ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Συστήματα Ψηφιακών Ανώνυμων Ερωτηματολογίων και Κρυπτογραφικές Κυκλικές Υπογραφές

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΔΗΜΗΤΡΙΟΣ-ΣΤΥΛΙΑΝΟΣ ΚΟΛΟΝΕΛΟΣ

Επιβλέπων : Αριστείδης Παγουρτζής
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2018

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Συστήματα Ψηφιακών Ανώνυμων Ερωτηματολογίων και Κρυπτογραφικές Κυκλικές Υπογραφές

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΔΗΜΗΤΡΙΟΣ-ΣΤΥΛΙΑΝΟΣ ΚΟΛΟΝΕΛΟΣ

Επιβλέπων : Αριστείδης Παγουρτζής
Αν. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 20η Ιουλίου 2018.

....	....	....
Αριστείδης Παγουρτζής	Παναγιώτης Τσανάκας	Αντώνιος Συμβώνης
Αν. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2018

....

Δημήτριος-Στυλιανός Κολονέλος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή κ.Παγουρτζή για τη βοήθεια και καθοδήγηση του καθ’ όλη τη διάρκεια εκπόνησης αυτής της Διπλωματικής και για την ευκαιρία που μου έδωσε να ασχοληθώ με την Κρυπτογραφία. Ακόμα, ευχαριστώ τους Υποψήφιους Διδάκτορες Παναγιώτη Γροντά και Αλέξανδρο Ζαχαράκη για το ενδιαφέρον τους και τις εποικοδομητικές συζητήσεις. Τέλος, θα ήθελα να ευχαριστήσω την οικόγενεια μου για τη συνεχή τους στήριξη και τους φίλους μου για όσα περάσαμε στα εύκολα και στα δύσκολα.

Δημήτριος-Στυλιανός Κολονέλος,

Αθήνα, 20η Ιουλίου 2018

Περίληψη

Η ραγδαία ανάπτυξη της Ψηφιακής Τεχνολογίας τις τελευταίες δεκαετίες έχει οδηγήσει στη μεταπήδηση πολλών φυσικών ενεργειών στον Ψηφιακό Κόσμο. Πολλές από αυτές τις ενέργειες περιλαμβάνουν αποστολή δεδομένων από ένα συμβαλλόμενο σε κάποιον άλλον πιστοποιημένα. Ένα παράδειγμα είναι τα Ηλεκτρονικά Ανώνυμα Ερωτηματολόγια, όπου κάποιος φτιάχνει κάποιες ερωτήσεις και ζητάει απαντήσεις από τους χρήστες. Σημαντικό σε αυτό είναι ότι θέλει συγκεκριμένοι χρήστες να απαντήσουν. Η Πιστοποιησιμότητα διασφαλίζει ότι μόνο οι επιθυμητοί χρήστες καταθέτουν απαντήσεις. Επίσης, η Ιδιωτικότητα έχει έρθει στο προσκήνιο τα τελευταία χρόνια και ειδικά στον τομέα της κρυπτογραφικής έρευνας. Για το λόγο αυτό, είναι απαραίτητο τα δεδομένα που στέλνονται να είναι ανώνυμα. Ο στόχος των Συστημάτων Ανώνυμων Ερωτηματολογίων είναι να επιλύσουν κρυπτογραφικά τη σύγκρουση μεταξύ Ανωνυμίας και Πιστοποιησιμότητας στις καταθέσεις δεδομένων, χωρίς να υποθέτουν έμπιστες τρίτες αρχές.

Στην παρούσα διπλωματική εργασία μελετάμε το πρόβλημα της κατασκευής Συστημάτων Ανώνυμων Ερωτηματολογίων. Στην άρχη μελετάμε σε βάθος το σύστημα Anonize ένα πλήρες σύστημα κρυπτογραφικών Συστημάτων Ad-Hoc Ερωτηματολογίων. Μετά κάνουμε κάποιες παρατηρήσεις πάνω στο Anonize που μας δίνουν κίνητρο να προτείνουμε νέα Συστημάτα Ad-Hoc Ερωτηματολογίων βασισμένα σε Κυκλικές Υπογραφές.

Μελετάμε τις Ανιχνεύσιμες Κυκλικές Υπογραφές και προτείνουμε ένα τροποποιημένο κρυπτογραφικό αρχέτυπο αλλάζοντες τες ώστε να επιτρέπουν Δυναμικό σχηματισμό ομάδων. Επίσης, αποδεικνύουμε την ασφάλεια των Δυναμικών Ανιχνεύσιμων Κυκλικών Υπογραφών. Μετά προτείνουμε ένα Σύστημα Ad-Hoc Ερωτηματολογίων βασισμένο σε αυτό το αρχέτυπο με σκοπό να αποτρέψουμε μια επίθεση στην Πιστοποιησιμότητα. Στη συνέχεια, μελετάμε τις Μικρές Συνδέσιμες Κυκλικές Υπογραφές και προτείνουμε ένα νέο Σύστημα Ad-Hoc Ερωτηματολογίων βασιμένο σε αυτές το οποίο, επίσης, αποτρέπει την επίθεση στην Πιστοποιησιμότητα και είναι πιο αποδοτικό. Τέλος, παρουσιάζουμε σύντομα μια ιδέα για προσθήκη Αδυναμίας-Απόδειξης στα Συστήματα Ad-Hoc Ερωτηματολογίων, λαμβάνοντας κάποιες υποθέσεις.

Λέξεις κλειδιά

Κρυπτογραφία, Ανώνυμα Ερωτηματολόγια, Ad-Hoc Ερωτηματολόγια, Κυκλικές Υπογραφές, Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές, Μικρές Συνδέσιμες Κυκλικές Υπογραφές, Αδυναμία-Απόδειξης

Abstract

Rapid growth of digital technology in the last few decades has led many acts to cross over from physical to digital world. Much of these acts involve data sending from one party to another in an authenticated manner. An example is electronic Anonymous Questionnaires, where a party creates some Questions and requests answers from users. Key to this is that she wants specific users to answer. Authenticity ensures that only intended users Submit answers. Additionally, privacy has gained much attention in last years, especially in cryptographic research field. Thus, it is necessary that data sending is anonymous. The aim of Anonymous Survey Systems is to solve cryptographically the conflict between Anonymity and Authenticity in data submissions, without considering trusted third parties.

In this thesis we study the problem of constructing Anonymous Survey Systems. At first we study, in depth, Anonize a fully-featured cryptographic Ad-Hoc Survey System. Then we make some observations on Anonize that motivate us to make new Ad-Hoc Systems proposals based on Ring Signatures.

We study Traceable Ring Signatures and we propose an altered cryptographic primitive by modifying them to allow Dynamic formation of groups. We, also, provide proofs of security for Dynamic Traceable Ring Signatures. Then we propose an Ad-Hoc Survey System based on this primitive with intention to prevent an authenticity attack. Afterwards, we study Short Linkable Ring Signatures and propose a new Ad-Hoc Survey Scheme based on them that, also, prevents the authenticity attack and is more efficient. Finally, we briefly present an idea to add Receipt-Freeness on Ad-Hoc Survey Schemes, considering some assumptions.

Key words

Cryptography, Anonymous Surveys, Ad-Hoc Surveys, Ring Signatures, Dynamic Traceable Ring Signatures, Short Linkable Ring Signatures, Receipt-Freeness

Contents

Part II English text
Anonymous Digital Survey Systems and Cryptographic Ring Signatures 51

List of Figures

Part I

Ελληνικό Κείμενο

Κεφάλαιο 1

Εισαγωγή

1.1 Ανώνυμα Ψηφιακά Ερωτηματολόγια

Ένα πρόβλημα που εμφανίζεται σε ένα τεράστιο εύρος εφαρμογών, σήμερα, είναι η συλογή ανώνυμων δεδομένων από ένα επιλεγμένο σύνολο ανθρώπων. Οι βασικοί στόχοι σε αυτές τις περιπτώσεις είναι η Ανωνυμία και η Πιστοποιησιμότητα. Δηλαδή, ότι ο συλλέκτης των δεδομένων δεν μπορεί να συνδέσει τα δεδομένα που περισυνέλεξε με κάποιο πρόσωπο και παράλληλα είναι βέβειος ότι μόνο οι στοχευμένοι άνθρωποι κατέθεσαν δεδομένα. Από εδώ και στο εξής θα αναφερόμαστε στα δεδομένα που ζητούνται ως ερωτηματολόγια και στα δεδομένα που δίνονται ως κατάθεση ερωματολογίου/απάντησης, ακόμα κι αν δεν πρόκειται για ερωτηματολόγια με την κλάσσικη έννοια.

Το σύνηθες στην πραγματική ζωή για τα ερωτηματολόγια είναι ο χρήστης (επώνυμα) να παραδίδει την απάντησή του στον συλλέκτη ερωτηματολογίων και ο δεύτερος να υπόσχεται ότι θα διατηρήσει την ανωνυμία του χρήστη. Αυτό το σενάριο υποθέτει έμπιστο συλλέκτη ερωτηματολογίων. Φυσικά, σε πολλές περιπτώσεις η ”υπόσχεση” συνοδεύεται από νομικές δεσμεύσεις. Παρόλα αυτά και πάλι μπορεί να έχουμε λόγους να μην τον εμπιστευόμαστε. Για παράδειγμα, ακόμα κι αν έχει πρόθεση να είναι τίμιος, μπορεί κι ο ίδιος να πέσει θύμα κάποιας υποκλοπής και να γίνει διαρροή των δεδομένων μας.

Σε γενικές γραμμές, αυτός είναι ο λόγος που χτίζουμε κρυπτογραφικά ασφαλή συστήματα χωρίς να υποθέτουμε έμπιστες τρίτες αρχές. Για το λόγο αυτό κατανοούμε ότι το πρόβλημα των Ανώνυμων Ερωτηματολογίων βρίσκεται στο αντικείμενο της Κρυπτογραφίας.

Μία εργασία που έχει να κάνει με το πρόβλημα αυτό είναι το Anonize, που εισήχθη το 2014 από τους Hohenberger, Myers, Pass και shelat [6]. Αυτό είναι ένα περιβάλλον που επιτρέπει σε αυτούς που ανοίγουν ένα Ερωτηματολόγιο να ζητήσουν συγκεκριμένα ανώνυμα δεδομένα από συγκεκριμένους χρήστες και στους χρήστες να αποστέλουν την απάντησή τους ανώνυμα. Το Anonize, εκτός από σπουδαία θεωρητική μελέτη στο πρόβλημα των Ανώνυμων Ερωτηματολογίων, είναι και ένα σύστημα που έχει υλοποιηθεί. Έχει υλοποιηθεί και δοκιμαστεί σε πραγματικές συνθήκες και αποτελεί ένα έμπιστο εργαλείο για πρακτικά προβλήματα. Επίσης, σε γνώση του συγγραφέα, είναι το μόνο πρακτικό εργαλείο το οποίο είναι ταυτόχρονα θεωρητικά θεμελιωμένο το οποίο έχει να κάνει με Ανώνυμα Ερωτηματολόγια. Οπότε, επηρέασε την Διπλωματική αυτή εργασία έντονα.

1.2 Σχέση με Ψηφιακές Ψηφοφορίες

Ένα πρόβλημα το οποίο σχετίζεται έντονα με τον κλάδο της Κρυπτογραφίας είναι η κατασκευή ασφαλών συστημάτων Ψηφιακών Ψηφοφοριών. Αυτό έχει να κάνει με την κατασκευή πρωτοκόλλων και εργαλείων που επιτρέπουν στους χρήστες να ψηφίσουν με ασφάλεια μέσω ηλεκτρονικών συσκευών. Αυτό είναι ένα πρόβλημα που, σε αντίθεση με τα ανώνυμα ερωτηματολόγια, έχει μελετηθεί εκτενώς από τους ερευνητές της κρυπτογραφίας και ακόμα συνεχίζει να μελετάται.

Το πρόβλημα των ψηφιακών ψηφοφοριών είναι παρόμοιο με το πρόβλημα των Ανώνυμων Ερωτηματολογίων. Μπορούμε να σκεφτόμαστε την κατάθεση ανώνυμων ερωτηματολογίων σαν ψηφοφορία αλλά με πιο περίπλοκη ψήφο. Οι βασικές απαιτήσεις είναι και πάλι Ανωνυμία και Πιστοποιησιμότητα. Το μοντέλο αυτό, όμως, είναι αρκετά απλό για τις εκλογές αφού αυτές έχουν πολύ περισσότερες απαιτήσεις ασφαλείας. Παρόλα αυτά η γενική ιδέα είναι παρόμοια.

Στην πραγματικότητα η παρούσα εργασία εκκινεί από τα συστήμα Ψηφιακών Ψηφοφοριών και πιο συγκεκριμένα από τη δουλειά των Pagourtzis, Grontas, Zacharakis and Zhang [7, 8, 9] που μελετάει το πρόβλημα των Ψηφιακών Ψηφοφοριών με έμφαση στην τέλεια ιδιωτικότητα και την άμυνα στις επιθέσεις εξαναγκασμού.

Από την άλλη, οι Ψηφιακές Ψηφοφορίες και τα ηλεκτρονικά Ανώνυμα Ερωτηματολόγια έχουν διαφορές. Το βασικό εμπόδιο για την άμεση εφαρμογή των υπάρχοντων συστημάτων ψηφιακών ψηφοφοριών σε ερωτηματολόγια έχει να κάνει με το γεγονός ότι συνήθως τα πρώτα αποτελούνται από δύο διακριτές φάσεις. Η πρώτη φάση, όπου χρήστης επώνυμα παραλαμβάνει ένα ανώνυμο credential που θα χρησιμοποιήσει για να ψηφίσει και η δεύτερη, η φάση της ψήφισης, όπου το ανώνυμο credential χρησιμοποιείται για να καταθέσει την ψήφο ο χρήστης. Αυτές οι φάσεις διαχωρίζονται από ένα μεγάλο χρονικό διάστημα, για παράδειγμα μίας μέρας. Ο χρόνος αυτός εμποδίζει τις επιθέσεις συσχέτισης-χρόνων μεταξύ παραλαβής του credential και κατάθεσής του μαζί με την ψήφο. Αυτό θα αποανωνυμοποιούσε το χρήστη.

Παρόλο που το παραπάνω δουλεύει για τα συστήματα Ψηφιακών Ψηφοφοριών, θα ήταν μη πρακτικό να εφαρμόσουμε τόσο μεγάλο χρονικό διάστημα μεταξύ των δύο φάσεων στην Κατάθεση Ανώνυμων Ερωτηματολογίων. Κι αυτό διότι δε θα ήταν πολύ πιθανό ο χρήστης να περιμένει τόσο πολύ χρόνο για να καταθέσει ερωτημαλόγιο. Ο λόγος είναι ότι οι εκλογές είναι κάτι που θεωρείται σημαντικό από αυτούς που ψηφίζουν. Από την άλλη τα ερωτηματολόγια είναι κάτι που οι άνθρωποι, εν γένει, δεν παίρνουν τόσο στα σοβαρά ή είναι ακόμα και απρόθυμοι να συμπληρώσουν. Οπότε το να περιμένουν π.χ. μία ημέρα για να καταθέσουν απάντηση είναι μη πρακτικό. Τέλος, οι εκλογές συνήθως έχουν περιορισμό στις ώρες που μπορεί κάποιος να ψηφίσει (π.χ. 7 το πρωί με 7 το βράδυ), έτσι η δεύτερη φάση μπορεί εύκολα να διαχωριστεί από την πρώτη χρονικά. Από την άλλη, τα ερωτηματολόγια μπορεί να έχουν μεγάλο χρονικό διάστημα συλλογής, για παράδειγμα 3 μήνες, κάτι που καθιστά το διαχωρισμό σε 2 φάσεις δύσκολο. Η παρατήρηση αυτή έγινε στην εργασία του Anonize [6].

Τέλος, οι εκλογές έχουν μεγάλο κοινωνικό αντίκτυπο. Για το λόγο αυτό, τα συστήματα Ψηφιακών Ψηφοφοριών οφείλουν να έχουν αναβαθμισμένες ιδιότητες ασφαλείας. Αντίθετα, στα Συστήματα Ανώνυμων Ερωτηματολογίων μπορεί να έχουμε λιγότερες ιδιότητες ασφαλείας και αυτό να είναι αποδεκτό.

Για όλους αυτούς τους λόγους, υπάρχει ανάγκη για κατασκευή Συστημάτων Ψηφιακών Ανώνυμων Ερωτηματολογίων, ανεξάρτητα από τα συστήματα Ψηφιακών Ψηφοφοριών.

Κεφάλαιο 2

Βασικές Κρυπτογραφικές έννοιες

2.1 Ψευδοτυχαίες συναρτήσεις

Μία ψευδοτυχαία συνάρτηση είναι ένα σύνολο συναρτήσεων $\{F_k : \{0,1\}^n \to \{0,1\}^n\}$. Με την επιλογή ενός τυχαίου σπόρου k επιλέγεται μία συνάρτηση από το σύνολο η οποία είναι ψευδοτυχαία. Δηλαδή, κανείς (PPT) δε μπορεί να ξεχωρίσει αν μία συναρτηση $f : \{0,1\}^n \to \{0,1\}^n$ επιλέχθηκε τυχαία από το χώρο των συναρτήσεων ή από το σύνολο $\{F_k\}$ μέσω ενός τυχαίου σπόρου.

Definition 2.1. *Έστω συνάρτηση $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ εύκολα υπολογίσιμη. Λέμε ότι η F είναι ψευδοτυχαία αν για κάθε PPT διαχωριστή D υπάρχει μία αμελητέα συνάρτηση $negl(\cdot)$ τ.ω.:*

$$\left| Pr[D^{F_k(\cdot)}(1^n) = 1] - Pr[D^{f_n(\cdot)}(1^n) = 1] \right| \leq negl(n)$$

όπου το k επιλέχθηκε ομοιόμορφα τυχαία από το $\{0,1\}^n$ και η f_n επιλέχθηκε τυχαία από το σύνολο των $\{0,1\}^n \to \{0,1\}^n$ συναρτήσεων.

Ο διαχωριστής πολυωνυμικού χρόνου D δεν μπορεί να δεχτεί σαν είσοδο τις συναρτήσεις διότι αυτό θα απαιτούσε $n \cdot 2^n$ -bit είσοδο και εκθετικά πολλούς ελέγχους. Για το λόγο αυτό έχει oracle access στις συναρτήσεις.

2.2 Σχήματα δέσμευσης

Τα σχήματα δέμσευσης επιτρέπουν σε κάποιον να δεσμευτεί σε μία τιμή διατηρώντας τη κρυφή, εώς ότου την αποκαλύψει σε μία μεταγενέστερη στιγμή. Τα σχήματα δέσμευσης έχουν δύο φάσεις και αφορούν δύο συμβαλλόμενους: τον αποστολέα S και τον παραλήπτη R [18]:

- **Φάση Δέσμευσης:** Ο αποστολέας δεσμεύεται σε μία τιμή m και στέλνει το κρυπτοκείμενο δέσμευσης $c \leftarrow Com(m)$ στον παραλήπτη.
- **Φάση Αποκάλυψης:** Ο αποστολέας "ανοίγει" το κρυπτοκείμενο δέσμευσης και αποκαλύπτει την τίμη δέσμευσης m, στέλνοντας την στον παραλήπτη. Ο παραλήπτης ελέγχει αν όντως το c είναι κρυπτοκείμενο δέσμευσης για το m.

Μία αναλογία από το φυσικό κόσμο σε αυτό είναι: ο αποστολέας δεσμεύεται σε ένα αντικείμενο κλειδώνοντάς το σε ένα κουτί και στέλνει το κουτί στον παραλήπτη. Όταν έρθει η ώρα της αποκάλυψης ξεκλειδώνει το κουτί και αποκαλύπτεται η αρχική του δέσμευση.

Δύο απαραίτητα χαρακτηριστικά στο σχήμα αυτό είναι ότι ο αποστολέας θέλει να παραμένει κρυφή η τιμή δέσμευσης του μέχρι να αποφασίσει να την αποκαλύψει και ότι όταν την αποκαλύψει, ο παραλήπτης θέλει να είναι σίγουρος πως πρόκειται όντως για την τιμή στην οποία έχει δεσμευτεί. Η πρώτη ιδιότητα ονομάζεται Απόκρυψη (hiding) και η δεύτερη Δέσμευση (Binding).

Definition 2.2. *Το* $\Pi = (Gen, Com, Ver)$ *έχει:*

1. *Τέλεια Απόκρυψη αν τα δύο probability ensembles* $\{Com_{ck}(m_0)\}, \{Com_{ck}(m_1)\}$ *ταυτίζονται.*

2. *Στατιστική Απόκρυψη αν τα δύο probability ensembles* $\{Com_{ck}(m_0)\}, \{Com_{ck}(m_1)\}$ *είναι στατιστικά κοντά.*

3. *Υπολογιστική Απόκρυψη αν τα δύο probability ensembles* $\{Com_{ck}(m_0)\}, \{Com_{ck}(m_1)\}$ *είναι υπολογιστικά μη-διακρίσιμα.*

Για την ιδιότητα της Δέσμευσης ορίζουμε το επόμενο πείραμα:

Πείραμα Δέσμευσης $Bind_{\mathcal{A},\Pi}(n)$

1) $ck \leftarrow Gen(1^n)$

2) $(com, m_0, m_1, d_0, d_1) \leftarrow \mathcal{A}(ck)$

$$output = \begin{cases} 1 & \text{αν } Ver(com, m_0, d_0) = Ver(com, m_1, d_1) = 1 \text{ και } m_0 \neq m_1 \\ 0 & \text{αλλιώς} \end{cases}$$

Definition 2.3. *Το* $\Pi = (Gen, Com, Ver)$ *έχει:*

1. *Τέλεια Δέσμευση αν για κάθε υπολογιστικά απεριόριστο αντίπαλο* $\mathcal{A}$*:*
   $Bind_{\mathcal{A},\Pi}(n) = 0$.

2. *Στατιστική Δέσμευση αν για κάθε υπολογιστικά απεριόριστο αντίπαλο* $\mathcal{A}$*:*
   $Bind_{\mathcal{A},\Pi}(n) \leq negl(n)$.

3. *Υπολογιστική Δέσμευση αν για κάθε PPT αντίπαλο* $\mathcal{A}$*:* $Bind_{\mathcal{A},\Pi}(n) \leq negl(n)$.

Definition 2.4. *Το* $\Pi = (Gen, Com, Ver)$ *είναι ένα σχήμα δέσμευσης αν είναι ορθό και έχει τις ιδιότητες: υπολογιστική απόκρυψη και υπολογιστική δέσμευση. Επίσης, οι αλγόριθμοι* Gen, Com *και* Ver *πρέπει να είναι PPT.*

2.3 Ψηφιακές Υπογραφές

Ένα σχήμα ψηφιακών υπογραφών [21] είναι ένα κρυπτογραφικό αρχέτυπο που επιτρέπει στο χρήστη να πιστοποιήσει ένα μήνυμα της επιλογής του. Διαισθητικά, μία ψηφιακή υπογραφή λειτουργεί όπως μία φυσική χειρόγραφη υπογραφή. Εκτός από πιστοποίηση της ταυτότητας του μηνύματος, οι ψηφιακές υπογραφές, επίσης, προστατεύουν την ακεραιότητα του μηνύματος: ο χρήστης υπογράφει ένα συγκεκριμένο μήνυμα. Προφανώς, ο στόχος όσον αφορά την ασφάλεια είναι ίδιος με τις φυσικές υπογραφές: μόνο ο υπογράφων μπορεί να δημιουργήσει μία υπογραφή και όλοι μπορούν να την επαληθεύσουν.

Definition 2.5. *Ένα σχήμα Ψηφιακών Υπογραφών είναι μία τριάδα PPT αλγορίθμων* $(Gen, Sign, Ver)$ *ως εξής:*
Έστω $\mathcal{M}$ *ο χώρος των μηνυμάτων.*

- Gen *είναι ο αλγόριθμος παραγωγής κλειδιών, ο οποίος παίρνει σαν είσοδο την παράμετρο ασφαλείας* 1^n *και δίνει στην έξοδο ένα ζεύγος κλειδιών: το μυστικό (ιδιωτικό) κλειδί και το αντίστοιχο δημόσιο κλειδί:* $(sk, vk) \leftarrow Gen(1^n)$.
- $Sign$ *είναι ο αλγόριθμος υπογραφής, ο οποίος παίρνει σαν είσοδο ένα μήνυμα (από το* $\mathcal{M}$*) και παράγει μία υπογραφή στο μήνυμα χρησιμοποιώντας το ιδιωτικό κλειδί:* $\sigma \leftarrow Sign_{sk}(m)$
- Ver *είναι ο αλγόριθμος επαλήθευσης υπογραφής, ο οποίος παίρνει σαν είσοδο μία υπογραφή και ένα μήνυμα και ελέγχει αν η υπογραφή είναι σωστή χρησιμοποιώντας το δημόσιο κλειδί. Τέλος, δίνει στην έξοδο 0 ή 1:* $Ver_{vk}(m, \sigma) \in \{0, 1\}$

Ορθότητα: Απαιτούμε για κάθε n *και για κάθε* $m \in \mathcal{M}$ *να ισχύει με συντριπτική πιθανότητα ότι:* $Ver_{vk}(m, Sign_{sk}(m)) = 1$

Ασφάλεια σχημάτων Ψηφιακών Υπογραφών. Η ιδιότητα ασφαλείας οπου θέλουμε να ισχύει είναι η δυσκολία-πλαστογράφησης (Unforgeability). Και αυτό να ισχύει ακόμα κι αν ο αντίπαλος έχει δει υπογραφές σε άλλα μηνύματα. 3 είδη επίθεσης πλαστογράφησης υπάρχουν:

- Universal Forgery: Ο αντίπαλος μπορεί να παράξει υπογραφή σε οποιοδήποτε μήνυμα της επιλογής του.
- Selective Forgery: Ο αντίπαλος μπορεί να παράξει υπογραφή σε ένα μήνυμα της επιλογής του.
- Existential Forgery: Ο αντίπαλος μπορεί να παράξει υπογραφή σε ένα τυχαίο μήνυμα (πιθανώς χωρίς ουσιαστικό περιεχόμενο).

Εμείς επιθυμούμε την πιο ισχυρή έννοια Unforgeability η οποία αμύνεται στην επίθεση Existential Forgery (άρα και στις άλλες δύο). Η ιδιότητα αυτή λέγεται Existential Unforgeability, τον ορισμό της οποίας δίνουμε παρακάτω [22].

Πείραμα Existential Unforgeability $Sig-forge^{EUF-CMA}_{\mathcal{A},\Pi}(n)$

1) $(sk, vk) \leftarrow Gen(1^n)$: ο $\mathcal{A}$ παραλαμβάνει το vk.
2) ο $\mathcal{A}$ έχει oracle access στο $Sign_{sk}(\cdot)$ με οποιοδήποτε μήνυμα της επιλογής του m.
3) $(m, \sigma) \leftarrow \mathcal{A}^{Sign_{sk}(\cdot)}(vk)$: τελικά ο $\mathcal{A}$ επιλέγει ένα μήνυμα m και δίνει στην έξοδο μία πλαστογραφημένη υπογραφή στο m που ευελπιστεί να είναι έγκυρη.

έξοδος: Έστω Q το σύνολο των μηνυμάτων που ο $\mathcal{A}$ έκανε query στο $Sign_{sk}(\cdot)$ oracle:

$$output = \begin{cases} 1 & \text{αν } m \notin Q \text{ και } Ver_{vk}(m, \sigma) = 1 \\ 0 & \text{αλλιώς} \end{cases}$$

Definition 2.6. *Ένα σχήμα Ψηφιακν Υπογραφών* $\Pi = (Gen, Sign, Ver)$ *έχει existential Unforgeability κάτω από adaptive chosen-message attack (EUF-CMA) αν για κάθε PPT αντίπαλο* $\mathcal{A}$*, υπάρχει μία αμελητέα συνάρτηση* $negl(n)$ *ώστε:*

$$Pr[Sig - forge_{\mathcal{A},\Pi}^{EUF-CMA}(n) = 1] \leq negl(n)$$

2.3.1 Τυφλές Υπογραφές

Ένα σχήμα τυφλών υπογραφών (Chaum [23]) επιτρέπει την πιστοποίηση ενός μηνύματος διατηρώντας το μήνυμα κρυφό. Ένας χρήστης $\mathcal{U}$ ζητάει πιστοποίηση ενός μηνύματος m από μία αρχή (υπογράφων $\mathcal{S}$), αλλά δε θέλει ο $\mathcal{S}$ να δει το μήνυμα. Οπότε, το τυφλώνει (blinding), καθιστώντας το κρυφό, και το στέλνει στην αρχή, η οποία το υπογράφει στα τυφλά. Αφού λάβει την τυφλή υπογραφή σ' ο $\mathcal{U}$, την αποτυφλώνει και παίρνει μία έγκυρη υπογραφή σ στο αρχικό μήνυμα m.

Definition 2.7. *Ένα σχήμα τυφλών υπογραφών είναι μία τριάδα αλγορίθμων* $(Gen, Sign, Ver)$ *ως εξής:*

- Gen *είναι ο αλγόριθμος παραγωγής κλειδιού, ο οποίος παίρνει στην είσοδο του την παράμετρο ασφαλείας* 1^n *και δίνει στην έξοδο ένα ζευγάρι κλειδιών, το μύστικο (ιδιωτικό) κλειδί και το αντίστοιχο δημόσιο κλειδί:* $(sk, vk) \leftarrow Gen(1^n)$.

- $Sign$ *είναι ένα πρωτόκολλο μεταξύ του χρήστη* $\mathcal{U}$ *και του υπογράφοντα* $\mathcal{S}$ *με κοινή είσοδο το* vk*. Η ιδιωτική είσοδος του* $\mathcal{U}$ *είναι ένα μήνυμα* m *και η ιδιωτική είσοδος του* $\mathcal{S}$ *είναι το μυστικό κλειδί* sk*. Στο τέλος του πρωτοκόλλου ο* $\mathcal{U}$ *έχει αποκτήσει μία υπογραφή* σ *στο* m *σαν ιδιωτική έξοδο.* $\sigma \leftarrow \langle \mathcal{S}(vk, sk), \mathcal{U}(vk, m) \rangle$

- Ver *είναι ο αλγόριθμος επαλήθευσης της υπογραφής ο οποίος δέχεται σαν είσοδο μία υπογραφή και ένα μήνυμα και ελέγχει αν η υπογραφή στο μήνυμα είναι ορθή με το δημόσιο κλειδί (και δίνει έξοδο 0 ή 1):* $Ver_{vk}(m, \sigma) \in \{0, 1\}$

Ο παραπάνω ορισμός του $Sign$ υπονοεί ότι ο χρήστης $\mathcal{U}$ χρησιμοποιεί ένα αλγόριθμο $Blind$ για να διατηρήσει το μήνυμα m κρυφό και έναν αλγόριθμο $Unblind$ για να μετατρέψει την υπογραφή σ' που παραλαμβάνει $\sigma \leftarrow Unblind(m, \sigma')$ και να διατηρήσει το σ ως ιδιωτική έξοδο. Πράγματι, άλλοι ορισμοί στη βιβλιογραφία ορίζουν τα σχήματα τυφλών υπογραφών ως μία πεντάδα αλγορίθμων $(Gen, Blind, Sign, Unblind, Ver)$.

Ασφάλεια σχημάτων τυφλών υπογραφών. Φυσικά, η απαίτηση για δυσκολία-πλαστογράφησης (Unforgeability) των "κλασσικών" υπογραφών παραμένει. Στην περίπτωση των τυφλών υπογραφών, όμως, ο αντίπαλος κάνει $Sign$ oracle queries σε μηνύματα που δε γνωρίζουμε καθώς είναι τυφλά. Έτσι, όταν στο τέλος καταθέσει μία υπογραφή σε ένα μήνυμα m δεν ξέρουμε αν κατάφερε να την πλαστογραφήσει ή την είχε ζητήσει από το oracle. Οπότε ο ορισμός του Unforgeability Experiment (Pointcheval και Stern [31]) απαιτεί αν ο αντίπαλος έκανε ℓ $Sign$ oracle queries να καταθέσει στο τέλος (τουλάχιστον) $\ell + 1$ υπογραφές. Παρακάτω δίνεται ο ορισμός της ιδιότητας [29, 32, 33].

Πείραμα one-more-forgery $Sig-onemoreforge_{\mathcal{A},\Pi}(n)$

1) $(sk, vk) \leftarrow Gen(1^n)$: ο $\mathcal{A}$ παραλαμβάνει το vk.

2) ο $\mathcal{A}(pk)$ συμμετέχει $\ell = poly(n)$ φορές στο διαδραστικό πρωτόκολλο $Sign$ με τον $\mathcal{S}$, όπου ο $\mathcal{A}$ αποφασίζει προσαρμοστικά πότε θα σταματήσει.

3) $\{(m_1,\sigma_1),(m_2,\sigma_2),...(m_k,\sigma_k)\} \leftarrow \mathcal{A}(vk)$

$$output = \begin{cases} 1 & \text{αν } m_i \neq m_j,\ \forall i \neq j \text{ και } Ver_{vk}(m_i,\sigma_i) = 1,\ \forall i \text{ και } k > \ell \\ 0 & \text{αλλιώς} \end{cases}$$

Definition 2.8. *Ένα σχήμα τυφλών υπογραφών* $\Pi = (Gen, Sign, Ver)$ *είναι unforgeable αν για κάθε PPT αντίπαλο* $\mathcal{A}$*, υπάρχει μία αμελητέα συνάρτηση* $negl(n)$ *τ.ω.:*

$$Sig-onemoreforge_{\mathcal{A},\Pi}(n) \leq negl(n)$$

Μία άλλη ιδιότητα που θέλουμε να τηρείται είναι η τυφλότητα (Blindness). Αυτό σημαίνει ότι η αρχή υπογραφής $\mathcal{S}$ δεν μπορεί να καταλάβει τι μήνυμα υπογράφει. Η ιδιότητα αυτή μερικές φορές αναφέρεται στη βιβλιογραφία και ως Unlinkability και ο ορισμός της δίνεται παρακάτω [32]:

Πείραμα Blinding $Blind_{\mathcal{A},\Pi}(n)$

1) $(vk, m_0, m_1, st_{find}) \leftarrow \mathcal{A}(find, 1^n)$

2) $b \leftarrow \{0,1\}$: επιλέγεται τυχαία

3) $st_{issue} \leftarrow \mathcal{A}^{\langle\cdot,\mathcal{U}(vk,m_b)\rangle^1,\langle\cdot,\mathcal{U}(vk,m_{1-b})\rangle^1}(issue, st_{find})$
ο $\mathcal{A}$ συμμετέχει σε δύο διαδραστικές εκτελέσεις του πρωτοκόλλου $Sign$ με τον $\mathcal{U}$ με είσοδο τα μηνύματα m_b και m_{1-b} αντίστοιχα.

4) έστω $(\sigma_0, \sigma_1) = (\perp, \perp)$ αν $\sigma_0 = \perp$ or $\sigma_1 = \perp$
Αν τουλάχιστον μία υπογραφή δεν είναι έγκυρη τότε και οι δύο τίθενται μη έγκυρες.

5) $b^* \leftarrow \mathcal{A}(guess, \sigma_0, \sigma_1, st_{issue})$
ο $\mathcal{A}$ παραλαμβάνει τις αποτυφλωμένες υπογραφές και προσπαθεί να μαντέψει ποια αντιστοιχεί στο m_0.

$$output = \begin{cases} 1 & \text{αν } b^* = b \\ 0 & \text{αλλιώς} \end{cases}$$

Definition 2.9. *Ένα σχήμα τυφλών υπογραφών* $\Pi = (Gen, Sign, Ver)$ *είναι τυφλό εάν για κάθε PPT αντίπαλο* $\mathcal{A}$*, υπάρχει μία αμελητέα συνάρτηση* $negl(n)$ *τ.ω.:*

$$\left| Blind_{\mathcal{A},\Pi}(n) - \frac{1}{2} \right| \leq negl(n)$$

Μερικώς Τυφλές Υπογραφές Σε μερικές εφαρμογές η αρχή της υπογραφής θέλει να συμπεριλαμβάνεται ένα χαρακτηριστικό στο μήνυμα της υπογραφής. Για παράδειγμα ένα συγκεκριμένο όνομα, μία ημερομηνία ή ένα χρηματικό ποσό. Επειδή όμως οι τυφλές υπογραφές δεν επιτρέπουν στην αρχή να δει το μήνυμα (είναι τυφλό) δεν μπορεί να ξέρει αν το χαρακτηριστικό συμπεριλαμβάνεται στο μήνυμα. Για το λόγο αυτό δημιουργήθηκε μία παραλλαγή των Τυφλών Υπογραφών, οι Μερικώς Τυφλές Υπογραφές [27, 34]. Στις υπογραφές αυτές το προς υπογραφή μήνυμα είναι μερικώς τυφλό και μερικώς φανερό, δηλαδή αποτελείται από ένα τυφλό και ένα φανερό κομμάτι. Έτσι, στο φανερό κομμάτι μπορεί να μπει η ζητούμενη πληροφορία.

2.3.2 Κυκλικές Υπογραφές

Στις υπογραφές αυτές ένα μήνυμα μπορεί να πιστοποιηθεί από ένα σύνολο ανθρώπων αλλά χωρίς να ξέρουμε ποιος συγκεκριμένα από το σύνολο το πιστοποίησε. Δηλαδή, το μόνο που επιβεβαιώνουν οι Κυκλικές Υπογραφές είναι ότι κάποιος από την ομάδα των ανθρώπων υπέγραψε.

Οι κυκλικές υπογραφές δεν εμπλέκουν κάποια έμπιστη αρχή για την αρχικοποίηση του συστήματος. Ο καθένας ad-hoc μπορεί να φτιάξει μία ομάδα ανθρώπων, χωρίς να ζητήσει τη συγκατάθεση τους ή να κάνουν κάποιο setup. Η ομάδα αυτή μπορεί στη συνέχεια να επεκταθεί. Την ομάδα ανθρώπων αυτή, των κυκλικών υπογραφών, την ονομάζουμε Ring (κύκλο). Έπειτα, ο καθένας από την ομάδα μπορεί να υπογράψει οποιοδήποτε μήνυμα Ανώνυμα.

Η έννοια των κυκλικών Υπογραφών εισήχθη το 2001 από τους Rivest, Shamir και Tauman, μαζί με το πρώτο σχήμα Κυκλικών Υπογραφών [36]. Το σχήμα αυτό παρουσιάζεται σύντομα παρακάτω

Ο πυρήνας του σχήματος είναι μία οικόγενεια συναρτήσεων που ονομάζονται combining συναρτήσεις:

Definition 2.10. *Μία οικόγενεια combining συναρτήσεων $C_{k,v}(y_1, ..., y_r)$, παίρνει σαν είσοδο ένα κλειδί k, μία αρχική τιμή v και αυθαίρετες τιμές $y_1, ...y_r \in \{0,1\}^b$. Χρησιμοποιεί ένας αλγόριθμο Συμμετρικής Κρυπτογράφησης E_k και παράγει μία έξοδο $z \in \{0,1\}^b$ έτσι ώστε να έχει τις ακόλουθες ιδιότητες:*

1. ***Μετάθεση για κάθε είσοδο:*** *Διατήρησε τις $n-1$ εισόδους σταθερές σε οποιεσδήποτε τιμές y_i, $i \in [r] \setminus \{s\}$ και άσε την s-οστή είσοδο να είναι μεταβλητή. Τότε για κάθε $s \in [n]$ η συνάρτηση $C_{k,v}(y_1, ..., y_{s-1}, \cdot, y_{s+1}, ..., y_r)$ είναι 1-1 απεικόνιση από το y_s στο z.*

2. ***Αποδοτική επίλυση για κάθε μοναδική είσοδο:*** *για κάθε $s \in [r]$, δεδομένων των z και y_i για κάθε $i \neq s$ είναι εφικτό να βρεθεί αποδοτικά ένα y_s τ.ω. $C_{k,v}(y_1, ..., y_r) = z$*

3. ***Ανέφικτη επίλυση για όλες τις εισόδους χωρίς trapdoors:*** *Δεδομένων των k, v, z είναι δύσκολο για κάθε PPT αντίπαλο να βρει $x_1, ..., x_r$ τ.ω. $C_{k,v}(g_1(x_1), ..., g_r(x_r)) = z$ χωρίς να αντιστρέψει καμία από τις $g_1 ..., g_r$*
   όπου $g_1 ..., g_r$ είναι trapdoor μονόδρομες μεταθέσεις.

Υποθέτουμε ότι κάθε πιθανός υπογράφων συνδέεται με ένα μοναδικό δημόσιο κλειδί μέσω ενός **PKI** (public key infrastructure). Έστω sk_i το αντίστοιχο ιδιωτικό κλειδί. Το δημόσιο κλειδί pk_i παράγεται από μία trapdoor μονόδρομη μετάθεση $g_i : \{0,1\}^b \to \{0,1\}^b$ όπου sk_i είναι το trapdoor. Επίσης, το σχήμα χρησιμοποιεί έναν αλγόριθμο Συμμετρικής Κρυπτογράφησης E_k, ο οποίος είναι μετάθεση του συνόλου $\{0,1\}^b$. Τέλος, χρησιμοποιεί μία συνάρτηση κατακερματισμού h.

RST Κυκλικές Υπογραφές

Έστω ένα **PKI** με κάθε χρήστη να έχει στην κατοχή του ένα ζεύγος ιδιωτικού-δημοσίου κλειδιού (sk_i, pk_i), μία μετάθεση Συμμετρικής Κρυπτογράφησης $E_k : \{0,1\}^b \to \{0,1\}^b$ και μία συνάρτηση κατακερματισμούand $h : \{0,1\}^* \to \{0,1\}^b$. Επίσης, μία combining συνάρτηση $C_{k,v} : (\{0,1\}^b)^r \to \{0,1\}^b$. Η Κυκλική Υπογραφή αποτελείται από τους αλγορίθμους:

- $Ring - Sign_{sk_s}(m, pk_1, ..., pk_r)$:
  - Υπολόγισε ένα κλειδί συμμετρικής κρυπτογράφησης $k = h(m, pk_1, ..., pk_r)$
  - Διάλεξε μία τιμή αρχικοποίησης τυχαία $v \leftarrow \{0,1\}^b$
  - Για κάθε άλλο μέλος του Ring $i \neq s$, διάλεξε τυχαία $x_i \leftarrow \{0,1\}^b$ και υπολόγισε $y_i = g_i(x_i)$
  - Για τον υπογράφων s (τον εαυτό σου), βρες ένα y_s τ.ω. $C_{k,v}(y_1, ..., y_r) = v$
  - Για τον υπογράφων s (τον εαυτό σου), χρησιμοπίησε το trapdoor για να βρεις το $x_s = g_s^{-1}(y_s)$

  Αποτέλεσμα: $\sigma = (pk_1, ...pk_r, v, x_1, ..., x_r)$

- $Ring - Verify_{pk_1,...,pk_r}(m, \sigma)$:
  - Υπολόγισε τα $y_i = g_i(x_i)$, για κάθε $i \in [r]$
  - Υπολόγισε το $k = h(m, pk_1, ..., pk_r)$
  - Έλεγξε εάν $C_{k,v}(y_1, ..., y_r) = v$

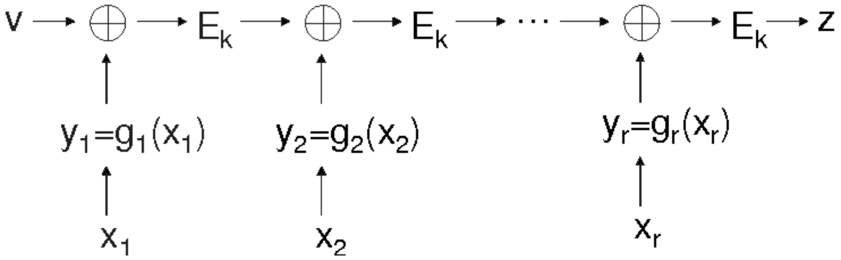
  Αποτέλεσμα: 0 ή 1 ανάλογα εάν η τελευταία εξίσωση ισχύει ή όχι.

