-code form and its security analysis. The protocol leverages an untrusted matchmaker and neither identities nor pseudo-identities are included in any

messages or stored in the matchmaker's database. All privacy relevant data stay within the control of the user. A security analysis confirmed that the HN protocol securely and privately accomplishes the competing demands of anonymity and authentication. A proof-of-concept implementation validated the approach, the fairness of the protocol was quantified, and a feasibility analysis demonstrated the practicality of computational costs and communication overhead in the context of real-world networks and systems, even with enhanced anonymity approaches, thereby bounding risk prior to incurring the full costs of development.

The SecretMatch™ Prom app leverages one embodiment of the patented HN protocol to achieve privacy-enhanced and fair matchmaking that supports identity linked wishes. The body of research also led to practical lessons learned and recommendations for privacy engineering in the modern era of rapidly evolving privacy legislation. Next steps after app store publication include design and derivation of SecretMatch™ apps for other contexts such as voting negotiations in legislative bodies, corporate mergers and acquisitions, and executive recruiting. The roadmap toward a quantum resistant SecretMatch™ has already begun with design of a Hybrid Post-Quantum Horne-Nair (HPQHN) protocol, aimed at securing against quantum enabled adversaries, along with performance testing that demonstrated efficient operations. Additionally, a broad spectrum of future research directions is planned such as enhancements to HPQHN, a fully Post Quantum HN (PQHN) protocol, a quantum-enabled SecretMatch™ technology, and more.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

*In dedication to my extremely supportive family...*

# CHAPTER 1

# INTRODUCTION

In *The Prom Problem* (TPP), Alice wishes to attend the prom, a formal dance, with Bob. She would like to determine whether Bob shares the same wish, but without anyone learning about her secret. In fact, if Bob does not feel the same way, she does not even want Bob to know that she inquired about the potentially shared wishes. TPP embodies a distinctive class of matchmaking challenges that further amplify the conflicting goals of anonymity and authentication in privacy-enhanced matchmaking, adding requirements for fairness and support for *identity linked wishes* (ILW) [1]. Fairness in matchmaking was equated early on with affording joint notification of wishes and equivalent exchange [2]. ILW are wishes that involve specific identities and are valid if and only if all involved identities have those same wishes.

A number of matchmaking protocols have attempted to match users with common wishes while achieving a variety of security properties or privacy protections, often striving to accomplish differing goals. But prior protocols are not sufficient for application in the context of TPP due in large part to a lack of fairness, and the inability to support ILW in a manner that preserves privacy of the users. In [1], TPP was characterized along with a defining set of security properties, the Horne-Nair (HN) protocol was put forth as a solution to TPP along with a pseudo-code example of one embodiment, and sketches of the proofs of security were presented. The pseudo-code example was based on the proof-of-

concept implementation that was used to validate the protocol. A subsequent feasibility evaluation then focused on quantifying the fairness of the HN protocol along with experimentation designed and executed to determine whether it might be practical to achieve high degrees of fairness, confidence in the matching result, and anonymity with application of the protocol in real-world networks and systems [3].

After presenting TPP, the HN protocol as a candidate solution, and demonstrating its feasibility prior to incurring the full costs of development of a production system, the next steps were an expanded security analysis along with design and development of a privacy-enhanced and fair matchmaking system. Although an overview of the architecture of the proposed system was presented in [4] accompanying highlights of a number of applications to domains beyond the context of dating and relationships, critical detailed design decisions remained before development of the system could proceed. From a practical perspective, due to avoidance of the centralized storage and processing of privacy-relevant data by the proposed system, it could prevent much of the damage from data breaches of matchmaking social systems like [5], which allegedly led to blackmail, ruined careers, and even suicide. The initial application targets the dating problem of TPP akin to the contextual scenario of Tinder, Match.com, and other matchmaking services. Thereafter, the body of work could be used as a basis for tailoring of applications for other domains such as voting negotiations in legislative bodies and recruiting of high level executives.

The remainder of this chapter is organized as follows. Section 1.1 gives a more detailed description of the problem. Section 1.2 then highlights the positioning of TPP within an established taxonomy of privacy-enhanced technologies. Subsequently, the

subsections of 1.3 describe a sampling of applications of a solution to TPP beyond the context of dating and relationships. Finally, Sections 1.4 and 1.5 present practical attack scenarios to facilitate an understanding of the threats to such a system and conclude with summary comments respectively.

## 1.1 Problem Description

In a number of countries, there is a semi-formal or formal dance near the end of the senior year of high school that tends to figure prominently in popular culture. In the United States this formal dance is referred to as the *prom*. There are analogous events in other countries that are termed *senior ball*, *grad*, *formal*, *debs*, and so on. It is possible to attend such an event alone, but students typically prefer to attend as a couple. This dating scenario serves as the framework through which *the Prom Problem* (TPP) was proposed [1]. In TPP, Alice secretly wishes to attend the prom with another student named Bob. Alice desires a risk-free, privacy-preserving method of discovering if Bob reciprocates the same feelings. More specifically, Alice requires a process whereby she and Bob will receive confirmation if they share the same wish, while no third party can learn any useful information. Moreover, if Bob does not share the same secret, Alice does not even want Bob to know that she inquired about it.

The term *identity linked wishes* (ILW) was coined to refer to wishes that are valid if and only if all involved parties have the same wishes. ILW can often be easily guessed (e.g., anyone could state that "Alice and Bob like each other"), but the veracity of such a guess would remain unknown without confirmation from each of the linked identities. Thus, TPP further amplifies the conflict between anonymity and authentication in matchmaking that was originally highlighted by Baldwin and Gramlich [2]. In fact, TPP

augments the goals of *privacy-enhanced matchmaking* as defined by Shin and Gligor [6] and necessitates the addition of the following requirements [1].

- Fairness – joint notification of wishes with equivalent exchange
- Support for identity linked wishes (ILW)

The two additional requirements significantly increase the complexity of the problem. For instance, prior protocols for matchmaking and related privacy-enhanced methodologies are often vulnerable to early termination attacks that prevent fairness and lead to a compromise of privacy when ILW are used rather than generic secrets that are not linked to specific identities. The support for that assessment becomes clear during the discussions of practical attack scenarios in section 1.3, related work in Chapter 2, and the challenges of fairness in 3.2. The security properties of privacy-enhanced matchmaking with ILW are presented in detail along with the security analysis in Chapter 3.

## 1.2 The Prom Problem in a Taxonomy of Privacy-Enhanced Technologies

Privacy-enhancing technologies (PETs), systems or methods that endeavor to accomplish some task with a concentration on preservation of user privacy, have been under development in various forms for some time. Early examples of PETs include privacy-enhanced Internet mail [7] [8] [9] and privacy-enhanced intrusion detection [10] [11]. A sampling of subsequent privacy-enhanced technological pursuits span a wide variety of applications including web personalization [12], search and data retrieval [13] [14], database interactions and data analytics [15] [16] [17], and social networking [18] [19]. An exhaustive list would certainly be much longer. Recognizing the importance of privacy-enhancing technologies, but also the challenges with analysis and comparison of different

approaches and applications, Heurix, Zimmermann, Neubauer, and Fenz proposed a universal taxonomy of PETs to facilitate systematic comparative analysis [20]. At the first level, their proposed taxonomy has the following seven dimensions.

- **Aim** – intent or means of achieving privacy
- **Aspect** – aspect of privacy addressed by the PET
- **Data** – type of data addressed by the PET
- **Foundation** – the underlying security model or conceptual and technical foundation from the general domain of security
- **Reversibility** – circumstances under which operations are reversible
- **Scenario** – clarifies the security model by identifying untrusted entities and threat actors
- **Trusted Third Party (TTP)** – degree of involvement of a trusted third party

| | | | | |
|---|---|---|---|---|
| **Aim** | ⇒ | Confidential, Deniable, Indistinguishable, Unlinkable | | |
| **Aspect** | ⇒ | Behavior, Content | | |
| | ⇒ | Identity | ⇒ | Anonymity |
| | | | | ⇒ Directionality = Single |
| | | | | ⇒ Directionality = Multi |
| | ⇒ | Identity | ⇒ | Pseudonymity |
| | | | | ⇒ Cardinality = Limited |
| | | | | ⇒ Cardinality = Unlimited |
| | | | | ⇒ Directionality = Single |
| | | | | ⇒ Directionality = Multi |
| | | | | ⇒ Holder = Individual |
| | | | | ⇒ Holder = Group |
| **Data** | ⇒ | Stored, Transmitted, Processed | | |

Figure 1.1. Taxonomy of privacy-enhanced technologies – part 1 [20]

Underneath each of the primary dimensions, the authors proposed a hierarchy of nodes representing properties and sub-properties of that dimension. Figures 1.1 and 1.2 present parts of the complete taxonomy. Refer to [20] for a more detailed discussion of each of the properties and sub-properties comprising the complete hierarchy.

| | | | | |
|---|---|---|---|---|
| **Foundation** | ⇒ | Security Model | ⇒ | Computational |
| | ⇒ | Security Model | ⇒ | Information Theoretic |
| | ⇒ | Cryptography | ⇒ | Asymmetric |
| | ⇒ | Cryptography | ⇒ | Non-Cryptographic |
| | ⇒ | Cryptography | ⇒ | Symmetric |
| | ⇒ | Cryptography | ⇒ | Unkeyed |
| **Reversibility** | ⇒ | Cooperation | ⇒ | Required |
| | ⇒ | Cooperation | ⇒ | Not Required |
| | ⇒ | Degree | ⇒ | Deniable |
| | ⇒ | Degree | ⇒ | Full |
| | ⇒ | Degree | ⇒ | None |
| | ⇒ | Degree | ⇒ | Partial |
| **Scenario** | ⇒ | Untrusted Client, Untrusted Server, Mutual, External | | |
| **TTP** | ⇒ | Frequency | ⇒ | Never |
| | ⇒ | Frequency | ⇒ | Permanently |
| | ⇒ | Frequency | ⇒ | Situational |
| | ⇒ | Phase | ⇒ | None |
| | ⇒ | Phase | ⇒ | Regular |
| | ⇒ | Phase | ⇒ | Setup |
| | ⇒ | Task | ⇒ | None |
| | ⇒ | Task | ⇒ | Operation |
| | ⇒ | Task | ⇒ | Validation |

Figure 1.2. Taxonomy of privacy-enhanced technologies – part 2 [20]

An analysis of the HN protocol for TPP with ILW in the context of the universal PET taxonomy yielded the properties of Figure 1.3 [4]. In the **Aim** dimension, indistiguishability, or exhibiting resistance to attempts to unambiguously distinguish one entity from another, is a primary goal. Other key goals in TPP include unlinkability pertaining to both users and wishes, as well as confidentiality. The property of deniability is not a fundamental goal of TPP or the HN protocol, but it could be achieved if desired for certain applications. In the **Aspect** dimension, privacy of content and multi-directional anonymity are critical challenges that must be resolved. Additionally, privacy of behavior is partially achieved via the HN protocol in that no wishes or identities are included in any messages or stored in the database. Traffic analysis attacks and related attempts to ascertain behavioral details may be thwarted by combining the use of encryption and/or anonymous communication channels.

| | | | |
|---|---|---|---|
| **Aim** | ⇒ Indistinguishable, Unlinkable, Confidential, Deniable* | | |
| **Aspect** | ⇒ Content, Behavior** | | |
| **Aspect** | ⇒ Identity | ⇒ Anonymity | ⇒ Direction=Multi |
| **Data** | ⇒ Stored, Transmitted, Processed | | |
| **Foundation** | ⇒ Security Model | ⇒ Computational | |
| **Foundation** | ⇒ Cryptography | ⇒ Asymmetric | |
| **Reversibility** | ⇒ Degree | ⇒ None | |
| **Reversibility** | ⇒ Cooperation | ⇒ N/A | |
| **Scenario** | ⇒ Mutual | ⇒ Sender/Receiver | |
| **Scenario** | ⇒ External | ⇒ Matchmaker/Interceptor | |
| **TTP** | ⇒ Frequency | ⇒ Never | |
| **TTP** | ⇒ Phase | ⇒ None | |
| **TTP** | ⇒ Task | ⇒ None | |

* Not a primary goal but could feasibly be achieved if desired
** Partial with the HN protocol alone; complete with anonymous communication channels

Figure 1.3. Taxonomy applied to TPP and the HN protocol [4]

All three of the **Data** privacy dimensions must be addressed with TPP and ILW including data being processed, data at rest (stored), and data being transmitted. The **Foundation** dimension for the HN protocol is determined by the underlying security model of the asymmetric cryptographic algorithms, and to a lesser extent the one-way hashing algorithms. In most implementations, that security model would be computational rather than information theoretic, the latter of which represents security against adversaries with unbounded computational resources.

Since operations in the HN protocol are not intended to be reversible, the **Reversibility** domain is not applicable. On the other hand, the **Scenario** dimension is a critical factor that distinguishes TPP and the HN protocol from many other PETs. In particular, the assumed mutual distrust between senders and receivers along with external threats in the form of eavesdroppers, active attackers, and even the third party matchmaker itself contrasts with solutions that rely on a TTP, semi-honest participants, or both. Lastly, due to the nature of the problem as conveyed by the applicable properties of the aim, aspect, and scenario dimensions, a TTP cannot be relied upon with TPP. Consequently, the **TTP** dimension is essentially not applicable with a frequency value of "never".

The proposed universal taxonomy of PETs was a useful tool for analysis of the problem and a practical framework within which to analyze the HN protocol for TPP with ILW enabling comparative analysis with other PETs. Nevertheless, a few opportunities for improvement became apparent. The first observation was that the taxonomy mixes domains such as **Scenario** that apply primarily to the problem being solved with domains specific to a particular solution like **Foundation**. For instance, two solutions for the same scenario might use different forms of cryptography and have differing underlying

security models. Hence, it might be worthwhile to separate the domains of the problem from the domains that apply to individual approaches to solving the problem. Furthermore, several security goals of matchmaking cannot be represented in the taxonomy in its current form. While some goals such as *matching result privacy* may be too specific to matchmaking to be incorporated into a "universal" taxonomy, the most notable omission that should be considered is the lack of an ability to capture a requirement for fairness. One approach to remedying that shortcoming would be to add a *Fairness* attribute to the **Aim** dimension [4].

## 1.3 Applications Beyond Dating and Relationships

The problem of fair and privacy-enhanced matchmaking with ILW was initially presented in the context of TPP because the complexities of dating and relationships are some of the most universal concepts. However, a solution to TPP has a number of other potential applications, contexts in which privacy-enhanced and fair matchmaking using ILW would be beneficial, that should be considered when designing a system for TPP. A representative sampling of applications beyond dating and relationships that were presented in [4] are subsequently discussed in turn.

### 1.3.1 Recruiting of High Level Executives

With the introduction of *trustable matchmaking*, Baldwin and Gramlich initially presented a protocol for risk-free matchmaking framing it in the context of recruiting for high level executive job positions [2]. Indeed, recruiting of high level executives, and in some cases recruiting of top tier technical talent, continues to be a challenge with un-

wanted risk. For example, suppose Company A is interested in hiring Alice for its Chief Executive Officer (CEO) position and they may not want to disclose it publicly. At the same time, although she is happily and gainfully employed, Alice might secretly be interested in new opportunities. In such cases, Company A and Alice could clearly benefit from a privacy-enhanced and fair matchmaking system that supports ILW.

### 1.3.2 Voting Negotiations in Legislative Bodies

In the legislative bodies of the world such as the Senate or Congress of the United State of America, the Parliament of the United Kingdom, or gatherings of royalty in monarchies, a considerable amount of time can be spent on voting negotiations. As with other matchmaking problems, these negotiations often exhibit a conflict between anonymity and authentication akin to TPP. Suppose a particularly important piece of legislation is coming up for a vote and Alice's party is expected to vote *No*. However, she has a strong conviction that she should vote *Yes* and she is willing to do so, but only if Bob, another member of her party, will also break party lines and vote in favor of the measure. If Bob does not share the same ILW, she does not even want Bob to know that she inquired about it and thusly considered going against party lines for fear of risking damage to her reputation, status, or career. Many of the threats in this case mirror the threats present in TPP. For example, there would be significant risks such as inference of thoughts and motives based on observed behavior and surrounding circumstances. Negotiations related to voting in legislative bodies could also benefit from a privacy-enhanced and fair matchmaking system that supports ILW.

### 1.3.3 Corporate Mergers and Acquisitions

High stakes negotiations can often be involved when it comes to corporate mergers and acquisitions (M&A). Consider a case where Company A is interested in acquiring Company B but, unbeknownst to Company A, Company B is already in talks with Company C about a similar venture. If Company A were to disclose its intentions and it were to become public knowledge, some of the various scenarios that might unfold could damage Company A's stock price, reputation, or negotiating position. Alternatively, if Company A were interested in acquiring Company B, yet the leadership of the latter perceives them to be an equal peer, then they may be offended at the suggestion of acquisition, while they might have been open to a merger. There are a number of possible scenarios in the area of corporate M&A with risks akin to those of TPP. Hence, corporate M&A is another potential beneficiary of a fair and privacy-enhanced matchmaking system that supports ILW.

### 1.3.4 Affluent and Institutional Investing

Institutional investors, as well as affluent individual investors, often buy and sell significant quantities of investment vehicles and those transactions can impact asset prices, market volatility, and more. If Alice were an affluent investor, or a key decision maker for an institutional investor, she might be pondering the purchase of a specific security. But consider a case where her interest in purchasing the security is contingent upon other factors such as an offering at a certain price point, agreement to undertake internal restructuring, or a promise by the entity to take actions expected to make an external societal impact. Relative to the large volume of daily transactions in global markets, the num-

ber that might benefit from privacy-enhanced and fair matchmaking with support for ILW would be a small minority. On the other hand, the potential impact of a risk-free matchmaking system to facilitate that minority could have significant ramifications. Note that this example is presented without regard to limitations imposed by legal regulations due primarily to the fact that such regulations can vary significantly across differing markets and between disparate asset classes. But any system empowering investor actions must take into account the laws and regulations applicable to the principalities in which the system is designed to operate.

### 1.3.5 Peace and Other Treaty Negotiations

Consider the case of two countries with a long history of hostilities toward one another. While both sides are likely weary of continued conflict, a number of variables come in to play preventing progress toward peace. For instance, if Alice and Bob represent the leadership of their principalities in the conflict, they each have a desire to be perceived by their people as steadfast and resolute for the morale of their citizens and to support career longevity. As a result, neither of them may be willing to publicly admit an inclination toward compromise or concessions that would be required for a peace treaty. In fact, they may not even be willing to disclose such intent privately unless they are certain that the other side has the same wishes. If the other party does not feel the same way, disclosing intent to compromise or give concessions could risk one's reputation, career, negotiating position, and more. Thus, peace and other treaty negotiations exemplify another case where a solution to TPP could also be an enabler of improved negotiations and po-

tentially foster positive outcomes in challenging situations that today are perceived by many to be nearly unsolvable.

## 1.4 Practical Attack Scenarios

To understand what practical threats exist, it is useful to first determine what concepts or entities should be protected in the given scenario. In TPP, the generic wishes that are not linked to any particular identities are not a secret. For example, "attend prom together" is not meaningful in a privacy-preserving sense without being linked to specific identities that share that wish. On the other hand, the identity linked wishes are an entity that must be protected. Otherwise stated, the linkability of identities to wishes requires protection. Moreover, the association of one identity to another should be protected even in the absence of the generic wishes since, due to the nature of the problem, when considered in context many details may be easily inferred. Yet another aspect of the TPP that must be protected is the related concept of fairness. An exchange that lacks joint notification of wishes and equivalent exchange would be unfair. To aid threat analysis, consider the following practical examples of attack threats highlighted in [1] that are expected to be commonplace with TPP.

**Inference**. As students wait in anticipation of the prom event that is rapidly approaching, Eve observes Alice and Bob communicating with a shared secret protocol. Given the timing, the act of simply observing two parties initiate an exchange reveals an unacceptable amount of information given that Eve can easily infer that they are inquiring about attending the prom with each other. Another risk is evident if Alice attempts to initiate a shared secret protocol with Bob with the prom approaching. Bob may not be

interested in Alice but he can infer her interest in him. Vulnerability to inference would violate a fundamental requirement for privacy preservation with TPP.

**Impersonation and False Disclosure.** Perhaps in an attempt to embarrass Bob, Trudy might try to impersonate Alice and complete the shared secret protocol with Bob to convince him that she knows "Alice and Bob want to attend prom together." She might then succeed in her goal of Bob's embarrassment when Bob found out that Alice is not interested in him and that it was really an imposter that he had interacted with.

**Early Termination.** There are a number of protocols for matchmaking and other related problems that cannot guarantee fairness (e.g., [21] [22] [23] [24]). In a protocol without fairness, Alice might terminate the protocol after receiving confirmation of a shared secret while preventing Bob from learning that same fact. If combined with the impersonation threat, Trudy might learn of the shared secret while Bob does not learn anything, not even that it was an imposter that he had interacted with.

**Data Compromise**. In any system or protocol with sensitive data stored in a repository such as a database, there is risk of privacy compromise. A pertinent example in the context of matchmaking was the Ashley Madison hack of 2015 that has allegedly resulted in consequences such as ruined relationships, damaged careers, blackmail, and even suicide [5] [25] [26] [27]. In that case, the data breach primarily involved users' associations with a service that facilitates cheating on one's significant other. As a result, one might infer a user's wishes. But consider how much worse it could have been if been details of identity linked wishes had also been revealed.

There will undoubtedly be other types of attacks on the system, but the aforementioned categories represent a sampling of realistic threats in the context of TPP. The

mere act of direct communication between parties results in linkability and vulnerability to inference given the small wish space of wishes that are simple and guessable. The introduction of a trusted matchmaker can help with authentication, and it prevents direct communication, yet there are questions about the extent to which any third party can be trusted. Evidence has shown that even otherwise trustworthy third party entities may not be able to protect the sensitive data entrusted to them. The Ashley Madison hack of 2015 [5] might be a relevant example, although the extent to which the company should have been considered trustworthy might be questionable. Another example was the compromise of the sensitive, personally identifiable information of millions of citizens detected in 2015 by the United States Office of Personnel Management (OPM), an entity that would have almost universally been considered trustworthy [28] [29]. The Equifax data breach of 2017, that affected nearly half of the US adult population, was yet another instance with severe consequences [30]. This sampling of what are expected to be common threats must be considered when attempting to understand TPP and design a system to provide privacy-enhanced and fair matchmaking with ILW.

## 1.5 Summary

Although TPP is presented as a security and privacy challenge associated with matchmaking for a school dance, there are a number of potential applications beyond the context of dating and relationships. A sampling of possible applications includes executive recruiting, voting negotiations in legislative bodies, corporate M&A, institutional and affluent investing, and even some of the most challenging peace or other treaty negotiations. In essence, any matchmaking challenge involving ILW that requires fairness and

preservation of privacy in the event of a non-match is a potential application of a solution to TPP.

# CHAPTER 2

# RELATED WORK

As originally highlighted by Baldwin and Gramlich [2], matchmaking challenges exemplify a conflict between authentication and anonymity. Thereafter, matchmaking protocols have evolved to satisfy varying security and privacy requirements and to solve a variety of different problems. Table 2.1 characterizes the matchmaking protocols most closely related to TPP along a number of dimensions corresponding with its key security properties. The protocols in the table, those that most readily lend themselves to direct comparison in the context of TPP, are discussed in more detail in sections 2.1 through 2.4. Sections 2.5 and 2.6 then address other matchmaking approaches and related research problems respectively, all of which were evaluated as potential candidates for ap-

Table 2.1. Comparison of matchmaking protocols [1]

| **Match-making Protocol** | **Protocol Properties** | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Wish Secrecy** | **Anonymity** | **Fairness** | **3rd Party** | **Dictionary Attack Resistance** | **Forward Privacy** | **ILW** |
| Baldwin / Gramlich | | | ✓ | Trusted | | | |
| Meadows | ✓ | | | Trusted (Init.) | | | |
| Zhang / Needham | ✓ | ✓ | | Untrusted | | | |
| Shin / Gligor | ✓ | ✓ | | None | ✓ | ✓ | |
| Horne / Nair | ✓ | ✓ | ✓ | Untrusted | ✓ | ✓ | ✓ |

plication to TPP. Unfortunately, each has one or more shortcomings in the context of TPP, which ultimately inspired development of the Horne-Nair (HN) protocol.

## 2.1 Trustable Matchmaking

In 1985, Baldwin and Gramlich introduced the idea of a server to perform risk free matchmaking in an important work that showed great foresight [2]. They highlighted the conflicting goals of authentication and anonymity. The researchers outlined the requirements for what they referred to as trustable matchmaking in the context of a corporation seeking to hire a vice president from the current employees of its competitors. In this case, the hiring company does not want to advertise their wishes to poach a key employee from the competition and the senior level managers at the competition do not wish to reveal their interest unless both parties are assured of matching wishes. Apart from properties common to database systems, Baldwin and Gramlich listed the requirements of matchmaking systems as the following:

1. User anonymity
2. Match authentication
3. Joint notification of wishes
4. Graceful degradation of privacy

The authors presented the Baldwin-Gramlich (BG) protocol for trustable matchmaking to accomplish the aforementioned goals. Their approach consisted of a query-response protocol using asymmetric cryptography, one-way hash functions, an anonymous message forwarding service, and fake transactions. The BG protocol has two main

phases involving (1) session key exchange via trusted third party matchmaker and (2) computation of the confirmation value, the encrypted wish.

As an example, suppose Alice (A) has wish $\boldsymbol{w}$ and she would like to determine whether Bob (B) has that same wish. She obtains B's public key and sends $\boldsymbol{w}$ along with a session key encrypted with B's public key to matchmaker (M). Similarly, assuming B shares the same wish, he initiates a similar inquiry. Note that $\boldsymbol{w}$ is used as the database key and M will respond with the associated data (e.g., encrypted session keys) to all who query with $\boldsymbol{w}$. This allows the exchange of session keys between A and B. For the second phase, the users encrypt $\boldsymbol{w}$ with the pair of session keys yielding encrypted wish $W_E$ which the users submit to M along with a pseudonym. M uses a one-way hash function to compute $\delta = \text{Hash}(W_E)$ and stores $\delta$ along with the pseudonym(s) in a separate database. In the case that there are two or more pseudonyms associated with the same $\delta$, M notifies the users of a match.

Unfortunately, the BG protocol has a number of weaknesses when considered in the context of TPP. Although the matchmaker's use of a one-way function to store $W_E$ was an attempt to defend against malicious users or administrators, the honest users must still trust M with $\boldsymbol{w}$ (which is not kept secret), to scramble $W_E$, to test equality of wishes, and to provide joint notification of matches. Moreover, in addition to requiring a trusted third party matchmaker, Zhang and Needham demonstrated that the query-response portion of the BG protocol was vulnerable to substitution attacks affording an attacker the opportunity to read a response message in the query-response protocol thereby compromising confidentiality by having the matchmaker encrypt the response with a chosen key [23].

## 2.2 Mutual Authentication of Suspicious Parties

The following year after presentation of the BG protocol, Meadows proposed a method for authentication of mutually suspicious parties where credentials would not be revealed without a match [21]. The problem differed somewhat from the BG scenario. The context for discussion involved organizations with representatives desiring to exchange secret credentials while maintaining confidentiality of those credentials in the event of a non-match. The Meadows protocol still required a trusted third party initially, but it avoided the requirement for continuous availability. It also assumed a high degree of trust between participants.

In the Meadows protocol, users exchange a signed certificate attesting to their affiliated organization as well as their public credentials (which are the result of using a one-way hash function with their secret credentials as input). After certificate verification and session key establishment, exchanged values are encrypted with that session key. The users perform an OK handshake followed by exchange of exponential values computed from the other's public credentials and the user's secret credentials checking for a match upon receipt of the message from the other party. This protocol is an interesting approach to mutual authentication of suspicious parties specific to the scenario in which it was presented. But the Meadows protocol is clearly not a good candidate for TPP with the high degree of trust in users affording the opportunity for falsified matches as well as a lack of fairness with vulnerability to early termination attacks among other challenges of application in this context.

## 2.3 Private Matchmaking

Zhang and Needham later put forth the idea of private matchmaking with additional privacy requirements and argued that the BG and Meadows protocols, as variants of mutual authentication protocols, were not suitable for private matchmaking [23]. Zhang and Needham outlined the requirements of private matchmaking as secrecy of wishes, anonymity of users, and authentication of wish matches. They proposed the Zhang-Needham (ZN) protocol to accomplish those goals. The ZN protocol makes use of a public database for the untrusted matchmaker M and assumes some form of anonymous communications for anonymously communicating with M. The main idea is that users have wish $\boldsymbol{w}$ that can be used to find other users that share that wish and authenticate the match. Users generate key K by applying a one-way hash function to $\boldsymbol{w}$, or by using the result of the hash as the seed for a pseudo-random number generator, and then encrypting $\boldsymbol{w}$ with K to produce $W_E$, which gets submitted to M. If anyone else has submitted $W_E$, then there must be a match. To facilitate communication between users with matching wishes, users can also encrypt a session key and the information necessary to establish communication with the same key K and publish it alongside $W_E$.

A number of issues prevent the ZN protocol from being used to solve TPP with ILW. For instance, the ZN protocol is vulnerable to dictionary attacks. The ZN protocol also cannot guarantee joint notification of wishes because any user could simply query the database without submitting anything. It also falls short with the other element of fairness – equivalent exchange. For example, even if a user was forced to include her own values with the $W_E$ query, a malicious user could simply include fake contact information or random data. This poses a major problem in the context of ILW since Alice could

learn that someone apparently "knows" $\boldsymbol{w}$, but anyone could guess that "Alice and Bob like each other", so the veracity of it would remain unknown. Furthermore, another user would have learned that someone is inquiring about that secret which could lead to inference attacks depending on the situation. While the ZN protocol is a clever approach to determining whether there are other users anonymously inquiring about the same wishes, it is not well-suited to afford security and privacy in the context of TPP with ILW.

## 2.4 Privacy-Enhanced Matchmaking

Shin and Gligor proposed the Shin-Gligor (SG) protocol to solve *privacy-enhanced matchmaking*, which added to the goals of private matchmaking by requiring forward privacy of wishes, forward privacy of identities, and dictionary attack resistance [6]. The SG protocol leverages existing approaches to password authenticated key exchange (PAKE) [31] [32] with proven security properties but uses wishes in place of the password. It also makes use of pseudonyms to mask users' identities and achieves authentication at the conclusion of a protocol execution via digital signature of an ordered concatenation of the protocol messages referred to as an execution transcript. Shin and Gligor rigorously defined the properties of privacy-enhanced matchmaking and argued that the SG protocol satisfied those properties, in several cases benefiting from the work of previous proofs that the underlying PAKE protocol satisfied the same properties.

The removal of a third party, whether trusted or untrusted, may be viewed as an advantage in some situations. However, the direct communication between users can result in vulnerability to traffic analysis and inference attacks affording Eve opportunities to compromise user privacy in TPP. Perhaps more importantly, a critical problem with the

SG protocol is that, like other matchmaking protocols without a trusted third party, it lacks fairness.

Consider an example attack in which Trudy impersonates Bob and executes the SG protocol with Alice using the secret ILWs "Alice and Bob want to attend prom together" in place of the password. After execution of the protocol, if Trudy evaluated the digital signature on the execution transcripts she would validate Alice's secret, identity linked wishes and thereby compromise her privacy. On the other hand, if Alice evaluated the digital signature on the execution transcripts she would learn that it must have been an imposter, but it would be too late because her privacy had already been compromised. This vulnerability is even highlighted by the author's definition of *detector resistance* conveying the idea that an adversary "cannot learn the real identity" of an honest user unless they "execute the interaction on a same wish" [6]. Thus, if Trudy and Alice executed the protocol on the same ILW as in this example, Trudy would indeed learn the identity of Alice and compromise her privacy.

## 2.5 Other Matchmaking Problems

Several other matchmaking protocols have been proposed over the years but they differ more significantly with regard to their goals, the problems they aim to solve, and their security properties. Although these protocols do not lend themselves as readily to direct comparison with those in Table 2.1, they are included here for completeness. In the context of TPP and ILW, these protocols suffer from many of the same drawbacks as the aforementioned matchmaking approaches.

### 2.5.1 Coupling via Trusted Third Party with Public Commitment

Comparing their work to a matchmaking TV show in Korea in which participants select and commit to others and are subsequently "coupled" by the host, Lee and Kim proposed a protocol that combined ElGamal asymmetric encryption, digital signatures, and zero knowledge proofs of knowledge for equality of discrete logarithms to accomplish a comparable task [22]. In Table 2.1, their protocol would most closely resemble the BG protocol in that it accomplishes fairness by placing complete faith in a trusted third party (TTP). The five stage protocol consists of registration, commitment, opening by the TTP, proof of coupling, and public verification. It also attempts to achieve alternative goals such as public commitment and public non-repudiation, which further complicates direct comparisons. In the end, among other issues, the requirement for full trust in the third party implies that this protocol is clearly unsuitable for TPP.

### 2.5.2 Private Discovery of Shared Topics of Interest

Atallah and Cho described a protocol for privately discovering others with shared interests that most closely resembles the ZN protocol in Table 2.1 [33]. The Atallah-Cho (AC) approach relies on the concept of a semantic tree-based hierarchy of topics of interest (TOIs) to achieve varying degrees of privacy. For example, if the user had private TOI $x_i$ = "Indian vegetarian recipes" then they might use ancestors $y_i$ such as "vegetarian recipes" for lesser privacy or "recipes" for greater privacy with the tradeoff being that more distance between private and public TOIs incurs more computational costs. In the AC protocol, users generate a 3-tuple consisting of the public TOI $y_i$ and a "random pair", the second value of which is related to the private TOI $x_i$ by way of the Diffie-Hellman

problem [34]. Users "commit" values by sending each 3-tuple to the matchmaker, referred to the Service Provider (SP), with whom they maintain an authenticated connection so it learns of all parties' $y_i$ values and their associated identities. Hence, the SP would know which users were inquiring about "vegetarian recipes" in the previous example but not that they were specifically seeking "Indian vegetarian recipes." The SP posts the 3-tuples to a public message board, which users later anonymously query for all 3-tuples matching $y_i$, an ancestor of $y_i$, or a descendant of $y_i$. As one can imagine, this could be quite a large number considering the number of users that might have TOI at the root node of "recipes." For each of the retrieved 3-tuples, the user attempts decryption using $x_i$ as the key and subsequently derives a symmetric key that will be shared between users $i$ and $j$ if $x_i = x_j$ or non-shared if $x_i$ != $x_j$. User $j$ then encrypts a message $m_j$ with each of the symmetric keys that were computed (for each of the 3-tuples for all $y_i$ entries as well as those for all ancestors and descendants) and the resulting ciphertexts are sent back to SP associated with the additional pair of commitments. The SP charges users a "credit" for each encrypted message submission with the intent of limiting excessive message sending and it forwards the message to the intended recipients. The recipients then attempt to decrypt the received ciphertext and obtain either a "useful" message if $x_i = x_j$ or alternatively a "useless" message.

There are a number of drawbacks to this approach, both in general and more importantly in the context of TPP with ILW. First of all, the algorithm does not afford efficient location of those with common interests in many, if not most, situations. For example, consider computations regarding all user inquiries about all TOIs in the hierarchy, both ancestors and descendants, related to "recipes" just to find someone interested in

"Indian vegetarian recipes" as in the previous example. Another challenge is that the algorithm cannot be employed in cases where the assumption of a "natural semantic tree hierarchy" of TOIs does not hold true. Another logistical challenge is the restriction that users must have private access to subsets of the public message board that SP also interacts with, but without the SP being able to learn what those subsets are. When applied to TPP, in addition to potential computation and communication overhead resulting from the many-to-many nature of the protocol, as with most prior matchmaking protocols, the AC protocol is vulnerable to impersonation or false disclosure with ILW. Trudy could easily impersonate Alice and prove interest in the TOI "Alice loves Bob" but that does not make the ILW true. Furthermore, the authors claim an untrusted SP, yet users utilize authenticated communication with the SP and they rely on the SP to implement fairness (e.g., it is assumed to not withhold messages, to not post identities to the public message boards, etc.). Contrary to the security model of TPP, the AC protocol also assumes semi-honest users. Unfortunately, if this does not hold true (as is the case in TPP), there are several opportunities for users to "cheat" the process. The AC protocol has a number of fundamental vulnerabilities making it unsuitable for use in TPP.

### 2.5.3 Shared Secret Verification

United States patent US 8,527,765 B2 contains disclosure of a method and system for shared secret verification [35]. The method works as follows. Alice encrypts her random number with Bob's public key, adds her secret to it, and sends the resulting value to Bob. Bob then subtracts his secret, decrypts the result with his private key, applies a one-way function, and sends the result of that hashing operation back to Alice. At that point, Alice

can compare with a hash of her original random number to determine whether or not they share the same secret. According to the protocol, Alice then sends her random number to Bob so that he can apply a one-way hash function to it to similarly determine the result. In the context of TPP, this protocol has a variety of vulnerabilities. For example, Alice and Bob communicate directly introducing vulnerability to inference. If they communicate anonymously attempting to avoid inference, the protocol is then vulnerable to impersonation. It also lacks fairness in that Alice can terminate the protocol early having learned the result while Bob has learned nothing. Comparable to other related works, this approach cannot be used to solve TPP with ILW.

### 2.5.4 Shared Document Comparison

In United States patent US 8,032,747 B2, Patrick disclosed a method and system to determine whether two parties possess the same document [24]. In this approach, Alice and Bob exchange random numbers, compute first and second values as the results of applying a one-way hash function to the concatenation of their document with two orderings of the random numbers, and subsequently exchanging those first and second values with equality of each pair verifying the match. If applied to TPP, in addition to being vulnerable to inference and impersonation, this protocol is clearly unfair as either party could terminate the protocol early and prevent the other from learning the outcome. It also assumes honest users as either party could easily falsify a result and cause false disclosure of a matching or non-matching result. Although one could potentially substitute ILW for the document under comparison, this protocol is clearly not a good fit for TPP.

### 2.5.5 Stable Marriage and Stable Roommate Problems

Given disjoint sets of N women and N men that have ranked members of the opposite sex in order of preference, the *stable marriage problem* is that of coupling the group into matching pairs, one from each group, such that no two individuals that are not matched together both prefer each other over their matched partners. Matchings are called *stable* if there are no non-matched couples, each of whom prefers one another to their partners. The stable marriage problem was originally put forth by Gale and Shapley in the context of the college admissions process [36]. They showed that a stable matching exists for every instance of the problem and provided an algorithm as a candidate solution to the problem. McVitie and Wilson later identified a method of finding all possible stable marriage assignments for a given problem instance [37]. Analyses of the Gale/Shapley and McVitie/Wilson approaches have shown them to have worst-case time complexity of $O(n^2)$ with an average case complexity of $O(n \log n)$ for the core algorithm logic [38] [39].

In the closely related *stable roommates* problem, a non-bipartite version of the stable marriage problem, the goal is to identify a stable matching of persons from a single group of cardinality $n$ where each has ranked the $n-1$ others. In contrast to the stable marriages problem, it was shown that instances of the stable roommates problem exist for which there is no stable matching assignment [36]. As a response to the open problem identified by Knuth [39], Irving presented a polynomial-time algorithm to compute a solution for instances of the stable roommates problem, if one exists [40]. By highlighting that the Gale/Shapley algorithm favors one gender over the other by generating a *male optimal* solution, Knuth also posed the problem of a solution that is more equitable [39]. Ir-

ving, Leather, and Gusfield responded with a solution to the "optimal" stable marriage problem, also known as the egalitarian version of the problem [41]. Their approach seeks to maximize the average satisfaction for all people and it was shown to be $O(n^4)$.

Subramanian explored the relationship between network stability and stable matching problems [42]. Among other things, that work showed that stable matching problems were comparable to problems of identifying stable configurations of X-networks and the NP-completeness of three-party stable marriage problems. Building on that work, Feder identified an $O(\min(n, \sqrt{K})$ algorithm for the weighted optimal stable marriages problem, where K is the weight of an optimal solution [43]. The same work also provided a proof of NP-completeness of the egalitarian stable roommates problem.

The stable marriage problem is a combinatorial assignment and optimization problem. The focus of related research has historically been on the complexity and performance of solutions, identification of solutions that are not more advantageous for one gender over the other, and exploration of the problem's relationship to other combinatorial challenges. To date, solutions have relied on the equivalent of a trusted third party that accepts all users' ranked lists of preferred mates and executes an algorithm to produce stable matchings. In contrast, in TPP users select one or more specific identities with whom they may share the same wish, and the focus is on a privacy-enhanced and fair method of confirming the shared secret. Solutions to TPP and the stable marriages problem have different inputs, goals, and outcomes. Furthermore, although dating and relationships were used as the framework within which TPP is most often discussed, the shared wishes could span a variety of problem areas and need only be identity linked.

Despite the differences, it might be worthwhile future work to explore whether a privacy-enhanced and fair solution to the stable marriages problem exists, and if so, to assess its feasibility. That observation is expanded upon in the future work section.

## 2.6 Related Research Problems

Researchers have explored a number of topics that, while not specifically matchmaking problems lending themselves to direct application for TPP, are closely related to the problem at hand. This section summarizes related endeavors including secure function evaluation (SFE), zero knowledge proofs, secret/private handshakes, and private set intersection while highlighting their shortcomings in the context of TPP and ILW.

### 2.6.1 Secure Function Evaluation

To our knowledge, the Prom Problem as presented is a new research problem that has yet to be explored. However, it is closely related to the growing set of problems collectively referred to as Secure Multi-party Computation (SMC) problems. Yao is widely credited with describing the first SMC problem (along with a cryptographic solution) in [44] by presenting the Millionaires' Problem. In that problem, Alice and Bob are millionaires that would each like to know which is richer without disclosing any other information to anyone. That work was extended from two-party to multi-party computation by Goldreich, Micali, and Wigderson in [45] and many others have contributed various solutions to the Millionaires' Problem and its variants [46] [47] [48]. Goldreich later recognized in [49] that specific solutions for particular SMC problems were more efficient than general solutions.

As such, researchers have proposed various approaches to solving specific SMC problems including private information retrieval [50] [51] [52] [53] [54] [55], privacy-preserving data mining and database querying [56] [57] [58] [59] [60] [61], privacy-preserving intrusion detection [62] [63], privacy-preserving statistical analysis [64] [65], privacy-preserving cooperative scientific computation [66], as well as private bidding and auction [67] [68], among others. Researchers have also explored applications of SMC protocols for tasks such as improving the resistance of AES implementations to side-channel attacks [69] and for privacy-preserving financial data analysis [70]. Meanwhile, others have sought to study the relative efficiency of various proposed SMC protocols or to compensate for their weaknesses. For example, the Tool for Automating Secure Two-partY computations (TASTY) was developed to implement and compare protocols that are based on efficient garbled circuits, homomorphic encryption, or a combination of the two [71] while other researchers have put forth mechanisms to ensure security against a bounded number of collusions [72].

Even today, new SMC problems continue to be identified where cooperative computation combines with privacy requirements in a particular domain. Sometimes known solutions to other SMC problems can be adapted or generalized to solve the new problems. In other cases, novel, application-specific solutions are better suited for real world applications. Regrettably, the inability to fairly accomplish joint notification of identity linked wishes with equivalent exchange results in vulnerabilities similar to prior match-making protocols and related research problems in the face of threats such as inference and impersonation.

### 2.6.2 Zero Knowledge Proofs

A zero knowledge proof (ZKP) is a process whereby a Prover may convince a Verifier that she has knowledge of a statement without revealing anything beyond validation of her knowledge of said statement. The idea was initially introduced by [73] [74] and it was later shown that ZKPs exist for all languages in NP [75] [76]. The traditional example used to convey the concept is that of a cave with right and left passageways connected by a password-protected door (more precisely, Ali Baba's Cave and secret phrase "open says me") [77]. After Alice enters the cave and takes one path or the other, Bob can stand outside of the cave, flip a coin, and ask Alice to come out via the randomly selected path. If Alice does not know the secret password, the probability that she can trick Bob and "win" by emerging from his selected path is ½. Hence, with each successive attempt in which Alice succeeds, the probability P(S) that she really knows the secret phrase increases at a rate of $P(S)=1-(½)^N$ where N is the number of correct responses.

The well-known authentication protocol developed by Fiege, Fiat, and Shamir [78] is a practical example that combines this probabilistic proving concept with the difficulty of finding a square root modulo N to accomplish a ZKP-based authentication. In fact, the Fiat-Shamir protocol can offer authentication with a certain degree of anonymity – somewhat reminiscent of the conflict between goals of anonymity and authentication in TPP [79]. Unfortunately, when attempting to utilize ZKP for TPP with ILW, many of the commonly recurring drawbacks of other approaches manifest themselves. Suppose one were to use ZKP to prove knowledge of the secret "Alice loves Bob" and then use ZKP again to authenticate as Alice, Bob, or a member of the group {Alice or Bob}. In such an example, it is not obvious how one could achieve fairness. Vulnerabilities to a variety of

attacks such as inference, impersonation, and other attempts of an active attacker to cheat become apparent. While the ZKP is an elegant cryptographic primitive of great significance, ZKP alone is not suitable in the context of TPP.

### 2.6.3 Private Handshakes

Balfanz et al. revived the concept of *secret handshakes* to authenticate suspicious parties as members of a secret group while preventing attackers from learning useful information [80]. The problem was later refined to *private handshaking* in which the service provider, or group administrator, cannot trace users to arrive at more efficient implementations that might be suitable for resource-constrained environments [81]. Private handshaking protocols typically rely on group memberships (e.g., possession of some secret random number) presumably assigned by some credential authority (CA), the use of pseudonyms, direct exchanges between Alice and Bob, and the Diffie-Hellman assumption. As argued by [80], the problem of secret handshakes is fundamentally different from that of matchmaking protocols. Similarly, [23] argued that matchmaking is much more than mutual authentication of suspicious parties (as in secret handshaking). One of the primary disadvantages of this protocol in general is that it requires withdrawal of an excessive number of single-use credentials from the CA in order to maintain unlinkability. But when attempting to augment private handshaking protocols for application to TPP, a number of other critical vulnerabilities become apparent such as lack of fairness, potential for inference attacks, potential for impersonation, reliance on a trusted third party (the CA), and more. For example, to use ILW as the group membership, a user would need to prove to the CA through the registration process that they know the secret “Alice

loves Bob" and that their identity is either Alice or Bob. Revealing such information to a third party unambiguously violates the core tenets of private and privacy-enhanced matchmaking. The vulnerabilities in this context, many of which stem from fundamental differences between the problems of private handshaking and privacy-enhanced matchmaking with ILW, show that private handshaking protocols are not appropriate for TPP.