Η μονόδρομη trapdoor μετάθεση που χρησιμοποιείται για το μήνυμα $m = q_i n_i + r_i$ ορίζεται ως:

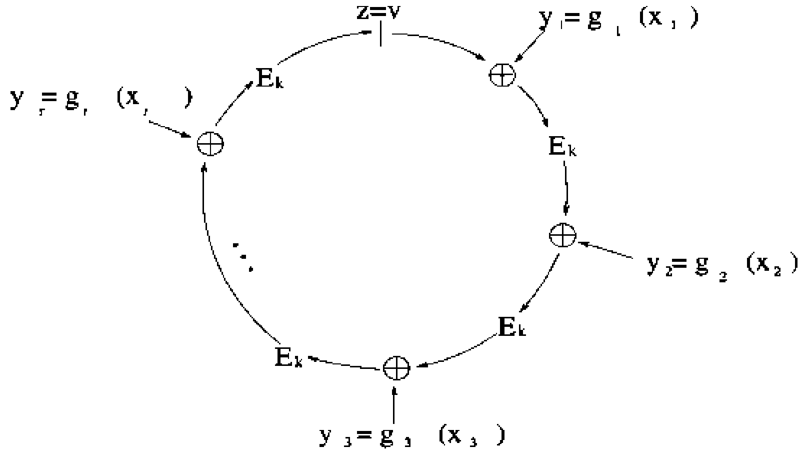
$$g_i(m) = \begin{cases} q_i n_i + f_i(r_i) & \text{αν } (q_i + 1)n_i \leq 2^b \\ m & \text{αλλιώς} \end{cases}$$

Κάθε μέλος του Ring έχει ένα δημόσιο κλειδί $pk_i = (n_i, e_i)$ όπως ορίζεται στο RSA. Το μονο που μένει είναι να οριστεί η combining συνάρτηση:

$$C_{k,v}(y_1, ..., y_r) = E_k(y_r \oplus E_k(y_{r-1} \oplus E_k(y_{r-2} \oplus E_k(... \oplus E_k(y_1 \oplus v)...)))) = z$$



Η σχέση $C_{k,v}(y_1, ..., y_r) = v$ οδηγεί το σχήμα Κρυπτογράφησης σε ένα κυκλικό σχήμα, το οποίο έδωσε το όνομα Κυκλικές στις Υπογραφές αυτές.



Ασφάλεια του RST σχήματος Κυκλικών Υπογραφών

- Anonymity: Για να σπάσει την Ανωνυμία ένας αντίπαλος πρέπει να ξεχωρίσει το x_s από όλα τα άλλα x_i. 'Ολα τα x_i εκτός του x_s δημιουργούνται τυχαία άρα ακολουθούν την ίδια (ομοιόμορφη) κατανομή. Έτσι τα y_i παράγονται τα οποία καθορίζουν μοναδικά το y_s και με τη σειρά του το $x_s = g_s^{-1}(y_s)$. Οπότε, διαισθητικά, καταλαβαίνουμε ότι το x_s παράγεται ομοιόμορφα τυχαία σαν αποτέλεσμα των τυχαίων επιλογών όλων των άλλων x_i. Σημαντικό είναι ότι το $C_{k,v}(y_1, ..., y_{s-1}, \cdot, y_{s+1}, ..., y_r)$ είναι μετάθεση του $\{0,1\}^b$. Οπότε το x_s είναι τέλεια μη-διακρίσιμο από κάθε άλλο x_i.

- Unforgeability: Είναι φανερό ότι η τρίτη ιδιότητα της combining συνάρτησης και η δυσκολία αντιστροφής της μονόδρομης μετάθεσης χωρίς το trapdoor οδηγούν σε Unforgeability του σχήματος.

Τα σχήματα Κυκλικών υπογραφών βρίσκουν εφαρμογή σε πολλά συστήματα όπως σχήματα Ψηφιακών Ψηφοφοριών, σχήματα e-Cash και, πρόσφατα, στα Κρυπτονομίσματα. Σε επόμενο κεφάλαια θα συζητήσουμε τη χρησιμότητα τους στα Ψηφιακά Ανώνυμα Ερωτηματολόγια.

2.4 Αποδείξεις Μηδενικής Γνώσης

Μία Απόδειξη Μηδενικής Γνώσης είναι ένα πρωτόκολλο που επιτρέπει σε κάποιον (Prover $\mathcal{P}$) να πείσει κάποιον άλλον (τον Verifier $\mathcal{V}$) για την εγκυρότητα ενός statement, χωρίς να αποκαλύπτει κάποια άλλη πληροφορία. Οι Αποδείξεις Μηδενικής Γνώσης εισήχθησαν από τους Goldwasser, Michali και Rackoff[47].

Ένα σύστημα Διαδραστικών Αποδείξεων είναι ένα πρωτόκολλο μεταξύ δύο διαδραστικές μηχανές Turing. [47, 48], των οποίων ο στόχος είναι να παράξουν μία έγκυρη απόδειξη ενός statement. Η πρώτη μηχανή Turing αντιπροσωπεύει τον Prover ($\mathcal{P}$) και η δεύτερη τον Verifier($\mathcal{V}$).

Με $\langle A, B\rangle(x)$ συμβοίζουμε την τυχαία μεταβλητή που αντιπροσωπεύει το output του B όταν έχει αλληλεπιδράσει με τον A με κοινή είσοδο το x [48].

Definition 2.11. *Ένα ζεύγος από διαδραστικές Μηχανές Turing $(\mathcal{P}, \mathcal{V})$ καλείται σύστημα διαδραστικής απόδειξης για μία γλώσσα L αν ο $\mathcal{V}$ είναι PPT και υπάρχει μία αμελητέα συνάρτηση $negl(\cdot)$ τ.ω. οι επόμενες συνθήκες να ισχύουν:*

- *Πληρότητα: για κάθε $x \in L$ υπάρχει ένας μάρτυρας w τ.ω.:*

$$Pr[\langle \mathcal{P}(w), \mathcal{V} \rangle (x) = 1] \geq 1 - negl(|x|)$$

- *Ορθότητα: για κάθε $x \notin L$ και κάθε διαδραστική μηχανή Turing $\mathcal{P}^*$:*

$$Pr[\langle \mathcal{P}^*, \mathcal{V} \rangle (x) = 1] \leq negl(|x|)$$

Για τη μηδενική γνώση απαιτούμε να υπάρχει ένας PPT Simulator S ο οποίος να μπορεί να προσομοιώσει το αποτέλεσμα της απόδειξης. Έτσι, μπορούμε να ισχυριστούμε ότι οτιδήποτε μαθαίνει ο Verifier μπορεί να το προσομοιώσει εκτελόντας τον Simulator. Άρα, δεν έμαθε τίποτα πέρα από την ορθότητα του statement που αποδείχθηκε. Παρακάτω δίνουμε τον ορισμό της μηδενικής γνώσης:

Definition 2.12. *Ένα σύστημα διαδραστικών αποδείξεων $(\mathcal{P}, \mathcal{V})$ για μία γλώσσα L, με σχέση μάρτυρα R_L, είναι* ***black-box zero-knowledge*** *αν υπάρχει PPT αλγόριθμος S τέτοιος ώστε για κάθε πολυώνυμο $p(n)$ και για κάθε PPT μηχανή $\mathcal{V}^*$ που χρησιμοποιεί $p(n)$ τυχαία νομίσματα τα ακόλουθα 2 ensembles είναι υπολογιστικά μη-διακρίσιμα:*

- $\{\langle \mathcal{P}(w), \mathcal{V}^*(aux) \rangle (x)\}_{x \in L, aux \in \{0,1\}^*}$ *for* $w \in R_L(x)$
- $\{S^{\mathcal{V}^*}(x, aux)\}_{x \in L, aux \in \{0,1\}^*}$

Αν τα δύο ensembles είναι στατιστικά κοντά τότε το $(\mathcal{P}, \mathcal{V})$ είναι ***black-box statistical zero-knowledge****. Αν τα δύο ensembles είναι ίδια τότε το $(\mathcal{P}, \mathcal{V})$ είναι* ***black-box perfect zero-knowledge****.*

online Simulation-Extractable Μη-Διαδραστική Απόδειξη Μηδενικής Γνώσης (oSE NIZK)
Είναι μία ειδική κατηγορία αποδείξεων μηδενικής γνώσης οι οποίες έχουν τα εξής χαρακτηριστικά:

- Μη-διαδραστικότητα: Ο Prover υπολογίζει μία απόδειξη μόνος του και τη στέλνει στο Verifier χωρίς άλλη αλληλεπίσραση.
- Extractability: Αποτελεί μία απόδειξη γνώσης του μάρτυρα. Δηλαδή αποδεικνύει όχι απλώς ότι το statement ισχύει αλλά και ότι ο Prover διαθέτει ένας μάρτυρα για το statement αυτό.
- Simulation-Extractability: Ο Prover μπορεί πρώτα να λάβει πολυωνυμικά πολλές Simulated αποδείξεις μηδενικής γνώσης του ίδιου πρωτοκόλλου κι έπειτα να φτιάξει τη δική του απόδειξη. Παρόλα αυτά οι αποδείξεις που έλαβε στην αρχή δεν τον βοηθούν στο να παράξει μία ψεύτικη απόδειξη. Άρα τπ πρωτόκολλο είναι ασφαλές
- online: επιτρέπει στο πρωτόκολλο να συμμετέχει ταυτόχρονα σε πολλές εκτελέσεις με κάποιο κακόβουλο Prover, ο οποίος θέλει να παράξει μία ψευδή απόδειξη. Άρα το πρωτόκολλο είναι ασφαλές ακόμα και κάτω από ταυτόχρονες εκτελέσεις.

Κεφάλαιο 3

Ανώνυμα Ερωτηματολόγια

3.1 Ad-Hoc Surveys - Anonize

Μία λύση στο πρόβλημα των Ανώνυμων Ερωτηματολογίων, από Κρυπτογραφικής σκοπιάς, δόθηκε το 2014 από τους Hohenberger, Myers, Pass και shelat [6]. Στην εργασία αυτή ορίζουν τυπικά ένα γενικό κρυπτογραφικό αρχέτυπο που επιλύει το πρόβλημα τα Ad-Hoc Surveys. Επίσης, ορίζουν τις ιδιότητες ασφαλείας που πρέπει να ικανοποιούνται. Στη σύνεχεια δίνουν μία γενική κατασκευή βασισμένη σε αφηρημένες κρυπτογραφικές κατασκευές. Τέλος, συγκεκριμενοποιούν τη γενική κατασκευή με υπάρχοντα κρυπτογραφικά αρχέτυπα. Η συγκεκριμενοποίηση αυτή αποτελεί το Anonize, το οποίο αξίζει να σημειωθεί πως έχει υλοποιηθεί και δοκιμαστεί σε πραγματικές συνθήκες.

Ad-Hoc Surveys Στα σχήματα Ad-hoc Surveys υπάρχουν 3 είδη οντοτήτων:

- Μία μοναδική Αρχή Εγγραφή (ΑΕ).
- Για κάθε ερωτηματολόγιο μία Αρχή του Ερωτηματολογίου.
- Χρήστες, η ταυτότητα των οποίων χαρακτηρίζεται από ένα id.

Τυπικά ένα σχήμα ad-hoc Survey είναι μία εφτάδα PPT αλγορίθμων:

$$(GenRA, RegUser^{RA}, RegUser^{\mathcal{U}}, GenSurvey, Authorized, SubmitSurvey, Check)$$

- $GenRA(1^n)$: δημιουργεί ένα ζευγάρι δημόσιου-ιδιωτικού κλειδιού vk_{RA}, sk_{RA} vk_{RA} δημοσιοποιείται, sk_{RA} παραμένει ιδιωτικό (κρυφό).
- $RegUser^{RA}(sk_{RA}, vk_{RA}, id)$: output *accept* ή *fail.*
  Εκτελείται από την Αρχή Εγγραφής και αλληλεπιδρά με ένα id για το εγγράψει. Αν το διαδραστικό πρωτόκολλο επιτύχει δίνει output *accept* αλλιώς *fail.*
- $RegUser^{\mathcal{U}}(1^n, vk_{RA}, id)$: output $cred_{id}$ ή *fail.*
  Εκτελείται από ένα id και αλληλεπιδρά με την Αρχή Εγγραφής. Στην περίπτωση που το id δεν έχει εγγραφεί πριν δίνει σαν output ένα master credential $cred_{id}$, το οποίο διατηρείται μυστικό. Διαφορετικά δίνει output *fail.*
- $GenSurvey(1^n, vid, L)$: output vk_{vid}.
  Εκτελείται από την αρχή του Ερωτηματολογίου. Το vid είναι ένα μοναδικό δημόσιο αναγνωριστικό του Ερωτηματολογίου που επιλέγεται από την αρχή του Ερωτηματολογίου. Το L είναι η

(αρχική) λίστα των id που συμμετέχουν στο ad-hoc group του Ερωτηματολογίου.
Το vk_{vid} είναι το δημόσιο κλειδί του ερωτηματολογίου.

- $Authorized(vid, vk_{vid}, id)$ output *accept* ή *fail*.
  Μπορεί να εκτελεστεί από τον καθένα για να ελέγξει εάν το id είναι στη λίστα των συμμετεχόντων, άρα έχει δικαίωμα να συμμετάσχει στο ερωτηματολόγιο vid.

- $SubmitSurvey(1^n, vid, vk_{vid}, m, cred_{id})$: output $Sub = (tok, m, tokauth)$.
  Εκτελείται από το χρήστη id. Για ένα συγκεκριμένο ερωτηματολόγιο vid (με παραμέτρους vk_{vid}), με το μοναδικό master credential $cred_{id}$ και μία απάντηση m προς κατάθεση ένα μοναδικό token μίας χρήσης tok και ένα $tokauth$ δημιουργούνται.
  Το token μίας χρήσης tok δεν προδίδει την ταυτότηα του id και είναι μοναδικό για κάθε vid. Όπως θα δούμε στη συνέχεια, το $tokauth$ αποδεικνύει ότι η κατάθεση της απάντησης m είναι ορθή.

- $Check(vk_{RA}, vid, vk_{vid}, Sub)$: output *accept* ή *fail*.
  Ελέγχει εάν η κατάθεση ερωτηματολογίου, Sub, είναι ορθή ή όχι. Μπορεί να εκτελεστεί από τον καθένα.

Ιδιότητες Ασφαλείας. Το παραπάνω σχήμα απαιτείται να έχει τις ιδιότητες της Ανωνυμίας και της Πιστοποιησιμότητας.
Για την Ανωνυμία θεωρούμε ότι:

1. Η Αρχή Εγγραφής , πολλές Αρχές Ερωτηματολογίων και πολλοί χρήστες ελέγχονται από τον αντίπαλο.
2. Ο αντίπαλος έχει ταυτοποιήσει το χρήστη σε παλιά ερωτηματολόγια (της επιλογής του).
3. Ο αντίπαλος θα ταυτοποιήσει το χρήστη σε μελλοντικά ερωτηματολόγια (της επιλογής του)..

και παρόλα αυτά δεν μπορεί να ταυτοποιήσει το χρήστη στο παρών ερωτηματολόγιο (που τον ενδιαφέρει).

Για την Πιστοποιησιμότητα θεωρούμε ότι:

1. Πολλές Αρχές Ερωτηματολογίων και πολλοί χρήστες ελέγχονται από τον αντίπαλο.
2. Ο αντίπαλος μπορεί να ζητήσει από οποιονδήποτε (τίμιο) χρήστη id κατάθεση ερωτηματολογίου με απάντηση της επιλογής του m και για κάθε ερωτηματολόγιο. Αν ο χρήστης δεν είναι εγγεγραμμένος τότε αναγκάζεται να εγγραφεί αλληλεπιδρώντας με την Αρχή Εγγραφής.

Ο αντίπαλος επιλέγει ένα ερωτηματολόγιο vid και ο στόχος του είναι να καταθέσει περισσότερες (έγκυρες) απαντήσεις από τον αριθμό των συμμετεχόντων χρηστών που ελέγχει. Υπάρχουν δύο τρόποι να πετύχει: είτε (τουλάχιστον) ένας συμμετέχων διεφθαρμένος χρήστης που ελέγχει καταφέρνει να καταθέσει δύο διαφορετικές απαντήσεις επιτυχημένα ή ο αντίπαλος καταφέρνει να περάσει μία (τουλάχιστον) κατάθεση με ένα id που δεν είναι στη λίστα συμμετεχόντων.

Γενική Κατασκευή Τα ad-hoc surveys χωρίζονται σε τρεις φάσεις: Εγγραφή Χρήστη, Δημιουργία Ερωτηματολογίου και Κατάθεση απαντήσεων ερωτηματολογίου. Οι Hohenberger et al. κατασκεύασαν ένα ad-hoc survey σχήμα χρησιμποιώντας τα παρακάτω κρυπτογραφικά αρχέτυπα:

- Ένα σχήμα Δέσμευσης $(Gen_{com}, Com, Open)$
- Ένα σχήμα Ψηφιακών Υπογραφών $(Gen, Sign, Ver)$
- Ένα σχήμα μερικώς Τυφλών Υπογραφών $(Gen', Blind', Sign', Unblind', Ver')$
- Μία οικογένεια Ψευδοτυχαίων συναρτήσεων $\{f_s\}$
- Μία online simulation-extractable μη-διαδραστική απόδειξη μηδενικής γνώσης στο μοντέλο του τυχαίου μαντείου (P, V, RO).

Φάση Εγγραφής Χρήστη

1. $GenRA(1^n)$: η αρχή Εγγραφής χρησιμοποιεί τον αλγόριθμο $Gen'(1^n)$ για να δημιουργήσει ένα ζευγάρι κλειδιών για Μερικώς Τυφλές Υπογραφές: (vk_{RA}, sk_{RA})

2,3. $\langle RegUser^{RA}(sk_{RA}, vk_{RA}, id), RegUser^{\mathcal{U}}(1^n, vk_{RA}, id)\rangle$:
ο χρήστης id δημιουργεί ένα τυχαίο $s \leftarrow \{0,1\}^n$, το τυφλώνει (χρησιμοποιώντας τον αλγόριθμο Com του σχήματος δέσμευσης) και το στέλνει στην αρχή εγγραφής.
Έπειτα, η αρχή Εγγραφής υπογράφει το $Com(s)$ μαζί με το id και στέλνει την υπογραφή πίσω στο χρήστη.
Έπειτα το id αποτυφλώνει την υπογραφή και παίρνει το τελικό credential $(s, Sign(id, s))$, το οποίο θα χρησιμοποιήσει για να καταθέσει μία απάντηση ερωτηματολογίου.

Φάση δημιουργίας Ερωτηματολογίου
Οποιοσδήποτε το επιθυμεί μπορεί να ανοίξει ένα ερωτηματολόγιο, καθιστώντας τον εαυτό του αρχή του Ερωτηματολογίου.
Η αρχή του Ερωματολογίου επιλέγει τους συμμετέχοντες (σύμφωνα με τα id τους) και τους τοποθετεί σε μία λίστα L. Επίσης, δημιουργεί ένα αναγνωριστικό του ερωματολογίου vid.

4. $GenSurvey(1^n, vid, L)$: Η αρχή του Ερωματολογίου δημιουργεί ένα ζευγάρι κλειδιών (vk_{SA}, sk_{SA}) για ψηφιακές υπογραφές και με το μυστικό κλειδί sk_{SA} και για κάθε $id \in L$ βάζει μία υπογραφή στο (vid, id).
Το (δημόσιο) output είναι η λίστα των υπογραφών $\tilde{L} = \{(id, Sign_{sk_{SA}}(vid, id))\}_{id \in L} = \{\big(id, \sigma_{id}^{vid}\big)\}_{id \in L}$ και το δημόσιο κλειδί vk_{SA}.
Συνοψίζοντας, το output είναι το $vk_{vid} = (vk_{SA}, \tilde{L})$

Φάση κατάθεσης απάντησης

5. $Authorized(vid, vk_{vid}, id)$: ο χρήστης id ελέγχει αν βρίσκεται στη λίστα με τις υπογραφές $\tilde{L}$ και εάν η υπογραφή στο id του είναι έγκυρη δηλαδή $Ver_{vk_{SA}}((vid, id), \sigma_{id}^{vid}) = 1$.

6. $SubmitSurvey(1^n, vid, vk_{vid}, m, cred_{id})$:εάν ο αλγόριθμος authorized έδωσε έξοδο 1 τότε ο χρήστης id κάνει τα παρακάτω:

   - Υπολογίζει ένα μοναδικό token $tok = f_{s_{id}}(vid)$, όπου η $f_{s_{id}}$ είναι μία ψευδοτυχαία συνάρτηση με τυχαίο σπόρο το s_{id} που υπέγραψε (στα τυφλά) η αρχή Εγγραφής στο χρήστη κατά τη φάση Εγγραφής.

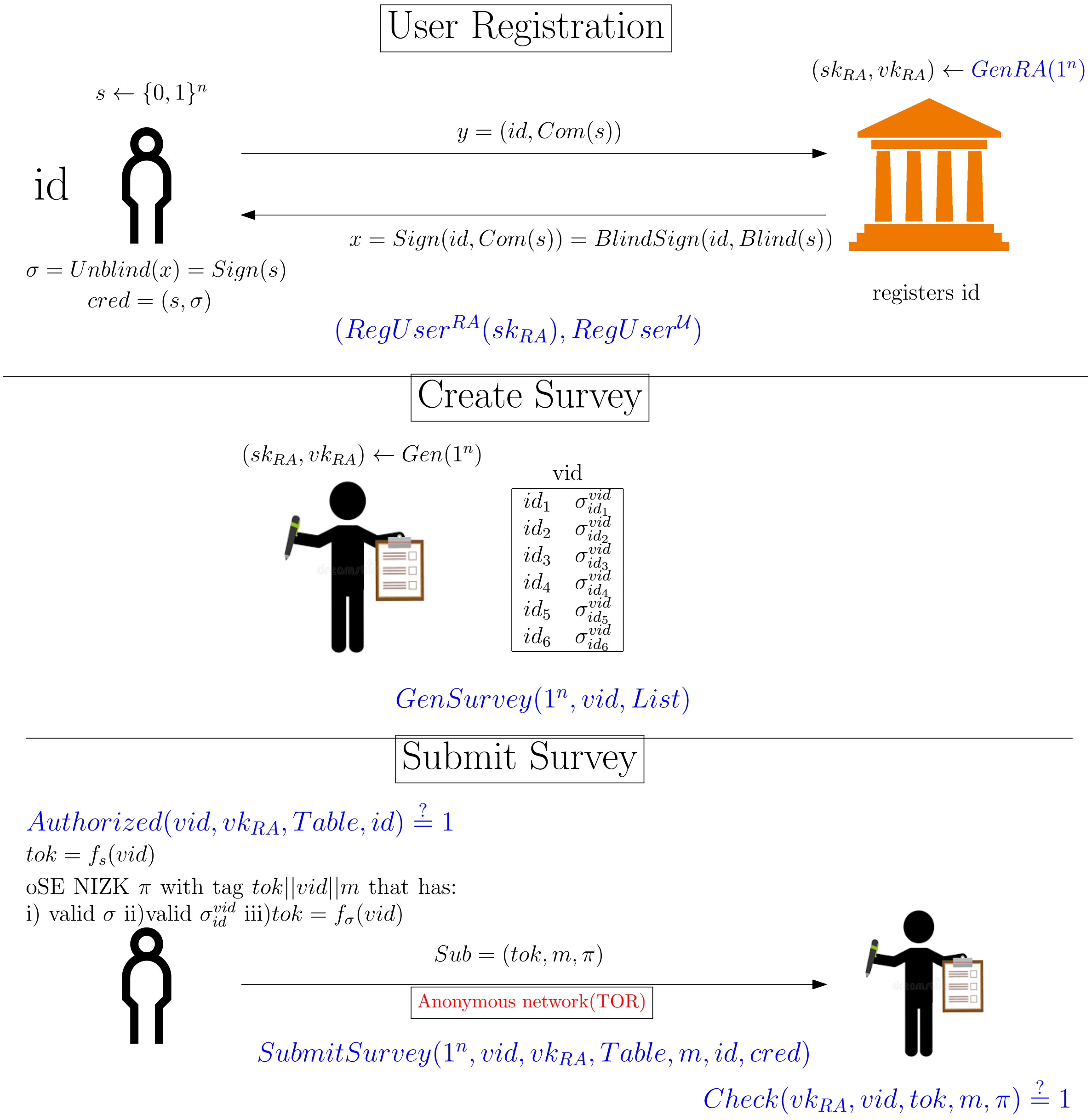
   - Σχηματίζει μία oSE μη-διαδραστική απόδειξη μηδενικής γνώσης π αποδεικνύοντας ότι το token υπολογίστικε ορθά $tok = f_{s_{id}}(vid)$ και ότι έχει έγκυρες υπογραφές από την αρχή του ερωτηματολογίου στο (id, s_{id}) και από την αρχή εγγραφής στο (vid, id).

   - Τέλος, δίνει output $Sub = (tok, m, \pi)$ και τη στέλνει στην αρχή Ερωτηματολογίου.

7. $Check(vk_{RA}, vid, vk_{vid}, Sub)$: η αρχή του ερωτηματολογίου παραλαμβάνει την κατάθεση απάντησης Sub και αποδέχεται αν η απόδειξη π επαληθεύεται.

Μετά την ολοκλήρωση

Κάποιες διακασίες ελέγχου μπορούν να γίνουν οποιαδήποτε στιγμή και από οποιονδήποτε. Οποιοσδήποτε θέλει μπορεί να εκτελέσει τον αλγόριθμο $Authorized(vid, vk_{vid}, id)$ για οποιοδήποτε id της επιλογής του για να δει αν το id ήταν στη λιστα των συμμετεχόντων ή όχι. Επιπροσθέτως, ανάλογα με την πολιτική του ερωτηματολογίου, τα αποτελέσματα μπορούν να δημοσιοποιηθούν ή όχι. Αν δημοσιοποιηθουν ο καθένας μπορεί να εκτελέσει τον αλγόριθμο $Check(vk_{RA}, vid, vk_{vid}, Sub)$ για κάθε απάντηση που κατατέθηκε για να ελέγξει την εγκυρότητα των καταθέσεων. Τέλος, ο καθένας μπορεί να πραγματοποιήσει ελέγχους για πιθανές διπλές καταθέσεις εξετάζοντας τα tokens tok και βλέποντας αν υπάρχει κάποιο token που εμφανίζεται 2 (ή περισσότερες) φορές.


User Registration
$(sk_{RA}, vk_{RA}) \leftarrow GenRA(1^n)$
$s \leftarrow \{0,1\}^n$
id
$y = (id, Com(s))$
$x = Sign(id, Com(s)) = BlindSign(id, Blind(s))$
$\sigma = Unblind(x) = Sign(s)$
$cred = (s, \sigma)$
registers id
$(RegUser^{RA}(sk_{RA}), RegUser^{\mathcal{U}})$
Create Survey
$(sk_{RA}, vk_{RA}) \leftarrow Gen(1^n)$
vid
$id_1 \quad \sigma_{id_1}^{vid}$
$id_2 \quad \sigma_{id_2}^{vid}$
$id_3 \quad \sigma_{id_3}^{vid}$
$id_4 \quad \sigma_{id_4}^{vid}$
$id_5 \quad \sigma_{id_5}^{vid}$
$id_6 \quad \sigma_{id_6}^{vid}$
$GenSurvey(1^n, vid, List)$
Submit Survey
$Authorized(vid, vk_{RA}, Table, id) \overset{?}{=} 1$
$tok = f_s(vid)$
oSE NIZK π with tag $tok||vid||m$ that has:
i) valid σ ii)valid σ_{id}^{vid} iii)$tok = f_\sigma(vid)$
$Sub = (tok, m, \pi)$
Anonymous network(TOR)
$SubmitSurvey(1^n, vid, vk_{RA}, Table, m, id, cred)$
$Check(vk_{RA}, vid, tok, m, \pi) \overset{?}{=} 1$

Anonize Το Anonize είναι μία υλοποίηση του παραπάνω γενικού σχήματος Ad-Hoc Surveys. Τα συγκεκριμένα αρχέτυπα που χρησιμοποιεί είναι τα εξής:

- Σχήμα Δέσμευσης του Pedersen [19].
- Ψευδοτυχαία συνάρτηση των Dodis-Yampolskiy [74].
- Σχήμα Ψηφιακών Υπογραφών των Boneh-Boyen [43]
- Σχήμα Μερικώς Τυφλών Υπογραφών που παράγεται από το συνδυασμό των σχημάτων δέσμευσης και Ψηφιακών Υπογραφών. Η κατασκευή περιγράφεται [6].
- oSE μη διαδραστική απόδειξη μηδενικής γνώσης που κατασκευάζεται από ένα Σ-πρωτόκολλο στο οποίο εφαρμόζουμε μετασχηματισμό Pass [20].

3.2 Προς ισχυροποίηση της ασφάλειας του Anonize: κάποιες παρατηρήσεις

3.2.1 Κακόβουλη αρχή Εγγραφής - επίθεση στην πιστοποιησιμότητα

Στον ορισμό της πιστοποιησιμότητας των ad-hoc surveys σχημάτων η αρχή εγγραφής θεωρείται τίμια. Οπότε παρατηρούμε ότι μία κακόβουλη αρχή εγγραφής μπορεί να συνεργαστεί με ένα κακόβουλο χρήστη id και να του υπογράψει πολλά s_{id}. Έπειτα, ο χρήστης μπορεί να καταβάλει όσες υποβολές όσα και τα υπογεγραμμένα s_{id} που έχει.

Αυτό είναι εφικτό, διότι η ταυτότητα του χρήστη δεν αποκαλύπτεται και οι υπογραφές από τις αρχές εγγραφής και ερωτηματολογίου αποδεικνύονται με μηδενική γνώση. Από την άλλη, το $tok = f_{s_{id}}(vid)$, που είναι φανερό θα είναι διαφορετικό για κάθε διαφορετικό s_{id} που η αρχή υπογράφει στον κακόβουλο χρήστη.

Το σενάριο αυτό επιτρέπει σε μία κακόβουλη αρχή εγγραφής να προσθέσει όσες απαντήσεις θέλει και να προκαλέσει μεγάλη ζημιά στην ακεραιότητα του αποτελέσματος του ερωτηματολογίου.

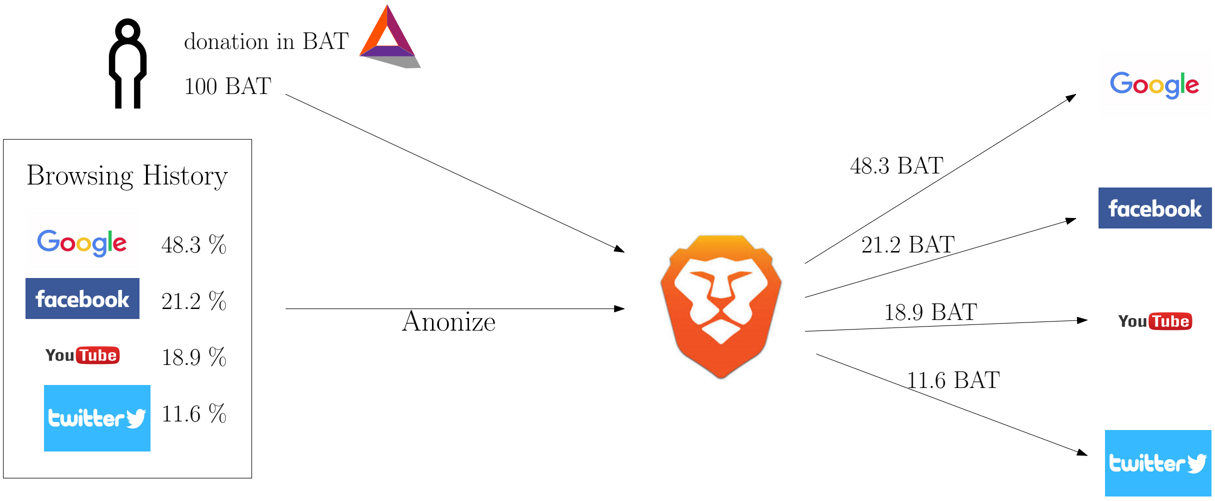
3.2.2 Εξαγορά ψήφου (έλλειψη Αδυναμίας-Απόδειξης)

Μία σημαντική έννοια στις ψηφιακές ψηφοφορίες είναι η Αδυναμία-Απόδειξης. Αυτή υπαγορεύει ότι ο ψηφοφόρος δεν πρέπει να κατέχει κάποια πληροφορία που θα τον βοηθήσει να αποδείξει πως ψήφησε. Αυτό εμποδίζει τον ψηφοφόρο από το να πουλήσει την ψήφο του.

Το Anonize δε διαθέτει Αδυναμία-Απόδειξης. Κάποιος μπορεί να αναρωτιέται γιατί να εξαγοράσει κάποιος κατάθεση Ανώνυμου ερωτηματολογίου. Αυτό είναι αρκετά υποκειμενικό και δεν είναι ένα ερώτημα που σχετίζεται με την Κρυπτογραφία. Η γνώμη μας είναι πως η εξαγορά απαντήσεων σε ερωτηματολόγια δεν είναι μακριά από την πραγματικότητα. Επίσης, ορισμένες φορές υπάρχει άμεσο οικονομικό αντίκτυπο από το αποτέλεσμα των ερωτηματολογίων.

Κίνητρο από την πραγματικότητα Μία εφαρμογή του Anonize είναι στον Brave Broswer. O Brave είναι ένας νέος browser που προσφέρει ad-blocking. Αλλά οι διαφημήσεις είναι η βασική πηγή κέρδους για πολλές ιστοσελίδες. Για το λόγο αυτό, ο Brave δίνει στους χρήστες την επιλογή να δωρίσει ένα ποσό στο Brave και αυτό το ποσό διαμοιράζεται δίκαια στις ιστοσελίδες σύμφωνα με το χρόνο που ξόδεψε ο χρήστης (ποσοστιαία) που έκανε τη δωρεά σε αυτές. Για παράδειγμα ο χρήστης αν πέρασε το 25% του χρόνου του στο ”website.web” και δωρίσει 10$, το ”website.web” θα πάρει 2.5$ (στην πραγματικότητα λίγο λιγότερα καθώς ο Brave κρατάει και ένα fee).

Παρόλα αυτά, οι χρήστες δεν μπορούν να στείλουν το ιστορικό τους στο Brave διότι τότε η ιδιωτικότητα τους θα παραβιαστεί. Έτσι, χρησιμοποιείται το Anonize για να σταλούν τα δεδομένα. Επίσης, τονίζουμε ότι οι πληρωμές είναι κι αυτές ανώνυμες, αφού χρησιμοποιείται ένα κρυπτονόμισμα το Basic Attention Token (BAT). Συνοψίζοντας, ο διαμοιρασμός των δωρεών στον Brave Browser απαιτεί ένα κρυπτονόμισμα για ανώνυμες συναλλαγές και το Anonize για Ανώνυμη κατάθεση προτιμήσεων ιστοσελίδων.



Μπορούμε να εντοπίσουμε ένα κίνητρο για εξαγορά ψήφου στην παραπάνω χρήση του Anonize. Ένας χρήστης μπορεί να συνεργαστεί με κάποια κακόβουλη ιστοσελίδα, έστω ”evil.web”. Η ”evil.web” θα αγοράσει την κατάθεση του χρήστη και ο χρήστης θα καταθέσει 100 % προτίμηση στο ”evil.web”. Έπειτα, ο χρήστης θα παρουσιάσει την απόδειξη στο ”evil.web” και θα πάρει ένα συμφωνημένο ποσοστό της δωρεάς πίσω. Οπότε ο χρήστης και το ”evil.web” βγαίνουν κερδισμένοι και ο Brave είναι ο πραγματικός χαμένος από αυτή τη συναλλαγή. Παρόλα αυτά, τονίζουμε ότι το σενάριο αυτό δεν είναι ρεαλιστικό στο παρόν σενάριο καθώς πρόκειται για δωρεές! Οπότε δεν είναι ρεαλιστικό ο χρήστης να θέλει ποσοστό της δωρεάς πίσω.

Η Αδυναμία-Απόδειξης δεν είναι, λοιπόν, κατά της άποψης μας, απαραίτητη στο παραπάνω. Θα ήταν όμως αν οι πληρωμές ήταν απαραίτητες, για παράδειγμα μελλοντικά σε κάποιο pro version του Brave. Παρόλα αυτά, ανεξαρτήτως του παραπάνω, η Αδυναμία-Απόδειξης είναι πολύ επιθυμητή ιδιότητα σε μεγάλης κλίμακας συστήματα Ανώνυμων Ερωτηματολογίων.

3.2.3 Μία side-channel επίθεση

Η αρχή του ερωματολογίου είναι αυτή που ενημερώνει τους συμμετέχοντες για την ύπαρξη του ερωτηματολογίου. Συνήθως στο Anonize αυτό γίνεται μέσω e-mail. Οπότε, η αρχή του ερωματολογίου επιλέγει ποιους χρήστες θα ενημερώσει και πότε. Οπότε αν η αρχή του ερωτηματολογίου είναι κακόβουλη και ο χρήστης απρόσεκτος, η πρώτη μπορεί να ενημέρωσει ένα μόνο χρήστη, παρόλο που θα έχει υπογράψει για τη συμμετοχή πολλών χρηστών. Έτσι, όταν θα λάβει απάντηση θα ξέρει την ταυτότητα του χρήστη που απάντησε. Άρα σπάει την ιδιότητα της Ανωνυμίας. Έπειτα, αν ο πρώτος χρήστης απαντήσει μπορεί να προχωρήσει με τον ίδιο τρόπο στο δεύτερο κτλ. Η επίθεση αυτή στην Ανωνυμία είναι side-channel και έχει να κάνει με τον ανθρώπινο παράγοντα και όχι με το κρυπτογραφικό πρωτόκολλο.

3.2.4 Η λίστα συμμετεχόντων είναι δημόσια

Ένα ζήτημα το οποίο το Anonize δεν επιλύει είναι ότι οι ταυτότητες των χρηστών που συμμετέχουν σε ένα ερωτηματολόγιο πρέπει να είναι δημόσιες. Διαφορετικά δεν μπορεί να λειτουργήσει ο αλγόριθμος Authorized. Σε κάποιες πραγματικές περιπτώσεις θα ήταν άβολο για το χρήστη να βρίσκεται σε μία δημόσια λίστα συμμετεχόντων ενός ερωτηματολογίου. Για παράδειγμα, αν πρόκειται για ένα ιατρικό ερωτηματολόγιο που απευθύνεται σε ασθενείς, θα ήταν παραβίαση της ιδιωτικότητας των ασθενών να δημοσιοποιηθεί μία λίστα με συμμετέχοντες. Η παρατήρηση αυτή γίνεται στη δημοσίευση του Anonize [6].

3.2.5 Οι απαντήσεις δεν υποβάλλονται ως κρυπτοκείμενα

Οι απαντήσεις υποβάλλονται ως φανερά μηνύματα και όχι ως κρυπτοκείμενα. Αυτό, σε μερικές περιπτώσεις, μπορεί να οδηγήσει σε αποανωνυμοποίηση του χρήστη. Αυτό έχει να κάνει με το είδος των ερωτήσεων του ερωτηματολογίου. Για παράδειγμα, αν ένα ερωτηματολόγιο κάνει αρκετές προσωπικές ερωτήσεις τότε οι φανερές απαντήσεις μπορούν να οδηγήσουν σε ταυτοποίηση.

Κεφάλαιο 4

Εναλλακτικές κατασκευές συστημάτων Ανώνυμων Ερωτηματολογίων

4.1 Ένα σύστημα Ανώνυμων Ερωτηματολογίων από Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές

Οι Ανιχνεύσιμες Κυκλικές Υπογραφές είναι μία τροποποίηση των Κυκλικών Υπογραφών που εισήχθη το 2007 απο του Fujisaki και Suzuki [11]. Αυτές λειτουργούν σαν κανονικές Κυκλικές Υπογραφές με τη διαφορά ότι κάθε υπογράφων έχει δικαίωμα 1 ανώνυμης υπογραφής. Τη δεύτερη φορά που θα υπογράψει ο ίδιος χρήστης αποκαλύπτεται η ταυτότητα του.

Στις υπογραφές αυτές οποιοσδήποτε το επιθυμεί μπορεί να φτιάξει μία ομάδα ανθρώπων οι οποιοί αντιπροσωπεύονται από τα δημόσια κλειδιά τους pk_i. Το διατεταγμένο σύνολο των δημοσίων κλειδιών, που αντιπροσωπεύει την ομάδα, το ονομάζουμε κύκλο (Ring), $Ring = \{pk_1, ..., pk_n\}$. Θεωρούμε ότι έχουμε ένα γεγονός (event) που σχετίζεται με την υπογραφή το οποίο αντιπροσωπεύεται από ένα tag. Από το $Ring$ και το tag δημιουργείται ένα τυχαίο $h = H(tag, Ring)$, όπου H είναι μία συνάρτηση σύνοψης. Έπειτα, ο i-οστός υπογράφων υψώνει το μυστικό κλειδί του στο h για να παραχθεί η υπογραφή του, $\sigma_i = h^{sk_i}$. Για την τελική Ανιχνεύσιμη Κυκλική Υπογραφή δημιουργεί μία υπογραφή n στοιχείων (όπου n ο αριθμός των συμμετεχόντων στο Ring) $\sigma = (\sigma_1, ..., \sigma_n)$ και τοποθετεί το σ_i στην i-οστή θέση. Στις υπόλοιπες θέσεις τοποθετεί μία τιμή συναρτήσει του σ_i και του μηνύματος m που υπογράφει. Έτσι, κάθε φορά που υπογράφει η i-οστή τιμή θα είναι ίδια σ_i και μετά τη δεύτερη φορά θα ανιχνευτεί η ταυτότητα του.

Για να εξασφαλιστεί ότι ο υπογράφων έβαλε το σ_i όντως στην i-οστή θέση και ότι όντως είναι μέλος του $Ring$, η υπογραφή συνοδεύεται από μία απόδειξη μηδενικής γνώσης που αποδεικνύει ότι έβαλε ένα σ_i που είναι της μορφής h^{sk_i} και μάλιστα στη θέση αυτή βρίσκεται και το δημόσιο κλειδί του pk_i, άρα είναι όντως ο i-οστός. Λόγω της ιδιότητας της μηδενικής γνώσης όποιοσδήποτε βλέπει την απόδειξη δεν μαθαίνει ούτε τα pk_i, sk_i και σ_i άρα η ανωνυμία διατηρείται. Όμως βεβαιώνεται ότι ο υπογράφων είναι κάποιος από τους $pk_1, ..., pk_n$. Αναλυτικότερα, το σχήμα των Ανιχνεύσιμων Κυκλικών Υπογραφών βρίσκεται παρακάτω.

Fujisaki-Suzuki Ανιχνεύσιμες Κυκλικές Υπογραφές

Έστω G μία ομάδα με τάξη ένα πρώτο q και γεννήτορα $g \in G$ και $H : \{0,1\}^* \to G$, $H' : \{0,1\}^* \to G$, $H'' : \{0,1\}^* \to \mathbb{Z}_q$ τρία τυχαία μαντεία. Έστω $L = (tag, pk_{[n]})$

- $Gen(1^n)$: εκτελείται από τον παίκτη i.
  Διάλεξε ένα μυστικό κλειδί $sk_i \leftarrow \mathbb{Z}_q$ τυχαία και υπολόγισε το δημόσιο κλειδί $pk_i = g^{x_i}$.
  Υποθέτουμε την ύπαρξη ενός ασφαλούς **PKI** όπου ο i κάνει την εγγραφή του δημομσίου κλειδιού του pk_i.

- $Sign_{sk_i}(L, m)$: εκτελείται από τον παίκτη i.

  1. Υπολόγισε το $h = H(L)$ και το $\sigma_i = h^{sk_i}$
  2. Υπολόγισε τα $A_0 = H'(L, m)$, $A_1 = \left(\frac{\sigma_i}{A_0}\right)^{1/i}$ και για κάθε $j \neq i$ υπολόγισε τα $\sigma_j = A_0 A_1^j$.
     $\sigma_{[n]} = (\sigma_1, ..., \sigma_n) = (A_0^{\frac{i-1}{i}} h^{sk_i \frac{1}{i}}, A_0^{\frac{i-2}{i}} h^{sk_i \frac{2}{i}}, ..., h^{sk_i}, ..., A_0^{\frac{i-n}{i}} h^{sk_i \frac{n}{i}})$
  3. Για τη γλώσσα $\mathcal{L} = \{(L, h, \{\sigma_1, ..., \sigma_n\}) | \exists sk_i \text{ s.t. } pk_i = g^{sk_i} \text{ and } \sigma_i = h^{sk_i}\}$
     Υπολόγισε ένα NIZK:

     – Διάλεξε τυχαία $w_i, \{c_j, z_j\}_{j \neq i} \leftarrow \mathbb{Z}_q$ και υπολόγισε το commit:
     $$a_i = g^{w_i}, a_j = g^{z_j} pk_i^{c_j} \text{ for all } j \neq i$$
     $$b_i = h^{w_i}, b_j = h^{z_j} \sigma_i^{c_j} \text{ for all } j \neq i$$
     – Υπολόγισε το challenge:
     $$c = H''(L, A_0, A_1, a_{[n]}, b_{[n]})$$
     $$c_i = c - \textstyle\sum_{j \neq i} c_j \ (mod\ q)$$
     – Υπολόγισε το response:
     $$z_i = w_i - c_i sk_i$$

     Το αποτέλεσμα είναι η απόδειξη $\pi = \{c_{[n]}, z_{[n]}\} = \{(c_1, .., c_n), (z_1, ..., z_n)\}$

  output: Η υπογραφή είναι η $\sigma = (A_1, \pi) = (A_1, c_{[n]}, z_{[n]})$

- $Verify_L(m, \sigma)$:

  1. Υπόλογισε τα $h = H(L)$, $A_0 = H'(L, m)$, $\sigma_j = A_0 A_1^j$ για κάθε $i \in [n]$
  2. Υπολόγισε τα $a_i = g^{z_i} pk_i^{c_i}$ και $b_i = h^{z_i} \sigma_i^{c_i}$ για κάθε $i \in [n]$
  3. Έλεγξε αν $H''(L, A_0, A_1, a_{[n]}, b_{[n]}) = \sum_{i=1}^{n} c_i \ (mod\ q)$

  Αποτέλεσμα: 1 αν η τελευταία εξίσωση ισχύει 0 αλλιώς.

- $Trace_L(m, \sigma, m', \sigma')$: όπου $\sigma = (A_1, c_{[n]}, z_{[n]})$ και $\sigma' = (A_1', c_{[n]}', z_{[n]}')$

  1. Υπολόγισε τα $h = H(L)$, $A_0 = H'(L, m)$, $\sigma_j = A_0 A_1^j$ για κάθε $i \in [n]$
     και $A_0' = H'(L, m')$, $\sigma_j' = A_0' A_1'^j$ για κάθε $i \in [n]$
  2. Σύγκρινε τα σ_i και σ_i' για κάθε $i \in [n]$

$$output = \begin{cases} pk_i & \text{if } \sigma_i = \sigma_i' \text{ and } \sigma_j \neq \sigma_j' \text{ for all } j \neq i \qquad \textbf{(1 exactly equal)} \\ \text{linked} & \text{if } \sigma_i = \sigma_i' \text{ for all } i \in [n] \qquad \textbf{(n exactly equal)} \\ \text{indep} & otherwise \qquad \textbf{(0 OR 2} \leq \textbf{equal} \leq \textbf{n-1)} \end{cases}$$

Ιδιότητες Ασφάλειας Οι ιδιότητες που πληρεί το παραπάνω σύστημα υπογραφών είναι οι εξής:

- Tag-Linkability: έστω ότι ο αντίπαλος φτιάχνει ένα tag και ένα $Ring = \{pk_1, ..., pk_n\}$ της επιλογής του όπου ελέγχει t μυστικά κλειδιά συμμετεχόντων, $t \leq n$ (μπορεί ακόμα και να τα ελέγχει όλα). Τότε δεν μπορεί να περάσει $t + 1$ ανεξάρτητες υπογραφές, τουλάχιστον 2 θα γίνουν traced ή linked.

- Anonymity: Ο αντίπαλος δεν μπορεί από την υπογραφή να καταλάβει την ταυτότητα του υπογράφοντος. Ορίζεται σαν παιχνίδι διακρισιμότητας μεταξύ δύο υπογραφών από τα ίδια $tag, Ring$ και ο αντίπαλος επίσης μπορεί σε άλλα tags και rings να έχει κάνει αποανωνυμοποίηση στους ίδιους υπογράφοντες. Ισχύει υποθέτοντας δυσκολία του προβλήματος DDH.

- Exculpability: Ο αντίπαλος στοχεύει σε κάποιον υπογράφοντα με δημόσιο κλειδί pk_i και αφού κάνει oracle queries για να δει την υπογραφή του σε άλλα events, έπειτα προσπαθεί να τον παγιδεύσει δίνοντας δύο υπογραφές που να τον κάνουν trace. Ισχύει υποθέτοντας δυσκολία του προβλήματος DLog. Επεκτείνεται εύκολα και σε πολυωνυμικα πολλούς στόχους-υπογράφοντες αντί για 1.

Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές Το παραπάνω σχήμα Ανιχνεύσιμων Κυκλικών Υπογραφών επιτρέπει Ανιχνευσιμότητα μόνο για στατικά Rings. Αν στο Ring προστεθούν εκ των υστέρων δημόσια κλειδιά (έστω και 1) τότε έχουμε $h' = H(tag, Ring') \neq h$, συνεπώς $h^{sk_i} = \sigma_i \neq \sigma'_i = h'^{sk_i}$. Αυτό σημαίνει ότι κάθε υπογράφων μπόρει να υπογράψει εκ νέου.

Στα σχήματα Ad-Hoc ανώνυμων Ερωτηματολογίων οι ομάδες συμμετεχόντων μπορούν να αλλάξουν δυναμικά. Για να μπορέσουμε να χρησιμοποιήσουμε τις Ανιχνεύσιμες Κυκλικές Υπογραφές για την κατασκευή Συστημάτων Ανώνυμων Ερωτηματολογίων τροποποιούμε το παραπάνω σχήμα ώστε να δέχεται δυναμικό σχηματισμό συμμετεχόντων στο Ring. Το σχήμα υπογραφών αυτό το ονομάζουμε Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές.

Τέλος, για τις ανάγκες των ανώνυμων Ερωτηματολογίων χρειαζόμαστε oSE NIZK, οπότε αντί για μετασχηματισμό Fiat-Shamir θα εφαρμόσουμε μετασχηματισμό Pass στο αρχικό Σ-πρωτόκολλο.

Οι τροποποιήσεις που κάνουμε είναι οι εξής:

- Αντί για $h = H(tag, Ring)$ πλέον $h = H(tag)$ στο νέο σχήμα. Έτσι, το σ_i παραμένει πλέον ίδιο και μετά από αλλαγή στο Ring οπότε ο χρήστης μπορεί να γίνει traced.

- Αντί για $A_0 = H'(tag, Ring, m)$ πλέον $A_0 = H'(tag, m)$ στο νέο σχήμα. Έτσι, αν στο αρχικό Ring είχαμε n κλειδιά και έπειτα προστέθηκαν νέα, τότε δύο υπογραφές πριν και μετά στο ίδιο μήνυμα m από το χρήστη i θα έχουν κοινές τις n πρώτες τιμές άρα θα γίνουν linked.

- Για το λόγο αυτό, πλέον τροποποιούμε τον αλγόριθμο $Trace$ ώστε να γίνονται linked δύο υπογραφές αν έχουν ≥ 2 κοινές τιμές αντί για όλες ίδιες όπως ήταν πριν.

- Αντί για Fiat-Shamir η απόδειξη μηδενικής γνώσης παράγεται με τη βοήθεια του μετασχηματισμού Pass.

Το σχήμα των Δυναμικών Ανιχνεύσιμων Κυκλικών Υπογραφών συνολικά φαίνεται παρακάτω.

Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές

Έστω G μία ομάδα με τάξη ένα πρώτο q και γεννήτορα $g \in G$ και $H : \{0,1\}^* \rightarrow G$, $H' : \{0,1\}^* \rightarrow G$, $H'' : \{0,1\}^* \rightarrow \mathbb{Z}_q$ τρία τυχαία μαντεία. Έστω $L = (tag, pk_{[n]})$ (στο παρελθόν ίσως ήταν λιγότεροι οι χρήστες και προστέθηκαν αργότερα άλλοι).

- $Gen(1^\lambda)$:εκτελείται από τον παίκτη i.
  Διάλεξε ένα μυστικό κλειδί $sk_i \leftarrow \mathbb{Z}_q$ τυχαία και υπολόγισε το δημόσιο κλειδί $pk_i = g^{x_i}$.
  Υποθέτουμε την ύπαρξη ενός ασφαλούς **PKI** όπου ο i κάνει την εγγραφή του δημομσίου κλειδιού του pk_i.

- $Sign_{sk_i}(L, m)$: is executed by player i.

  1. Υπολόγισε το $h = H(tag)$ και το $\sigma_i = h^{sk_i}$
  2. Υπολόγισε τα $A_0 = H'(tag, m)$, $A_1 = \left(\frac{\sigma_i}{A_0}\right)^{1/i}$ και για κάθε $j \neq i$ υπολόγισε τα $\sigma_j = A_0 A_1^j$.
     $\sigma_{[n]} = (\sigma_1, ..., \sigma_n) = (A_0^{\frac{i-1}{i}} h^{sk_i \frac{1}{i}}, A_0^{\frac{i-2}{i}} h^{sk_i \frac{2}{i}}, ..., h^{sk_i}, ..., A_0^{\frac{i-n}{i}} h^{sk_i \frac{n}{i}})$
  3. Για τη γλώσσα $\mathcal{L} = \{(L, h, \{\sigma_1, ..., \sigma_n\}) | \exists sk_i$ s.t. $pk_i = g^{sk_i}$ και $\sigma_i = h^{sk_i}\}$ υπολόγισε ένα oSE NIZK π

  output: Η υπογραφή είναι $\sigma = (A_1, \pi)$

- $Verify_L(m, \sigma)$: εκτελείται από τον καθένα

  1. Υπολόγισε τα $h = H(tag)$, $A_0 = H'(tag, m)$, $\sigma_j = A_0 A_1^j$ for each $i \in [n]$
  2. Έλεγξε αν η απόδειξη π είναι ορθή $Ver(\pi) = 1$

  Αποτέλεσμα: 1 αν η τελευταία εξίσωση ισχύει 0 αλλιώς.

- $Trace_L(m, \sigma, m', \sigma')$: όπου $\sigma = (A_1, c_{[n]}, z_{[n]})$ και $\sigma' = (A'_1, c'_{[n]}, z'_{[n]})$

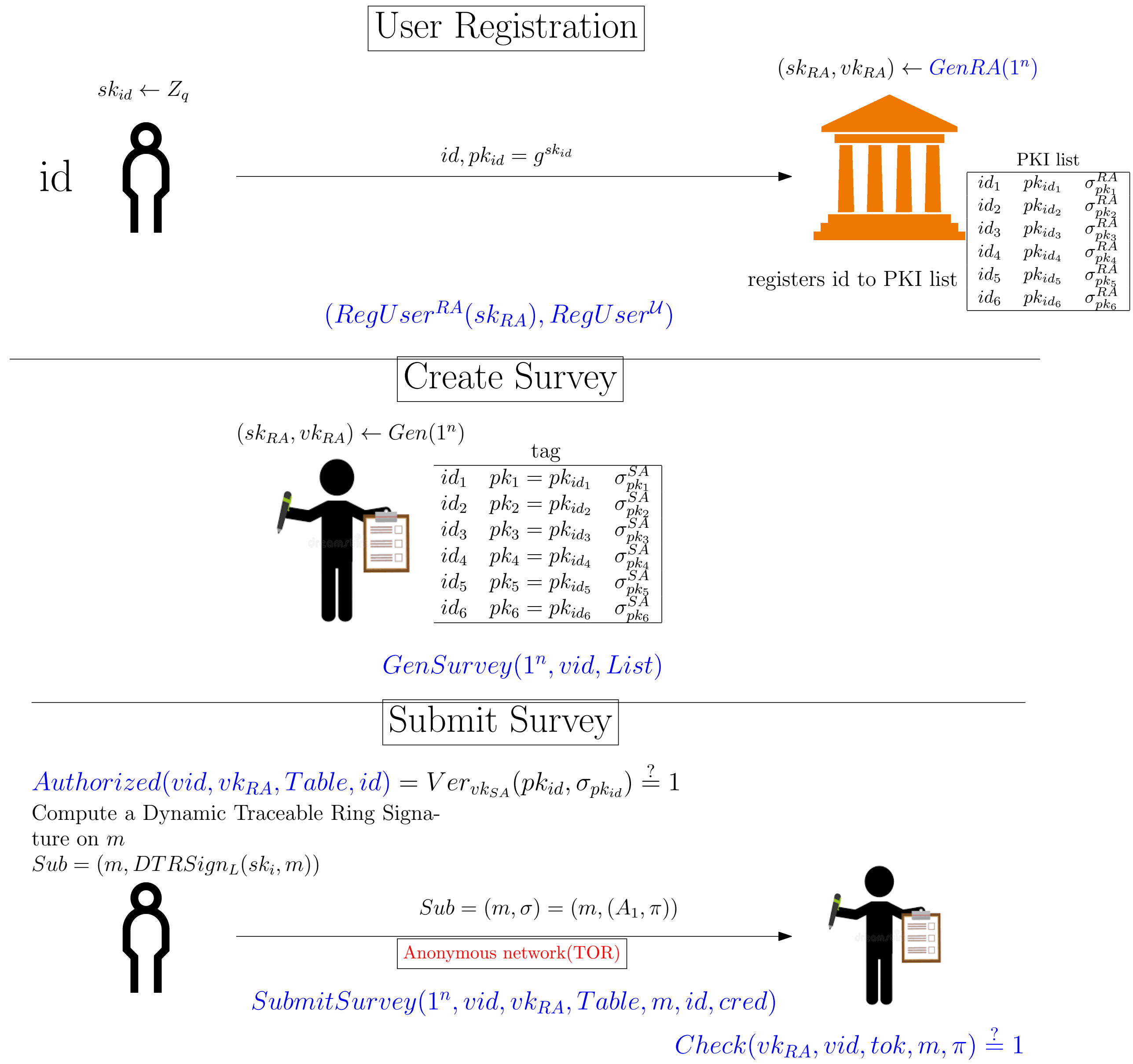
  1. Υπολόγισε τα $h = H(tag)$, $A_0 = H'(tag, m)$, $\sigma_j = A_0 A_1^j$ για κάθε $i \in [n]$
     και τα $A'_0 = H'(tag, m')$, $\sigma'_j = A'_0 A_1'^j$ για κάθε $i \in [n]$
  2. Σύγκρινε τα σ_i και σ'_i για κάθε $i \in [n]$

$$output = \begin{cases} pk_i & \text{if } \sigma_i = \sigma'_i \text{ and } \sigma_j \neq \sigma'_j \text{ for all } j \neq i \qquad \textbf{(1 exactly equal)} \\ \text{linked} & \text{if } \sigma_i = \sigma'_i \text{ for all } i \in [n] \qquad \textbf{(}1 < \textbf{equal)} \\ \text{indep} & otherwise \qquad \textbf{(0 exactly equal)} \end{cases}$$

Επισημαίνουμε ότι για το σχηματισμό του challenge στο oSE NIZK παραμένει ως όρισμα το Ring όπως πριν $c = H''(L, A_0, A_1, \cdot, \cdot)$. Αυτό δεν επηρεαζεί την Ανιχνευσιμότητα αφού ο αλγόριθμος Trace δεν εξετάζει την απόδειξη γνώσης.

Αποδεικνύεται ότι το σχήμα έχει τις απαιτούμενες ιδιότητες ασφαλείας: Tag-Linkability, Anonymity και Exculpability.

Πρόταση για Σύστημα Ανώνυμων Ερωτηματολογίων Χρησιμοποιούμε τις Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές για να κατασκευάσουμε ένα σύστημα Ανώνυμων Ad-Hoc Ερωτηματολογίων. Η ιδέα είναι ότι για την υποβολή της απάντησης ο χρήστης χρησιμοποιεί το σχήμα υπογραφών για να υπογράψει την απάντηση m. Τέλος, ο χρήστης κατά την εγγραφή του στέλνει το δημόσιο κλειδί του και η Αρχή Εγγραφής είναι υπεύθυνη για την πιστοποίηση.


User Registration
$(sk_{RA}, vk_{RA}) \leftarrow GenRA(1^n)$
$sk_{id} \leftarrow Z_q$
id
$id, pk_{id} = g^{sk_{id}}$
PKI list
id_1 pk_{id_1} $\sigma^{RA}_{pk_1}$
id_2 pk_{id_2} $\sigma^{RA}_{pk_2}$
id_3 pk_{id_3} $\sigma^{RA}_{pk_3}$
id_4 pk_{id_4} $\sigma^{RA}_{pk_4}$
id_5 pk_{id_5} $\sigma^{RA}_{pk_5}$
id_6 pk_{id_6} $\sigma^{RA}_{pk_6}$
registers id to PKI list
$(RegUser^{RA}(sk_{RA}), RegUser^{\mathcal{U}})$
Create Survey
$(sk_{RA}, vk_{RA}) \leftarrow Gen(1^n)$
tag
id_1 $pk_1 = pk_{id_1}$ $\sigma^{SA}_{pk_1}$
id_2 $pk_2 = pk_{id_2}$ $\sigma^{SA}_{pk_2}$
id_3 $pk_3 = pk_{id_3}$ $\sigma^{SA}_{pk_3}$
id_4 $pk_4 = pk_{id_4}$ $\sigma^{SA}_{pk_4}$
id_5 $pk_5 = pk_{id_5}$ $\sigma^{SA}_{pk_5}$
id_6 $pk_6 = pk_{id_6}$ $\sigma^{SA}_{pk_6}$
$GenSurvey(1^n, vid, List)$
Submit Survey
$Authorized(vid, vk_{RA}, Table, id) = Ver_{vk_{SA}}(pk_{id}, \sigma_{pk_{id}}) \overset{?}{=} 1$
Compute a Dynamic Traceable Ring Signature on m
$Sub = (m, DTRSign_L(sk_i, m))$
$Sub = (m, \sigma) = (m, (A_1, \pi))$
Anonymous network(TOR)
$SubmitSurvey(1^n, vid, vk_{RA}, Table, m, id, cred)$
$Check(vk_{RA}, vid, tok, m, \pi) \overset{?}{=} 1$

Το σύστημα αυτό προτείνεται με βασικότερο στόχο να αντιμετωπίσει την επίθεση στην πιστοποιησιμότητα που περιγράφηκε στο κεφάλαιο 3.2.1. Αυτό επιτυγχάνεται διότι πλέον κάθε υπογραφή που βάζει η Αρχή Εγγραφής είναι δημόσια. Στο Anonize η υπογραφή μπαίνει ως μάρτυρας στην απόδειξη μηδενικής γνώσης και δε φαίνεται πουθενά. Αντίθετα, στο παραπάνω προταθέν σχήμα κάθε ιδιωτικό κλειδί έχει και ένα δημόσιο το οποίο δίνεται ως είσοδος για την επαλήθευση της υπογραφής. Άρα η Αρχή Εγγραφής δεν μπορεί να περάσει ψεύτικα κλειδιά διότι ο χρήστης που επαληθεύει δε θα τα χρησιμοποιήσει.

4.2 Ένα σύστημα Ανώνυμων Ερωτηματολογίων από Μικρές Συνδέσιμες Κυκλικές Υπογραφές

Οι Μικρές Συνδέσιμες Κυκλικές Υπογραφές [5, 12] είναι παραλλαγές των Κυκλικών Υπογραφών οι οποίες λειτουργούν σαν αυτές αλλά επιτρέπουν Σύνδεση μεταξύ δύο υπογραφών αν αυτές έχουν μπει από το ίδιο άτομο. Δηλαδή, αν κάποιος μέσα από το Ring βάλει 2 υπογραφές (στα ίδια ή σε διαφορετικά μηνύματα) τότε όλοι μπορούν να καταλάβουν ότι πρόκειται για υπογραφές από το ίδιο άτομο, χωρίς όμως να ξέρουν ποιο είναι αυτό το άτομο. Επίσης, είναι μικρές με την έννοια ότι το μέγεθος τους είναι σταθερό ως προς τον αριθμό των συμμετεχόντων στο Ring. Οι βασικές διαφορές από τις υπογραφές του προηγούμενου κεφαλαίου είναι πως δεν είναι πλέον Ανιχνεύσιμες αλλά Συνδέσιμες και πως το μέγεθος τους είναι σταθερό αντί για γραμμικό.

Για την κατασκευή τους χρησιμοποιείται ένας Συσσωρευτής με πεδίο μονής κατεύθυνσης [79]. Οι Συσσωρευτές με πεδίο μονής κατεύθυνσης είναι ένα εργαλείο που χρησιμοποιείται για να ”συσσωρεύσει” πολλές τιμές σε μία μόνο τιμή αφήνοντας παράλληλα ένα αποδεικτικό συσσώρευσης. Χρησιμοποιήθηκε στην Κρυπτογραφία για να κατασκευαστούν Μικρές Κυκλικές Υπογραφές [79]. Αυτοί, μέσω της συσσώρευσης, επιτρέπουν στην απόδειξη (μηδενικής γνώσης) συμμετοχής στο Ring να έχει πλέον σταθερό μέγεθος αντί για γραμμικό. Κι αυτό διότι αντί για n εξισώσεις κάποιος αρκεί να δώσει απόδειξη αντιστοίχισης σε μία συσσωρευμένη τιμή μέσω μίας μόνο εξίσωσης.

Τέλος, για να έχουν επιπλέον και την ιδιότητα της Συνδεσιμότητας προστέθηκε και ένα μοναδικό αναγνωριστικό (token) [5, 12]. Έτσι, κάθε φορά που ένας χρήστης υπογράφει ένα μήνυμα (στο ίδιο Ring και event) θα προκύπτει το ίδιο μοναδικό αναγνωριστικό και θα ξέρουμε ότι η υπογραφή προέρχεται από το ίδιο άτομο.

Παρακάτω δίνονται οι υλοποιήσεις του Συσσωρευτή και της κατασκευής του μοναδικού αναγνωριστικού όπως δίνονται στα [79] και [5, 12] αντίστοιχα.

Η τετράδα $(\{F_\lambda\}, \{X_\lambda\}, \{Z_\lambda\}, \{R_\lambda\})$ είναι ένας Συσσωρευτής με πεδίο μονής κατεύθυνσης:

- $F_\lambda = \{f\}$ όπου $f : (\mathbb{Z}_n^*)^2 \times \mathbb{Z}_{n/4} \to (\mathbb{Z}_n^*)^2, \qquad f(u,x) = u^x \pmod n$
- $X_\lambda = \{e$ πρώτος$|\, (\frac{e-1}{2} \in RSA_\ell) \wedge |e - 2^\ell| < 2^\mu\}$, όπου $\lambda - 2 > \ell$
- $Z_\lambda = \{(e_1, e_2)|e_1, e_2$ είναι διαφορετικοί $\ell/2$-bit πρώτοι $\wedge\ |e_2 - 2^{l/2}| < 2^\mu\}$
- $R_\lambda = \{(x, (e_1, e_2)) \in X_\lambda \times Z_\lambda | x = 2e_1e_2 + 1\}$

Το μοναδικό αναγνωριστικό παράγεται ως εξής:
$tok = \theta(e_1, e_2) = \tilde{g}^{e_1+e_2}$, όπου $\tilde{g}$ είναι ένα τετραγωνικό υπόλοιπο mod N.
Από τη συνάρτηση $\theta(\cdot)$ απαιτούμε να είναι δύσκολο για κάποιον που γνώριζει δύο δημόσια κλειδιά pk_0, pk_1 και ένα $z = \theta(sk_b)$ να μπορεί να καταλάβει πάνω σε ποιο από τα δύο ιδιωτικά κλειδιά εφαρμόστηκε η θ για να παραχθεί το z.

Η απαίτηση αυτή ικανοποιείται για τη συγκεκριμένη υλοποίηση της θ υποθέτοντας ότι ισχύει η εξής υπόθεση:

Definition 4.1. *(Link Decisional RSA Assumption) Έστω ένας N, λ-bit ακέραιος και γινόμενο δύο ασφαλών πρώτων, ένας γεννήτορας g του συνόλου των τετραγωνικών υπολοίπων mod N $QR(N)$, $n_0 = p_0q_0$ και $n_1 = p_1q_1$, όπου p_0, q_0, p_1, q_1 είναι πρώτοι με $poly(\lambda)$-μέγεθος. Κανένας PPT* *αλγόριθμος δεδομένων των N, g, n_0, n_1 και $g^{p_b+q_b}$, όπου $b \in \{0,1\}$ τυχαίο bit, δεν μπορεί να υπολογίσει b' τ.ω. $b' = b$ με πιθανότητα μη-αμελητέα κοντά στο $1/2$.*

Τέλος, το σχήμα Μικρών Συνδέσιμων Κυκλικών Υπογραφών δίνεται παρακάτω:

Μικρές Συνδέσιμες Κυκλικές Υπογραφές

- $Init(1^\lambda)$: διαλέγει τις παραμέτρους του Συσσωρευτή $desc$ και $\tilde{g} \in QR(N)$. Αυτές είναι οι παράμετροι $param$.

- $Key-Gen(1^\lambda, desc)$: ο υπογράφων i εκτελεί τον αλγόριθμο Sample του Συσσωρευτή W για να πάρει ένα ζεύγος δημοσίου-ιδιωτικού κλειδιού $(sk_i, pk_i) = ((e_1, e_2), (2e_1e_2 + 1))$.
  Μετά υπολογίζει μία απόδειξη μηδενικής γνώσης $PoK\{sk_i : (pk_i, sk_i) \in \mathcal{R}\}$ και στέλνει το pk_i, π στην αρχή που πιστοιποιεί τα δημόσια κλειδιά (CA).
  Αν η CA επαληθεύσει την απόδειξη στέλνει μία πιστοποίηση για το pk_i και την αποθηκεύει στη λίστα πισοποιήσεων.

- $Sign_{sk_i}(Ring, param, m)$: εκτελείται από τον υπογράφων i.

  1. Υπολόγισε $w = f(u, pk_{[n]} \setminus \{pk_i\}) = u^{pk_1 \cdots pk_{i-1} pk_{i+1} \cdots pk_n}$ and $v = f(u, pk_{[n]}) = w^{pk_i}$
  2. Υπολόγισε:

$$\sigma' = SPK \left\{ \begin{matrix} (w, (e_1, e_2), pk_i) : pk_i = 2e_1e_2 + 1 \ \wedge \ |pk_i - w^\ell| < 2^\mu \ \wedge \\ \wedge \, |e_1 - 2^{\ell/2}| < 2^\mu \ \wedge \ w^{pk_i} = v \ \wedge \\ \wedge \ tok = \tilde{g}^{e_1+e_2} \end{matrix} \right\} (m)$$

  Αποτέλεσμα: Η απόδειξη είναι $\sigma = (v, tok, \sigma')$

- $Verify_{pk_{[n]}, \tilde{g}}(m, \sigma)$: ελέγχει εάν $v = u^{pk_1, \cdots, pk_n}$ και εάν η απόδειξη σ' είναι έγκυρη για το m.

- $Link_L(m_1, \sigma_1, m_2, \sigma_2)$: είναι linked ανν $tok_1 = tok_2$

Προτείνουμε ένα σύστημα Ανώνυμων Ερωτηματολογίων όμοιο με αυτό της προηγούμενης ενότητας με τη μόνη διαφορά ότι αντί για Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές για την υποβολή των απαντήσεων χρησιμοποιούνται οι Μικρές Συνδέσιμες Κυκλικές Υπογραφές. Αλλά για την παραγωγή τους αντί για μετασχηματισμό Fiat-Shamir εφαρμόζουμε μετασχηματισμό Pass.

Στην ουσία, το σύστημα αυτό αποτελεί μία βελτίωση του προηγούμενου αφού έχει όμοιες ιδιότητες αλλά οι απαντήσεις πλέον έχουν σταθερό μέγεθος ως προς το πλήθος των συμμετεχόντων στο ερωτηματολόγιο αντί για γραμμικό που είχαμε στο προηγούμενο κεφάλαιο. Τέλος, επισημαίνουμε ότι και το σύστημα αυτό αμύνεται στην επίθεση Πιστοποιησιμότητας του κεφαλαίου 3.2.1.

4.3 Μία Ιδέα για Αδυναμία-Απόδειξης κάτω από περιορισμούς

Στην ενότητα 3.2.2 επισημάναμε την ανάγκη για αδυναμία εξαγοράς απαντήσεων στα συστήματα Ερωτηματολογίων. Η αντίστοιχη ιδιότητα ονομάζεται Αδυναμία-Απόδειξης. Το ενδιαφέρον είναι ότι σε πολλές περιπτώσεις μπορούμε να θεωρήσουμε ότι η αρχή του Ερωτηματολόγιου επιθυμεί την Αδυναμία-Απόδειξης, όπως στην Περίπτωση του Brave Browser του κεφαλαίου 3.2.2.

Στην ενότητα αυτή, παρουσιάζουμε μία ιδέα για προσθήκη της ιδιότητας Αδυναμίας-Απόδειξης, θεωρώντας την Αρχή του Ερωτηματολογίου τίμια ως προς την ιδιότητα αυτή. Δηλαδή, θεωρούμε ότι τηρεί το πρωτόκολλο σε ότι αφορά την Αδυναμία-Απόδειξης. Ως προς τις ιδιότητες της Ανωνυμίας και της Πιστοποιησιμότητας θεωρούμε ότι η Αρχή του Ερωτηματολογίου είναι κακόβουλη. Δηλαδή, ότι δεν ακολουθεί το πρωτόκολλο και κάνει ότι μπορεί για να σπάσει την Ανωνυμία και την Πιστοποιησιμότητα.

Τέλος, η ιδέα που θα παρουσιάσουμε αφορά απλά ερωτηματολόγια με απλές ερωτήσεις, οι απαντήσεις των οποίων είναι αριθμοί και ότι τα αποτέλεσματα που εξάγονται προκύπτουν από απλή επεξεργασία των απαντήσεων, εφαρμόζοντας γραμμικές πράξεις.

Η ιδέα είναι η εξής:

1. Ο χρήστης στέλνει την απάντηση του m κανονικά, χρησιμοποιώντας το Anonize στην Αρχή του Ερωτηματολογίου $Sub = (tok, m, \pi)$.

2. Η Αρχή του Ερωτηματολογίου δεσμεύεται στο m, $c \leftarrow Com(m)$ και υπολογίζει μία απόδειξη μηδενικής γνώσης καθορισμένου επαληθευτή η οποία επίσης είναι online Simulation-Extractable (oSE) π' αποδεικνύοντας ότι $Com(m) = c$.

3. Ο χρήστης χρησιμποιεί το Anonize ξανά για να καταθέσει το κρυπτοκείμενο δέσμευσης ως απάντηση $Sub = (tok, c', \pi'')$

4. Αν οι δύο καταθέσεις έχουν ίδιο tok αλλά διαφορετικό $c' \neq c$ τότε η Αρχή του Ερωτηματολογίου απορρίπτει την κατάθεση. Αλλιώς αποδέχεται και δημοσιοποιεί την κατάθεση.

Όταν η Αρχή του Ερωματολογίου θελήσει να υπολογίσει το αποτέλεσμα όλες οι απαντήσεις προστίθενται ομομορφικά και έπειτα το αποτέλεσμα ανοίγει:

$$Res = Open(c_1 \cdot ... \cdot c_n) = Open(Com(m_1) \cdot ... \cdot Com(m_n)) = Dec(Com(m_1 + ... + m_n)) = \sum_{i=1}^{n} m_i$$

Ένα παράδειγμα υλοποίησης είναι με χρήση του σχήματος Δέσμευσης Pedersen. Αν $c = g^m h^r$ τότε η Αρχή του Ερωματολογίου μπορεί να βασίσει το ZK proof στο πρωτόκολλο του Schnorr [26] απόδειξη γνώσης εκθέτη π' ότι: ZK$\{r : \frac{c}{g^m} = h^r\}$.

Τέλος, η απόδειξη π' πρέπει να είναι Καθορίσμενου Επαλθευτή διότι ο χρήστης δεν πρέπει να είναι σε θέση να τη χρησιμοποιήσει για να αποδείξη σε ένα τρίτο ότι $c = Com(m)$. Μία συγκεκριμένη πρόταση για Απόδειξη Καθορισμένου Επαληθευτή είναι η η Διαψεύσιμη Απόδειξη Μηδενικής Γνώσης του Pass[20].

Αδυναμία-Απόδειξης Από τη στιγμή που η Αρχή του Ερωτηματολογίου δεν αποκαλύπτει την αρχική κατάθεση $Sub = (tok, m, \pi)$ κανένας χρήστης δεν μπορεί να πουλήσει την ψήφο του (την έχουμε θεωρήσει τίμια ως προς την Αδυναμία-Απόδειξης). Κι αυτό διότι ένας τρίτος από το c δεν μπορεί να λάβει κάποια πληροφορία που να αφορά το m. Επίσης, η απόδειξη μηδενικής γνώσης εφόσον είναι Καθορισμένου Επαληθευτή δεν μπορεί να πείσει κάποιον τρίτο, αγοραστή ψήφου, ότι $\frac{c}{g^m} = h^r$.

Από την άλλη, το Anonize διατηρεί την Ανωνυμία και την Πιστοποιησιμότητα ακόμα και εναντίον κακόβουλης αρχής του Ερωτηματολογίου. Τέλος, μπορούμε αντί για το Anonize να χρησιμοποιήσουμε τα προτεινόμενο σχήματα Ανώνυμων Ερωτηματολογίων που παρουσιάσαμε στο κεφάλαιο αυτό. Με τον τρόπο αυτό το σύστημα αμύνεται και στην Επίθεση Πιστοποιησιμότητας και έχει και Αδυναμία-Απόδειξης, κάτω από τις υποθέσεις που αναφέραμε στην αρχή.

Κεφάλαιο 5

Μελλοντική Έρευνα

Οι κατευθύνσεις για μελλοντική έρευνα είναι οι εξής:

1. **Ad-hoc Ερωτηματολόγια με Αδυναμία-Απόδειξης:** Μας ενδιαφέρει ιδιαίτερα η έννοια της Αδυναμίας-Απόδειξης και η προσθήκη της ιδιότητας αυτής χωρίς να θεωρούμε κάποια έμπιστη αρχή.

2. **Χρήση των σχημάτων Ad-hoc Survey για ψηφοφορίες:** Μία άλλη κατεύθυνση είναι η μελέτη ενδεχόμενης χρήσης των Ad-Hoc ερωτηματολογίων για Ψηφιακές Ψηφοφορίες και ενδεχόμενες τροποποιήσεις των υπάρχοντων συστημάτων Ad-Hoc Ερωτηματολογίων για να έχουν τις απαραίτητες ιδιότητες ασφαλείας για εκλογές.

3. **Ερωτηματολόγια με κατάθεση απαντήσεων ως κρυπτοκείμενα:** Επισημάναμε την ανάγκη για ερωτηματολόγια με απαντήσεις ως κρυπτοκείμενα στην ενότητα 3.2.5. Προυσίασαμε σύντομα μία ιδέα που αφορά απλά αποτελέσματα στην ενότητα 4.3. Όμως, μας ενδιαφέρει το ίδιο σενάριο για πιο περίπλοκα αποτελέσματα ερωτηματολογίων και αποτελεί ενδεχόμενη κατεύθυνση για μελλοντική μελέτη.

4. **Συσσωρευτής για Δυναμικές Ανιχνεύσιμες Κυκλικές Υπογραφές:** Αξίζει να εξεταστεί αν μπορεί να υπάρξει Συσσωρευτής για να μειώσει το μέγεθος της απόδειξης μηδενικής γνώσης των Δυναμικών Ανιχνεύσιμων Κυκλικών Υπογραφών. Με τον τρόπο αυτό, θα αποκτούσαμε υπογραφές σταθερού μεγέθους.

5. **Υλοποίηση:** Φυσικά, μία υλοποίηση είναι πάντα επιθυμητή για τα κρυπτογραφικά σχήματα. Τόσο για πρακτική δοκιμή του συστήματος και της αποδοτικότητας του όσο και για πραγματική χρήση.

Part II

English text

Anonymous Digital Survey Systems and Cryptographic Ring Signatures

Chapter 1

Introduction

1.1 A few words about Cryptography

Generally, we may say that Cryptography is a field that resolves trust problems among people. Historically, at first Cryptography was a way to hide written messages from a potential adversary. Later, with the arrival of computers Cryptography turned to the digital world, and was a way to encrypt electronic messages.

Nowadays, the field of Cryptography is related to almost anything that has to do with the digital world. However, it is related to real life trust problem. And as more and more significant physical acts move to computers and the Internet more and more trust conflicts appear. For example, money transactions, electronic Voting and message exchange are only a few acts that are performed through the Internet. Additionally, private data are exponentially growing in the web.

The initial solution to protect privacy and integrity is nearly always to put trust to a party that promises to keep its word and protect the data. Cryptography's goal is to remove the need to trust a party and replace it with computational and mathematical protocols that are commonly accepted and in many cases proven to be safe.

1.2 Anonymous electronic Surveys

A problem that arises in a huge number of situations today is anonymous data collection from targeted groups. The main goals in this setting are Anonymity and Authenticity. That is, the collector should not be able to link the data collected to a person and at the same time ensure that only targeted users can participate in the data collection. From now on we will refer to the data asked as Survey and to the data given as Survey Collection/Submission/Answer, even if it is not a survey in the classic sense. For example if a website keeps analytics from a user we may still call it Survey Collection. Essentially, from data perspective it is the same process.

The motivation to preserve Anonymity is clear. Even in questions with less significance (e.g. most preferable tv programs) users are likely to be influenced if they are non-anonymous, so we will have biased data and misleading results. This problem is magnified in more sensitive data (e.g. medical data) or situations where users are afraid to state their opinion, where the survey result may be completely unworthy. That's why both users and survey collectors desire Anonymity. On the other hand, Authenticity is necessary, because the survey creator wants a specific group of people to answer to the survey (e.g. people between 18 and 30 years old), otherwise the survey is of no interest at all!

The typical setting in real life is that the user (non-anonymously) passes the survey to the survey collector and the latter promises that she will maintain user's anonymity. This scenario assumes that we trust the collector. Of course in many cases the 'promise' is bound to law, but still we may have reasons not to trust her. For example, it is possible that the survey collector herself falls victim to (cyber or physical) theft and thief leaks our sensitive data.

Generally, this is the reason to build cryptographically secure systems that do not require any physical trust (such as legal). And for this, we understand that Anonymous Survey Collection is problem within the scope of Cryptography.

The real problem is that the properties described are conflicting. We are asking for a person to identify herself and at the same time be anonymous! We point out that it is trivial to construct a system possessing only Anonymity or only Authenticity. But it seems hard to obtain one possessing both, without putting some trust to some party (users or authorities).

A work that deals with this problem is Anonize, introduced in 2014 by Hohenberger, Myers, Pass and shelat in [6]. That is an environment that allows Surveys' initiators to ask for certain anonymous data from specified users and users to submit them anonymously, but at the same time only authorized users can submit answers. Anonize is, apart from great theoretical study on anonymous surveys, also an implemented system. It has been tested in real life conditions and constitutes a liable tool to practical problems. Also, to writer's knowledge, until today it is the only practical tool that is theoretically grounded and deals directly with anonymous questionnaires problem. Thus, it has deeply influenced this thesis.

1.3 Relation with Electronic Voting Systems

A problem that intersects with Cryptography is construction of secure electronic Voting Systems. That is creating protocols and tools that would allow voters to securely vote remotely with electronic devices (voting specific or general purpose like smartphones and electronic computers). That is a problem that, unlike anonymous questionnaires, has been extensively studied from cryptographers and still research keeps going.

One may observe that e-Voting and Anonymous Surveys are two quite similar problems. We can view anonymous questionnaires as voting, but with a more complex vote. The basic requirements are the same: Anonymity and Authenticity. Although this is a simple model as elections have much more security requirements, the general idea is similar.

In fact this work has a beginning from e-Voting systems and more specifically the work of Pagourtzis, Grontas, Zacharakis and Zhang [7, 8] and especially [9], which study the problem of e-Voting systems emphasizing on everlasting privacy and Coercion Resistance. That is privacy that cannot be violated even in the distant future when more powerful computers are going to appear (e.g. quantum computers). Coercion Resistance is a highly desired property of elections, in general, stating that it should not be able to force a voter's choice even if coercer is physically present in the voting moment (with some constraints). As Anonymous Surveys are not studied as much, the above work on elections was motivational.

On the other hand, e-Voting and electronic Surveys have differences. The basic obstacle to apply directly e-Voting systems to Anonymous Surveys is that usually the first ones consist of two distinct phases. A phase where the user non-anonymously gets a credential, that carries no link to the identity, that will allow the user to vote and the voting phase, where the credential is used and user casts her vote. These phases are separated with a time lag, say a day. That time lag prevents time-correlation attacks that would de-anonymize the voter. If there were no time lag then it would be possible that right after getting her credential the voter will cast her vote and everyone could correlate the credential with the vote.

Although the above works for e-Voting, it would be highly impractical to apply a time lag to Anonymous Survey submissions. That is because it is unlike that a user would wait such a long time to submit a survey. The reason behind this is that elections is something that, in general, is taken more seriously from people so putting a long time lag (say a day) is applicable. Conversely, questionnaires often is something taken lightly and even people are reluctant to submit, much less wait a day to submit. And even if one is willing to submit, once she got her credential she may forget to use the system again to submit the answer one day later. Finally, elections usually have time constraints, for example parliament elections usually occur between 7am and 7pm, which allows phase separation, while questionnaires may occur for months or even years. This observation was pointed out in the Anonize paper [6].

Finally, as elections is something that has a great impact to society and is considered by people of great importance e-Voting systems must have great security properties. On the other hand, depending on the applications, questionnaires may be practical while possessing less security properties. However, the level of security of electronic Survey Submission scheme is debatable and requires caution.

For all these reasons, the need for an application specific scheme for questionnaires and data feedback appears. But still, much ideas are taken from e-Voting paradigm.

1.4 Useful notions

Throughout this reading we are going to use some notion and notation that we find appropriate to clarify now. In general, we adopt many concepts and notations from [10], which an interested reader can consult for more details.

- **PPT algorithm:** Probabilistic Polynomial time Algorithm.

- **Language L:** A set of strings with a specified structure. The structure may described by a property.

- **Witness Relation:** R_L a relation that is efficiently computable and $(x, y) \in R_L$ if x is a witness that $y \in L$. We denote $R_L(y)$ as the set of all witnesses x that $y \in L$.

- **Negligible function:** $\mu : \mathbb{N} \leftarrow \mathbb{R}$ is negligible if for every positive polynomial $poly(n)$ there is a big enough $n_0 > 0$ s.t. for every $n > n_0$

$$\mu(n) < \frac{1}{poly(n)}$$

For example 2^{-n} is a negligible function.

- **Probability Ensemble:** A family of random variables $\{X_i\}_{i \in I}$, where $I \subseteq \mathbb{N}$ is a set of indices.

- **Statistically Close:** Two random variables X, Y over a finite domain D are statistically close if

  $$\sum_{t \in D} |Pr[X = t] - Pr[Y = t]| = negl(\lambda)$$

  where λ is a security parameter.
  Correspondingly, two probability ensembles $\{X_n\}_{n \in I}, \{Y_n\}_{n \in I}$ over a finite domain D are statistically close if

  $$\sum_{t \in D} |Pr[X_n = t] - Pr[Y_n = t]| = negl(n)$$

- **Use as a black box:** the notation is $A^{B(\cdot)}$ where A uses B as a black-box. This means that A has access to (polynomial in number) queries to $B(\cdot)$ for inputs of her choice.

1.5 Thesis Organization

In chapter 2 we provide the basic cryptographic background that we are going to build on in next chapters.

Chapter 3 is about Zero-Knowledge proofs. This is a cryptographic primitive that is very useful as a component of complex cryptographic protocols. Anonymous Surveys require Zero Knowledge proofs with strong security properties and this is the reason we emphasize on providing the theory behind Zero Knowledge proofs that are necessary to build secure Anonymous electronic Survey systems.

In chapter 4 we describe Anonize[6], the Anonymous Survey Collection System that we referred to on section 1.2. Furthermore, we make some observations that motivate our work in the subsequent chapter.