### 2.6.4 Private Set Intersection

Just as it sounds, the problem of private set intersection (PSI) involves determining the intersection of users' private sets (the elements they have in common), without the participants learning anything about the other elements of their respective sets. Relatively efficient approaches to PSI have been proposed for a security model with semi-honest participants and there are extensions that strive to accomplish security with malicious clients and servers. For example, [82] proposed PSI based on oblivious polynomial evaluation using set elements as roots of a $v$-degree polynomial $f = \prod_{i=1}^{v}(t - c_i) = \sum_{i=0}^{k} \alpha_i t^i$. Alice encrypts the coefficients with her public key for an additively homomorphic cryptosystem and sends to the other participant (e.g., to Bob or to a Server in a client-server model). Given Bob's set, $B$, he then homomorphically evaluates $f$ for each $b_j \in B$ where $b_j = 0$ if that element is in the intersection of Bob's and Alice's sets. Finally, Bob chooses random numbers $r_j$ and returns $u_j = E(r_j f(s_j) + b_j)$ from which Alice can determine the intersecting elements. Others such as [83] and [84] have developed alternative approaches, efficiency improvements, and demonstrated relative performance of PSI protocols. Common threats to PSI protocols include set intersection at-

tacks, element detection determining elements in a private set, and dictionary attacks. However, additional challenges are encountered if considered for application to TPP. In most, if not all, cases participants communicate directly with one another, leading to vulnerability to inference attacks. Furthermore, since mutual PSI protocols are typically constructed via two instantiations of one-way PSI, they cannot accomplish fairness due to the early termination attacks and potential for falsification of results. Finally, PSI protocols fundamentally do not support ILW and they lack the ability to achieve the conflicting goals of anonymity and authentication of ILW in a privacy-preserving manner.

### 2.6.5 Interlocked Private Set Intersection and Private Handshakes

Duan proposed interlocking of a secret handshake (SH), also known as private handshakes, and private set intersection (PSI) protocols to build privacy-preserving systems not possible with either protocol alone [85]. One proposed interlocked protocol executes PSI to match elements followed by SH to authenticate credentials. The second proposed protocol first executes SH to verify a shared secret group credential and then derives a common value for use as a homomorphic "random" number for their PSI protocol based on partial homomorphism. However, regardless of the ordering of SH and PSI protocols, this approach does not afford security and privacy in TPP or other contexts using ILW. For example, since neither protocol provides fairness, then either ordering is vulnerable to cheating such as early termination and falsification of results. Consider an example where Alice and Bob first execute PSI to confirm matching elements and then engage in an SH protocol. At the end of the SH protocol, Bob could intentionally send a value that will fail verification resulting in Bob having gained an unfair advantage (he has verified

the secret while Alice has learned nothing). Alternatively, consider an example where Alice and Bob first execute the SH protocol. Assuming neither party cheats at this stage, they will have confirmed each other's membership in the same group. With ILW, that group would only consist of Alice and Bob making it vulnerable to inference attacks. If the prom were quickly approaching, Bob could easily infer that Alice is most likely inquiring about the prom. Moreover, this consideration of groups highlights another major problem – the credential authority (CA) effectively becomes a trusted third party in this case. If the CA were compromised (or the CA itself were malicious), it could reveal evidence directly linking Alice and Bob thereby compromising their privacy. It could also issue an adversary Trudy a credential for the group containing only Alice and Bob enabling her to successfully complete the SH and PSI protocols, regardless of the ordering, by simply guessing the wish "want to attend prom together." The interlocked SH and PSI protocols proposed by Duan might be useful in scenarios where Alice wishes to "vote yes if any other member of my party votes yes," but it is insufficient in TPP where Alice wishes to "vote yes if and only if Bob will also vote yes."

### 2.6.6 Privacy Preserving Profile Matching

In the problem of *privacy preserving profile matching* (PPPM), user profiles consist of groups of attribute values that are often represented as vectors or sets. Profiles are considered to contain at least some information that a user might want to keep private. For example, a typical profile might include hobbies, lifestyle details, interests, gender preferences, contacts, and the like. The context most often used to present PPPM is that of profile matching in mobile social networks in settings such as airports, dance clubs,

hospitals, or large entertainment events. A majority of PPPM related research can be divided into either approaches dealing with coarse-grained profiles or approaches dealing with fine-grained profiles. Coarse grained profiles are often represented as Boolean sets or vectors and associated solutions employ *private set intersection* (PSI) or *private cardinality of set intersection* (PCSI) [82] [83] [84] [86] [87] [88] [89] [90]. Meanwhile, fine grained profiles are typically represented as vectors of integers or real numbers and associated solutions often employ *private dot-product* (PDP) computations [91] [92] [93] [93] [94]. The latter group most often employs homomorphic encryption in the form of the Paillier cryptosystem [95], a primary enabler.

Such systems for PPPM are typically asymmetric leading to lack of fairness as the two parties learn differing amounts of information about the result. As an example, Zhang et al. defined three levels of privacy corresponding with different asymmetries with respect to the information learned by each party and presented protocols targeting each privacy level [93]. For instance, in their *Level-II privacy* definition, Alice learns *f(u,v)* while Bob learns nothing. Similarly, in the highest level of privacy (*Level-III privacy*), Alice learns *f(u,v)* exceeds her matching threshold while Bob again learns nothing.

Moreover, similar to other SMC methods, most protocols to date are vulnerable to early termination attacks, which have also been referred to as runaway attacks in the context of PPPM [96]. The reliance on fully honest or semi-honest participants is not realistic for many contexts including for TPP. Zhu, Du, Li, and Gao recognized this lack of fairness as a major concern for privacy-preserving friend matching systems and introduced a method for accomplishing PPPM using blind vector transformations [96]. Unfortunately, the protocol relies on a trusted third party called the verifier to enable fair-

ness. Additionally, the initiating user begins by encrypting her profile with her public key and sending the result to the other matching participant. The line of thinking was likely that only the initiating user could decrypt it to discover the profile attribute values. However, this approach is vulnerable to dictionary style attacks since other users and third parties could guess profile values and encrypt with the initiator's public key to see if the result is a match. The smaller, more predictable, or generally lower entropy the profile attribute space is, the more severe this vulnerability would become.

Solomon, Xu, and Zhang also recognized the unfairness of PSI based protocols and particularly sought to prevent cheating by lying about profile attributes via the use *of authorized private set intersection* (APSI) [97]. In their scheme, authenticated users are required to present profile attributes to a certification authority (CA) for signing. While it accomplishes the stated goal of preventing malicious users from using arbitrary sets of profile attributes, there are other privacy and security related shortcomings. For example, the CA constitutes a trusted third party, profile attributes are not truly private, and there is no level of anonymity or unlinkability available. Moreover, it does not mitigate other asymmetries of PSI based protocols such as users learning differing amounts of information about the relationship of the sets and vulnerability to early termination attacks.

In some cases, researchers have endeavored to accomplish profile matching goals in ways that do not generally fall into the PSI or PDP categories. For instance, Liang et al. described two-party protocols for *explicit Comparison-based Profile Matching* (eCPM), *implicit Comparison-based Profile Matching* (iCPM), and *implicit Predicate-base Profile Matching* (iPPM) to accomplish profile matching with certain definitions of anonymity [98]. The protocol relies on a Trusted Central Authority for bootstrapping, profile gener-

ation, and pseudonym assignment, along with generation and assignment of user certificates. The primary threat model involves semi-honest users and curious, passive, but possibly colluding attackers on the network observing traffic with intent to learn about users' profiles. The protocols combine homomorphic encryption with an auto-regressive moving average model to accomplish profile matching that satisfies the specified definitions of anonymity. Their definitions of anonymity actually represent the confidentiality of profile data and *fully anonymous* is described as the condition in which, after executing multiple instances of the protocol with a user, the attacker cannot determine the profile of the other user beyond simply guessing from the set of possible profile values. The eCPM protocol allows users to compare a specific attribute without disclosing the actual value and only the initiator finds out the comparison result. With iCPM, the initiator can receive some message from the responder rather than the result of the comparison. Finally, iPPM is a variant that allows comparison of multiple attributes rather than a single attribute. Beyond simply associating anonymity with confidentiality of private profile data, participants communicated directly with one another which can lead to inference attacks and potentially enable straightforward de-anonymization. Moreover, in addition to the partial reliance on a trusted third party and the assumption of honest or semi-honest participants, the protocols are unfair in that different amounts of information are learned by each participant.

The work of Zhang, Li, and Liu summarized deficiencies of prior protocols such as vulnerability to early termination attacks, unfairness or asymmetry (referred to as *unverifiability*), reliance on a trusted third party, and computational costs that were perceived to be too expensive. The authors went on to describe an alternative approach to privacy

preserving friend matching in social networks [99]. The approach treats one or more attribute values as the secret with which to encrypt a message that includes a session key for later communication. It is then hypothesized that only those with matching profile values would be able to efficiently open the "message in a sealed bottle" to obtain the session key and further communicate with the initiator. This approach strongly resembles that of the previously proposed Zhang/Needham (ZN) private matching protocol [23]. It also shares the vulnerability to dictionary style attacks and it cannot prevent impersonation attacks.

Finally, Li et al. argued that prior systems that are based on the Paillier Cryptosystem [95] exhibited prohibitively high computational overhead, that prior protocols were unfair or asymmetric (referred to as not verifiable), and they presented a perturbation-based protocol for fine-grained profile matching using random data, vector dot-products, and $\ell_1$ (Manhattan) distance computations [100]. The scheme assumes a semi-honest security model and requires the use of cooperative helpers to execute the protocol. To begin the protocol, Alice would generate a random vector $r$ that is the same length as the private profile vector $u$. Alice then computes $x = u + r$, sends the resulting $x$ to Bob, and sends $r$ to the cooperating Helper. Note the reliance on an honest Helper that plays the role of a trusted third party (TTP) since the private profile vector $u$ could be directly computed given knowledge of $x$ and $r$. The protocol also relies on the Helper role to behave honestly and to fairly communicate accurate results to each of the parties. Furthermore, the protocol must assume encrypted communications. Otherwise, a passive eavesdropper could

also readily compute the private profile data. Consequently, the authors call out 128-bit AES for end-to-end encryption with the protocol.

In summary, PPPM problems attempt to match users based on profiles of attributes that express preferences and interests according to certain matching criteria like a threshold number of equivalent attributes. The majority of PPPM protocols either employ PSI for coarse-grained matching or PDP for fine-grained matching, although other approaches have been proposed with different tradeoffs. Most prior PPPM protocols are unfair in that there are clear asymmetries with respect to the amount of information learned by each participant. The few protocols that do attempt to accomplish some definition of fairness employ elements that equate to TTPs and only offer partial fairness in the face of specific threats in most cases. While PPPM results in pair-wise matchings of users, the inputs and goals of PPPM are different from TPP, which requires fairness and support for private matching of ILW. Other common weaknesses of PPPM protocols in the context of TPP include vulnerability to inference attacks when users communicate directly with one another, impersonation, dictionary style attacks, and vulnerability to early termination attacks.

## 2.7 Related Work Summary

After Baldwin and Gramlich introduced the conflict between anonymity and authentication in matchmaking challenges, a number of notable efforts have attempted to solve variations of matchmaking challenges with differing privacy and security requirements. Table 2.1 compares key security properties of the matchmaking protocols most closely related to the present work. Additionally, related research problems include secure func-

tion evaluation, zero knowledge proofs, private handshakes, private set intersection, and combinations thereof. The prior art matchmaking protocols, as well as related research problems, have fundamental shortcomings when applied to TPP and challenges involving ILW. Among others, the list of deficiencies includes:

- Lack of wish secrecy (lack of wish unlinkability)
- Lack of anonymity (lack of user unlinkability)
- Lack of fairness (lack of joint notification of wishes with equivalent exchange)
- Lack of forward privacy of wishes and identities
- Reliance on direct communication between users (vulnerable to inference)
- Reliance on a trusted third party
- Vulnerability to dictionary-style attacks
- Failure to support privacy preservation with ILW
- Vulnerability to inference, impersonation, data compromise, and/or early termination attacks

The unsuitability of the various approaches led to the development of the Horne-Nair (HN) protocol for fair and privacy-enhanced matchmaking with identity linked wishes. The HN protocol, which is described in more detail in Chapter 3, does not rely on direct communication between users thusly avoiding inference attacks. It instead leverages an untrusted third party. The protocol uses a gradual release process affording fairness while resisting attempts at early termination attacks. As for secrecy of wishes and anonymity, the protocol does not include identities or pseudo-identities in any messages, nor

are they stored in the third party's database. In fact, throughout execution of the protocol, users stay in control of their private or sensitive data and independently compute all intermediate and final values required for successful confirmation of shared ILW. The design of the HN protocol, as well as the proposed privacy-enhanced and fair matchmaking system, concentrated on security and privacy preservation when confronted with both passive and active adversaries. The proposed matchmaking method and system satisfy the required security properties for TPP with ILW while avoiding the shortcomings of prior matchmaking protocols and related research endeavors in this context.

# CHAPTER 3

# A SOLUTION TO THE PROM PROBLEM

TPP represents a special class of matchmaking challenges embodying the need for fair and privacy-enhanced matchmaking with ILW. The Ashley Madison hack of 2015 was a harsh reminder that the ramifications of security failures in matchmaking problems can be severe. The Horne-Nair (HN) protocol was put forth as a candidate solution to TPP [1] [101]. In the HN protocol, neither identities nor pseudo-identities are included in protocol messages or stored in a matchmaker's database. Users do not interact directly to combat inference attacks, yet the protocol avoids reliance on a trusted third party. Another important facet of the HN protocol is that sensitive data stay in control of the client at all times. Furthermore, the protocol employs a gradual release process to ensure that a malicious user could not gain a sizeable advantage, and the protocol is flexible enough to leverage external identity management systems, encryption key management systems, and more. The present chapter provides multiple perspectives on the protocol details to facilitate design and development of a privacy-enhanced and fair matchmaking system that supports ILW.

## 3.1 The Horne-Nair Protocol

The overview of the system for fair and privacy-enhanced matchmaking in Figure 3.1 serves as an example of the context for the protocol. In the client-server architecture,

clients communicate with the matchmaker via a means such as the Internet, a local area network (LAN), and optionally by way of an anonymity network. All private or sensitive data is managed on the client-side, thusly eliminating a central repository aggregating sensitive information, as is common in many comparable systems, and that could lead to a major compromise of privacy. Fundamentally, the HN protocol consists of a challenge, counter-challenge, independent computation of a verification value, and an alternating gradual release process utilized in a manner that affords anonymity and authentication of ILW while preserving the privacy of the users. Direct communication between users would introduce vulnerability to inference attacks. Hence, the system employs a third party called the matchmaker. The matchmaker is untrusted and akin to the public database of the ZN protocol [23], and also comparable to broadcast communication. The

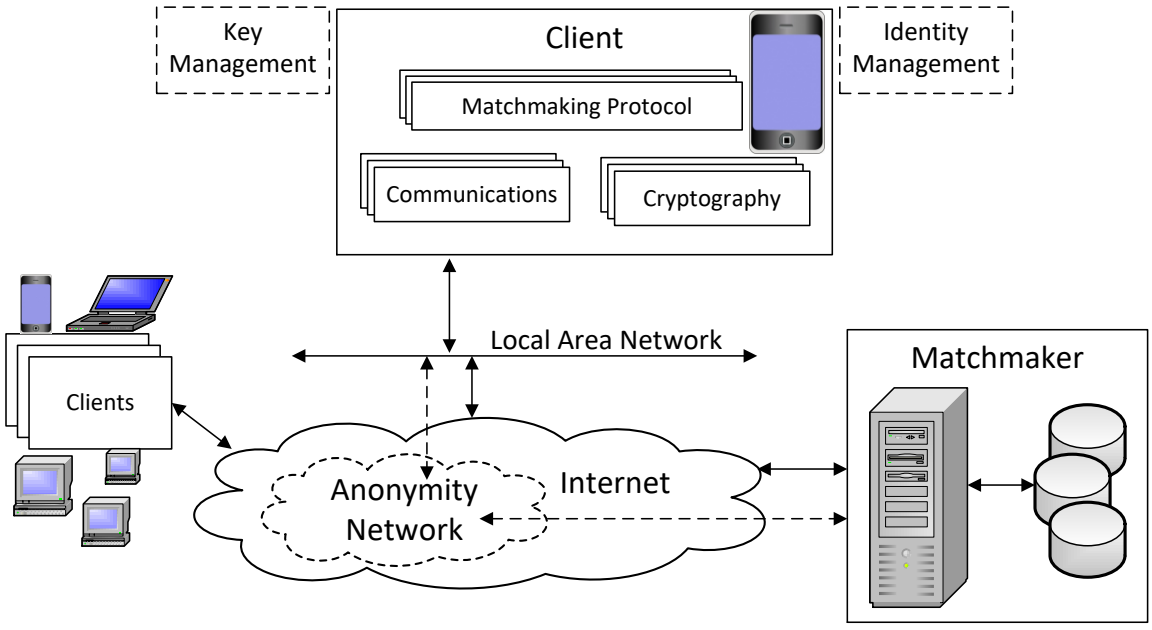



Figure 3.1. Overview of fair and privacy-enhanced matchmaking system

HN protocol is anonymous given that neither identities nor pseudo-identities are components of any messages. Therefore, identities are never revealed to the matchmaker and they are never stored in its database. Additionally, users may communicate over anonymous communication channels such as virtual private networks (VPNs) or onion routing networks to defend against traffic analysis attacks. In the sub-sections that follow, the protocol is detailed including preliminaries, overview and details of the protocol, a pseudo-code embodiment that is algorithm agnostic, and specifics of an RSA-based implementation.

### 3.1.1 Preliminaries

A number of concepts, definitions, and descriptions are important prerequisites to a complete understanding of the proposed protocol. As such, preliminary concepts including key privacy, the untrusted matchmaker, anonymous communications, groups, and applicable notation are now discussed in turn.

**Key Privacy** – The term *key privacy* was used in [102] to refer to the security requirement that, given a ciphertext, it is infeasible for someone to determine which public key was used to create it. In that work, Bellare et al. demonstrated that ElGamal and some variants of RSA exhibit key privacy. This property, which can also be considered a form of recipient anonymity, is an important consideration during algorithm selection and implementation of the HN protocol.

**Untrusted Matchmaker** – In the security model of TPP, all parties are considered untrustworthy. Consequently, the HN protocol does not rely on a TTP, but instead leverages an untrusted matchmaker. That matchmaker is a publicly available database, alt-

hough access may be via Application Programming Interface (API) and some implementations may restrict access to anonymously authorized users of the system. In many ways, the untrusted matchmaker of the HN protocol resembles that of the ZN protocol [23]. The matchmaker of the HN protocol is also expected to have some level of integrity, but a lack of integrity would not result in a privacy or security compromise [1]. The matchmaker is not trusted with users' wishes, users' identities, or with the responsibility of enforcing fairness, which mitigates a number of potential vulnerabilities and contributes to the HN protocol's suitability for TPP.

**Anonymous Communications** – The HN protocol itself affords a certain level of anonymity given that neither identities nor pseudo-identities are ever included directly in any messages or stored in the database. However, as with most any online protocol, traffic analysis attacks and attempts to unmask and associate users remain realistic threats. In circumstances where anonymity, unlinkability, and identity concealment in general warrant additional measures, the use of anonymous communication mechanisms may be common. A number of protocols for anonymous communication have been proposed and implemented such as [103] [104] [105] [106] [107] [108]. Two examples of commonly used approaches to achieving some level of anonymity include the use of Virtual Private Networks (VPNs) and onion routing networks such as Tor [104].

**Groups** – In matchmaking protocols, the ability to locate other users with common wishes is an important challenge in the context of other privacy and security requirements of TPP. In many approaches, this results in an all-to-all communication problem that is infeasible in practical applications. Zhang and Needham recognized this and attempted a solution in the way of using the output of a one-way hash function as an encryption key

with the ZN protocol [23]. Unfortunately, that approach proved vulnerable to dictionary attacks with the threat being amplified by a small wish space. In the HN protocol, although not strictly required, the concept of *groups* may be incorporated to limit the population of users that will receive a given challenge. With this scheme, users are assumed to belong to various groups such as "all male members of Alice's senior class", "all students within a 20 mile radius", random sets of users, and so on depending on the context. Note that for privacy preservation, it is important that groupings be sufficiently large. Group management may be handled entirely external to the matchmaking system. In that case, users could anonymously poll the matchmaker for challenges using the group identifier. In this way, the population receiving each challenge may be limited while the system need not know anything about group membership or user identities.

Table 3.1. Examples of notation used

| | |
|---|---|
| $PU_A$ | Alice's public key |
| $PR_A$ | Alice's private key |
| *E(K, P)* | Encrypt plaintext P with the specified key K |
| *D(K, C)* | Decrypt ciphertext C with the specified key K |
| $R_A$ | Alice's random data (nonce) |
| *H(X)* | One-way hash function applied to input X |
| *X+Y* | Concatenation of X and Y |
| $e_{AL}$, $d_{AL}$, $n_L$ | Components of Alice's "long" RSA key |
| $e_{BS}$, $d_{BS}$, $n_S$ | Components of Alice's "short" RSA key |

**Notation** – The notation used was adapted from that of [109], a common reference text in undergraduate and graduate courses studying cryptography and network security. Some examples of the notation used to communicate details of the protocol appear in Table 3.1.

### 3.1.2 Protocol Overview

An execution of a sample embodiment of the HN protocol involves the following steps which are further described below [3].

1. **Generate Challenge**. Alice selects a user $U$ (e.g., Bob), $G_i \in G \mid ID_U \in G_i$, and her nonce $R_A$. She then computes and sends $X := E(PU_B, w + E(PR_A, R_A))$ to M.
2. **Receive Challenge**. Bob receives challenge $X$ (e.g., by anonymously querying M with $G_i$), computes $\{w+F\} := D(PR_B, X)$, and chooses user $U'$ with whom he may share generic wish $\boldsymbol{w}$.
3. **Generate Counter-Challenge**. Assuming $U'$ is Alice, Bob sends $Y := E(PU_A, E(PR_B, R_B))$ to M.
4. **Receive Counter-Challenge**. Alice queries M with $X$ and receives counter-challenge $Y$.
5. **Compute Verifier.** Based on information available to each user, Alice computes verifier $V_{AB} := H( H(R_A) + H(D(PU_B,D(PR_A,Y)) ) )$ and Bob computes verifier $V_{AB} := H( H(D(PU_A, F)) + H(R_B) )$.

6. **Gradual Release of Verifier.** Aside from the first bit, $\forall\ b_i \in V_{AB} \mid 0 = i\ (mod\ 2)$ Alice releases $b_i$ after Bob releases $b_i$-*1* and $\forall\ b_j \in V_{AB} \mid 1 = j\ (mod\ 2)$ Bob releases $b_j$ after Alice releases $b_j$-*1*.

Initially, Alice has a generic wish $\boldsymbol{w}$ such as "attend prom together" that will be indirectly linked to the identities of Alice and Bob via the protocol resulting in ILW. Anyone can guess ILW such as this, but the veracity of such wishes would be unknown. For example, Trudy may be able to prove that she knows that "Alice and Bob want to attend prom together", but that does not prove anything about the veracity of the statement. The goal is for precisely Alice and Bob to anonymously locate each other and authenticate the wishes in a privacy-enhanced and fair manner, thusly validating the ILW. To begin the process, Alice generates random data $R_A$ and she selects the $G_i$ of a group of which Bob is a member (e.g., all male members of her senior class). She encrypts $R_A$ with $PR_A$ and encrypts the result appended to generic wish $\boldsymbol{w}$ with $PU_B$ yielding challenge value $X$. The challenge $X$ is submitted anonymously to the matchmaker M along with the $G_i$.

Note in step 1 that, although the idea of encrypting with one's private key may seem counter-intuitive or non-standard, in this context it can be considered a form of rudimentary signature with recovery and *key privacy* [102]. This is a key contributor to achieving the conflicting goals of authentication and anonymity with ILW. In step 2, Bob receives challenge $X$ (e.g., via anonymously querying M with $G_i$) and decrypts it with $PR_B$ to reveal generic wish $\boldsymbol{w}$ and encrypted random data $F$ (which is equal to Alice's $E(PR_A,R_A)$). The other members of the group $G_i$ might also receive $X$ and try to decrypt it with their private keys, but this would yield random data and the process would terminate. In step

3, Bob chooses user *U'* with whom he may share secret ***w***, selects his random data $R_B$, encrypts $R_B$ with $PR_B$, and then encrypts the result using $PU_{U'}$ yielding the counter-challenge value *Y*. Note that since the counter-challenge *Y* is associated with the original challenge *X* that included ***w***, generic wish ***w*** is an optional component of counter-challenge.

To complete this phase of the process, Bob sends *Y*, the counter-challenge associated with challenge *X*, to M. Next, in step 4, Alice receives counter-challenge *Y* (e.g., by querying M with *X*). At this stage, each participant has the data required to independently compute verifier $V_{AB}$ as delineated in step 5. The resulting value can then be used for confirmation of the shared ILW. In the steps listed, generic wish ***w*** was not included with the counter-challenge. If *Y* had instead included ***w***, then Alice would have computed $\{w,L\} := D(PR_A,Y)$ and $V_{AB} = H(H(R_A)+H(D(PU_B,L)))$. At the conclusion of step 5, if Bob had chosen Alice as user *U'*, then Alice and Bob would have independently computed the same value for $V_{AB}$.

Lastly, step 6 describes the process for confirmation of the verification value via anonymous, alternating gradual release of the bits of $V_{AB}$ to the public database of M. Upon the first non-matching bit, the process could terminate and neither of the parties would have a sizeable advantage, and thusly neither would have learned any useful information. However, security can be improved by using random bits to complete the process. The use of such random decoy bits is assumed to be used from this point forward. It follows from this process that successful verification of the shared ILW requires involvement of public and private keys of each participant in addition to random data, also known as a nonce, originating from each participant. The concept of fairness is dis-

cussed in the detailed protocol description of section 3.1.3. Its relationship to user confidence in the matching result and an approach to quantifying the fairness of executions of the HN protocol are presented in section 3.2.

### 3.1.3 Detailed Description

A pseudocode example summarizing one of the simplest embodiments of the HN protocol as originally presented in [1] is depicted in Figures 3.2-3.4. The protocol is divided into Algorithm 1a of Figure 3.2 representing the initiator's actions (e.g., Alice), Algorithm 1b of Figure 3.3 representing the responder's actions (e.g., Bob), and Algorithm 2 of Figure 3.4 representing the gradual release process that is executed by both participants. The subsequent detailed description mirrors the order of the steps from the high level protocol overview of section 3.1.2.

**Algorithm 1a: AttemptMatch (Initiator:Alice)**

**Input** : Generic wishes w ∈ *W*, max error ε
**Output** : True if ILW match, false otherwise

**1** /*** Compute Challenge ***/
**2** Choose user U | $ID_U \in U$
**3** GID ← $G_i \in G$ | $ID_U \in G_i$
**4** $R_A$ ← GetCryptoRandom()
**5** C ← Encrypt($PR_A$, $R_A$)
**6** X ← Encrypt($PU_U$, w+C)
**7** Send(*out* GID, *out* X)
**8** /*** Respond to Counter Challenge ***/
**9** Receive(*out* X, *in* Y) is false )
**10** L ← Decrypt($PR_A$, Y)
**11** N ← Decrypt($PU_U$, L)
**12** P ← H(N);
**13** $V_{AB}$ ← H( H($R_A$) + P );
**14** /*** Confirmation Phase ***/
**15** result ← GradualRelease(*out* $V_{AB}$, *out* ε)
**16** return result;

Figure 3.2. Algorithm 1a pseudocode for one embodiment of the HN protocol [1]

**Generate Challenge (Algorithm 1a, lines 1-7)**. This code segment corresponds with step 1 from the protocol overview. Alice begins by choosing user $U$ (e.g., Bob in the example scenario) with whom she may secretly share generic wish $\boldsymbol{w}$. Alice also selects a group that the user is a member of, such as all persons within a 20 mile radius, and she generates a cryptographically random number, often referred to as a nonce, denoted $R_A$. She then encrypts $R_A$ with $PR_A$ resulting in intermediate value $C$. She then utilizes $PU_U$ to encrypt $C$ and generic wish $\boldsymbol{w}$ yielding the challenge value $X$. This accomplishes the following three essential tasks:

1. It associates Alice and Bob with the generic wish $\boldsymbol{w}$, yielding ILW.
2. It anonymously authenticates the challenge as having originated from Alice.
3. It affords confidentiality. That is, only Bob could feasibly unlock the challenge and successfully complete the protocol.

Also important to note is the fact that, given the key privacy property, it is infeasible for Bob to determine the sender via another method besides successfully completing the protocol with matching ILW. With all necessary values selected or computed, Alice then completes the first step by anonymously sending the GID and $X$ to the matchmaker M. Note that although the group identifier is not a requirement of the protocol, it is used here as a convenient enabler of anonymous communication with the intended recipient. This is an approach one may employ to maintain recipient anonymity while avoiding all-to-all communication problems of protocols using mechanisms comparable to broadcast.

**Query for Challenge (Algorithm 1b, lines 1-6).** This code segment corresponds with step 2 from the protocol overview. At some point, M provides the challenge to the recipients that are members of $G_i$ including Bob. This could be accomplished via push

notification, or provided in response to polling with $G_i$ as is implied by the Receive method and its arguments in the second line of the Algorithm 1b pseudocode. When the members of $G_i$, receive challenge $X$, they will decrypt it with their private keys. For all users apart from the intended recipient, this will result in random data and the protocol terminates. But when Bob decrypts $X$ with his private key $PR_B$ it reveals generic wish $\boldsymbol{w}$ and encrypted random data $F$ (equivalent to Alice's value $C$). Bob then considers with whom he might share generic wish $\boldsymbol{w}$ and chooses user $U'$ (that might or might not be Alice). Bob subsequently decrypts $F$ with $PU_{U'}$ resulting in $G$. If the selected user $U'$ is Alice, then Bob's computed value $G$ would be equivalent to $R_A$. However, he has no way to determine whether he chose correctly at this stage or not. Finally, Bob computes $J$ as $H(G)$, which will be a necessary component of the verification value $V_{AB}$.

**Algorithm 1b: AttemptMatch (Responder:Bob)**

**Input** : None
**Output** : True if ILW match, false otherwise

```
1   /*** Respond to Challenge ***/
2   Receive(out GID, in X)
3   w+F ← Decrypt(PR_B, X)
4   Choose user U' | ID_U' ∈ U
5   G ← Decrypt(PU_U', F)
6   J ← H(G)
7   /*** Compute Counter Challenge ***/
8   R_B ← GetCryptoRandom()
9   K ← Encrypt(PR_B, R_B)
10  Y ← Encrypt(PU_U', K)
11  Send(out X, out Y)
12  V_AB ← H( J + H(R_B) );
13  /*** Confirmation Phase ***/
14  result ← GradualRelease(out V_AB, out ε)
15  return result
```

Figure 3.3. Algorithm 1b pseudocode for one embodiment of the HN protocol [1]

**Generate Counter-Challenge (Algorithm 1b, lines 7-12).** This code segment corresponds with step 3 from the protocol overview. Bob now prepares a counter-challenge associated with $X$ by generating his random data $R_B$ and encrypting it with his private key $PR_B$ yielding $K$. Bob then encrypts $K$ with $PU_U$ so that $U'$ is the only user that will be able to respond correctly, ultimately resulting in a matching $V_{AB}$. Because Bob's counter-challenge is associated with $X$, Alice's initial challenge that itself included $\boldsymbol{w}$, it is not necessary for Bob to include $\boldsymbol{w}$ with the counter-challenge. However, if an implementation did include $\boldsymbol{w}$, then the series of computations used to calculate $V_{AB}$ varies slightly as discussed previously. Finally, Bob sends the original challenge $X$, along with counter-challenge $Y$, to M. At this point, Bob also has all of the data necessary to compute the verification value $V_{AB}$ as in line 12 of Algorithm 1b by concatenating $J$ with $H(R_B)$ and applying a one-way hash function to the result.

**Query for Counter-Challenge (Algorithm 1a, lines 8-13).** This code segment corresponds with step 4 from the protocol overview, and it represents the final phase prior to attempts at confirmation of the verification value. Alice queries M with the original challenge $X$ and receives the associated counter-challenge $Y$. She then decrypts it with her private key $PR_A$ yielding intermediate value $L$. Next, she decrypts $L$ with $PU_U$ (i.e., $PU_B$ if she chose Bob) yielding $N$. After calculating $H(N)$ to obtain $P$, she has all of the information needed to compute the verifier $V_{AB}$ by applying a one-way hash function to the result of concatenating $H(R_A)$ with $P$.

**Gradual Release of Verifier (Algorithm 1a, lines 14-15; Algorithm 1b, lines 13-14; Algorithm 2).** Upon entry to this phase of the protocol, Alice and Bob would have independently computed the same value for $V_{AB}$ if they share the same ILW. In the last

lines of Algorithms 1a and 1b prior to returning the final result, the users gradually disclose their confirmation value one bit at a time in an alternating fashion. Pseudocode for one straightforward embodiment of this operation appears in Figure 3.4. The initiator of the gradual release process would provide bits $\{b_0, b_2, \ldots b_{N-2}\}$ while the other user would provide bits $\{b_1, b_3, \ldots b_{N-1}\}$. The probability of matching ILW increases with each matching bit during the process.

**Algorithm 2: GradualRelease**

**Input** : Verifier $V_{AB}$, max error ε, Boolean first
**Output** : True if match with >= (1-ε) confidence, false otherwise

```
1   index ← if first then 1 else 2
2   match ← true
3   while index < max and error > ε
4       if first then
5           value ← match ? V_AB[index] : Rand( )
6           Release(out index, out value)
7           QueryBit(out index+1, in next)
8           match ← next == V_AB[index+1]
9       else
10          QueryBit(out index, in next)
11          match ← next equal to V_AB[index]
12          value ← match ? V_AB[index] : Rand( )
13          Release(out index+1, out value)
14      error ← match ? 0.5^⌊(index+1)/2⌋ : 1
15      index ← index + 2
16  return error < ε
```

Figure 3.4. Algorithm 2 pseudocode for one embodiment of the HN protocol [1]

In the event that it was a non-match, a participant could terminate the protocol upon the first incorrect bit and neither party could have obtained a sizeable advantage. As mentioned in the overview, a more secure alternative might be to complete the process with

random bits. Such an approach using decoy bits is reflected by the pseudo-code of Figure 3.4, and it is likely to be utilized in most real-world implementations.

For confirmation values of a reasonable length (e.g., $|V_{AB}| = 512$ bits), the gradual release process may at first seem overly complex considering potential communication overhead, particularly when compounded with the overhead of anonymous communication channels. That concern was identified for further investigation as described in later sections describing the experimental methodology and results of testing. But for now, consider that satisfactory confidence in the matching result could be achieved with far fewer bits, and consequently with far fewer rounds of communication, for most practical situations. The probability of guessing a single correct bit is ½. Therefore, given the number of correct bits released N, error ε, and confidence λ=(1-ε), then the confidence of each user can be calculated as $\lambda=(1-(½)^{\lceil N/2 \rceil})$ and $\lambda=(1-(½)^{\lfloor N/2 \rfloor})$. Note that the subtle difference for the two equations is the ceiling versus floor operators. Consequently, if one desired 99.6% confidence, it could be achieved by gradually releasing 8 matching bits per person, or 16 total bits. Furthermore, >99.99% confidence in the matching result can be achieved with 14 bits contributed from each user totaling 28 total bits released. The pseudocode of Figure 3.4 enables tuning of the tradeoff between performance and confidence in the matching result by factoring in the desired confidence to determine the number of bits to be released.

The HN protocol was initially presented in a cryptographic algorithm agnostic manner to avoid unnecessarily limiting the potential of such a system and method [1] [3] [101]. Given the continually evolving nature of the computer security industry, and the

incessant growth in computational power, an important feature is the avoidance of an immutable reliance on one specific algorithm that could extend the life of the technology beyond the life of any particular cryptographic algorithm that relies on a security model of computational hardness.

Upon practical analysis, or attempts to implement the HN protocol, one quickly arrives at the realization that two pairs of public keys with different sizes are required. Notice that the protocol incorporates encapsulation of encrypted results. As an example, consider an implementation using the well-known RSA algorithm [110]. If the outer layer of encryption utilized a 4,096 bit modulus yielding 512 byte ciphertexts, then the "smaller" key pair might use a 2,048 bit modulus yielding 256 byte ciphertexts to support encapsulation as used in the protocol. The concatenation of more data with a ciphertext output with the smaller modulus implies that a larger modulus is required to encrypt the outer package in which it is then encapsulated. Another example might be using 3,072 and 6,144 bit key pairs. In addition to improved security, increased key sizes afford the opportunity to allocate more bytes to the (encoded) secret, the random data, or both.

Consider now a particular implementation of the HN protocol as described in [4] that is based on the RSA algorithm for the asymmetric cryptography [110] and SHA3-512 [111] for the one-way hash function. The components of the "large" key pair belonging to Bob can be represented as $e_{BL}$, $d_{BL}$, and $N_L$. Likewise, the components of the "small" key pair belonging to Alice can be denoted $e_{AS}$, $d_{AS}$, and $N_S$. Therefore, an execution instance of the HN protocol in which Alice and Bob choose each other as the identities linked to generic wishes such as "attend prom together" might be manifested by the following steps. Note that knowledge of users' public keys is assumed. This could be ac-

complished in any of several ways such as a peer-to-peer web-of-trust resembling that of PGP [112], a new or established public key infrastructure (PKI), or an alternative third party key management system.

**Generate Challenge**. In the first phase, Alice generates her nonce $R_A$ and then computes challenge *X* as follows where ***w*** is a generic wish such as "attend prom together". The resulting challenge is sent to matchmaker M associated with the group identifier $G_i$. (e.g., the group could be all male students in Alice's high school).

$$X = (w + (R_A)^{d_{AS}} mod\ n_S)^{e_{BL}} mod\ n_L$$

**Receive Challenge**. Next, Bob queries M anonymously by providing $G_i$ and M replies with the challenge value *X*. Bob uses $PR_{BL}$ to decrypt the challenge yielding generic wish ***w***, and encrypted random data *F*. He then chooses a user with whom he may share ***w***. In this instance, he chooses Alice, and composes counter-challenge *Y*.

$$\{w + F\} = (X)^{d_{BL}}\ mod\ n_L$$

$$Y = ((R_B)^{d_{BS}}\ mod\ n_S)^{e_{AL}}\ mod\ n_L$$

**Receive Counter-Challenge and Computer Verifier**. At some point in time, Alice queries M with *X* and receives counter-challenge *Y*. Now the participants have the necessary values to independently calculate the confirmation verifier $V_{AB}$. Alice and Bob can compute the verifier in the following way, where *T(x)* denotes truncation of input parameter *x*. Note that the $V_{AB}$ values will be equal if and only if Alice and Bob share the same ILW.

$$Alice: V_{AB} = H(H(R_A) + H(((T(Y))^{d_{AL}}\ mod\ n_L)^{e_{BS}}\ mod\ n_S))$$

$$Bob: V_{AB} = H(H((F)^{e_{AS}}\ mod\ n_S\ ) + H(R_B))$$

**Gradual Release**. Having independently computed $V_{AB}$ values for confirmation, Alice and Bob gradually reveal the bits of $V_{AB}$ in an alternating fashion to ensure that neither party can obtain a sizeable advantage and ensure joint notification of ILW with equivalent exchange. Given the notation $Rel_A(b_i)$ that represents Alice's release of the $i^{th}$ bit, one of the simplest embodiments of the gradual release process can be described as follows.

$$\forall b_i \in V_{AB} \mid (0 = i \bmod 2) \wedge i \neq 0,\ Rel_A(b_i) \because Rel_B(b_{i-1})$$

$$\forall b_j \in V_{AB} \mid (1 = j \bmod 2),\ Rel_B(b_j) \because Rel_A(b_{j-1})$$

In the event of matching bits of $V_{AB}$, each users' confidence in the matching result can be calculated via the formula $\lambda = 1 - \varepsilon$ where error $\varepsilon$ is $(½)^{\lceil N/2 \rceil}$ and $(½)^{\lfloor N/2 \rfloor}$. It is important to note the subtle difference of the floor and ceiling operators in the exponents.

### 3.1.4 Security Analysis

The security properties of TPP with ILW were presented in [1], along with sketches of the proofs that the HN protocol satisfies the required properties. The contents of the following subsections mirror those previously presented security properties, claims, and proof sketches. An expanded analysis concentrating on security against early termination attacks, and security against eavesdropping, was also presented in [113].

#### 3.1.4.1 Security Properties of Fair and Privacy-Enhanced Matchmaking with ILW

Shin and Gligor defined privacy-enhanced matchmaking protocol as one that exhibits the security properties of detector resistance, impersonation resistance, user unlinkability, wish unlinkability, and matching result privacy [6]. But some definitions, or portions

thereof, either conflict with the requirements of TPP, or fail to accomplish key goals. As an example, there is no notion of fairness. Moreover, their definition of detector resistance states that the adversary learns the identity of the honest user if executing the SG protocol with the same wish. In TPP, Trudy could execute the SG protocol with Alice using the wish "Alice loves Bob" and thereby compromise Alice's privacy. Upon failed verification of the execution transcripts, Alice would learn that it was not Bob she had interacted with, yet her privacy would have already been compromised. Due to these shortcomings, the properties of [6] were augmented as required for applicability in the context of TPP and ILW [1].

**Fairness:** Initially referred to as *joint notification of wishes* with *equivalent exchange* by Baldwin and Gramlich [2], fairness in matchmaking captures the idea that one party cannot learn the result of the execution while preventing the other party from knowing the same outcome. A notion of fairness was adopted by [1] that is similar to that of [114] and others. That is, a matchmaking protocol is considered fair if the probability that the participants compute an identical and correct outcome is approximately equal.

**Definition 1:** We say a matchmaking protocol is *fair* if, for any probabilistic polynomial time adversary $A$, and two users $U_1$ and $U_2$ executing the protocol, the probability that $A$ can cause either of the following is negligible:

1. $U_1$ receives answer b while $U_2$ receives ¬b
2. Only one of the users receives a response

**Impersonation Resistance with ILW:** The essence of this property is resistance to impersonation attacks which are of paramount importance with ILW. That is, anyone can guess that "Alice loves Bob", but the ILW would only be valid in this case if believed by

precisely Alice and Bob. An adversary should not be able to impersonate either party and gain an advantage. The definition of impersonation resistance given in [6] was refined to be applicable in the context of ILW. This property requires that an adversary other than one of the linked identities cannot be erroneously authenticated as a linked identity to an honest user.

**Definition 2:** A matchmaking protocol has the property of *impersonation resistance with identity linked wishes* if, for adversary *A*, the probability that *A* wins the following experiment is negligible:

1. *A* selects victim user *V* and a target user *T* from user-space *U* and a wish ***w*** from wish-space *W* that is linked to identities *V* and *T* (*A* will try to impersonate *V* to *T*).
2. *A* has an interaction with *T* who is running the protocol with input ***w***.

If *T* accepts *V* as a matching partner, then *A* wins.

**Wish Unlinkability with ILW:** The property of *wish unlinkability* was put forth by Shin and Gligor as the element that affords forward privacy of wishes and its definition in the context of ILW resembles that of Definition 4 from [6]. Wish unlinkability embodies the notion that, given transcripts of interactions between two sets of users, an adversary should not be able to determine whether the interactions were with the same wish ***w*** or with different wishes ***w*** and ***w'***.

**User Unlinkability with ILW:** The property of *user unlinkability* was put forth by Shin and Gligor as the element that affords forward privacy of identities and its definition in the context of ILW resembles that of Definition 5 from [6]. As described, user unlinkability embodies the idea that, given an execution transcript from user *U*, an adversary cannot feasibly determine that a new transcript involved *U*.

**Matching Result Privacy with ILW:** The property of *matching result privacy* embodies the notion that, given a transcript from honest users that are equally likely to have executed the protocol on the different or identical ILW, an adversary would be unable to determine whether the ILW were identical or different with a probability more than negligibly greater than ½. In essence, an adversary can do no better than guessing.

**Detector Resistance with ILW:** As defined in [6], the property of *detector resistance* is not applicable in the context of TPP and ILW. This is because the definition of the property itself acknowledges that the adversary learns the identity of the honest user after executing the protocol on identical wishes. That is, the very definition of detector resistance, which was modeled with an indistinguishability property, would result in a compromise of the honest user's privacy in the context of TPP with ILW. In that definition, after the adversary successfully completes the protocol with either user $U_0$ or $U_1$, the adversary learns their real identities, and guesses whether the interaction was with $U_0$ or $U_1$. However, in the context of ILW an adversary should not be able to successfully execute the protocol on identical ILW unless the adversary was one of the linked identities. The threat of being discovered cheating may deter some adversaries, but it does not prevent the compromise of privacy.

3.1.4.2 Protocol Security Analysis

**Definition 3:** A protocol is said to be *privacy-enhanced with identity linked wishes* (ILW) if it exhibits the properties of fairness, impersonation resistance with ILW, wish unlinkability with ILW, user unlinkability with ILW, and matching result privacy with ILW.

**Theorem 1:** The Horne-Nair protocol satisfies the requirements of secure, privacy-enhanced matchmaking with identity linked wishes.

**Proof: Fairness** - Assume that an adversary *A* is able to prevent fairness. The analysis focuses on the first fairness property because the second property is trivially satisfied given that all participants in the gradual release process can observe each bit of $V_{AB}$ that is released. If *A* is able to prevent fairness, then *A* must have the ability to achieve a non-negligible disparity in the certainty of a matching result for each participant. Early termination of the gradual release is the only approach to possibly achieving that goal. But the largest advantage that *A* might be able to obtain is a single bit difference. As the number of bits released grows larger, the difference between the users' levels of confidence in the matching result monotonically approaches zero. Hence, for a reasonably sized N, fairness cannot be prevented. Moreover, by varying the desired level of confidence in the matching result, the security and performance of the algorithm may be tuned to achieve the proper tradeoff for particular contexts.

**Impersonation Resistance with ILW -** Assume that an adversary *A* is able to convince target user *T* of a successful match of wishes linked to victim user *V* and target user *T*. To succeed, *T* must confirm the correctness of all bits supplied by *A* during the gradual release process. The knowledge of intermediate values J and P would be required to compute the correct value of $V_{AB}$. Computation of the values for J and P requires the use of private keys of both *T* and *V*. Consequently, *A* would need to use the private keys of *V*, circumvent the cryptographic algorithms being used, correctly guess, or otherwise determine the bits of $V_{AB}$ in order to win. But the first two possibilities contradict the basic security assumptions of the secrecy of private keys and the security of the underlying

cryptographic algorithms. As for the case of correctly guessing or otherwise determining the bits of $V_{AB}$, the probability that $A$ could guess $V_{GUESS}$ which equals $V_{AB}$ to cause a false match is proportional to $(½)^{N/2}$. As N increases, the probability of successfully guessing the correct verifier quickly approaches zero. Aside from guessing the bits, there is also the possibility of a collision with the one-way function. However, the probability of a collision should be negligible with a secure, cryptographic hash function. All possibilities for the adversary to succeed lead to a contradiction. Hence, $A$ cannot impersonate $V$ and convince $T$ of a match of the ILW.

**Wish Unlinkability with ILW -** Assume that an adversary $A$ is able to determine, with probability more than negligibly greater than ½, whether two executions used the same ILW. To do better than guessing, $A$ must be able to learn the linked identities in addition to the generic portion of the wish $\boldsymbol{w}$ such as "attend prom together". Since the linked identities are the focus of the analysis of user unlinkability with ILW, the analysis of wish unlinkability concentrates on the generic portion of the wish. In order to determine the generic wish $\boldsymbol{w}$, $A$ must either be able to circumvent the underlying encryption scheme or $A$ must possess the private key of the intended recipient. However, that would contradict the basic security assumptions of the underlying cryptographic algorithms and of secrecy of the private key. Hence, $A$ cannot win the experiment.

**User Unlinkability with ILW -** Assume that an adversary $A$ is able to determine that a new transcript involves user $U$ with probability more than negligibly greater than ½. To do so, $A$ must be able to recognize similarity between transcripts uniquely indicative of $U$'s participation, or otherwise determine an identity based solely on the transcript. Considering first the case of determining an identity directly based on the transcript, the

design of the HN protocol leverages a form of encapsulated digital signatures of random data with recovery to link identities. Neither identities nor pseudo-identities are included with any messages. It follows then that *A* must be able to circumvent the *key privacy* property of the underlying cryptographic algorithm with probability δ where $\delta > ½$. But possession of such a capability would contradict the fundamental assumptions of key privacy and security of the underlying cryptosystem. Next, with regard to the case of observing commonality amongst the transcripts, since the group identifier represents an arbitrarily large group of users, it would be of little utility. Furthermore, it is only an optional part of the protocol that may be used as a convenience to avoid all-to-all communication problems of some prior protocols. Encrypted random data represent a majority of the contents of messages. The messages would vary significantly between executions, as even a small change in the input random data should result in significant changes to the cipher-text due to the Avalanche effect of cryptographic algorithms. If adversary *A* were able to detect commonality amongst the transcripts that is uniquely indicative of user *U*, it would contradict the foundational assumption of security of the underlying cryptographic algorithms. Consequently, given a transcript from an execution involving user *U*, adversary *A* cannot determine that a new transcript also involves *U*.

**Matching Result Privacy with ILW -** Assume that an adversary *A* is able to determine whether a given transcript represents an execution using either the same wishes, or different wishes, with a probability more than negligibly exceeding ½. Hence, *A* must be able to verify the correctness of $V_{AB}$, which itself requires knowledge of the values J and P, or knowledge of each user's random data $R_A$ and $R_B$. Since the values are computed locally and not directly included in the message, the only way for *A* to accomplish the

task would be if *A* possessed the ability to circumvent the underlying cryptographic algorithms with probability δ where δ > ½. But that contradicts the foundational assumption of security of the underlying cryptographic algorithms. Consequently, adversary *A* is able to do no better than guessing whether a given transcript represents an interaction involving either the same or different wishes. □

3.1.4.3 Security Against Early Termination Attacks

An early termination attack occurs when one party attempts to gain a sizeable advantage by terminating the HN protocol execution prior to completion of the gradual release process. Given confidence of an honest user $\lambda_H$, confidence of an adversary denoted $\lambda_A$, and $\Delta\lambda = (\lambda_A - \lambda_H)$, we say that the adversary has a sizeable advantage if $\Delta\lambda > 0.10$ and $\lambda_A > 0.90$, or $\Delta\lambda > 0.20$ and $\lambda_A > 0.80$. That is, the adversary has a sizeable advantage if the adversary achieves a reasonably high level of confidence in the matching result as well as a significantly greater level of confidence compared to the honest user. Theorem 2 and its proof formalize the notion of a sizeable advantage with a specific upper bound. This expanded proof of security was originally presented in [113].

**Theorem 2:** In the HN protocol, given number of bits released prior to termination N, the advantage that adversary *A* can gain over honest user *H* has an upper bound of $\Delta\lambda = 0.0625$ advantage with confidence $\lambda_A = 0.9375$. All other cases result in either unacceptably low confidence or a more negligible advantage.

**Proof:** The correct-guess probability *Pr(G)* for bit $b_i \in \{0,1\}$ is 0.5. Assuming matching bits, since each participant contributes half of the bits, confidence values for an *N* bit release may be computed as

$$\lambda_A = (0.5)^{\lceil N/2 \rceil} \text{ and}$$

$$\lambda_H = (0.5)^{\lfloor N/2 \rfloor},$$

with the notable difference being the floor versus ceiling operators in the exponent. When *N* is even, confidence values are equal, and the execution is fair. Hence, the focus of the proof is early termination with an odd number of bits. Assume the worst case that the honest user initiates the process releasing the first bit, yielding a potential advantage for the adversary.

If *N=1*, confidence values would be $\lambda_H = 0.0$ and $\lambda_A = 0.5$. But $\lambda_A = 0.5$ is no better than simply guessing, hence there is no advantage.

If *N=3*, then $\lambda_H = 0.5$, $\lambda_A = 0.75$, and $\Delta\lambda = 0.25$. There is a one in four chance of a false match. That is, $0.75 < 0.80$ and there is no sizeable advantage.

If *N=5*, then $\lambda_H = 0.75$, $\lambda_A = 0.875$, and $\Delta\lambda = 0.125$. Here, $\Delta\lambda > 0.10$ by a small margin but $\lambda_A < 0.90$ and there is no sizeable advantage.

If N=7, then $\lambda_H = 0.875$ and $\lambda_A = 0.9375$. There is a disparity of only $\Delta\lambda = 0.0625$ and no sizeable advantage. Let this be the basis of *N=k*.

$\forall N_i \mid i > 7 \wedge (i \bmod 2) = 1$. It needs to be shown that:

$$N = k+2 \rightarrow \lambda_H (N=k+2) > \lambda_H (N=k) \wedge$$

$$\Delta\lambda (N=k+2) < \Delta\lambda (N=k)$$

It is known that:

$$N_{k+2} > N_k \;\rightarrow\; (0.5)^{N_{k+2}/2} < (0.5)^{N_k/2}$$

$$\rightarrow \left[(0.5)^{\lceil N_{k+2}/2 \rceil} - (0.5)^{\lfloor N_{k+2}/2 \rfloor}\right] < \left[(0.5)^{\lceil N_k/2 \rceil} - (0.5)^{\lfloor N_k/2 \rfloor}\right]$$

Therefore, by mathematical induction we conclude that an adversary cannot gain a sizeable advantage with early termination attacks on the HN protocol. □

3.1.4.4 Security Against Eavesdropping

Without loss of generality, consider an embodiment of the HN protocol that is based on the RSA algorithm. This logic would also generalize to embodiments of the HN protocol based on other asymmetric cryptographic algorithms with comparable security properties and valid underlying assumptions. Assume that the cryptographic hash function *H(x)* is a secure and collision resistant one-way function. Lastly, assume that random data are generated via cryptographically secure means. If the message exchange leveraged Transport Layer Security (TLS) 1.3+, then security could be argued based on the strength of AES and inability to obtain any of the message contents. However, let us instead assume that the messages can be obtained by the eavesdropper (e.g., via compromise of matchmaker or lack of encrypted communication channels) and show that the protocol is still secure in the event of successful message transcript recovery. This expanded proof of security was also originally presented in [113].

**Theorem 3:** In the HN protocol, an attacker who successfully intercepts all of the messages for an execution instance cannot determine the linked identities ($ID_A$ or $ID_B$), the correct verification value ($V_{AB}$), or the identity linked wishes.

**Proof:** A proof of security against eavesdropping can be based primarily on the hardness of breaking the RSA cryptosystem via separate evaluation of each of the messages in an execution transcript.

**1. Challenge X -** The challenge *X* is computed as $E(PU_B, w + E(PR_A, R_A))$. Hence, to

obtain $\boldsymbol{w}$ or the encrypted random data, the adversary would require access to private key data $PR_B$ or alternatively to break the RSA cryptosystem. But that would violate either the assumption of confidentiality of $PR_B$, or the assumption of the security of the RSA algorithm. Additionally, there is no identity or pseudo-identity information included in the message for recovery.

**2. Counter-Challenge Y -** The counter-challenge $Y$ is computed as $E(PU_A, w + E(PR_B, R_B))$. To obtain $\boldsymbol{w}$ or the encrypted random data would require access to private key data $PR_A$ or alternatively to break the RSA cryptosystem. But that would violate either the assumption of confidentiality of $PR_A$, or the assumption of the security of the RSA algorithm. Additionally, there is no identity or pseudo-identity information included in the message for recovery.

**3. Verifier $V_{AB}$ -** The verification value is independently computed by Alice as $H(H(R_A)+H(D(PU_B,D(PR_AY))))$, and by Bob as $H(H(D(PU_A,F))+H(R_B))$. Hence, computation of the correct verification value requires the use of public and private keys from both of the linked identities. Otherwise, to compromise the protocol one would need to break the RSA cryptosystem. Here again, that would violate either the assumption of confidentiality of $PR_A$ and $PR_B$, or the assumption of the security of the RSA algorithm. Additionally, there is no identity or pseudo-identity information included in the verifier $V_{AB}$ for recovery. Due to those facts, combined with the properties of the secure one-way hash function, there is no way to relate the bits of $V_{AB}$ to $X$, $Y$, or their components in such a way as to determine the identities or the identity linked wishes.

Therefore, an eavesdropper cannot determine the linked identities, correct verification value, or the identity linked wishes. □

## 3.2 Quantifying Fairness

Baldwin and Gramlich equated fairness in matchmaking with joint notification of wishes and equivalent exchange [2]. The following complementary perspective on fairness in matchmaking was provided in [1]. A matchmaking protocol is fair if the probability that each of the participants compute the same, correct result is approximately equal. Otherwise stated, it is fair only if none of the involved parties can achieve a sizeable advantage. More formally, given users $U_1$ and $U_2$, the probability that a probabilistic polynomial time adversary can cause one of the following is negligible:

1. $U_1$ receives answer b while $U_2$ receives $\neg b$

2. Only one of the users receives a response

All users see each bit that is released in the HN protocol. Therefore, the second property is trivially satisfied, and attempts to quantify fairness of the HN protocol focused on the first requirement that all users receive the same answer (i.e., equivalent exchange). Substantial efforts have contributed to quantifying fairness in resource allocation problems leading up to attempts at developing a unified theory [115] or framework [116] for incorporating the variety of proposed formulas. In the context of TPP, a fairness index was proposed that incorporates concepts from Jain's index [117] with established definitions of fairness in matchmaking and details of the HN protocol [3]. A key contribution of Jain's work was the notion of reducing fairness to the selection of a suitable allocation metric and the proper approach to quantifying equality.

There are two broad approaches an adversary might use in attempts to avoid joint notification with equivalent exchange, thusly resulting in disagreement as to the outcome of the matchmaking process in violation of the first required property for fairness. The goal

of the adversary in those cases would be to cause either a false negative or a false positive. First, if the adversary impersonated $U_2$ and engaged in the protocol with $U_1$, early termination might be employed in attempts to achieve a non-negligible advantage with regard to confidence in the matching result yielding a potential false negative. On the other hand, the adversary may instead attempt to introduce a false positive by guessing the correct bits of $V_{AB}$ during the gradual release process.

Upon the first incorrect bit during the gradual release process, confidence drops to zero. In such a case, random decoy bits would be used to complete the process, and an adversary would not be able to determine whether bits were guessed correctly or not. Assuming that correct bits are released, recall that the confidence of each user can be calculated as $\lambda = 1 - \varepsilon$ where error $\varepsilon$ is $(½)^{\lfloor N/2 \rfloor}$ and $(½)^{\lceil N/2 \rceil}$ for $U_1$ and $U_2$ respectively if $U_1$ initiated the gradual release. At first, it may be unclear as to whether formulas for fairness in resource allocation might be applicable to matchmaking problems such as TPP. However, [3] proposed that matchmaking could be considered to be a special case of a resource allocation problem with a single shared resource and that each user's confidence in the matching result is the appropriate allocation metric. The following formula for a fairness index can then be derived by applying the concepts first proposed by Jain in [117] resulting in a measure to quantify fairness that is a function of *N*, the number of correct bits released.

$$Fairness\ Index\ f(x) = f(N) = \frac{[1 - {}^{1}/_{2}{}^{\lfloor N/2 \rfloor}]^2}{[1 - {}^{1}/_{2}{}^{\lceil N/2 \rceil}]^2}$$

This function for the fairness index has the desirable properties of continuity and boundness. That is, the theoretical lower bound is 0.0 representing no fairness whatsoev-

er, and the upper bound is 1.0 representing perfect fairness. Yet the range of potential values is continuous on the spectrum from lower to upper bounds. Intuitively, executions of the HN protocol exhibit perfect fairness with $f(x)$=1.0 when an even number of bits are released given that the users have equivalent confidence in the matching result. On the other hand, executions that terminate with an odd number of bits released result in differing confidence values hence some level of unfairness.

Considering first the case of a single bit release, the initiator would have no confidence in the matching result yet the other user might have 50% confidence in a matching result resulting in $f(x)$=0.0. As argued in [3], this case is not considered to be unfair in a practical sense given that 50% confidence is no better than a coin toss and no real advantage would be accomplished. Thus, the practical lower bound on fairness index values is the case of a three bit release which would result in a fairness index value of $f(x)$=0.4444. While the tendency might be to interpret this worst case as achieving a non-negligible advantage in violation of the requirements for fairness, this actually highlights an advantage of this approach. Although the difference in confidence values is maximized in this worst case with a 25% disparity, the confidence of the adversary is still only 75%, a confidence level upon which it would be unwise to act in virtually any practical scenario. In the event of early termination attacks, the circumstances in which the advantage of an adversary is greatest correspond with the lowest confidence values thereby negating the perceived advantage. As depicted in Figure 3.5, user confidence and fairness index values monotonically approach the highest confidence and perfect fairness values as the number of bits released increases. Even though the length of verification value $V_{AB}$ might seem large for such a gradual release process, with 512 bits for instance

for SHA3-512 as the hash function, high degrees of confidence and fairness can be achieved with matching release of much smaller numbers of bits. As an example, consider that only 28 total bits released can result in over 99.99% confidence and a fairness index value of $f(x)$=1.0.

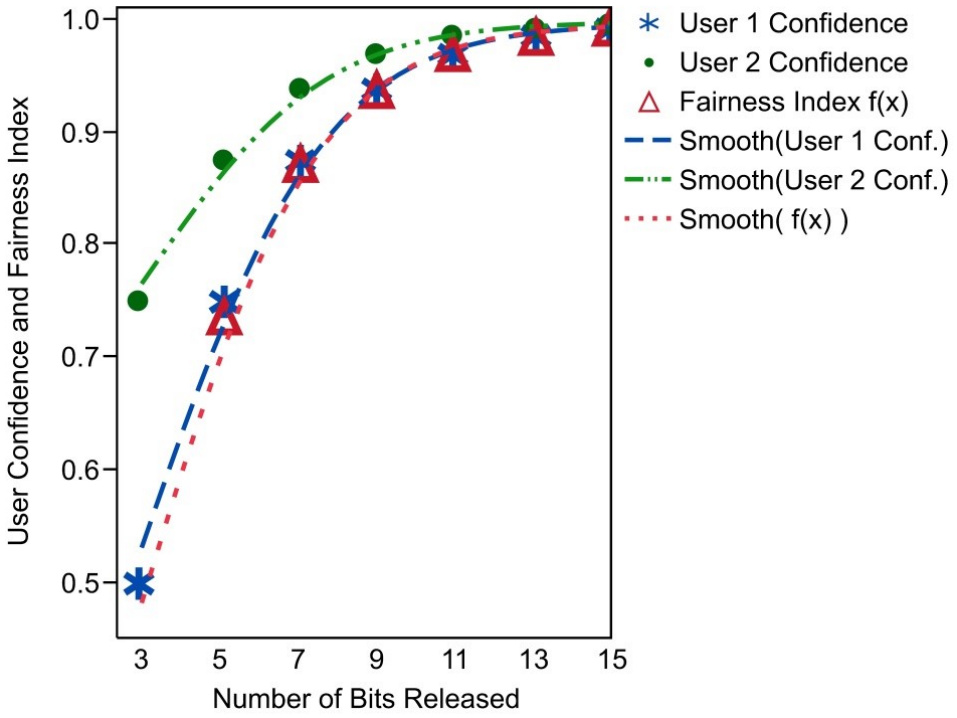



Figure 3.5. User confidence and fairness index versus number of bits released [4]

# CHAPTER 4

# EXPERIMENTAL METHODOLOGY AND RESULTS

A proof-of-concept implementation was developed initially to validate the correctness of the HN protocol. Subsequently, prior to incurring the full costs of design and development of the production system, a desire to manage risks led to an assessment of the practicality of the approach, as well as experimental testing to gain a better understanding of the tradeoffs that would be involved. Hence, a feasibility evaluation was conducted including the quantification of fairness as discussed previously, along with two phases of performance testing isolating the computational costs and communication overhead of the protocol. To that end, the proof-of-concept implementation, experimental methodology, and results are now described.

## 4.1 Proof-of-Concept Implementation

The proof-of-concept implementation was initially developed to verify the correctness of the protocol. For language selection, important factors included familiarity, performance, and support for desired target platforms. The work in [118] compared energy efficiency and performance of 27 software programming languages for specific tasks. Among other things, the results suggested that that C/C++ tended to exhibit the fastest runtimes and appear in the Pareto optimal sets for a combination of energy, time, and memory objectives. Java slightly edged out C# further down the list, but all four lan-

guages appear in approximately the top half of Pareto optimal sets for the energy, time, and memory combination. Although C/C++ may be best from an energy consumption and performance standpoint, Java and C# are likely better suited for sharing significant portions of common code for a variety of platforms that includes mobile devices relative to C/C++. An informal test program was written in Java to compare relative performance with C# for basic cryptographic operations that would be required for the HN protocol, such as creation of multiple asymmetric key pairs, and asymmetric encryption and decryption operations. The results showed that, for the environment and operations at hand during proof-of-concept development, C# exhibited a significant performance advantage for the specific operations being performed. Due to time constraints, no further analysis was undertaken at the time to determine the root causes of the observed performance disparity and the source code for the proof-of-concept application was written in C#.

The standard libraries in the System.Security.Cryptography namespace did not allow for encryption with one's private key. A search led to a comparison of potential alternatives and the ArpanTECH.RSAx libraries were selected. The ArpanTECH.RSAx libraries are available under the terms of the Code Project Open License (CPOL) [119] at The Code Project web site [120]. The CPOL is not recognized by the Free Software Foundation[1] (FSF) as a free software license due to prohibitions in 5.d on selling, leasing or renting the work by itself or a portion thereof, and 5.f prohibiting use for "illegal, immoral or improper purposes" [121]. However, despite those violations of components of the FSF requirements, it is actually a license that can be advantageous for commercial development. Some fundamental aspects of the license that foster commercial use include ex-

[1] www.fsf.org

plicit recognition that code and executables can be used in commercial applications, modified to create derivative works, and redistributed. The CPOL also includes a perpetual, worldwide, royalty-free patent license. The implementation used a 4,096 bit modulus for the "large" key pairs, a 2,048 bit modulus for the "small" key pairs, and SHA3-512 for the hashing algorithm.

The proof-of-concept test application and the computational performance focused implementation that is instrumented with performance timers, both execute the entire protocol locally to avoid dealing with communication overhead that would have added complexity and required precautions to ensure that results were not unintentionally impacted. However, the variant used for testing communication overhead, and the deployed application, both require a mechanism for communication between the clients and the server. Consequently, Google Protocol Buffers were selected for client-server communication, primarily due to performance, utility, and compatibility with multiple programming languages including C++, C#, Go, Java, and Python [122]. Protocol Buffers provide a mechanism for serialization of structured data. In practice, the software engineer defines messages in a *proto* definition file and compiles the messages with the *protoc* compiler yielding classes that encapsulate the data and provide serialization and deserialization functionality. Protocol Buffers are available under the BSD license, a permissive academic license that does not have the copy-left requirements of reciprocal licenses.

Upon startup, the proof-of-concept application creates cryptographic key pairs for the players if they do not already exist and initializes a simple matchmaker. Figure 4.1 contains a screenshot of the application after startup. The text area provides status at key times during execution. It is assumed that Alice always initiates the protocol and that she

always chooses Bob. The tester can act as Bob and choose whether Bob chooses Alice, or Eve when confronted with the step of linking ideas to generic wishes. If Bob chooses Alice, then the protocol execution proceeds accordingly and ultimately confirms the shared ILW as in Figure 4.2. On the other hand, in the event that Bob chooses Eve (Eve here could actually be any user other than Alice), the dialog would instead indicate that the gradual release process failed and that the verification values were not equal. For testing and debugging purposes, whether the process results in a match or non-match, the contents of the verification values are included with the notification message in a human readable form. Refer to Appendix A pertaining to the source code of the proof-of-concept and performance testing applications.

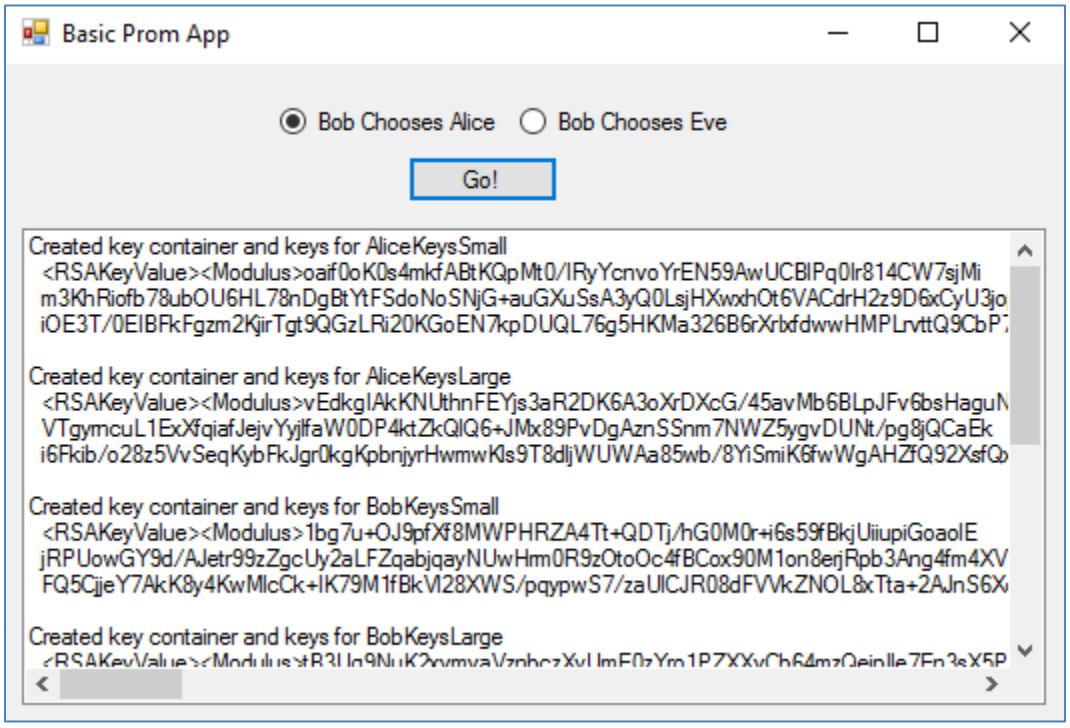



Figure 4.1. Proof-of-concept application upon startup

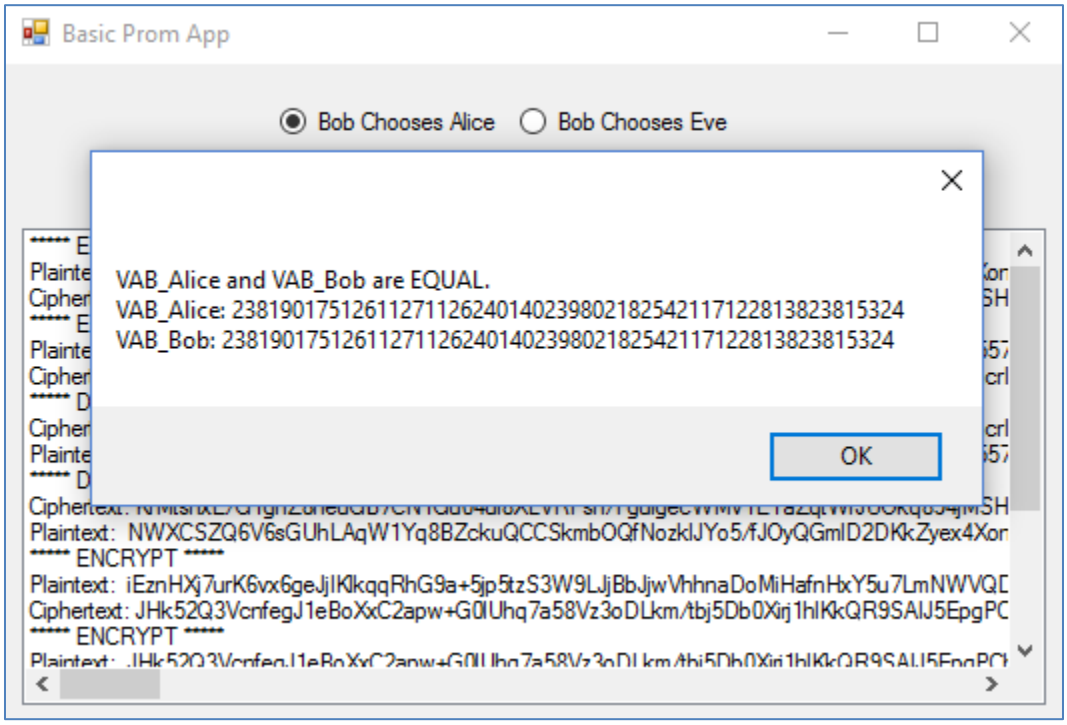



Figure 4.2. Proof-of-concept application test with successful matching result

### 4.1.1 Need for Two Key Lengths

Upon practical analysis, or attempts to implement the HN protocol, one quickly arrives at the realization that two pair of public keys with different sizes are required. Notice that the protocol incorporates encapsulation of encrypted results. While the algorithm agnostic pseudocode presentation of the algorithm abstracts away the complexity of key pairs with different lengths, the RSA-based sample embodiment makes the requirement clear. As an example for consideration, in the context of an RSA-based implementation, if the outer layer utilized a 4,096 bit modulus yielding 512 byte ciphertexts, then the "smaller" key pair might use a 2,048 bit modulus yielding 256 byte ciphertexts to support encapsulation as used in the protocol. Indeed, the proof-of-concept implementa-

tion used those key lengths. Another example might be using 3,072 and 6,144 bit key pairs. In addition to improved security, increased key sizes afford the opportunity to allocate more bytes to the (encoded) secret, the random data, or both. That comes with the cost of added computational runtime. But based upon the results of feasibility testing, the extra computations should be well within the range of practical runtimes.

### 4.1.2 Specific Security Concerns

Software engineers should employ established best practices during development to produce secure software. Just of few of the many excellent references on design and development of secure software systems include [123], [124], and [125]. Beyond those fundamentals, a number of security-related pitfalls specific to implementation of the HN protocol are now discussed.

#### 4.1.2.1 Key Privacy

When selecting encryption algorithms, one important consideration in the context of our protocol is that of *key privacy* as described by [102]. This concept can also be considered *recipient anonymity* in that, given a ciphertext, an eavesdropping adversary cannot feasibly determine which key (from a set of known public keys) was used to create it. Bellare, Boldyreva, Desai, and Pointcheval showed that some encryption schemes, or variants thereof, did or did not exhibit key privacy under chosen-plaintext attacks. Thus, the key privacy property of encryption algorithms should be considered during algorithm selection. On the other hand, one potential mitigating factor in the HN protocol is the

reality that a large majority of the plaintext inputs consist of random data, thereby complicating cryptanalysts' efforts.

#### 4.1.2.2 Padding Vulnerability

During the phase of the protocol in which Bob received challenge $X$, Bob decrypted the challenge yielding the generic wish $\boldsymbol{w}$ and encrypted random data $F$. Bob then chose user $U'$ with whom he may share the secret wish $\boldsymbol{w}$. In cryptographic algorithms employing structured padding schemes, it is at this point that the padding scheme may be a weakness. For example, consider the use of RSA with Optimal Asymmetric Encryption Padding (OAEP). If Bob chose a user other than Alice and attempted the decryption operation, most implementations would abort fairly rapidly with an error message upon trouble processing the pad bytes. This introduces a vulnerability to brute force attacks iterating through the user space to attempting to unmask the sender. Fortunately, in the HN protocol all messages are the desired input length so that no padding is needed. In fact, the vast majority of the bytes are cryptographically random data adding a significant amount of entropy. For this reason, variants of the encryption and decryption process were implemented in the RSAx libraries to avoid the padding vulnerability by omitting the operations that lead to vulnerability when applied to the HN protocol. Since use of these variants could actually introduce weaknesses if used in other contexts, the source code documentation was updated with additional warnings.

Table 4.1. Complexity analysis of the HN protocol [3]

| Type of Operation | Alice | Bob | Matchmaker | Total |
|---|---|---|---|---|
| # of Asymmetric Encryptions | 2 | 2 | 0 | 4 |
| # of Asymmetric Decryptions | 2 | 2 | 0 | 4 |
| # of String Concatenations | 2 | 2 | 0 | 4 |
| # of One-Way Hashes | 3 | 3 | 0 | 6 |
| # of Messages | $c + \frac{\log(1-\lambda)}{\log 0.5}$ | $c + \frac{\log(1-\lambda)}{\log 0.5}$ | $2 \times \left(c + \frac{\log(1-\lambda)}{\log 0.5}\right)$ | $4 \times \left(c + \frac{\log(1-\lambda)}{\log 0.5}\right)$ |

## 4.2 Performance Testing for Feasibility Analysis

The HN protocol makes use of multiple asymmetric encryption and decryption operations and the gradual release process results in a number of rounds of communications with the total number of messages dependent upon the desired levels of confidence and fairness. The results of a complexity analysis of the HN protocol in Table 4.1 summarize the performance of the protocol [3]. In the analysis, λ again represents the desired level of confidence in the matching result and *c* denotes a small constant such as one if push notifications were used or two to three if a polling mechanism were employed. This motivated an experimental performance evaluation to determine the feasibility of the HN protocol prior to incurring the costs of development for a complete matchmaking system. The evaluation involved two phases of experimentation focusing on computational costs and communication overhead respectively. The primary questions that the experimentation sought to answer included:

1) Is computational performance of the HN protocol practical?

2) Is communication overhead of the HN protocol practical?

3) Is a high degree of fairness practical?

4) Is a high degree of anonymity practical?

### 4.2.1 Computational Overhead

The first phase of feasibility testing addressed the first of the primary research questions attempting to assess the practicality of the protocol. The computational overhead measurements concentrated on the most computationally intensive steps in the protocol that dominate overall runtime of the CPU bound portions of the overall process. The experimental methodology and results of computational testing are now presented in turn.

#### 4.2.1.1 Methodology

The matchmaker in the HN protocol is effectively a public database and the most computationally intensive tasks all occur on the client side. Hence, the evaluation concentrated on the runtime of client-side computations. Throughput of the system was not scrutinized because the number of responses will likely be constrained to a small constant to mitigate risks of brute force attacks. Enforcing the limit could be accomplished via technical controls, by associating a cost with each attempt, a combination of controls and cost, or another approach. In the HN protocol, the challenge and counter-challenge steps, and to a lesser extent the computation of the verifier $V_{AB}$, are the most significant contributions to computational costs. The test application executed 100 iterations of the HN protocol using random users from a pool of users and random values with each iteration. The source code was instrumented with high precision performance counters with <= 1

μs resolution to measure the runtime of the code segments implementing the computations involved with the challenge, counter-challenge, and verification values. The variety of systems upon which the test application was executed incorporated a representative sampling of processors that included legacy and recent architectures, budget to higher-end processors, and both desktop and mobile variants.

Table 4.2. Computational performance results [3]

| **CPU** | **Average Runtime** | | |
|---|---|---|---|
| | ***Challenge (ms)*** | ***Counter-Challenge (ms)*** | ***Compute Verifier (ms)*** |
| Intel® Atom™ Z3740D, 1.13 GHz | 188.123 | 191.255 | 0.0364 |
| Intel® Pentium® N350, 2.16 GHz | 114.779 | 109.071 | 0.0287 |
| AMD Athlon™ 64 X2 4200+, 2.2 GHz | 105.765 | 101.977 | 0.0215 |
| AMD Athlon™ M320, 2.1 GHz | 100.161 | 99.906 | 0.0145 |
| Intel® Core™ i5 5200U, 2.2 GHz | 57.591 | 54.978 | 0.0092 |
| Intel® Core™ i7 4770, 3.4 GHz | 44.158 | 43.963 | 0.0074 |

#### 4.2.1.2 Results

Table 4.2 presents a summary of the initial computational performance results. The mean runtimes for execution of the challenge and counter-challenge phases ranged from around 191 ms for the Intel® Atom™ Z3740D mobile processor at an operating frequency of 1.33 GHz to around 44 ms on the Intel® Core™ i7 CPU at 3.4 GHz. As expected,

the mean runtimes for the computation of verifier $V_{AB}$ were orders of magnitude smaller and they are thusly not significant contributors to overall runtimes. Expanded performance results presented in [113] also included the Snapdragon™ 835 mobile processor, which performed a bit worse than the Atom™ at around 250 ms to 280 ms. Even so, the runtimes for the most expensive operations were demonstrated to be practical overall, even on legacy, budget, and mobile processors. The results confirm the computational feasibility of the protocol in response to the first research question.

### 4.2.2 Communication Overhead

The second phase of feasibility testing addressed the last three research questions aimed at assessing the practicality of the communication overhead involved, and of achieving high degrees of fairness and anonymity. Given that the main protocol is a small constant number of rounds, the communication overhead measurements concentrated on the gradual release process, which dominates the overall runtime of the I/O bound portions of the process. The experimental methodology and results of communications testing are now presented in turn.

#### 4.2.2.1 Methodology

When evaluating communication overhead of the HN protocol, the primary concern is that of runtime rather than throughput. Although achieving high throughput and scalability with databases and client-server applications can pose significant challenges, the subject is well-studied. Common approaches to throughput and scalability challenges would be applicable in the present context such as the use of clusters, redundancy, geographic

distribution, and eventual consistency (e.g., see [126], [127], and [128]). The foundational communication-related question pertaining to the HN protocol is whether the runtime performance of the gradual release process might be perceived as reasonable to users of the system. The utilization of anonymous communication channels further amplifies the overhead of the gradual release process. In order to assess feasibility, a test program instrumented the source code for the gradual release process with high resolution performance timers to measure the runtime of the process. Ten iterations were executed for each test case of releasing N bits where $N \in \{8,10,12\ldots58\}$. That range represents test cases where users' confidence in the matching result ranges from 0.9375 to 0.999999998. The spectrum of varying numbers of bits released and confidence values also covered a broad range of fairness index values. For reference, Table B.1 of Appendix B presents the numbers of bits released {3…44} along with corresponding user confidence and fairness index values.

Additionally, all tests were executed employing VPN services, onion routing, and combinations thereof to test the feasibility of achieving varying levels of anonymity. As configured, the tests used the Tor [104] onion routing overlay network and VyprVPN [129] with OpenVPN using 256 bit AES and SHA-256 for the virtual private network. To avoid potential introduction of biases that could result from writing custom source code for comparison of the varying anonymity approaches, an Anonabox Pro [130] was used between the client and network, configured appropriately to cover the variety of test cases. Figure 4.3 presents the geography of the test configuration. The client test application executed in the Dallas, TX, USA area while the server test application ran on Amazon Web Services (AWS) [131] at data centers in Boardman and Portland, OR, USA.

Initially, the tests were executed with direct communication between clients and the server for a baseline without any anonymity layers. Subsequently, the same tests were executed with use of Tor, VPN services, and combinations of the two using VPN servers in Austin, TX, USA, Seattle, WA, USA, and Paris, France. Figures 4.4 and 4.5 show screenshots of the executing server and client test applications respectively. The operation of anonymity approaches were validated at a high level by evaluating the IP addresses of clients from the matchmaker's perspective and confirming that the clients appeared to originate from locations that deviated from the true origin of the clients' communications.

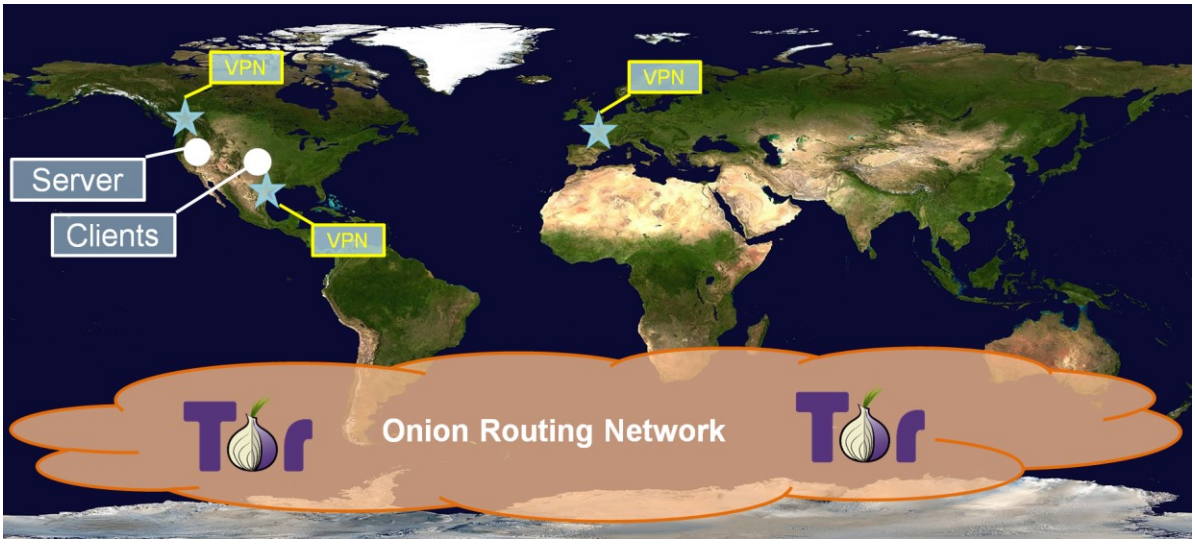



Figure 4.3. Geography of test configuration [2]

[2] Background Earth Image Source: NASA Visible Earth

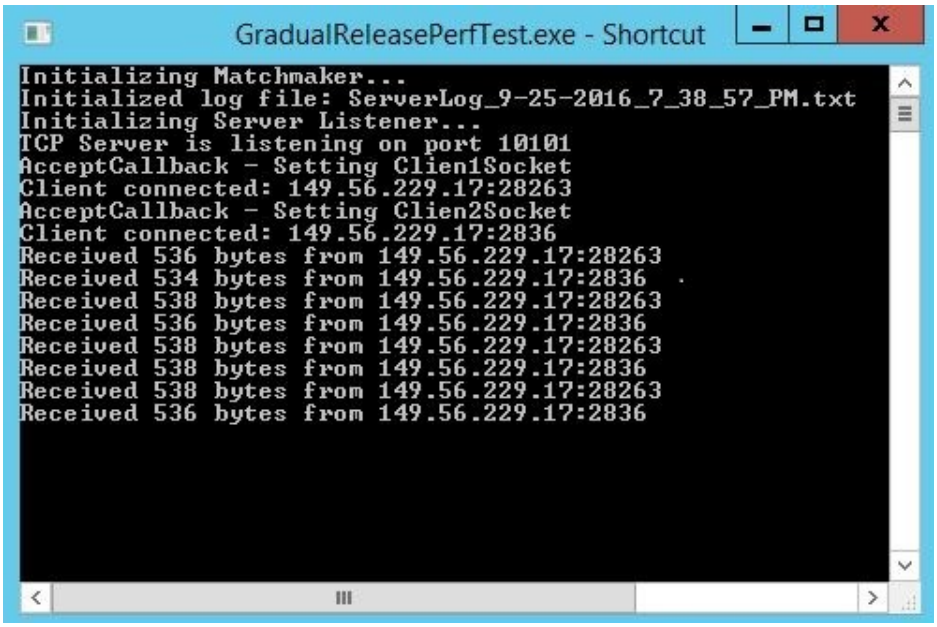



Figure 4.4. Server application used for testing communication overhead

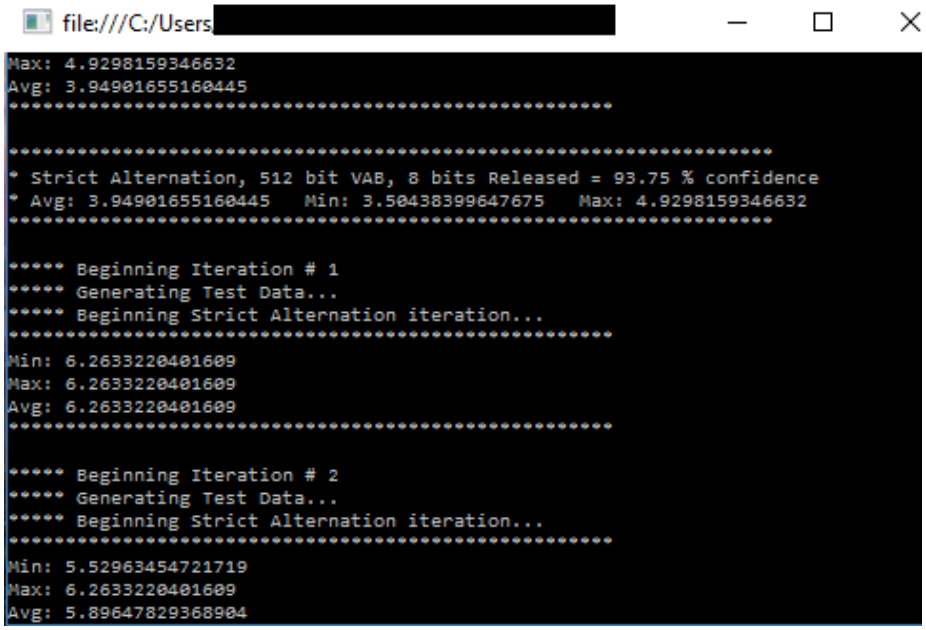



Figure 4.5. Client application used for testing communication overhead

4.2.2.2 Results

The data of Figure 4.6 represent the communication performance results measured as mean runtime in seconds for the gradual release of even numbers of bits from 8 to 58, corresponding with confidence values from 93.75% to 99.9999998%. The communication costs in the form of average runtimes associated with different anonymity approaches and a range of confidence values are presented graphically as data points. The graph also includes the ellipsoid density composed of confidence curves and density contours that are derived from the bivariate normal distribution of the data. The data set appears in Table C.1 of Appendix C which presents the number of bits released {8…58}, confidence values, and mean runtimes for each test case. Figure 4.7 presents a different perspective on the experimental results by plotting average runtimes against the confidence in the matching result when using each anonymity approach. The three remaining research questions are now discussed in turn with conclusions that can be drawn from the experimental results.

*A high degree of confidence is practical.* As evidenced by the contents of Table B.1, the degree of confidence in the matching result and the fairness index are functions of the number of (correct) bits that have been successfully released. When an even number of bits have been released, the fairness index value is 1.0 conveying that no user has an advantage over the other. In cases of early termination with an odd number of bits released, the fairness index values monotonically approach 1.0. The highest confidence test case was a 58 bit release corresponding with 99.9999998% confidence in the matching result. If one desired to maximize confidence by releasing all bits of $V_{AB}$, an extrapolation of the results of Table C.1 reveals that the process might take one minute or more depending on

factors such as the performance of communication links, geographic distances, and the anonymity approach being used. But the experimental data show that a high degree of confidence can be achieved with fewer bits. For example, by releasing 44 bits, one can achieve 99.99998% confidence with average runtime $t$ of $5 < t < 25$ seconds. This highlights a key advantage of this approach to ensuring fairness. The process can be fine-tuned to achieve the appropriate tradeoff between the desired level of confidence in the matching result and runtime performance allowing for customization to suit applications in disparate problem domains.

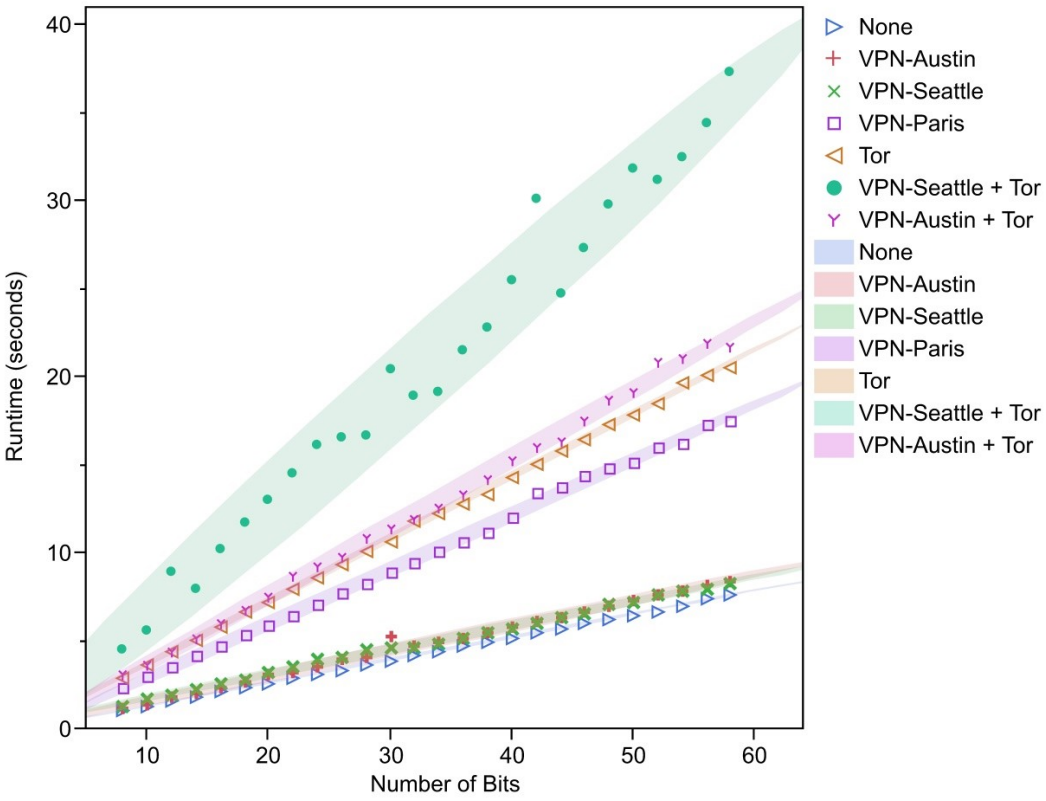



Figure 4.6. Runtime versus number of bits released with ellipsoid density [3]

*A high degree of fairness is practical.* As discussed previously, the degree of fairness is a function of the number of correct bits gradually released. For even numbers of bits released, the fairness index value is 1.0 representing identical levels of user confidence in the matching result. While theoretically a high degree of fairness can thusly be achieved with a very small number of bits, the confidence in the matching result would be relatively low. Another way of understanding the question of fairness is to consider the degree of unfairness that a misbehaving adversary might be able to achieve. The practical lower bound on the fairness index is 0.4444 corresponding with the case where the adversary has confidence that there is a ¾ chance of a matching result while Alice's confidence is no better than randomly guessing. The specific interpretation of this case corresponding with the highest degree of unfairness that the adversary can achieve would require context for detailed evaluation. However, in general, it would rarely be a wise idea to act on a suspicion when there is a ¼ chance that the claims are false and no real proof exists.

With increased numbers of bits released, the difference between the users' confidence values becomes more negligible as the fairness index value approaches 1.0. The experimentation was conducted using real-world networks rather than controlled lab configurations suggesting that high degrees of confidence and fairness are practical, even when employing different approaches to achieve anonymity. As an example, 99.9939% confidence, corresponding with a fairness index of $0.9998779 < f(x) < 0.9999389$, was demonstrated with runtime $3 < t < 17$ seconds depending on the anonymity approach. The experimental data confirm that, in addition to high confidence, a high degree of fairness is also practical with the HN protocol under real-world conditions.