Chapter 5 is the contribution of our work. We propose two Ad-hoc Survey Collection Schemes based on ring signatures. More specifically, we present Traceable Ring Signatures [11], we propose a variant, Dynamic Traceable Ring Signatures and we provide a proof of security for this primitive. Then we propose an Anonymous Ad-hoc Survey Scheme based on Dynamic Traceable Ring Signatures. Secondly, we present Short Linkable Ring Signatures [12, 5] and we propose an Anonymous Ad-hoc Survey Scheme based on them. Finally, we propose a change on Anonize protocol that is in the direction of adding receipt-freeness under constraints.

In Chapter 6 we summarize this thesis and describe directions for future work.

Chapter 2

Cryptographic background

2.1 Indistinguishability

The notion of computational indistinguishability is central to the theory of cryptography. We say that two probability ensembles are indistinguishable if no probabilistic polynomial time algorithm is able to decide whether a value came from a random variable from the first or the second ensemble. This notion was introduced in the context of ciphertexts to formalize security of encryption schemes [13], but is now used to formalize many cryptographic schemes.

Definition 2.1. *Two probability ensembles $X = \{X_n\}_{n \in \mathbb{N}}$ and $Y = \{Y_n\}_{n \in \mathbb{N}}$ are computationally indistinguishable if for every PPT distinguisher algorithm D there exist a negligible function $negl$ such that:*

$$|Pr[D(X_n, 1^n) = 1] - Pr[D(Y_n, 1^n) = 1]| \leq negl(n)$$

2.2 Cryptographic assumptions

Modern Cryptography principles require proofs of security for any scheme. So the steps to establish a cryptographic protocol are: first design the protocol operation, then state the necessary security properties and afterwards prove that the protocol holds them. A great majority of constructions cannot be unconditionally proven secure, instead rely on other assumptions. These assumption are mostly from the computational complexity theory. The reason that we rely on such assumptions is that the computational complexity theory field has not answered significant questions. For example a scheme may rely on the assumptions that integer factorization is in NP class and consequently on the assumption that $P \neq NP$. Of course, many conjectures are rational to believe, so this is a decent method to proceed.

Generally, we can separate cryptographic assumptions into two types the ’general’ and the ’concrete’ ones. ’General’ assumptions are for example one-way functions existence, enhanced trapdoor permutation existence or even commitment scheme existence. These assumptions, mostly, help us construct abstract schemes. On the other, hand, ’concrete’ assumptions assume hardness of a specific problem. For example, Discrete Logarithm Problem hardness, Integer Factorization hardness and Bilinear Decisional Diffie-Hellman hardness are some example assumptions that some cryptographic protocols consider.

Considering usual and simple assumptions to construct a cryptographic schemes is, obviously, much preferable than considering odd ones. This is because common assumptions have been extensively studied and practically tested and are still considered true. On the other hand, the lack of study of unusual assumptions leave the possibility of later discovery of ways to break them.

Finally, we mention that after considering assumptions one has to prove that these assumptions imply that the scheme is secure. The typical way to do this is by showing a reduction from the protocol security to the, generally accepted, assumptions taken. This goes like this: we show that if the proposed protocol is not secure then someone can solve a problem, which is commonly assumed hard. Then our protocol is indeed secure. The above thought of process is a reduction. In next chapters we will see some examples of reductions in cryptography.

2.2.1 Discrete Logarithm Problem

The discrete logarithm problem (DLOG) is one of the most well studied problems, regarding cryptography. The problem is this:

Definition 2.2 (DLOG). *Given $\mathbb{G}$, a cyclic group of order q with a generator g and $h \in \mathbb{G}$ output an $x \in \mathbb{Z}_q$ s.t. $g^x = h$*

This is a problem that is considered hard for big enough q. The best algorithm has $\tilde{O}(\sqrt{q})$ complexity. So we need exponential (in the security parameter) size of groups. That's because q, in cryptographic schemes, is very large, for example $q = 2^{256}$, so even $\sqrt{q} = 2^{128}$ is prohibitive. The (desired) output of DLOG x is denoted as $Log_g(h)$

2.2.2 Computational Diffie-Hellman Problem

Computational Diffie-Hellman Problem is closely related to Diffie-Hellman key exchange [14], a key exchange protocol that affected strongly modern cryptography. Given a cyclic group $\mathbb{G}$, of order q and generator g, one party (say Bob) generates a secret $b \leftarrow \mathbb{Z}_q$ randomly and the other (say Alice) generates $a \leftarrow \mathbb{Z}_q$. Bob sends $B = g^b$ and Alice sends $A = g^a$. Then they both can compute the key $K = g^{ab} = A^b = (g^a)^b = (g^b)^a = B^a$. The amazing part of this scheme is that all the exchanges may occur in public and yet nobody (except Alice and Bob) learns the private key!

Obviously, if one could solve the DLOG then she would be able to find a, b and consequently the key. But DLOG hardness assumption is not enough to preserve the security of the exchange. Even if the adversary cannot solve DLOG, she may be able to find a y s.t. $y = g^{ab}$ given only $A = g^a, B = g^b$ and not a, b. The last is the so-called Computational Diffie-Hellman Problem and it is assumed hard for a PPT algorithm to solve.

Definition 2.3 (CDH). *Given $\mathbb{G}$, a cyclic group of order q with a generator g and $y_1 = g^{x_1}, y_2 = g^{x_2}$ output $y \in \mathbb{G}$ s.t. $y = g^{x_1 \cdot x_2}$*

Again, Computational Diffie-Hellman is well studied and is widely considered hard. So, one may safely construct a cryptographic scheme based on CDH assumption. Furthermore, it is easy to prove that CDH is easier (or at most has the same difficulty) than the DLP. If one can solve DLP then she can also solve CDH.

2.2.3 Decisional Diffie-Hellman Problem

Decisional Diffie-Hellman Problem is the decisional variant of the computational problem. Given three values $y_1, y_2, y \in \mathbb{G}$ can we decide whether or not y is the corresponding Diffie-Hellman key? $(y_1 = g^a, y_2 = g^b, y = g^\gamma) \in \mathbb{G}^e$ is called a Diffie-Hellman tuple if $\gamma = a \cdot b$. The formal definition of the Decisional-Diffie Hellman Problem is below:

Definition 2.4. *Given $\mathbb{G}$, a cyclic group of order q with a generator g distinguish between $(g^a, g^b, g^{a \cdot b})$ and (g^a, g^b, y), where $(g^a, g^b, g^{a \cdot b})$ is a DH tuple and $y \leftarrow \mathbb{G}$ is a random element.*

Decisional Diffie-Hellman is considered hard in some groups, although, as we will see later, it is easy in some others. Finally, DDH is easier than (or at least as hard as) CDH.

$$DDH \leq CDH \leq DLOG$$

2.3 Random Oracle Model

Another significant notion of modern cryptography is the random oracle model(ROM), introduced by Bellare and Rogaway [15]. Theoretically, a random oracle is what its name states: a ”magic” black box which receives an input x and answers unpredictably an output $H(x)$. By unpredictably, we mean uniformly at random. Furthermore, the random oracle if queried on the same input for a second time it must return the same output. So, we can view it as a function.

The random oracle model is used to prove security of protocols. If a protocol is provably secure in the random oracle model then all parties have oracle access to a random oracle. They query it on inputs and it provides answers. In that case, we demand that the random oracle is chosen randomly and that random oracle queries are kept secret. However, when constructing the security reduction, in many cases, we let a simulator (a theoretical entity necessary for the reduction) observe the random oracle and program the random oracle, i.e. choose the answers to the queries made. The model without a random oracle existence is called standard model (or ’plain’ model).

The random oracle model is a highly controversial cryptographic tool. That is because, it is not possible to instantiate it in reality. Firstly, once the random oracle is instantiated with a real function $\hat{H}$ the adversary, theoretically speaking, knows the answer to all inputs! Of course, we rely on the fact that the adversary is polynomially bounded and thus is not able to compute all the exponential in number outputs. Anyway once given to the adversary the random oracle is not random anymore. Another thing is that for the reduction, as stated above, the simulator holds the random oracle and thus sees the queries. But if one is given the random oracle function $\hat{H}$ there is actually no need to query the RO to get an answer, she will compute it herself and the simulator will not learn the query. In fact, Canneti, Goldreich and Halevi in [16] proved that there exist schemes secure on the random oracle model, though insecure for every possible implementation of the RO!

Additionally, in practice random oracles are instantiated with hash functions (e.g. SHA-2). Although some hash functions are widely considered safe this is not proven. And even if they are safe as hash functions they may not be safe as random oracle functions. For example maybe the output is not uniformly distributed. And finally maybe it is safe to use a hash function as a random oracle instantiation for a scheme Π and still be unsafe for another scheme Π'. So safety in practice relies heavily on safety of the hash function.

So the question is why do we use the random oracle model? First of all, there are many protocols that cannot be proved secure in the standard model but can be proved secure in the random oracle model. And it is believed that a proof in the random oracle model is better than no proof at all. Furthermore, in many cases schemes in the random oracle model are much more efficient in practice. And usually an efficient scheme in the random oracle model is more preferable than inefficient in the standard model. Finally, in practice, there are no real attacks to schemes in the random oracle.

We state that random oracles in theory inherently hold the standard hash function properties of first and second preimage resistance and collision resistance.

2.4 Pseudorandom Functions

In some cases what we want to achieve is a random function. That is a function randomly picked from the set of all possible functions (with a fixed domain and range). Say we are interested in functions $f : \{0,1\}^n \rightarrow \{0,1\}^n$. Let the set of these kind of functions be $\mathbf{Func}_n$. f's domain includes 2^n inputs that take n-bit strings as output. Thus, a function f can be represented by a lookup table of 2^n rows with n columns each, so $n \cdot 2^n$ bits. This leads us to the conclusion that there are $2^{n \cdot 2^n}$ different f functions, so $|\mathbf{Func}_n| = 2^{n \cdot 2^n}$.

A truly random function is a function picked uniformly from $\mathbf{Func}_n$, with probability $1/2^{n \cdot 2^n}$. One certain way to generate a random function would be to fill randomly every input. But, as stated above the inputs (2^n in number) are way too much to compute and store. That's why we would like to find an indirect way to generate it. This is done by a cryptographic primitive called pseudorandom function families, introduced by Goldreich, Golwasser and Micali [17].

A pseudorandom function family is a collection of functions $\{F_k : \{0,1\}^n \rightarrow \{0,1\}^n\}$, that choosing a random key $k \leftarrow \{0,1\}^n$ give a random function. Say the resulting distribution is F. Intuitively, the distribution F must be indistinguishable from the ideal random function choice distribution. Of course, two distributions cannot be the same. For example a pseudorandom function F_k has 2^n possible results while a truly random $2^{n \cdot 2^n}$. But we demand that no PPT algorithm can distinguish them.

Definition 2.5. *Let $F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ be an efficient function. We say that F is pseudorandom if for all PPT distinguishers D there exists a negligible function $negl(\cdot)$ s.t.:*

$$\left| Pr[D^{F_k(\cdot)}(1^n) = 1] - Pr[D^{f_n(\cdot)}(1^n) = 1] \right| \leq negl(n)$$

where k is chosen uniformly at random from $\{0,1\}^n$ and f_n is chosen uniformly at random from $\mathbf{Func}_n$.

One observation is that PPT D cannot take as input the functions themselves because that would require a $n \cdot 2^n$ -bit input and exponential checks. That's why we let the distinguisher have oracle access to the functions. Finally, we state that pseudorandom function can be constructed under standard cryptographic assumption.

2.5 Commitment Schemes

In some cases, there is a need for some party to commit to a value, keeping it secret, until it decides to reveal it. This is what a commitment scheme does. It involves two parties, a sender S and a receiver R. A commitment scheme consist of two phases [18]:

- **Commit Phase:** Sender commits to a value m and sends the commitment $c \leftarrow Com(m)$ to the receiver.
- **Reveal Phase:** Sender decommits her initial commitment and sends the value m to the Receiver. Then Receiver checks if c is truly a commitment of m.

To understand it better, a physical analogy is this: we (the sender) have a hidden item and commit to it putting it in a locked box (we only have the key) and we give the box to the receiver. At a later time we choose to reveal the item, so we open the box.

Sometimes, in commitment schemes maybe the decommitment value may differ from m. In this case S sends $d(m)$ and not the actual message. The simpler scenario presented above is the one we will come across in this work.

Furthermore, we note that the above description is about non-interactive commitment schemes, where the communication is in one round. There are commitment schemes of multiple rounds, though we will not concern ourselves with them in this work. So, whenever we say commitment schemes, we will, actually, refer to non-interactive ones.

Two characteristics that are essential in the above setting is that sender wants to ensure that her message is kept secret until she desires to reveal it. That is, the commitment does not help the receiver to find m. This property is called hiding. On the other hand, receiver wants to ensure that sender is truly commited to the value m and cannot cheat in the reveal phase presenting another m' as the commited value. This property is called binding. Hiding and Binding are the two security properties that every commitment scheme must hold. Of course, Correctness must, also, hold (as in any cryptographic protocol).

Definition 2.6. $\Pi = (Gen, Com, Ver)$ *is:*

1. *Perfect Hiding if the probability ensembles* $\{Com_{ck}(m_0)\}, \{Com_{ck}(m_1)\}$ *are identical.*
2. *Statistical Hiding if the probability ensembles* $\{Com_{ck}(m_0)\}, \{Com_{ck}(m_1)\}$ *are statistically close.*
3. *Computational Hiding if the probability ensembles* $\{Com_{ck}(m_0)\}, \{Com_{ck}(m_1)\}$ *are computationally indistinguishable.*

For the binding property we define the following experiment (Fig.2.1):

Definition 2.7. $\Pi = (Gen, Com, Ver)$ *is:*

1. *Perfect Binding if for every computationally unbounded adversary* $\mathcal{A}$*:*
   $Bind_{\mathcal{A},\Pi}(n) = 0.$

Binding Experiment $Bind_{\mathcal{A},\Pi}(n)$

1) $ck \leftarrow Gen(1^n)$

2) $(com, m_0, m_1, d_0, d_1) \leftarrow \mathcal{A}(ck)$

$$output = \begin{cases} 1 & \text{if } Ver(com, m_0, d_0) = Ver(com, m_1, d_1) = 1 \text{ and } m_0 \neq m_1 \\ 0 & otherwise \end{cases}$$

Figure 2.1: Binding Experiment

2. *Statistical Binding if for every computationally unbounded adversary* $\mathcal{A}$*:* $Bind_{\mathcal{A},\Pi}(n) \leq negl(n)$.

3. *Computational Binding if for every PPT adversary* $\mathcal{A}$*:* $Bind_{\mathcal{A},\Pi}(n) \leq negl(n)$.

Definition 2.8. $\Pi = (Gen, Com, Ver)$ *is a commitment scheme if it has correctness, computational hiding and computational binding. Additionally,* Gen, Com *and* Ver *should be PPT algorithms.*

Commitment schemes are extremely useful cryptographic tools both as stand-alone protocols and as ingredients to build other cryptographic protocols (e.g. zero knowledge proofs). So, from theoretic scope they have been studied extensively and at the same time have been used in many real life cryptographic applications.

2.5.1 Pedersen commitment scheme

Pedersen commitment scheme is a perfectly hiding and computationally binding. It was introduced by Pedersen in 1991 [19] and its security is based on the Discrete Logarithm assumption.

Pedersen Commitment Scheme

Let $\mathbb{G}$ be a group of prime order q with generator g. Pedersen commitment scheme is a tuple of algorithms (Gen, Com, Ver):
Setup Receiver R executes $Gen(1^n)$ and generates $h \in \mathbb{G}$.
Commit Sender S chooses randomly $r \in \mathbb{Z}_q$, computes $Com(m; r) = g^m h^r = c$ and sends it to R.
Reveal S sends the decommitment (m', r') to R. Normally an honest sender sends the original $(m', r') = (m, r)$. R executes

$$Ver(c, m', r') = \begin{cases} 1 & \text{if } c = g^{m'} h^{r'} \\ 0 & otherwise \end{cases}$$

Figure 2.2: Pedersen perfectly hiding commitment scheme

Theorem 2.9. *The above is a commitment scheme, which is perfectly (information-theoretically) hiding and computationally binding, assuming DLP problem is hard, for any* $m \in \mathbb{Z}_q$ *and any* r

chosen uniformly at random from $\mathbb{Z}_q$.

Proof. **Perfect hiding** : for any $m, m' \in \mathbb{Z}_q$, $Com(m_0)$ and $Com(m_1)$ follow identical distribution.
$Pr[Com(m_0) = c] = Pr[g^m h^r = c] = Pr[h^r = cg^{-m}] = Pr[r = Log_h(cg^{-m})] \overset{r,r'\text{ uniformly random}}{=}$
$Pr[r' = Log_h(cg^{-m'})] = Pr[h^{r'} = cg^{-m'}] = Pr[h^{r'} g^{m'} = c] = Pr[Com(m_1) = c]$

Computational binding : Assume a PPT adversary $\mathcal{A}$ algorithm can choose (m_0, r_0, m_1, r_1) s.t. $g^{m_0} h^{r_0} = g^{m_1} h^{r_1}$ and $(m_0, r_0) \neq (m_1, r_1)$ to win the binding game $Bind_{\mathcal{A},\Pi}(n)$. Let $h = g^x$ (g is generator of $\mathbb{G}$).

$$g^{m_0} g^{xr_0} = g^{m_1} g^{xr_1} \Leftrightarrow g^{m_0 + xr_0} = g^{m_1 + xr_1} \Leftrightarrow m_0 + xr_0 = m_1 + xr_1 \Leftrightarrow x = \frac{m_1 - m_0}{r_0 - r_1}$$

So $\mathcal{A}$ can solve DLP, which was asummed hard. □

2.5.2 A simple construction using a random oracle

We refer to a second commitment scheme construction in random oracle model [20], which we are going to use in next chapters to construct a zero knowledge proof.

Random oracle Commitment Scheme

Setup Let $RO : \{0,1\}^{2n} \to \{0,1\}^{\ell(n)}$ be a random oracle, where $\omega(log(n)) \leq \ell(n) \leq poly(n)$.
Commit Sender S chooses randomly $r \in \{0,1\}^n$, computes $Com(m; r) = RO(m, r) = c$ and sends it to R.
Reveal S sends the decommitment (m', r') to R. Normally an honest sender sends the original $(m', r') = (m, r)$. R executes

$$Ver(c, m', r') = \begin{cases} 1 & \text{if } c = RO(m', r') \\ 0 & otherwise \end{cases}$$

Figure 2.3: A commitment scheme based on random oracles

Hiding: The above commitment scheme is computationally hiding because an adversary can find the inverse of $c = RO(m, r)$ only by making an oracle query on input (m, r). But this is extremely unlikely to happen because every query has (considering uniform random oracle) $\frac{1}{2^{\ell(n)}}$ chances of success, and adversary makes only polynomial number of queries.
Binding: The above commitment scheme is computationally binding because the adversary in order to break binding need to find a collision $(m', r') \neq (m, r)$ s.t. $RO(m', r') = c = RO(m, r)$, which, of course, happens with negligible probability (due to collision resistance property).

2.6 Digital Signatures

A Digital Signature [21] is a cryptographic primitive that allows a user to authenticate a message of her choice. Intuitively, a Digital Signature acts as a paper Signature. Apart from authenticating the identity of the signer, Digital Signatures, also, preserve the integrity of the message: the user signs a specific message. The obvious goal that Digital Signatures have to achieve is that only the signer can create a Signature and everyone can verify it.

Definition 2.10. *A Digital Signature Scheme is a triple of PPT algorithms* $(Gen, Sign, Ver)$ *such that:*
Let $\mathcal{M}$ *be the message space.*

- Gen *is the key generation algorithm that takes as input the security parameter* 1^n *and outputs a pair of keys the secret (private) key and the corresponding verification (public) key:* $(sk, vk) \leftarrow Gen(1^n)$*.*
- $Sign$ *is the message signing algorithm that takes a message (from* $\mathcal{M}$*) as input and produces a signature on the message using the secret key:* $\sigma \leftarrow Sign_{sk}(m)$
- Ver *is the signature verification algorithm that takes as input a signature and a message and checks if the signature is correct using the verification key (and outputs 0 or 1):* $Ver_{vk}(m, \sigma) \in \{0, 1\}$

Correctness: We require that for every n *and every* $m \in \mathcal{M}$ *with overwhelming probability* $Ver_{vk}(m, Sign_{sk}(m)) = 1$

Security of Digital Signature Schemes. As mentioned before the basic security notion that Digital Signatures should capture is similar to real-life: nobody should be able to forge a signature, even if she can ask for signatures on messages of her choice. Basically, there are 3 types of attacks:

- Universal Forgery: The adversary can generate a signature to any message of her choice.
- Selective Forgery: The adversary can generate a signature to 1 (meaningful) message of her choice.
- Existential Forgery: The adversary can generate a signature to random (possibly meaningless) message.

It is clear that security against existential forgery is the strongest one. Next we give the definitions of selective and existential Unforgeability [22].

Definition 2.11. *A signature scheme* $\Pi = (Gen, Sign, Ver)$ *has selective Unforgeability under adaptive chosen-message attack (SUF-CMA) if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $negl(n)$ *such that:*

$$Pr[Sig - forge_{\mathcal{A},\Pi}^{SUF-CMA}(n) = 1] \leq negl(n)$$

Definition 2.12. *A signature scheme* $\Pi = (Gen, Sign, Ver)$ *has existential Unforgeability under adaptive chosen-message attack (EUF-CMA) if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $negl(n)$ *such that:*

$$Pr[Sig - forge_{\mathcal{A},\Pi}^{EUF-CMA}(n) = 1] \leq negl(n)$$

Selective Unforgeability Experiment $Sig - forge_{\mathcal{A},\Pi}^{SUF-CMA}(n)$

1) $m^* \leftarrow \mathcal{A}$: $\mathcal{A}$ commits on an $m^* \in \mathcal{M}$, on which she has to forge a signature.
2) $(sk, vk) \leftarrow Gen(1^n)$: $\mathcal{A}$ receives vk.
3) $\mathcal{A}$ has oracle access to $Sign_{sk}(\cdot)$ on any message m of her choice.
4) $(m^*, \sigma^*) \leftarrow \mathcal{A}^{Sign_{sk}(\cdot)}(vk)$: $\mathcal{A}$ finally outputs a possible forgery of a signature on m^*

output: Let Q denote the set of messages that $\mathcal{A}$ queried on the signing oracle:

$$output = \begin{cases} 1 & \text{if } m^* \notin Q \text{ and } Ver_{vk}(m^*, \sigma^*) = 1 \\ 0 & otherwise \end{cases}$$

Figure 2.4: Selective Unforgeability Experiment

Existential Unforgeability Experiment $Sig - forge_{\mathcal{A},\Pi}^{EUF-CMA}(n)$

1) $(sk, vk) \leftarrow Gen(1^n)$: $\mathcal{A}$ receives vk.
2) $\mathcal{A}$ has oracle access to $Sign_{sk}(\cdot)$ on any message m of her choice.
3) $(m, \sigma) \leftarrow \mathcal{A}^{Sign_{sk}(\cdot)}(vk)$: $\mathcal{A}$ finally chooses a message m and outputs a possible forgery of its signature.

output: Let Q denote the set of messages that $\mathcal{A}$ queried on the signing oracle:

$$output = \begin{cases} 1 & \text{if } m \notin Q \text{ and } Ver_{vk}(m, \sigma) = 1 \\ 0 & otherwise \end{cases}$$

Figure 2.5: Existential Unforgeability Experiment

2.6.1 Blind Signatures

A Blind Signature Scheme allows authentication of a message while still keeping its content private. The scheme captures this scenario: a user $\mathcal{U}$ asks for authentication of a message m from an authority (Signer $\mathcal{S}$), but does not want $\mathcal{S}$ to see the message. So she blinds the message sends it to the authority and the authority signs it. After $\mathcal{U}$ gets the blind signature σ' she unblinds it and gets a valid signature σ.

A clear analogy from real world, adopted from [10], is this: user covers a check with a carbon paper lined envelope and sends it to the bank, asking a specific amount to be transfered. The bank signs the outside of the envelope and returns it to the user. User removes the envelop and gets a signed check, which can spend now. Because of the envelope the bank never learned what was the check about. Obviously, a problem with this is that bank can be fooled into signing a 100$ check (check is blind), while was told to transfer 1$. This problem led to a new variant, Partially Blind Signatures, which is presented in the next chapter.

Blind Signatures were introduced by David Chaum in 1983 [23]. From then many schemes have been proposed (e.g. [24, 25, 26]) and its security has been well studied. Blind Signatures have been studied in context of e-Voting Systems and e-Cash, which are their main applications. Also, a great amount of variants have been introduced (Partially Blind Signatures [27], Fair Blind Signatures [28] and recently Conditional Blind Signatures [7])

The formal definition of a blind signature scheme is presented below [29, 23, 30]

Definition 2.13. *A Blind Signature Scheme is a triple $(Gen, Sign, Ver)$ of algorithms and protocols:*

- *Gen is the key generation algorithm that takes as input the security parameter 1^n and outputs a pair of keys the secret (private) key and the corresponding verification (public) key: $(sk, vk) \leftarrow Gen(1^n)$.*

- *$Sign$ is a two-party protocol between a user $\mathcal{U}$ and a signer $\mathcal{S}$ with common input vk. The private input of $\mathcal{U}$ is a message m and the private input of $\mathcal{S}$ is the secret key sk. At the end of the protocol $\mathcal{U}$ obtains a signature σ on m as private output. $\sigma \leftarrow \langle \mathcal{S}(vk, sk), \mathcal{U}(vk, m) \rangle$*

- *Ver is the signature verification algorithm that takes as input a signature and a message and checks if the signature is correct using the verification key (and outputs 0 or 1): $Ver_{vk}(m, \sigma) \in \{0, 1\}$*

The above definition of $Sign$ implicitly states that the user $\mathcal{U}$ uses a $Blind$ algorithm to keep the m private and an $Unblind$ algorithm to change the signature σ' that she receives, thus keep the output $\sigma \leftarrow Unblind(m, \sigma')$ private. In fact, other definitions in the literature consider blind signature schemes as a five-tuple of algorithms $(Gen, Blind, Sign, Unblind, Ver)$.

Security of Blind Digital Signature Schemes. It is natural that the requirement for Unforgeability we had on 'plain' digital signatures in the previous chapter remains. We demand that only the authority (the signer) can authenticate messages.

The problem with the Definition of Existential Unforgeability is that the adversary makes oracle queries to get signatures and afterwards tries to forge one on a new message. But in the present setting we can't say if she got a signature or she forged it (because it was queried while blind).

One way to solve this would be to remove the oracle access from the definition, but this would be an unrealistic assumption. So Pointcheval and Stern gave an Unforgeability definition of Blind Signature Schemes [31] which demands one-more-forgery resistance. This means that if a user gets ℓ signatures then she cannot present more than ℓ (unblinded) signatures afterwards. The definition below follows the formalism from [29] and partially [32, 33].

One-more-forgery Experiment $Sig - onemoreforge_{\mathcal{A},\Pi}(n)$

1) $(sk, vk) \leftarrow Gen(1^n)$: $\mathcal{A}$ receives vk.
2) $\mathcal{A}(pk)$ engages in $\ell = poly(n)$ adaptive, parallel and arbitrarily interleaved interactive protocols $Sign$ with $\mathcal{S}$, where $\mathcal{A}$ decides in an adaptive fashion when to stop.
3) $\{(m_1, \sigma_1), (m_2, \sigma_2), ...(m_k, \sigma_k)\} \leftarrow \mathcal{A}(vk)$

$$output = \begin{cases} 1 & \text{if } m_i \neq m_j,\ \forall i \neq j \text{ and } Ver_{vk}(m_i, \sigma_i) = 1,\ \forall i \text{ and } k > \ell \\ 0 & otherwise \end{cases}$$

Figure 2.6: One-more Forgery Experiment

Definition 2.14. *A Blind Signature Scheme* $\Pi = (Gen, Sign, Ver)$ *is unforgeable if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $negl(n)$ *such that:*

$$Sig - onemoreforge_{\mathcal{A},\Pi}(n) \leq negl(n)$$

The second security condition that we want is Blindness. This, informally speaking, means that Signer cannot find out what she is signing. It is random for her. So blindness property protects User from a malicious signer, who tries to violate user's privacy. The same security notion is sometimes called Unlinkability. The definition below follows [32].

Blinding Experiment $Blind_{\mathcal{A},\Pi}(n)$

1) $(vk, m_0, m_1, st_{find}) \leftarrow \mathcal{A}(find, 1^n)$
2) $b \leftarrow \{0, 1\}$: is chosen randomly
3) $st_{issue} \leftarrow \mathcal{A}^{\langle \cdot, \mathcal{U}(vk, m_b)\rangle^1, \langle \cdot, \mathcal{U}(vk, m_{1-b})\rangle^1}(issue, st_{find})$
   $\mathcal{A}$ engages in two parallel (and arbitrarily interleaved) interactive signing protocols with $\mathcal{U}$ on messages m_b and m_{1-b} respectively.
4) set $(\sigma_0, \sigma_1) = (\perp, \perp)$ if $\sigma_0 = \perp$ or $\sigma_1 = \perp$
   If at least one signature is invalid then both (unblinded) signatures are set invalid.
5) $b^* \leftarrow \mathcal{A}(guess, \sigma_0, \sigma_1, st_{issue})$
   $\mathcal{A}$ receives the unblinded signatures and tries to guess which one corresponds to the first message m_0.

$$output = \begin{cases} 1 & \text{if } b^* = b \\ 0 & otherwise \end{cases}$$

Figure 2.7: Blinding Experiment

Definition 2.15. *A Blind Signature Scheme* $\Pi = (Gen, Sign, Ver)$ *is blind if for all PPT adversaries* $\mathcal{A}$ *(working in modes* $find$*,* $issue$ *and* $guess$*), there exists a negligible function* $negl(n)$ *such that:*

$$\left| Blind_{\mathcal{A},\Pi}(n) - \frac{1}{2} \right| \leq negl(n)$$

2.6.2 Partially Blind Signatures

In some cases, an authority apart from signing a message needs to include an attribute to the signed message. For example, as described above, a signature may authorize a certain quantity to a transaction. The problem is that blinding does not allow the signer to verify that the attribute is truly included in the message. Signer only controls the public and private key and not the message signed.

One way to overcome this is to use different public (and corresponding private) key for different attributes. For instance 100$ would have its own key and and 1$ another one. So a cheating user cannot deceive signer to sign a 1$ transaction as 100$, as 1$ message with 100$ public key for verification will not be valid.

But this results to inefficient schemes as every user has to store many verification keys. Particularly, there are applications where a signature is 'valid until' a date, for example expires in two weeks. Then authority has to change keys every two weeks and users have to be updated. This is more obvious in e-Voting schemes, where someone would have to use different keys for each vote!

A partially blind signature scheme comes to solve this kind of problems. The idea is to include an unblinded part in the message, so that signer can see it and decide if it is valid. So now the message consists of, roughly, two parts: an information as plaintext, which we call common information, and a blinded message. The common information is agreed between the user and the signer and is related to an attribute (e.g. it states an expiration date).

Partially blind signatures were introduced by Abe and Fujisaki in 1996 [27] and Abe and Okamoto gave a formal definition [34]:

Definition 2.16. *A Partially Blind Signature Scheme is a triple* $(Gen, \langle\mathcal{S},\mathcal{U}\rangle, Ver)$ *of algorithms and protocols:*

- Gen *is the key generation algorithm that takes as input the security parameter* 1^n *and outputs a pair of keys the secret (private) key and the corresponding verification (public) key:* $(sk, vk) \leftarrow Gen(1^n)$ *.*

- $Sign$ *is a two-party protocol between a singer* $\mathcal{S}$ *and a user* $\mathcal{U}$ *with common input* vk *and* **info***. The private input of* $\mathcal{U}$ *is a message* m *and the private input of* $\mathcal{S}$ *is the secret key* sk*. At the end of the protocol* $\mathcal{U}$ *obtains a signature* σ *on* m *as private output.* $\sigma \leftarrow \langle\mathcal{S}(vk, \textbf{info}, sk),\mathcal{U}(vk, \textbf{info}, m)\rangle$
where **info** *is the common preagreed information.*

- Ver *is the signature verification algorithm that takes as input a signature and checks if it is correct with respect to the message and the common input* $m||\textbf{info}$ *using the verification key (and outputs 0 or 1):* $Ver_{vk}(m, \textbf{info}, \sigma) \in \{0, 1\}$

The security requirements are a direct extension of the classical ones. We need Completeness, Partial Blindness and Unforgeability. Security Definitions were introduced in [34].

Completeness. We require that two parties $\mathcal{S},\mathcal{U}$ that stick to the singing protocol create a signature σ that is verified with overwhelming probability.

Unforgeability. Partially blind signatures must withstand forgery attacks similar to Digital Signatures (as **info** is plaintext) and Blind Signatures (as m is blind) at the same time. So a would be forger $\mathcal{A}$ has two options:

i) forge **info**.

ii) for fixed **info**, after $\ell_{\textbf{info}}$ successful execution of $Sign$ protocol with common information **info**, $\mathcal{A}$ produces $\ell_{\textbf{info}} + 1$ signatures with common information **info**.

The case (i) is the same as case (ii) for $\ell_{\textbf{info}} = 0$. So, security definition of Unforgeability must take into consideration only the second case. We omit the formal definition as given in [34] as it is much like Unforgeability for Blind Signatures, but for any common information **info**.

Partial Blindness. Intuitively, Partial Blindness captures the idea that the blinded part of the message must be kept secret from the signer. So Signer cannot tell if a signature σ came from a message m or from another one m', if both messages were signed with the same common information **info** $=$ **info**$'$.

Again, the only difference from the 'classical' definition of blindness is the common information, which of course should be the same for both signatures. Otherwise, it is clear that Signer can decide which signature was generated from a message, observing the common informations. We refer to [34] for the formal definition.

Finally, we note that partially blind signatures are a generalized notion of blind signatures, as one can transform the first to the latter by fixing common input to a single string (e.g. 0). However, the reverse is not always that easy.

2.6.3 A construction of Partially Blind Signatures

In this section, we present a partially blind signature construction but with a slight difference in the security of blinding. The construction was introduced in the context of Anonize [6]. The scheme uses a digital signature scheme and a commitment scheme. Let's say we have a secure digital signature scheme $(Gen, Sign, Ver)$ and a secure commitment scheme $(Gen_{com}, Com, Ver_{com})$. The Partially blind signature scheme is defined as:

In plain words a user $\mathcal{U}$ blinds her secret message m by committing to it and sends it to the signer $\mathcal{S}$ along with the unconcealed part **info** .Then $\mathcal{S}$ signs them and send the signature σ back to $\mathcal{U}$. We mark that the signature scheme must be able to sign a message in 2 blocks and output a single signature. Finally, the user unblinds the signature by outputting the random tape r used in the commitment scheme along with the original signature σ. The verification is trivial: an unblinded signature (r, σ) for $(\textbf{info}, m)$ is valid if the corresponding blind signature σ on the blinded message $(\textbf{info}, Com_{ck}(m; r))$ is valid. Thus, we infer that one may verify the signature without possessing the unblinded version.

The verification procedure was designed this way because the above scheme does not hold blinding as we defined it above. Instead, it holds a blinding variant called weak blinding. This security notion is similar to the 'traditional'blinding but requires that the adversarial signer $\mathcal{S}^*$ cannot see the unblinded signature. So the user unblinds the signature she gets but does not send

A Partially Blind Signature scheme

Let's say we have a signature scheme $(Gen, Sign, Ver)$ and a commitment scheme $(Gen_{com}, Com, Ver_{com})$. The Partially blind signature scheme is defined as:

- $Gen'(1^n)$: outputs $(vk_{bl}, sk_{bl}) = ((ck, vk), sk)$
  where $ck \leftarrow Gen_{com}(1^n)$ and $(vk, sk) \leftarrow Gen(1^n)$
- $\langle \mathcal{S}, \mathcal{U} \rangle$:
  - $Blind'_r((ck, vk), m) = Com_{ck}(m; r)$
  - $Sign'_{sk}(\textbf{info}, c) = Sign_{sk}(\textbf{info}, c)$
  - $Unblind'_r((ck, vk), \sigma) = (r, \sigma)$
- $Ver'_{(ck,vk)}((\textbf{info}, m), (r, \sigma)) = Ver_{vk}((\textbf{info}, Com_{ck}(m; r)), \sigma)$

Figure 2.8: An abstract construction of a weakly blind Partially blind signature scheme

it to the (potentially malicious) signer $\mathcal{S}^*$. The only information that adversary gets (about the unblinded version) is whether or not the unblinded signature is valid.

We will not include the formal definition of weak blinding, as the only different part from the blinding experiment of Fig.2.7 is that in the step 5 the adversary $\mathcal{A}$ does not get σ_0, σ_1 as input but instead two bits o_1, o_2 indicating whether or not the unblinded signature is verified successfully. The formal definition can be found in the original paper [6].

Of course, weak blinding (as its name points!) is weaker notion than blinding, but the above protocol is a simple one proven secure in the standard model and easy to implement. Furthermore, for the survey collection application weak blinding is sufficient as we will see in next chapters.

Theorem 2.17. *If $(Gen, Sign, Ver)$ is a secure signature scheme and $(Gen_{com}, Com, Ver_{com})$ a secure commitment scheme then the scheme of Fig. 2.8 is partially blind signature scheme (in the sense of weak blinding).*

Proof. **Completeness** is trivial.

Unforgeability: Assume that an adversary $\mathcal{A}$ makes $\ell_{\textbf{info}}$ signature queries $(\textbf{info}, \cdot)$ to $\mathcal{S}$ for a specific **info** and manages to output $\ell_{\textbf{info}} + 1$ signature pairs $\{(\textbf{info}, m_i), (r_i, \sigma_i)\}$ and all m_i are different.

	Blinding ($\mathcal{U}$)	Signing ($\mathcal{S}$)	Unblinding ($\mathcal{U}$)
m_1	$Com(m_1; r_1) = c_1$	$Sign(c_1) = \sigma_1$	(r_1, σ_1)
m_2	$Com(m_2; r_2) = c_2$	$Sign(c_2) = \sigma_2$	(r_2, σ_2)
$\vdots$	$\vdots$	$\vdots$	$\vdots$
m_ℓ	$Com(m_\ell; r_\ell) = c_\ell$	$Sign(c_\ell) = \sigma_\ell$	(r_ℓ, σ_ℓ)
			$(r_{\ell+1}, \sigma_{\ell+1})$

There two cases:

- If all outputted σ_i are different (thus all c_i were different) then this means that the adversary managed to forge a signature. This contradicts the Unforgeability of the secure signature scheme.

- If two outputted $\sigma_i = \sigma_i'$ are the same then they came from the same $c_i = Com(m_i; r_i) = Com(m_i', r_i') = c_i'$ and still $m_i \neq m_i'$ (from hypothesis). This means that the adversary managed to break the binding property of the commitment scheme.

By contradiction the scheme is unforgeable.

Weak blinding: Assume a malicious signer $\mathcal{S}^*$. In the weak blinding experiment $\mathcal{S}^*$ chooses two messages m_0, m_1 (and a common **info**) and send them to the user $\mathcal{U}$, who blinds them (by committing to them) and send them back and $\mathcal{S}^*$ sign them (in blind) and send the signature to $\mathcal{U}$. Finally, $\mathcal{U}$ sends a bit for each signature indicating whether or not the signature is valid.

So what the adversary has is two messages m_0, m_1, two commitments on them c_b, c_{1-b} and two bits o_b, o_{1-b} and needs to find the value of bit b, i.e. whether c_b is a commitment of m_0 or m_1. And obviously two blind signatures σ_b, σ_{1-b}, which created herself.
First of all, it is easy to see that $\mathcal{S}^*$ can simulate herself o_b, o_{1-b}:

$$o_b = Ver_{vk}'((\textbf{info}, m_b), (r_b, \sigma_b)) = Ver_{vk}((\textbf{info}, Com_{ck}(m_b; r_b)), \sigma_b) = Ver_{vk}((\textbf{info}, c_b), \sigma_b)$$

vk, c_b, σ_b are known to $\mathcal{S}^*$ so we conclude that she can simulate o_b. Thus according to the simulation paradigm (which we are going to analyze in the next chapter) she does not gain any knowledge by receiving o_b, o_{1-b}.

To conclude, what the adversary gains is c_b, c_{1-b}. If she can distinguish them then she can break the hiding property of the commitment scheme. So weak blinding holds.

□

2.7 Public Key Infrastructure

Public Key Cryptography brought great potential to encrypted communication. One can encrypt a message with a Bob's public key so that only Bob can decrypt it, with his corresponding private key. So one has to create a public-private key pair and publish the public key. This, in contrast to Private Key Cryptography, disengage the users from private key exchange.

The problem that arises is that the sender wants to be sure that the public key is truly linked to the user. Public key infrastructure is an arrangement that resolves this problem. In that there is a certificate authority (CA). CA is responsible for generating certificates for public keys, usually by signing them with a digital signature. So a user, normally, communicates with CA to verify the identity of a public key and afterwards encrypts a message with that public key.

Although PKI is simple, there are some difficulties in it. First of all, CA should be trusted otherwise it is possible that the public key does not bind to the claimed identity. Secondly, it is possible that the sender may not know how to obtain receiver's public key. Furthermore, it is obvious that the user-receiver should have already setup a public/secret key and register to the PKI. We will discuss in section 2.9.2 an alternative to PKI the Identity-Based Encryption.

2.8 Ring Signatures

We consider a scenario where a message can be authenticated by a group of people, but at the same time the signer's identity should remain hidden. Basically, the signature should only ensure that someone from the group signed, while preserving anonymity. In this sense, group signatures were introduced in 1991 [35]. Group signatures allow a predefined group of people to sign anonymously a message. However, they involve a trusted group manager, who can revoke any signer's anonymity and finally is responsible for setting up the group signature.

Ring Signatures came as a solution to the same problem as group signatures, but without involving any trusted group manager, thus anonymity cannot be revoked. Anyone, without any setup can form a group of people, called ring, in an ad-hoc manner and the group may, also, be expanded at a later instant. Afterwards, any individual in the ring can sign a message anonymously. Furthermore, in group signatures it is difficult to change the group dynamically. Ring signatures settle this as well. The assumption that we make is that every user is associated with a public information (public key or id in id-based ring signatures).

The notion of ring signatures and the first construction came from Rivest, Shamir and Tauman in 2001 [36]. A great number of variants have been proposed, consequently, some of which are threshold ring signatures, linkable ring signatures, traceable ring signatures, verifiable ring signatures and id-based ring signatures. We will look into linkable and traceable ring signatures as they are the ones we find more relevant to Anonymous Questionnaires submission problem.

Concerning security of ring signatures, basically, we require Anonymity and Unforgeability. The reader can find an in-depth study of security of ring signatures and various types of attacks in [37].

The above scheme finds applications in various systems including e-Voting systems, e-Cash systems and, recently, cryptocurrencies. For example, some cryptocurrencies allow a payment to be made by a ring of people anonymously, so they utilize ring signatures. In later chapters we will discuss their potential usefulness in Anonymous Survey Systems.

2.8.1 Rivest-Shamir-Tauman Ring Signature (RSA-based)

For a better grasp, we briefly present here the first ring signature construction[36], which is based on RSA public key cryptosystem [38].

The core of the scheme is a family of functions called combining functions:

Definition 2.18. *A combining function family $C_{k,v}(y_1, ..., y_r)$, which take as input a key k, an initialization value v and arbitrary values $y_1, ...y_r \in \{0,1\}^b$. It uses a Symmetric encryption algorithm E_k as a sub-procedure and produces an output $z \in \{0,1\}^b$ s.t. it has the following properties:*

1. ***Permutation on each input:** Fix $n-1$ inputs to any values y_i, $i \in [r] \setminus \{s\}$ and let the s-th input be variable. Then for each $s \in [n]$ the function $C_{k,v}(y_1, ..., y_{s-1}, \cdot, y_{s+1}, ..., y_r)$ is one-to-one mapping from y_s to z.*

2. ***Efficiently solvable for any single input:*** *for each $s \in [r]$, given z and y_i for all $i \neq s$ it is possible to find efficiently a y_s s.t. $C_{k,v}(y_1, ..., y_r) = z$*

3. ***Infeasible to solve verification equation for all inputs without trapdoors:*** *Given k, v, z it is hard for a PPT adversary to find $x_1, ..., x_r$ s.t. $C_{k,v}(g_1(x_1), ..., g_r(x_r)) = z$ without inverting any of $g_1 ..., g_r$*
   where $g_1 ..., g_r$ are trapdoor one-way permutations.