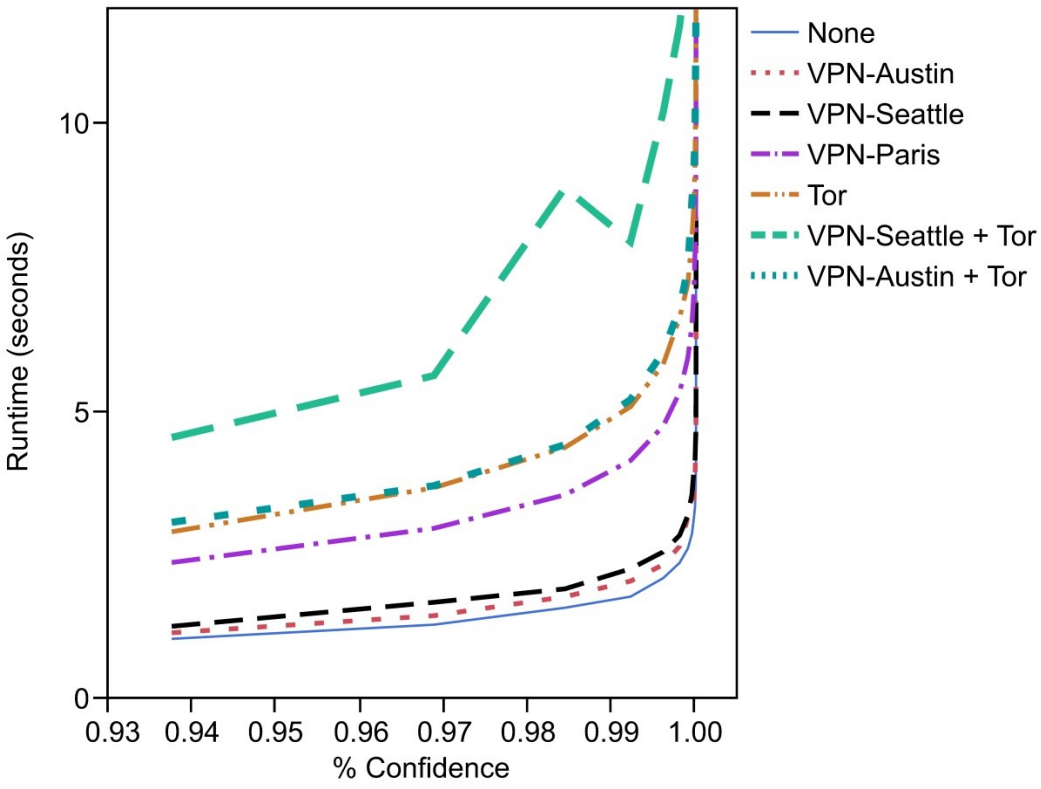



Figure 4.7. Runtime versus percent confidence per anonymity approach [3]

*A high degree of anonymity is practical.* Recall that the first round of testing was without any extra attempts to achieve anonymity to serve as a baseline. The use of Tor, VPNs, and combinations of the two were then tested to quantify the relative costs of those common approaches to anonymity. The results of Figure 4.6 and Table C.1 convey predominantly linear performance with respect to the number of bits released. A more significant deviation from the linear relationship was observed with the usage of Tor and certain combinations of Tor usage with VPN service. It was hypothesized that the observed variation could be the result of quality of service concerns with Tor such as those reported in [132]. However, further experimentation would be required to attempt to iso-

late the cause(s). But even with the variation observed, runtimes remained within the realm of feasibility for the expected applications of the HN protocol. When focusing on the relative performance of the approaches, the runtime with VPN service varied by geographic location of the VPN servers, but observed runtimes ranged from approximately 1.1 to 2 times that of the baseline. Similarly, observed runtimes with Tor usage ranged from approximately 2.5 to 3 times that of the baseline. Relative runtimes using combinations of Tor with VPN service ranged 4 and 6 times that of the baseline. In summary, the experimental results showed that runtimes of the HN protocol were practical with a variety of approaches to achieve anonymity. Furthermore, in the typical use case, a user would not wait in real-time for a response, but the matchmaking process is expected to occur asynchronously over minutes, hours, or even days. In such cases, the gradual release process would occur in the background on user devices and would be imperceptible to users in most cases. In essence, from a user's perspective, there might often be an answer waiting the next time they launch the application.

### 4.2.3 Limitations

The use of a disparate sampling of systems for computational performance testing and real-world networks rather than controlled simulations for communications performance testing had both advantages and disadvantages. If a comparison of the performance of the processors themselves had been the emphasis, a controlled hardware configuration with different CPUs being the only variation would have been preferable. However, the goal was to confirm the feasibility of the protocol on a wide spectrum of consumer devices. The set of target devices was intentionally not limited to high-end or mid-range

hardware but also included legacy, budget, and mobile devices. When testing communication performance, controlled systems and networks would have afforded a better opportunity to do root cause analysis upon observations like the appearance of significantly greater variance between test cases when using a combination of VPN service and the Tor network. But there is also the possibility that such anomalies might not be observed with a much more controlled study. In such a case, the controlled study would likely not have reflected the real performance that users of the system might typically experience. In that respect, the use of real-world networks was considered a more accurate test to determine practicality of the approach. In summary, although there may be some disadvantages to the test approach relative to more tightly controlled systems and networks, it was the most well-suited to accomplish the overall goal of this study, which was to test the feasibility of the solution to TPP with ILW under real-world circumstances and thereby limit risk prior to more extensive development and subsequent deployment. It was decided that the best way to understand the performance that the user base might experience, assess feasibility of the protocol, and better understand the tradeoffs from users' perspectives was to test with the same devices and networks that the deployed application would utilize.

### 4.2.4 Summary of Experimental Results

Recall that the key research questions being evaluated to assess feasibility of the HN protocol were as follows:

1) Is computational performance of the HN protocol practical?
2) Is communication overhead of the HN protocol practical?

3) Is a high degree of fairness practical?

4) Is a high degree of anonymity practical?

Following a characterization of fairness in the HN protocol, and its relation to users' confidence, test applications were developed and executed to evaluate computational costs and communication overhead using a variety of test cases. In each case, the experimental data supported affirmative answers to the research questions. The computational performance results demonstrated the operations that involve asymmetric cryptographic algorithms that dominate computational runtimes overall to be practical, even for budget and mobile processors with more limited resources. The communication overhead was also evaluated for a wide range of numbers of bits released corresponding with a wide range of confidence and fairness values. The communication overhead was also evaluated using a range of approaches to anonymity including VPNs, Tor onion routing, and combinations thereof.

The results, which were initially presented in [3], showed that high degrees of confidence, fairness, and anonymity are all practical in the context of real-world networks and systems in which such a matchmaking system might be deployed. Moreover, the relative impacts of varying confidence, fairness, and anonymity were quantified, which should support and inform decisions during the design and evolution of the production system for privacy-enhanced and fair matchmaking with ILW.

# CHAPTER 5

# PRIVACY-ENHANCED AND FAIR MATCHMAKING SYSTEM

After an initial comprehensive literature review, and concurrently with definition of TPP and ILW, the protocol under development took a number of forms during its evolution. Multiple directions were explored building upon a variety of related veins of prior research including zero knowledge proofs, secure function evaluation, and a scheme leveraging an ephemeral key based dual-signature scheme. Continual evaluation of the spectrum of security and privacy requirements in the context of critical use cases led to the discovery of unacceptable vulnerabilities in each iteration until protocol evolution culminated in the HN protocol in the presently presented form. A proof-of-concept implementation validated the approach [1], a provisional patent application initiated protection of the intellectual property in 2016 [101], an approach for quantifying the fairness of executions of the HN protocol was proposed and performance testing demonstrated the feasibility of the protocol [3]. Beyond demonstrating practicality of computational costs and communication overhead, the results quantified relative impacts of varying levels of confidence, fairness, and anonymity to assist with identification of appropriate tradeoffs as the HN protocol is applied to a variety of problems with requirements resembling those of TPP with ILW. Additionally, United States patent 10,432,400 was granted in 2019 confirming exclusivity over the claimed invention [101].

Following an analysis of TPP within a taxonomy of PETs, and highlights of multiple applications of a solution to TPP beyond dating and relationships, attention turned to design and development of a more full-featured production system [4]. Although the server component could be generic to support a variety of scenarios requiring fair and privacy-enhanced matchmaking with ILW, the initial client application focuses on TPP itself as dating and relationships are perhaps the most universal of concepts.

## 5.1 System Architecture

An overview of the architecture for the proposed privacy-enhanced and fair matchmaking system with support for ILW, as described in [4], is depicted in Figure 3.1. The software solution is naturally a suitable candidate for a client-server architecture. The clients could be web browser based, native applications, or even implemented in firmware. As such, a client app may run on a wide range of devices from desktops or laptops to mobile devices, or even dedicated matchmaking hardware. As presented in the diagram, a plurality of connectivity options exist including connection via Local Area Network (LAN), direct connection to the Internet, and connections that may occur via anonymity networks. Meanwhile, the Matchmaker represents the server (or servers) in the client-server model. As proposed, it is effectively a public database but with an Application Programming Interface (API). Although the API is not strictly required, it is included in the design as a convenience to better facilitate protocol execution, and to assist with detecting and preventing certain misuse cases.

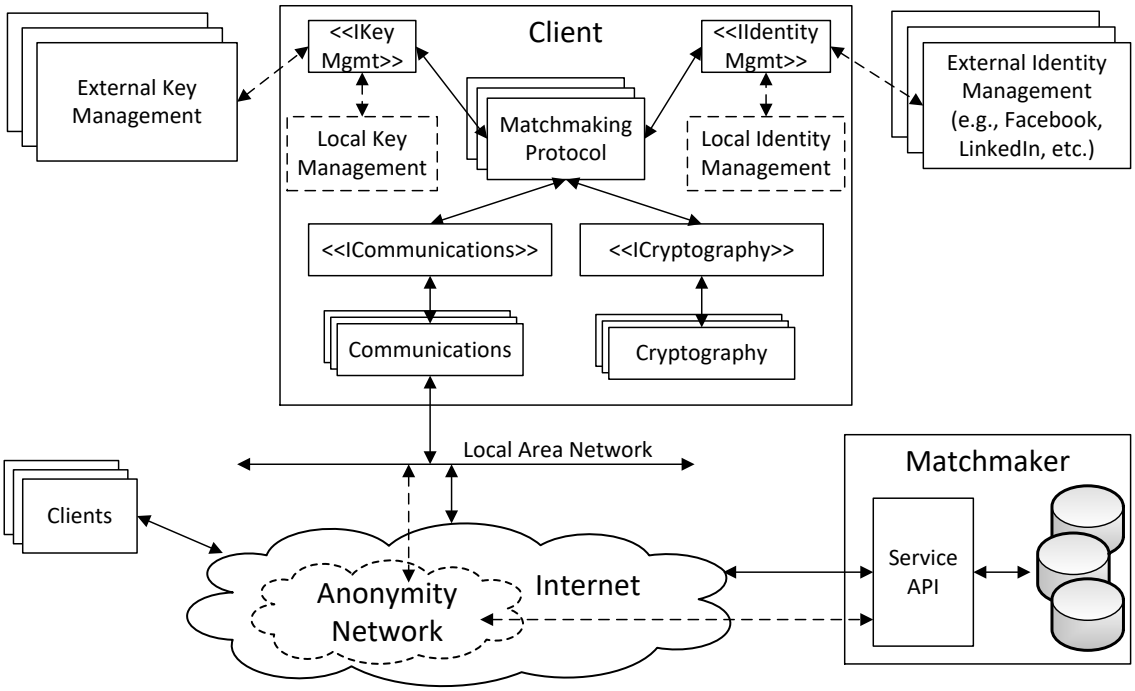



Figure 5.1. Architecture of proposed privacy-enhanced and fair matchmaking system [4]

In the diagram of Figure 5.1, internal aspects of the software detailed design were provided for a sample client application. Note again that in the HN protocol neither identities, nor pseudo-identities, are included in any messages or stored in the Matchmaker's database. The avoidance of aggregated, privacy relevant data significantly reduces potential consequences of data breaches. Consequently, the third party involved with the matchmaking system can be effectively a public database, which is somewhat comparable to broadcast communications. All private information is processed on the client-side, giving the user and the client application the primary role in ensuring confidentiality and privacy preservation. Two critical components of the system are those for identity management and cryptographic key management. Those entities can be internal or external to

the client application, which is the reason for dashed lines in the diagram. In varying matchmaking contexts, other components may also be required. Examples might include an entity for group management, or for centralized prom event management in the case of the SecretMatch™ Prom app. Design decisions related to potential group and event management components are further discussed in Section 6 as examples of applying "privacy by design" [133].

The software architecture uses an interface-oriented design to improve changeability and allow for identity and key management that is either internal or external to the application. In many cases, users already place some level of trust in external entities for identity management and those may be good choices for specific applications. For instance, all Facebook® users may be *potential* users of the prom application and other dating related matchmaking systems, while all LinkedIn® users may be *potential* users of the matchmaking system for fair and privacy-enhanced executive recruiting. There are also other identity management systems in which users already place some level of trust, such as that of their employer. Likewise, client applications might initially generate cryptographic key pairs but leverage key management external to the application. For example, in such a case every Facebook® user might have associated encryption keys (e.g., either self-generated or system generated decoys). In other cases, it may be more desirable to leverage identity or key management that is entirely internal to the application. Such an implementation might utilize a web-of-trust model akin to the one popularized with PGP encryption [112], or with a particular mobile device and identifying details of individuals in its associated contact list.

The design of the proposed system also leverages the concept of interfaces for communication and cryptographic libraries to foster agile adaptability as the client applications are developed for varying application domains requiring fair and privacy-enhanced matchmaking with ILW. As an example, although the proof-of-concept application utilized Google Protocol Buffers [134] for efficient communication, large production systems with publicly available APIs may benefit more from other standards such as Representational State Transfer (REST) APIs [135] using JavaScript Object Notation (JSON) [136] [137]. Similarly, an implementation using an encryption approach with a security model based on computational hardness may need to be replaced with a different algorithm as the security industry and cryptanalysis continue to evolve. The interface-oriented design of the client applications fosters adaptability, which is critical given the variety of potential applications of the technology.

In summary, there is at least one inevitability common to both software engineering and security engineering – that change is virtually guaranteed. After the initial version of the system, both planned and unforeseen enhancements will be necessary. For instance, the business climate might force the need to implement an unexpected monetization strategy into a mobile app. Alternatively, it may be desirable for performance reasons to replace a relational DBMS with a distributed solution like Hadoop or an in-memory database as very large amounts of non-volatile RAM become commonplace. Or the cryptographic algorithms and hash algorithms in the initial implementation may become insecure due to new attacks or technological advancements (e.g., imagine that an original implementation of some security application had hard-coded the Data Encryption Standard algorithm [138] [139] or that quantum computers became commoditized shortly after

release [140]). With the inevitability of change in mind, the planned, full-featured implementation employs an interface-oriented design with APIs for fundamental concepts such as the library of cryptography and hash algorithms, communications facilities, and separate user and group management systems. This interface oriented design allows the implementation to vary, while retaining a consistent interface for improved changeability. With the rapid pace of change in the software and security disciplines, designing with the future in mind is a necessity for maintainability, extensibility, and sustainable development moving forward.

## 5.2 Initial Project Planning

The initial project planning phase involved important decisions that would have long-term consequences such as programming languages and development stacks. The factors considered and the formal decision making process aimed at ensuring adequate engineering justification supporting the decisions made are now discussed. The focus of this phase was to utilize sound engineering decision analysis processes to manage risks related to certain foundational choices. The processes for the client and server applications are further detailed in turn.

### 5.2.1 Client Application

The initial client application targets TPP because the concept of dating and relationships is almost universal and it also represents a multi-billion dollar industry. The client application uses an interface oriented design as mentioned previously. While section 5.4 discusses key detailed design decisions, many of which pertain to the system as a whole,

a few detailed design decisions such as client-side programming language, development stack, and cryptographic libraries are specific to the client application. To ensure that decisions with far-reaching impacts were supported by sound engineering justification, weighted decision matrices akin to the Pugh Matrix [141] were used to facilitate decision making. Processes that utilize formal decision support mechanisms like this often do not strictly require selection of the highest scoring option(s), but the scores should be taken into consideration. Ideally, the results provide support for the final choice. For the client app, the choice of which programming language and associated development stack to leverage for the client application was viewed as a critical design decision with major, long-term consequences. Thus, the decision matrix presented in Table 5.1 was produced to facilitate decision making amongst the top candidate options. An initial review of options narrowed the field of considerations to Native Mobile Applications (Java, Swift, & C#), React Native (JavaScript), Xamarin (C#), and Flutter (Dart). Each programming stack has its strengths and weaknesses. In the table, each criterion was weighted and scored for each candidate language/stack with factors such as a common codebase and support for required technical features being two of the most important considerations. Tables 5.2 and 5.3 provide weighting and scoring legends for interpreting the results.

Table 5.1 Client-Side language / development stack decision matrix

| **Criteria** | **Weight** | ***Native Apps (Java, Swift, & C#)*** | ***React Native (JavaScript)*** | ***Xamarin (C#)*** | ***Flutter (Dart)*** |
|---|---|---|---|---|---|
| Common Codebase | 4 | 1 | 3 | 4 | 3 |
| General Support Libraries | 3 | 4 | 3 | 4 | 3 |
| Required Crypto Support | 4 | 3 | 3 | 3 | 3 |

| | | | | | |
|---|---|---|---|---|---|
| Maturity of Framework / Libraries | 3 | 4 | 4 | 4 | 2 |
| Likelihood of Longevity | 3 | 4 | 3 | 3 | 3 |
| Familiarity / Ease of Adoption | 4 | 2 | 3 | 4 | 1 |
| Performance | 3 | 3 | 2 | 3 | 2 |
| REST API / JSON support | 4 | 4 | 4 | 4 | 4 |
| Google Protocol Buffers support | 2 | 1 | 0 | 4 | 4 |
| Potential Re-use from Proof-of-concept | 2 | 0 | 0 | 2 | 0 |
| Development Tool Support | 4 | 4 | 4 | 4 | 4 |
| Manageable Complexity | 4 | 2 | 4 | 4 | 4 |
| Industry/Community Momentum (TIOBE) | 3 | 2 | 2 | 3 | 2 |
| Industry/Community Popularity (TIOBE) | 3 | 4 | 4 | 4 | 2 |
| **RAW SCORE:** | | 37 | 39 | 50 | 37 |
| **WEIGHTED SCORE:** | | 139 | 122 | 271 | 230 |

Table 5.2. Decision matrix weighting criteria

| Weight | Significance |
|---|---|
| 1 | Minimally Significant |
| 2 | Somewhat Significant |
| 3 | Significant |
| 4 | Very Significant or Critical |

Table 5.3. Decision matrix scoring legend

| Score | Satisfaction / Limitations | Likelihood |
|---|---|---|
| 0 | None, Not Applicable | 0-10% |
| 1 | Minimal, Major Limitations | 10-25% |
| 2 | Somewhat, Minor Limitations | 26-74% |
| 3 | Significant, Very Few Limitations | 75-89% |
| 4 | Complete Satisfaction, No Limitations | 90-100% |

It is important to note that the results captured engineering judgement at the time the decision was made. This represents a snapshot in time, and many of the scores might change if the process were repeated at other points in time. For instance, Flutter was in Beta testing at the time of the study, hence its lower relative score for Maturity of Framework/Libraries. If it exited Beta testing and attained broad adoption, then the score would improve. At the time of writing, Flutter still supported only apps for Android and iOS. Hence, with regard to cross-platform support, Xamarin/C# still had the advantage in that category with support for more platforms.

Another detail worth noting was the inclusion of Google Protocol Buffer (GPB) Support as a criterion. At the time of the study, the decision of REST/JSON versus GPB had not yet been finalized. The scoring for performance was based on the results of Pereira et al. [118] and derived by dividing the Pareto optimal set for time and memory performance into tiers. Those performance results were general in nature, and not specific to the proposed development stacks, but they were considered to be useful relative scoring metrics as a factor in the formal decision making process.

The industry/community momentum and popularity scoring again represented snapshots in time that were derived from TIOBE index data [142]. The purpose for including momentum and popularity as decision factors was to include objective measures that ideally correlate with other factors that might be perceived as more subjective scoring such as likelihood of longevity or likelihood of long-term support for quality development and test tools.

Ultimately, the platform/language combination of Xamarin and C# was the highest scoring option. With several good options available, each having its advantages and dis-

advantages, the formal decision making activity confirmed the perception of a slight advantage for the Xamarin/C# option. Consequently, it was selected during the initial planning and design of the client application. This afforded the opportunity to target Android and iOS in addition to traditional Windows-based environments with a majority common codebase.

Upon completion of the initial Android app, customizing the small percentage of code for iOS and Windows, along with thorough testing on those platforms, will be significantly more cost effective than implementing separate native applications for each platform. Appendix D contains a listing of steps necessary to establish the development environment for the SecretMatch™ Prom client application including a listing of tools, components, or other dependencies and the software licenses where applicable.

### 5.2.2 Server Application

While initially for use with TPP specific client application, the server-side application represents the Matchmaker in the HN protocol and it was designed in a generic way to support future applications in other problem domains. The programming language selection can have significant implications with regard to long-term maintainability, extensibility, scalability, tool support, and other factors. Rather than letting programming language decision be a biased selection based purely on preferences, a weighted decision matrix akin to the Pugh Matrix [141] was again used to ensure that the choice was supported by sound engineering reasoning. Table 5.4 contains the fully populated matrix and Tables 5.5 and 5.6 giving the scoring legends.

Table 5.4. Server-side programming language decision matrix

| **Criteria** | **Weight** | ***Node.js / JavaScript*** | ***Python*** | ***Java*** | ***C#*** | ***Go*** |
|---|---|---|---|---|---|---|
| Linux Support | 4 | 4 | 4 | 4 | 4 | 4 |
| Windows Support | 3 | 4 | 4 | 4 | 4 | 4 |
| MacOS Support | 2 | 4 | 4 | 4 | 4 | 4 |
| RDBMS Support (e.g., MySQL) | 4 | 4 | 4 | 4 | 4 | 4 |
| NoSQL DB Support (e.g., Mongo DB) | 3 | 4 | 4 | 4 | 4 | 4 |
| Maturity of Framework / Libraries | 3 | 4 | 4 | 4 | 4 | 3 |
| Likelihood of Longevity | 3 | 4 | 4 | 4 | 3 | 3 |
| Familiarity / Ease of Adoption | 4 | 3 | 3 | 3 | 4 | 2 |
| Performance (Pareto Opt/ Time+Mem) | 4 | 2 | 2 | 3 | 3 | 4 |
| REST API / JSON support | 4 | 4 | 4 | 4 | 4 | 4 |
| Google Protocol Buffers support | 3 | 3 | 4 | 4 | 4 | 4 |
| Potential Re-use from Proof-of-concept | 2 | 0 | 0 | 0 | 2 | 0 |
| Development Tool Support | 4 | 4 | 4 | 4 | 3 | 3 |
| Manageable Complexity | 4 | 3 | 4 | 2 | 2 | 4 |
| Industry/Community Momentum (TIOBE) | 4 | 4 | 4 | 2 | 3 | 2 |
| Industry/Community Popularity (TIOBE) | 4 | 4 | 4 | 4 | 4 | 2 |
| **RAW SCORE:** | | 55 | 57 | 54 | 56 | 51 |
| **WEIGHTED SCORE:** | | 193 | 200 | 188 | 193 | 178 |

Table 5.5. Decision matrix weighting criteria

| **Weight** | **Significance** |
|---|---|
| 1 | Minimally Significant |
| 2 | Somewhat Significant |
| 3 | Significant |
| 4 | Very Significant or Critical |

Table 5.6. Decision matrix scoring legend

| Score | Satisfaction / Limitations | Likelihood |
|---|---|---|
| 0 | None, Not Applicable | 0-10% |
| 1 | Minimal, Major Limitations | 10-25% |
| 2 | Somewhat, Minor Limitations | 26-74% |
| 3 | Significant, Very Few Limitations | 75-89% |
| 4 | Complete Satisfaction, No Limitations | 90-100% |

While there is some specificity to the designer or moderate subjectivity to criteria such as likelihood of longevity or manageable complexity, a majority of the criteria were scored using objective measures. The most important factors, those corresponding with the highest weightings, included operating system support, support for the leading candidate databases, performance in terms of time and memory usage, and support for the leading candidate communication protocols. The scoring for *performance* was based on the results of Pereira et al. [118] and derived by dividing the Pareto optimal set for time and memory performance into tiers. The industry/community momentum and popularity scoring was again derived from the TIOBE index [142].

After careful analysis of the language choices, there were a number of options that might be a good fit for the server-side development. In the end, the field of options was reduced to a choice between Python and JavaScript using Node.js. Both options have relatively mature libraries for server-side REST API development and they each have interfaces for the variety of database storage options that might be selected. Basic REST APIs were developed with JavaScript/Node.js and with Python to see if any strengths or weaknesses became apparent that might cause an application developer to lean towards one language option over the other.

In the end, it was determined that, while the performance comparison suggested that Node.js might ultimately edge out Python in terms of time and memory performance, either language should be able to scale adequately for the matchmaker. The database interface and associated queries would most likely be the bottleneck with respect to the matchmaker's performance. The trial development comparison seemed to confirm the idea that the complexity of the solution would be more manageable with a Python based solution relative to JavaScript and Node.js. Given the potential for "technical debt" resulting from increased complexity, and the impact that experience has shown such complexity can have on long-term extensibility and maintainability of the system, Python was chosen for the server-side development. Appendix E contains a listing of steps necessary to establish the development environment for the SecretMatch™ server-side Matchmaker application.

## 5.3 Enhancement Analysis and Prioritization

The project proposal included prioritization of three potential enhancements to the system and selection of the highest priority for further analysis and incorporation into the initial design and implementation. The enhancements were assessed with regard to theoretical and practical considerations to determine the highest priority to be incorporated with the initial production system. The remaining two will be planned for a future release after the system has matured. The following sub-sections describe the proposed enhancements along with an explanation of the outcome of analysis and option chosen for incorporation into the initial system design.

### 5.3.1 Prevention of Certain Brute Force Attacks

Certain brute force attacks on the system and protocol are expected. Hence, potential prevention of such attempts should be considered during detailed design. In particular, the proposed attacks for analysis and possible prevention included excessive provision of challenge values to over-populate the database or cover large portions of the user space (e.g., to "find anyone willing to attend the prom with me"), excessive attempts to respond to a challenge via counter-challenges, and brute force attacks on the gradual release process itself.

### 5.3.2 Integration of Temporal Constraints

The notion of temporal constraints would add value to the overall system by effectively assigning an expiration date to inquiries about ILW. For example, the ILW "Alice + Bob + attend prom together" might be a privacy threat after the prom event is over (e.g., imagine that Bob actually attended with Eve) and such an inquiry should thusly be considered invalid after a certain point in time. However, the concept could be more complicated than it initially sounds. Would the server be responsible for removing "expired" wishes? If so, that places some level of trust in the untrusted third party. Is there a mechanism whereby the ILW could "self-destruct" or be otherwise invalidated to avoid a compromise of privacy after a certain period of time? This would be an important feature, but there are also trade-offs to consider. It is also worth noting that an extremely small time-to-live constraint would prove useful in real-time or near real-time contexts such as privacy-enhanced matchmaking in a singles bar or party-like setting.

### 5.3.3 Integration of Geographic Constraints

The notion of geographic constraints would add value to the overall system by assigning a geographic area within which some element of the protocol is constrained. This could take multiple forms. There may be user groups associated with particular geographic regions or instances of the protocol execution itself could be constrained to users that are verified to be within the geographic region during execution of the protocol. Questions of responsibly assigning responsibilities and where to place trust again come into play when considering incorporation of geographic constraints.

### 5.3.4 Prioritization of System Enhancements

The need for temporal constraints and prevention of brute force attacks both represent security and privacy risks to be mitigated. While geographic constraints could also potentially mitigate certain risks, the probability and impact of the others imply that they should be prioritized over geographic constraints when following customary risk management procedures. With regard to temporal constraints, a side benefit of expiration for wishes is that it can contribute to keeping the size of the matchmaker's database manageable. Having a default maximum time-to-live for wishes can also help mitigate some of the damage that could be caused by brute force attacks that result in excessive database entries. Upon scoring with a traditional risk management probability/impact matrix, both temporal constraints and prevention of brute force attacks were categorized as high risks that would need to be mitigated at some point. The prioritized list when taking into account cost-benefit analysis focused on costs of implementing the mitigation or corrective action appears as follows (from highest priority to lowest priority):

1. Temporal Constraints
2. Prevention of Certain Brute Force Attacks
3. Geographic Constraints

Given this prioritization, along with the risk analysis results identifying numbers 1 and 2 as both having high risk scores, the ultimate decision was to start with a basic implementation supporting temporal constraints as well as designing in a foundation for taking the next step in preventing certain brute force attacks in the form of server-side API key restrictions.

## 5.4 Key Detailed Design Decisions

A number of critical detailed design decisions were required for the full matchmaking system. Those decisions can impact many aspects of the system such as performance, security, maintainability, interoperability, extensibility, and more. Key decisions highlighted in this section relate to the type of database to use for the matchmaker, anonymity approach, how to present adjustable fairness and confidence levels to the user (if at all), human factors, identity management, key management, group management, communication protocols and API design, cryptographic algorithms and libraries, and more.

### 5.4.1 Type of Database

Choosing the right database is a foundational decision with long-term consequences for server-side data storage in systems like SecretMatch™. While relational databases dominated the landscape for decades, the rapid adoption of NoSQL ("Not only SQL") databases for use cases prioritizing scalability, high data volume (i.e., "big data"), and

variety or complexity of data in recent years has complicated the decision making process with more tradeoffs to consider. NoSQL databases have been classified by their varying data models into four groups [143]:

1. Key-Value Stores
2. Document Stores
3. Column Oriented Stores
4. Graph Databases

Key-value stores are comparable to dictionary or map data structures. Most key-value stores are relatively unstructured and schema-less, with in-memory data storage often affording high performance for specific use cases. Examples of key-value stores include memcached [144] and Redis [145]. Meanwhile, as the name suggests, document stores are characterized by a data model oriented around documents, often in a common format like JSON. That affords more complex structuring of data than key-value stores (e.g., via nesting), but usually without the restrictions of a schema. Examples of document stores include MongoDB [146] and CouchDB [147]. Column oriented stores, also called column family stores, exemplify a data model that is column-centric. Column oriented databases are commonly used to afford scalability with extremely large datasets. Cassandra [148] and Bigtable [149] are well-known column oriented databases, the latter of which has a data model that Chang et al. described as a "sparse, distributed, persistent multidimensional sorted map" [150]. In Bigtable, the primary indices are row and column keys as well as timestamps with the indexed values being raw bytes. Finally, the data model of graph databases centers around graph-based structures and they are often well-suited

for data sets with a plurality of relationships. Examples of graph-based databases include GraphDB [151] and RedisGraph [152] [153].

While non-relational storage options often referred to as NoSQL databases vary significantly and exceptions abound, many are generally associated with the notion of eventual consistency [127] and BASE properties (*Basically Available, Soft state,* and *Eventually consistent*) [154] in contrast to the traditional ACID properties (*Atomicity, Consistency, Isolation,* and *Durability*) of relational databases. Pritchett asserted that "BASE is diametrically opposed to ACID" [154]. An alternative, less dramatic perspective would be to consider NoSQL databases exhibiting BASE properties as exploiting a tradeoff to achieve greater scalability by relaxing the notion of consistency. Indeed, NoSQL marketers and industry adopters often use performance and scalability as primary justification for selection over relational competitors. Some studies comparing performance of relational and non-relational databases such as [155] have supported the assertion of improved performance and scalability with NoSQL databases, while other researchers have presented mixed results [156] or logical discussions questioning common arguments in favor of NoSQL performance superiority [157]. In support of the notion that relaxing consistency requirements can result in performance improvements, Klein et al. found that achieving strong consistency as opposed to eventual consistency resulted in a 10% - 25% reduction in throughput [158]. Moreover, that study also found significant performance differences even when comparing only NoSQL databases and suggested that NoSQL database performance is significantly influenced by the extent to which the query capabilities and data model fit the use cases of the application.

The idea that alignment of an application's use cases with the data model and query capabilities of a database directly correlates with performance likely extends beyond NoSQL databases to include relational storage options as well. When evaluating the core concepts for fair and privacy-enhanced matchmaking with the HN protocol, the conceptual model maps readily to the relational framework. Figure 5.2 presents a simplified ER diagram for the matchmaker database. Note that a challenge X can have multiple match attempts, characterized by different counterchallenge values Y. Normalization analysis resulted in the concept of X-Y association linking a challenge with one or more match attempts. The group identifier (GID) was also considered for factoring out to a separate conceptual entity mapping user identifiers to group identifiers to reduce redundancy. However, the analysis of memory space savings did not suggest that this normalization step would be value-added. Additionally, the privacy-focused approach taken for group management, as discussed in Section 5.4.7, also suggested less benefit from that step.

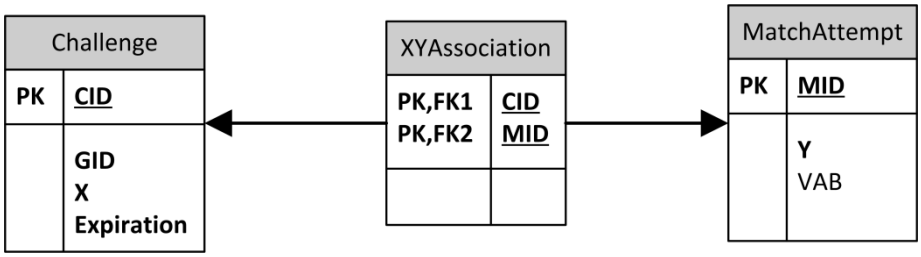



Figure 5.2. Simplified ER diagram for matchmaker database

When choosing whether to leverage a traditional relational database versus the variety of NoSQL options, the decision came down to three main criteria that favored the relational approach. First, the data model of TPP using the HN protocol maps well to rela-

tional databases as the entities and their relationships are well defined by the protocol. Secondly, the strict consistency notion of the ACID model was preferable over the eventual consistency of the BASE approach in the context of fair and privacy-enhanced matchmaking, particularly with real-time or near real-time executions of the HN protocol such as live matchmaking at a party.

```
-- SQL for TPP Database (MySQL/MariaDB)
-- v0.11

-- Create the database schema
CREATE DATABASE MATCHDB;
USE MATCHDB;

-- Create the database tables
CREATE TABLE CHALLENGE (
    CID INT unsigned NOT NULL AUTO_INCREMENT PRIMARY KEY,
    GID INT unsigned NOT NULL,
    X VARBINARY(1024) NOT NULL,
    Expiration DATE NOT NULL);

CREATE TABLE MATCHATTEMPT (
    MID INT unsigned NOT NULL AUTO_INCREMENT PRIMARY KEY,
    Y VARBINARY(1024) NOT NULL,
    VAB1to64 BIT(64),
    VAB65to128 BIT(64),
    VIndex TINYINT UNSIGNED DEFAULT 0 );

CREATE TABLE XYASSOC (
    CID INT unsigned NOT NULL,
    MID INT unsigned NOT NULL,
    FOREIGN KEY(CID) REFERENCES CHALLENGE(CID),
    FOREIGN KEY(MID) REFERENCES MATCHATTEMPT(MID) );

-- Grant a user DBA level access - insert real username and passwrod first
CREATE USER IF NOT EXISTS 'username'@'localhost' IDENTIFIED BY 'insert-password-here';
GRANT all on matchdb.* to 'username'@'localhost';

```

Figure 5.3. Sample SQL to create a matchmaker database

Finally, the capabilities of SQL map well to the query requirements for the HN protocol with the added benefit of portability across RDBMS solutions from different vendors whereas portability of query language is less common across NoSQL options. In summary, the data model, query capabilities, and consistency properties led to the decision to use a relational database for the initial fair and privacy enhanced matchmaking system. Figure 5.3 gives sample SQL DDL that could be used to create a basic matchmaker database for a system using the patented HN protocol.

### 5.4.2 Anonymity Approach

An important detailed design decision for the proposed system was whether to incorporate an enhanced anonymity approach, and if so, which one. Among other things, the list of factors to consider includes:

1. Performance impact
2. Complexity
3. Privacy and security requirements of the specific context
4. Compatibility with existing software, systems, and networks
5. Potential impact on usability

The experimentation initially reported in [3], and further described in Chapter 4, demonstrated that use of enhanced anonymity approaches like VPNs, Tor onion routing, and even combinations of VPN with Tor exhibited acceptable performance for real world systems and networks. Regarding the second factor, addition of VPN, Tor, or both to the initial matchmaking system would clearly add complexity. But the complexity would be warranted if benefits outweighed the potential drawbacks. Hence, the decision reduced to

the other three factors of privacy in context, compatibility, and potential impact on usability.

The HN protocol affords some level of privacy out-of-box in that neither identities nor pseudo-identities are included in any messages or stored in the matchmaker's database. Privacy relevant data stay in possession of the user. Consequently, for many users, the enhanced privacy techniques may not be required in the context of matchmaking for a school dance. As a result, the decision was further reduced to considerations of compatibility and usability. Some networks are configured to block VPN and/or anonymity network traffic, which would negatively impact usability. Moreover, some users may regularly utilize VPN software, which itself could introduce compatibility and usability challenges if SecretMatch™ also used its own VPN or anonymity network implementation as well. Based on a study with 109,780 participants in 2018, it was estimated that 18% of Internet users in North America and Europe had used a VPN or proxy server within the last month, with that number increasing to 30% in the Asia Pacific region [159]. Furthermore, the size of the VPN market worldwide was expected to increase from $15.64 billion USD in 2016 to $35.73 billion USD in 2022, more than doubling in market size over a period of just six years [160].

Ultimately, an evaluation of the cost-benefit tradeoffs of incorporating enhanced anonymity into the initial SecretMatch™ system resulted in the decision to avoid incorporation of one specific implementation. With a large and increasing number of users already leveraging VPNs and other privacy enhancing technologies, the risk of decreased usability due to user interface complexity and potential compatibility issues with software and networks in use outweighed potential benefits of layering additional technologies onto

the initial production system. That decision should likely be re-evaluated after broad adoption of the initial system. User perceptions should always be taken into consideration. In Ken Thompson's Turing Award Lecture from 1984, he famously described a modified compiler to inject a Trojan horse into login authentication routines [161]. He further described how the compiler could propagate the Trojan injection mechanism upon recompilation from unmodified source code. In the end, users of modern technology must place trust somewhere. At present, some users have already placed a degree of trust with various VPN providers they are currently using. The initial goal with the system is to avoid decisions that could unnecessarily limit use of those technologies with SecretMatch™, while working to earn user trust over time such that acceptance of a built-in VPN or anonymity network will ultimately be accepted as affording equal or greater enhancement to user privacy. That will also give time to more thoroughly study potential compatibility issues between adoption of different solutions to maximize usability when built-in enhanced anonymity features may be pushed to production as part of the SecretMatch™ technology roadmap.

### 5.4.3 Human Factors and the User Interface

An infographic titled "30 User Experience Statistics You Should Not Ignore" from Experience Dynamics and UX Measure aggregated user experience (UX) statistics from a variety of sources including Google, Forrester Research, Salesforce, CapGemini, Nielsen, AppDynamics and the University of London [162]. A few notable statistics from that infographic include:

1. 52% of users said a bad mobile experience made them less likely to engage with a company
2. 90% of users reported having stopped using an app due to poor performance
3. 86% of users reported having uninstalled at least one mobile app because of poor performance
4. 73% of companies that were not currently conducting UX testing planned to do so within the next 12 months
5. In 2014, it was predicted that by 2020, customer experience (inclusive of UX) would overtake price and product as the key brand differentiator

As the statistics suggest, the more users interact with applications and web sites, the more important properly designed UX becomes. Human computer interactions, human factors, and the design of the user experience of the overall SecretMatch™ Prom system will be an important contributor to eventual success of the deployed application. A number of collections of design guidelines, or design rules, have been proposed over the years [163] [164]. While such design rules have at times received a bit of criticism, the importance of using design guidelines has been evidenced by broad adoption of such design rules in the software engineering industry [165] [166] [167], arguably with some level of success. Ideally, user experience testing might be performed with specific metrics and goals using a group of testers that is accurately representative of the expected user community. However, such user experience testing can be resource intensive and it does not preclude one from employing design guidelines to potentially improve designs prior to initiating the testing. While design rules are rarely universal, and they require interpretation in context, Jeff Johnson lent additional credibility to the use of design guidelines by

highlighting the cognitive and perceptual science underlying a number of common design guidelines [168]. Table 5.7 presents the set of design guidelines used to evaluate design choices during the evolution of the user experience for the initial client application. The collection is a combination of guidelines derived from Shneiderman/Plaisant (SP) [169], Nielsen/Molich (NM) [164], Stone et al. (S) [170], and Johnson (J) [168]. Each guideline is summarized along with one or more of the rule sets from which it was derived.

Table 5.7. Summary of design guidelines and sources[3]

| User Interface Design Guideline | Source(s) |
|---|---|
| 1. Consistency | SP, NM, S, J |
| 2. Visible Status / Informative Feedback | SP, NM, S, J |
| 3. Prevent Errors | SP, NM, S |
| 4. User Control | SP, NM |
| 5. Simplicity | SP, S, J |
| 6. Facilitate Learning & Recognition, not Recall | NM, J |
| 7. Task Focused | NM, J |
| 8. Design for Responsiveness | J |
| 9. Consider the Gestalt Principles of Proximity, Symmetry, Closure, & Common Fate | J |

### 5.4.4 An Example: User Selection of Fairness and Confidence

The HN protocol design affords the opportunity to fine tune the confidence and fairness values to achieve performance-related tradeoffs via the tunable gradual release process. Based on the work reported in [3], and further described in Chapter 4, the relative impacts of varying the confidence and fairness values for executions of the HN protocol

[3] SP = Shneiderman/Plaisant [174], NM = Neilsen/Molich [162], J = Johnson [166], S = Stone [175]

are well understood. Figure 3.5 showed that confidence and fairness values approach 100% confidence and perfect fairness as the number of bits increases. The results of performance testing summarized in Figure 4.6 further demonstrated that feasible runtimes were achieved even with large numbers of bits released, but that acceptable confidence might be achieved in less time with fewer bits released. An analysis of options for presenting this tradeoff to the user for a tunable gradual release process emphasized that it was not desirable to require users to have a detailed understanding of the HN protocol and its properties to successfully use the software to achieve fair and private matchmaking.

The foremost principle for consideration is that the product should afford privacy and security by default. That notion is in line with well-established security engineering and privacy by design principles [133]. For that reason, the default setting for "normal" operations should offer reasonably high confidence and fairness. Precisely what constitutes "reasonably high confidence and fairness" will vary depending on the context of the fair and privacy-enhanced matchmaking. For SecretMatch™ Prom, the default lower bound for confidence was established to be 99.9939% confidence that is achieved with 28 bits released, a process that averaged less than four seconds of runtime to accomplish as demonstrated during performance testing.

Having established a default value, the next step was to decide the appropriate level of granularity to offer to the user given the need to minimize the depth of knowledge required of the HN protocol to securely and confidently use the matchmaking system. In line with guidelines such as those of Shneiderman [163], Nielsen and Molich [164], and Johnson [168], the best option was to present two to three levels of granularity to the user

to foster rapid learning and user confidence in control, minimize short term memory load, minimize opportunities for errors, and minimize time requirements. Consequently, a simple slider control that would be familiar to users of modern operating systems was used to provide access to settings of *Normal*, *High*, or *Paranoid* levels of confidence with the default of *Normal* confidence equating to the aforementioned 28 bit release and 99.9939% confidence in the matching result. The settings above the default would clearly correspond with higher levels of confidence in the matching result, but slightly longer delays in being informed of matching results. A tool tip style notification was used to confirm the selection and concisely inform users of the tradeoff between performance and confidence in the matching result. Figure 5.4 presents an example of the confidence slider on a test version of the Settings page in the SecretMatch™ Prom app.

It should be noted that this analysis was specific to the context at hand for dating and relationships, but the factors being considered and the process used establish a model for decision making when the same facet of design is considered for future applications such as recruiting or voting negotiations in legislative bodies. For future work, user interface testing with empirical goals and metrics may be considered, akin to the process described in [171], to refine the user interface and maximize the benefit of user interface enhancements for a more positive overall user experience. Over time, such a process may also be considered for incorporation into the standard development practice for future releases of the system.

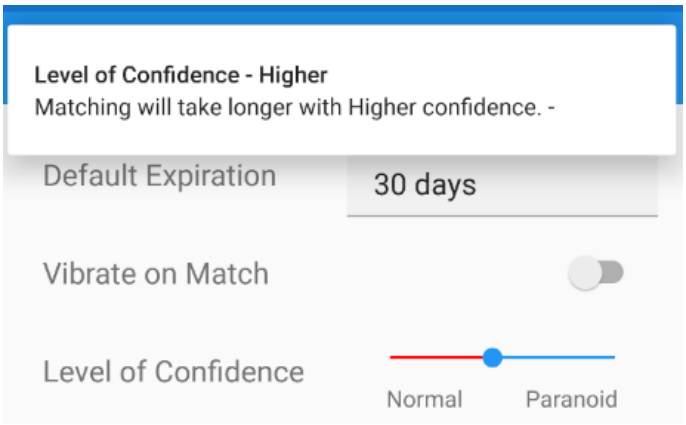



Figure 5.4. Sample user interface for setting desired level of confidence

### 5.4.5 Group Management

A number of approaches to group management are possible such as being handled by the matchmaker, a service that is peer to the matchmaker, a third party service, or management in a peer-to-peer fashion. The addition of group management to the untrusted matchmaker would add complexity and aggregate more information than might be desirable in a single location. The use of a separate peer or a third party service for group management would alleviate some of those concerns, but it would add additional required communications as well as another dependency. Meanwhile, peer-to-peer approaches have had some success, but that too would add unnecessary complexity to the application and the matchmaking process by adding more communication overhead and more potential for malicious activities. To avoid those problems, the TPP client application was designed to generate group identifiers in a consistent manner, thereby alleviating the need for another party to have knowledge of, or communicate regarding, the group manage-

ment activities. The group assignment approach devised for the present application design utilizes the output of an agreed upon, collision resistant one-way hash function applied to the defining attribute(s) of the group. For example, if *H(x)* represents application of the hash function to input *x*, sample group identifiers might include the following for the Prom focused application:

*H("School=Southlake High School")* or

*H( "School=Northlake High School" + "Gender=Male")*.

### 5.4.6 Identity Management

Similar to group management, a number of approaches to identity management are possible such as being handled by the matchmaker, a service that is peer to the matchmaker, a third party service, or in a peer-to-peer fashion. The addition of identity management to the untrusted matchmaker would add complexity and aggregate more information than might be desirable in a single location. It would also assign some level of trust to the untrusted matchmaker. The use of a separate peer or a third party service for identity management would alleviate some of those concerns, but it would add additional required communications as well as another dependency. The concept of identity is an important component of the functionality and usability of the system. At the same time, it is a critical contributor to the security and privacy afforded by the system. To understand the decisions made regarding identity, it is important to clarify the primary ways in which identities are used in the context of SecretMatch™. The main uses of identity for the prom matchmaking context include:

1. Access to contacts for matchmaking

2. Establish one's own public-private key pair
3. Access to public keys associated with contacts

For use case number one, in the initial application, contacts may be selected for matchmaking from either the local device's contacts list, or from Facebook. Contacts, sometimes referred to as friends, are considered to be uniquely identified by an email address and/or phone number. Access controls of the device, along with the user granting the SecretMatch™ Prom app permission to access the local device's contact list, are used with the assumption that identity has already been established in that context. Additionally, a login via Facebook feature was added to afford access to contacts from Facebook. But access to a Facebook user's contacts/friends list is one of a number of privacy relevant permissions that requires extensive review by Facebook prior to app approval. That final submission for review will occur when the app is nearing publication as it requires app walkthrough videos from the final app for consideration.

With regard to the second use case, a public-private key pair is generated for the specific devices and user upon initial execution and setup of the SecretMatch™ Prom app. The private key is then securely stored in the device's secure storage area, while the public key is registered with the key management service as associated with the primary device email address. As described in future work, the plan for future phases of development includes use of phone number as identifier, and verification via text message code.

Finally, access to the contacts, whether from the local device or Facebook friend lists, should grant access to email address and/or phone number, which can be used to retrieve public keys from the key management service. Akin to the approach taken for group management, the design uses the hash of an email address or password as the unique

identifier with which to associate key information. In this way, users can identify contacts without depending on an external identity management service. In this case, the identity management interface would simply compute and return the unique identifier locally, and all users should independently compute the same identifiers for the same identities. For other SecretMatch™ apps, the interface oriented design should allow for alternatives where third party involvement may be desirable for identity management.

### 5.4.7 Key Management

The area of key management is another important component of the infrastructure required for success of the HN protocol. Challenges with secure key management and public key infrastructure are common across industries and systems. For SecretMatch™ Prom, key management could be handled by the matchmaker, a service that is peer to the matchmaker, a third party service, or in a peer-to-peer fashion. In the prom context, the security sensitivity is not high enough to justify the potential usability challenges with taking a purely peer-to-peer approach. Doing so would add friction and the detraction from ease-of-use would almost certainly have a negative impact on adoption. The third party option was eliminated due primarily to cost given limited funding available. The primary goal with the key management decision was to strike the right balance in the tradeoff between cost, ease-of-use, and security. After assessing options, it was determined that the initial system would use a peer service to the matchmaker for key management. In this way, the key management service would be distinct from the matchmaker and avoid aggregation of too much information with a single entity.

### 5.4.8 Cryptographic Libraries

The selection of cryptographic libraries only impacts the client application given that the server does not perform cryptographic operations in the HN protocol. During testing of the proof-of-concept implementation, it was determined that a possible software defect in the RSAx libraries manifested itself in the `RSAx.RSAProcess` method with execution occasionally taking an incorrect logic branch when evaluating the sign of the `BigInteger` result from modular exponentiation. The result is an occasional incorrect output from the encryption or decryption operation. If RSAx were selected for the production system, the issue would need to be investigated, corrected as needed, and further testing would be required to give confidence that no additional problems remain. Designing and implementing complex cryptographic algorithms is a challenging endeavor, and even small errors can compromise entire protocols and systems. The selection of cryptographic libraries for the production system is a crucial contributor to the utility and security of the resulting system. Consequently, either corrective action may be necessary if RSAx were used for the production system, or an alternative would need to be chosen. The built-in .NET framework `System.Security.Crypgoraphy` namespace contains well tested and widely trusted implementations of RSA and other cryptographic operations. But it does not offer certain operations required for the HN protocol. Hence, the following three options were considered for the production system:

1. Correct the potential defect with RSAx libraries, perform significant testing, and use the final corrected version.
2. Augment the .NET Framework `System.Security.Cryptography` classes with support for additional required operations.

3. Select a trusted alternative implementation with support for the required operations.

Development of correct cryptographic implementations is a notoriously challenging task, and security experts generally recommend adoption of established, trusted, and well tested implementations where feasible to avoid defects that can significantly weaken the system. After careful analysis, the third option was determined to be the lowest risk option, with the Bouncy Castle collection of cryptographic APIs as the choice for the production system. The Bouncy Castle libraries are developed and maintained by a not-for-profit association named the Legion of Bouncy Castle Inc. [172]. They offer Java and C# based implementations that have obtained FIPS certification via the NIST Cryptographic Module Validation Program [173]. Note that the Bouncy Castle libraries are available under an academic license [174] that is comparable to the MIT license [175], and is consequently compatible with business usage and intellectual property protection.

### 5.4.9 Temporal Constraints

The notion of temporal constraints, which add value to the overall system by effectively assigning an expiration date to inquiries about ILW, was prioritized for inclusion with the initial system. To understand why, consider that the ILW "Alice + Bob + attend prom together" might be a privacy threat after the prom event is over (e.g., imagine that Bob actually attended with Eve) and such an inquiry should thusly be considered invalid after a certain point in time. The ideal solution would be to identify a sort of self-destruct mechanism whereby challenges or counter-challenges are no longer valid after some point in time. That would avoid having to trust either the matchmaker or the client side

to filter out expired requests. However, given the challenges with that approach, the initial implementation associates an expiration date with challenges, and relies on the matchmaker to remove expired database entries and check for expiration prior to returning results. This is in the best interest of the matchmaker as well, given that it will help limit the size of the database to manageable levels and avoid excess capacity requirements due to invalid, expired data. The enhanced notion of self-destructing match attempts has been identified as an opportunity for future research.

Implementing support for temporal constraints involved the following core elements:

1. Addition of Expiration Date to the matchmaker API
2. Addition of Expiration Date to the database schema
3. Implementation of database clean-up maintenance activities to remove expired match data for performance reasons
4. Addition of logical checks with the server-side REST API processing to avoid returning any expired match data that have not yet been purged from the database.

### 5.4.10 Communication Protocols

The proof-of-concept implementation utilized Google Protocol Buffers as described previously. Another key design decision involved selection of the communication protocol and support libraries for the production system. The two leading candidates for consideration include continuing to use Google Protocol Buffers (GPB), or switching to JavaScript Object Notation (JSON) data with a Representational State Transfer (REST) API. Although GPB performed well in the proof-of-concept phase, REST APIs have

become somewhat of a de facto standard across a large segment of web-based applications with APIs. In general, an analysis of advantages and disadvantages of each approach determined that GPB would most often be the better fit for internal interfaces within component based systems or microservices architectures while standard REST APIs tend to be more advantageous for externally accessible web interfaces. As Mark Masse stated in [176], REST is a "description of how the World Wide Web works" and if the Web were considered to have an operating system, "its architectural style is REST." Hence, to maximize opportunities for future expansion and interoperability, a REST API was selected as the communications mechanism for the production system. Table 5.8 presents an descriptive overview of the REST API for the Matchmaker service.

Table 5.8. Matchmaker REST API description

| HTTP Method | URI | Inputs | Action |
|---|---|---|---|
| PUT | /api/v1/x | GID, X, Exp | Persist GID, X to DB |
| GET | /api/v1/x | GID | Return X for GID |
| PUT | /api/v1/y | GID, X, Y | Persist GID, X, Y to DB |
| GET | /api/v1/y | X | Return Y for X |
| POST | /api/v1/vab-bit | Index, bit, Y | Persist bit of $V_{AB}$ |
| GET | /api/v1/vab-bit | Index, Y | Return bit of $V_{AB}$ |
| GET | /api/v1/vab-int | Y | Return $V_{AB}$ as integer |
| GET | /api/v1/vab-bin-str | Y | Return $V_{AB}$ as binary string |
| GET | /api/v1/test | -- | Return test confirmation |
| PUT | /api/v1/test | -- | Return test confirmation |
| POST | /api/v1/test | -- | Return test confirmation |
| DELETE | /api/v1/test | -- | Return test confirmation |

A REST API "Design Rulebook" [176] commonly applied in industry was employed to ensure consistent implementation of best practices. A few of the most notable guidelines that were adhered to for the initial matchmaker API include (refer to [176] for details and justification):

- Use '-' (hyphen) rather than '_' (underscore) to improve URI readability.
- Prefer lower case letters in URI paths.
- Do not use CRUD (Create, Read, Update, Delete) function names in URIs.
- Use the `no-cache` directive only if absolutely necessary. Using a small `max-age` value instead can at least allow some advantages from caching.
- Use PUT to insert a new resource or update/replace an existing resource.
- Use POST to add a new resource within a collection or to execute a function-oriented controller.
- Use the subset of standard status response codes from RFC 2616 that apply to the REST API design.
    - Do not use 200 ("OK") and communicate errors in the body.
    - Do not use 302 ("Found").

## 5.5 The SecretMatch™ Privacy-Enhanced and Fair Matchmaking System

After practicing sound engineering processes to support critical design decisions with long-term consequences, the initial SecretMatch™ production system was developed. Multiple research phases including protocol design, security analysis, proof-of-concept development, and experimentation for feasibility analysis formed a sound basis and reduced risk prior to incurring the full costs of development of the production system. The

resulting initial matchmaking system is now detailed, divided into discussions that elaborate on the details for the Matchmaker (server) as well as the SecretMatch™ Prom app (client) respectively. The archives of source code and related files are described in Appendix F.

### 5.5.1 The SecretMatch™ Matchmaker (Server)

The heart of the initial implementation of the server-side SecretMatch™ Matchmaker component is the publicly accessible database as described in Section 5.4.1. The backend uses SQL DDL to create a database that is compliant with the interface of MySQL and MariaDB. The development and test phases used MariaDB. However, many web hosts and service providers provide MySQL databases. The Matchmaker implementation is compatible with either.

Rather than requesting application designers to interface directly with the public database, the Matchmaker instead implements a REST API through which client applications can interact with the database, as described in Section 5.4.10. The server was implemented in Python and it leverages Flask and Flask-Restful packages. Flask is a lightweight implementation of the Python Web Server Gateway Interface (WSGI) [176], a standardized interface between Python web frameworks and Web servers. Flask was chosen for its simplicity relative to popular competing frameworks like Django. The Flask-Restful extension adds support for lightweight RESTful API development on top of Flask [178]. Additionally, the PyMySQL package [179] was used to interface with the MySQL database [180] as well as MySQL compatible forks like MariaDB [181]. Appendix E describes the steps necessary to reproduce the server-side development envi-

ronment including Python, dependency packages, a light-weight integrated development environment (IDE), REST API test tools, and the backend database.

After installing and configuring MySQL or MariaDB, the SQL DDL from the file CreateTPPDatabase_v*.*.sql can be used to create the Matchmaker's database. Upon creation, Figure 5.5 shows the result from logging in, selecting the matchdb database, and executing the "show tables" command. Note that the screenshot was created on a Windows based VM, but the database, SQL, and DDL are all cross-format and work just as well on Linux and other Unix-like operating systems supported by the chosen database (MySQL or MariaDB).

```
C:\>mysql -u root -p
Enter password: ****************
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 9
Server version: 10.2.14-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use matchdb;
Database changed
MariaDB [matchdb]> show tables;
+--------------------+
| Tables_in_matchdb  |
+--------------------+
| challenge          |
| matchattempt       |
| xyassoc            |
+--------------------+
3 rows in set (0.00 sec)
```

Figure 5.5. Sample console output from creation of matchmaker database

Beyond the SQL DDL file for creation of the database, the following constitute the most important Python files with core components of the Matchmaker server.

- Main.py – As the main entry point of the web service, this file defines request parsers, classes and methods to process HTTP requests, definition of routes for the REST API, and the host app itself including port and IP information.

- HelperFunctions.py – The helper functions defined in this file perform more extensive validation beyond that of the Flask request parsers, and pass resource data on to the Matchmaker class after validation, where appropriate. Consequently, this file also defines the instance of the Matchmaker.
- Matchmaker.py – The Matchmaker class defines the interface to the matchmaking database, defining the database interface logic corresponding to the resources and HTTP methods invoked via the REST API.

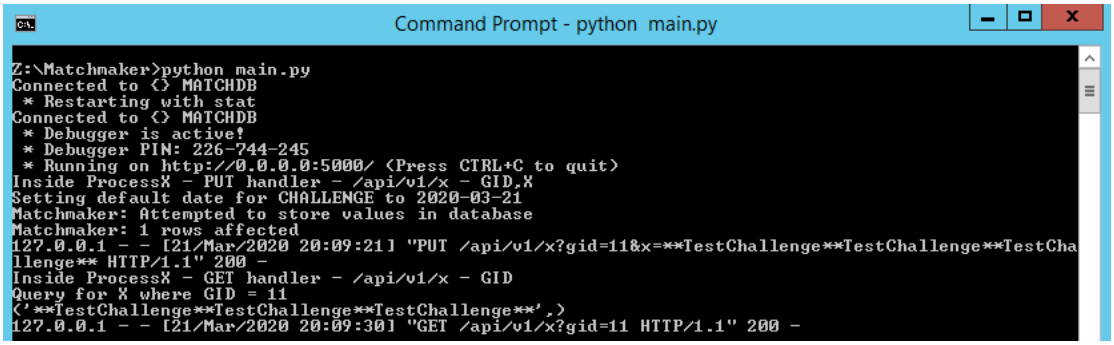



Figure 5.6. Example console output from matchmaker with debugger

At startup, the Matchmaker establishes a connection to the database in preparation to respond to matchmaking related requests such as storing a challenge or counter-challenge value, and gradual release of bits of a verifier. Additionally, the Flask routes and configuration are setup, and the web server starts listening for HTTP requests to process.

To start the Matchmaker server-side software manually, simply execute `python main.py` from the command line. Figure 5.6 shows console output from starting the Matchmaker with a debugger active and processing PUT and GET methods associated with the challenge X, but using human readable test values. Figures 5.7 and 5.8 show

examples of the use of the RESTED Firefox extension [181] and Postman [182] respectively for REST API testing. The tests executed in each of the screenshots correspond with the debugging output from the server in Figure 5.6.

To prevent certain abuse cases, an API key feature was added to the Matchmaker. A straightforward approach to requiring an API key was to use the decorator pattern. To do so, the `wraps` decorator factory was imported from `functools`. After defining the decorator function using the `@wraps` decorator, then requiring the API key was as simple as adding the `@require_appkey` decorator before each route handler.

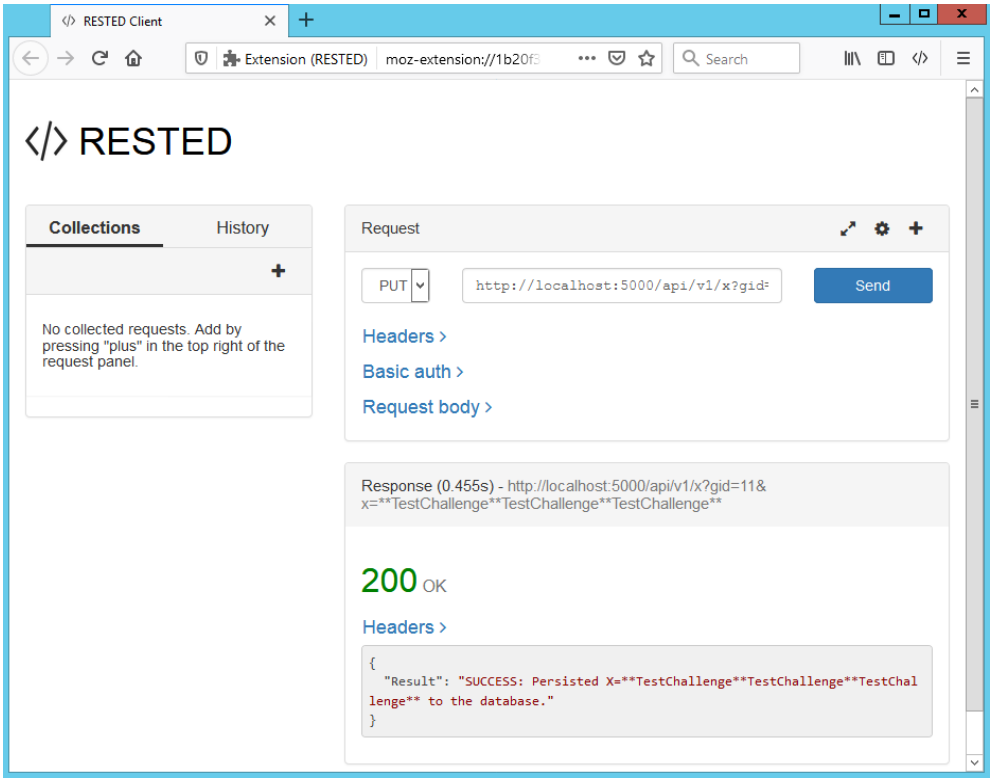



Figure 5.7. Example REST API testing with RESTED Firefox extension [182]

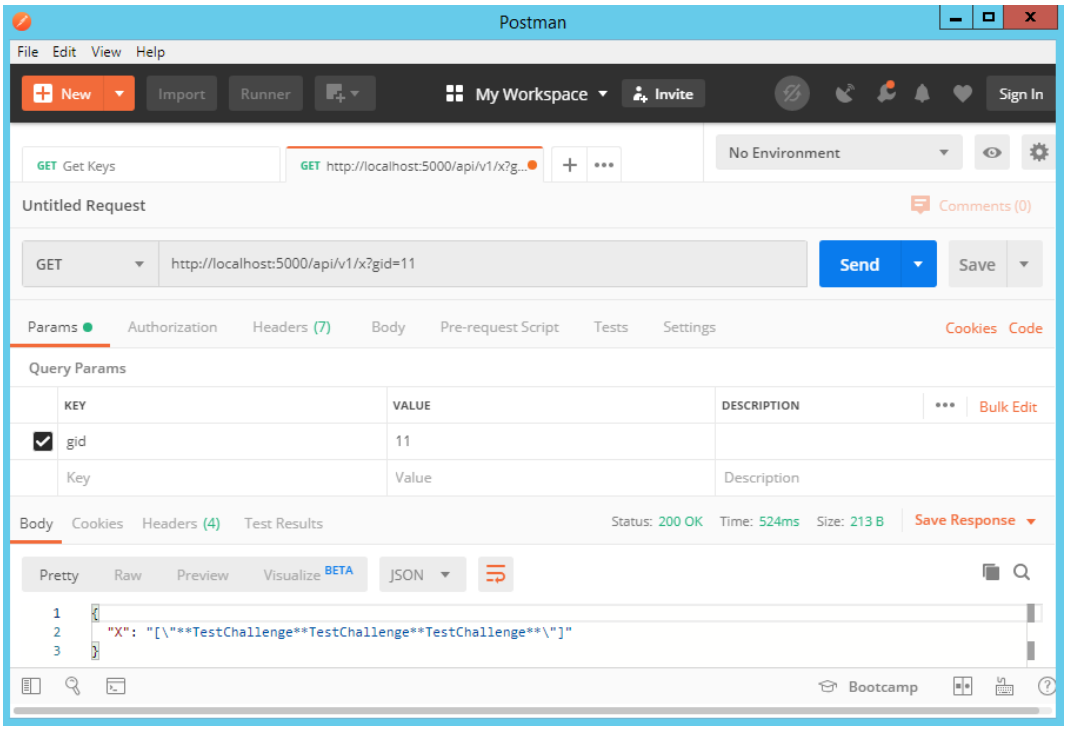



Figure 5.8. Example REST API testing with Postman [183]

### 5.5.2 The SecretMatch™ Prom App (Client)

Modern applications often endeavor to support multiple platforms like Android, Apple iOS, and Windows based devices. The three ecosystems have different user interface paradigms, native applications written in different programming languages (e.g., Java, Objective-C/Swift, and C#/C++/etc. for Android, iOS, and Windows respectively), and different APIs to interact with the operating system. Rather than implementing separate apps in different programming languages, a cross-platform solution was desirable such as Xamarin [184], Flutter™ [185], or via Progressive Web Apps with technologies like React or React Native [186]. Given the long-term implications of making the right technology choice, a formal decision making process was used as described in Section 5.2.1.

Upon completion of the formal decision making process, C#/Xamarin was chosen for the client side app. Once considered to be mostly a language for programming Windows applications, the open source Mono project [187] has endeavored to bring a cross-platform experience to C# developers by extending support beyond Windows to macOS and Linux. More recently, the growing adoption of .NET Standard, .NET Core, and Xamarin.Forms have further fostered adoption of the familiar C# language for cross-platform applications reaching Android, iOS, and more.

A typical Xamarin app has one or more cross-platform projects in which code is common across all platforms. Then, there is also a project corresponding with each specific targeted platform, where any platform specific code usually resides along with platform specific resources. The SecretMatch™ Prom app consists of a `SecretMatchProm` cross-platform project with a large majority of the code. There are also `SecretMatchProm.Android`, `SecretMatchProm.iOS`, and `SecretMatchProm.UWP` projects used to build the applications specific to each operating system (Android, iOS, and Windows). The Xamarin.Forms library abstracts away many of the platform specific differences with regard to user interface frameworks, and instead presents a common framework of controls, layouts, pages, and navigation options [188].

The overall organization of the SecretMatchProm solution is shown in Figure 5.9. It also shows the detailed hierarchical structure of the SecretMatchProm cross-platform project, which contains a majority of the code, as well as the structure of the SecretMatchProm.Android project with Android specific resources. With C# and Xamarin, user interfaces can be handled declaratively via eXtensible Application Markup Language (XAML), programmatically with C# code, or via a mixture of the two.

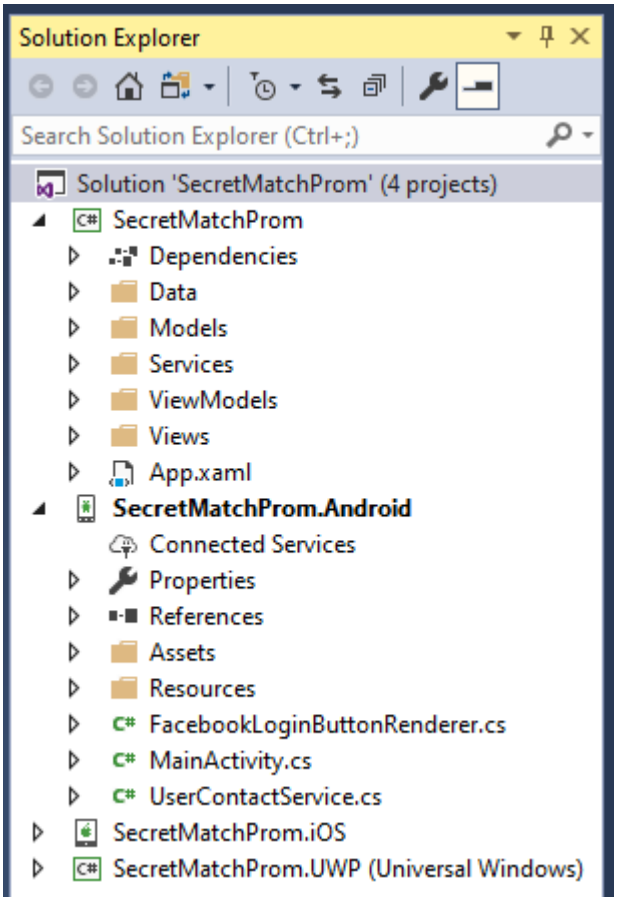



Figure 5.9. SecretMatch™ Prom solution and project organization

Like many applications with XAML based user interfaces, the SecretMatch™ Prom app makes use of the Model-View-ViewModel (MVVM) architectural design pattern. The MVVM pattern facilitates clean separation between the view (user interface), model (data), and the view-model (business logic). Figure 5.10 shows a conceptual overview of the MVVM approach as applied to the C#/Xamarin based solution.

An example of the declarative nature of XAML is shown in Figure 5.11, which contains markup code from ShareAppPage.xaml, one of the simplest user interface compo-

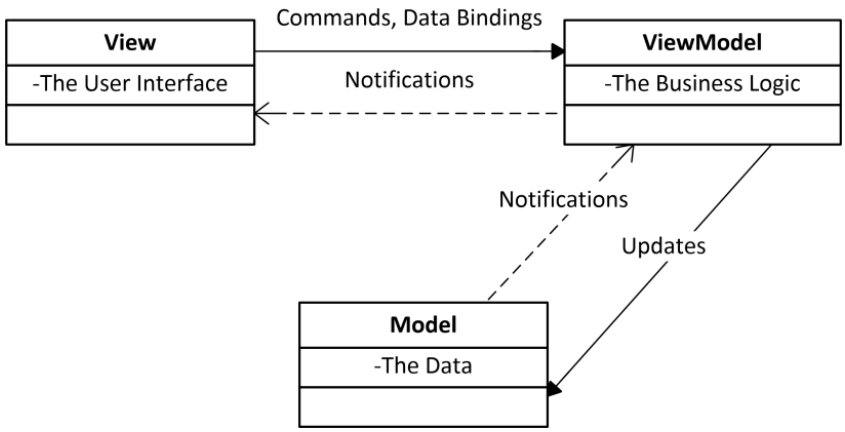



Figure 5.10. Conceptual overview of the MVVM architecture

nents in the SecretMatch™ Prom app. The outermost visual element is the `ContentPage`. It contains properties and controls such as `ContentPage.BindingContext`, `ContentPage.Resources`, and a `Grid` layout along with multiple `Image` elements and a `Button` control.

Note that the "+" symbols in the left hand column represent collapsed sections of code for brevity of display. Each control also has its own properties such as `Padding`, `Thickness`, `Height`, `Margin`, `Text`, and so on that allow declarative customization. Properties like `Source` and `Clicked` can define a data source and an event handler function respectively. In the case of an event handler, the code implementing the handler for `ShareAppPage.xaml` might be in either `ShareAppPage.xaml.cs` or in `ShareAppViewModel.cs` (corresponding to the ViewModel in MVVM). An example of what the associated `ShareAppViewModel.cs` file with class definition might look like is given in Figure 5.12. In the example, the `ICommand` interface is used to define event

handlers that utilize the `TryOpenAync` method from the `Xamarin.Essentials.Launcher` class.

```
1   <?xml version="1.0" encoding="utf-8" ?>
2   <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3                xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4                x:Class="SecretMatchProm.Views.ShareAppPage"
5                xmlns:vm="clr-namespace:SecretMatchProm.ViewModels"
6                Title="{Binding Title}"
7                Visual="Material">
8       <ContentPage.BindingContext>
9           <vm:ShareAppViewModel />
10      </ContentPage.BindingContext>
11      <ContentPage.Resources>
12          <ResourceDictionary...>
18      </ContentPage.Resources>
19      <Grid>
20          <Grid.RowDefinitions>
21              <RowDefinition Height="Auto" />
22              <RowDefinition Height="*" />
23          </Grid.RowDefinitions>
24          <ScrollView Grid.Row="1">
25              <StackLayout Orientation="Vertical" Padding="16,10,16,10" Spacing="10">
26                  <Grid>
27                      <Grid.RowDefinitions>
28                          <RowDefinition Height="Auto" />
29                      </Grid.RowDefinitions>
30                      <Grid.ColumnDefinitions>
31                          <ColumnDefinition Width="Auto" />
32                          <ColumnDefinition Width="Auto" />
33                      </Grid.ColumnDefinitions>
34                      <Image Source="Facebook128.png" Grid.Row="0" Grid.Column="0"
35                             HorizontalOptions="CenterAndExpand" />
36                      <Image Source="Twitter.png" Grid.Row="0" Grid.Column="1"
37                             HeightRequest="60" WidthRequest="60"
38                             HorizontalOptions="CenterAndExpand" />
39                  </Grid>
40                  <Button Margin="0,10,0,0" Text="Share SecretMatch™ Prom"... />
43              </StackLayout>
44          </ScrollView>
45      </Grid>
46  </ContentPage>
47
```

Figure 5.11. Sample XAML code from ShareAppPage.xaml

```
1   // ****************************************
2   // Filename: ShareAppViewModel.cs
3   // Copyright (c) 2020, Dwight Horne
4   // All rights reserved.
5   // ****************************************
6   using System;
7   using System.Windows.Input;
8
9   using Xamarin.Forms;
10
11  namespace SecretMatchProm.ViewModels
12  {
13      public class ShareAppViewModel : BaseViewModel
14      {
15  ...
18
19          public ShareAppViewModel()
20          {
21              Title = "Share SecretMatch™ Prom";
22
23
24              EmailPrivacyFairyCommand = new Command(() =>
25                      Xamarin.Essentials.Launcher.TryOpenAsync(new Uri(PRIVACY_FAIRY_URI)));
26
27              EmailSecretMatchPromCommand = new Command(() =>
28                      Xamarin.Essentials.Launcher.TryOpenAsync(new Uri(SECRET_MATCH_URI)));
29          }
30
31          public ICommand EmailPrivacyFairyCommand { get; }
32          public ICommand EmailSecretMatchPromCommand { get; }
33      }
34  }
```

Figure 5.12. Sample C# code from ShareAppViewModel.cs

Breaking down the SecretMatchProm project by sub-folder, the Models folder contains classes that encapsulate the data being stored, represented, or otherwise processed. The Models project includes the following C# files, which describe some of the elements of the `SecretMatchProm.Models` namespace:

- Event.cs – Details of a prom event
- HomeMenuItem.cs – Defines menu item types and home menu item data
- Item.cs – Items details (e.g., Person) from Master-Detail ContentPage pattern
- MatchStatus.cs – Encapsulates status of a match attempt
- Msg***.cs – Various classes represent messages used with REST API
- SchoolData.cs – Encapsulates data for high schools, loaded at runtime

- UserContact.cs – An app user, person from contact list, Facebook friend, etc.

The Services folder contains files that define the `SecretMatchProm.Services` namespace. These consist primarily of service interfaces, service implementations of those interfaces, and support classes for the services. Some of the pertinent files include:

- ByteAppend.cs – Convenient methods for manipulating collections of bytes
- CryptoKeyApiManager.cs – Implementation of ICryptoKeyApiManager
- CryptoService_RSA.cs – RSA-based cryptographic service provider
- FacebookApiManager.cs – Encapsulates Facebook API specific features
- HashService_SHA.cs – SHA-based hash service for speed, not security
- HNProtocolService.cs – Implements the details of the HN protocol service
- ICryptographyService.cs – Interface for cryptography services
- ICryptoHashService.cs – Interface for cryptographic hash services
- ICryptoKeyApiManager.cs – Interface for crypto key management service
- IDataStore.cs – Interface for data store for contact/friend information
- IMatchingApiManager.cs – Interface for matchmaking services
- MatchingApiManager.cs – Implementation of matchmaking service
- MockCryptoKeyApiManager.cs – Mock crypto key management for testing
- MockDataStore.cs – Mock data store of contacts/friends for testing
- MockUserContactService.cs – Mock service to retrieve contact/friend data
- UserContactService.cs – Service to retrieve contact/friend data

The ViewModel folder contains files with definitions of the classes composing the `SecretMatchProm.ViewModels` namespace. These classes correspond with the ViewModel role in MVVM. Some examples of elements in this namespace include:

- AboutViewModel.cs – Business logic for the *About* information page
- BaseViewModel.cs – Core business logic shared by most ViewModels
- FBLoginViewModel.cs – Business logic supporting Facebook login
- FeedbackViewModel.cs – Business logic for the *Feedback* page
- ItemDetailViewModel.cs – Business logic behind the contact details page
- ShareAppViewModel.cs – Business logic for *Share App* page

Finally, the Views folder within the SecretMatchProm cross-platform project define the user interface elements in a common way that translate to device specific experiences when built for target platforms. Some examples of files describing components of the `SecretMatchProm.Views` namespace include:

- AboutPage.xaml – XAML describing the layout/contents of *About* page
- AboutPage.xaml.cs – Code-behind for the *About* information page
- CheckMatchStatusListPage.xaml – XAML describing *Match Status* page
- CheckMatchStatusListPage.xaml.cs – Code-behind for *Match Status* page
- ChooseSchoolPage.xaml – XAML describing *Choose School* page
- ChooseSchoolPage.xaml.cs – Code-behind for *Choose School* page
- FacebookLoginButton.cs – Facebook login button, bindings, and handlers
- FBLoginPage.xaml – XAML describing *Facebook Login* page
- FBLoginPage.xaml.cs – Code-behind for *Facebook Login* page

- FeedbackPage.xaml – XAML describing *Feedback* page
- FeedbackPage.xaml.cs – Code-behind for *Feedback* page
- ItemsPage.xaml – XAML describing items (Contacts) in Master-Detail pattern
- ItemsPage.xaml.cs – Code-behind for items (Contacts)
- ItemDetailPage.xaml – XAML describing item (Contact) details in Master-Detail pattern
- ItemDetailPage.xaml.cs – Code-behind for item (Contact) details in Master-Detail pattern
- MatchDetailsPage.xaml – XAML describing *Match Details* page
- MatchDetailsPage.xaml.cs – Code-behind for *Match Details* page
- MenuPage.xaml – XAML describing *Menu* page
- MenuPage.xaml.cs – Code-behind for *Menu* page
- SettingsPage.xaml – XAML describing *Settings* page
- SettingsPage.xaml.cs – Code-behind for *Settings* page
- ShareAppPage.xaml – XAML describing *Share App* page
- ShareAppPage.xaml.cs – Code-behind for *Share App* page
- TryMatchingPage.xaml – XAML describing *Start a Matching* page
- TryMatchingPage.xaml.cs – Code-behind for *Start a Matching* page

Aside from the cross-platform SecretMatchProm project, a small amount of platform specific code is required in certain cases. For instance, the SecretMatchProm.Android project contains the Android specific code, while the SecretMatchProm.iOS and SecretMatchProm.UWP will contain small amounts of code specific to the Apple iOS

and Windows (Universal Windows) platforms respectively. Examples of platform specific requirements for the Android version of SecretMatch™ can be found in the following files describing elements of the `SecretMatchProm.Droid` namespace:

- MainActivity.cs – General platform specifics like registering and initializing `DependencyService` factory methods for retrieving platform specific implementations where required (e.g., with Toast Notifications)
- FacebookLoginButtonRenderer.cs – Handles specifics of Facebook login button functionality with the Facebook SDK for Android
- UserContactService.cs – Leverage `DependencyService` factory for Android specific API to retrieve Contact information stored on the device

The concept of operations for the resulting Android-based client application is now presented in a use-case oriented ordering with a focus on the primary tasks a typical SecretMatch™ Prom user might want to perform. To initiate a private matching inquiry, the user would first either choose a person of interest, or select the school holding the prom event of interest. The photos used for testing purposes during the final academic research phases were from generated.photos, a source of unique photos with professional quality appearance that were produced using generative machine learning [189]. The Use License grants permission for non-commercial usage. If SecretMatch™ were to become commercial in nature (e.g., through corporate or venture capital funding), then either usage must cease, or a paid license would need to be acquired.

**Task 1: Select a friend or contact of interest for matching.** On the Contacts page, the user can scroll through the list or use search for a large number of contacts. Selecting a contact shows the Contact Details page, from which one could start a secret matching

inquiry if a school's prom event has already been selected. However, if a particular school's prom event has not yet been selected, the user would get a prompt to select an event, the next step in the process. The screenshots of Figure 5.13 show screens scrolling through the list of contacts, searching for a contact, and viewing contact details.

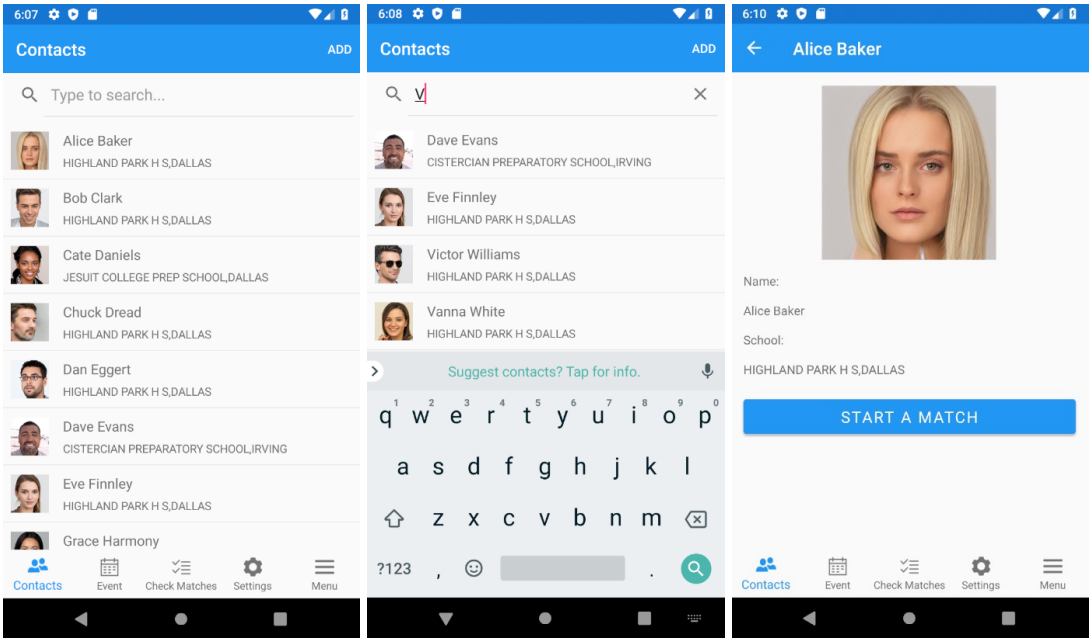



Figure 5.13. Selecting a contact/friend of interest for private matching

**Task 2: Select a school hosting the prom event of interest.** Selection of a particular prom event was simplified down to selection of State (within the United States of America), School, and Year. The first step of selecting the State of interest is to narrow the list of potential schools to choose from, and to simplify the school selection process by avoiding ambiguities from schools across state lines that have the same name. States like California and Texas, for example, each have thousands of prom-eligible schools

within their borders. The use of the device's location was considered for automatic selection, but border cities can add significant complexity in that case.

During event selection, the first iteration of the app only allows selection of the current year, or the following year. This is for security reasons, but also for improved usability through prevention of accidental errors (selecting event in the past, or far too distant future). Figure 5.14 shows the Event page with state of TX selected, the Choose School page showing search functionality for ease of use, and the toast notification that confirms selection of a particular school's prom event.

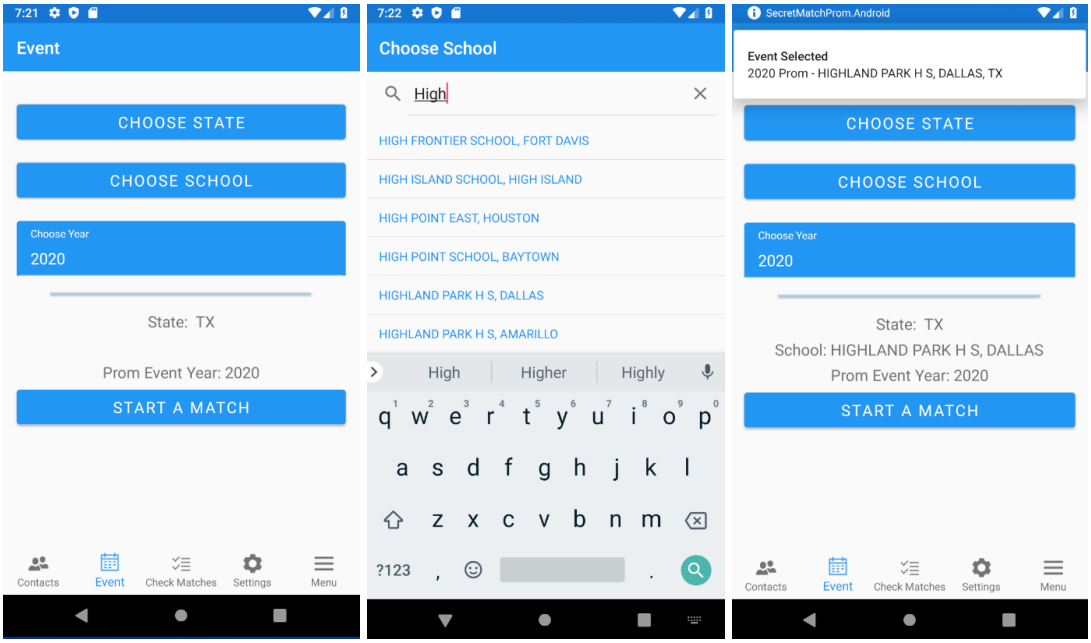



Figure 5.14. Selecting the school holding the prom event of interest

**Task 3: Initiate a private matching inquiry.** The user can initiate a private matching inquiry from either the Contact Details page, or from the Event page, as long as both a Contact and an Event have been selected. A confirmation prompt asks the user to con-

firm details and intent to match to avoid accidental private matching inquiries. Figure 5.15 provides screen shots related to initiating a matching inquiry. The images from left to right show options upon selection of a Contact from the Contact Details page, options upon selection of event details from the Event page, and confirmation upon initiating a private matching inquiry from the Start a Matching page. Upon receiving the toast notification, the "Started Matching Attempt" message confirms that the challenge $X$ in the HN protocol has been computed and submitted to the Matchmaker.

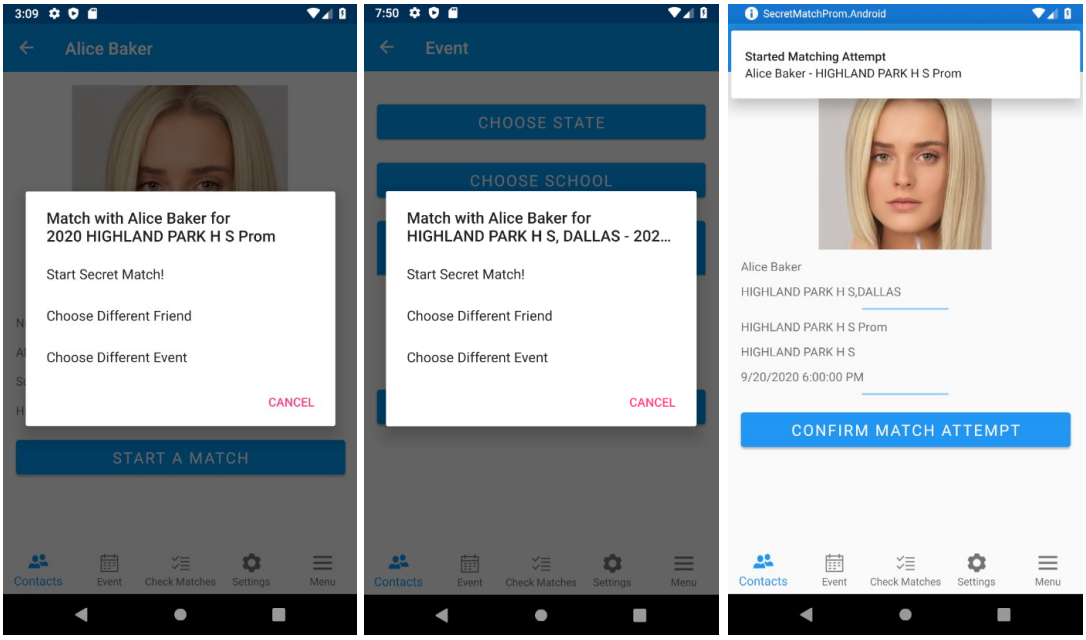



Figure 5.15. Initiating a private matching inquiry (send challenge to server)

**Task 4: Check status of private matching inquiries.** To check on the status of matching inquiries, the user selects the Check Matches option from the navigation bar. Figure 5.16 shows the user interface representing the list of match inquiries with associ-

ated status. The status is conveyed via recognizable iconography as well as text synopsis including:

- Success! You found a SecretMatch!
- Match failed, but still a secret!
- No match yet, but still a secret!

Selection of a match status item in the list shows a Match Details page with the pertinent details for the matching attempt such as contact information, event information, and the status of the matching attempt. The status conveying that matching is in progress can represent either awaiting a counter-challenge or performing the gradual release process in the HN protocol.

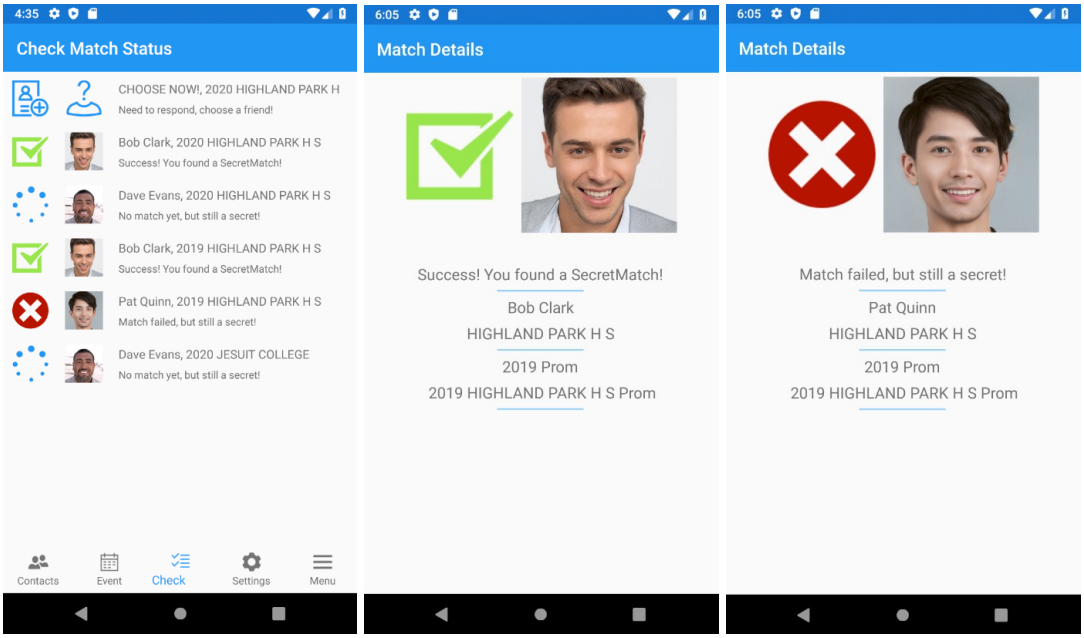



Figure 5.16. Checking status of private matching inquiries

**Task 5: Respond to a private matching inquiry.** When the user has a challenge available, it is communicated to the user via icons for "unknown person" and "add contact" for the applicable contact and match status respectively on the Match Status Page. Figure 5.17 shows the process of responding to a private matching inquiry. Tapping a row needing selection of a candidate contact for matching prompts the user, "You have a secret admirer!" It gives the prom event information and asks if the user would like to select a friend for attempted private matching. Upon selecting "YES", the user's contact list is displayed. Then, when the user selects a contact for private matching, the backend processing computes the appropriate value for counter-challenge $Y$ and submits the results back to the Matchmaker's database. Finally, the match status updates to show that the matching is in progress.