We assume that each possible signer is associated with a public pk_i via a **PKI** (public key infrastructure). Let the corresponding secret key be sk_i. The public key pk_i implies a trapdoor one-way permutation $g_i : \{0,1\}^b \to \{0,1\}^b$ and sk_i is the trapdoor. The scheme apart from the combining function, also, uses a Symmetric Encryption Algorithm E_k, which is permutation over $\{0,1\}^b$ and a random oracle h.

RST Ring Signature

Let a **PKI** with each user possessing a pair of secret-public key (sk_i, pk_i), a Symmetric Encryption permutation algorithm $E_k : \{0,1\}^b \to \{0,1\}^b$ and a random oracle $h : \{0,1\}^* \to \{0,1\}^b$. Also, a combining function $C_{k,v} : (\{0,1\}^b)^r \to \{0,1\}^b$. The ring signature consists of the algorithms:

- $Ring - Sign_{sk_s}(m, pk_1, ..., pk_r)$:
  - Compute the symmetric encryption key $k = h(m, pk_1, ..., pk_r)$
  - Choose initialization value randomly $v \leftarrow \{0,1\}^b$
  - For all the other ring members $i \neq s$, pick random $x_i \leftarrow \{0,1\}^b$ and compute $y_i = g_i(x_i)$
  - For the signer s (yourself), find the y_s such that $C_{k,v}(y_1, ..., y_r) = v$
  - For the signer s (yourself), use the trapdoor to find $x_s = g_s^{-1}(y_s)$

  Output: $\sigma = (pk_1, ...pk_r, v, x_1, ..., x_r)$

- $Ring - Verify_{pk_1,...,pk_r}(m, \sigma)$:
  - Compute $y_i = g_i(x_i)$, for each $i \in [r]$
  - Compute $k = h(m, pk_1, ..., pk_r)$
  - Check if $C_{k,v}(y_1, ..., y_r) = v$

  Output: 0 or 1 depending on whether or not the last equality holds.

Figure 2.9: Rivest-Shamir-Tauman Signature scheme

The trapdoor one way permutation that is used for message $m = q_i n_i + r_i$ is defined as:

$$g_i(m) = \begin{cases} q_i n_i + f_i(r_i) & \text{if } (q_i + 1)n_i \leq 2^b \\ m & otherwise \end{cases}$$

which is basically an extension of RSA function $f_i(x) = x^{e_i}(mod\, n_i)$, but fixes the domain to b-bits for all ring members. Each member has public key $pk_i = (n_i, e_i)$ as defined in RSA scheme.

What remains now is to instantiate the combining function:

$$C_{k,v}(y_1, ..., y_r) = E_k(y_r \oplus E_k(y_{r-1} \oplus E_k(y_{r-2} \oplus E_k(... \oplus E_k(y_1 \oplus v)...)))) = z$$

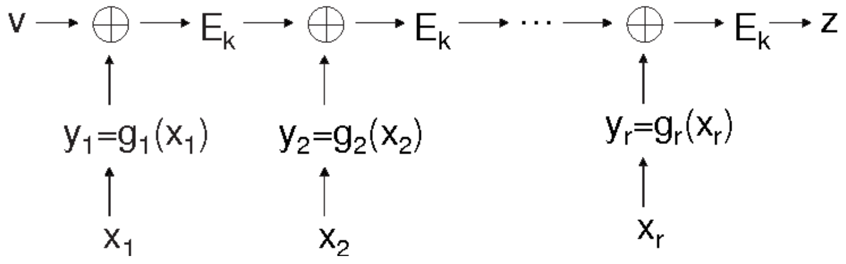


Figure 2.10: The combining function.

Figure 2.10 illustrates the combining function construction. Firstly, it is clear that the above is permutation over $\{0, 1\}^b$ as g_i, E_k and XOR are permutations. Secondly, it is efficiently solvable to find the y_s given all the other y_i, z, k and v as one may run the evaluations from start forward and from finish backwards (E_k^{-1} is easy to compute) to compute the missing y_s. So properties 1 and 2 of combining functions hold. As for property 3, there is a tight reduction shown in the original paper.

The consistency condition $C_{k,v}(y_1, ..., y_r) = v$ leads to a ring shape (see Fig. 2.11), which gave the name of ring signature to the scheme. Many later schemes do not utilize a combining function with a ring shape, nevertheless are still called ring signatures as they hold the same properties as the above scheme. This is, also, the reason that instead of a group we call the set of users $\{pk_1, ..., pk_r\}$ a ring.

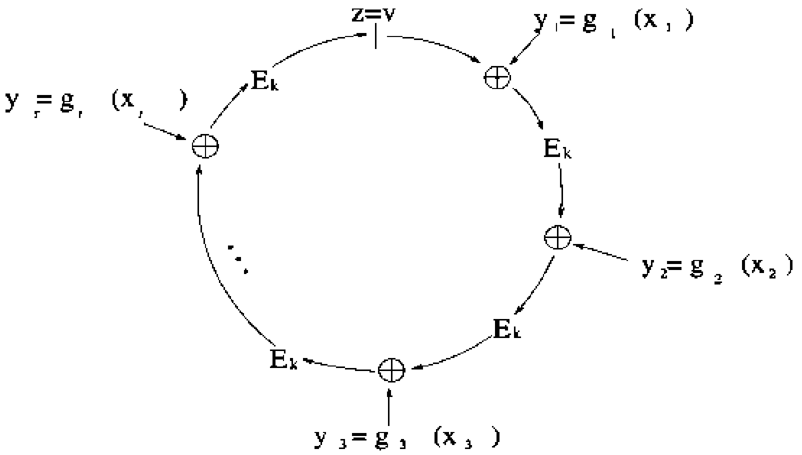


Figure 2.11: The combining function, when setting the output value to be equal to the initialization value.

Security of RST ring signature scheme

- Anonymity: To break anonymity an adversary would have to distinguish x_s from all the other x_i. All x_i but x_s are generated randomly and thus follow the same (uniform) distribution. Then y_i are produced which they uniquely specify y_s and in turn $x_s = g_s^{-1}(y_s)$. So intuitively we get that x_s is generated uniformly at random as a result of random choices of all the other x_i. The key fact is that $C_{k,v}(y_1, ..., y_{s-1}, \cdot, y_{s+1}, ..., y_r)$ is a permutation over $\{0,1\}^b$. So x_s is perfectly indistinguishable from any other x_i and, thus, anonymity holds.

- Unforgeability: It clearly comes from the third property of the combining function and from the hardness of trapdoor one way permutation inversion assumption.

2.9 Bilinear Maps

A subject that has gained tremendous attention from Cryptography researchers in the last two decades is Bilinear Maps. Bilinear Maps are functions that take two points from two (different or not) groups and associate it to a point of another group. A key property of this function is bilinearity. In the literature, one can find them called, also, pairings or bilinear pairings.

Definition 2.19. *Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be two cyclic groups of order q. Function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map if it satisfies:*

(1) (Bilinearity) for all $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and $a, b \leftarrow \mathbb{Z}_q$

$$e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$$

(2) (Non-Degeneracy) If g_1 generates $\mathbb{G}_1$ and g_2 generates $\mathbb{G}_2$ then $e(g_1, g_2)$ generates $\mathbb{G}_T$

(3) (Computability) e can be efficiently conmputed.

Non-degeneracy excludes trivial functions that map every tuple to the same point. An equivalent expression would be $e(g, g) \neq 1$.

Only a few categories of groups have been found to have bilinear maps. The most common constructions are based on Weil or Tate pairings. From cryptographic perspective, and in the present work, we take bilinear maps as black-boxes. We only state that they apply in elliptic curves. An extensive study on elliptic curves and bilinear maps constructions can be found in [39]. Finally, $\mathbb{G}_1$ and $\mathbb{G}_2$ can potentially be the same group and may (or may not) exist an efficient homomorphism between them (they are isomorphic as they have the same order).

For Cryptographic applications the most important property is bilinearity. An impact of bilinearity is that decisional Diffie-Hellman in groups with efficient pairing is easy. Say we have the tuple (g^a, g^b, g^c), where $g \in \mathbb{G}$, and we want to determine if $c = ab$. If there exists a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ then we try the equality $e(g^a, g^b) \overset{?}{=} e(g^c, g)$ if it holds then $e(g^a, g^b) = e(g, g)^{ab} = e(g^{ab}, g) = e(g^c, g)$ so it is a Diffie-Hellman tuple otherwise it is not. So in polynomial time we solve the DDH problem in $\mathbb{G}$ with probability 1. Actually, 'breaking' some elliptic curves was the one of the first applications of bilinear maps in Cryptography [40].

On the other hand, Computational Diffie-Hellman is still considered hard in group with bilinear pairings. Actually, there are groups where Decisional Diffie-Hellman is easy and Computational Diffie-Hellman is as hard as Discrete Logarithm [41]. These groups are called gap Diffie-Hellman groups, regardless on whether or not they have an efficient Bilinear Map.

2.9.1 Relevant Assumptions

Two problems related to groups with bilinear maps are (Decisional) Bilinear Diffie-Hellman Problem and n-(Decisional) Bilinear Diffie-Hellman Inversion. These are two problems that assumed to be hard and are used to construct cryptographic schemes.

Bilinear Diffie-Hellman[42] As stated in the previous section 'classic' Decisional Diffie-Hellman in groups with pairings is easy. So a new problem is defined that generalizes for three exponents.

Let $\mathbb{G}$ be group of prime order q with a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. It has a computational and a decisional variant, exactly as 'classic' Diffie-Hellman.

Definition 2.20 (CBDH)**.** *Given $(g, g^a, g^b, g^c) \in \mathbb{G}^4$ compute $e(g, g)^{abc}$.*

Definition 2.21 (DBDH)**.** *Given the tuples $(g, g^a, g^b, g^c, e(g, g)^{abc})$ and (g, g^a, g^b, g^c, r), where $g, g^a, g^b, g^c \in \mathbb{G}$ and $r \leftarrow \mathbb{G}_T$ random element, distinguish which of them is a real BDH tuple.*

We, also, present the definition of the above problems for asymmetric pairings, i.e. when $\mathbb{G}_1 \neq \mathbb{G}_2$.

Let $\mathbb{G}, \hat{\mathbb{G}}$ be groups of prime order q with a bilinear map $e : \mathbb{G} \times \hat{\mathbb{G}} \rightarrow \mathbb{G}_T$.

Definition 2.22 (CBDH for asymmetric bilinear groups)**.** *Given $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b) \in \mathbb{G}^3 \times \hat{\mathbb{G}}^3$ compute $e(g, \hat{g})^{abc}$.*

Definition 2.23 (DBDH for asymmetric bilinear groups)**.** *Given the tuples $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, e(g, \hat{g})^{abc})$ and $(g, g^a, g^c, \hat{g}, \hat{g}^a, \hat{g}^b, r)$, where $y \leftarrow \mathbb{G}_T$ random element, distinguish which of them is a real BDH tuple.*

n-Bilinear Diffie-Hellman Inversion[43] Let $\mathbb{G}$ be group of prime order q with a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$.

Definition 2.24 (Computational n-BDHI)**.** *Given $(g, g^x, g^{x^2}, ..., g^{x^n}) \in \mathbb{G}^{n+1}$ compute $e(g, g)^{1/x}$*

Definition 2.25 (Decisional n-BDHI)**.** *Given tuples $(g, g^x, g^{x^2}, ..., g^{x^n}, e(g, g)^{1/x}) \in \mathbb{G}^{n+1} \times \mathbb{G}_T$ and $(g, g^x, g^{x^2}, ..., g^{x^n}, r) \in \mathbb{G}^{n+1} \times \mathbb{G}_T$ distinguish which is the one with an inversion.*

Again we redefine for asymmetric bilinear maps: Let $\mathbb{G}, \hat{\mathbb{G}}$ be groups of prime order q with a bilinear map $e : \mathbb{G} \times \hat{\mathbb{G}} \rightarrow \mathbb{G}_T$.

Definition 2.26 (Computational n-BDHI for asymmetric bilinear groups)**.** *Given $(g, g^x, \hat{g}, \hat{g}^x, g^{\hat{x}^2}, ..., g^{x^n}) \in \mathbb{G}^2 \times \hat{\mathbb{G}}^{n+1}$ compute $e(g, \hat{g})^{1/x}$*

Definition 2.27 (Decisional n-BDHI for asymmetric bilinear groups)**.** *Given tuples $(g, g^x, \hat{g}, \hat{g}^x, g^{\hat{x}^2}, ..., g^{x^n}, e(g, \hat{g})^{1/x}) \in \mathbb{G}^2 \times \hat{\mathbb{G}}^{n+1} \times \mathbb{G}_T$ and $(g, g^x, \hat{g}, \hat{g}^x, g^{\hat{x}^2}, ..., g^{x^n}, \qquad r \qquad) \in \mathbb{G}^2 \times \hat{\mathbb{G}}^{n+1} \times \mathbb{G}_T$ distinguish which is the one with an inversion.*

2.9.2 Applications in Cryptography

The first positive applications of bilinear maps in Cryptography, that gained much attention, are one-round 3-party Diffie-Helman key exchange [42] and Boneh and Franklin's identity-based Encryption [44].

Tripartile Diffie-Hellman key exchange in one round The idea is simple: we have three parties that want to exchange a private key over a public channel. Say Alice, Bob and Carol. The problem is a clear generalization of 'classic' two-party Diffie-Hellman. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map over group $\mathbb{G}$ of order q.

- Each Party picks randomly her secret:
  Alice: $a \leftarrow \mathbb{Z}_q$
  Bob: $b \leftarrow \mathbb{Z}_q$
  Carol: $c \leftarrow \mathbb{Z}_q$
- Alice, Bob and Carol broadcast g^a, g^b, g^c respectively.
- Each party compute the final key $e(g, g)^{abc}$
  Alice: $e(g^b, g^c)^a$
  Bob: $e(g^a, g^c)^b$
  Carol: $e(g^a, g^b)^c$

So the idea is that the bilinear map allows each party (even third parties) to 'solve' Diffie-Hellman (not actually solve, but find its corresponding point in $\mathbb{G}_T$). After that nobody can find the key unless she has the secret exponent that is missing, exactly as in Diffie-Hellman.

From this application came the Bilinear Diffie-Hellman problem. It is clear that its hardness ensures security to the above scheme.

Identity-based Encryption Identity-Based Encryption(IBE) was proposed by Shamir in 1984 [45] as an alternative to PKI (section 2.7). The idea is the public key is the identity string itself. So if Alice wants to send an email to Bob "Bob@email.com" she will use that string to encrypt it. Bob may have not yet setup a private key. Afterwards Bob communicates with Private Key Generator (PKG) to get his private key and decrypt the message. Bob should authenticate himself to PKG. PKG uses her master secret key to generate the corresponding private key and it is obvious that should be trusted, because she knows all the private keys! Private Key Generator replaces the Certificate Authority of PKI.

The problem was defined in 1984 and remained an open problem until 2001, when Boneh and Franklin introduced an IBE scheme based on Bilinear Maps[44]. At the same time Cocks introduced an IBE scheme based on quadratic residues [46]. We briefly present the Boneh-Franklin IBE scheme.

Let $\mathbb{G}$ be a group of prime order q with generator g and bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let $h_1 : \{0, 1\}^* \to \mathbb{G}$ and $h_2 : \mathbb{G}_T \to \{0, 1\}^n$, where n is the length of plaintext messages, be two hash functions.

- **Setup**: PKG picks randomly **master-key** $s \leftarrow \mathbb{Z}_q$ and g^s is the public key of PKG.

- **Encrypt**: If Alice wants to send a message $m \in \{0,1\}^n$ to ID (e.g. ”Bob”) she picks $r \leftarrow \mathbb{Z}_q$ and computes:

$$C = (g^r, m \oplus h_2(g_{ID}^r)) \qquad \text{where} \qquad g_{ID} = e(h_1(ID), g^s)$$

- **Extract**: PKG generates the private key of ID and send it to ID: $d_{ID} = h_1(ID)^s$.

- **Decrypt**: ID decrypts $C = (C_1, C_2)$ with her private key d_{ID}:

$$Decrypt((C_1, C_2), d_{ID}) = C_2 \oplus h_2(e(d_{ID}, C_1))$$

Correctness:

$$\begin{aligned} Decrypt(Encrypt(ID,m), Extract(ID)) &= (m \oplus h_2(g_{ID}^r)) \oplus h_2(e(d_{ID}, g^r)) \\ &= m \oplus h_2(e(h_1(ID), g^s)^r) \oplus h_2(e(h_1(ID)^s, g^r)) \\ &= m \oplus h_2(e(h_1(ID), g)^{sr}) \oplus h_2(e(h_1(ID), g)^{sr}) \\ &= m \oplus 0 \\ &= m \end{aligned}$$

The above Encryption Scheme is secure against CCA-attacks and its security is based on Bilinear-Diffie Hellman assumption.

Other applications The above schemes brought huge development in pairing based cryptography in the last two decades. Especially, the Boneh-Franklin scheme, which was seminal as it solved a problem which was open for years. So many variants of Identity based encryptions schemes were developed based on bilinear maps like hierarchical IBE (HIBE). Furthermore, many signature schemes like short, blind, aggregate, ring and unique Signatures are some applications of pairings in Cryptography. Finally, even some new zero-knowledge proof systems use bilinear maps.

Chapter 3

Zero Knowledge Proofs

Zero Knowledge Proofs (ZK proofs) are one of the most important notions of cryptography with a wide range of applications, introduced initially by Goldwasser, Michali and Rackoff[47]. A Zero Knowledge Proof is a protocol that allows one party (the Prover) to convince another party (the Verifier) of the validity of a statement, without revealing any information, other than the validity itself.

3.1 Interactive Proof Systems

An Interactive Proof System is a protocol between two interactive Turing Machines[47, 48], whose goal is to produce a valid proof of a statement. The first Turing machine represents the Prover ($\mathcal{P}$) and the second one the Verifier($\mathcal{V}$).

An interactive Turing Machine(ITM) is a Turing Machine equipped with five tapes: a read-only input tape, a work tape, a random tape, a read-only communication tape and a write-only communication tape. The random tape contains an infinite sequence of random bits.

By $\langle A, B\rangle(x)$ we denote the random variable representing the (local) output of B when interacting with machine A on common input x, when the random input to each machine is uniformly and independently chosen[48].

Definition 3.1. *A pair of Interactive Turing Machines $(\mathcal{P}, \mathcal{V})$ is called an interactive proof system for a language L if $\mathcal{V}$ is polynomial- time and there is a negligible function $negl(\cdot)$ such that the following two conditions hold:*

- *Completeness: for every $x \in L$ there exists a witness w such that:*

$$Pr[\langle\mathcal{P}(w), \mathcal{V}\rangle(x) = 1] \geq 1 - negl(|x|)$$

- *Soundness: for every $x \notin L$ and every ITM $\mathcal{P}^*$:*

$$Pr[\langle\mathcal{P}^*, \mathcal{V}\rangle(x) = 1] \leq negl(|x|)$$

In case where $\mathcal{P}^*$ is restricted to be PPT, $(\mathcal{P}, \mathcal{V})$ is called Interactive Argument System.

3.2 Zero Knowledge Proof - Definitions

The above definition does not refer to any information leaked to the Verifier at all. So the next step is to define an Interactive Proof System that, additionally, prevents any information leakage. But before doing this, we have to define what is Zero Knowledge.

The Simulation Paradigm. Intuitively, an Interactive Proof System is Zero Knowledge (ZK) if whatever a (possibly cheating) Verifier $\mathcal{V}^*$ learns after the interaction with the Prover $\mathcal{P}$ can be, also, learned without interacting with $\mathcal{P}$. So we require that the output of an Interactive Proof between a Prover and any Verifier cannot be distinguished from a simulated output, that did not come from interaction. That implies that Zero Knowledge is equivalent to the existence of an (efficient) Simulator S.

We give below definitions of zero-knowledge introduced by Goldreich and Oren [49] that takes into account any prior knowledge that the Verifier may have (auxiliary input):

Definition 3.2. *An Interactive Proof System $(\mathcal{P}, \mathcal{V})$ for a language L, with witness relation R_L, is* ***(auxiliary input) zero-knowledge*** *if for every PPT interactive machine $\mathcal{V}^*$ there exists a PPT algorithm S (the Simulator) such that the following two ensembles are computationally indistinguishable:*

- $\{\langle \mathcal{P}(w), \mathcal{V}^*(aux)\rangle(x)\}_{x\in L, aux\in\{0,1\}^*}$ *for* $w \in R_L(x)$
- $\{S(x, aux)\}_{x\in L, aux\in\{0,1\}^*}$

If these two ensembles are statistically close then $(\mathcal{P}, \mathcal{V})$ is ***statistical zero-knowledge****.*
If these two ensembles are identical then $(\mathcal{P}, \mathcal{V})$ is ***perfect zero-knowledge****.*

A slight detail in the above definition is that the Simulator depends on the Verifier. So differently acting Verifiers may have different Simulators. The notion of a single Simulator for each possible Verifier is a variant called black-box Simulation Zero Knowledge. The black-box Simulator uses $\mathcal{V}^*$ as a black-box to simulate the interaction with $\mathcal{P}$:

Definition 3.3. *An Interactive Proof System $(\mathcal{P}, \mathcal{V})$ for a language L, with witness relation R_L, is* ***black-box zero-knowledge*** *if there exists a PPT algorithm S such that for every polynomial $p(n)$ and for every PPT interactive machine $\mathcal{V}^*$ using $p(n)$ random coins the following two ensembles are computationally indistinguishable:*

- $\{\langle \mathcal{P}(w), \mathcal{V}^*(aux)\rangle(x)\}_{x\in L, aux\in\{0,1\}^*}$ *for* $w \in R_L(x)$
- $\{S^{\mathcal{V}^*}(x, aux)\}_{x\in L, aux\in\{0,1\}^*}$

If these two ensembles are statistically close then $(\mathcal{P}, \mathcal{V})$ is ***black-box statistical zero-knowledge****.*
If these two ensembles are identical then $(\mathcal{P}, \mathcal{V})$ is ***black-box perfect zero-knowledge****.*

Both definition described let the verifier cheat. So a protocol has to be strong enough to prevent attacks from malicious verifiers, who do not keep the protocol and their moves depend on the Prover's moves. Another variant of zero-knowledge, though weaker, is **honest-verifier zero-knowledge (HVZK)** [50]. In this setting, we require that the verifier executes the protocol faithfully. So Simulator uses a Verifier that sticks to the protocol, as a black-box, to produce a transcript identically distributed to the one produced by conversation of $\mathcal{P}$ and $\mathcal{V}$.

A result that makes zk Proofs significant for Cryptography is that every NP Language has a zero knowledge proof $NP \subseteq ZK$. The result came from Goldreich, Micali and Wigderson [51] and has brought many practical applications of zk Proofs.

3.3 Proofs of Knowledge

Until now we have talked about proving the validity of a statement. A stronger requirement would be that someone, additionally, can prove possession of a witness for the statement. This notion is formalized by the introduction of a machine called extractor E. E should be able to extract a witness each time a malicious prover $\mathcal{P}^*$, who does not posses a witness, convinces $\mathcal{V}$. This is done by using $\mathcal{P}^*$ as a black-box.

Proof of Knowledge first appeared as an idea in [47] and was extensively studied and defined in [52].

Definition 3.4. *An Interactive Proof System $(\mathcal{P}, \mathcal{V})$ for a language L is **proof of knowledge** if there exists a PPT machine E and a negligible function $negl(n)$ such that for every polynomial $p(n)$ and for every PPT interactive machine $\mathcal{P}^*$ using $p(n)$ random coins:*

$$Pr[\langle \mathcal{P}^*(aux), \mathcal{V}\rangle(x) = 1] < Pr[E^{\mathcal{P}^*}(x, aux) \in R_L(x)] + negl(n)$$

for every $x \in \{0,1\}^*$ and every auxiliary input aux

We need to clarify that proof of knowledge does not imply zero-knowledge. In fact, these are two independent notions. That, of course, does not prevent us from combining them to construct a zero-knowledge proof of knowledge in one protocol.

Proofs of knowledge are very useful tools for authentication systems and anonymous credentials. Thus, as we will see later, are suitable for applications like e-Voting Systems and Anonymous Questionnaire Systems.

3.3.1 Σ-Protocols

Σ-Protocols are a well-studied class of Proofs of Knowledge. They are very important cryptographic primitives as they constitute the base for more complex constructions of zero knowledge proofs. The first Σ-Protocol was introduced by Schnorr in 1989 (Schnorr's Protocol [26]) to prove knowledge of a Discrete Logarithm.

Σ-Protocols are called 3-round protocols (a, c, z), because they consist of three interactive phases: commit phase, challenge phase and response phase. Commit (a) is generated from the Prover and sends it to the Verifier. Then the Verifier generates uniformly at random the challenge (c) and send it back to $\mathcal{P}$. Finally, Prover computes the response (z) with respect to the statement x and a, c and sends it to $\mathcal{V}$. At the end, the proof that $x \in L$ is checked by $\mathcal{V}$.

Additionally, they hold two security properties: Special soundness and Honest Verifier Zero Knowledge(HVZK) [48, 53].

Definition 3.5. *A 3-round Interactive Proof System $(\mathcal{P}, \mathcal{V})$ for a language L, with witness relation R_L, is **special sound** if there exists a PPT machine E such that given any two accepting transcripts with the same commit value $(a, c, z), (a, c', z')$ for proving $x \in L$ can extract a witness $w \in R_L(x)$:*

$$E(a, c, z, c', z') \to w \in R_L(x)$$

Special soundness implies proof of knowledge [54] with extraction error $1/|\mathcal{C}|$, where $\mathcal{C}$ is the challenge space.

Theorem 3.6. *Special Soundness ⇒ PoK*

Definition 3.7. *A protocol Π is a Σ-Protocol for a language L if it is 3-round protocol (a, c, z), where c is chosen uniformly at random from challenge space $\mathcal{C}$, and it has Completeness, Special Soundness and Honest Verifier Zero Knowledge.*

Zero Knowledge. What if verifier $\mathcal{V}$ is not honest? That is the case of 'original' zero knowledge definition. The answer is that Σ-Protocols cannot be proven zero knowledge and preserve soundness at the same time. Actually Goldreich and Krawczyk showed that a language L has 3-round black-box zero knowledge interactive proof if and only if $L \in BPP$ [2].

The problem is the challenge space $\mathcal{C}$. For large $|\mathcal{C}|$ we get negligible soundness error but no ZK, whereas with large we get the opposite.

Theorem 3.8. *If a Σ-Protocol Π has challenge space of size $|\mathcal{C}| = poly(\lambda)$ then Π is ZK with soundness $1/|\mathcal{C}|$ (non-negligible)*

A positive result is that we can construct an interactive proof that is black-box zero knowledge (in the standard model) by adding a round at the beginning of the protocol. Firstly $\mathcal{V}$ commits to a challenge c and afterwards the Σ-Protocol is executed normally. So we get a 4-round zero knowledge protocol (with negligible soundness error).

3.4 Non-interactive zero knowledge proofs

Until now we have been discussing interactive proof systems. That is, to create a proof both parties have to participate and exchange messages. However, in most actual protocol constructions the only contribution of verifier is to send a random string to the prover. So, intuitively, if both parties had agreed to a common randomness from scratch, randomness provided by the verifier would not be necessary anymore. This would enable a non-interactive proof formation. A proof could be made only by the prover and sent directly to the verifier for check.

The above concept is called non-interactive zero knowledge proof (NIZK for short) and it has been widely studied over the years, introduced by Blum,Feldman and Micali [55]. NIZK proofs have many cryptographic applications (e.g. Digital Signature Schemes, Group Signatures, Encryption Schemes). Additionally, they are extremely useful tools for multiparty cryptographic constructions like electronic voting schemes, e-Cash systems and Anonymous Questionnaire Systems, as we will see in next chapters.

As we said, to construct a NIZK proof we need a source of randomness, which is also called a shared object. Two are the most common shared objects for NIZK constructions: **Random Oracles (RO)** and **Common Reference Strings (CRS)**. As discussed in previous chapter a random oracle can be used to generate the necessary random challenge of the protocol. Common Reference String is a public random tape that is, also, used to obtain the challenge. The problem with the CRS is that we consider a trusted third party to sample a distribution to set it.

(NI)ZK in RO model. In this work we will see NIZK constructions in the RO model. This is why we emphasize in this model. NIZK in RO model was formally introduced by Bellare and Rogaway in 1993 [15]. The difference lies in zero knowledge property. To define it, we use, again, a simulator $\mathcal{S}$, which is now allowed to program the random oracle. So $\mathcal{S}$ chooses the answers of RO in order to simulate the proof. Subsequent Simulation models do not allow RO programmability (Non-programmable random oracles), but we will not discuss them. Below we give an informal definition of ZK proofs in Random oracle:

Definition 3.9. *A protocol is (black-box) zero knowledge in the random oracle model if:*

1) *Completeness is satisfied for all random oracles RO.*

2) *Soundness is satisfied with probability over all random RO.*

3) *There exist a simulator $\mathcal{S}$ s.t.*
$\{(RO, \langle \mathcal{P}^{RO}, \mathcal{V}^{*RO} \rangle)\}$ *and* $\{\mathcal{S}^{\mathcal{V}^*}\}$ *are indistinguishable.*

where $\mathcal{S}$ has the ability to watch $\mathcal{P}$'s and $\mathcal{V}^*$'s RO queries and choose the answers (program RO).

We note that completeness and soundness properties are no different than the ones in the standard model, except that parties have access to random oracle queries. That's why we omit a more formal definition.

The above definition applies both to interactive and non-interactive proofs. Similarly, we can define non-interactive proofs of knowledge (NIZK-PoK) in the random oracle model.

3.4.1 Fiat-Shamir transformation

A seminal work of Fiat and Shamir is the so called Fiat-Shamir transformation (also called Fiat-Shamir heuristic in the literature) [56] (1986). That is a technique that takes a Σ-protocol and converts it to a NIZK PoK. Initially it was introduced to create Digital Signatures, but it is, also, widely applied independently to remove interaction from proofs of knowledge.

So, assume we have a Σ-protocol with three rounds (a, c, z), where c is the challenge, uniformly chosen from the challenge space $\{0, 1\}^{\ell}$ by the (honest) verifier. Fiat-Shamir transformation is that we let c be chosen randomly from the random oracle: $c \leftarrow RO(a, x)$, where x is the statement to be proven. Afterwards z is computed normally, according to the protocol but with the above c, see Fig.3.1.

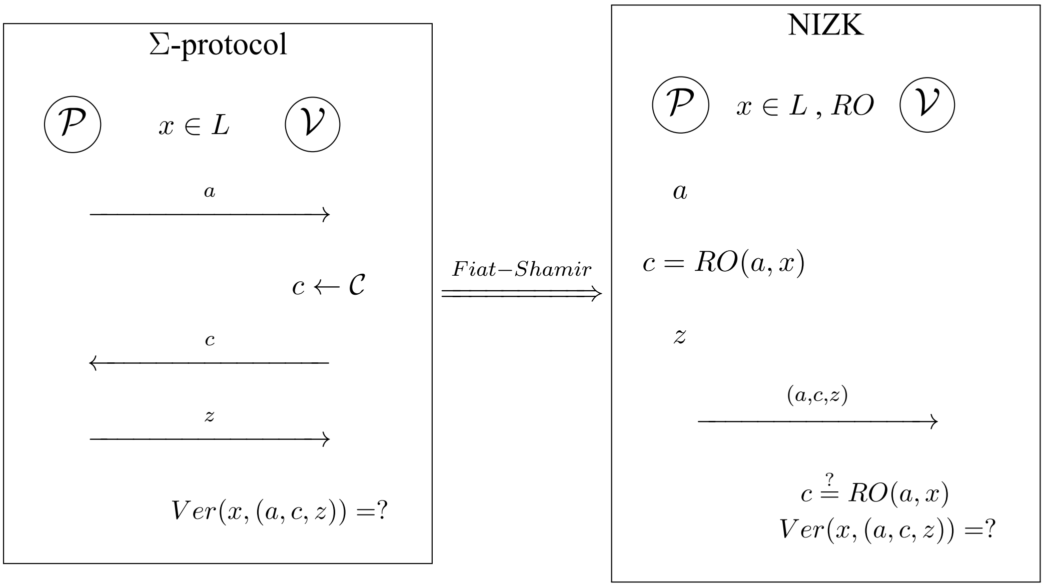


Figure 3.1: Fiat-Shamir transformation. From Σ-protocol to NIZKPoK

Theorem 3.10. *Assume* Π *is a* Σ*-protocol for language L with 3 rounds (a,c,z) with challenge space* $\{0,1\}^\ell$ *and* $\ell = \omega(log\lambda)$. $RO : \{0,1\}^* \to \{0,1\}^\ell$ *is a random oracle. Then* $(x, (a, c, z))$, *where* $c = RO(a, x)$ *is a Zero Knowledge Proof of Knowledge for* $x \in L$ *in the random oracle model.*

Proof. (Proof Sketch, we omit the details)
Completeness: it follows from the Σ-protocol completeness.
Soundness: we know that the soundness error is $1/|\{0,1\}^\ell| = \frac{1}{2^\ell}$, following the soundness error of the Σ-protocol, which is $negl(\lambda)$ as $\ell = \omega(log\lambda)$.
Zero-knowledge: Let *HVSim* be the honest-verifier simulator in the Σ-protocol.
The simulator after fiat-Shamir is:

$$\begin{aligned} Sim(x) \;\; &: \;\; (a, c, z) \leftarrow \textit{HVSim(x)} \\ & \;\;\;\;\; \text{program } RO(a, x) = c \end{aligned}$$

We note that c is sampled uniformly, assuming that RO gives uniformly random outputs.
PoK: E gets $(x, (a, c, z))$ from $\mathcal{P}^*$.
Then reprograms RO to give a different c': $c' \leftarrow RO(a, x)$, rewinds $\mathcal{P}^*$ back to where a was generated and hopes to get a new z' on (a, c').
Possessing (a, c, z) and (a, c', z') allows E to extract a witness w, due to special soundness. □

We observe that simulator does not use rewind. On the contrary, extractor uses rewind and programmability of the random oracle. These are traits that we are going to discuss on next chapters, that refer to concurrent executions of a protocol.

3.5 Zero Knowledge Proof Composition - Concurrency

The original setting in which cryptographic protocols were investigated consisted of a single execution of the protocol at a time. In real world it is common to have many executions of the same zero knowledge protocol with the same entities at the same time. This is a natural situation on the Internet, on e-Voting and Anonymous Survey Systems or even in blockchain applications. For example on e-Voting Systems we have an Election Authority who engages concurrently in zero knowledge protocols as a Verifier.

This fact has motivated many researchers to study zero knowledge protocols in a variety of composition operations. The main question was, at first, whether the zero knowledge condition is preserved under these operations. This was (and is until today) studied in depth from the late 90s. A few examples of work in literature are [49, 57, 58, 59, 60, 61, 62, 2, 63]. Another matter that arose was to preserve the extraction property for Proofs of Knowledge [20, 64, 65, 66].

By composition of protocols we mean that honest parties stick to the protocol and act independently in each execution. On the other hand, the adversary has the ability to choose her acts adaptively according to what happened in other executions. So now the adversary has an advantage compared to the single execution context.

In this chapter we give a first introduction to concurrency issues in zero knowledge, while it is certainly a subject with much more depth. For further study we suggest the book of Alon Rosen on concurrent zero knowledge [1] and the work of Rafael Pass on [64].

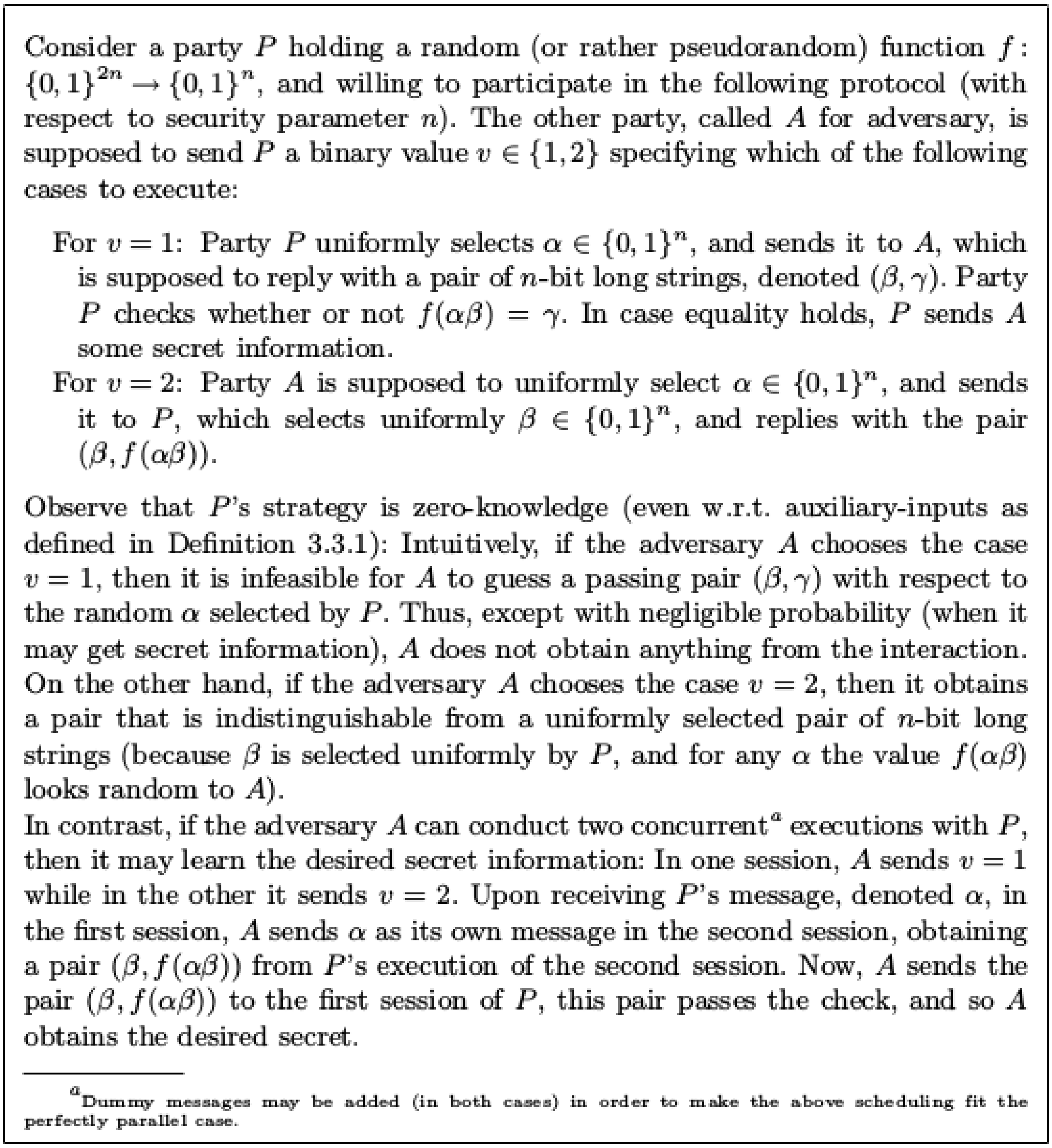
The main composition types involving only one protocol are 3: sequential, parallel and concurrent execution.

Sequential composition. The protocol is executed polynomially many times in sequence. That is, one execution of the protocol starts after the termination of the previous. Goldreich and Krawczyk proved in [2] that the original definition of zero knowledge that does not include auxiliary input is not sequential-zero knowledge. This is natural because Verifier gains information from each execution which is not taken into account. On the other hand, auxiliary input zero-knowledge proofs are proven to be closed under sequential composition [49].

Parallel composition. The protocol is executed polynomially many times in parallel. The i-th message, thus, of each instance is sent at (approximately) the same time. Of course, each protocol execution starts ad finishes at the same time. Parallel executions of the same protocol are, sometimes, used to reduce the soundness error, when the single protocol has high (but bounded away from 1) error, preserving the round-efficiency.

For years it was a common belief that zero knowledge is closed under parallel executions (Parallel-Composition Conjecture). But in [2] there is a counter-example that shows that in general, zero knowledge is not closed under parallel executions (see Fig.3.2). Of course, there are zero knowledge protocols for NP languages that are parallel-zero knowledge assuming standard cryptographic assumptions (e.g. [63]).

Concurrent composition. Concurrent zero knowledge is the more general and the most interesting composition of three. In this model we have asynchronous communication. So the messages at each protocol can be sent at any time. This model reflects more realistically what happens in



Consider a party P holding a random (or rather pseudorandom) function $f: \{0,1\}^{2n} \to \{0,1\}^n$, and willing to participate in the following protocol (with respect to security parameter n). The other party, called A for adversary, is supposed to send P a binary value $v \in \{1,2\}$ specifying which of the following cases to execute:

For $v = 1$: Party P uniformly selects $\alpha \in \{0,1\}^n$, and sends it to A, which is supposed to reply with a pair of n-bit long strings, denoted (β, γ). Party P checks whether or not $f(\alpha\beta) = \gamma$. In case equality holds, P sends A some secret information.

For $v = 2$: Party A is supposed to uniformly select $\alpha \in \{0,1\}^n$, and sends it to P, which selects uniformly $\beta \in \{0,1\}^n$, and replies with the pair $(\beta, f(\alpha\beta))$.

Observe that P's strategy is zero-knowledge (even w.r.t. auxiliary-inputs as defined in Definition 3.3.1): Intuitively, if the adversary A chooses the case $v = 1$, then it is infeasible for A to guess a passing pair (β, γ) with respect to the random α selected by P. Thus, except with negligible probability (when it may get secret information), A does not obtain anything from the interaction. On the other hand, if the adversary A chooses the case $v = 2$, then it obtains a pair that is indistinguishable from a uniformly selected pair of n-bit long strings (because β is selected uniformly by P, and for any α the value $f(\alpha\beta)$ looks random to A).

In contrast, if the adversary A can conduct two concurrent[a] executions with P, then it may learn the desired secret information: In one session, A sends $v = 1$ while in the other it sends $v = 2$. Upon receiving P's message, denoted α, in the first session, A sends α as its own message in the second session, obtaining a pair $(\beta, f(\alpha\beta))$ from P's execution of the second session. Now, A sends the pair $(\beta, f(\alpha\beta))$ to the first session of P, this pair passes the check, and so A obtains the desired secret.

[a] Dummy messages may be added (in both cases) in order to make the above scheduling fit the perfectly parallel case.

Figure 3.2: A counter-example (from [1] based on [2]) to the parallel repetition conjecture for zero-knowledge protocols.

real life scenarios, thus it is desirable in many applications, including Anonymous Survey Systems as we will see in next Chapters.

3.5.1 Concurrent zero knowledge

It is clear that sequential and parallel executions models are special cases of this model, thence concurrent zero knowledge is the most difficult to achieve. The first work that considered concurrency in the context of zero knowledge was published in 1998 by Dwork, Naor and Sahai [57]. In this paper it is marked why 'usual' zero knowledge proofs do not contain zero-knowledgeness under concurrent executions. The problem is that the adversary (controlling Verifiers) is able to schedule the answers in a way that the (PPT) Simulator cannot work anymore. So according to the

Simulation paradigm we discussed above absence of Simulator means gain of Knowledge for the adversary.

On Fig.3.3 we present the scheduling that defaces the Simulator (from [57]). Consider a 4-round zero knowledge protocol, with Simulator S, that is put under concurrent executions and an adversary $\mathcal{V}^*$ that controls all verifiers. Naturally $\mathcal{V}^*$ chooses the time to send her messages, so she is able to schedule them as shown in Fig.3.3.

Suppose that the Simulator does not need to rewind during the first two rounds but it needs rewinding after the third round. Simulating interaction with $\mathcal{V}_n$ is straightforward, exactly as in a single protocol. However, simulating interaction $\mathcal{V}_{n-1}$ requires rewinding back to round 1 of $\mathcal{V}_{n-1}$. So now S needs to simulate interaction with $\mathcal{V}_n$ again, because randomness of all the subsequent sessions is modified! Say that $R(n)$ is the number of rewinds in the case of n verifiers then:

$$\begin{aligned} &R(n) = R(n-1) + 1 + R(n-1) = 2R(n-1) + 1 \\ &\text{with } R(1) = 1 \end{aligned}$$

The above implies that $R(n) = \Omega(2^n)$ which is no longer efficient. Therefore, we cannot simulate the total interaction. That's why zero knowledge is not preserved (in general) under concurrent composition.

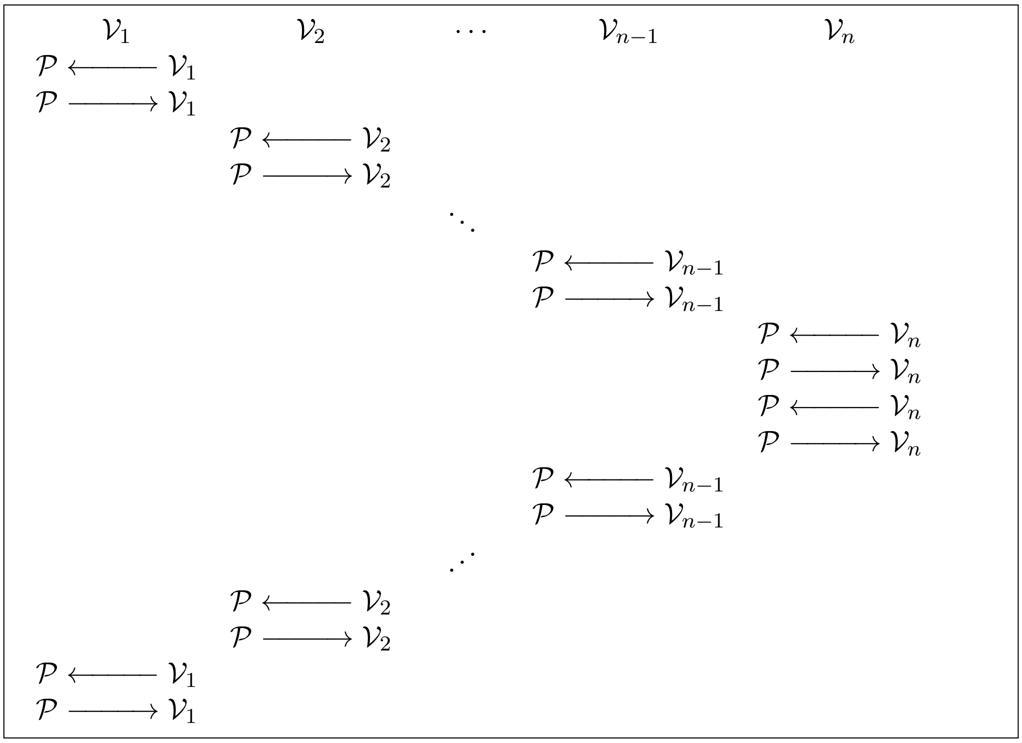

$\mathcal{V}_1$
$\mathcal{V}_2$
$\cdots$
$\mathcal{V}_{n-1}$
$\mathcal{V}_n$
$\mathcal{P} \longleftarrow \mathcal{V}_1$
$\mathcal{P} \longrightarrow \mathcal{V}_1$
$\mathcal{P} \longleftarrow \mathcal{V}_2$
$\mathcal{P} \longrightarrow \mathcal{V}_2$
$\ddots$
$\mathcal{P} \longleftarrow \mathcal{V}_{n-1}$
$\mathcal{P} \longrightarrow \mathcal{V}_{n-1}$
$\mathcal{P} \longleftarrow \mathcal{V}_n$
$\mathcal{P} \longrightarrow \mathcal{V}_n$
$\mathcal{P} \longleftarrow \mathcal{V}_n$
$\mathcal{P} \longrightarrow \mathcal{V}_n$
$\mathcal{P} \longleftarrow \mathcal{V}_{n-1}$
$\mathcal{P} \longrightarrow \mathcal{V}_{n-1}$
$\iddots$
$\mathcal{P} \longleftarrow \mathcal{V}_2$
$\mathcal{P} \longrightarrow \mathcal{V}_2$
$\mathcal{P} \longleftarrow \mathcal{V}_1$
$\mathcal{P} \longrightarrow \mathcal{V}_1$

Figure 3.3: Concurrent schedule by an Adversary controlling all $\mathcal{V}_i$

The approach by Dwor, Naor and Sahai [57] was to add timing constraints. With these constraints they achieved constant round concurrent zero knowledge, but this was not a pure asynchronous model. Many constructions have been achieved thereafter considering variants of timing constraints (e.g. [54, 67, 63]. Yet, the most interesting setting is the unconstrained.

Fig.3.3 scheduling showed that the setback to simulate concurrent executions is rewinding. It is trivial to construct a Simulator for concurrent composition if the original simulator (of the single execution) does not use rewinds. These are called straight-line (or online) simulators. So a straight-line simulatable proof is concurrent zero knowledge. This, as we will discuss later, makes it easy to construct concurrent non-interactive zero knowledge proofs.

Intuitively, rewinding is the only advantage simulator has over the honest prover (in the standard model without considering CRS or random oracle existence). This points out that rewinding is somehow inherent to black-box simulation. So impossibility results have come. Kilian, Petrank and Rackoff showed impossibility for for 4-round concurrent zero knowledge proofs with black-box simulation [58](for non-trivial languages outside $\mathcal{BPP}$). Then Rosen showed impossibility for 7-round protocols [60]. The final bound came from Canetti, Kilian, Petrank and Rosen [61] and is almost logarithmic, $\Omega(\frac{logn}{loglogn})$ rounds. Their result is below:

Theorem 3.11. *Let $r : \mathbb{N} \to \mathbb{N}$ be a function so that $r(n) = o(\frac{logn}{loglogn})$. Suppose that $\langle \mathcal{P}, \mathcal{V} \rangle$ is an $r(\cdot)$-round proof system for a language L (i.e. on input x, the number of messages exchanged is at most $r(|x|)$), and that concurrent executions of $\mathcal{P}$ can be simulated in polynomial-time using black-box simulation. Then $L \in \mathcal{BPP}$. The theorem holds even if the proof system is only computationally-sound (with negligible soundness error) and the simulation only computationally-indistinguishable (from the actual executions).*

The last interesting result that we are going to mention is that the above bound can be matched. First concurrent zero knowledge argument construction for every was by Kilian, Petrank and Rackoff [59] and required $\mathbb{O}(n^{\epsilon})$ rounds for every NP language. Then in 2002 Prabhakaran, Rosen and Sahai showed the existence of $\tilde{\mathbb{O}}(logn)$ round concurrent zero knowledge argements for all NP languages under perfectly-hiding commitment schemes existence assumption [62].

Theorem 3.12. *Assuming the existence of perfectly-hiding commitment schemes, there exists an $\tilde{\mathbb{O}}(logn)$-round black-box concurrent zero-knowledge proof system for every language $L \in NP$.*

We conclude that the round-complexity of black-box concurrent zero-knowledge is $\tilde{\Theta}(logn)$ rounds.

3.6 Concurrent Knowledge Extraction

As stated above, in Proof of Knowledge protocols there is a witness extractor. However, in concurrent executions of a protocol with the same (possibly malicious) Prover P^*, Extractor must extract (polynomially) many witnesses simultaneously. Intuitively, similar problems to concurrent zero knowledge can occur.

Indeed, similar scheduling can be made by the adversarial prover that can outplay an Extractor that uses rewinding. This, again, can lead to exponential number of rewinds (in the number of executions). A notable fact that differs from zero knowledge is that even in NIZK Proofs of Knowledge some Extractors still work by rewinding. For example Fiat-Shamir Proofs of Knowledge use rewinds to extract the witness. In this context, the rewinding problem was first mentioned explicitly by Shoup and Gennaro [3], see Fig. 3.4

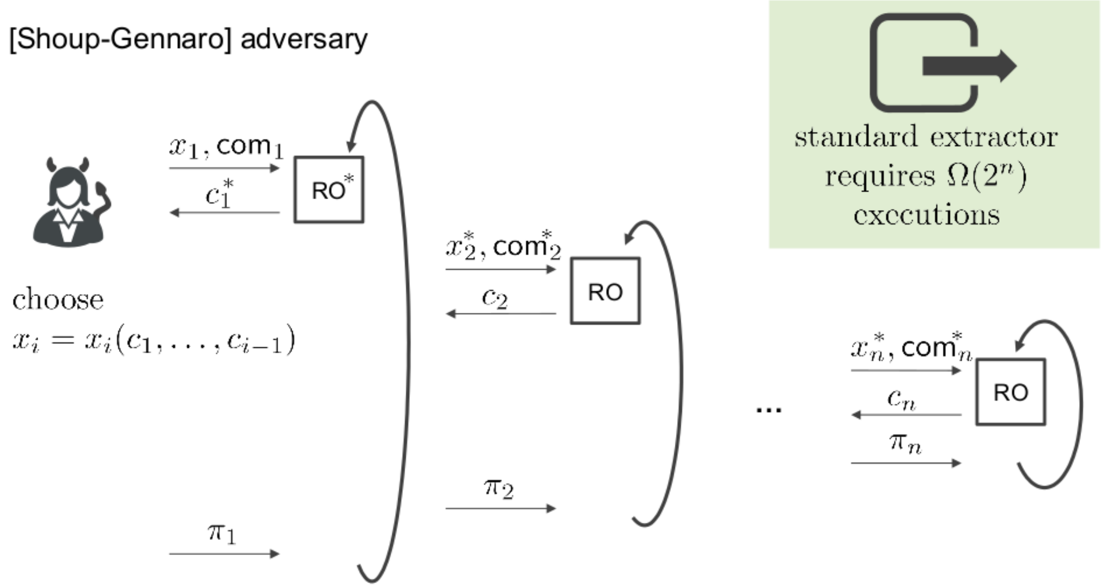


Figure 3.4: Rewinding extractors fail in concurrent setting, as discussed in [3]. Figure from [4]

The solution to this problem was to construct Extractors that do not need rewinding, the Online (or Straight-line) Extractors. Fischlin, presented in 2005 an alternative transformation to Fiat-Shamir that takes a Σ-protocol and converts it to online extractable NIZK PoK in the Random Oracle model [68], which is now called Fischlin transformation. Another interesting construction of online extractors, prior to Fischlin transformation, that we are going to see in later Chapters, was introduced by Pass [20].

In both constructions the random oracle cannot be programmed by the Extractor, i.e. Extractor cannot choose the answers to the oracle queries that both parties (prover and extractor) make. So, malicious prover $\mathcal{P}^*$ makes oracle queries and then Extractor $\mathcal{E}$ makes oracle queries. The extractor cannot rewind and cannot program the random oracle. The only advantage that is given to $\mathcal{E}$ is that it can inspect the random oracle queries that $\mathcal{P}^*$ did. Restricting the extractor is the success key to extract witnesses (in polynomial time) under concurrent executions.

Definition 3.13. *A zero knowledge proof $(\mathcal{P}, \mathcal{V})$ is online extractable in the random oracle model if there is a PPT extractor E such that for any (possibly malicious) prover $\mathcal{P}^*$:*

$$Pr[x, z \leftarrow \mathcal{P}^*; \pi \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle; w \leftarrow E(view, Q) : Ver(\pi) = 1 \wedge R_L(x, w) \neq 1] = negl(|x|)$$

where Q is the list of RO queries made by $\mathcal{P}^*$ and $view$ is the view of $\mathcal{V}$.

The above is the formal definition of online extraction property in the random oracle model. Basically it is same as extraction definition but $\mathcal{E}$ has not black-box access to $\mathcal{P}^*$, which implies no rewinding, and cannot program the RO. Furthermore, online extraction can , also, be defined in the common reference string model but we do not deal with that.

Finally, we mention that recently (2017) Bernhard et al. [66] showed that every non-interactive zero knowledge proof of knowledge in the random oracle model, with an adaptive adversary should have online extractor. That is, even the most clever rewinding extractor fails in the concurrent (and adaptive) setting.

3.7 Man-in-the-middle-attacks (non-malleability/Simulation-extractability/Simulation-soundness)

The classic definition of zero knowledge requires that an adversary does not learn anything rather the validity of the statement: $x \in L$. But what if an adversary first receives a proof of $x \in L$ and then tries to prove $\tilde{x} \in L$, without having a witness? This scenario is called *man-in-the-middle* attack (see Fig. 3.5). In this attack an adversary stands between the prover and the verifier, receives the messages on their behalf and chooses adaptively the messages sent and the timing. So she controls the interaction completely. An attack of this type is the 'Mafia scam' attack [69].

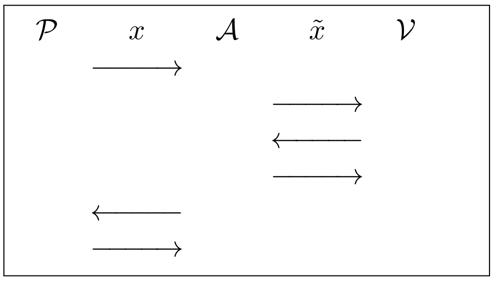


Figure 3.5: Man-in-the-middle Adversary in interactive ZK proofs with an example scheduling.

Three security notion that are related to the *man-in-the-middle* attack are non-malleability [70, 71], simulation-extractability [72] and simulation-soundness [71].

Non-malleability. Intuitively, non-malleability in the context of interactive proof protocols is not different from non-malleability in CCA-secure encryption context. We require that a proof for x cannot be transformed to constitute a new proof for $\tilde{x}$ (except when $x = \tilde{x}$). Non-malleable zero knowledge protocols were defined in the seminal work of Dolev, Dwork and Naor in 1991 [70]. The definition we give follows the formalization of [72].

So there are two executions, the real and the ideal. The real is the man-in-the-middle execution, which lets the adversary $\mathcal{A}$ interact with the prover $\mathcal{P}$ and with the verifier $\mathcal{V}$ at the same time. The ideal is a stand-alone execution with an adversary $\mathcal{S}$ as a prover, who can only interact with the verifier $\mathcal{V}$, see Fig. 3.6.

We require that the output of the Man-in-the-middle execution and the Stand-alone execution are negligibly close (the outputs are random variables). The idea behind this is that being a man-in-the-middle does not make any (noticeable) difference than just being a prover.

Let $\mathbf{mim}^{\mathcal{A}}_{\mathcal{V}}(x, w, z)$ be the random variable describing the output of $\mathcal{V}$ in the above man-in-the-middle experiment. In case of $x = \tilde{x}$, $\mathbf{mim}^{\mathcal{A}}_{\mathcal{V}}(x, w, z)$ is defined $\perp$.
Let $\mathbf{sta}^{\mathcal{S}}_{\mathcal{V}}(x, \tilde{x}, z)$ be the random variable describing the output of $\mathcal{V}$ in the above stand-alone experiment

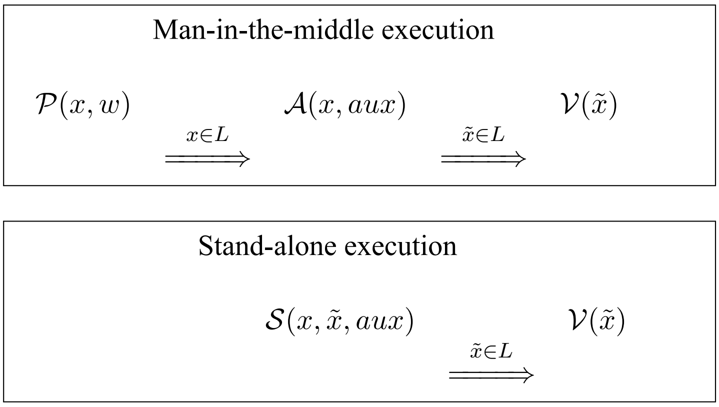


Figure 3.6: Man in the middle execution and stand-alone execution.

Definition 3.14. *An interactive proof $\langle \mathcal{P}, \mathcal{V} \rangle$ for a language L is said to be non-malleable if for every PPT man-in-the-middle adversary $\mathcal{A}$, there exists a PPT stand-alone prover $\mathcal{S}$ such that for every $(x, w) \in L \times R_L(x)$, every $\tilde{x} \in \{0,1\}^{|x|}$ so that $x \neq \tilde{x}$ and every $z \in \{0,1\}^*$:*

$$Pr[\textbf{\textit{mim}}_{\mathcal{V}}^{\mathcal{A}}(x, w, z) = 1] < Pr[\textbf{\textit{sta}}_{\mathcal{V}}^{\mathcal{S}}(x, \tilde{x}, z) = 1] + negl(|x|)$$

The above does not include zero knowledges as a property. Of course, if the non-malleable interactive proof is additionally zero knowledge then it is a non-malleable zero knowledge proof.

Definition 3.15. *A family $\{\mathcal{P}, \mathcal{V}\}$ of interactive proofs is said to be non-malleable zero knowledge if it is both non-malleable and zero knowledge.*

Simulation-Soundness. Sahai took the man-in-the-middle scenario one step further and considered a new notion called simulation-soundness [71]. In this we consider that an adversary $\mathcal{A}$ can receive many (polynomially bounded) simulated proofs for statements of her choice (left interaction) and afterwards try to deceive an honest verifier $\mathcal{V}$ with a false proof (right interaction). A key detail is that some of the simulated proofs may be for false statements $x \notin L$. So, for simulation-soundness, we require that any PPT adversary, even after seeing polynomially many simulated proofs (possibly on false statements), cannot prove something invalid to the verifier (except with negligible probability).

Following [73] we give below the formal definition of simulation-soundness.

Definition 3.16. *Let $\Pi = (\mathcal{P}, \mathcal{V}, S = (S_1, S_2)$ be an unbounded NIZK proof system for a language L. We say that Π is simulation-sound if for all non-uniform probabilistic polynomial-time adversaries $\mathcal{A}$, we have that:*

$$\textbf{SIMSOUND} - \textbf{Expt}_{\mathcal{A},\Pi}(n) = negl(n)$$

where $\textbf{SIMSOUND} - \textbf{Expt}_{\mathcal{A},\Pi}(n)$ is the following experiment:

SIMSOUND − Expt$_{\mathcal{A},\Pi}(n)$

- $(\Sigma, \tau) \leftarrow S_1(1^k)$
- $(x, p) \leftarrow A^{S_2(\cdot,\Sigma,\tau)}(\Sigma)$

Let Q be list of proofs given by S_2 above

$$output = \begin{cases} 1 & \textit{if } p \notin Q \;\; \textit{and } x \notin L \textit{ and } \mathcal{V}(x, p, \Sigma) = 1 \\ 0 & otherwise \end{cases}$$

Simulation-Extractability. Another way to deal with man-in-the-middle attack on an interactive proof system (with zero knowledge or without) is a security variant called simulation-extractability, introduced by Pass and Rosen [72]. Loosely speaking, the demand of this notion is that both left and right interactions of $\mathcal{A}$ can be simulated by a PPT machine, while outputting a witness for the statement $\tilde{x}$ proved by $\mathcal{A}$ in the right interaction. Informally, we can imagine that simulation-extractability requires one simulation(for left interaction), one simulation (for right interaction) and one extraction (of witness for right interaction).

The definition of simulation-extractability as introduced in [72] is given below. Though, we have excluded the tag for simplicity, so the definition is not about tag-based proofs.

Definition 3.17. *A family $\{\mathcal{P}, \mathcal{V}\}$ of interactive proofs is said to be simulation-extractable if for any man-in-the-middle adversary $\mathcal{A}$, there exists a PPT machine (SIM, EXT) such that:*

1) The ensembles $\{SIM(x, aux)\}_{x,aux}$ and $\{view_{\mathcal{A}}(x, aux)\}_{x,aux}$ are statistically close.

2) Let $\tilde{x}$ be the right hand side statement appearing in $SIM(x, aux)$. If the right hand side interaction is accepting the output of $EXT(x, z)$ consists of a witness w so that $R_L(\tilde{x}, w) = 1$.

Seemingly, simulation-extractability is the stronger notion of three presented above. In fact, Pass and Rosen proved formally that: $SIM - EXT\ ZK \Rightarrow NMZK\ PoK$ [72]. Furthermore, it is not difficult to see that, given the above definition of simulation-soundness, simulation-extractability is equivalent to simulation-sound proofs of knowledge. To conclude, as we will discuss later, in real life protocols, where the adversary is given many capabilities, such as Anonymous Survey Systems, simulation-extractability is a highly desirable property.

Chapter 4

Anonymous Survey Collection

A general solution to the anonymous survey problem described in the introduction (see section 1.2) is introduced by Hohenberger, Myers, Pass and shelat in [6]. In this work they define the notion of Ad-hoc Survey scheme, which is a new cryptographic protocol. The formal definition comes with the related security requirements.

Intuitively, an Ad-hoc Survey scheme is a protocol allowing anyone to select an Ad-hoc group of individuals to collect feedback from. So an initiator creates a survey and selects a group of individuals to answer it. We say ad-hoc because anyone can create a list of individuals non-interactively (i.e. without asking the individuals) knowing only their identities (e.g. mail address or name). Additionally, this list can grow dynamically, meaning that extra identities can be added at any time. We note, though, that the creation is ad-hoc given that a user has registered in the system, but only once in her life, as we will see later.

Of course, what we want to be achieved cryptographically is Authenticity and Anonymity. Authenticity means that only users in the ad-hoc group can participate in survey submission and that they can only submit their answers once. Anonymity means that nobody can relate a survey answer to an identity.

4.1 Ad-hoc Surveys

Ad-hoc Survey Scheme is a protocol involving three types of entities:

- A unique registration Authority(RA).
- Survey Authorities(SA).
- Users characterized by an id each.

Formally an ad-hoc Survey Scheme Γ is a tuple of seven PPT algorithms and PPT interactive protocols:

$$(GenRA, RegUser^{RA}, RegUser^{\mathcal{U}}, GenSurvey, Authorized, SubmitSurvey, Check)$$

- $GenRA(1^n)$: outputs a pair of keys vk_{RA}, sk_{RA}
  vk_{RA} is made public, sk_{RA} is private key of RA.
- $RegUser^{RA}(sk_{RA}, vk_{RA}, id)$: interactive PPT which outputs *accept* or *fail*.
  It is executed by the RA and interacts with an id, to register it. If the interactive protocol succeeded it outputs *accept* otherwise *fail*.

- $RegUser^{\mathcal{U}}(1^n, vk_{RA}, id)$: interactive PPT which outputs $cred_{id}$ or *fail*.
  It is executed by an id and interacts with the RA. In case id has not registered before it outputs an unlinkable master credential $cred_{id}$, which is kept private by id. Otherwise outputs *fail*.

- $GenSurvey(1^n, vid, L)$: outputs vk_{vid}.
  It is executed by an SA. vid is a unique public identifier of the survey chosen by SA. L is the (initial) list of identities participating in the ad-hoc group of the Survey.
  vk_{vid} is the survey public-key.

- $Authorized(vid, vk_{vid}, id)$ outputs *accept* or *fail*.
  It can be executed by anyone to check if id is authorized to participate in the survey vid.

- $SubmitSurvey(1^n, vid, vk_{vid}, m, cred_{id})$: outputs $Sub = (tok, m, tokauth)$.
  Executed by the user id. For a specific survey vid (with parameters vk_{vid}), with the unique master credential $cred_{id}$ and an answer m to be submitted unique one-time token tok and $tokauth$ are created.
  One-time token tok carries no link to the id and is unique with respect vid.
  $tokauth$, as will see later, is an authenticator for correctness of submission.

- $Check(vk_{RA}, vid, vk_{vid}, Sub)$: outputs *accept* or *fail*.
  Checks whether or not the submission Sub is correct. It can be executed by anyone.

Security requirements. The scheme described above need to satisfy three properties: Correctness, Anonymity (or Unlinkability) and Authenticity.
For Anonymity we assume that:

1. RA, many SAs and many corrupted users collude and are controlled by the adversary.
2. Adversary has identified the user in past surveys (of her choice).
3. Adversary will identify the user in future surveys (of her choice).

and yet adversary cannot identify the user in the current survey.

We observe that the definition mirrors the one of CCA-secure encryption and is formally defined with an indistinguishability game between a challenger and an adversary.

For Authenticity we assume that:

1. Many SAs and many corrupted users collude and are controlled by the adversary.
2. Adversary can ask from any (honest) user id survey submission with content m of her choice and for any survey. If the user is not registered then she is forced to register with the RA.

Adversary chooses a survey vid and her goal is to submit more answers than the number of the participant corrupted users. There are two ways to succeed in her attack either a corrupted user submits two different answers successfully or she passes a survey submission with an id that is not authenticated.

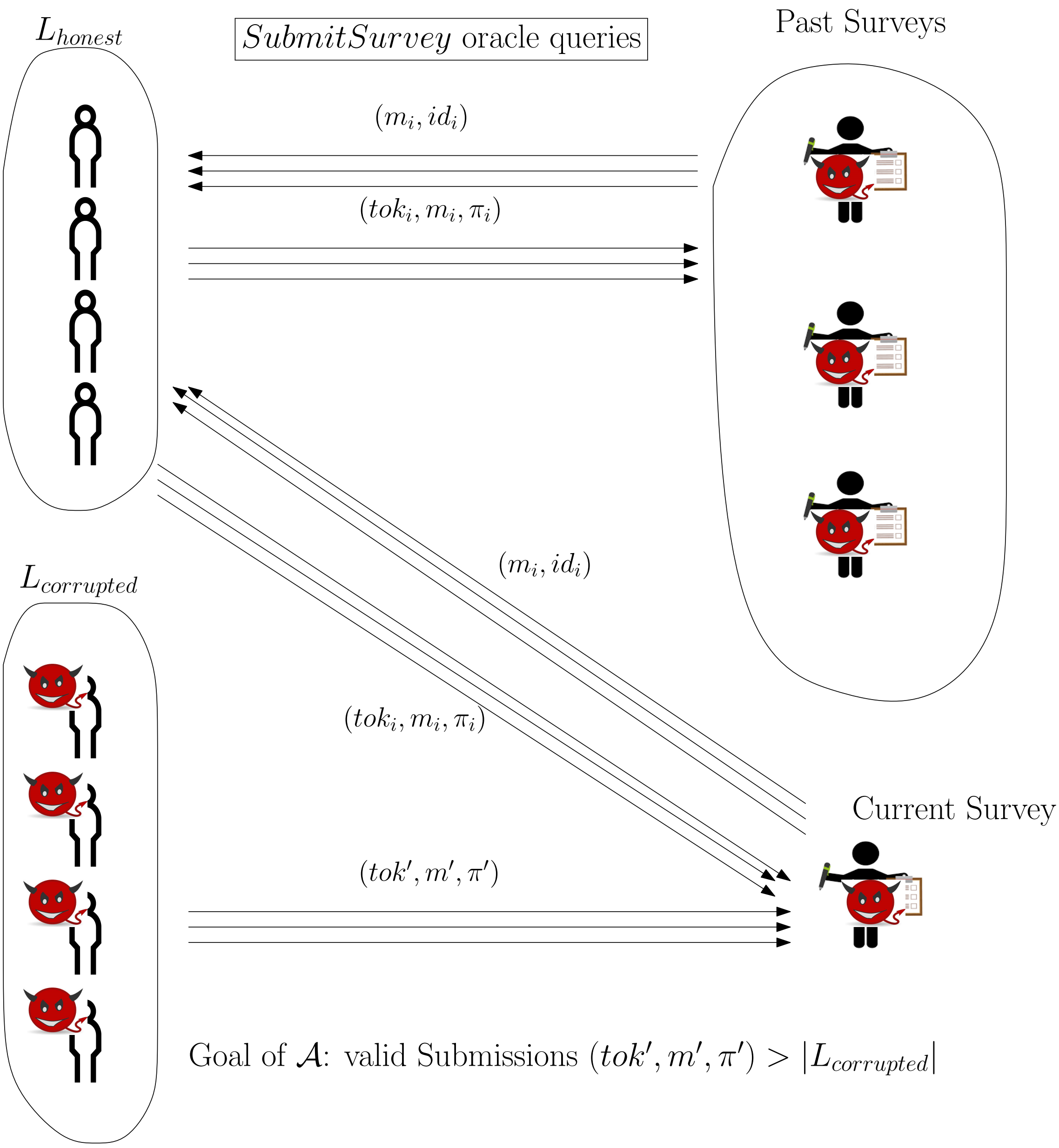


Figure 4.1: Authenticity Experiment for ad-hoc survey scheme.

So the security requirement is that any adversary fails in the above experiment (described in Fig. 4.1).

Finally, an Ad-hoc Survey scheme is correct if considering all parties honest (follow the protocol) these happen with overwhelming probability:

1. Every user register successfully, interacting with the RA i.e.
   $out[RegUser^{RA}(sk_{RA}, vk_{RA}, id) \leftrightarrow RegUser^{\mathcal{U}}(1^n, vk_{RA}, id)] \neq fail$

2. For every survey initiation all users intended to be in the list are successfully included:
   $Authorized(vid, vk_{vid}, id) \neq fail$ for every id $\in L$

3. Every user id, who submits answer is successfully verified:
   $Check(vk_{RA}, vid, vk_{vid}, Sub) \neq fail$

4. All tokens are unique, i.e. the below is not possible:
   there exist $id, id' \in L$ s.t. $Sub_{id} = (tok, m, tokauth)$ and $Sub_{id'} = (tok, m', auth')$ have the same tok.

In conclusion, an ad-hoc survey scheme is a general cryptographic primitive introduced in [6]. In the next chapter we are going to talk about a concrete instantiation, called Anonize, based on known cryptographic tools that we saw in introductory chapters and general cryptographic assumptions.

4.2 Anonize - a concrete ad-hoc scheme

We can, roughly, categorize the ad-hoc survey scheme algorithms into three phases: User registration, Survey Creation and Survey Submission, that correspond to what happens in reality. In the original paper, Hohenberger et al. constructed an ad-hoc scheme using the below cryptographic primitives:

- A Commitment Scheme $(Gen_{com}, Com, Open)$
- A Digital Signature Scheme $(Gen, Sign, Ver)$
- A Partially Blind Signature Scheme $(Gen', Blind', Sign', Unblind', Ver')$
- A family of Pseudorandom functions $\{f_s\}$
- An online simulation-extractable NIZK in the RO model (P, V, RO).

The Partially blind signature is constructed as shown in section 2.6.3, using a commitment scheme and a digital signature scheme. So, the commitment scheme is only used in the protocol to construct the partially blind signature scheme.

Registration Phase

1. $GenRA(1^n)$: RA uses $Gen'(1^n)$ to create a pair of partially blind signature keys: (vk_{RA}, sk_{RA})

2,3. $\langle RegUser^{RA}(sk_{RA}, vk_{RA}, id), RegUser^{\mathcal{U}}(1^n, vk_{RA}, id)\rangle$:
id generates a random seed $s \leftarrow \{0,1\}^n$, blinds it (using Com of commitment scheme) and sends it to the RA.
Afterwards the RA signs it along with the id and send the signature to the user id.
Then id unblinds it and gets her credential $(s, Sign(id, s))$, which is going to use to submit a survey answer.

Survey Creation Phase
Anyone who wishes to initiate a Survey becomes Survey authority.
SA chooses the participants (based on ids) and put them in a list L. It, also, generates a Survey identifier vid.

4. $GenSurvey(1^n, vid, L)$: SA generates a pair of signature keys (vk_{SA}, sk_{SA}) and with the latter puts a signature for each $id \in L$ on (vid, id).
The (public) output is the list of signatures $\tilde{L} = \{(id, Sign_{sk_{SA}}(vid, id))\}_{id \in L} = \{\left(id, \sigma_{id}^{vid}\right)\}_{id \in L}$ and the verification key vk_{SA}.
The output is summarized in $vk_{vid} = (vk_{SA}, \tilde{L})$

Survey Submission Phase

5. $Authorized(vid, vk_{vid}, id)$: user id checks if she is in the signed list $\tilde{L}$ and if the signature is valid i.e. $Ver_{vk_{SA}}((vid, id), \sigma_{id}^{vid}) = 1$.
6. $SubmitSurvey(1^n, vid, vk_{vid}, m, cred_{id})$: if authorized outputted 1 then id does the below:
   - Computes the unique token $tok = f_{s_{id}}(vid)$, where $f_{s_{id}}$ is a pseudorandom function with random seed the s_{id} signed (in blind) from RA in the registration phase.
   - Forms a tag-based oSE NIZK π with tag $tok||vid||m$ proving that the token was computed fairly $tok = f_{s_{id}}(vid)$ and she has valid signatures on (id, s_{id}) and (vid, id) from RA and SA respectively.
   - Finally, outputs $Sub = (tok, m, \pi)$ and sends it to the SA.
7. $Check(vk_{RA}, vid, vk_{vid}, Sub)$: SA receives submission Sub and accepts if the proof π is verified, with respect to the tag $tok||vid||m$.

After Completion
Some auditing procedures can take place anytime and by anyone. Whoever wants can execute $Authorized(vid, vk_{vid}, id)$ for id of her choice to see if see was in the list of participants. Furthermore, depending on the policy of the survey, the results can be made public. If so, anyone can execute $Check(vk_{RA}, vid, vk_{vid}, Sub)$ for every answer submitted to check the validity of the survey results. Finally, anyone can perform checks for possible double submissions by inspecting tokens tok and see if the same token appears two (or more) times.

For real instantiation of Ad-hoc Surveys one needs to find specific primitives for pseudorandom function, Digital Signature and Commitment Scheme so that they are ”stitched together” to construct the desired NIZK proof.

One possible instantiation of an Ad-hoc Survey scheme is Anonize. We note here, though, that ad-hoc schemes are general primitives which have endless possible instantiations. Anonize uses the below concrete cryptographic protocols:

The commitment scheme The commitment scheme is Pedersen, which was presented in section 2.5.1. Pedesren is information-theoretically hiding and is reduced to DLP problem. We repeat that the commitment algorithm is:

$$Com(m; r) = g^m h^r$$

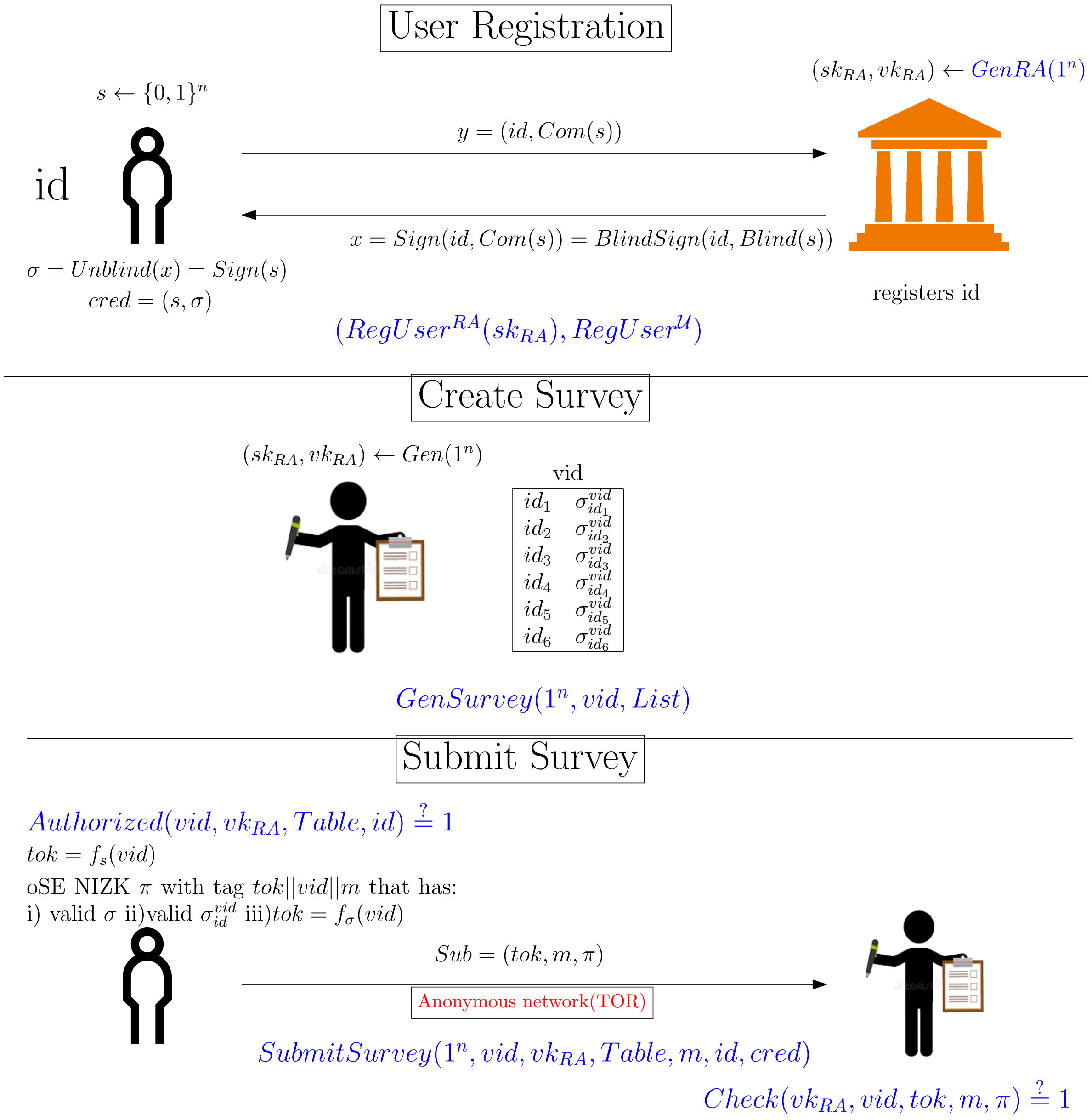

User Registration
$(sk_{RA}, vk_{RA}) \leftarrow GenRA(1^n)$
$s \leftarrow \{0,1\}^n$
$y = (id, Com(s))$
id
$x = Sign(id, Com(s)) = BlindSign(id, Blind(s))$
$\sigma = Unblind(x) = Sign(s)$
$cred = (s, \sigma)$
registers id
$(RegUser^{RA}(sk_{RA}), RegUser^{\mathcal{U}})$
Create Survey
$(sk_{RA}, vk_{RA}) \leftarrow Gen(1^n)$
vid
id_1 $\sigma^{vid}_{id_1}$
id_2 $\sigma^{vid}_{id_2}$
id_3 $\sigma^{vid}_{id_3}$
id_4 $\sigma^{vid}_{id_4}$
id_5 $\sigma^{vid}_{id_5}$
id_6 $\sigma^{vid}_{id_6}$
$GenSurvey(1^n, vid, List)$
Submit Survey
$Authorized(vid, vk_{RA}, Table, id) \overset{?}{=} 1$
$tok = f_s(vid)$
oSE NIZK π with tag $tok||vid||m$ that has:
i) valid σ ii)valid σ^{vid}_{id} iii)$tok = f_\sigma(vid)$
$Sub = (tok, m, \pi)$
Anonymous network(TOR)
$SubmitSurvey(1^n, vid, vk_{RA}, Table, m, id, cred)$
$Check(vk_{RA}, vid, tok, m, \pi) \overset{?}{=} 1$

Figure 4.2: Anonize System

The Pseudorandom function Anonize utilizes Dodis-Yampolskiy Pseudorandom function [74]. It is described in Figure 4.3.

Dodis-Yamploskiy is a pseudorandom function family assuming hardness of Decisional n-Bilinear Diffie-Hellman Inversion for asymmetric groups, defined in section 2.9.1.

The Digital Signature Scheme The digital signature scheme that is used in Anonize is the Boneh-Boyen digital signature. That is a digital signature scheme that was implicitly defined in [43]. Although, this work was focused on an Identity Based Encryption Scheme in the standard model, the Signature scheme came as a result.

The initial scheme has selective unforgeability under the Decisional Bilinear Diffie Hellman assumption. To achieve existential unforgeability a security reduction to DBDH was constructed but gave the adversary subexponential power. So we conclude that Boneh-Boyen Signature Scheme

Dodis-Yampolskiy

Let a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ on groups of order q and $e(g_1, g_2)$ the generator of $\mathbb{G}_T$.
Let $s \in \mathbb{Z}_q$ be the random seed.

$$F_s(m) = e(g_1, g_2)^{\frac{1}{m+s}} \qquad \forall m \text{ s.t. } (m+s) \neq 0$$

Figure 4.3: Dodis-Yamplolskiy Pseudorandom function

is existentially secure given that DBDH problem is still hard for adversaries with subexponential power, which we call Subexponential DBDH assumption.

Boneh-Boyen

Let a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ on groups of order q and g_1, g_2 generators of $\mathbb{G}_1, \mathbb{G}_2$ resp.

- **Gen**(1^n): $a \leftarrow \mathbb{Z}_q \qquad u, v, h \leftarrow \mathbb{G}_1 \qquad \mathcal{U} = e(g_1, g_2)^a$

  $$sk = a \quad vk = (u, v, h, \mathcal{U})$$

- **Sign**(sk, m_0, m_1): $w \leftarrow \mathbb{Z}_q$

  $$\sigma_1 = g_1^a(u^{m_0}v^{m_1}h)^w, \;\; \sigma_2 = g_2^w, \;\; \sigma_3 = g_1^w$$

  output $\sigma = (\sigma_1, \sigma_2, \sigma_3)$

- **Ver**$(vk, m_0, m_1, \sigma_1, \sigma_2)$:

  $$e(\sigma_1, g_2) \overset{?}{=} \mathcal{U} \cdot e(u^{m_0}v^{m_1}h, \sigma_2) \;\; \text{and} \;\; e(\sigma_3, g_2) \overset{?}{=} e(g_1, \sigma_2)$$

Figure 4.4: Boneh-Boyen Signature Scheme

The Partially Blind Signature Scheme As stated before we use the construction of section 2.6.3 that combines a commitment scheme and a signature scheme to construct the partially blind signature scheme that satisfies weak blinding.

Partially Blind Signature instantiation

Let a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ on groups of order q and g_1, g_2 the generator of $\mathbb{G}_1, \mathbb{G}_2$ resp.

- **Gen′**(1^n): $a \leftarrow \mathbb{Z}_q \qquad u, v, h \leftarrow \mathbb{G}_1 \qquad \mathcal{U} = e(g_1, g_2)^a$

  $$sk = a \qquad vk = (u, v, h, \mathcal{U})$$

- **Blind′**$_r(vk, s)$: $d \leftarrow \mathbb{Z}_q$

  $$c = Com(d; s) = g_1^d v^s$$
  $$\pi = PoK\{(s, d) : \gamma = g_1^d v^s\} \text{ (oSE NIZK)}$$

  output (c, π)

- **Sign′**$(sk, (m, c))$: $w \leftarrow \mathbb{Z}_q$

  $$\sigma_1 = g_1^a (u^m \gamma h)^w, \quad \sigma_2 = g_2^w, \quad \sigma_3 = g_1^w$$

  output $\sigma = (\sigma_1, \sigma_2, \sigma_3)$

- **Unblind′**$_r(vk, (\sigma_1, \sigma_2, \sigma_3))$: $(\sigma_1 / \sigma_3^d, \sigma_2)$

- **Ver′**$_{vk}((m, s), (\sigma_1, \sigma_2))$: $e(\sigma_1, g_2) \overset{?}{=} \mathcal{U} \cdot e(u^m v^s h, \sigma_2)$

Figure 4.5: Partially blind signature scheme from Dodis-Yampolskiy pseudorandom function and Boneh-Boyen Signature Scheme.

The Online Simulation-Extractable Non-Interactive Zero Knowledge Proof (oSE NIZK) We present, now, a construction of oSE NIZK proofs which is used in Anonize. The construction is introduced in [20]. It is presented in two steps: first a transformation is used to convert a Σ-protocol to an online-extractable (see definition 3.13 in section 3.6) Σ-protocol, which is called Ω-protocol, and afterwards Fiat-Shamir transformation is used to get the final protocol.

Definition 4.1. *A protocol Π is an Ω-Protocol for a language L with witness-relation $R_L(\cdot)$ in the random oracle model if it is a Σ-protocol and additionally it is online extractable.*

Step 1: We transform a given Σ-protocol to an Ω-protocol.

The transformation uses a special commitment scheme called online Extractable commitment scheme, introduced by Pass in [20]. That is, similar to oE ZK proofs, commitments where the committed value can be extracted from a PPT extractor E_{com}, which only inspects the random oracle queries the sender did and does not use rewind. It is easy to prove that the Random oracle based commitments scheme presented in ch. 2.5.2 is online extractable. Thus, it is the one used.

Definition 4.2 (online extractable commitments)**.** *A commitment scheme is online extractable in RO model if there exists a PPT extractor E_{com} such that for each commitment c if the com-*

mitter C succeds in decommiting to x then E_{com} extracts x with overwhelming probability:

$$Pr[E_{com}(c, Q) = x] = 1 - negl(n)$$

where Q is the list of oracle queries made by C.