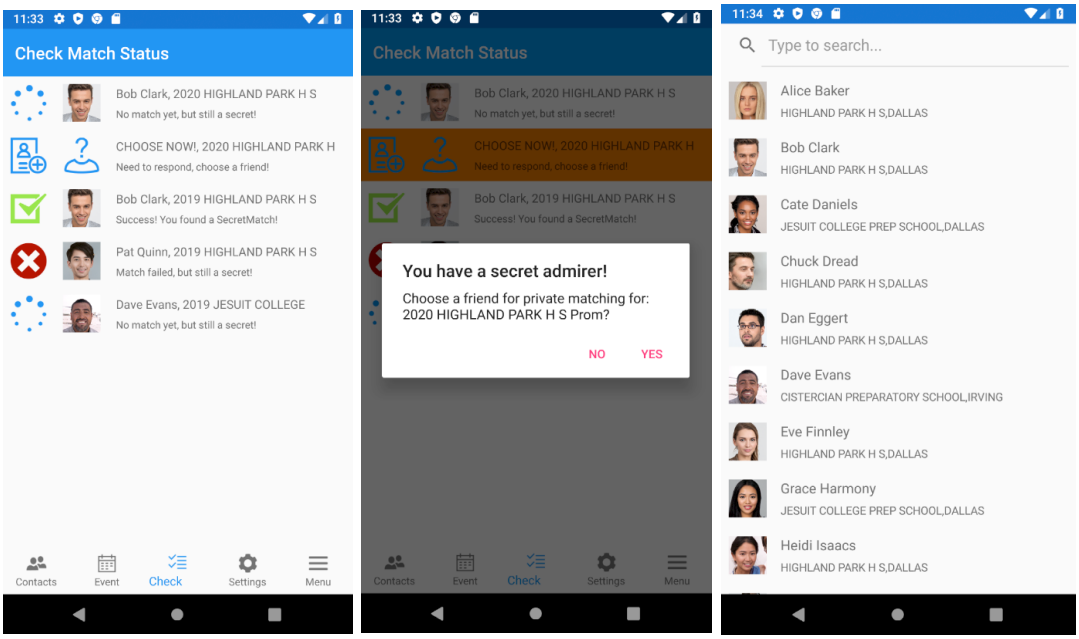



Figure 5.17. Responding to private matching inquiries

**Task 6: Optionally sign in to Facebook for additional friend options.** Figure 5.18 shows the user interface for logging in to Facebook to potentially gain access to more friends for private matching. The process uses the Facebook API through which the user logs in and grants the app the requested permissions. The application saves off a token that can be used for subsequent Graph API calls for a limited period of time. Fortunately for user privacy, but unfortunately for well-meaning app developers, greater restrictions have been added to Facebook permissions such as access to the user's friends list. Those restrictions bring return on investment from this feature into question. That topic is further discussed in Section 6.2 as a tradeoff analysis that will be needed with future work.

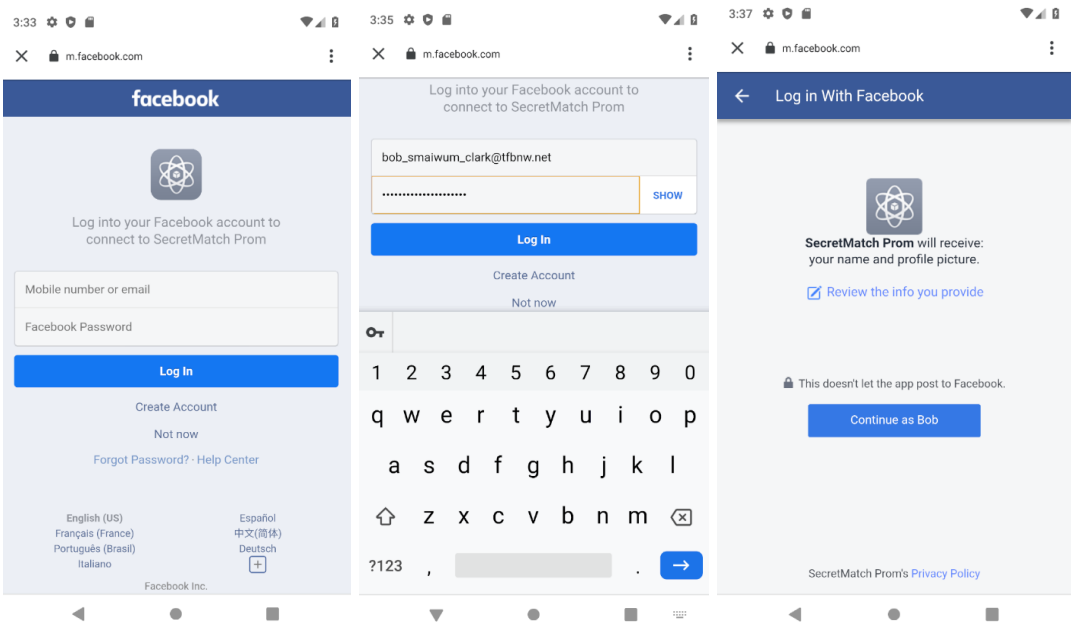



Figure 5.18. Signing in to Facebook for additional friend options

**Task 7: Customize the experience via application settings, review app information, or share the app with friends.** Figure 5.19 presents an example of the Settings Page where the user can change the default expiration for private matching inquiries, enable vibration when a match is confirmed, or change the level of confidence in the matching result from *Normal* to *Higher* or *Paranoid* settings. A toast notification also informs the user of the tradeoff with additional time required for higher levels of confidence. From the About Page, the user can read about the app, view the privacy policy, or view the terms and conditions of usage. The Figure also shows the Menu page from which the user can access ancillary features that were deemed not sufficiently frequent tasks to warrant their own tabs on the navigation bar.

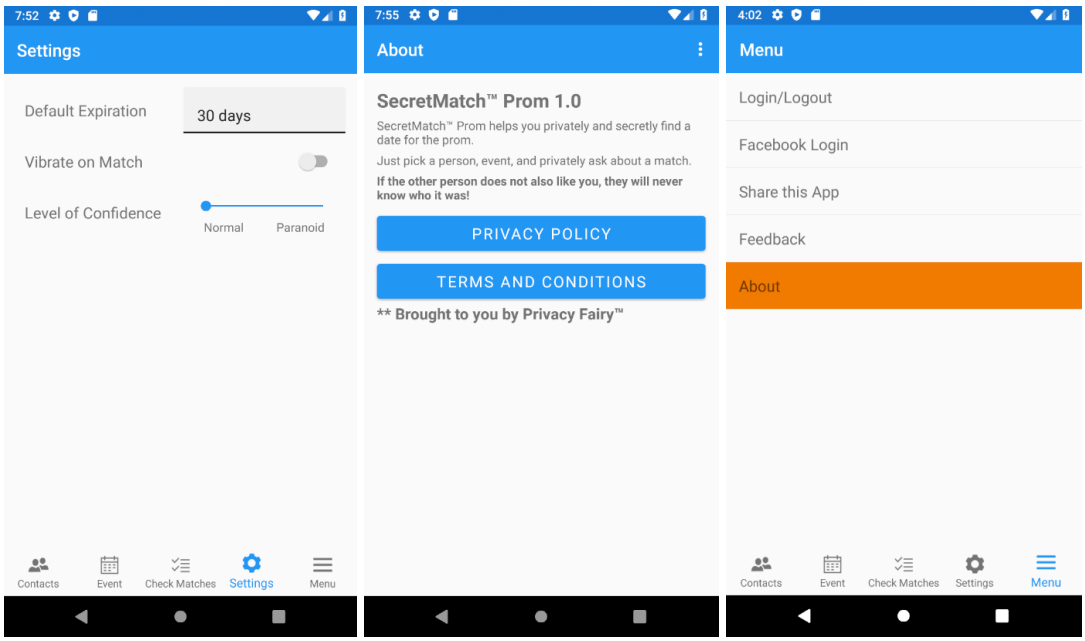



Figure 5.19. Customizing application settings, sharing the app, and feedback

In summary, the SecretMatch™ Prom app leverages a familiar navigation framework and a task oriented perspective for fair and privacy-enhanced matchmaking with ILW. The graphical interface and overall user experience design hides much of the complexity of the HN protocol and matchmaking process, while still affording privacy protections and the ability to customize aspects of matchmaking such as the default expiration for a given challenge (i.e., matching inquiry) and the level of confidence desired.

## 5.6 Preparing SecretMatch™ for Quantum Enabled Adversaries

The advent of quantum enabled adversaries introduces new challenges to many current technologies that rely on classical notions of computational hardness as a basis for security or privacy guarantees. In particular, two threats that become practical with the introduction of working quantum computers with sufficiently large numbers of qubits include Peter Shor's period finding algorithm that can be used to accomplish polynomial-time factoring of large primes [190] [191] and Grover's search algorithm that can be used to accomplish quadratic speed-up with finding pre-images of a function [192] [193]. These were the primary threats under consideration when evaluating the security of the HN protocol against quantum-enabled adversaries. They are also very real threats under consideration by the industry at large given potential vulnerability of commonly used security foundations such as RSA, Diffie-Hellman, elliptic curve cryptography (ECC) [194], and even cryptographic hash functions.

### 5.6.1 Post-Quantum Cryptography

One possible path to a quantum resistant variant of the HN protocol involves post-quantum, or quantum resistant, cryptography. The National Institute of Standards and Technology (NIST) in the United States validated concerns about the threats to security posed by quantum computers with the launch of its Post Quantum Cryptography (PQC) project in 2012 [195]. NIST followed by announcing a world-wide PQC competition in 2016. The goal of the competition is to evaluate proposed algorithms hypothesized to be secure against quantum computers, eventually leading to PQC standardization. Select milestones highlighting the progression of the PQC competition include [196] [197]:

- Feb 2016 – NIST announced PQC "competition" at PQCrypto 2016
- Nov 2017 – Deadline for first round submissions
- Apr 2018 – First NIST PQC Standardization conference
- Jan 2019 – NIST published NISTIR 8240 first round status report
- Mar 2019 – Deadline for 2nd Round submissions
- May 2019 – NIST presentation at PQCrypto 2019 conference: *Round 2 of the NIST PQC "Competition" – What was NIST Thinking?* [197]
- Aug 2019 – Second NIST PQC Standardization conference
- 2020 – 2021 – Algorithm selection or Third Round of standardization process

The primary selection criteria include security, cost/performance, and algorithm/implementation characteristics. The competition began with 82 submissions in the first round which were narrowed to 26 candidates for the second round involving 157 different submitters from 17 countries and 13 different states spanning 4 continents.

Table 5.9. Post-quantum cryptographic algorithm details [199]

| Post-Quantum Algorithm | \|pk\| [a] | \|sk\| [a] | \|c\| [a] | \|ss\| [a] | IND-CCA | NIST Level[b] | Foundation |
|---|---|---|---|---|---|---|---|
| FrodoKEM-640-AES | 9616 | 19872 | 9736 | 16 | Yes | AES128 | Lattice/LWE |
| FrodoKEM-640-cSHAKE | 9616 | 19872 | 9736 | 16 | Yes | AES128 | Lattice/LWE |
| FrodoKEM-976-AES | 15632 | 31272 | 15768 | 24 | Yes | AES 192 | Lattice/LWE |
| FrodoKEM-976-cSHAKE | 15632 | 31272 | 15768 | 24 | Yes | AES192 | Lattice/LWE |
| NewHope-512-CCA-KEM | 928 | 1888 | 1120 | 32 | Yes | AES128 | Lattice/RLWE |
| NewHope-1024-CCA-KEM | 1824 | 3680 | 2208 | 32 | Yes | AES256 | Lattice/RLWE |
| Kyber-512-CCA-KEM | 736 | 1632 | 800 | 32 | Yes | AES128 | Lattice/MLWE |
| Kyber-768-CCA-KEM | 1088 | 2400 | 1152 | 32 | Yes | AES192 | Lattice/MLWE |
| Kyber-1024-CCA-KEM | 1440 | 3168 | 1504 | 32 | Yes | AES256 | Lattice/MLWE |
| Sidh-p503 | 378 | 32 | 378 | 126 | No | AES128 | Isogeny |
| Sidh-p751 | 564 | 48 | 564 | 188 | No | AES192 | Isogeny |
| Sike-p503 | 378 | 434 | 402 | 16 | Yes | AES128 | Isogeny |
| Sike-p751 | 564 | 644 | 596 | 24 | Yes | AES192 | Isogeny |

[a] *|pk|=length of public key in bytes, |sk|=length of secret key in bytes, |c|=length of ciphertext in bytes, |ss|=length of shared secret in bytes*
[b] *Claimed NIST level derived from scaled ratings of 1-5 associated with submissions*

At the time of evaluation of possible paths to quantum resistance, a subset of the candidate PQC algorithms had been implemented as part of the Open Quantum Safe (OQS) project and were available in *liboqs* library of OQS algorithms [198]. The *liboqs* libraries have an interface oriented design in that each algorithm is accessible via a common key encapsulation mechanism (KEM) interface. However, the differences between sizes of public keys, secret keys, ciphertexts, and shared secrets can still complicate attempts to leverage common code in some cases. For instance, the post-quantum cryptographic algorithm details pertaining to the variants in *liboqs* during the research described in [199] appear in Table 5.9. The implementations themselves are in C, while wrappers were available in C#, C++, Go, and Python. A wrapper was also under development for Java.

The details of each of the proposed algorithms could fill dissertations and theses of their own, and are consequently beyond the scope of this text. However, a number of useful resources are available that can provide varying levels of detail including NISTIR 8240 [196], in which Section 3 gives a high-level overview of each of the second round

candidates. Meanwhile, [200] introduced the Learning with Errors (LWE) problem with associated cryptosystem and [201] surveyed the evolution of lattice based cryptography. The Frodo algorithm as implemented in OQS is an example that is based on the standard LWE problem and algebraically unstructured lattices [202]. Some other useful resources include [203] describing the NewHope algorithm based on the Ring LWE (RLWE) problem and power-of-two cyclomatic ring, [204] describing Kyber based on the Module LWE (MLWE) problem, as well as [205] and [206] describing Supersingular Isogeny Diffie-Hellman (SIDH) and Supersingular Isogeny Key Encapsulation (SIKE) respectively. Finally, Longa combined details of isogeny-based approaches with practical guidance for early adopters in [207].

### 5.6.2 The Hybrid Post-Quantum Horne-Nair Protocol (HPQHN) Protocol

The proof-of-concept implementation and the initial embodiment of the SecretMatch™ Prom app are based in part on the RSA algorithm. Unfortunately, the RSA algorithm is one of a number of cryptographic techniques that are weakened when confronted with adversaries empowered by quantum computers and Shor's algorithm. Additionally, the protocol significantly leverages one-way hash functions, which are also potential targets of attack for quantum enabled adversaries using Grover's algorithm. But in the coming era of quantum supremacy, the NIST PQC competition offers hope. Since design of a more quantum safe embodiment of the HN protocol occurred in parallel with early stages of the NIST PQC standardization process, and prior to extensive cryptanalytic scrutiny of the candidate algorithms, the more secure approach during the transition period was to develop a Hybrid Post-Quantum HN (HPQHN) protocol. The goal of the

HPQHN protocol was to maintain the same level of classical security afforded by the standard HN protocol as described in [1] and [113], while also affording quantum resistance by incorporating the strengths of PQC mechanisms. The essence of the approach taken to transform the HN protocol into the HPQHN variant was to combine PQC encapsulation of challenge and counter-challenge values with the notion of leveraging PQC based entropy as a required component for computation of the verifier $V_{AB}$.

*Preliminaries:* Let $\{c_i,ss_i\} = \Psi(pk_B)$ represent KEM encapsulation that, produces outputs of ciphertext $c_i$ and shared secret data $ss_i$ when provided input PQC public key $pk_B$ for user B. Let $\{ss_i\} = \Psi'(c_i,sk_B)$ represent decapsulation that reveals shared secret data $ss_i$ when provided ciphertext $c_i$ and secret key $sk_B$. Let $E_A(key,m)$ and $E_S(key,m)$ represent asymmetric and symmetric encryption of message $m$ with key $k$. Let $D_A(key,m)$ and $D_S(key,m)$ represent asymmetric and symmetric decryption of message $m$ with key $k$.

The HPQHN protocol can then be described by the following steps:

**1. Generate Quantum Resistant Challenge:**

a. *Alice selects user Bob* $\in U$, $G_i \in G \mid ID_B \in G_i$, *and her random data* $R_A$
b. *Alice computes* $X = E_A(PU_B, w + E_A(PR_A, R_A))$
c. *Alice computes* $\{c_{a1}...c_{an}, ss_{a1}...ss_{an}\} = \{\Psi_1(pk_B)...\Psi_n(pk_B)\}$
d. *Alice sets* $sk_{as} \in \{ss_{a1}...ss_{an}\}$, $iv_{as} \in \{ss_{a1}...ss_{an}\}$, $pqv_{as} \in \{ss_{a1}...ss_{an}\} \mid sk_{as} \neq iv_{as} \neq pqv_{as}$
e. *Alice computes* $X_{PQ} = E_S(sk_{as},X)$ *using* $iv_{as}$ *as needed*
f. *Alice sends* $\{G_i, X_{PQ}, c_{a1}...c_{an}\}$ *to M*

**2. Receive Quantum Resistant Challenge:**

a. *Bob anonymously queries M with* $G_i$ *and receives* $\{X_{PQ}, c_{a1}...c_{an}\}$
b. *Bob computes* $\{ss_{a1}...ss_{an}\} = \Psi'_1(c_{a1},sk_B)...\Psi'_n(c_{an},sk_B)$
c. *Bob sets* $sk_{as} \in \{ss_{a1}...ss_{an}\}$, $iv_{as} \in \{ss_{a1}...ss_{an}\}$, $pqv_{as} \in \{ss_{a1}...ss_{an}\} \mid sk_{as} \neq iv_{as} \neq pqv_{as}$
d. *Bob computes* $X = D_S(sk_{as}, X_{PQ})$ *using* $iv_{as}$ *as needed*
e. *Bob computes* $\{w+F\} = D_A(PR_B, X)$
f. *Bob chooses user U′ with whom he may share w*

**3. Generate Quantum Resistant Counter-Challenge:**

a. *Bob chooses his random data* $R_B$
b. *Assuming user U′ is Alice, Bob computes* $Y = E_A(PU_A, w + E_A(PR_B, R_B))$
c. *Bob computes* $\{c_{b1}...c_{bn}, ss_{b1}...ss_{bn}\} = \{\Psi_1(pk_A)...\Psi_n(pk_A)\}$
d. *Bob sets* $sk_{bs} \in \{ss_{b1}...ss_{bn}\}$, $iv_{bs} \in \{ss_{b1}...ss_{bn}\}$, $pqv_{bs} \in \{ss_{b1}...ss_{bn}\} \mid sk_{bs} \neq iv_{bs} \neq pqv_{bs}$
e. *Bob computes* $Y_{PQ} = E_S(sk_{bs}, Y)$ *using* $iv_{bs}$ *as needed*
f. *Bob sends* $\{G_i, X_{PQ}, Y_{PQ}, c_{b1}...c_{bn}\}$ *to M*

**4. Receive Quantum Resistant Counter-Challenge:**

a. *Alice anonymously queries M with* $X_{PQ}$ *and receives counter challenge* $\{Y_{PQ}, c_{b1}...c_{bn}\}$
b. *Alice computes* $\{ss_{b1}...ss_{bn}\} = \Psi'_1(c_{b1},sk_A)...\Psi'_n(c_{bn},sk_A)$
c. *Alice sets* $sk_{bs} \in \{ss_{b1}...ss_{bn}\}$, $iv_{bs} \in \{ss_{b1}...ss_{bn}\}$, $pqv_{bs} \in \{ss_{b1}...ss_{bn}\} \mid sk_{bs} \neq iv_{bs} \neq pqv_{bs}$
d. *Alice computes* $Y = D_S(sk_{bs}, Y_{PQ})$ *using* $iv_{bs}$ *as needed*

**5. Compute Quantum Resistant Verifier:**

a. *Alice:* $V_{AB} = H( H(R_A) + H( D_A(PU_B, D_A(PR_A, Y) ) ) + H(pqv_{as}) + H(pqv_{bs}) )$
b. *Bob:* $V_{AB} = H( H( D_A(PU_A, F) ) + H(R_B) + H(pqv_{as}) + H(pqv_{bs}) )$

It is important to distinguish between Alice's symmetric key $sk_{as}$ derived from a shared secret and Alice's personal PQC key pair $\{sk_A, pk_A\}$ with $sk_A$ known only to her, and public key $pk_A$. Bob has comparable sets of PQC keys and $pqv_{as}$ represents PQC verification bits that are used to compute $V_{AB}$. It is assumed PQC public keys are associated in some way with the PQC ciphertexts if different keys are used over time (e.g., when utilizing ephemeral keys). That association could be accomplished in multiple ways.

In the first step, Alice begins in a way that resembles the original embodiment of the HN protocol. She then uses her PQC KEM with Bob's PQC public key $pk_B$ to compute ciphertexts and shared secret data. Note that the algorithm details in Table 5.9 show that the PQC approaches vary in the number of bits of shared secret data associated with an

execution of the protocol. More than one key is needed, with the precise number depending on the amount of shared data needed for the desired level of security versus the details of the selected PQC candidate submission. Alice encapsulates the classical HN challenge inside a PQC envelope, and only Bob could open it to correctly reveal that challenge.

In the second step, the initial operation of step 2e would result in seemingly random data (or otherwise fail depending on details of the selected algorithm and its implementation) with very high probability to any user $ID_K \in G_i$, where $ID_K \neq ID_B$. Hence, the process would only continue past step two for Bob. In step 3b, inclusion of $\boldsymbol{w}$ is optional since $Y$ is associated with $X$, which included $\boldsymbol{w}$. This is another detail that is common across both HN and HPQHN protocol embodiments.

In the fourth step, Alice anonymously queries the matchmaker database with post-quantum hardened challenge $X_{PQ}$ and receives the post-quantum counter-challenge $Y_{PQ}$ along with ciphertexts $c_{b1}...c_{bn}$. These are comparable to Bob's actions interpreting the challenge in steps 2a and 2b, and reverse the actions of Bob's steps 3b through 3e. At this point, both Alice and Bob are ready to compute the verifier $V_{AB}$. Note that another key component of HPQHN is inclusion of PQC based entropy in the computation of the verification value that will match if and only if Alice and Bob are the ones inquiring and they do indeed share the same wishes.

An analysis of the HN protocol was previously completed describing the HN protocol for TPP in the context of the taxonomy of PETs from [20]. That analysis also resulted a proposal for the addition of a **Fairness** dimension with attributes of *Joint Notification* and *Equivalent Exchange* [4]. Figure 5.20 shows the results of analyzing HPQHN in

the context of the same PET taxonomy as detailed in [199]. HPQHN retains the requirements for fairness from HN, but expands with new attributes in the **Foundation** dimension. The hybrid post-quantum HN protocol adds attribute values representing the use of *Symmetric* and *Post-Quantum Cryptography (PQC)*.

| | | |
|---|---|---|
| **Aim** | ⇒ Indistinguishable, Unlinkable, Confidential, Deniable* | |
| **Aspect** | ⇒ Content, Behavior* | |
| **Aspect** | ⇒ Identity | ⇒ Anonymity ⇒ Direction=Multi |
| **Data** | ⇒ Stored, Transmitted, Processed | |
| **Fairness** | ⇒ Joint Notification, Equivalent Exchange | |
| **Foundat.** | ⇒ Security Model | ⇒ Computational |
| **Foundat.** | ⇒ Cryptography | ⇒ Asymmetric, Symmetric, PQC |
| **Revers.** | ⇒ Degree | ⇒ None |
| **Revers.** | ⇒ Cooperation | ⇒ N/A |
| **Scenario** | ⇒ Mutual | ⇒ Sender/Receiver |
| **Scenario** | ⇒ External | ⇒ Matchmaker/Interceptor |
| **TTP** | ⇒ Frequency | ⇒ Never |
| **TTP** | ⇒ Phase | ⇒ None |
| **TTP** | ⇒ Task | ⇒ None |

Figure 5.20. HPQHN analyzed within a taxonomy of PETs [199]

### 5.6.3 Impact to the SecretMatch™ System

The results of a complexity analysis of the HPQHN protocol appear in Table 5.10. When compared with the analysis of the original HN protocol, the primary impacts are in the way of computational costs. There are additional one-way hash function executions required, but they should only negligibly affect the overall runtime relative to cryptographic operations. The larger contributors to increased computational complexity, and thus to increased runtime requirements, are the newly added cryptographic operations

including the PQC encapsulations/decapsulations as well as the symmetric encryptions/decryptions. On the other hand, the overhead in terms of the number of messages required for an execution of the protocol remains unchanged, although the sizes of some messages will increase by a constant number of bytes. The complexity analysis suggested that the focus of additional testing should be the impact of the additional PQC and symmetric cryptographic operations on runtime performance in the challenge and counter challenge phases of HPQHN.

Table 5.10. HPQHN complexity analysis [199]

| Operation | Alice | Bob | Matchmaker |
|---|---|---|---|
| Asymmetric Encryptions | 2 | 2 | 0 |
| Asymmetric Decryptions | 2 | 2 | 0 |
| String Concatenations | 2 | 2 | 0 |
| One-way Hashes | 5 | 5 | 0 |
| Post-Quantum Encapsulation | 1 to N | 1 to N | 0 |
| Post-Quantum Decapsulation | 1 to N | 1 to N | 0 |
| Symmetric Encryptions | 1 | 1 | 0 |
| Symmetric Decryptions | 1 | 1 | 0 |
| Number of Messages | $c + \frac{log(1-\lambda)}{log(0.5)}$ | $c + \frac{log(1-\lambda)}{log(0.5)}$ | $2 \times (c + \frac{\log(1-\lambda)}{\log(0.5)})$ |

The interface-oriented design of the SecretMatch™ Prom client application will help to limit the new development and subsequent rework required when extending support to the HPQHN protocol. Figure 5.21 reflects an architectural overview of the SecretMatch™ system with the components primarily affected shaded in green, and the components secondarily affected shaded in blue. Beyond extending the matchmaking protocol implementation from HN to HPQHN, the cryptographic interface must be extended to support PQC and symmetric operations. Less significant changes are also expected in the Communications API to support additional message parameters and/or larger challenge/counter-challenge values, and to the key management API to support PQC keys and potentially symmetric keys depending on the desired persistence level for those keys. In the Matchmaker (server) itself, the main changes would be to extend or modify database schemas to store larger values and additional fields. There would also be changes to the REST API constituting the matchmaking service to external entities that resemble or mirror the modifications to the Communications interface of the client app.

### 5.6.4 Implementation and Test Results for HPQHN Proof-of-Concept

The performance testing began with the original RSA-based test program used to evaluate the HN protocol. Since the 32-bit system included with the original testing was no longer available due to hardware failure, the program was compiled targeting 64-bit architectures with a simple modification of compiler options. The next steps involved building the libraries from the Open Quantum Safe (OQS) project [198] and integrating the libraries into the test project. The challenge and counter-challenge phases were then augmented to reflect the operations of HPQHN utilizing the post-quantum encapsulations

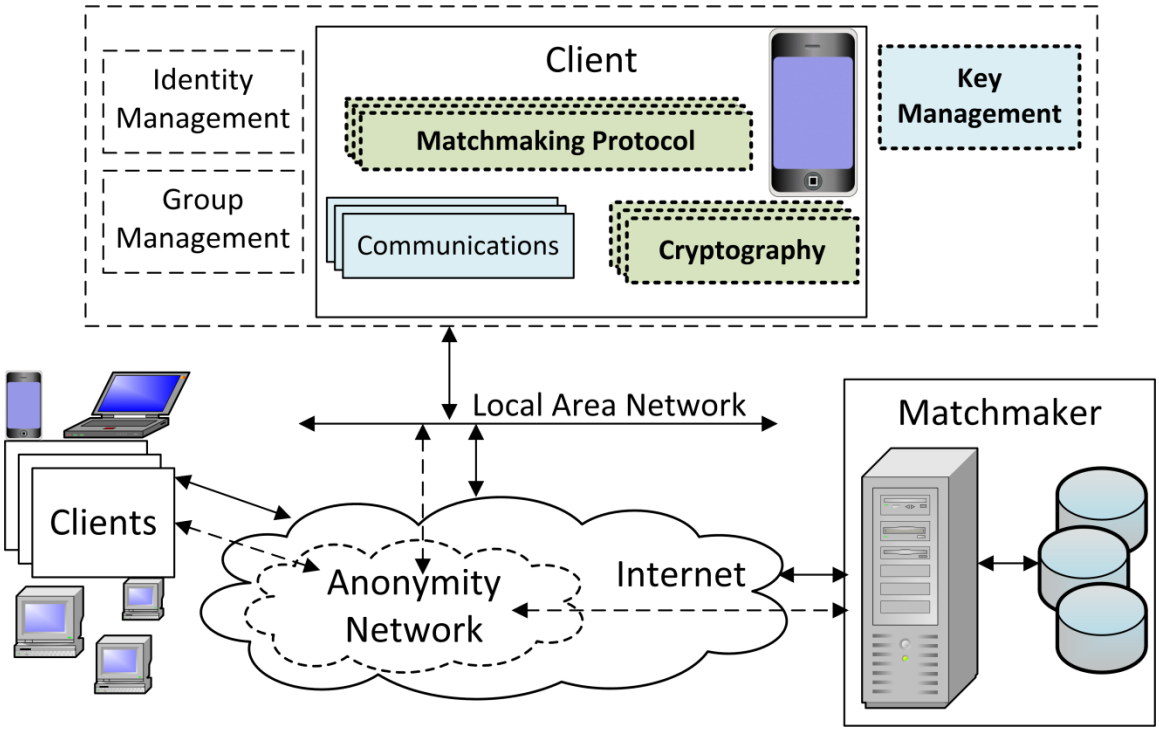



Figure 5.21. Overview of the SecretMatch™ system with primarily/secondarily affected areas shaded green/blue [199]

and decapsulations with Kyber CCA and NewHope CCA KEMs, along with AES256 for the symmetric encryption operations of the hybrid protocol design. The test systems included Qualcomm® Snap-dragon™, Intel® Pentium®, AMD Athlon™, and Intel® Core™ i5 and i7 central processing units representing a spectrum of use cases from mobile devices to budget laptops and high performance desktops.

The results of the performance testing originally described in [199] appear in Table 5.11. The most significant difference observed was actually a performance gain of 40% - 48% with no code changes by simply producing a 64-bit optimized build of the original test application, as is apparent from the first two result columns. Meanwhile, the increase

in runtime from the additional PQC and symmetric encryption operations in the HPQHN protocol was almost negligible, adding only about 0.5% to 2.0% to overall runtimes compared to the classical HN protocol operations. If these proposed post-quantum KEMs stand up to rigorous cryptanalysis, the results suggest that HPQHN is fit for practical use, and that security against quantum-enabled adversaries may be relatively affordable despite certain major advantages of said adversaries in the post-quantum era.

Table 5.11. Results of HPQHN performance testing [199]

| CPU[a] | Op[b] | Op-Opt[b] | Op-KYBER[b] | Op-NEWHOPE[b] |
|---|---|---|---|---|
| Snapdragon | 265.997 | N/A | 268.471 | 269.591 |
| Pentium | 111.925 | 57.607 | 58.066 | 58.024 |
| AMD M320 | 100.034 | 56.070 | 56.609 | 56.315 |
| Intel Core i5 | 56.285 | 33.305 | 35.205 | 33.967 |
| Intel Core i7 | 44.061 | 23.861 | 24.872 | 24.031 |

[a]*Qualcomm® Snap-dragon™ 835 2.2 GHz; Intel® Pentium® N350 2.16 GHz; AMD Athlon™ M320 2.1 GHz;* Intel® Core™ i5 5200U 2.2 GHz*;* Intel® Core™ i7 4770 3.4 GHz

[b]*Op-* represents average of challenge and counter challenge phases in milliseconds (ms). Op=Original, Op-Opt=Optimized, Op-Algorithm=Optimized PQC Hybrid using Kyber/New Hope algorithms*

The source code for the HPQHN test application is described in Appendix G. The results of performance testing demonstrated that integrating the HPQHN protocol with the SecretMatch™ Prom app should not significantly impact overall runtime in such a way as

to negatively impact usability. An HPQHN based SecretMatch™ technology would retain the classical strength of the HN protocol, while also adding security against quantum-enabled adversaries. During the ongoing NIST PQC standardization process, it would not be advisable to place 100% of security in the hands of PQC algorithms that have not yet faced the rigorous cryptanalytic scrutiny of the full "competition". Indeed, many of the initially proposed algorithms were weakened or defeated within days or weeks [197]. If quantum-enabled adversaries became commonplace, the PQC envelopes and augmented verification values that include PQC based entropy afford additional protections. However, if the PQC algorithms being used were broken during standardization or shortly thereafter, the design should still be as strong as the original HN protocol.

System designers and developers must give increasing attention to quantum enabled adversaries to provide security and privacy going forward. The hybrid post-quantum Horne-Nair (HPQHN) protocol design represents a key step toward quantum resistance for the SecretMatch™ PET during the transitional PQC standardization process, as the availability of quantum computers with growing numbers of qubits continues to increase. The next steps toward the HPQHN enabled SecretMatch™ Prom app, as well as fully Post Quantum HN (PQHN) and Quantum Enabled HN (QEHN) protocols, are discussed further in the future work subsections of Chapter 6.

# CHAPTER 6

# CONCLUSIONS AND FUTURE WORK

To put TPP and related research into perspective, it is useful to consider the privacy risks of modern matchmaking systems. For instance, Tinder is a popular matchmaking service that reportedly had tens of millions of users, made 15 million matches per day in 2014, and was projected to earn $75 million in 2015, less than three years after it launched [208] [209]. Fast forward to 2019, and Tinder brought in $1.2 billion in revenue, over half of parent company Match's total of $2.1 billion for the same year, which also includes Match.com, Hinge, and OkCupid [210] [211]. But the millions of subscribers also surrender significant privacy relevant information to such matchmaking services. In a recent article, an author describes being horrified after she requested a copy of her personal information that Tinder maintained in accordance with European privacy laws, when she subsequently received 800 pages of her self-described "deepest, darkest secrets" [212]. As was learned from the Ashley Madison data breach of 2015 [5], which allegedly led to blackmail, espionage, ruined careers, and even suicide, the ramifications of a privacy invasion involving Tinder or similar matchmaking services could be enormous.

The broad popularity of such electronic matchmaking systems demonstrates that the early work of Baldwin and Gramlich on trustable matchmaking showed great foresight. Modern matchmaking challenges resembling TPP and involving ILW necessitate a novel

approach to protect user privacy. The present system for privacy-enhanced and fair matchmaking endeavors to fulfill that need. Embodying the spirit of privacy by design (PBD), the HN and HPQHN protocols keep privacy relevant data in control of the users to which the data pertain, and avoid sharing private matching data with others, including third party services. The remainder of this chapter summarizes conclusions drawn, highlights of contributions of this research, and gives an overview of the TPP related research roadmap going forward.

## 6.1 Concluding Remarks

The various phases of TPP related research described herein have been documented in a number of peer-reviewed publications including four conference papers, a journal paper, and provisional and final United States patent applications resulting in a grant of patent exclusivity for the invention.

Related publications contributing to this body of research to date include:

1. *The Prom Problem: Fair and privacy enhanced matchmaking with identify linked wishes* – IEEE ICCST, 2016 [1]
2. *Method and system for privacy preserving disclosure of a shared, identity linked secret*," U.S. Provisional Patent Application, 2016 [213]
3. "*A feasibility analysis of fair and privacy-enhanced matchmaking with identity linked wishes*," ACISP, 2017 [3]
4. "*Privacy-enhanced and fair matchmaking system – Applications and analysis of protocol, architecture, and performance*," SAM, 2017 [4]

5. *Method and system for privacy preserving disclosure of a shared, identity linked secret*," Final Application, 2017 & U.S. Patent 10,432,400, 2019 [101]
6. "*A new privacy-enhanced technology for fair matchmaking with identity linked wishes*," IEEE ISJ, 2019 (Early Access) & 2020 [113]
7. "*Toward a quantum resistant SecretMatch™ privacy enhanced technology – a case study*," SAM, 2019 [199]

Some of the key contributions from those publications and the remainder of the body of work described herein include:

- Formulated The Prom Problem (TPP) and security requirements for fair and privacy-enhanced matchmaking with identity linked wishes (ILW) and contrasted it with a thorough review of prior work
- Expounded upon the adversary model of TPP and identified a number of practical attack threats that must be defended against
- Proposed the Horne-Nair (HN) protocol for fair and privacy-enhanced matchmaking along with a security analysis and proof-of-concept implementation of one embodiment
- Performed analysis of the HN protocol for TPP in the context of a taxonomy of PETs to better characterize its place within the spectrum of privacy enhancing technologies
- Quantified fairness of individual executions of the protocol via a formula for the fairness index
- Presented a complexity analysis of the HN protocol and experimental test results analyzing computational and communication overhead that demonstrated

the feasibility of the protocol for real-world use, even with enhanced anonymity such as VPNs, Onion routing, or both techniques

- Identified a number of potential application contexts for the HN protocol that can benefit from fair and privacy enhanced matchmaking with identity linked wishes such as voting negotiations in legislative bodies, recruiting of high level executives, or corporate mergers and acquisitions
- Provided an overall matchmaking system design leveraging an embodiment of the HN protocol
- Provided perspective on the contributions of PETs like SecretMatch™ in a landscape of rapidly evolving privacy regulations, giving a number of examples and lessons learned related to privacy by design, cost savings with regulatory compliance, and viewing privacy as a potential competitive advantage
- Developed the HN protocol enabled SecretMatch™ Prom app and server-side matchmaker following sound engineering principles for technology selection, and guiding design decisions with privacy by design principles
- Began work toward a quantum resistant embodiment of the SecretMatch™ system with design of the Hybrid Post-Quantum Horne-Nair (HPQHN) private matchmaking protocol
- Provided updated complexity analysis and updated PET taxonomy analysis for HPQHN
- Extended original HN test application to HPQHN and demonstrated that the additional computational requirements were relatively small, resulting in a practical approach suitable for use in a Hybrid Post-Quantum SecretMatch™

### 6.1.1 Privacy Engineering in Modern Product Development

As highlighted in [113], those developing software related products and services must pay more attention than ever to good privacy engineering practices or potentially face significant regulatory challenges, financial penalties, and reputational damage. But many years prior to landmark privacy-focused legislation such as the General Data Protection Regulation (GDPR) of the European Union (EU) [214] or the California Consumer Privacy Act (CCPA) [215], Cavoukian coined the term *privacy by design* (PbD) and provided a vision of a world in which companies emphasized privacy assurances as a top priority [133]. The foundational principles of PbD put forth can be summarized as follows:

1. Be proactive, not reactive. Focus on prevention rather than remediation.
2. Privacy should be the default.
3. Privacy should be embedded into requirements and design rather than retroactively considered after development.
4. Strive for a positive sum rather than zero sum. Strive for a "win-win" that assures privacy while also achieving the desired outcome.
5. Full lifecycle security is critical to assuring privacy.
6. Seek openness, visibility, and transparency for all stakeholders.
7. Respect user privacy and prioritize users' interests.

Of course, the line of thinking behind PbD pre-dated this specific enumeration of principles. But this clear and concise articulation of PbD can be a beneficial guide to practitioners. In fact, article 25 of the GDPR is even titled "Data protection by design and by default", clearly reflecting the essence of PbD.

To better understand the potential role of PETs such as SecretMatch™ in the modern era of rapidly evolving privacy legislation, the following important elements of the GDPR were summarized in [113].

- **Scope** – The GDPR applies to all citizens of the EU, irrespective of their physical location. The GDPR has derogations for organizations with fewer than 250 employees as well as certain exceptions for areas such as scientific research, national security, and law enforcement.
- **Personal Data** – The data protection principles of the GDPR apply to identifiable or identified natural persons, and even pseudonimized data may be considered personally identifiable. Beyond personally identifiable information (PII), protections can extend to other privacy relevant data such as genetic data, protected health information, or behavioral data.
- **Data Minimization** – The concepts of data minimization and temporal limits on data retention are important concepts addressed early in the legislation.
- **Penalties** – Upper limits on financial penalties are defined as the larger amount of either 4% of total annual turnover or 20,000,000 EUR (over $22 million USD at the time of this writing).
- **Fair Processing** – Fair and lawful processing requires clear identification of what data are collected and how the data are processed and used. Lawful processing further requires that data collection and use has an "explicit and legitimate" purpose.
- **Informed Consent** – Fair and lawful processing also requires informed and explicit consent. Data subjects must be informed of, and consent to, data col-

lection, usage, and retention policies. Data subjects must also be informed of their rights with regard to said data, and those rights can be exercised.

- **Data Subject Rights** – The required data subject rights include rights to portability (e.g., to request and receive a copy of all personal data), erasure (the "right to be forgotten"), to withdraw consent, and the right to have errors corrected.
- **Security** – Adequate technical controls must be used to ensure the confidentiality and security of data during processing and storage. Timely disposal is also required.
- **Data Breach Notification** – Data breaches involving privacy relevant data must be reported to data protection authorities within 72 hours of discovery.

The GDPR took effect on 25 May 2018, and its regulatory framework has resulted in significant financial penalties for companies that were determined to be non-compliant. For example, a few of the administrative fines and penalties that have been imposed include the following instances, while a more complete list may be found at [216].

- **Google Inc. – 50,000,000 EUR, 21 January 2019** - Fine imposed by French Data Protection Authority for insufficient legal basis for data processing
- **Austrian Post – 18,000,000 EUR, 23 October 2019** – Fine imposed by Austrian Data Protection Authority for insufficient legal basis for data processing
- **Deutsche Wohnen SE – 14,500,000 EUR, 30 October 2019** – Fine imposed by Data Protection Authority of Berlin (Germany) for non-compliance with GDPR data processing principles

- **1&1 Telecom GmbH – 9,550,000 EUR, 09 December 2019** – Fine imposed by Federal Commissioner for Data Protection of Information (BfDI) (Germany) for insufficient technical and organizational measures to ensure information security
- **Google LLC – 7,000,000 EUR, 11 March 2020** – Fine imposed by Data Protection Authority of Sweden for insufficient fulfillment of data subjects rights.
- **National Revenue Agency – 2,600,000 EUR, 28 August 2019** – Fine imposed by Data Protection Commission of Bulgaria (KZLD) for insufficient technical and organizational measures to enforce information security.

In the United States, the state of California recently enacted the California Consumer Protection Act (CCPA) [215] and thereby brought a new level of privacy protections to residents of that state. Given the difficulty for entities in cyberspace to apply compliance based policies only to citizens of one state, many more citizens of the United States are likely to benefit from improved privacy related policies. The CCPA resembles the GDPR in some areas such as having a notion of data subject rights that includes a right to access personal data and a right to be forgotten. However, the regulations differ in other areas such as scope and consent. The CCPA applies to entities with gross revenues of at least $25 million USD, entities that derive 50% or more of revenue from sales of consumer information, or that processes personal information of 50,000 or more consumers, households, or devices. A more stark contrast appears when considering the notion of consent. While the GDPR requires informed consent for lawful processing (i.e., “opt in”), the CCPA instead requires the ability to “opt out” of data collection and processing policies.

The potential financial penalties for non-compliance with privacy regulations like the GDPR clearly warrant significant attention to consumer privacy. Although many business leaders may view such privacy regulations as costly and burdensome, a focus on good privacy engineering practices such as privacy by design and privacy by default could significantly limit risk and minimize additional efforts required for regulatory compliance as privacy law continues to evolve in coming years. Moreover, business leaders that embrace data privacy may even find that it can lead to competitive advantage in a society that will continue to be increasingly aware of privacy related risks and deficiencies based in part on the proliferation of breach notification laws and financial penalties for non-compliance with data privacy law.

### 6.1.2 Lessons Learned and Recommendations for Privacy Engineering

Through the process of developing a new PET for fair and privacy-enhanced matchmaking with identity linked wishes, a number of lessons learned, observations, and recommendations may benefit future privacy engineering endeavors in a climate of heightened awareness and evolving legislation. Some of the lessons learned and recommendations as originally reported in [113] include the following:

**Minimize or eliminate PII and privacy-relevant data.** Akin to the agile development principle of maximizing the amount of work not done [217], a corresponding privacy engineering manifesto principle might be to maximize the amount of privacy-relevant data that is not collected, stored, or processed. Many regulatory hurdles and financial risks can be avoided entirely with elimination of processing and storage of privacy-relevant data. A fair and private matchmaking process involving ILW unavoidably in-

volves sensitive and privacy-relevant data. But the HN protocol was designed to accomplish the goal while avoiding the transmission, storage, or processing of privacy-relevant data to or by any entity outside of the client device. The implementation also uses group identifiers for improved efficiency. But the privacy-focused decision was made to avoid the need for a central repository of grouping information by generating group identifiers via common mechanism within the client device. Moreover, rather than employing centralized storage and retrieval of event information, a similar decision was made to uniquely identify individual events without the need for a central event management entity. By intentionally emphasizing PbD, the ultimate system design eliminated storage or processing of privacy-relevant data, resulting in significant potential savings in regulatory costs and possible penalties for non-compliance.

**Limit risk with prototyping, proof-of-concept, and performance testing when applicable.** The costs of design and development of a complex system can be quite high. Consequently, failed software projects are a major concern. Estimates of the percentage of failed projects and contributing factors have been widely studied (e.g., [218], [219], and [220]). Projects can be deemed a failure for many reasons such as encountering technical obstacles or significantly exceeding budgetary constraints. In the case of TPP and the HN protocol, key technical risks were mitigated via proof-of-concept testing to give assurance of correctness [1]. Then, technical risks were further mitigated with performance testing and feasibility evaluation prior to incurring the full costs of development [3]. Finally, rapid prototyping of user interfaces and mockups yielded opportunities for early user feedback, fostering the right balance of tradeoffs between usability, security, and privacy.

**A PET is more than just a privacy-preserving protocol.** The correctness of a privacy-preserving protocol is foundational to success of a PET. But threat analysis must also be conducted at the system level because the attack surface extends far beyond the protocol itself. Side channel attacks, mishandling of sensitive data in memory or storage, misconfiguration of security relevant settings, or implementation weaknesses such as buffer overflow and cross-site-scripting vulnerabilities can result in the defeat of even the most secure protocols. Additional points of weakness can include selection of encryption algorithms, hash algorithms, encryption key management, or insecure random number generation. Security and privacy engineering does not stop after design of a secure protocol, but a system level perspective must be used to evaluate security and privacy throughout the product development lifecycle.

**Design for flexibility – the only guarantee is change.** Given the rapid pace of technological advancement, seek to design protocols in an algorithm agnostic way to the extent possible. Similarly, design software and APIs to cost-effectively accommodate change. For example, the REST APIs for communications were designed with versioning in mind to accommodate change. Similarly, the SecretMatch™ Prom app uses an interface oriented design to facilitate efficient adoption of new cryptographic algorithms, alternate key management services or data storage mechanisms, and even an alternative matchmaking implementation as future embodiments of the HN protocol evolve.

**Evaluate risks of software dependencies and plan for security testing, regular maintenance, and updates.** The English scholar and poet John Donne famously wrote that *no man is an island*. If Donne were a modern software engineer, he might instead say that *no source code is an island* or that *no software product is an island*. Modern

software is built on top of complex technology stacks using programming language libraries, open source components, commercial technologies, and third party services as well as internal and external APIs. In 1984, Ken Thompson famously reflected on the fundamental nature of trust at the most basic level, identifying the risk of a seemingly unavoidable, self-propagating backdoor [161]. Consider how much greater the challenge of ensuring trust with the vast attack surface of complex modern software systems must be. From the beginning, it is critical to vet dependencies, to incorporate security testing such as security-focused static code analysis and vulnerability scanning, and to plan for regular maintenance and updates to potentially vulnerable elements throughout the software stack.

**Maintain awareness of evolving privacy regulations as a contributor to risk management and cost-benefit analysis.** Since the GDPR was drafted in 2016, businesses have had to adapt as they experienced a rapid evolution of the landscape of privacy regulations. It is important that both current and projected legislation be considered during risk management activities. Cost-benefit equations can change significantly with the changing regulatory landscape, just as cost, likelihood, and impact vary with evolving security and privacy threats.