At first, we have a Σ-protocol $\Pi = (a, c, z)$. We transform it to get Π' by applying this: the prover chooses uniformely at random two challenges c_0, c_1 and precomputes the corresponding responses z_0, z_1. Then she commits to them $\gamma_0 = Com(z_0), \gamma_1 = Com(z_1)$ with an online extractable commitment scheme and sends the pair of executions $(a, c_0, \gamma_0), (a, c_1, \gamma_1)$. The verifier choose at random a challenge bit $b \leftarrow \{0, 1\}$ and sends it back to the prover. $\mathcal{P}$ is now obligated to open the commitment indicated by the challenge bit γ_b (and send it to $\mathcal{V}$) and $\mathcal{V}$ can now verify that (a, c_b, γ_b) is valid. We observe that Π' has special soundness but as the challenge space is restricted to $\{0, 1\}$ the soundness error is $1/2$.

The above is the reason to construct the second protocol Π'', which is the fianl outputting protocol of step 1. For Π'' we use the common soundness-error reduction trick of parallel repetitions. So the prover instead of sending one pair of executions, now sends t pairs for t different first moves $a_1, ..., a_t$. So now the soundness error is 2^{-t}. Thus if we set $t = n$ we get a negligible soundness error. The procedure described is in Fig.4.6. The resulting protocol Π'' is the Ω-protocol output of the first step.

Theorem 4.3. *If Π is a Σ-protocol for language L and the commitment scheme is online extractable then Π'' is an Ω-protocol for L in the random oracle model.*

Proof sketch. (The complete proof can be found in [64])

Completeness and **HVZK** are inherent from Π.

Special-Soundness: Π' is special sound as two repetitions with the same first move $(\{(a, c_0, \gamma_0), (a, c_1, \gamma_1)\},$ and $(\{(a, c_0, \gamma_0), (a, c_1, \gamma_1)\}, 1, z_1)$ give two instances (a, c_0, z_0) and (a, c_1, z_1) of the initial protocol Π. So from Π's special soundness we can extract a witness.
Finally, Special soundness of Π'' comes directly from special soundness of Π'.

Online Extraction: We will show that Π' has an online extractor. Let E be the desired online extractor of Π'. E can only access $\mathcal{P}^*$'s random oracle queries and view of $\mathcal{V}$. Let Q be the list of them. E gets the view of $\mathcal{V}$ and retrieves $\{(a, c_0, \gamma_0), (a, c_1, \gamma_1)\}$. Then uses the online extractor of the commitment scheme $E_{com}(view, Q)$ to extract the openings z_0 and z_1. So now from special soundness E can use $(a, c_0, z_0), (a, c_1, z_1)$ to extract the witness w.
In the same way we can construct an online extractor for Π'' □

Step 2: In step 2 we take the Ω-protocol Π'' from step 1 and transform it to an online Simulation-Extractable non-interactive zero-knowledge $\tilde{\Pi}$. The final protocol after Fiat-Shamir transformation is in Fig.4.7. Intuitively, the simulation-extractability comes from the online-extraction property of Π'' and from the simulation capability coming HVZK property. The

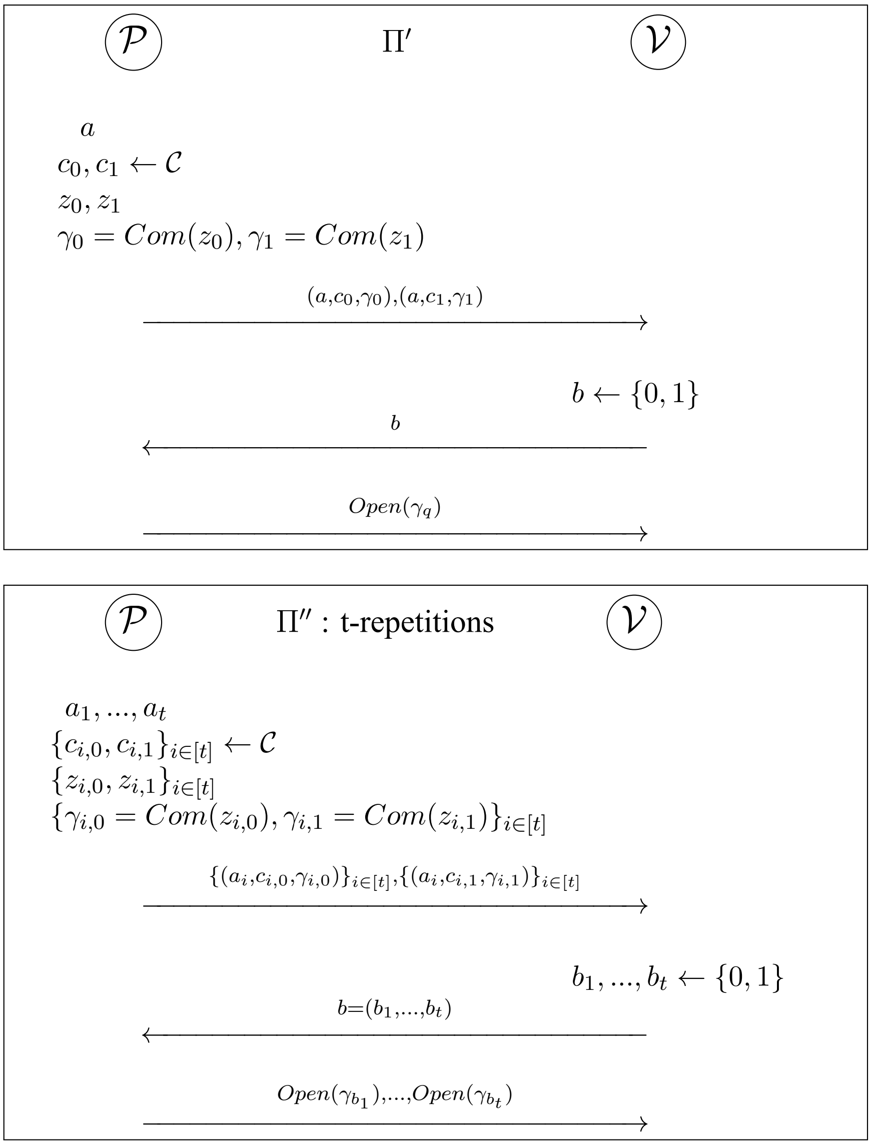

$\mathcal{P}$
Π'
$\mathcal{V}$
a
$c_0, c_1 \leftarrow \mathcal{C}$
z_0, z_1
$\gamma_0 = Com(z_0), \gamma_1 = Com(z_1)$
$(a,c_0,\gamma_0),(a,c_1,\gamma_1)$
$b \leftarrow \{0,1\}$
b
$Open(\gamma_q)$
$\mathcal{P}$
Π'' : t-repetitions
$\mathcal{V}$
$a_1, ..., a_t$
$\{c_{i,0}, c_{i,1}\}_{i\in[t]} \leftarrow \mathcal{C}$
$\{z_{i,0}, z_{i,1}\}_{i\in[t]}$
$\{\gamma_{i,0} = Com(z_{i,0}), \gamma_{i,1} = Com(z_{i,1})\}_{i\in[t]}$
$\{(a_i,c_{i,0},\gamma_{i,0})\}_{i\in[t]},\{(a_i,c_{i,1},\gamma_{i,1})\}_{i\in[t]}$
$b_1, ..., b_t \leftarrow \{0,1\}$
$b=(b_1,...,b_t)$
$Open(\gamma_{b_1}),...,Open(\gamma_{b_t})$

Figure 4.6: Step 1, presented in substeps. A transformation $\Pi \Rightarrow \Pi''$.

non-interactive property comes, obviously, from Fiat-Shamir. We omit the formal definition, which is much more technical. The reader can find it in the original paper of Anonize [6].

So the above abstract transformation leaves us only the responsibility to find a Σ-protocol proving intended statements. Afterwards, by applying the above transformation we get an oSE NIZK protocol, which has much stronger security properties. This facilitates us as it is often easier to construct a Σ-protocol for a language L.

The language that concerns us for Anonize is:

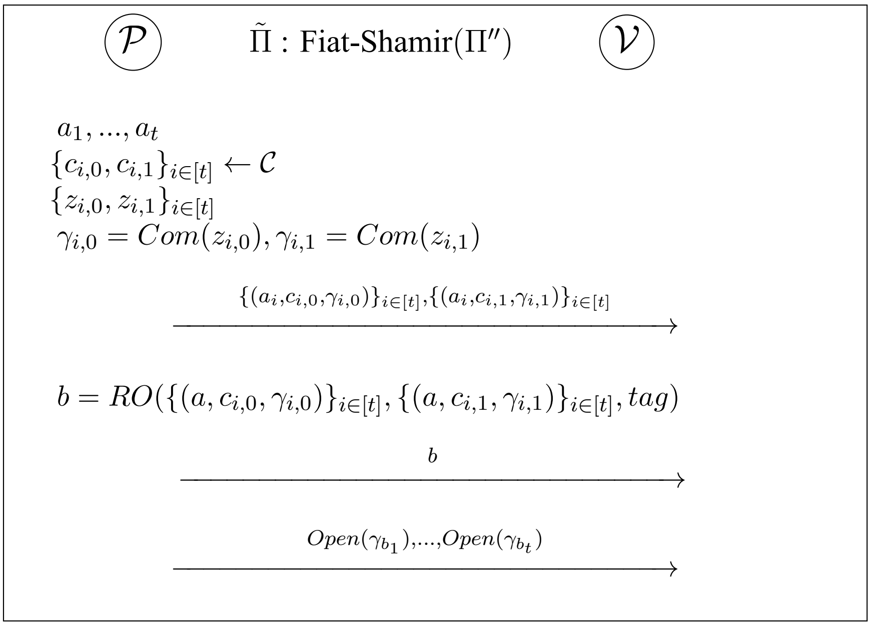

$\mathcal{P}$
$\tilde{\Pi}$: Fiat-Shamir(Π'')
$\mathcal{V}$
$a_1, ..., a_t$
$\{c_{i,0}, c_{i,1}\}_{i\in[t]} \leftarrow \mathcal{C}$
$\{z_{i,0}, z_{i,1}\}_{i\in[t]}$
$\gamma_{i,0} = Com(z_{i,0}), \gamma_{i,1} = Com(z_{i,1})$
$\{(a_i,c_{i,0},\gamma_{i,0})\}_{i\in[t]},\{(a_i,c_{i,1},\gamma_{i,1})\}_{i\in[t]}$
$b = RO(\{(a, c_{i,0}, \gamma_{i,0})\}_{i\in[t]}, \{(a, c_{i,1}, \gamma_{i,1})\}_{i\in[t]}, tag)$
b
$Open(\gamma_{b_1}),...,Open(\gamma_{b_t})$

Figure 4.7: The final protocol

$$L = \left\{ \begin{array}{c} (tok, vid, vk_{RA}, vk_{SA}) : \exists\ s, id, \sigma_s, \sigma_{id}^{vid}\ s.t. \\ i)\ Ver'(vk_{RA}, (id, s), \sigma) = 1 \\ ii)\ Ver(vk_{SA}, (vid, id), \sigma_{id}^{vid}) = 1 \\ iii)\ f_s(vid) = tok \end{array} \right\}$$

So the user id wants to prove that:

i) has a signature from the RA on (id, s_{id}): $\sigma = (\sigma_1, \sigma_2) = (g^x(u^{id}v^{s_{id}}h)^w, g_2^w)$

ii) has a signature from the SA on (vid, id): $\sigma_{id}^{vid} = (\sigma_3, \sigma_4) = (g^y(u_{SA}^{vid}v_{SA}^{id}h_{SA})^{w_{SA}}, g_2^{w_{SA}})$

iii) The token is valid: $tok = e(g, g_2)^{1/(s_{id}+m)}$

and of course that id, s_{id}, vid are the same in each case.

Hohenberger et al. instead of constructing a Σ-protocol with 6 witnesses $s_id, id, \sigma_1, \sigma_2, \sigma_3, \sigma_4$, which is definitely a tough case, they use a trick to decrease the witnesses to 4 and thus the protocol complication. They rerandomize the signatures and send the second piece of each (rerandomized) signature:

Choose at random d_1, d_2 and compute:

$$\tilde{\sigma} = (s_1, s_2) = (\sigma_1 \cdot ((u^{id}v^{s_{id}}h)^{d_1}), \sigma_2 \cdot g_2^{d_1}) = (g^x(u^{id}v^{s_{id}}h)^{w+d_1}, g_2^{w+d_1})$$
$$\tilde{\sigma}_{id}^{vid} = (s_3, s_4) = (\sigma_3 \cdot (u_{SA}^{vid}v_{SA}^{id}h_{SA})^{d_2}, \sigma_4 \cdot g_2^{d_2}) = (g^y(u_{SA}^{vid}v_{SA}^{id}h_{SA})^{w_{SA}+d_2}, g_2^{w_{SA}+d_2})$$

The user (the prover) sends to the SA (the verifier) s_2 and s_4 and now is left to prove that:

$$L' = \left\{ \begin{array}{c} (tok, vid, vk_{RA}, vk_{SA}, s_2, s_4) : \exists\, s, id, s_1, s_3 \; s.t. \\ i) e(g, g_2)^x \cdot e(h, s_2) = e(s_1, g_2) \cdot e(u^{id} v^{s_{id}}, s_2)^{-1} \\ ii) e(g, g_2)^y \cdot e(u_{SA}^{vid} h_{SA}, s_4) = e(s_3, g_2) \cdot e(v_{SA}^{s_{id}}, s_4)^{-1} \\ iii) e(g, g_2) \cdot tok^{-vid} = tok^{s_{id}} \end{array} \right\}$$

where x, y are the secret keys of RA and SA resp.
We make some useful observation:

1. The signature scheme allows us to rerandomize the signature without spoiling its validity (malleability).
2. A prover can provide a valid rerandomized signature only if she has an initial one.
3. If the user was to give the initial signature she would betray her identity as the adversary has access to the signatures of the RA and SA. That's why rerandomization is needed.

So the statements to be proven are changed to make it more simple and make it much easier to find a proper Σ-protocol:

$\mathcal{P}$ $\mathcal{V}$

$b_1, b_2 \leftarrow \mathbb{Z}_q$ and $J_1, J_2 \leftarrow \mathbb{G}$
$E_1 = e(J_1, g_2) \cdot e(u^{b_1} v^{b_2}, s_2)^{-1}$
$E_2 = e(J_2, g_2) \cdot e(v_{SA}^{b_2}, s_4)^{-1}$
$E_3 = tok^{b_2}$

$$\xrightarrow{(E_1, E_2, E_3)}$$

$c \leftarrow \mathbb{Z}_q$

$$\xleftarrow{c}$$

$z_1 = b_1 + c \cdot id$
$z_2 = b_2 + c \cdot s_{id}$
$z_3 = s_1^c \cdot J_1$
$z_4 = s_3^c \cdot J_2$

$$\xrightarrow{(z_1, z_2, z_3, z_4)}$$

Verification

$$E_1 \cdot e(g, g_2)^{xc} \cdot e(h, s_2)^c \overset{?}{=} e(z_3, g_2) \cdot e(u^{z_1} v^{z_2}, s_2)^{-1}$$
$$E_2 \cdot e(g, g_2)^{yc} \cdot e(u_{SA}^{vid} h, s_2)^c \overset{?}{=} e(z_4, g_2) \cdot e(v^{z_1}, s_2)^{-1}$$
$$E_3 \cdot e(g, g_2)^c \cdot tok^{-c \cdot vid} \overset{?}{=} tok^{z_2}$$

Figure 4.8: Σ-protocol for L'

The above is transformed into an oSE NIZK using the transformation described and we have the protocol we need.

One would wonder why do we need such a strong protocol for an ad-hoc scheme. The answer to this is that the protocol gives much freedom to its users. Everyone can initiate a Survey and become a Survey Authority and even in many Surveys at the same time, while at the same time play the role of a user in other surveys. And this can be made simultaneously! In fact, concurrent scenarios are natural in this ad-hoc setting.

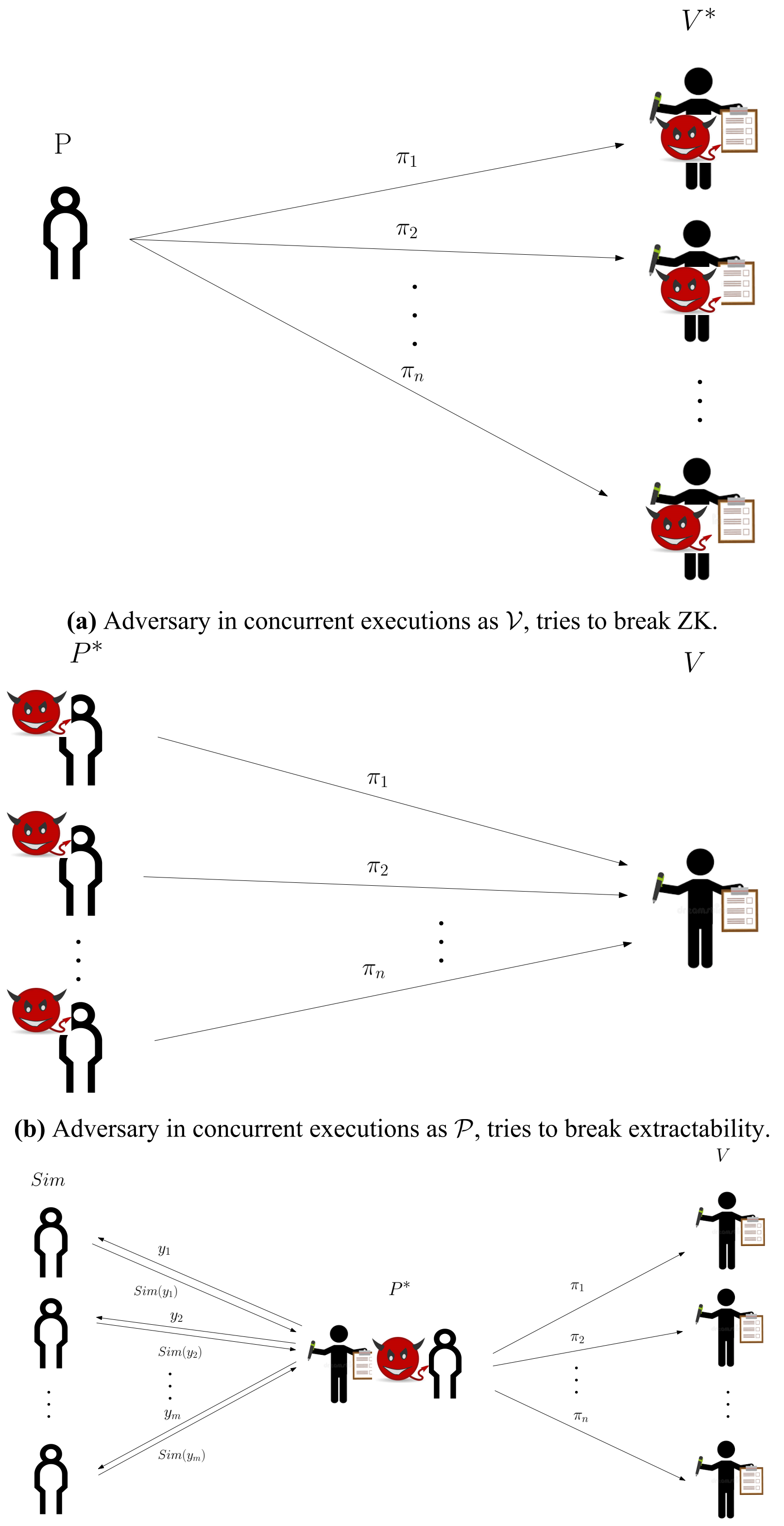
So a multi-services scheme is really convenient but comes with great security requirements. Indeed, if we observe the security definitions we see that the adversary $\mathcal{A}$ is given much freedom:

- $\mathcal{A}$ can concurrently interact as a verifier with a user (prover) in many executions, by controlling many surveys in which the user is participant and the adversary is SA (Fig.4.9a). This leads to concurrent ZK requirement.

- $\mathcal{A}$ can concurrently interact as a prover with an honest verifier in many executions. This can be done if the adversary controls many users participating in the same Survey (Fig. 4.9b). This leads to concurrent PoK requirement.

- $\mathcal{A}$ may receive some proofs as SA and then try to create an invalid proof and send it to a Verifier (SA) as a user (Fig.4.9c). This leads to simulation soundness requirement.

If we combine the second and the third attack we get online simulation-extractability requirement. Finally, concurrent ZK holds as the protocol is non-interactive.



(a) Adversary in concurrent executions as $\mathcal{V}$, tries to break ZK.

(b) Adversary in concurrent executions as $\mathcal{P}$, tries to break extractability.

(c) Adversary first receives proofs as $\mathcal{V}$, and then tries to break soundness as $\mathcal{P}$.

Figure 4.9: Possible attacks on the zero knowledge protocol

4.3 Towards strenghtening Anonize: some remarks

In the section we present some observations on Anonize that in writer's opinion could lead to real life problems. So we present five types of scenarios that could motivate for security improvements on the protocol.

4.3.1 Malicious RA-authenticity breaking

In the definition of authenticity of ad-hoc survey schemes the RA is considered honest (in contrast to anonymity where can be malicious). So we observe that a malicious RA can collaborate with a malicious user id, or even be the user, and sign many s_{id}. Afterwards, a user can cast as submissions as the number of the signed s_{id} she possesses.

This is possible, because the identity of the user is preserved and signature from RA and SA is proven in zero knowledge. So nobody can check that the submission was made with the same SA signature many times because it is conceived. On the other hand $tok = f_{s_{id}}(vid)$, which is visible would be different for each (maliciously signed) s_{id} and the malicious user would have many valid tokens.

Although, the above scenario requires a very strong adversary, who controls the registration authority, it still causes great damage to the reliability of the survey. That's because the adversary can cast as many answers as she wants and completely change the result.

In chapter 5 we will define alternative ad-hoc survey schemes in an effort to confront this type of attack.

Honest Submission

ZK proof of knowledge for:

1. (id, s_{id}) valid (RA) signature
2. (vid, id) valid (SA) signature
3. $tok = f_{s_{id}}(vid)$ valid

Malicious Submission

Malicious RA signs a second credential to user id in collusion with the user.
ZK proof of knowledge for:

1. (id, s^*_{id}) valid (RA) signature
2. (vid, id) valid (SA) signature
3. $tok = f_{s^*_{id}}(vid)$ valid

Figure 4.10: Double submission from id. Above are the statements.

4.3.2 Vote Buying (lack of receipt-freeness)

An important notion in (electronic and physical) Voting systems is receipt-freeness. That is a voter should not posses any information that would help her prove how she voted. The vote itself should hide any evidence that a malicious user can deploy. This prevents a voter from selling her vote to a party. The notion was first introduced in the seminal work of Benaloh and Tuinstra [75] and was subsequently studied and formalized [76, 77, 78]. Today, receipt-freeness is such significant that an e-Voting system is not considered safe if it does not satisfy the notion. Our general concern for receipt-freeness, that applies in this situation too, came from prior works of Pagourtzis, Grontas, Zacharakis and Zhang [7, 8, 9].

We state that Anonize does not have receipt-freeness.

- One clear receipt is d_2, the randomness used to rerandomize the SA signature before sending s_4 to the verifier. The user provides the initial signature σ_4 and d_2 and claims that $s_4 = \sigma_4^{d_2}$ for a Submission. Only the real sender of s_4 (thus sender of the Submission) can provide d_2. Computing a d_2 requires solving a discrete logarithm. We note that σ_4 is in a public list and s_4 is part of the Submission-vote.

- Another receipt is s_{id}, the credential. By providing s_{id} one can verify that the token of a submitted message is created by the id by checking that $tok = f_{s_{id}}(vid)$. We state, though, that this is an extreme case where the vote-seller would want deeply to sell her vote. That's because s_{id} is master credential that would allow the buyer to identify the user id in all past Surveys that has participated. For future Surveys it is not a problem as Anonize give user the option to register again, if she claims credential loss.

- b_2, the random value in the commit phase of the Σ-protocol is another receipt as computing a b_2 s.t. $E_3 = tok^{b_2}$ is reduced to DLP. But this betrays s_{id} so it is equivalent to providing s_{id}

On the above we assumed that Submissions are either public or leaked to the vote-buyer. We note that Submissions must be made public in applications where public verifiability is demanded. It is safe to presume that in important Surveys (which are the only scenarios that motivate one to pay for a Submission) public verifiability is necessary, thus Submissions are known to the malicious vote-buyer.

One may wonder why would someone pay for Submissions in Surveys. The answer to this is too subjective and is outside the scope of cryptography. Our view is that paying for opinion submissions is not far from real life situations. Furthermore, there may be direct economic impacts of Survey results.

Real life motivation One application of Anonize is in Brave Browser. Brave Browser is a new browser that offers ad-blocking. But advertisements are important for Internet economy because they are the main source of profit for most website providers. So Brave give user the option to donate an amount and this amount is alloted fairly to providers based on the time that user spent on their website. For example if a user spent 25% of browsing time to ”website.web” and donates 10\$, ”website.web” is going to get 2.5\$ (actually it would e a little lower as Brave company keeps a fee).

However, users cannot outsource their history to Brave because then their privacy is violated deeply. So it uses Anonize to get the feedback and users are anonymous. Furthermore, the history is not solidly sent. Instead every donation is divided into micropayments and each day (or in specified time intervals) a micropayment and a website preference are submitted via Anonize. And each website has as much odds to be chosen as the time (in percentage) that the user spent to the website. After many submissions the payments converge to the real percentages of preferences. We note that payments themselves are anonymous too, as they are made through a cryptocurrency named

Basic Attention Token (BAT). To summarize, donation payments in Brave need a cryptocurrency for anonymous payments and Anonize for anonymous website preference submission (see fig.4.11)

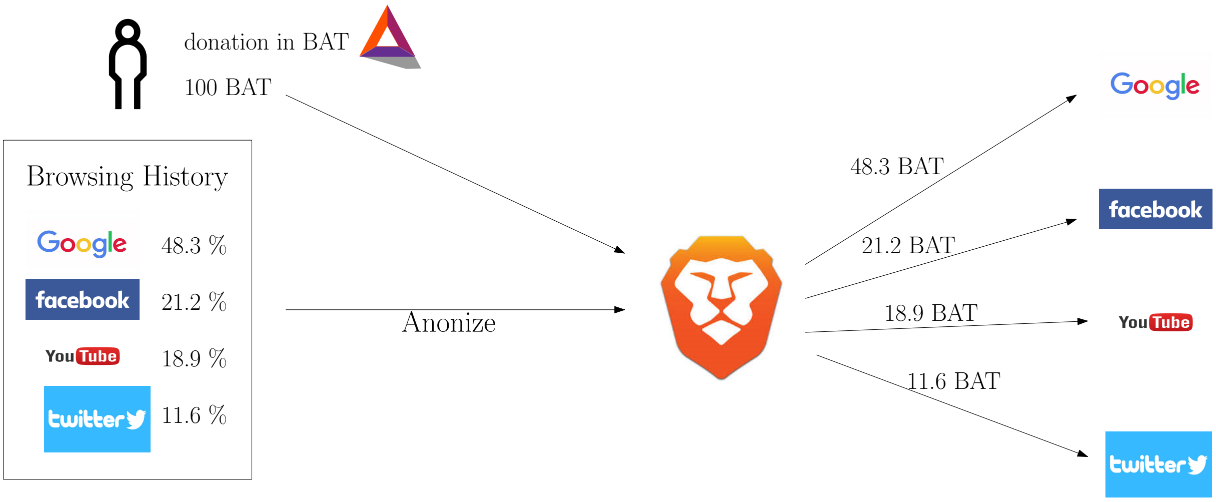


Figure 4.11: The procedure of donations' allocation in Brave.

We can see a motivation of vote-selling in the above use of Anonize. A user may collude with a malicious website ”evil.web” to sell a vote to the latter. The vote-submission would always be ”evil.web”, so that always this website takes the amount. Afterwards, a user will take an agreed percentage back. So, user and ”evil.web” are both happy and the loser is the system Brave. We state, though, that this scenario is not realistic in the current, because if one donates she will not desire to take some of her money back, as donation itself is unselfish. So she will not be motivated to sell her vote (although the provider is still motivated to buy one!).

Receipt-freeness is not yet, in our opinion, necessary in above. But it would be if payments were compulsory, for instance in a pro version. Furthermore, the described setting is only a motivation. Independently, receipt-freeness is a property that is desired in a large-scale Survey submission system.

4.3.3 A side-channel timing attack

This attack was found in practice and may be dealt with easily by cautious users of Anonize. After using Anonize to initiate a Survey we noticed that the Survey Initiator notifies users about survey's creation via e-mail. Actually, SA chooses which users she wants to inform and even not send e-mail to anyone. So if the survey authority is malicious and the user incautious it is possible that only one user is informed, although many users are included formally with signatures. This means that seemingly many users are authorized, but only one answer is expected at the time from the SA. And the submission leaves the message as plaintext, so the anonymity is broken. After one user is trapped and accomplishes a submission the process can go on to another user.

We mention again that this entail a user that trusts some private method of communication to be informed like a private e-mail with a single receiver or an oral notification for the existence of the Survey.

4.3.4 Participants list is public

An issue that Anonize does not resolve is that participant identities of a survey must be public. Otherwise, one cannot check if she is authorized. The problem is that in the list the identities are as plaintexts so anyone can see who are authorized to complete a survey. In some real life situations, it would be really uncomfortable for a user to be publicly seen in a Survey participants' list and in some others it would actually be considered serious privacy breaking. One example is medical Surveys, where participants are often patients. This weakness of Anonize is observed by the original authors in the introductory paper [6].

4.3.5 Answers are submitted as plaitexts

This can lead, in some cases, to deanonymization of the user. In fact, answers are themselves linked to user's identity in some Survey. For example if a Survey asks users their age, this is something that can lead to straight deanonymization.

This, in our opinion, is more or less unavoidable. That's because survey collector, who conducts the research wants to process them in the end to deduce statistical values. In the current state of cryptography, there are statistical queries on ciphertexts that would be really difficult to be done. On the other hand, processing ciphertexts or applying a function to ciphertexts are two problems that are interesting on their own. In fact, homomorphic encryption and functional encryption are two fields, independent from anonymous questionnaires and e-voting, that are of great interest today and deal with these kinds of problems.

Chapter 5

Alternative ad-hoc survey scheme constructions

5.1 Traceable Ring Signatures

Traceable Ring Signatures are extensions of ’plain’ Ring Signatures. In this concept a member of a group, namely ring, signs anonymously a message with respect to a tag and the ring, exactly as in ring signatures. The distinction is that for a particular pair of tag and ring each signer can sign only once. Signing twice will cause to deanonymization, namely traceability, except if a signer signs twice the same message where we have linkability but without anonymity revocation.

So, we can view traceable ring signatures as a scheme that restricts anonymity of ’classic’ ring signatures in a fruitful way. The first signature of a signer remains unlinkable, while the second (and any upcoming) is traced (or linked). This restriction suits perfectly the needs of Anonymous Surveys, but others schemes’ as well like e-Voting or e-Cash. That’s because double signatures (double submission, double vote or double spending respectively) are failures in these systems. We note that a signature is traceable strictly in the context of a certain event (for example a Survey), represented by a tag value, and a ring, represented by a list of public keys $pk_{[n]} = \{pk_1, ..., pk_n\}$. That’s very convenient as it allows a user with a certain public key participate in many events and rings.

The notion of traceable ring signatures was introduced by Fujisaki and Suzuki in 2007[11]. The signature is based on a Σ-OR protocol that proves that the signer belongs to the ring and the Fiat-Shamir heuristic to make it NIZK. So the ring signature part is roughly accomplished with the 1-out-of-n participation NIZK. The traceability is accomplished by forcing the i-th part of a signature $\sigma = (\sigma_1, ..., \sigma_n)$ be the same if the signature comes from the same user i (of course if the tag and the ring are the same). Actually, $\sigma_i = h^{sk_i}$ depends only from the secret key sk_i of the user. So, same σ_i traces two signatures. This alone would make it easy for a malicious ring member to forge a signature and entrap the user i by simply copying σ_i in the i-th position inject her $\sigma_j = h^{sk_j}$ and proving membership with her real sk_j. On account of this, all σ_j, $j \in [n]$ depend from σ_i and (L, m), thus an adversary j cannot inject her $\sigma_j = h^{sk_j}$ to the signature and has to prove membership with sk_i, which of course does not posses.

We use the notation $X_{[n]} = \{X_1, ..., X_n\}$.

Traceable Ring Signature

Let G be a multiplicative group of prime order q with generator $g \in G$ and $H : \{0,1\}^* \to G$, $H' : \{0,1\}^* \to G$, $H'' : \{0,1\}^* \to \mathbb{Z}_q$ three random oracles. Let $L = (tag, pk_{[n]})$

- $Gen(1^n)$: is executed from player i.
  Pick a secret key $sk_i \leftarrow \mathbb{Z}_q$ randomly and compute the public key $pk_i = g^{x_i}$.
  We assume a **PKI** and that i registers her public key pk_i.

- $Sign_{sk_i}(L, m)$: is executed from player i.

  1. Compute $h = H(L)$ and $\sigma_i = h^{sk_i}$
  2. Compute $A_0 = H'(L, m)$, $A_1 = \left(\frac{\sigma_i}{A_0}\right)^{1/i}$ and for each $j \neq i$ compute $\sigma_j = A_0 A_1^j$.
     $\sigma_{[n]} = (\sigma_1, ..., \sigma_n) = (A_0^{\frac{i-1}{i}} h^{sk_i \frac{1}{i}}, A_0^{\frac{i-2}{i}} h^{sk_i \frac{2}{i}}, ..., h^{sk_i}, ..., A_0^{\frac{i-n}{i}} h^{sk_i \frac{n}{i}})$
  3. For the Language $\mathcal{L} = \{(L, h, \{\sigma_1, ..., \sigma_n\}) | \exists sk_i \text{ s.t. } pk_i = g^{sk_i} \text{ and } \sigma_i = h^{sk_i}\}$ Compute a NIZK:
     - Pick randomly $w_i, \{c_j, z_j\}_{j \neq i} \leftarrow \mathbb{Z}_q$ and compute the commit:
       $a_i = g^{w_i}, a_j = g^{z_j} pk_i^{c_j}$ for all $j \neq i$
       $b_i = h^{w_i}, b_j = h^{z_j} \sigma_i^{c_j}$ for all $j \neq i$
     - Compute the challenge:
       $c = H''(L, A_0, A_1, a_{[n]}, b_{[n]})$
       $c_i = c - \sum_{j \neq i} c_j \ (mod\ q)$
     - Compute the response:
       $z_i = w_i - c_i sk_i$

     Output the proof $\pi = \{c_{[n]}, z_{[n]}\} = \{(c_1, .., c_n), (z_1, ..., z_n)\}$

  output: The signature is $\sigma = (A_1, \pi) = (A_1, c_{[n]}, z_{[n]})$

- $Verify_L(m, \sigma)$:

  1. Compute $h = H(L)$, $A_0 = H'(L, m)$, $\sigma_j = A_0 A_1^j$ for each $i \in [n]$
  2. Compute $a_i = g^{z_i} pk_i^{c_i}$ and $b_i = h^{z_i} \sigma_i^{c_i}$ for each $i \in [n]$
  3. Check if $H''(L, A_0, A_1, a_{[n]}, b_{[n]}) = \sum_{i=1}^{n} c_i \ (mod\ q)$

  output: 1 if the last equation holds or 0 otherwise.

- $Trace_L(m, \sigma, m', \sigma')$: where $\sigma = (A_1, c_{[n]}, z_{[n]})$ and $\sigma' = (A'_1, c'_{[n]}, z'_{[n]})$

  1. Compute $h = H(L)$, $A_0 = H'(L, m)$, $\sigma_j = A_0 A_1^j$ for each $i \in [n]$
     and $A'_0 = H'(L, m')$, $\sigma'_j = A'_0 A_1'^j$ for each $i \in [n]$
  2. Compare σ_i and σ'_i for each $i \in [n]$

$$output = \begin{cases} pk_i & \text{if } \sigma_i = \sigma'_i \text{ and } \sigma_j \neq \sigma'_j \text{ for all } j \neq i \qquad \textbf{(1 exactly equal)} \\ \text{linked} & \text{if } \sigma_i = \sigma'_i \text{ for all } i \in [n] \qquad \textbf{(n exactly equal)} \\ \text{indep} & otherwise \qquad \textbf{(0 OR 2} \leq \textbf{equal} \leq \textbf{n-1)} \end{cases}$$

Figure 5.1: Fujisaki-Suzuki Traceable Ring Signature

We must clarify that when two signatures are 'indep' the regular condition is that all $\sigma_i \neq \sigma'_i$.

If two σ_i are equal $\sigma_i = \sigma'_i$ and $\sigma_j = \sigma'_j$ then:

$$\sigma_i = \sigma'_i \Rightarrow A_0 A_1^i = A'_0 (A'_1)^i \Rightarrow A_0^j A_1^{ij} = (A'_0)^j (A'_1)^{ij}$$
$$\sigma_j = \sigma'_j \Rightarrow A_0 A_1^j = A'_0 (A'_1)^j \Rightarrow A_0^i A_1^{ij} = (A'_0)^i (A'_1)^{ij}$$

If we divide the equations we get: $A_0^{j-i} = (A'_0)^{j-i} \Rightarrow A_0 = A'_0 \Rightarrow H'(L,m) = H'(L,m')$
So we conclude that there must occur a collision in the random oracle H', which, of course, happens with negligible probability.

Security Properties Originally, in ring signature schemes we expect Anonymity and Unforgeability to hold. Of course, these properties should, also, be present in traceable ring signatures. Furthermore, we require tag-linkability, which means that no adversary can bypass the traceability for two signatures, even if she controls every user in the ring. That is, she cannot pass two signatures with the same sk without being noticed (traced or linked). Another, security notion which is not so obvious is called exculpability. This states that an adversary cannot entrap a user by passing two signatures that trace her. This would exclude a user from the event, thus it is something that requires caution.

We present the formal definitions below: Of course we need the above experiments to happen with negligible probability.

Tag-Linkability Experiment

1. $\big(L, \{(m^{(1)}, \sigma^{(1)}), ..., (m^{(n+1)}, \sigma^{(n+1)})\}\big) \leftarrow \mathcal{A}(1^n)$:
   Adversary controls all the users of the ring, creates all the public and secret keys and the ring of n users $pk_{[n]} = \{pk_1, ..., pk_n\}$
2. **output** 1 if:
   - $Verify_L(m^{(i)}, \sigma^{(i)}) = 1$ for all $i \in [n+1]$
   - $Trace_L(m^{(i)}, \sigma^{(i)}, m^{(j)}, \sigma^{(j)}) =$ indep for all $i, j \in [n+1]$ and $i \neq j$

Anonymity Experiment

1. $(pk_0, sk_0), (pk_1, sk_1) \leftarrow Gen(1^n)$.
2. $b \leftarrow \{0, 1\}$ is chosen randomly
3. The adversary $\mathcal{A}$ has oracle access to
   - $Sign_{sk_b}$ oracle : only with one message for each L
   - $Sign_{sk_0}, Sign_{sk_1}$ oracles for each L.
   - Cannot query $Sign_{sk_b}(L, m)$ and $Sign_{sk_0}(L, \tilde{m})$ or $Sign_{sk_1}(L, \tilde{m})$ for the same L.
   - Each query has the restriction that L that contains both pk_0 and pk_1
4. $b' \leftarrow \mathcal{A}^{Sign_{sk_b}, Sign_{sk_0}, Sign_{sk_1}}(pk_0, pk_1)$

output 1 if: $b = b'$

Exculpability Experiment

1. $pk \leftarrow \mathcal{A}(1^n)$: $\mathcal{A}$ targets a pk that is stored to the **PKI** and (pk, sk) has been generated by $Gen(1^n)$.
2. The adversary $\mathcal{A}$ has oracle access to $Sign_{sk}$ on any (L, m) as long as L contains pk.
3. $(L, m, \sigma), (L, m', \sigma') \leftarrow \mathcal{A}^{Sign_{sk}}(pk)$

output 1 if:

- $Verify_L(m, \sigma) = 1$ and $Verify_L(m', \sigma') = 1$.
- At most one of $(m, \sigma), (m', \sigma')$ is linked with a query answer. This means that $\mathcal{A}$ forged at least one.
- $Trace_L(m, \sigma, m', \sigma') = pk$

Of course we need the above experiments to happen with negligible probability, except for the Anonymity Experiment which we want to happen with probability negligibly close to $1/2$.

We briefly explain the intuition behind the proof of security for these properties in the above traceable ring signature scheme. For simplicity we don't include the formal proofs, though one can find them in the original paper [11]:

- **Tag-Linkability:** Say an adversary manages to pass an untraced new signature $(L, m^{(n+1)}, \sigma^{(n+1)})$ (other than the n first). Say
  $Trace_L(m^{(i)}, \sigma^{(i)}, m^{(n+1)}, \sigma^{(n+1)}) =$ indep
  Then either $\mathcal{A}$ cheated by finding two $\sigma_k^{(i)} = \sigma_k^{(n+1)}$ and $\sigma_l^{(i)} = \sigma_l^{(n+1)}$. But this is negligibly rare, as stated in the previous page. Or $\sigma_k^{(i)} \neq \sigma_k^{(n+1)}$ for all $k \in [n]$ and cheated in the NIZK : $\sigma_i^{(n+1)} \neq h^{sk_i}$ for every $i \in [n]$. That is without an sk_i achieved in producing a NIZK, which again happens with negligible probability.

- **Anonymity:** Anonymity reduces to Decisional Diffie-Hellman problem. We give the intuition behind the reduction. Let a tuple $(g, g_1, u, T) = (g, g^x, g^y, T)$ that we want to find if is or not DDH tuple. In the Anonymity experiment do these:

  - flip a coin and set b its value
  - Set $pk_b = g^y, \quad pk_{1-b} = g^y \cdot g^t = g^{y+t} \quad$ (t random)
  - For each H query answer $h = H(L) = g^{r_1} g_1^{r_2} = g^{r_1 + x r_2}$
  - For $Sign_{sk_b}$ set $\sigma_b = u^{r_1} T^{r_2}$
  - For $Sign_{sk_{1-b}}$ set $\sigma_{1-b} = u^{r_1} T^{r_2} h^t$
  - Simulate the NIZK proofs

  If DDH tuple then: $T = g^{xy}$ and

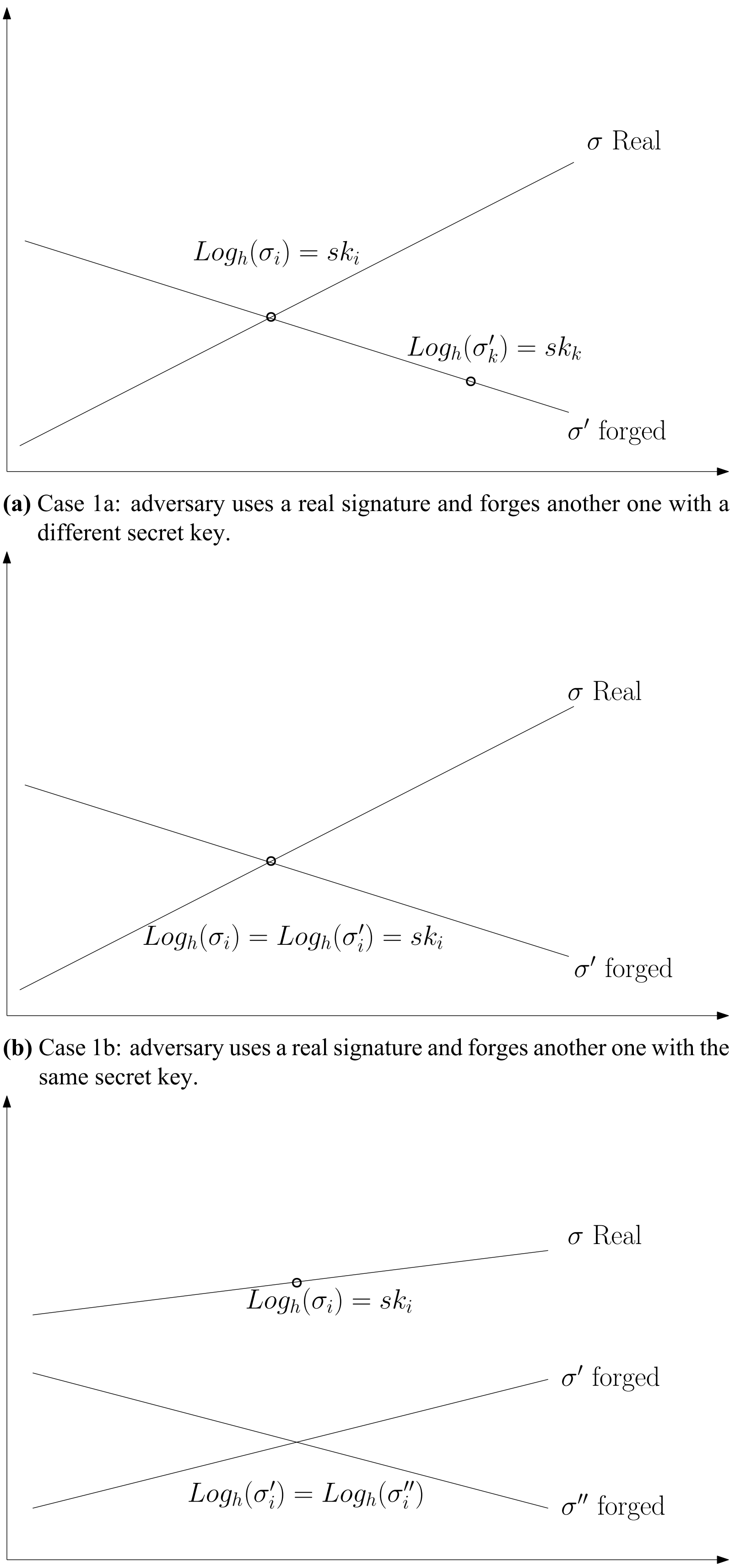
  $$\begin{aligned} \sigma_b &= g^{yr_1} g^{xyr_2} = g^{y(r_1 + yr_2)} = h^y \\ \sigma_{1-b} &= g^{yr_1} g^{xyr_2} h^t = g^{y(r_1 + yr_2)} h^t = h^{y+t} \end{aligned}$$

  So they are valid signatures for $sk_b = y$ and $sk_{1-b} = y + t$.
  If the tuple is not DDH then the signature is random
  So if an adversary can break the Anonymity she will distinguish the first (valid) signature from the second and we can exploit it to break the Decisional Diffie-Hellman.