## 6.2 Future Work

A number of opportunities for future research follow from this body of exploring the challenge of privacy-enhanced and fair matchmaking with identity linked wishes (ILW). Some of those opportunities can be generally grouped into three categories including enhancements to the SecretMatch™ PET, continued study of HPQHN and quantum re-

sistance, and topics further exploring privacy-enhanced technologies given the rapidly evolving landscape of privacy related legislation.

### 6.2.1 Enhancements to the SecretMatch™ Privacy-Enhanced Technology

The roadmap of future work begins with final steps to ready the app for app stores. First, an evaluation of the cost-benefit of the Facebook Login feature will result in the decision of whether or not to release that feature with v1.0 of the SecretMatch™ Prom app. The app allows selection of any contact from the user's contact list on the applicable device. As originally envisioned, the Facebook Login feature would afford access to all Facebook friends, and in a future version, even allow matching with Facebook users that the SecretMatch™ Prom user was not yet connected with via Friend lists. However, with the evolving Facebook Platform Policy [221] and Facebook's recent emphasis on privacy [222], access to information about a user's friends is now significantly restricted. While this is a win for user privacy, it introduces challenges and limitations for developers with legitimate needs for the data. Unfortunately, the widespread misuse of personal data for a few has resulted in roadblocks for all.

The first hurdle is to request the permissions and go through the review process. This involves identification of all required permissions (e.g., `user_friends` or `user_photos`), submission of descriptions of every way in which the permissions are used, and submission of video walk-throughs demonstrating all of the ways in which the permissions are used [223]. Another challenge is that testing with the true Facebook API using these permissions cannot be accomplished without going through the approval process. But the approval process itself requires video walkthroughs demonstrating use of

the permissions. This circular dependency can be overcome using mock data, but it introduces additional friction nonetheless. Beyond deprecation of the permissions to get information about people's friends, with v2.0 of Facebook's Graph API, the utility of the functionality that is available is limited to accessing data of other friends that both have the same app installed and have also granted it the `user_friends` permission [224]. The fact that the `read-friendlists` permission only allows the app access to user-defined custom friend lists that are manually created by the Facebook user adds still more friction, exhibiting the epitome of a tradeoff between usability and privacy.

One potential mechanism to improve the situation might be to add use of Facebook's Invitable Friends API to invite friends to use the app. However, for a private matchmaking app with the goal of anonymity in the absence of confirmed ILW, the feature could be a privacy risk. Additionally, continuing down that path would negatively impact time-to-market, an important consideration for most all new technologies. Consequently, although the Facebook Login is present and functional in the app implementation, the cost-benefit equation surrounding addition of retrieval of friend lists to potentially allow access to more contacts for matching may have swung like a pendulum toward the costs and risks far outweighing the benefits of retaining the feature.

After the final decision and steps regarding the Facebook integration, and additional stages of testing and refinement, the first version of the app will be the Android version in the Google Play store. One last evaluation of the enhancements discussed in Section 6.2.2 or the additional system enhancements described as future work will be conducted to consider whether one or more should be included with SecretMatch™ Prom v1.0. For example, the foundation for API key enforcement was included with the initial imple-

mentation of the server-side matchmaker to help limit risk of certain brute force attacks like flooding the system with false challenges for denial of service, or generally flooding with excessive challenges or counter-challenges with malicious intent. Beyond the use of API keys, additional steps may be considered to address possible misuse cases.

For instance, consider a case in which Bob receives a challenge for his high school prom. He might theoretically try to submit counter-challenges for every female in his school, attempting to circumvent the intended privacy-protections. The initial implementation mitigates this risk by allowing only a single counter-challenge associated with a given challenge. In the future, this could be expanded to a small constant number greater than one while still adequately protecting privacy. Potential alternative solutions to this and similar attempts at brute force attacks may also be explored such as associating a cost with each submission, which could double as a monetization strategy. Other possible options include use of a mediated escrow system, the use of policy-based enforcement where the threat of losing access to the system permanently might deter misuse, or even a combination of these approaches. Each of these techniques has both advantages and disadvantages that should be considered. For instance, the use of an escrow based system might re-introduce a requirement for some level of trust in a third party, yet a key goal of this research was to avoid the need for trust in a third party to the fullest extent possible. The ultimate selection of the appropriate enhancements to mitigate additional brute force attacks will require a careful evaluation of tradeoffs.

After any potential modifications, and when final testing is complete, the next step will be ensuring that the APK file is within the size limit for the targeted Android ver-

sion. If not, it will need to be split into parts. Certain highlights of the remaining steps involved to publish the app to the Google Play store include:

- Create Google developer account and accept the Developer Agreement
- Review distribution countries, enter payment card information, and complete the registration process
- Create the SecretMatch™ Prom app and enter required or desired information such as:
    - App name, description, and extended description
    - High resolution icon
    - Promotional feature graphics
    - Screenshots in accordance with screenshot specifications
    - App type and content rating information
    - Privacy policy URL
    - Pricing and distribution information

Additionally, the app will need to either use Google Play app signing or be locally signed. Upon uploading the app, it will need to go through the Google Play app review process before publishing. Prior to publication and availability to all users, the current plan is to first have a limited release for beta testing purposes with up to 100 testers [225]. The testing phase will both provide early feedback about the client app, and validate readiness of the server-side matchmaker for more broad adoption.

Following wide-spread availability of the SecretMatch™ Prom Android app in the Google Play store, attention will turn to completion and publication of the Apple iOS and Windows apps. While the large majority of the app is common code, a few platform spe-

cific features will need to be finalized for Windows and iOS. The process for the Windows app will be relatively straightforward, not requiring additional hardware or software beyond the development environment. Both the Android and Windows apps can be built and tested on a standard Windows PC. However, creation of the iOS app is a bit more involved.

Development for iOS can occur on an Apple device running macOS and Visual Studio for Mac, or on a Windows computer if there is a Mac computer accessible on the network for remote compilation and debugging. Another option is potentially running a Windows VM on a Mac computer. The Apple device must also have Xcode installed. For more specific hardware requirements, refer to [226]. The Apple App Store publication and review process that will be followed is described at [227]. It will involve setup of App ID and entitlements, providing app details such as icon and description, setup of App Store provisioning profile, updating build configurations, configuring the SecretMatch™ Prom app in iTunes Connect, and submitting to Apple for review and publishing.

Apart from the beta testing and final publication of the Android, iOS, and Windows apps in various app stores, a number of different avenues of future research have been identified spanning specializations in computer science from user experience and databases to quantum computing and the intersection of technology and privacy law. The roadmap for future work includes numerous possibilities, some of which are now further detailed in the following subsections.

#### 6.2.1.1 User Experience Testing and Alternative Interface Techniques

The design of the user interface for the SecretMatch™ Prom app was guided by established user interface guidelines and heuristics as described in Section 5.4.3. Next steps in the human computer interfaces (HCI) and user experience (UX) domains include follow-on experimentation to empirically assess the design of the user interface and to potentially assess the validity of the design guidelines in this context. Depending on the outcome, further study may also evaluate alternatives to the current user interface with empirically based UX testing with specific usability goals and metrics akin to the process described in [171]. Over time, if the SecretMatch™ Prom app becomes widely adopted and well-funded, a similar UX evaluation process may be incorporated into the standard development process for future releases of the system.

A number of studies have concentrated on evaluation of user interfaces in specific contexts such as for people with Parkinson's disease [228], people with Alzheimer's disease [229], or for children [230] [231]. The applications of fair and privacy-enhanced matchmaking with ILW beyond dating and relationship may offer new contexts in which to similarly re-evaluate traditional design guidelines and produce practical guidance for optimizing interfaces with certain groups of the population based on attribute, for employees in certain job roles, and more. Consideration of interfaces beyond mouse and touchscreen may also give opportunities to evaluate proposed guidelines such as those for hands-free speech interaction [232] [233] or gesture based interaction [234]. However, it also provides an interesting challenge when considering a possible speech interface with such privacy relevant data as that used for matchmaking. In fact, a number of topics evaluating privacy and the human-computer interface may present themselves.

Preliminary work with a brain-computer interface (BCI) and the use of brain wave patterns as personally unique identifiers in parallel with the current research has shown promise for real-time authentication of users. That preliminary work resembles the work of Kanaga et al. [235] in that it leveraged Electroencephalography (EEG) for non-invasive authentication. But the next step was to accomplish real-time, ongoing authentication to add an extra layer of security and counter certain threats. Some SecretMatch™ apps may provide good opportunities for further study evaluating the strengthening of security and privacy via such a mechanism in the context of real-world applications. Beyond real-time authentication, it may also provide opportunities for novel BCIs to perform private matchmaking in particular contexts, thereby potentially improving privacy over voice interfaces that can be eavesdropped on or touch/gesture interfaces that can be spied upon. Finally, as artificial intelligence and machine learning (AI/ML) continue to become more pervasive, AI/ML features or interfaces may be integrated with a future version of SecretMatch™ technologies. If so, that may lead to opportunities to build upon recent human-AI interface research such as [236] or [237].

#### 6.2.1.2 Additional SecretMatch™ Prom System Enhancements

A number of opportunities for enhancement to the SecretMatch™ Prom app have been identified. For example, one avenue for exploration will be to attempt to design some type of self-destructing mechanism to enforce temporal constraints. The initial system design makes use of temporal constraints with the validity of identity linked wishes expiring after a user specified period of time. However, enforcement currently relies on a bit of trust in the Matchmaker. Luckily, the process of purging the database of expired

challenges actually benefits the Matchmaker as it helps to prevent the database from growing to an infeasible size, while at the same time limiting data storage costs that could otherwise render the system infeasible from a financial standpoint. An alternative implementation could place the expiration checking logic in the client app, but that would only shift the requirement of trust from the server to the client implementation. Instead, a mechanism whereby challenge values self-destruct and become useless for matchmaking, or to recover any useful information whatsoever, would further enhance the value proposition of the system from a privacy perspective by further reducing reliance on trust at any point in the matchmaking lifecycle.

Other enhancement opportunities present themselves when considering the age old challenges of secure key distribution and public key infrastructure. As with many systems, the secure distribution and storage of cryptographic keys can be a fundamental weakness if not handled correctly. In the initial system, a user must register her self-generated public key with the key server (although this will be handled by the app and technically "invisible" to the user). The device information is used as the primary authenticity verifier. However, addition of a further verification step would be desirable from a security standpoint. In particular, if the public key is associated with an email address, a challenge-response protocol to prove the submitter really has access to the specified email address would improve the validation of the public key. Similarly, if a mobile phone number were associated with the public key, a text message or phone call could be used for the challenge-response validation process. Other approaches are also possible. An additional layer of validation such as this is a top enhancement priority. Beyond the initial authentication and validation step, a mechanism for periodic re-

validation may later be considered to mitigate threats such as phone numbers or email addresses lost or no longer used. A novel approach to periodic re-validation that properly balances security with usability could also have benefits far beyond SecretMatch™ apps. Of course this may be specific to the Prom app, as SecretMatch™ apps for corporate mergers and acquisitions or voting negotiations in legislative bodies might likely utilize the trusted and established authentication systems and public key infrastructures of the businesses and governments involved respectively.

Finally, in some instances, employing the notion of fake or decoy messages might add additional security or privacy to the system. For example, consider the case in which Alice submits a challenge intended for Bob with ILW roughly equivalent to "Alice + Bob + Southlake High School Prom 2020". If Alice later finds that there is a corresponding counter-challenge, in the absence of fake/decoy messages, Alice could infer that it is likely that Bob wants to attend the prom with someone. Fortunately, the statement that "FILL-IN-THE-NAME wants to go to the prom with ***someone***" would essentially be universally true within the high school population in the prom context, so no new information is really learned. However, in other SecretMatch™ apps for different contexts, it will be important to reconsider this case to ensure no useful information could be deduced. But in the case that it could become a weakness in certain contexts, then a solution might be to utilize fake messages as a mitigation technique.

In the system design that does not include the generic wishes (e.g., "attend Southlake 2020 prom together") in the counter-challenge, the use of decoy messages may be straightforward since the contents of the decrypted outer envelope are also just encrypted random data in essence. However, if an implementation included the generic wishes in

the counter-challenge, either in human readable form or direct mapping to human readable form, then it may instead be a good scenario to explore employing the notion of Honey Encryption (HE) [238] [239]. In HE, much like a Honeypot, valid looking objects can serve to confuse, distract, or trap adversaries. After the initial HE proposals, others adapted the idea to a scheme for encoding and decoding of human readable message content for appropriate length human language decoy messages [240]. Consequently, future work will evaluate further adaption of the notion of HE in the fair and privacy-enhanced matchmaking context to offer additional security against potential inference attacks via HE with decoy messages revealing natural language or otherwise legitimate looking fake responses.

#### 6.2.1.3 Client Applications Beyond the Prom

A number of other problems have been identified beyond the context of dating and relationships that require fair and privacy-enhanced matchmaking with ILW as reported in [4]. The research that led development of the HN protocol and the SecretMatch™ Prom app can be readily adapted for those other problems. Some of the planned future applications of the SecretMatch™ PET include the following:

- **Recruiting of High Level Corporate Executives** – The original BG protocol introduced trustable matchmaking in the context of executive recruiting [2]. A privacy-enhanced solution to TPP offers further benefits to recruiting efforts involving high level executives. Consider the case where Alice, as Vice President of her company, may be open to new opportunities, but she does not want her employer to know that. Company B, that is not her current employer, may be interested in

hiring Alice for their CEO position, but they do not want to advertise that fact either. In this case, a fair and privacy-enhanced matchmaking system like SecretMatch™ could facilitate this transaction involving ILW with executive recruiting.

- **Voting Negotiations in Legislative Bodies** – In legislative bodies around the world, members of different parties or ideological groupings may be hesitant to break from expected norms or even to suggest such a possibility. For example, consider a divisive piece of legislation that Alice's party is expected to vote against. Alice may want to vote for the legislation against expectations, but unwilling to do so and risk potential career damage unless Bob, another member of her party, is also willing to vote to affirm the legislation. But she cannot approach Bob directly because she does not even want him to know that she considered voting for the legislation unless he shares the same ILW. This and similar negotiations in legislative bodies constitute another beneficial adaption of the SecretMatch™ PET.
- **Corporate Mergers and Acquisitions (M&A)** – Consider the circumstance where Company A may be interested in acquiring Company B, but disclosing this could result in negative public criticism or it could introduce challenges if Company B's leadership did not share the same ILW. For example, Company B's leadership may interpret the suggestion as having hostile intent or be otherwise insulted if they expected more of a "merger of equals" than an outright acquisition. A privacy-enhanced and fair matchmaking system that supports ILW would offer the opportunity to match secret interests between CEOs or other leaders, of-

fering a risk-free way to explore the various M&A possibilities that might not otherwise be feasible.

- **Affluent and Institutional Investing or Private Equity** – Stock market transactions involving large quantities of securities can result in increased share price volatility or other unintended consequences. Similarly, private equity firms or activist investors have at times publicly applied pressure or advocated changes that were not well received and caused negative impacts to stock prices, public perceptions, or otherwise decreased the probability of achieving the desired goal. A system like SecretMatch™ offers the ability to privately identify shared ILW such as a wish to make a significant investment in a security if and only if some contingent criteria might be satisfied (e.g., at a certain stock price or if a certain action were taken for societal benefit). This potential application offers interesting challenges and tradeoffs with additional complexities such as a more complex nature of the "generic wishes" that may require a more expressive and flexible manifestation, along with navigation of a complex landscape of regulations in the financial sector to avoid even the perception of any possible wrongdoing.
- **Peace and Other Treaty Negotiations** – Most citizens can likely identify volatile situations in modern times requiring negotiations between countries, nations, or other organized people groups. Examples of such situations include avoiding proliferation of nuclear or other weapons of mass destruction, or peace treaty negotiations between long-time warring factions. Consider a case where Alice and Bob are leaders representing their respective countries in peace negotiations after decades of conflict. In this case, neither Alice nor Bob may be willing to admit

willingness to compromise for fear of weakening their negotiating position or their reputation with their own country's citizens. Yet, after decades of hostility, both may strongly desire a compromise and an end to the fighting. A system affording fair and privacy-enhanced matchmaking with ILW offers the ability to secretly and privately find that the leaders share the same desire to compromise, while avoiding the risk of anyone finding out should one of the leaders not share the same ILW. Indeed, it is possible that SecretMatch™ could open doors to peace and other treaty negotiations that might not otherwise be possible.

In summary, follow on work will further adapt the SecretMatch™ technology for other contexts requiring a privacy-enhanced and fair matchmaking system that supports ILW. The first step on this leg of the journey will be to prioritize the possible future applications based on potential to present additional interesting research challenges, positive benefits to society, and monetization potential to provide financial support for future research and development endeavors.

#### 6.2.1.4 Case Study for a NoSQL Matchmaker

Although relational databases dominated the data storage technology landscape for decades, the recent rapid adoption of NoSQL ("Not only SQL") databases for use cases prioritizing scalability, "big data", and variety or complexity of data in recent years has complicated the decision making process with more tradeoffs to consider. As described in the Section 5.4.2, the database selection phase considered both traditional relation databases as well as several categories of NoSQL storage options, concluding that a relational database was the preferred choice for the initial system. Studies comparing per-

formance of relational and non-relational databases such as [155] have supported the assertion of improved performance and scalability with NoSQL databases. Meanwhile, other researchers have presented mixed results [156] or logical discussions questioning common arguments in favor of NoSQL performance superiority [157]. Klein et al. found that achieving strong consistency as opposed to eventual consistency resulted in a 10% - 25% reduction in throughput [158], supporting the notion that relaxing consistency requirements can result in performance improvements. But that study also found significant performance differences when comparing only NoSQL databases, suggesting that NoSQL database performance is strongly influenced by extent to which the query capabilities and data model of the database fit the use cases of the application.

Given the performance results from experimentation of the HN protocol, the database is ultimately expected to be the limiting factor on performance and scalability as the user base grows in size. Consequently, a NoSQL evaluation study is planned to investigate potential performance improvements and tradeoffs comparing relational and NoSQL options in the context of the SecretMatch™ system. There are a number of general comparative analysis papers such as [241], [155], [242], and [243]. Researchers have also explored performance of certain relational and NoSQL database options in specific contexts unrelated to TPP such as Geo Web Services [244], or more generally presenting timing of specific operations like read, write, delete, and fetching all keys as in [156].

The expected contributions of this phase of research and expected outcomes are twofold. First, it will inform specific technology decisions with respect to the SecretMatch™ technology roadmap. But more importantly, it will fill a specific gap in the research landscape that was identified during the database selection phase. That is, the lessons

learned and experimentation will be used to produce a set of criteria, heuristics, and other considerations to guide developers in both academia and industry in database selection efforts by highlighting benefits, risks, tradeoffs, and other considerations to facilitate the decision making process with respect to database selection – a technology choice with critical long-term implications in many contexts.

#### 6.2.1.5 Extended Study of Enhanced Privacy

The experimental evaluation of the performance and feasibility of the HN protocol had positive results, even with mechanisms for enhanced privacy including a VPN, Tor onion routing, and a combination of the two mechanisms. However, for usability reasons, it was determined that the initial system should not include those features out-of-box. One reason is that increasing numbers of users are employing VPNs and anonymity networks already, and those users may represent the most privacy conscious users of the system, and consequently the ideal target user base. But inclusion of current implementations of VPNs or Tor onion routing could introduce compatibility problems preventing users for employing their own choice privacy enhancing services, while also forcing those users to place an even greater level of trust in the SecretMatch™ system for enhanced anonymity purposes.

Future work is planned to evaluate the performance and feasibility of additional approaches to enhanced privacy including I2P [245], Riffle [246], and mix-in-place networks (MIPNets) [247]. Moreover, this line of research will also endeavor to find a path to incorporating the best option or options in such a way as to minimize potential compatibility issues with user-chosen services layered underneath, while simultaneously mini-

mizing any additional level of trust the user might need to place in the SecretMatch™ system by using its included enhanced privacy feature.

### 6.2.2 HPQHN and Quantum Resistance

Quantum resistance will continue to be an important topic as society approaches an era of quantum enabled adversaries. This vein of work expands on the initial HPQHN results for development of an HPQHN enabled embodiment of the SecretMatch™ Prom app as an intermediate step along the journey to a fully Post-Quantum HN (PQHN) protocol. Additionally, the possibility of a quantum-enabled SecretMatch™ technology will also be investigated.

#### 6.2.2.1 Expanded Results and the Hybrid Post-Quantum SecretMatch™ App

With currently used public key cryptosystems on the Internet being vulnerable to quantum enabled adversaries, the stakes are high for endeavors seeking post-quantum algorithms, also known as quantum secure algorithms. The Hybrid Post-Quantum HN (HPQHN) is a hybrid embodiment of the HN protocol that leverages both classical and post-quantum cryptography (PQC) to accomplish fair and privacy-enhanced matchmaking with ILW [199]. The need for a hybrid solution is due to the lack of full confidence in the PQC algorithms in early stages of the NIST standardization process, and prior to significant cryptanalytic scrutiny. The HPQHN protocol was demonstrated to have feasible performance across multiple CPU architectures from mobile to budget and high-end desktop processors using the Kyber and NewHope PQC algorithms.

The next phase of HPQHN research will expand testing to include, at a minimum, the Frodo and SIDH/SIKE PQC implementations. It may also narrow comparative analysis to specific PQC operations to uncover relative empirical performance characteristics of the candidate algorithms with a focus on interpretation for certain use cases such as protocols that might leverage ephemeral keys. The study may also perform the tests with multiple programming languages to quantify the potential performance penalty from use of the various wrapper implementations for the Open Quantum Safe libraries. This detailed characterization of relative performance and tradeoffs will focus on remaining candidates in the NIST PQC competition and thereby benefit system designers and other practitioners. It will also contribute to the body of PQC performance results such as [248] by adding the breadth of results across multiple architectures and individual cryptographic operations, as well as quantifying the potential performance penalties of adopting OQS with different programming languages.

After the complete performance characterization, one or more PQC algorithms will be selected and used to produce an HPQHN based embodiment of the SecretMatch™ Prom app. This version of the app will afford additional quantum safety during the interim period during which quantum computers are rapidly becoming more practical with larger numbers of qubits, but academia and industry do not yet have full confidence in one or more of the PQC candidates due to lack of lengthy cryptanalytic scrutiny.

#### 6.2.2.2 Fully Post-Quantum Horne-Nair Protocol (PQHN)

The natural next step in the evolution of the SecretMatch™ PET after the HPQHN enabled app will be a non-hybrid, fully post-quantum embodiment that does not utilize

RSA or similar classical encryption at all. After the NIST PQC standardization process is complete, and both academia and industry have full confidence in one or more PQC algorithms, the goal will be to have a fully post-quantum HN protocol (PQHN) design ready for incorporation into a PQHN based SecretMatch™ apps. One major research hurdle that must be overcome with this approach is to identify a PQC based solution affording a form of signature with recovery, akin to the design of the original HN protocol. If such a solution cannot be identified, then an alternative embodiment of the HN protocol would need to be devised that still satisfies the full complement of security properties as required for privacy-enhanced and fair matchmaking with support for ILW, including achieving the conflicting goals of authentication and anonymity.

#### 6.2.2.3 Quantum Enabled SecretMatch™ Privacy Enhanced-Technology (PET)

While one perspective on the arrival of practical quantum computers may view the technology as an imminent threat to classical cryptography, an equally valid perspective may view quantum computation as an enabler of a new generation of secure computation and communication. In contrast to PQC efforts, which generally take classical approaches to achieving quantum resistance, quantum enabled solutions attempt to leverage quantum mechanical properties to achieve security and privacy goals. This line of TPP research will evaluate the full spectrum of current and future quantum cryptography candidate concepts with the goal of developing one or more embodiments of a Quantum Enabled HN protocol (QEHN), or a quantum-enabled variant that accomplishes security and privacy properties comparable to the HN protocol in the context of TPP with ILW.

When quantum cryptography is mentioned, it often is used primarily to refer to quantum key distribution (QKD). As a foundational element of secure communications, key distribution rightly deserves attention. Certain facets of quantum physics applied to computation afford new opportunities to securely distribute cryptographic key material, even in the face of quantum-enabled adversaries. The subject of QKD has been extensively studied with a focus on well-known protocols such as BB84 [249] that leverages photon polarization states and Artur Akert's EPR protocol [250] that is based on entangled photon pairs. Protocols like BB84 have been proven to be theoretically secure even against classical or quantum adversaries [251], and QKD protocols have been commercialized by companies such as ID Quantique of Switzerland and Quintessence Labs in Australia.

Research endeavors in the area of secure multiparty quantum computation will also be considered. Common assumptions of multiparty quantum computation protocols include availability of pairwise quantum channels as well as classical broadcast channels. For instance, Ben-Or et al. [252] expanded on the work of [253], highlighting that strictly fair multiparty computation was not possible if $n/2$ out of $n$ players could cheat, while presenting a statistically secure, universally composable multiparty quantum computation protocol that could tolerate up to $\lfloor (n-1)/2 \rfloor$ cheaters. Others have focused on specific problems like a quantum version of Yao's millionaire problem [254] or secure multiparty quantum summation [255] [256] [257]. The active research landscape of secure multiparty quantum computation will be surveyed for candidate building blocks as part of the search for a quantum enabled HN protocol.

Beyond QKD and secure multiparty quantum computation, other quantum based primitives may also be considered as potential enablers. Despite generalized arguments that unconditionally secure quantum bit commitment is impossible [258] [259], the underlying ideas behind quantum bit commitment proposals may be included in the search. Moreover, approaches such as verifiable quantum secret sharing (VQSS) [252] [260], quantum authentications schemes (QAS) [261], and the family of self-duel QAS of [252] will also be evaluated for application, or for inspiration of a truly novel solution. The search for a QEHN protocol embodiment, or a variant thereof that may leverage quantum mechanical properties to achieve fair and privacy-enhanced matchmaking could eventually lead to a quantum enabled SecretMatch™ technology with its own advantages, disadvantages, and tradeoffs to consider.

### 6.2.3 Evolving Privacy Law and PETs Beyond SecretMatch™

As societal perspectives on privacy and privacy related legislation continue to evolve, the intersection of privacy law and rapid technological advancement should continue to be an important research domain. Comparisons such as the discussion in Section 6.1.1 of the GDPR and CCPA highlight many common threads, yet also some variation such as the notion of "opt in" versus "opt out" when it comes to privacy protections. Meanwhile, other comparisons yield more significant dichotomies. For example, the GDPR of the EU opens by stating that [214]: "*The protection of natural persons in relation to the processing of personal data is a fundamental right*." The GDPR goes on to enact significant requirements for the protection of privacy relevant data, which are accompanied by significant potential penalties for non-compliance. The CCPA in the United States also es-

tablished some of the most stringent privacy protections to date in that country [215]. Meanwhile, societal developments elsewhere could be perceived by some as moving in the opposite direction with respect to privacy protections [262] [263] [264] [265]. The growing number of disparate approaches to privacy protections and regulations (or the lack thereof in some cases) can pose significant challenges to businesses aiming to provide products and services to consumers. Yet embracing privacy protections could also be viewed as a potential differentiator.

The work reported in [113] reflected on the role of SecretMatch™, other privacy-enhanced technologies, and the notion of privacy by design (PBD) in the context of the GDPR, CCPA, and the evolving regulatory landscape. Meanwhile, as privacy related legislation continues to evolve, the rapid advancement of technology in parallel introduces new questions and research challenges. For example, machine learning (ML) and artificial intelligence (AI) introduce new risks to privacy and new problems to be solved [266] [267], yet they could also potentially lead to enablers for enhanced privacy as evidenced by the appearance of workshops on privacy-enhancing AI [268]. The prominence of growing numbers of ever increasing volumes of "big data" across industries from consumer products and services to healthcare, finance, and more also continue to introduce privacy related challenges as well as research opportunities [269].

Future research endeavors in this area may include analysis of existing PETs or design and development of novel PETs beyond SecretMatch™. As with the fair and privacy-enhanced matchmaking solutions presented in this body of TPP related research, other new PETs could also potentially contribute to more cost-effective compliance with privacy law by minimizing or eliminating processing or storage of privacy relevant data. The

search for new PETs may start with problems related in some way to TPP. For instance, consider a matchmaking case resembling a sort of love triangle where Alice is interested in Bob, yet Bob is interested in Cate, and so on. There would be some utility in a system that could give hints or make recommendations to facilitate matchmaking in more complicated scenarios such as this. However, the obvious solution to such a system is for a third party to aggregate information about matching interests or preferences, thereby enabling potential recommendations. But this again is counter to the principal goals of the SecretMatch™ system for fair and privacy-enhanced matchmaking, which seeks to avoid aggregation of privacy relevant information or reliance on some level of trust in a third party. A PET that is able to accomplish fair and privacy-enhanced matchmaking facilitation in love triangle type scenarios would need to overcome fundamentally opposing goals akin to the challenges of TPP. Yet a solution, if possible, could be another significant contribution to the field. Another potential step might be to extend the present matchmaking protocol for TPP, which is primarily two-party in nature, to one or more generalized N-party variants of the problem.

As mentioned in section 2.5.5, another opportunity may present itself by revisiting the stable marriages problem from the perspective of fair and privacy-enhanced matchmaking. The very premise of the stable marriages problem is to provide sensitive, privacy-relevant data to a third party with the goal of achieving an optimal matching. Users provide ranked lists of all members of the other group in order of their level of interest in those persons, in exchange for the outcome of the matching service. It is possible that combinations of techniques and lessons learned from TPP, combined with solutions to

related problems like PPPM (discussed in section 2.6.6), may lead to a new PET for a fair and privacy-enhanced variant of the stable marriages problem.

The search for new privacy enabling systems is also expected to venture far beyond matchmaking and matchmaking related problems to other domains as well. Ideal outcomes of this branch of future research endeavors might include introduction of additional novel PETs, further analysis of challenges and opportunities at the intersection of privacy and security with rapidly advancing fields like big data analytics, ML, and AI. Finally, experience from industry shows that comprehensive, yet concise and practical guidance would be of significant benefit for designers of products and services as they attempt to navigate the rapidly evolving regulatory landscape and balance the need for effective monetization strategies with natural persons' rights to protection of personal data.

# APPENDIX A

## Source Code of Original Test Applications

The archived source code for the TPP test applications were made available to committee members at the time of research proposal via SMU OneDrive as password protected zip archives. Passwords were provided to authorized users at that time. The test applications included the following:

1. Proof-of-concept Test Application

   BasicPromApp_ProofOfConcept.zip

2. Feasibility Analysis – Test Application for Computational Costs

   BasicPromApp_CompPerf2-vs2013.zip

3. Feasibility Analysis – Test Applications for Communication Overhead

   GradualReleasePerfTest_ClientServer.zip

# APPENDIX B

## Confidence and Fairness Index Values

Table B.1. Confidence and Fairness Index Values for Bits Released {3…44}

| Number of Bits | $U_1$ Confidence $\lambda_{U1}$ | $U_2$ Confidence $\lambda_{U2}$ | Fairness Index f(x) |
|---|---|---|---|
| 3 | 0.5 | 0.75 | 0.4444444 |
| 4 | 0.75 | 0.75 | 1.0 |
| 5 | 0.75 | 0.875 | 0.7346939 |
| 6 | 0.875 | 0.875 | 1.0 |
| 7 | 0.875 | 0.9375 | 0.8711111 |
| 8 | 0.9375 | 0.9375 | 1.0 |
| 9 | 0.9375 | 0.96875 | 0.9365245 |
| 10 | 0.96875 | 0.96875 | 1.0 |
| 11 | 0.96875 | 0.984375 | 0.9685059 |
| 12 | 0.984375 | 0.984375 | 1.0 |
| 13 | 0.984375 | 0.9921875 | 0.9843140 |
| 14 | 0.9921875 | 0.9921875 | 1.0 |
| 15 | 0.9921875 | 0.99609375 | 0.9921722 |
| 16 | 0.99609375 | 0.99609375 | 1.0 |
| 17 | 0.99609375 | 0.998046875 | 0.9960899 |
| 18 | 0.998046875 | 0.998046875 | 1.0 |
| 19 | 0.998046875 | 0.999023438 | 0.9980459 |
| 20 | 0.999023438 | 0.999023438 | 1.0 |
| 21 | 0.999023438 | 0.999511719 | 0.9990232 |
| 22 | 0.999511719 | 0.999511719 | 1.0 |
| 23 | 0.999511719 | 0.999755859 | 0.9995117 |
| 24 | 0.999755859 | 0.999755859 | 1.0 |
| 25 | 0.999755859 | 0.999877930 | 0.9997558 |
| 26 | 0.999877930 | 0.999877930 | 1.0 |
| 27 | 0.999877930 | 0.999938965 | 0.9998779 |
| 28 | 0.999938965 | 0.999938965 | 1.0 |
| 29 | 0.999938965 | 0.999969482 | 0.9999389 |
| 30 | 0.999969482 | 0.999969482 | 1.0 |
| 31 | 0.999969482 | 0.999984741 | 0.9999694 |
| 32 | 0.999984741 | 0.999984741 | 1.0 |
| 33 | 0.999984741 | 0.999992371 | 0.9999847 |
| 34 | 0.999992371 | 0.999992371 | 1.0 |
| 35 | 0.999992371 | 0.999996185 | 0.9999923 |
| 36 | 0.999996185 | 0.999996185 | 1.0 |
| 37 | 0.999996185 | 0.999998093 | 0.9999961 |

| | | | |
|---|---|---|---|
| 38 | 0.999998093 | 0.999998093 | 1.0 |
| 39 | 0.999998093 | 0.999999046 | 0.9999981 |
| 40 | 0.999999046 | 0.999999046 | 1.0 |
| 41 | 0.999999046 | 0.999999523 | 0.9999990 |
| 42 | 0.999999523 | 0.999999523 | 1.0 |
| 43 | 0.999999523 | 0.999999762 | 0.9999995 |
| 44 | 0.999999762 | 0.999999762 | 1.0 |

# APPENDIX C

## Results from Performance Testing

Table C.1. Mean gradual release runtimes per anonymity approach for bits {8…58}

| Number of Bits | Confidence | Mean Gradual Release Runtime (seconds) per Anonymity Approach | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | None | VPN (Austin, TX) | VPN (Seattle, WA) | VPN (Paris, FR) | Tor | Tor + VPN (Austin, TX) | VPN + Tor (Seattle, WA) |
| 8 | 93.75% | 1.0487 | 1.1550 | 1.2669 | 2.3770 | 2.9114 | 3.0760 | 4.5526 |
| 10 | 96.875% | 1.2951 | 1.4522 | 1.6846 | 2.9711 | 3.6763 | 3.7122 | 5.6282 |
| 12 | 98.4375% | 1.5898 | 1.7814 | 1.9187 | 3.5536 | 4.3784 | 4.4270 | 8.8983 |
| 14 | 99.21875% | 1.7833 | 2.0534 | 2.2678 | 4.1504 | 5.0874 | 5.2126 | 7.9578 |
| 16 | 99.60938% | 2.1061 | 2.3402 | 2.5631 | 4.7518 | 5.8280 | 6.0389 | 10.2172 |
| 18 | 99.80469% | 2.3729 | 2.6605 | 2.8512 | 5.3408 | 6.6390 | 6.8098 | 11.7242 |
| 20 | 99.90234% | 2.6213 | 3.1570 | 3.2000 | 5.9360 | 7.2271 | 7.4931 | 12.9736 |
| 22 | 99.95117% | 2.8743 | 3.2330 | 3.5607 | 6.5047 | 8.0114 | 8.6712 | 14.5650 |
| 24 | 99.97559% | 3.1640 | 3.4984 | 4.0069 | 7.1077 | 8.6345 | 9.2643 | 16.1405 |
| 26 | 99.98779% | 3.3518 | 3.9380 | 4.1309 | 7.7619 | 9.3662 | 9.8200 | 16.5268 |
| 28 | 99.99390% | 3.6352 | 4.1373 | 4.4880 | 8.2639 | 10.1068 | 10.8621 | 16.6817 |
| 30 | 99.99695% | 3.8962 | 5.2780 | 4.5803 | 8.8953 | 10.6436 | 11.4349 | 20.1314 |
| 32 | 99.99847% | 4.1731 | 4.6931 | 4.5990 | 9.5199 | 11.7933 | 11.9700 | 18.8979 |
| 34 | 99.99924% | 4.3681 | 4.9296 | 4.8779 | 10.0917 | 12.2628 | 12.5494 | 19.2008 |
| 36 | 99.99962% | 4.7141 | 5.2141 | 5.1831 | 10.6005 | 12.8162 | 13.3775 | 21.5500 |
| 38 | 99.99981% | 4.9401 | 5.5405 | 5.4865 | 11.2131 | 13.3539 | 14.2360 | 22.7796 |
| 40 | 99.99990% | 5.1805 | 5.8642 | 5.7480 | 12.0283 | 14.2736 | 15.2921 | 25.4774 |
| 42 | 99.99995% | 5.4813 | 6.1176 | 6.0480 | 13.4693 | 15.0270 | 16.0830 | 30.1797 |
| 44 | 99.99998% | 5.7262 | 6.3765 | 6.3070 | 13.7255 | 15.7677 | 16.4071 | 24.7220 |
| 46 | 99.9999881% | 5.9904 | 6.3336 | 6.6000 | 14.3952 | 16.4440 | 17.5206 | 27.3134 |
| 48 | 99.9999940% | 6.2288 | 6.9907 | 7.1423 | 14.8118 | 17.3404 | 18.7541 | 29.8093 |
| 50 | 99.9999970% | 6.4283 | 7.2774 | 7.2077 | 15.1788 | 17.8796 | 19.1432 | 31.8037 |
| 52 | 99.9999985% | 6.7179 | 7.6004 | 7.6780 | 15.9864 | 18.5511 | 20.8230 | 31.2396 |
| 54 | 99.9999993% | 7.0231 | 7.8960 | 7.8025 | 16.2884 | 19.7122 | 21.0748 | 32.4537 |
| 56 | 99.9999996% | 7.4263 | 8.1444 | 7.9169 | 17.2811 | 20.0961 | 22.0000 | 34.3868 |
| 58 | 99.9999998% | 7.6134 | 8.3936 | 8.3222 | 17.5412 | 20.5967 | 21.7767 | 37.3347 |

APPENDIX D

Reproducing the Client-Side Development Environment

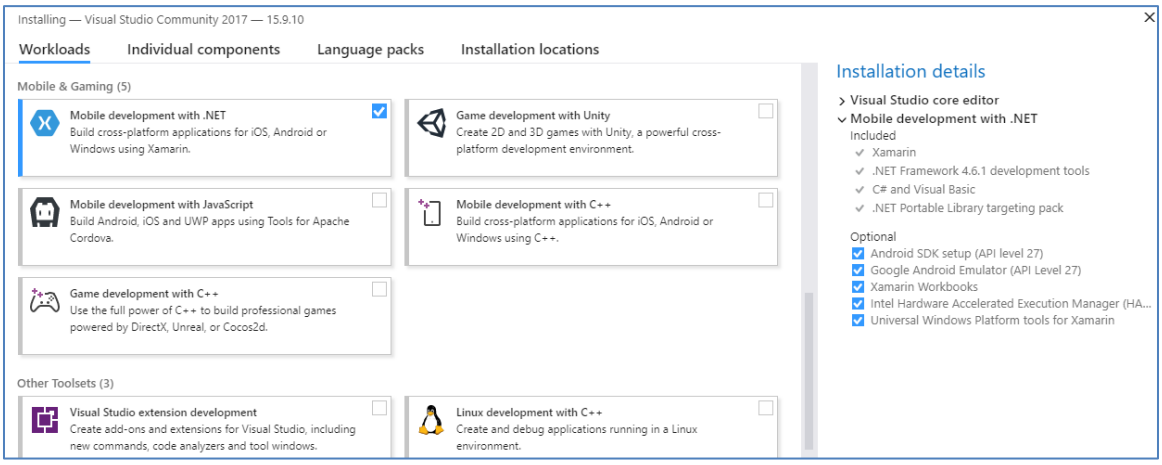
1. Install Microsoft Visual Studio Community 2017 – Version 15.9.19 or greater

   Note: Professional or Enterprise versions are also acceptable.

   a. During installation under Workloads, at a minimum be sure to choose:

      **Mobile development with .NET** – Build cross-platform applications for iOS, Android, or Windows using Xamarin

   b. At a minimum, optional components should also include Android SDK, Android Emulator, Xamarin Workbooks, & Universal Windows Platform tools for Xamarin




2. Install Syncfusion Metro Studio 5 or greater

   Note: This is only required if creating or editing icons for the app

3. Use Android Device Manager to create and manage virtual Android devices

4. Use Android SDK Manager (Android SDKs and Tools) to obtain the desired SDK versions for targeting (must be >= Android 5.0 Lollipop – API Level 21)

5. Use NuGet Package Manager to manage NuGet packages for Solution (after opening SecretMatchProm.sln file)

   Note: See Table D.1 for applicable NuGet package information for the initial version of the SecretMatch™ Prom app. At present it targets Android. A few more dependencies will likely be added for completion of iOS and Windows apps.

Table D.1. NuGet package information for initial version of SecretMatch™ Prom

| Plug-in Name | Version | License |
|---|---|---|
| Xamarin.Forms | 4.4.0 | MIT License |
| Xamarin.Forms.Visual.Material | 4.4.0 | MIT License |
| Xamarin.Facebook.* | 5.11.2 | MIT License |
| Xamarin.Facebook.Android | 5.11.2 | MIT License |
| Xamarin.Android.Support | 28.0.0.3 | MIT License |
| Toasts.Forms.Plugin | 3.3.2 | MIT License |
| Portable.BouncyCastle | 1.8.5.2 | Bouncy Castle License (MIT like) |
| Plugin.Share | 7.1.1 | MIT License |
| NETStandard.Library | 2.0.3 | MIT License |
| Newtonsoft.Json | 12.0.3 | MIT License |
| Flurl | 2.8.2 | MIT License |
| Flurl.Http | 2.4.2 | MIT License |
| Microsoft.NETCore | 6.2.9 | Microsoft Software License |
| Sqlite-net-pcl | 1.6.292 | MIT License |

# APPENDIX E

## Reproducing the Server-Side Development Environment

1. Install Python 3.6 or later

   Source: https://www.python.org/downloads/

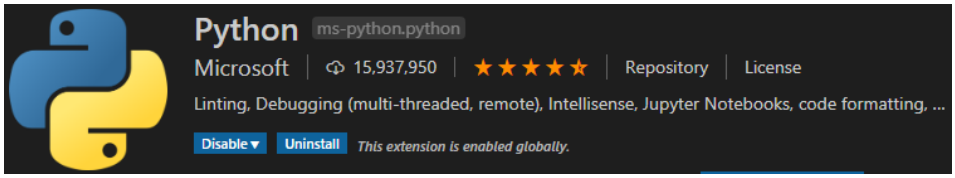
   License: Python Software Foundation (PSF) License Agreement

   https://docs.python.org/3/license.html#terms-and-conditions-for-accessing-or-otherwise-using-python

2. Install Visual Studio Code (or alternatively, Python IDE of choice)

   Source: https://code.visualstudio.com/Download

   a. Within Visual Studio Code, install Python extension from Microsoft (for Pything linting, debugging, etc.)




   Note: Visual Studio Code supports Mac, Linux, and Windows

   License: MIT License (source code); Microsoft Visual Studio Code License (binary)

   https://code.visualstudio.com/license

3. Install REST API testing tool(s)

   a. Firefox with RESTED extension

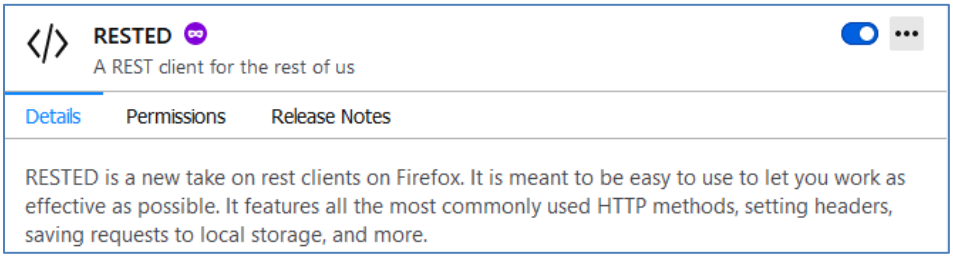
Source: https://www.mozilla.org/en-US/new/

License: Mozilla Public License

https://www.mozilla.org/en-US/MPL/

Source: Firefox > Add-ons Manager > Extensions > RESTED

License: GNU General Public License v3.0




b. Google Chrome with Developer Mode enabled (Extension optional)

Source: https://www.google.com/chrome/

License: Google Chrome Terms of Service (binary)

https://www.google.com/chrome/privacy/eula_text.html

c. Postman

Source: https://www.postman.com

License: Postman EULA

https://www.postman.com/licenses/postman-eula

4. Install MariaDB (or MySQL Community Edition) if testing with local instance rather than a hosted database

Source: https://downloads.mariadb.org

License: GNU General Public License Version 2

https://mariadb.com/kb/en/mariadb-license/

Source: https://dev.mysql.com/downloads/

License: Dual License Model – GNU General Public License or Oracle Commercial License

https://dev.mysql.com/doc/refman/8.0/en/introduction.html

5. Install (via 'pip install') the core Python modules necessary for matchmaker features including:
    a. pymysql
    b. flask
    c. flask_restful

# APPENDIX F

## Source Code for the SecretMatch™ Prom Client and Server Applications

The source code and associated files for the initial server-side matchmaker and the SecretMatch™ Prom client application were archived on the SMU OneDrive. The files are copyright protected as applicable, including the embodiment of the patented HN protocol as implemented. Access will be limited to authorized parties only. Any access required will be arranged on a case-by-case basis.

1. Matchmaker server side portion of SecretMatch™

   SecretMatchServer_v0.9.zip

2. SecretMatch™ Prom app

   SecretMatchClient_v0.9.zip

# APPENDIX G

## Source Code for HPQHN Performance Testing

The source code for the Hybrid Post-Quantum Horne-Nair (HPQHN) test application was archived as a password protected zip file on the SMU OneDrive. The files are copy-right protected as applicable, including the embodiment of the patented HN protocol as implemented. Access will be limited to authorized parties only. Any access required will be arranged on a case-by-case basis.

1. Proof-of-concept HPQHN Test Application

HPQHN-Test-And-Original-20190424.zip

# REFERENCES


[1] D. Horne and S. Nair, "The Prom Problem: Fair and privacy-enhanced matchmaking with identity linked wishes," in *IEEE International Carnahan Conference on Security Technology (ICCST)*, Orlando, FL, 2016.

[2] R. W. Baldwin and W. C. Gramlich, "Cryptographic protocol for trustable matchmaking," in *1985 IEEE Symposium on Security and Privacy*, Oakland, CA, 1985.

[3] D. Horne and S. Nair, "A feasibility evaluation of fair and privacy-enhanced matchmaking with identity linked wishes," in *22nd Australasian Conference on Information Security and Privacy (ACISP)*, Auckland, New Zealand, 2017.

[4] D. Horne and S. Nair, "Privacy-Enhanced and Fair Matchmaking System - Applications and Analysis of Protocol, Architecture, and Performance," in *16th International Conference on Security and Management (SAM)*, Las Vegas, NV, 2017.

[5] Office of the Australian Information Commissioner and Office of the Privacy Commissioner of Canada, "Joint investigation of Ashley Madison by the Privacy Commissioner and Acting Australian Information Commissioner," 2016.

[6] J. S. Shin and V. D. Gligor, "A new privacy-enhanced matchmaking protocol," *IEICE Transactions on Communication,* vol. 96, no. 8, pp. 2049-2059, 1 August 2013.

[7] S. T. Kent, "Internet privacy enhanced mail," *Communications of the ACM,* vol. 36, no. 8, pp. 48-60, 1 August 1993.

[8] D. Balenson, *Privacy enhancement for internet electronic mail: Part III: algorithms, modes, and identifiers,* 1993.

[9] J. Linn, *Privacy enhancement for internet electronic mail: Part I: message encryption and authentication procedure,* 1993.

[10] M. Sobirey, S. Fischer-Hubner and K. Rannenberg, "Pseudonymous audit for privacy enhanced intrusion detection," in *Information Security in Research and Business*, Boston, MA, Springer, 1997, pp. 151-163.

[11] R. Buschkes and D. Kesdogan, "Privacy enhanced intrusion detection," in *Proceedings of the Conference on Multilateral Security for Global Communication*, 1999.

[12] A. Kobsa, "Privacy-enhanced web personalization," in *The adaptive web*, Springer Berlin Heidelberg, 2007, pp. 628-670.

[13] S. M. Bellovin and W. R. Cheswick, *Privacy-enhanced searches using encrypted bloom filters,* vol. 22, 2004.

[14] N. Cao, C. Wang, M. Li, K. Ren and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *Transactions on parallel and distributed systems,* vol. 25, no. 1, pp. 222-233, 2014.

[15] K. W. O'Flaherty, R. G. Stellwagen Jr., T. A. Walter, R. M. Watts, D. A. Ramsey

and A. W. Veldhuisen, "Privacy-enhanced database". United States of America Patent 6,253,203, 26 June 2001.

[16] C.-Y. Chow, M. F. Mokbel and W. G. Aref, "Casper*: Query processing for location services without compromising privacy," *ACM Transactions on Database Systems (TODS),* vol. 34, no. 4, 2009.

[17] M. Raykova, H. Zhao and S. Bellovin, "Privacy enhanced access control for outsourced data sharing," *Financial cryptography and data security,* pp. 223-238, 2012.

[18] R. Heatherly, M. Kantarcioglu and B. Thuraisingham, "Preventing private information inference attacks on social networks," *IEEE Transactions on Knowledge and Data Engineering,* vol. 25, no. 8, pp. 1849-1862, 2013.

[19] E. Aimeur, S. Gambs and A. Ho, "Towards a privacy-enhanced social networking site," in *IEEE International Conference on Availability, Reliability, and Security (ARES'10)*, Krakow, Poland, 2010.

[20] J. Heurix, P. Zimmermann, T. Neubauer and S. Fenz, "A taxonomy for privacy enhancing technologies," *Computers & Security,* vol. 53, pp. 1-17, September 2015.

[21] C. Meadows, "A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party," in *1986 IEEE Symposium on Security and Privacy*, Oakland, CA, 1986.

[22] B. Lee and K. Kim, "Secure matchmaking protocol," in *International Conference on Information Security and Cryptology*, Seoul, Korea, 2000.

[23] K. Zhang and R. Needham, "A private matchmaking protocol," 2001.

[24] K. N. Patrick, "Comparison of documents possessed by two parties". U.S. Patent 8,032,747, 4 October 2011.

[25] J. Bromwich, "Ashley Madison users face threats of blackmail and identity theft," *The New York Times,* 27 August 2015.

[26] A. Hern, "Spouses of Ashley Madison users targeted with blackmail letters," *The Guardian,* 3 March 2016.

[27] S. Thielman, "Toronto police report two suicides associated with Ashley Madison hack," *The Guardian,* 24 August 2015.

[28] J. Eng, "OPM Hack: Government Finally Starts Notifying 21.5 Million Victems," NBC News, 1 October 2015. [Online]. Available: https://www.nbcnews.com/https://www.nbcnews.com/tech/security/opm-hack-government-finally-starts-notifying-21-5-million-victims-n437126. [Accessed 17 August 2017].

[29] E. Weise, "Second hack at OPM may have been worse than first," *USA Today,* 12 June 2015.

[30] H. Berghel, "Equifax and the Latest Round of Identity Theft Roulette," *Computer,* pp. 72-76, December 2017.

[31] V. Boyko, P. MacKenzie and S. Patel, "Provably secure password-authenticated

key exchange using Diffie-Hellman," in *Advances in Cryptology--EUROCRYPT 2000*, Brugs, Belgium, 2000.

[32] M. Bellare, D. Pointcheval and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *Advances in Cryptology - EUROCRYPT 2000*, Brugs, Belgium, 2000.

[33] M. J. Atallah and Y. Cho, "Private discovery of shared interests," in *Proceedings 9th International Conference on Information and Communications Security*, Zhengzhou, China, 2007.

[34] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory,* vol. 22, no. 6, pp. 644-654, 1976.

[35] P. J. M. Veugen, M. O. Van Deventer and V. B. Klos, "Shared secret verification method and system". U.S. Patent 8,527,765, 3 September 2013.

[36] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *The American Mathematical Monthly,* vol. 69, no. 1, pp. 9-15, 1962.

[37] D. G. McVitie and L. B. Wilson, "The stable marriage problem," *Communications of the ACM,* vol. 14, no. 7, pp. 486-490, 1 July 1971.

[38] S. Y. Itoga, "The upper bound for the stable marriage problem," *Journal of the Operational Research Society,* vol. 29, no. 8, pp. 811-814, 1978.

[39] D. E. Knuth, "Stable marriage and its relation to other combinatorial problems," in *An introduction to the mathematical analysis of algorithms*, vol. 10, American Mathematical Society, 1997.

[40] R. W. Irving, "An efficient algorithm for the "stable roommates" problem," *Journal of Algorithms,* vol. 6, pp. 577-595, 1985.

[41] R. W. Irving, P. Leather and D. Gusfield, "An efficient algorithm for the "optimal" stable marriage," *Journal of the Association of Computing Machinery,* vol. 34, no. 3, pp. 532-543, 1987.

[42] A. Subramanian, "A new approach to stable matching problems," *Technical Report, STAN-CS-89-1275,* 1989.

[43] T. Feder, "A new fixed point approach for stable networks and stable marriages," *Journal of Computer and System Sciences,* vol. 45, pp. 233-284, 1992.

[44] A. C. Yao, "Protocols for secure computations," in *23rd Annual Symposium on Foundations of Computer Science*, Chicago, IL, 1982.

[45] O. Goldreich, S. Micali and A. Wigderson, "How to play any mental game," in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, New York, NY, 1987.

[46] I. Ioannidis and A. Grama, "An efficient protocol for Yao's millionaires' problem," in *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, Big Island, Hawaii, 2003.

[47] F. Boudot, B. Schoenmakers and J. Traore, "A fair and efficient solution to the socialist millionaires' problem," *Discrete Applied Mathematics,* vol. 111, no. 1, pp. 23-36, 2001.

[48] W. Du and M. J. Atallah, "Secure multi-party computation problems and their applications: a review and open problems," in *Proceedings of the 2001 workshop on New security paradigms*, Cloudcroft, NM, 2001.

[49] O. Goldreich, "Secure multi-party computation," Manuscript. Preliminary version, 1998.

[50] B. Chor, O. Goldreich, E. Kushilevitz and M. Sudan, "Private information retrieval," in *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, Milwaukee, WI, 1995.

[51] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: single database, computationally-private information retrieval," in *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, Miami Beach, FL, 1997.

[52] Y. Gertner, Y. Ishai, E. Kushilevitz and T. Malkin, "Protecting data privacy in private information retrieval schemes," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, Dallas, TX, 1998.

[53] G. Di-Crescenzo, Y. Ishai and R. Ostrovsky, "Universal service-providers for database private information retrieval," in *PODC '98 Proceedings of the seventeenth annual ACM symposium on Principals of distributed computing*, Puerto Vallarta, Mexico, 1998.

[54] Y. Ishai and E. Kustilevitz, "Improved upper bounds on information-theoretic private information retrieval," in *STOC '99 Proceedings of the thirty-first annual ACM symposiom on Theory of computing*, Atlanta, GA, 1999.

[55] C. Cachin, S. Micali and M. Stadler, "Compuationally private information retrieval with polylogarithmic communication," in *EUROCRYPT '99 Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, Prague, 1999.

[56] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *SIGMOD '00 Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, Dallas, TX, 2000.

[57] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Advances in Cryptology-CRYPTO 2000*, Santa Barbara, CA, 2000.

[58] Y. Lindell and B. Pinkas, "Privacy preserving data mining," *Journal of cryptology,* vol. 15, no. 3, pp. 177-206, 1 July 2002.

[59] J. Vaidya, C. Clifton and Y. M. Zhu, Privacy preserving data mining, vol. 19, S. S. &. B. Media, Ed., 2006.

[60] D. Agrawal and C. C. Aggarwal, "On the design and quantification of privacy preserving data mining algorithms," in *PODS '01 Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, Santa Barbara, CA, 2001.

[61] B. Hore, S. Mehrotra and G. Tsudik, "A privacy-preserving index for range queries," in *VLDB '04 Proceedings of the Thirtieth International Conference on Very Large Data Bases*, Toronto, Canada, 2004.

[62] P. Lincoln, P. Porras and V. Shmatikov, "Privacy-preserving sharing and correction

of security alerts," in *SSYM '04 Proceedings of the 13th conference on USENIX Security Symposium - Volume 13*, San Diego, CA, 2004.

[63] J. Biskup and U. Flegel, "Transaction-based pseudonyms in audit data for privacy respecting intrusion detection," in *Recent Advances in Intrusion Detection*, Toulouse, 2000.

[64] W. Du and M. J. Atallah, "Privacy-preserving cooperative statistical analysis," in *Proceedings 17th Annual Computer Security Applications Conference, ACSAC 2001*, New Orleans, LA, 2001.

[65] W. Du, Y. S. Han and S. Chen, "Privacy-preserving multivariate statistical analysis: Linear regression and classification," in *Proceedings of the 2004 SAIM international conference on data mining*, Lake Buena Vista, CA, 2004.

[66] W. Du and M. J. Atallah, "Privacy-preserving cooperative scientific computations," *csfw,* vol. 1, p. 273, 11 June 2001.

[67] C. Cachin, "Efficient private bidding and auctions with an oblivious third party," in *CCS '99 Proceedings of the 6th ACM conference on Computer and communications security*, Singapore, 1999.

[68] M. Naor, B. Pinkas and R. Sumner, "Privacy preserving auctions and mechanism design," in *EC '99 Proceedings of the 1st ACM conference on Electronic commerce*, Denver, CO, 1999.

[69] E. Prouff and T. Roche, "Higher-order glitches free implementation of the AES using secure multi-party computation protocols," in *Cryptographic Hardware and Embedded Systems - CHES 2011*, Nara, Japan, 2011.

[70] D. Bogdanov, R. Talviste and J. Willemson, "Deploying secure multi-party computation for financial data analysis," in *Financial Cryptography and Data Security*, 2012.

[71] W. Henecka, S. Kogl, A.-R. Sadeghi, T. Schneider and I. Wehrenberg, "TASTY: tool for automating secure two-party computations," in *CCS '10 Proceedings of the 17th ACM conference on Computer and communications security*, Chicago, IL, 2010.

[72] S. Gorbunov, V. Vaikuntanathan and H. Wee, "Functional encryption with bounded collusions via multi-party computation," in *Advances in Cryptology - CRYPTO 2012*, 2012.

[73] S. Goldwasser, S. Micali and C. Rackoff, "The knowledge complexity of interactive proof systems," in *STOC '85 Proceedings of the seventeenth annual ACM symposium on Theory of computing*, Providence, RI, 1985.

[74] S. Goldwasser, S. Micali and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on computing,* vol. 18, no. 1, pp. 186-208, 1989.

[75] M. Ben-Or, S. Goldwasser, J. Kilian and A. Wigderson, "Multi-prover interactive proofs: How to remove intractability assumptions," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*, Chicago, IL, 1988.

[76] O. Goldreich, S. Micali and A. Wigderson, "Proofs that yield nothing but their

validity or all languages in NP have zero-knowledge proof systems," *Journal of the ACM (JACM),* vol. 38, no. 3, pp. 690-728, 1991.

[77] J.-J. Quisquater, M. Quisquater, M. Quisquater, M. Quisquater, L. Guillou, M. Annick Guillou, G. Guillou, A. Guillou, G. Guillou and S. Guillou, "How to explain zero-knowledge protocols to your children," in *Conference on the Theory and Application of Cryptology*, New York, NY, 1989.

[78] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Conference on the Theory and Application of Cryptographic Techniques*, Santa Barbara, CA, 1986.

[79] M. Stamp, Information security: principles and practice, Hoboken, NJ: John Wiley & Sons, 2011.

[80] D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon and H.-C. Wong, "Secret handshakes from pairing-based key agreements," in *Proceedings of the 2003 Symposium on Security and Privacy*, Berkeley, CA, 2003.

[81] J. H. Hoepman, "Private handshakes," *ESAS,* vol. 4572, pp. 31-42, 2007.

[82] M. J. Freedman, K. Nissim and B. Pinkas, "Efficient private matching and set intersection," in *Proceedings of the International conference on the theory and applications of cryptographic techniques (EUROCRYPT '04)*, Interlaken, Switzerland, 2004.

[83] L. Kissner and D. Song, "Privacy-preserving set operations," *Crypto,* vol. 3621, pp. 241-257, 2005.

[84] E. De Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity," *Financial Cryptography,* vol. 10, pp. 143-159, 2010.

[85] P. Duan, Oblivious handshakes and sharing of secrets of privacy-preserving matching and authentication protocols, Texas A&M University, 2011.

[86] Y. Sang, H. Shen, Y. Tan and N. Xiong, "Efficient protocols for privacy preserving matching against distributed datasets," in *International Conference on Information and Communications Security*, 2006.

[87] Q. Ye, H. Wang and J. Pieprzyk, "Distributed private matching and set operations," in *International Conference on Information security practice and experience*, 2008.

[88] Q. Xie and U. Hengartner, "Privacy-preserving matchmaking for mobile social networking secure against malicious users," in *Proceedings of the Ninth Annual International Conference on Privacy, Security and Trust (PST)*, 2011.

[89] M. Li, N. Cao, S. Yu and W. Lou, "Findu: Privacy-preserving personal profile matching in mobile social networks," in *Proceedings of IEEE INFOCOM 2011*, 2011.

[90] M. Li, S. Yu, N. Cao and W. Lou, "Privacy-preserving distributed profile matching in proximity-based mobile social networks," *IEEE Transactions on Wireless Communications,* vol. 12, no. 5, pp. 2024-2033, 2013.

[91] I. Ioannidis, A. Grama and M. Atallah, "A secure protocol for computing dot-products in clustered and distributed environments," in *Proceedings of the*

*International Conference on Parallel Processing*, 2002.

[92] W. Dong, V. Dave, L. Qiu and Y. Zhang, "Secure friend discovery in mobile social networks," in *2011 Proceedings of IEEE INFOCOM*, 2011.

[93] R. Zhang, J. Zhang, Y. Zhang, J. Sun and G. Yan, "Privacy-preserving profile matching for proximity-based mobile social networking," *IEEE Journal on Selected Areas in Communications,* vol. 31, no. 9, pp. 656-668, 2013.

[94] C. Hu, R. Li, W. Li, J. Yu, Z. Tian and R. Bie, "Efficient privacy-preserving schemes for dot-product computation in mobile computing," in *Proceedings of the 1st ACM Workshop on Privacy-Aware Mobile Computing*, 2016.

[95] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International Conference on the Theory and Applications of Cryptographic Techniques*, 1999.

[96] H. Zhu, S. Du, M. Li and Z. Gao, "Fairness-aware and privacy-preserving friend matching protocol in mobile social networks," *IEEE Transactions on Emerging Topics in Computing,* vol. 1, no. 1, pp. 192-200, 2013.

[97] S. Sarpong, C. Xu and X. Zhang, "An authenticated privacy-preserving attribute matchmaking protocol for mobile social networks," *International Journal of Network Security,* vol. 17, no. 3, pp. 357-364, 2015.

[98] X. Liang, X. Li, K. Zhang, R. Lu, X. Lin and X. S. Shen, "Fully anonymous profile matching in mobile social networks," *IEEE Journal on Selected Areas in Communications,* vol. 31, no. 9, pp. 641-655, 2013.

[99] L. Zhang, X.-Y. Li and Y. Liu, "Message in a sealed bottle: Privacy preserving friending in social networks," in *IEEE 33rd International Conference on Distributed Computing Systems (ICDCS)*, 2013.

[100] R. Li, H. Li, X. Cheng, X. Zhou, K. Li, S. Wang and R. Bie, "Perturbation-Based Private Profile Matching in Social Networks," *IEEE Access,* vol. 5, pp. 19720-19732, 2017.

[101] S. Nair and R. D. Horne, "Method and system for privacy preserving disclosure of a shared, identity linked secret". United States Patent 10,432,400, 1 October 2019.

[102] M. Bellare, A. Boldyreva, A. Desai and D. Pointcheval, "Key-privacy in public-key encryption," in *International Conference on the Theory and Application of Cryptology and Information Security*, Gold Coast, Australia, 2001.

[103] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM,* vol. 24, no. 2, pp. 84-90, 1981.

[104] P. F. Syverson, D. M. Goldschlag and M. G. Reed, "Anonymous connections and onion routing," in *1997 IEEE Symposioum on Security and Privacy*, Oakland, CA, 1997.

[105] M. K. Reiter and A. D. Rubin, "Crowds: anonymity for Web transactions," *ACM Transactions on information and systems security (TISSEC),* vol. 1, no. 1, pp. 66-92, 1998.

[106] M. J. Freedman and R. Morris, "Tarzan: A peer-to-peer anonymizing network

layer," in *Proceedings of the 9th ACM conference on Computer and communications security*, Washington, DC, 2002.

[107] J. Kong and X. Hong, "ANODR: anonymous on demand routing with untraceable routes for mobile ad-hoc networks," in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, Annapolis, MD, 2003.

[108] R. Sherwood, B. Bhattacharjee and A. Srinivasan, "P5: A protocol for scalable anonymous communication," *Journal of Computer Security,* vol. 13, no. 6, pp. 839-876, 2005.

[109] W. Stallings and M. P. Tahiliani, Cryptography and network security: principles and practice, vol. 6, London: Pearson, 2014.

[110] R. L. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM,* vol. 21, no. 2, pp. 120-126, 1978.

[111] M. J. Dworkin, "SHA-3 standard: Permutation-based hash and extendable-output functions," Federal Inf. Process. Stds. (NIST FIPS) - 202, National Institute of Standards and Technology, Gaithersburg, 2015.

[112] A. Abdul-Rahman, "The PGP Trust Model," *EDI-Forum: the Journal of Electronic Commerce,* vol. 10, no. 3, pp. 27-31, 1997.

[113] D. Horne and S. Nair, "A new privacy-enhanced technology for fair matchmaking with identity linked wishes," *IEEE Systems Journal,* 14 March 2019.

[114] D. Beaver and S. Goldwasser, "Multiparty computation with faulty majority," in *Conference on the Theory and Application of Cryptology (CRYPTO '89)*, Santa Barbara, CA, 1989.

[115] T. Lan, D. Kao, M. Chiang and A. Sabharwal, "An axiomatic theory of fairness in network resource allocation," in *Proceedings of the 29th Conference on Information Communication (INFOCOM '10)*, San Diego, CA, 2010.

[116] C. Joe-Wong, S. Sen, T. Lan and M. Chiang, "Multiresource allocation: Fairness-efficiency tradeoffs in a unifying framework," *IEEE/ACM Transactions on Networking (TON),* vol. 21, no. 6, pp. 1785-1798, 2013.

[117] R. Jain, D.-M. Chiu and W. R. Hawe, A quantitative measure of fairness and discrimination for resource allocation in shared computer system, vol. 38, Hudson, MA: Eastern Research Laboratory, Digital Equipment Corporation, 1984.

[118] R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J. P. Fernandes and J. Saraiva, "Energy efficiency across programming languages: How do energy, time, and memory relate?," in *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering (SLE)*, Vancouver, BC, Canada, 2017.

[119] "CPOL: Code Project Open License," CodeProject, 15 April 2008. [Online]. Available: https://www.codeproject.com/info/cpol10.aspx. [Accessed 24 September 2017].

[120] A. Jati, "RSA Library with Private Key Encryption in CSharp," The Code Project, 15 July 2012. [Online]. Available:

http://www.codeproject.com/Articles/421656/RSA-Library-with-Private-Key-Encryption-in-Csharp. [Accessed 5 July 2015].

[121] Licensing and Compliance Lab, "Various Licenses and Comments About Them - GNU Project - Free Software Foundation," Free Software Foundation, Inc., 26 August 2017. [Online]. Available: https://www.gnu.org/licenses/license-list.html#cpol. [Accessed 24 September 2017].

[122] "Developer Guilde | Protocol Buffers | Google Developers," Google, 31 May 2017. [Online]. Available: https://developers.google.com/protocol-buffers/docs/overview. [Accessed 24 September 2017].

[123] M. Howard and D. LeBlanc, Writing secure code, Pearson Education, 2003.

[124] M. Howard and S. Lipner, The security development lifecycle, Redmond: Microsoft Press, 2006.

[125] G. McGraw, Software security: building security in, Addison-Wesley Professional, 2006.

[126] R. Buyya, R. Ranjan and R. N. Calheiros, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *International Conference on High Performance Computing & Simulation (HPCS '09)*, Leipzig, Germany, 2009.

[127] W. Vogels, "Eventually consistent," *Communications of the ACM,* vol. 52, no. 1, pp. 40-44, 2009.

[128] V. Cardellini, M. Colajanni and P. S. Yu, "Geographic load balancing for scalable distributed web systems," in *8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOT '00)*, San Fransisco, CA, 2000.

[129] G. Golden Frog, "VyperVPN - World's Most Powerful and Secure VPN | Golden Frog," Golden Frog, GmbH, [Online]. Available: https://www.goldenfrog.com/vyprvpn. [Accessed 25 September 2017].

[130] "Compare Anonabox Tor and VPN Routers," anonabox, [Online]. Available: https://www.anonabox.com/compare-routers.html. [Accessed 25 September 2017].

[131] "Amazon Web Services (AWS) - Cloud Computing Services," Amazon Web Services, Inc., [Online]. Available: https://aws.amazon.com/. [Accessed 25 September 2017].

[132] R. Pries, W. Yu, S. Graham and X. Fu, "On performance bottleneck of anonymous communication networks," in *2008 International Symposium on Parallel and Distributed Processing IPDPS 2008*, 2008.

[133] A. Cavoukian, "Privacy by Design - The 7 Foundational Principles," Information and Privacy Commissioner of Ontario, Toronto, Ontario, Canada, 2009.

[134] K. Varda, "Protocol buffers: Google's data interchange format," Google Open Source Blog, 7 July 2008. [Online]. Available: https://opensource.googleblog.com/2008/07/protocol-buffers-googles-data.html. [Accessed 12 September 2017].

[135] R. T. Fielding and R. N. Taylor, "Principled design of the modern web architecture," *ACM Transactions on Internet Technology (TOIT),* vol. 2, no. 2, pp. 115-150, 2002.

[136] D. Crockford, "The application/json media type for javascript object notation (json)," The Internet Society, 2006. [Online]. Available: https://tools.ietf.org/html/rfc4627.

[137] T. Bray, "The JavaScript Object Notation (JSON) Data Interchange Format," Internet Engineering Task Force (IETF), 2014. [Online]. Available: https://tools.ietf.org/html/rfc7159.

[138] National Bureau of Standards, "Data Encryption Standard - Federal information processing standards publication 46, FIPS 46-3," US Department of Commerce, 1977.

[139] E. Biham and A. Shamir, Differential cryptanalysis of the data encryption standard, Springer Science & Business Media, 2012.

[140] M. A. Nielsen and I. L. Chuang, "Quantum computation and quantum information," *Quantum,* vol. 546, 2000.

[141] S. Pugh, Creating innovative products: Using total design, Addison-Wesley, 1996.

[142] TIOBE software BV, "TIOBE index," TIOBE software BV, March 2018. [Online]. Available: https://www.tiobe.com/tiobe-index/. [Accessed 24 March 2018].

[143] R. Hecht and S. Jablonski, "NoSQL evaluation: A use case oriented survey," in *Proceedings of the 2011 International Conference on Cloud and Service Computing*, 2011.

[144] Dormando, "memcached - a distributed memory object caching system," [Online]. Available: http://memcached.org/. [Accessed 25 01 2020].

[145] S. Sanfilippo, "Redis," Redis Labs, [Online]. Available: https://redis.io. [Accessed 25 01 2020].

[146] MongoDB, Inc., "The most popular database for modern apps | MongoDB," MongoDB, Inc., [Online]. Available: https://www.mongodb.com/. [Accessed 15 01 2020].

[147] The Apache Software Foundation, "Apache CouchDB," The Apache Software Foundation, [Online]. Available: https://couchdb.apache.org/. [Accessed 25 01 2020].

[148] The Apache Software Foundation, "Apache Cassandra," The Apache Software Foundation, [Online]. Available: https://cassandra.apache.org/. [Accessed 25 January 2020].

[149] Google, "Cloud Bigtable | Google Cloud," Google, [Online]. Available: https://cloud.google.com/bigtable/. [Accessed 25 January 2020].

[150] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, Burrows, Mike, T. Chandra, A. Fikes and R. E. Gruber, "Bigtabler: A distributed storage system for structured data," *ACM Transactions on Computer Systems (TOCS),* vol. 26, no. 2, 2008.

[151] "GitHub - sones/sones: sones GraphDB - open source graph database," sones GmbH, [Online]. Available: https://github.com/sones/sones. [Accessed 25 January 2020].

[152] Redis Labs, "RedisGraph | Redis Labs," Redis Labs, [Online]. Available: https://redislabs.com/redis-enterprise/redis-graph/. [Accessed 25 January 2020].

[153] "GitHub - RedisGraph/RedisGraph: A graph database as a Redis module," GitHub, Inc., [Online]. Available: https://github.com/RedisGraph/RedisGraph. [Accessed 25 January 2020].

[154] D. Pritchett, "BASE: An Acid Alternative," *Queue,* vol. 6, no. 3, pp. 48-55, May 2008.

[155] C. Győrödi, R. Győrödi, G. Pecherle and A. Olah, "A comparative study: MongoDB vs. MySQL," in *015 13th International Conference on Engineering of Modern Electric Systems (EMES)*, 2015.

[156] Y. Li and S. Manoharan, "A performance comparison of SQL and NoSQL databases," in *2013 IEEE Pacific Rim Conference on Communications*, 2013.

[157] M. Stonebraker, "SQL databases v. NoSQL databases," *Communications of the ACM,* vol. 53, no. 4, pp. 10-11, 2010.

[158] J. Klein, I. Gorton, N. Ernst, P. Donohoe, K. Pham and C. Matser, "Performance evaluation of NoSQL databases: a case study," in *Proceedings of the 1st Workshop on Performance Analysis of Big Data Systems*, 2015.

[159] O. Valentine, "VPN Usage and Trends Around the World in 2018 - GlobalWebIndex," GlobalWebIndex, 2 July 2018. [Online]. Available: https://blog.globalwebindex.com/chart-of-the-day/vpn-usage-2018/. [Accessed 25 January 2020].

[160] S. Liu, "VPN Market Size Worldwide 2016-2022 | Statista," Statista, 7 June 2018. [Online]. Available: https://www.statista.com/statistics/542817/worldwide-virtual-private-network-market/. [Accessed 25 January 2020].

[161] K. Thompson, "Reflections on trusting trust," *Communications of the ACM,* vol. 27, no. 8, pp. 761-763, 1984.

[162] UX Measure, "30 UX Statistics You Should Not Ignore! [INFOGRAPHIC] | Experience Dynamics," Experience Dynamics, 18 March 2015. [Online]. Available: https://www.experiencedynamics.com/blog/2015/03/30-ux-statistics-you-should-not-ignore-infographic. [Accessed 25 January 2020].

[163] B. Shneiderman, Designing the user interface: Strategies for effective human-computer interaction, 1st ed., Reading, MA: Addison-Wesley, 1987.

[164] J. Nielsen and R. Molich, "Heuristic evaluation of user interfaces," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1990.

[165] Apple Inc., "Themes - iOS - Human Interface Guidelines - Apple Developer," Apple Inc., 2020. [Online]. Available: https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/. [Accessed 25 January 2020].

[166] Google, "User Interface & Navigation - Android Developers," Google, 27

December 2019. [Online]. Available: https://developer.android.com/guide/topics/ui/. [Accessed 25 January 2020].

[167] Microsoft, "Guidelines - Win32 Apps | Microsoft Docs," Microsoft, 17 January 2020. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/uxguide/guidelines. [Accessed 25 January 2020].

[168] J. Johnson, Designing with the mind in mind: simple guide to understanding user interface design guidelines, Elsevier, 2013.

[169] B. Shneiderman, C. Plaisant, M. Cohen, S. Jacobs, N. Elmqvist and N. Diakopoulos, Designing the user interface: strategies for effective human-computer interaction, Pearson, 2016.

[170] D. Stone, C. Jarrett, M. Woodroffe and S. Minocha, User Interface Design and Evaluation, 1st Edition, Morgan Kaufmann, 2005.

[171] R. Hartson and P. S. Pyla, The UX Book: Process and guidelines for ensuring a quality user experience, Elsevier, 2012.

[172] Legion of the Bouncy Castle Inc., "The Legion of the Bouncy Castle C# Cryptography APIs," 27 October 2018. [Online]. Available: https://www.bouncycastle.org/csharp/. [Accessed 25 January 2020].

[173] National Institute of Standards and Technology, U.S. Department of Commerce, "Certificate Detail - Cryptographic Module Validation Program | CSRC - Certificate #2792," 14 November 2016. [Online]. Available: https://csrc.nist.gov/projects/cryptographic-module-validation-program/Certificate/2792. [Accessed 25 January 2020].

[174] Legion of the Bouncy Castle Inc., "License - bouncycastle.org," 2019. [Online]. Available: https://bouncycastle.org/licence.html. [Accessed 25 January 2020].

[175] Opensource.org, "The MIT License | Open Source Initiative," [Online]. Available: https://opensource.org/licenses/MIT. [Accessed 25 January 2020].

[176] M. Masse, REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces, O'Reilly Media, Inc., 2011.

[177] P. Eby, "PEP 3333 -- Python Web Server Gateway Interface v1.0.1," Python Software Foundation, 26 September 2010. [Online]. Available: https://www.python.org/dev/peps/pep-3333/. [Accessed 21 February 2020].

[178] K. Burke, K. Conroy, R. Horn, F. Stratton and G. Binet, "Flask-RESTful -- Flask-RESTful 0.3.8 Documentation," [Online]. Available: https://flask-restful.readthedocs.io/en/latest/. [Accessed 21 February 2020].

[179] "PyMySQL - PyPI," Python Software Foundation, 18 December 2018. [Online]. Available: https://pypi.org/project/PyMySQL/. [Accessed 21 February 2020].

[180] "MySQL :: MySQL Community Edition," Oracle Corporation, [Online]. Available: https://www.mysql.com/products/community/. [Accessed 21 February 2020].

[181] "Downloads - MariaDB," MariaDB Corporation, [Online]. Available: https://downloads.mariadb.org/. [Accessed 21 February 2020].

[182] E. H, "RESTED - Get this extension for Firefox (en-US)," [Online]. Available:

https://addons.mozilla.org/en-US/firefox/addon/rested/. [Accessed 21 February 2020].

[183] "Postman | The Collaboration Platform for API Development," Postman, Inc., [Online]. Available: https://www.postman.com/. [Accessed 21 February 2020].

[184] C. Petzold, Creating Mobile Apps with Xamarin.Forms, Redmond, WA: Microsoft Press, 2016.

[185] "Flutter Documentation - Flutter," Google, [Online]. Available: https://flutter.dev/docs. [Accessed 25 January 2020].

[186] A. Biørn-Hansen, T. A. Majchrzak and T.-M. Grønli, "Progressive Web Apps: The Possible Web-native Unifier for Mobile Development," in *International Conference on Web Information Systems and Technologies (WEBIST)*, 2017.

[187] "Documentation - Mono," Mono Project, 2020. [Online]. Available: https://www.mono-project.com/. [Accessed 25 January 2020].

[188] A. Del Sole, Xamarin.Forms Succinctly, Morrisville, NC: Syncfusion, Inc., 2018.

[189] "Generated Photos | Unique, worry-free model photos," Generated Media, Inc., 2019. [Online]. Available: https://generated.photos/. [Accessed 21 February 2020].

[190] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*, 1994.

[191] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review,* vol. 41, no. 2, pp. 303-332, 1999.

[192] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996.

[193] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Physical review letters,* vol. 79, no. 2, pp. 325-328, 1997.

[194] M. Roetteler, M. Naehrig, K. M. Svore and K. Lauter, "Quantum resource estimates for computing elliptic curve discrete logarithms," in *International Conference on the Theory and Application of Cryptology and Information Security*, Cham, 2017.

[195] National Institute of Standards and Technology (NIST), "Post Quantum Cryptography | CSRC," U.S. Department of Commerce, 3 January 2017. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography. [Accessed 24 January 2020].

[196] G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, Y.-K. Liu, C. Miller, D. Moody, R. Peralta, R. Perlner, A. Robinson and D. Smith-Tone, "NISTIR 8240 - Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process," National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Gaithersburg, MD, 2019.

[197] D. Moody, "Round 2 of the NIST PQC "Competition" - What was NIST

thinking?," National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Gaithersburg, MD, 2019.

[198] D. Stebila and M. Mosca, "Post-quantum key exchange for the internet and the open quantum safe project," in *International Conference on Selected Areas in Cryptography (SAC)*, 2016.

[199] D. Horne and S. Nair, "Toward a quantum resistant SecretMatch privacy enhanced technology–A case study," in *Proceedings of the International Conference on Security and Management (SAM)*, 2019.

[200] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM (JACM),* vol. 55, no. 6, pp. 84-93, 2009.

[201] C. Peikert, "A decade of lattice cryptography," *Foundations and Trends® in Theoretical Computer Science,* vol. 10, no. 4, pp. 283-424, 2016.

[202] J. Bos, C. Costello, L. Ducas, I. Mironov, M. Naehrig, V. Nikolaenko, A. Raghunathan and D. Stebila, "Frodo: Take off the ring! practical, quantum-secure key exchange from LWE," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[203] E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe, "Post-quantum key exchange—a new hope," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, 2016.

[204] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler and D. Stehlé, "CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM," in *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2018.

[205] C. Costello, P. Longa and M. Naehrig, "Efficient algorithms for supersingular isogeny Diffie-Hellman," in *Annual International Cryptology Conference*, 2016.

[206] R. Azarderakhsh, M. Campagna, C. Costello, L. Feo, B. Hess, A. Jalali, D. Jao, B. Koziel, B. LaMacchia, P. Longa and M. Naehrig, "Supersingular isogeny key encapsulation," Submission to the NIST Post-Quantum Standardization project, 2017.

[207] P. Longa, "A note on post-quantum authenticated key exchange from supersingular isogenies," *IACR Cryptology ePrint Archive,* p. 267, 2018.

[208] S. Bertoni, "Tinder Swipes Right to Revenue, Will Add Premium Service in November," Forbes, 20 October 2014. [Online]. Available: https://www.forbes.com/sites/stevenbertoni/2014/10/20/tinder-swipes-right-to-revenue-will-add-premium-service-in-november/. [Accessed 12 October 2017].

[209] K. Giuliano, "Tinder swipes right on monetization," CNBC, 2 March 2015. [Online]. Available: https://www.cnbc.com/2015/03/02/-monetization.html. [Accessed 12 October 2017].

[210] A. Carman, "Tinder made $1.2 billion last year off people who can’t stop swiping - The Verge," Vox Media, LLC, 4 February 2020. [Online]. Available: https://www.theverge.com/2020/2/4/21123057/tinder-1-billion-dollars-match-group-revenue-earnings. [Accessed 14 February 2020].

[211] CNBC LLC, "Match Group earnings Q2 2019: Tinder adds 1.5 million subscribers," CNBC LLC, A Division of NBCUniversal, 6 August 2019. [Online]. Available: https://www.cnbc.com/2019/08/06/match-group-earnings-2019-tinder-subscribers.html. [Accessed 14 February 2020].

[212] J. Duportail, "I asked Tinder for my data. It sent me 800 pages of my deepest, darkest secrets," The Guardian, 26 September 2017. [Online]. Available: https://www.theguardian.com/technology/2017/sep/26/tinder-personal-data-dating-app-messages-hacked-sold. [Accessed 12 October 2017].

[213] S. Nair and D. Horne, "Method and system for privacy preserving disclosure of a shared, identity linked secret". United States Patent 62/412,627, 25 October 2016.

[214] Regulation, General Data Protection, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46," *Official Journal of the European Union (OJ),* vol. 59, no. 1-88, p. 294, 2016.

[215] California Legislature, "AB-375 Privacy: personal information: businesses," *Legislative Counsel's Digest,* 28 June 2018.

[216] "GDPR Enforcement Tracker - list of GDPR fines," CMS Law . Tax, [Online]. Available: https://www.enforcementtracker.com/. [Accessed 12 March 2020].

[217] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber and J. Sutherland, "Principles behind the Agile Manifesto," 2001. [Online]. Available: https://agilemanifesto.org/principles.html. [Accessed 25 02 2020].

[218] N. Cerpa and J. M. Verner, "Why did your project fail?," *Communications of the ACM,* vol. 52, no. 12, pp. 130-134, 2009.

[219] M. H. N. Nasir and S. Sahibuddin, "Critical success factors for software projects: A comparative study," *Scientific research and essays,* vol. 6, no. 10, pp. 2174-2186, 2011.

[220] P. L. Bannerman, "Risk and risk management in software projects: A reassessment," *Journal of Systems and Software ,* vol. 81, no. 12, pp. 2118-2133, 2008.

[221] "Platform Policy - Facebook for Developers," Facebook, 2020. [Online]. Available: https://developers.facebook.com/policy. [Accessed 12 March 2020].

[222] N. Statt, "Facebook CEO Mark Zuckerberg says the 'future is private' - The Verge," Vox Media, LLC., 30 April 2019. [Online]. Available: https://www.theverge.com/2019/4/30/18524188/facebook-f8-keynote-mark-zuckerberg-privacy-future-2019. [Accessed 12 March 2020].

[223] "Sample Submission - Facebook Login - Documentation - Facebook for Developers," Facebook, 2020. [Online]. Available: https://developers.facebook.com/docs/facebook-login/review/sample-submission. [Accessed 12 March 2020].

[224] "FAQ - App Development," Facebook, 2020. [Online]. Available: https://developers.facebook.com/docs/apps/faq. [Accessed 12 March 2020].

[225] "Use testing tracks to get invaluable early feedback from users - Google Play | Android Developers," Google, 2020. [Online]. Available: https://developer.android.com/distribute/best-practices/launch/test-tracks. [Accessed 12 March 2020].

[226] "System Requirements - Xamarin | Microsoft Docs," Microsoft, 16 October 2019. [Online]. Available: https://docs.microsoft.com/en-us/xamarin/cross-platform/get-started/requirements. [Accessed 12 March 2020].

[227] "Publishing Xamarin.iOS Apps to the App Store - Xamarin | Microsoft Docs," Microsoft, 25 June 2018. [Online]. Available: https://docs.microsoft.com/en-us/xamarin/ios/deploy-test/app-distribution/app-store-distribution/publishing-to-the-app-store. [Accessed 12 March 2020].

[228] F. Nunes, P. A. Silva, J. Cevada, A. C. Barros and L. Teixeira, "User interface design guidelines for smartphone applications for people with Parkinson's disease," *Universal Access in the Information Society,* vol. 15, no. 4, pp. 659-679, 2016.

[229] F. Ghorbel, E. Métais, N. Ellouze, F. Hamdi and F. Gargouri, "Towards accessibility guidelines of interaction and user interface design for Alzheimer's disease patients," in *Tenth International Conference on Advances in Computer-Human Interactions*, 2017.

[230] H. S. A. Latiff, R. Razali and F. F. Ismail, "User interface design guidelines for children mobile learning applications," *International Journal of Recent Technology and Engineering,* vol. 8, no. 3, pp. 3311-3319, 2019.

[231] W.-C. Wang, C.-C. Chen and K.-C. Wu, "Exploring the interface design of assisting children to find books in the library using smartwatches," in *2017 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, 2017.

[232] C. Murad, C. Munteanu, L. Clark and B. R. Cowan, "Design guidelines for hands-free speech interaction," in *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, 2018.

[233] J. R. Lewis, Practical speech user interface design, CRC Press, 2016.

[234] E. McAweeney, H. Zhang and M. Nebeling, "User-driven design principles for gesture representations," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018.

[235] E. G. M. Kanaga, R. M. Kumaran, M. Hema, R. G. Manohari and T. A. Thomas, "An experimental investigations on classifiers for Brain Computer Interface (BCI) based authentication," in *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, 2017.

[236] S. Amershi, D. Weld, M. Vorvoreanu, A. Fourney, B. Nushi, P. Collisson, J. Suh, S. Iqbal, P. Bennett, K. Inkpen and J. Teevan, "Guidelines for human-ai interaction," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019.

[237] S. S. Sundar, "Rise of machine agency: A framework for studying the psychology

of human–ai interaction (HAII)," *Journal of Computer-Mediated Communication,* 2020.

[238] A. Juels and T. Ristenpart, "Honey encryption: Security beyond the brute-force bound," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2014.

[239] A. Juels and T. Ristenpart, "Honey encryption: Encryption beyond the brute-force barrier," *IEEE security & privacy,* vol. 12, no. 4, pp. 59-62, 2014.

[240] A. Omolara, A. Jantan, O. Abiodun and H. Poston, "A novel approach for the adaptation of honey encryption to support natural language message," in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, 2018.

[241] O. Hajoui, D. Rachid, M. Talea and Z. I. Batouta, "An advanced comparative study of the most promising NoSQL and NewSQL databases with a multi-criteria analysis method," *Journal of Theoretical and Applied Information Technology,* vol. 81, no. 3, p. 579, 2015.

[242] J. Han, E. Haihong, G. Le and J. Du, "Survey on NoSQL database," in *2011 6th international conference on pervasive computing and applications*, 2011.

[243] B. G. Tudorica and C. Bucur, "A comparison between several NoSQL databases with comments and notes," in *2011 RoEduNet international conference 10th edition: Networking in education and research*, 2011.

[244] S. Schmid, E. Galicz and W. Reinhardt, "WMS performance of selected SQL and NoSQL databases," in *International Conference on Military Technologies (ICMT) 2015*, 2015.

[245] B. Zantout and R. Haraty, "I2P data communication system," in *Proceedings of ICN*, 2011.

[246] A. Kwon, D. Lazar, S. Devadas and B. Ford, "Riffle," in *Proceedings on Privacy Enhancing Technologies*, 2016.

[247] N. Nipane, I. Dacosta and P. Traynor, ""Mix-in-Place" anonymous networking using secure function evaluation," in *Proceedings of the 27th Annual Computer Security Applications Conference*, 2011.

[248] H. An, R. Choi, J. Lee and K. Kim, "Performance evaluation of liboqs in open quantum safe project (part I)," in *2018 Symposium on Cryptography and Information Security (SCIS 2018)*, 2018.

[249] C. H. Bennett and G. Brassard, "Quantum cryptography: public key distribution and coin tossing," in *Conf. on Computers, Systems and Signal Processing*, Bangalore, India, 1984.

[250] A. K. Ekert, "Quantum cryptography based on Bell's theorem," *Physical review letters,* vol. 67, no. 6, p. 661, 1991.

[251] P. W. Shor and J. Preskill, "Simple proof of security of the BB84 quantum key distribution protocol," *Physical review letters,* vol. 85, no. 2, p. 441, 2000.

[252] M. Ben-Or, C. Crépeau, D. Gottesman, A. Hassidim and A. Smith, "Secure

multiparty quantum computation with (only) a strict honest majority," in *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, 2006.

[253] C. Crépeau, D. Gottesman and A. Smith, "Secure multi-party quantum computation," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, 2002.

[254] H.-Y. Jia, Q.-Y. Wen, T.-T. Song and F. Gao, "Quantum protocol for millionaire problem," *Optics Communications,* vol. 284, no. 1, pp. 545-549, 2011.

[255] X.-B. Chen, G. Xu, Y.-X. Yang and Q.-Y. Wen, "An efficient protocol for the secure multi-party quantum summation," *International Journal of Theoretical Physics,* vol. 49, no. 11, pp. 2793-2804, 2010.

[256] W. Liu, Y.-B. Wang and W.-Q. Fan, "An novel protocol for the quantum secure multi-party summation based on two-particle bell states," *International Journal of Theoretical Physics,* vol. 56, no. 9, pp. 2783-2791, 2017.

[257] Z. Ji, H. Zhang, H. Wang, F. Wu, J. Jia and W. Wu, "Quantum protocols for secure multi-party summation," *Quantum Information Processing,* vol. 18, no. 6, pp. 168-186, 2019.

[258] D. Mayers, "Unconditionally secure quantum bit commitment is impossible," *Physical review letters,* vol. 78, no. 17, p. 3414, 1997.

[259] H.-K. Lo and H. F. Chau, "Is quantum bit commitment really possible?," *Physical Review Letters,* vol. 78, no. 17, p. 3410, 1997.

[260] Y.-G. Yang, Y.-W. Teng, H.-P. Chai and Q.-Y. Wen, "Verifiable quantum (k, n)-threshold secret key sharing," *International Journal of Theoretical Physics,* vol. 50, no. 3, pp. 792-798, 2011.

[261] H. Barnum, C. Crépeau, D. Gottesman, A. Smith and A. Tapp, "Authentication of quantum messages," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

[262] Z. Tai, "Casting the ubiquitous net of information control: Internet surveillance in China from golden shield to green dam," *International Journal of Advanced Pervasive and Ubiquitous Computing (IJAPUC),* vol. 2, no. 1, pp. 53-70, 2010.

[263] Y. Zhang, S. Zhang, Y. Zhang, J. Tao and P. Wang, "A large-scale empirical study of Internet users' privacy leakage in China," in *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, 2019.

[264] A. Lv and T. Luo, "Authoritarian practices in the digital age| Asymmetrical power between Internet giants and users in China," *International Journal of Communication,* vol. 12, pp. 3877-3895, 2018.

[265] Y. Chen and A. S. Cheung, "The transparent self under big data profiling: Privacy and Chinese legislation on the social credit system," *The Journal of Comparative Law,* vol. 12, no. 2, pp. 356-378, 2017.

[266] G. Z. Jin, "Artificial intelligence and consumer privacy," National Bureau of Economic Research, 2018.

[267] X. Li and T. Zhang, "An exploration on artificial intelligence application: From security, privacy and ethic perspective," in *2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, 2017.

[268] S. Wilson, S. Ghanavati, K. Ghazinour and N. Sadeh, "Privacy-enhancing artificial intelligence and language technologies (PAL 2019): Preface to the proceedings," in *CEUR Workshop Proceedings*, 2019.

[269] S. Yu, "Big privacy: Challenges and opportunities of privacy study in the age of big data," *IEEE Access,* vol. 4, pp. 2751-2763, 2016.