- **Exculpability:** For exculpability we distinguish two cases: either the adversary uses a signature she got from the oracle and forges the second one or she forges both signatures. And in case 1 we have two subcases. Either $\mathcal{A}$ used the same witness sk_i with the target or different sk_k.

  We state again that the forged signature (with overwhelming probability) go with a known witness for the NIZK proof. So the adversary should know $i : Log_h(\sigma_i) = Log_g(pk_i)$.

  Furthermore, parts of each signature $\sigma_j = A_0 A_1^i$ have a line where points $(j, Log_h(\sigma_j))$ lay on the line $y = Log_h(A_1) \cdot i + Log_h(A_0)$. This comes directly from σ_j equation. We will use this visualization to explain better why excupability stands, see fig.5.2



(a) Case 1a: adversary uses a real signature and forges another one with a different secret key.

(b) Case 1b: adversary uses a real signature and forges another one with the same secret key.

(c) Case 2: adversary outputs two forged signatures. σ is not outputted (it is there for better comprehension).

Figure 5.2: Lines that $(j, Log_h(\sigma_j))$ lay on for each Exculpability attack.

We will show that in each case adversary success in entrapping player i only with negligible probability.

Case 1a: $sk_1, ..., sk_n$ are fixed, thus sk_i, sk_k are fixed and it happened that line $y = Log_h(A_1) \cdot i + Log_h(A_0)$ passes through these (fixed) points. So $Log_h(A_0)$ is determined but $A_0 = H'(L, m)$ so adversary has probability at most $(n-1) \cdot \frac{q_{H'}}{q} = negl(\lambda)$ to succeed, where $q_{H'}$ is the number of H' queries, which is polynomial.

Case 1b: Adversary needs to find sk_i to create a valid NIZK proof. So she has to solve $\sigma_i = h^{sk_i}$ for sk_i which is the DLOG problem.

Case 2: For each forged signature adversary has a secret key say sk_k and sk_l. So:

$$sk_k = Log_h(A'_1) \cdot k + Log_h(A'_0)$$
$$sk_l = Log_h(A''_1) \cdot l + Log_h(A''_0)$$

Also we assumed that adversary entrapped player i with σ' and σ'' so:

$$\sigma'_i = \sigma''_i \Rightarrow Log_h(\sigma'_i) = Log_h(\sigma''_i)$$
$$Log_h(A'_1) \cdot i + Log_h(A'_0) = Log_h(A''_1) \cdot i + Log_h(A''_0)$$

Again sk_k, sk_l are fixed so from three equations above we get an equation:

$$c_1 \cdot Log_h(A'_0) + c_2 \cdot Log_h(A''_0) = c_3$$

where c_1, c_2, c_3 are fixed so the adversary has success probability $\frac{q^2_{H'}}{q}$ for each pair of secret keys and $\frac{(n-1)(n-2)}{2} \cdot \frac{q^2_{H'}}{q}$ totally.

One may notice that unforgeability experiment was not defined. This is because tag-linkability and exculpability implies unforgeability. Let an unforgeability adversary $\mathcal{A}^{Sign_{sk_i}}$ with oracle access to sign. Firstly, for $L = (tag, pk_{[n]})$ she gets from the oracle n valid signatures on L: $[n] = (\{(L, m^{(1)}, \sigma^{(1)}), ..., (L, m^{(n)}, \sigma^{(n)})\})$. Afterwards, she forges a new signature $(L, m^{(n+1)}, \sigma^{(n+1)})$. If $\mathcal{A}$ succeeds then either $(L, m^{(n+1)}, \sigma^{(n+1)})$ is not traced with another one, which contradicts tag-linkability, or is traced with another one say $(L, m^{(i)}, \sigma^{(i)})$, which contradicts exculpability, because $\mathcal{A}$ managed to entrap i. That's why in traceable ring signatures we don't have to explicity deal with unforgeability.

Theorem 5.1. *If a traceable ring signature scheme is tag-linkable and exculpable then it is unforgeable.*

5.2 An extension of Fujisaki-Suzuki TRS

5.2.1 Dynamic traceable ring signatures(DTRS)

In the above scheme we observe that an event consists of a tag and a ring of individuals $L = (tag, pk_{[n]})$. So, a signature is traced in respect with these two specific parameters. However, an original characteristic of ring signatures is the ad-hoc formation of the group of individuals, named ring. In Fujisaki-Suzuki TRS presented above any dynamic extension of the initial ring, even addition of one user only, ruins the traceability.

Say, for example, we have an initial $L = (tag, \{pk_1, ..., pk_n\})$ and all users have signed. Now, say that afterwards we want to add a new user with public key pk_{n+1}. The new parameters are $L' = (tag, \{pk_1, ..., pk_n, pk_{n+1}\})$, which of course leads to another $h' = H(L')$. So, every signer of the initial ring can sign again because $\sigma_i' = (h')^{sk_i} = H(L')^{sk_i} \neq H(L)^{sk_i} = \sigma_i$ for each $i \in [n]$ (except for a negligible collision). That means that traceability is lost, and even worse initial users can contribute to the event with two signatures and the later user pk_{n+1} only with one.

On account of this, we propose a slight variation of the original scheme, which we believe allows ad-hoc formation of rings. The idea is simple: we remove the ring $pk_{[n]}$ from the random oracles H and H' inputs, but it remains in the H'' input. So now we have $h = H(tag)$ and $A_0 = H'(tag, m)$ and $c = H''(L, A_0, A_1, \cdot, \cdot)$. Let n be the number of initial members and $n + k$ the number of the final ones, thus k members are added. What we achieve with the previous slight modification is that if a signer signs after the addition of the k new members then the first n parts of her signature $\sigma_1, ..., \sigma_n$ will be the same as if she has signed before the addition. With this we achieve dynamic traceability. The ring is included in the signature but let it remain in H''. That is to confirm that one signed with respect to a signature. We present the full scheme in Fig.5.3.

Another change in the protocol is that if two signatures have two or more but not n same parts we regard this linked, instead of independent. In fact, in the original protocol two independent signatures have $2 \leq \#\{i|\sigma_i = \sigma_i'\} \leq n - 1$ only with negligible probability, but is included in the $Trace$ algorithm so as the scheme has statistical correctness.

In our variation two signatures may be linked not only if exactly n parts are the same. For example let an initial ring with n_1 members. Then consecutive addition of n_2 after a period n_3 and finally n_4 members. One from the second group may sign instantly $\sigma_{[n_1+n_2]}$ (when ring had n_1+n_2 members) and again with the message in the end $\sigma'_{[n_1+n_2+n_3+n_4]}$ (when ring had $n_1+n_2+n_3+n_4$ members). Then these signatures will have $n_1 + n_2$ same parts. So two signatures in the above example which are linked may have n_1, $n_1 + n_2$, $n_1 + n_2 + n_3$ or $n_1 + n_2 + n_3 + n_4$ same parts.

One choice would be to keep the critical numbers($n_1, n_2, ...$ and so on) to detect linkability. Although it is cryptographically sound, in practice it may lead to mistakes, as $Trace$ algorithm may be executed by anyone and one loss of a critical value can lead to misinterpretations. To avoid this we just say that two signatures are independent only when 0 parts are the same, traced when exactly 1 part is the same and linked for all the other values. This leaves an error to the system when two independent signatures happen to have 2 for example same parts, however this happens with negligible probability. So the scheme now has a negligible probabilty to fail.

Finally, as the scheme is intended to be used to construct an ad-hoc survey submission scheme, we pay attention to concurrent executions scenarios. For that we propose that the NIZK is generated using the Pass transformation that we saw in section 4.2, instead of Fiat-Shamir. With this we get a oSE NIZK, suited for the needs of ad-hoc Surveys. We mention, though, that Fischlin transformation [68] would, also, fit our needs.

Dynamic Traceable Ring Signature (DTRS)

Let G be a multiplicative group of prime order q with generator $g \in G$ and $H : \{0,1\}^* \to G$, $H' : \{0,1\}^* \to G$, $H'' : \{0,1\}^* \to \mathbb{Z}_q$ three random oracles.
Say that the current state of the ring includes n users $L = (tag, pk_{[n]})$ (in the past maybe there were less and some were later added).

- $Gen(1^\lambda)$: is executed by player i.
  Pick a secret key $sk_i \leftarrow \mathbb{Z}_q$ randomly and compute the public key $pk_i = g^{x_i}$.
  We assume a **PKI** and that i registers her public key pk_i.

- $Sign_{sk_i}(L, m)$: is executed by player i.

  1. Compute $h = H(tag)$ and $\sigma_i = h^{sk_i}$
  2. Compute $A_0 = H'(tag, m)$, $A_1 = \left(\frac{\sigma_i}{A_0}\right)^{1/i}$ and for each $j \neq i$ compute $\sigma_j = A_0 A_1^j$.
     $\sigma_{[n]} = (\sigma_1, ..., \sigma_n) = (A_0^{\frac{i-1}{i}} h^{sk_i \frac{1}{i}}, A_0^{\frac{i-2}{i}} h^{sk_i \frac{2}{i}}, ..., h^{sk_i}, ..., A_0^{\frac{i-n}{i}} h^{sk_i \frac{n}{i}})$
  3. For the Language $\mathcal{L} = \{(L, h, \{\sigma_1, ..., \sigma_n\}) | \exists sk_i$ s.t. $pk_i = g^{sk_i}$ and $\sigma_i = h^{sk_i}\}$ compute an oSE NIZK π

  output: The signature is $\sigma = (A_1, \pi)$

- $Verify_L(m, \sigma)$: executed by anyone

  1. Compute $h = H(tag)$, $A_0 = H'(tag, m)$, $\sigma_j = A_0 A_1^j$ for each $i \in [n]$
  2. Check if the proof is valid $Ver(\pi) = 1$

  output: 1 if the last equation holds or 0 otherwise.

- $Trace_L(m, \sigma, m', \sigma')$: where $\sigma = (A_1, c_{[n]}, z_{[n]})$ and $\sigma' = (A_1', c_{[n]}', z_{[n]}')$

  1. Compute $h = H(tag)$, $A_0 = H'(tag, m)$, $\sigma_j = A_0 A_1^j$ for each $i \in [n]$
     and $A_0' = H'(tag, m')$, $\sigma_j' = A_0' A_1'^j$ for each $i \in [n]$
  2. Compare σ_i and σ_i' for each $i \in [n]$

$$output = \begin{cases} pk_i & \text{if } \sigma_i = \sigma_i' \text{ and } \sigma_j \neq \sigma_j' \text{ for all } j \neq i \qquad \textbf{(1 exactly equal)} \\ \text{linked} & \text{if } \sigma_i = \sigma_i' \text{ for all } i \in [n] \qquad (1 < \textbf{equal)} \\ \text{indep} & otherwise \qquad \textbf{(0 exactly equal)} \end{cases}$$

Figure 5.3: A variant of Fujisaki-Suzuki Traceable Ring Signature

where the zero-knowledge proof is:

Zero-Knowledge proof π for

$\mathcal{L} = \{(L, h, \{\sigma_1, ..., \sigma_n\}) | \exists sk_i$ s.t. $pk_i = g^{sk_i}$ and $\sigma_i = h^{sk_i}\}$

take the Σ-protocol:

- ($\mathcal{P} \rightarrow \mathcal{V}$) Pick randomly $w_i, \{c_j, z_j\}_{j \neq i} \leftarrow \mathbb{Z}_q$ and compute the commit:
  $a_i = g^{w_i}, a_j = g^{z_j} pk_i^{c_j}$ for all $j \neq i$
  $b_i = h^{w_i}, b_j = h^{z_j} \sigma_i^{c_j}$ for all $j \neq i$
- ($\mathcal{V} \rightarrow \mathcal{P}$)Compute the challenge:
  $c \leftarrow G$
  $c_i = c - \sum_{j \neq i} c_j \ (mod\ q)$
- ($\mathcal{P} \rightarrow \mathcal{V}$) Compute the response:
  $z_i = w_i - c_i sk_i$

And transform it to oSE NIZK Proof of knowledge with a Pass transformation, using a new random oracle H''' to construct the online extractable commitment scheme needed.
In the final step the challenge is computed using $H''(L, A_0, A_1, \cdot, \cdot)$. So the ring is included.

Figure 5.4: A variant of Fujisaki-Suzuki Traceable Ring Signature

5.2.2 Security

We redefine the notions of Tag-Linkability, Anonymity and Exculpability to fit in the new dynamic model. We, only, make small changes to the original definitions. For the security proofs we follow the ones in the original scheme [11] and alter them whenever it is necessary.

For the proofs we will need the following lemma:

Lemma 5.2. *Suppose that an adversary $\mathcal{A}$ outputs (L, m, σ) and is verified then:*
$Pr[\#\{i \in [n] | Log_h(\sigma_i) = Log_g(y_i)\} = 0] = negl(n)$

Proof. It comes directly from the (online) extraction property of the zero knowledge proof π. □

Tag-Linkability The definition is the same. The only difference is that the $n+1$ signatures may have different lengths, which implies that they were created in earlier time, when ring was smaller.

Tag-Linkability Experiment

1. $\big(L, \{(m^{(1)}, \sigma^{(1)}), ..., (m^{(n+1)}, \sigma^{(n+1)})\}\big) \leftarrow \mathcal{A}(1^\lambda)$:
   Adversary controls all the users of the ring, creates all the public and secret keys and the ring of n users $pk_{[n]} = \{pk_1, ..., pk_n\}$
2. **output** 1 if:
   - $Verify_L(m^{(i)}, \sigma^{(i)}) = 1$ for all $i \in [n+1]$
   - $Trace_L(m^{(i)}, \sigma^{(i)}, m^{(j)}, \sigma^{(j)}) =$ indep for all $i, j \in [n+1]$ and $i \neq j$

Theorem 5.3. *The above scheme is tag-linkable in the random oracle model.*

Proof. Say that the adversary successfully outputs $n+1$ signatures
$\big(L, \{(m^{(1)}, \sigma^{(1)}), ..., (m^{(n+1)}, \sigma^{(n+1)})\}\big) \leftarrow \mathcal{A}(1^n)$ which are all valid and
$Trace_L(m^{(i)}, \sigma^{(i)}, m^{(j)}, \sigma^{(j)}) =$ indep for each pair.
From Lemma 5.2 each signature has a σ_i s.t. $\sigma_i = h^{sk_i}$ except with negligible probability $(n+1) \cdot negl(\lambda)$.
There are n public-secret keys and $n+1$ signatures. So from pigeonhole's principle at least 2 used the same secret key, say $\sigma^{(k)}$ and $\sigma^{(l)}$. But then $\exists i : \sigma_i^{(k)} = \sigma_i^{(l)}$.
Finally, either k, l don't have another common part and are traced either they have and are linked, which contradicts the assumption. □

Anonymity Anonymity definition is very similar to the one stated above. However we restrict the opponent's $Sign$ oracles to one per tag and not one per $L = \{tag, Ring\}$. This is obvious as in our scheme signer can put one signature per tag, independently of the Ring, which is actually its goal. So, second query for the same person would obviously betray the identity of person.

Anonymity Experiment

1. $(pk_0, sk_0), (pk_1, sk_1) \leftarrow Gen(1^n)$.
2. $b \leftarrow \{0,1\}$ is chosen randomly
3. The adversary $\mathcal{A}$ has oracle access to
   - $Sign_{sk_b}$ oracle : only with one message for each tag
   - $Sign_{sk_0}, Sign_{sk_1}$ oracles for each tag.
   - Cannot query $Sign_{sk_b}(L, m)$ and $Sign_{sk_0}(L, \tilde{m})$ or $Sign_{sk_1}(L, \tilde{m})$ for the same tag.
   - Each query has the restriction that L that contains both pk_0 and pk_1
4. $b' \leftarrow \mathcal{A}^{Sign_{sk_b}, Sign_{sk_0}, Sign_{sk_1}}(pk_0, pk_1)$

output 1 if: $b = b'$

Theorem 5.4. *The above scheme is Anonymous under the decisional Diffie-Hellman assumption in the Random Oracle Model.*

The proof is quite the same as in the original scheme so we omit it. The idea of the reduction was presented in section 5.1. For the detailed reduction see the original paper [11]. Actually, instead of L we have tag and everything else is the same in the proof.

An interesting difference is that witness extraction that is made at the end of the reduction is performed online, without rewinding, as we applied Pass transformation instead of Fiat-Shamir.

Exculpability Security definition is seemingly quite the same but what changes in essence is that adversary can add new public keys of her choice in the ring to entrap the target. We prove that the scheme is exculpable even in that case.

Exculpability Experiment

1. $pk \leftarrow \mathcal{A}(1^n)$: $\mathcal{A}$ targets a pk that is stored to the **PKI** and (pk, sk) has been generated by $Gen(1^n)$.
2. The adversary $\mathcal{A}$ has oracle access to $Sign_{sk}$ on any (L, m) as long as L contains pk.
3. $(L, m, \sigma), (L', m', \sigma') \leftarrow \mathcal{A}^{Sign_{sk}}(pk)$
   where $L = (tag, \{pk_1, ..., pk_n, \}$ and $L' = (tag, \{pk_1, ..., pk_{n+r}\})$ have the same same tag and (possibly) different Rings, $r \geq 0$

output 1 if:

- $Verify_L(m, \sigma) = 1$ and $Verify_L(m', \sigma') = 1$.
- At most one of $(m, \sigma), (m', \sigma')$ is linked with a query answer. This means that $\mathcal{A}$ forged at least one.
- $Trace_L(m, \sigma, m', \sigma') = pk$

Theorem 5.5. *The above scheme is tag-linkable under the discrete logarithm assumption in the random oracle model.*

Proof. Again the capabilities of an adversary are summed up in two cases, where the first has 2 subcases. For more details see fig.5.2. We note that cheating in the NIZK protocol as a malicious prover is another case that is not explicitly stated, though it is still negligible to happen.

However that public and secret keys that is able to choose to entrap the target user are not fixed anymore. So we reconstruct the proofs in each case.

Case 1a: Say that the adversary chose a secret key sk_k to form the NIZK proof.
If sk_k was in the previous ring $k < n$ then it is fixed so there is no difference from the original proof. The probability of success is at most $(n-1) \cdot \frac{q_{H'}}{q} = negl(\lambda)$.
If sk_k is a new secret key that was added then it was chosen by the adversary adaptively.
From hypothesis we have $\sigma_i = \sigma'_i$ (i is traced) so:

$$\begin{aligned} A'_0 A'^i_1 = \sigma_i \Rightarrow A'_0 \cdot \left(\left(\frac{\sigma_k}{A'_0} \right)^{1/k} \right)^i = \sigma_i &\Rightarrow A'_0 \cdot \frac{\sigma_k^{i/k}}{A'^{i/k}_0} = \sigma_i \\ &\Rightarrow \sigma_k^{i/k} = A'^{i/k-1}_0 \cdot \sigma_i \\ &\Rightarrow \sigma_k = A'^{1-k/i}_0 \cdot \sigma_i^{k/i} \\ &\Rightarrow h^{sk_k} = A'^{1-k/i}_0 \cdot \sigma_i^{k/i} \end{aligned}$$

where σ_i and i are fixed and adversary has only polynomial tries of $k > n$
So adversary has to find sk_k, $A'_0 = H'(tag, m')$ and k s.t. the above equation holds.
So either has to solve a DLOG problem or find a random oracle answer that fits.
Say adversary $\mathcal{A}$ makes $q_{H'}$ (polynomially bounded) oracle queries and tries $K = poly(\lambda)$

different k. Then her probability of success is at most $K \cdot \frac{q_{H'}}{q} + K \cdot Adv_{DLOG}$ which is negligible.

Case 1b: This is exactly as in the original non-dynamic scheme.
Adversary needs to find sk_i to create a valid NIZK proof. So she has to solve $\sigma_i = h^{sk_i}$ for sk_i which is the DLOG problem.

Case 2: For each forged signature adversary has a secret key say sk_k and sk_l where k, l and sk_k, sk_l are chosen adaptively. Also we assumed that adversary entrapped player i with σ' and σ'' so:

$$\begin{aligned}\sigma'_i = \sigma''_i &\Rightarrow A'_0 A'^{i}_1 = A''_0 A''^{i}_1 \\ &\Rightarrow A_0'^{\frac{i-k}{i}} \sigma_k'^{\frac{i}{k}} = A''_0 l^{\frac{i-l}{i}} \sigma_l''^{\frac{i}{l}} \\ &\Rightarrow A_0'^{\frac{i-k}{i}} \left(h^{sk_k}\right)^{\frac{i}{k}} = A''_0 l^{\frac{i-l}{i}} \left(h^{sk_k}\right)^{\frac{i}{l}}\end{aligned}$$

So the adversary must either solve a DLOG to determine sk_k, sk_l or find proper $A'_0 = H'(L, m')$ and $A''_0 = H'(L, m'')$. Again say $q_{H'}$ is the number of oracle queries and K, L the number of k, l resp. that the adversary tried to solve the equation.
The overall success probability is $K \cdot L \cdot \frac{q^2_{H'}}{q} + K \cdot L \cdot Adv_{DLOG}$ which is negligible. □

Unforgeability Finally unforgeability is a result of theorem 5.1

5.3 A proposed Ad-hoc Survey Scheme from Dynamic Traceable Ring Signatures

5.3.1 Scheme definition

We propose a new Ad-hoc Survey Scheme based on the above modified traceable ring signature scheme and any Digital Signature Scheme (it doesn't have to allow signing in parts as in Anonize). We remind that ad-hoc survey schemes consist of a tuple of algorithms.

$$(GenRA, RegUser^{RA}, RegUser^{\mathcal{U}}, GenSurvey, Authorized, SubmitSurvey, Check)$$

So the seven algorithms of the scheme are defined as follows:

System Setup Registration Authority's role is to maintain a PKI. So it is the Certificate Authority of the PKI (see section 2.7). So RA generates a public/secret key pair, with which is going to sign the public keys in the registration phase.

$$(sk_{RA}, vk_{RA}) \leftarrow \underline{GenRA(1^\lambda)}$$

User Registration Phase User is registered to the PKI.

$\underline{RegUser^{\mathcal{U}}(1^n, vk_{RA}, id)}$: chooses at random a secret key $sk_{id} \leftarrow \mathbb{Z}_q$ and sets the corresponding public key $pk_{id} = g^{sk_{id}}$. Then sets up a secure session with RA and sends pk_{id} together with an oSE NIZK PoK that knows a valid secret key. Afterwards checks if RA put a valid signature with $Ver_{vk_{RA}}$ algorithm if the Signature Scheme.

$\underline{RegUser^{\mathcal{U}}(1^n, sk_{RA}, id)}$: Signs pk_{id} with sk_{RA} and stores it to the public key list of PKI.

Survey Creation Anyone can be a SA. SA creates a pair of public/secret keys (sk_{SA}, vk_{SA}) of a digital signature scheme and a tag for the Survey.

$\underline{GenSurvey(1^n, vid, L)}$: SA takes the public keys that correspond to users of her choice from PKI and put them on a the list of participants. Finally with sk_{SA} she signs all public keys of the list.

The public keys in the list form a ring and together with the tag form the $L = (tag, \{pk_1, ..., pk_n\})$. More public keys-participants can later be added.

Survey Submission The user id checks if she is authorized to submit answer to the survey.

$\underline{Authorized(vid, vk_{vid}, id)}$: checks if she is on the participants' list and sees if the corresponding signature is valid $Ver_{vk_{SA}}(pk_{id}, \sigma_{id}) = 1$.

Then completes the Survey. Completion is a message m. Then uses the dynamic traceable ring signature scheme to sign the message.

$\underline{SubmitSurvey(1^n, L, m, sk_{id})}$: $Sub = Sign_{sk_{id}}(L, m)$

Validation Finally, anyone can check if a submission is valid by checking the validity of the corresponding traceable ring signature. Furthermore anyone can check that each public key of the list correspond to the declared id by checking the PKI and check the validity of the corresponding signature. id contains the e-mail of a user anyone can check by e-mailing id that RA didn't cheat in storing a false pk_{id} for id.

Anyone can execute $Trace$ of DTRS protocol to see if two submissions came from the same person.

$\underline{Check(L, Sub)}$: $Verify_L(m, \sigma)$

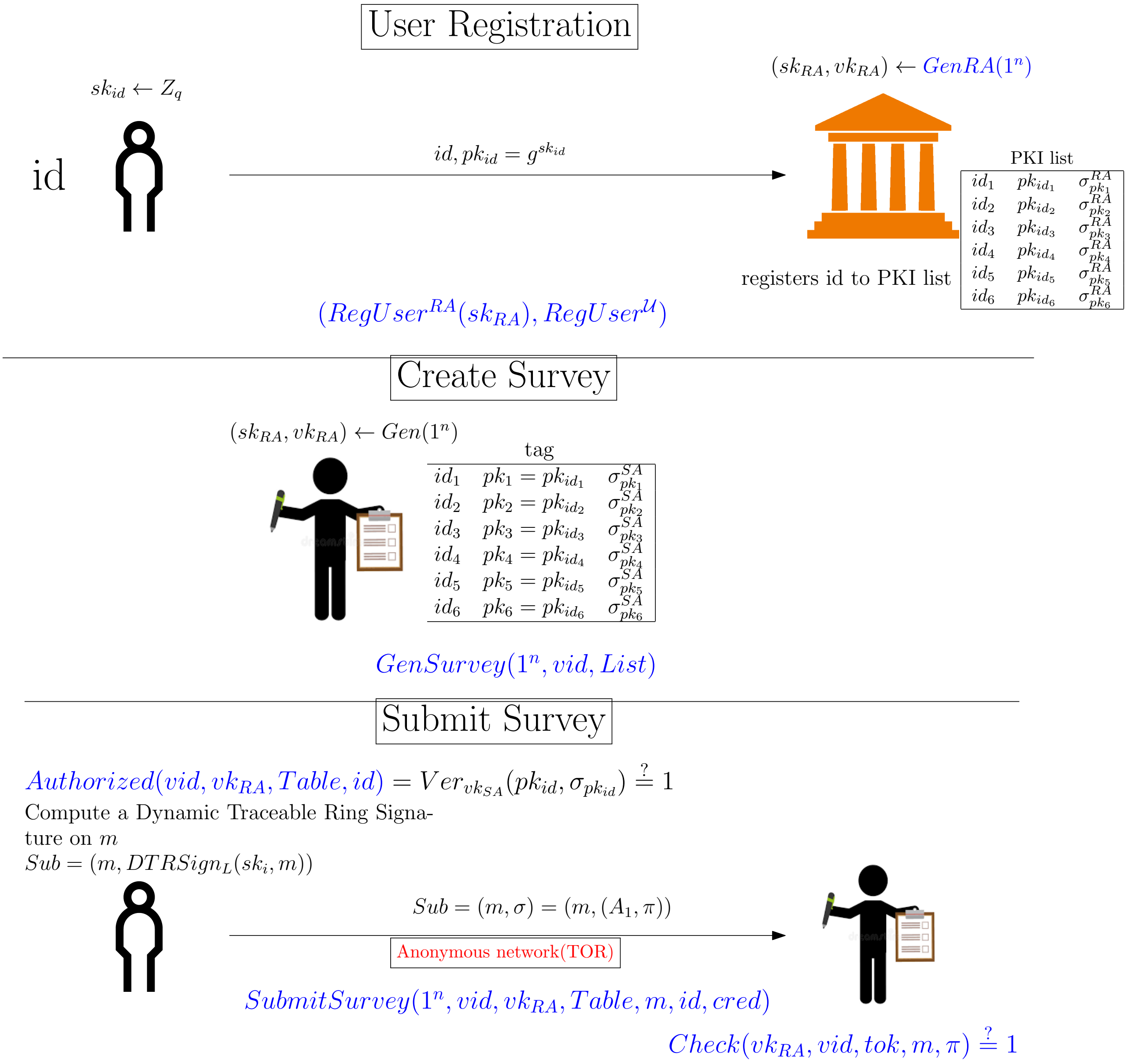
The scheme is summarized in figure 5.5.

5.3.2 Security remarks

The change in the Σ-protocol transformation, from Fiat-Shamir, which was the initial one, to Pass was made to prevent concurrent attacks that were discussed in section 4.2. Pass transformation gives us an online simulation-extractable non-interactive zero knowledge proof of knowledge, which fits our requirements.

Anonymity, comes directly from Dynamic Traceable Ring Signature Anonymity. Similarly, authenticity comes from unforgeability property and from tag-Linkability, which does not allow double submissions to remain untraced.

If we could compel pk_i peer-to-peer validation by emails we would be able to let the registration authority be malicious in the authenticity experiment, in contrast to Anonize authenticity, where she is considered honest. The reason is explained in next section.


User Registration
$(sk_{RA}, vk_{RA}) \leftarrow GenRA(1^n)$
$sk_{id} \leftarrow Z_q$
id
$id, pk_{id} = g^{sk_{id}}$
registers id to PKI list
$(RegUser^{RA}(sk_{RA}), RegUser^{\mathcal{U}})$
Create Survey
$(sk_{RA}, vk_{RA}) \leftarrow Gen(1^n)$
$GenSurvey(1^n, vid, List)$
Submit Survey
$Sub = (m, \sigma) = (m, (A_1, \pi))$
Anonymous network(TOR)
$SubmitSurvey(1^n, vid, vk_{RA}, Table, m, id, cred)$
$Check(vk_{RA}, vid, tok, m, \pi) \overset{?}{=} 1$

PKI list

id_1	pk_{id_1}	$\sigma^{RA}_{pk_1}$
id_2	pk_{id_2}	$\sigma^{RA}_{pk_2}$
id_3	pk_{id_3}	$\sigma^{RA}_{pk_3}$
id_4	pk_{id_4}	$\sigma^{RA}_{pk_4}$
id_5	pk_{id_5}	$\sigma^{RA}_{pk_5}$
id_6	pk_{id_6}	$\sigma^{RA}_{pk_6}$

tag

id_1	$pk_1 = pk_{id_1}$	$\sigma^{SA}_{pk_1}$
id_2	$pk_2 = pk_{id_2}$	$\sigma^{SA}_{pk_2}$
id_3	$pk_3 = pk_{id_3}$	$\sigma^{SA}_{pk_3}$
id_4	$pk_4 = pk_{id_4}$	$\sigma^{SA}_{pk_4}$
id_5	$pk_5 = pk_{id_5}$	$\sigma^{SA}_{pk_5}$
id_6	$pk_6 = pk_{id_6}$	$\sigma^{SA}_{pk_6}$

$Authorized(vid, vk_{RA}, Table, id) = Ver_{vk_{SA}}(pk_{id}, \sigma_{pk_{id}}) \overset{?}{=} 1$

Compute a Dynamic Traceable Ring Signature on m

$Sub = (m, DTRSign_L(sk_i, m))$

Figure 5.5: Proposed Ad-Hoc Survey Scheme from DTRS

5.3.3 Resilience to the first authenticity attack

In section 4.3.1 we underlined a possible attack that a malicious adversary could achieve in Anonize. The above ad-hoc scheme endures this attack as anyone who wishes to check a submission must use the public keys of the list $\{pk_1, ..., pk_n\}$. But public keys are public and one for each user. So a user cannot possibly have two different secret keys to form a NIZK proof. And if she does the new public key won't be on the list of participants so no NIZK proof can be verified using that.

In Anonize the problem is that the secret credential is 'lost' in zero-knowledgeness so nobody can tell which secret keys was used, The only thing that a verifier can tell is whether or not the credential comes with a valid RA signature. In our scheme public key implies use of a certain secret key which (although we don't know it) is unique for each user.

5.3.4 Efficiency of DTRS based scheme

We are free to choose any signature scheme for SA and RA signatures, in contrast to Anonize where signatures must bind with the pseudorandom function. So we may choose the most efficient and have very efficient user registration and Survey creation.

Each Submission contains a DTR signature so it has size $1 + 2n \cdot \omega(log\lambda)$. That's because $2n$ is the size of the initial Fiat Shamir proof and $\omega(log\lambda)$ is the overhead from Pass transformation. So verifying a submission would require $2n \cdot \omega(log\lambda)$ checks and verifying all n submissions $2n^2 \cdot \omega(logn)$ checks.

Furthermore to check for possible double malicious double submissions we need to execute $Trace$ algorithm $O(n^2)$ times. Each $Trace$ requires n equality checks so totally we need $O(n^3)$ equality checks.

So the overall check time would be $O(n^3) + 2n^2 \cdot \omega(log\lambda)$ time (if e.g. we choose n repetition in Pass transformation).

5.3.5 Discussion on DTRS based scheme

Now it is clear why we altered the initial TRS protocol to let the ring dynamically extend. Without this, we could't claim that the proposed ad-hoc scheme is really ad-hoc, as it does not allow additions to the initial group of users.

What we gained from the above scheme proposal is first of all an alternative ad-hoc survey scheme! But the goal was to defend from section 4.3.1 authenticity attack as discussed previously, which we achieved. We are relieved from considering RA honest (as long as email verification of pk_i is possible for each id).

As for efficiency, admittedly the scheme is worse than Anonize. That's because in Anonize the proof is constant sized without Pass transformation overhead. So in Anonize verification is done in $n \cdot \omega(log\lambda)$ time and double-vote checking in $O(n^2)$. However, we believe that freedom in signature scheme choice in registration and survey creation phases will lead to more efficient phases, strictly for the first two phases.

5.4 Short Linkable Ring Signatures

In an effort to improve efficiency of the above proposed scheme, we propose a new ad-hoc survey scheme based on Short Linkable Ring Signatures (SLRS). As discussed the basic source of inefficiency in the DTRS-based ad-hoc scheme is the linear size of the ZK proof for each signature. Short Linkable Ring Signatures give signatures of constant size.

The Scheme that we utilize comes from [5], a work of Au et al., who extended a prior work of Tsang and Wei [12]. Basically, they use an accumulator to 'compress' a ring of n members to just a value. For each value there is a witness of correct accumulation. This lets us construct small ZK proofs that instead of n equalities prove one equality.

5.4.1 Accumulators with one-way domain

Accumulators with one-way domain were introduced by Dodis, Kiayias, Nicolosi and Shoup in [79]. In their work they used them to construct a short Ring Signature Scheme.

Accumulator An Accumulator family is a pair of $(\{F_\lambda\}, \{X_\lambda\})$, where F_λ is a family of function s.t. each $f \in F_\lambda$ is defined $f : U_f \times X_\lambda \to U_f$ and X_λ is the value domain. Additionally the following should hold:

1. (efficient generation) There exists an efficient algorithm $G(1^\lambda)$ that outputs an $f \in F_\lambda$.

2. (efficient evaluation) Any $f \in F_\lambda$ is computed in $poly(\lambda)$ time.

3. (quasi-commutativity) For all $\lambda \in \mathbb{N}$, $f \in F_\lambda$, $u \in U_f$, $x_1, x_2 \in X_\lambda$

$$f(f(u, x_1), x_2) = f(f(u, x_2), x_1)$$

Quasi-cummutativity is the most important property because it allows us to apply the accumulator to a group of x_i's and, independently of the sequence of computations that we chose, the result will be the same. For this we adopt the donation $f(u, \{x_1, ..., x_n\}) \hat{=} f(...f(u, x_1)..., x_n)$. This property will allow us to map a ring of public keys to a single value, and have a consensus of the value, while hiding the sequence.

Furthermore a desired security property that we want an accumulator to possess is collision resistance. That is given a $u \in U_f$ no PPT adversary can find w, x, X s.t. $f(w, x) = f(u, X)$, where $X = \{x_1, ..., x_s\}$ for any s. If an accumulator is collision resistant then we can claim that w is a witness that x has been accumulates within v if $f(w, x) = v$. That's because collision resistance indicates that it is hard to find another witness.

Accummulators with one way domain An Accumulator with one-way domain is a quadruple $(\{F_\lambda\}, \{X_\lambda\}, \{Z_\lambda\}, \{R_\lambda\})$ such that $(\{F_\lambda\}, \{X_\lambda\})$ is a collision-resistant accumulator, R_λ is a relation over $X_\lambda \times Z_\lambda$ and the following hold:

1. (efficient verification) There exists an efficient algorithm D such that $D(x, z) = 1$ iff $(x, z) \in R_\lambda$ and 0 otherwise for $x \in X_\lambda$, $z \in Z_\lambda$

2. (efficient sampling) There exists a PPT algorithm W s.t. $W(1^\lambda)$ outputs $(x, z) \in R_\lambda$

3. (one wayness) It is computationally hard to compute a preimage z' of x that was sampled with W:

$$Pr[(x, z) \leftarrow W(1^\lambda); z' \leftarrow \mathcal{A}(1^\lambda, x) : (x, z') \in R_\lambda] = negl(\lambda) \qquad \text{for all PPT } \mathcal{A}$$

The intuition behind this extension of initial accumulators is to enable secret/public key generation and validation with accumulators. So R_λ is the relation between the secret and the public key, W is the generator algorithm and one-wayness preserves the secret key's secrecy.

An instantiation The implementation for Accumulators with one-way domain presented in [79] is the below:
$(\{F_\lambda\}, \{X_\lambda\}, \{Z_\lambda\}, \{R_\lambda\})$ where

- $F_\lambda = \{f\}$ where $f : (\mathbb{Z}_n^*)^2 \times \mathbb{Z}_{n/4} \to (\mathbb{Z}_n^*)^2, \qquad f(u, x) = u^x \text{ (mod } n)$
- $X_\lambda = \{e \text{ prime} | \left(\frac{e-1}{2} \in RSA_\ell\right) \wedge |e - 2^\ell| < 2^\mu\}$, where $\lambda - 2 > \ell$
- $Z_\lambda = \{(e_1, e_2) | e_1, e_2$ are distinct $\ell/2$-bit primes $\wedge\ |e_2 - 2^{l/2}| < 2^\mu\}$
- $R_\lambda = \{(x, (e_1, e_2)) \in X_\lambda \times Z_\lambda | x = 2e_1e_2 + 1\}$

RSA integer is an $n = pq$ with $|p| = |q|$ and p, q primes. A prime p is called safe prime if $p = 2p' + 1$ where p' is prime. Collision-resistance is based on strong RSA assumption. Finally, one-wayness holds assuming hardness of factoring RSA integers.

Definition 5.6. *(Strong RSA Assumption) No PPT algorithm given a random λ-bit safe prime product N and a random $z \in (\mathbb{Z}_n^*)^2$ can output $u \in \mathbb{Z}_n^*$, $e \in \mathbb{N}$ s.t. $u^e = z$ (mod N)*

5.4.2 Short Linkable Ring Signatures

The SLRS scheme of [5] uses signatures of knowledge (SPK) for the $Sign$ algorithm. These are basically Σ-protocols, that prove knowledge of a secret key, that are transformed using Fiat-Shamir heuristic. But to create the challenge we concatenate the original input of random oracle with the message that is signed, $c = RO(a, m)$ instead if $c = RO(a)$. We present the construction in steps.

First approach Say we have a PKI with registered public keys that correspond to secret keys that are connected with a key relation $(sk, pk) \in \mathcal{R}$. Say we have a ring of keys $pk_{[n]} = \{pk_1, ..., pk_n\}$. A first approach to construct ring signatures would be:

$$SPK\{sk : (sk, pk_1) \in \mathcal{R} \vee (sk, pk_2) \in \mathcal{R} \vee\ ...\ \vee (sk, pk_n) \in \mathcal{R}\}(m)$$

But that signature would have linear size $O(n)$.

Short Ring Signature Here is where accumulators with one-way domain are used. They come to 'compress' the n above equalities to a single one. The key point is that accumulators provide a witness as well, that is used to construct a ZK proof.
Say $f(u, \{pk_1, ..., pk_n\}) = v$ and $f(u, \{pk_1, ..., pk_n\} \setminus \{pk_i\}) = w$.
From quasi-commutativity we have

$$f(w, pk_i) = f(f(u, pk_{[n]} \setminus \{pk_i\}), pk_i) = f(u, pk_{[n]}) = v$$

And w is a witness, v is public. So the signature of user i will be:

$$SPK\{(w, sk_i, pk_i) : (sk_i, pk_i) \in \mathcal{R} \wedge f(w, pk_i) = v\}(m)$$

This is roughly the construction of [79]. Now it is clear why quasi-commutativity is important. Furthermore, collision resistance of the accumulator ensures us that there is a witness for a real public key and one-wayness preserves the secret key.

The above construction is a short ring signature as it has constant size.

Short Linkable Ring Signature To add linkability in the scheme Tsang and Wei [12] added a new PK-bijective mapping:

Definition 5.7. *(PK-bijectivity) Let $\mathcal{R} \subseteq Z \times X$ be a one-way samplable NP-relation. A mapping $\theta : Z \to X$ is PK-bijective if it satisfies:*

1) *The mapping is one-way and bijective*

2) *Let (x_0, y_0) and (x_1, y_1) be two random samples of $\mathcal{R}$ with $y_0 \neq y_1$. Let $b \in \{0, 1\}$ be a fair coin and $z = \theta(x_b)$. For each PPT algorithm that takes z as input the probability to distinguish if $b = 0$ or $b = 1$ is negligibly close to $1/2$.*

Furthermore it is special PK-bijective if it also satisfies:

3) *Same as (2) but (x_0, y_0) and (x_1, y_1) are not fixed. It holds for any (x_0, y_0),(x_1, y_1).*

A signer will use the (special) PK-bijective mapping to create a unique token that ensures linkability $tok = \theta(sk)$. Property (2) ensures that her identity is not betrayed. Each time that she uses θ for the same parameters the same token is going to appear, which makes the signature linkable. Furthermore, she wil add to the ZK proof a part that proves honest computation of tok So the signature now is:

$$SPK\{(w, sk_i, pk_i) : (sk_i, pk_i) \in \mathcal{R} \wedge f(w, pk_i) = v \wedge tok = \theta(sk)\}(m)$$

The final instantiation of Short Linkable Ring Signatures uses the accumulator of section 5.4.1 and $\theta(e_1, e_2) = \tilde{g}^{e_1+e_2}$, where $\tilde{g}$ is a quadratic residue mod N.

Short Linkable Ring Signature scheme

- $Init(1^{\lambda})$: chooses the parameters of the accumulator $desc$ and $\tilde{g} \in QR(N)$. These are $param$.
- $Key - Gen(1^{\lambda}, desc)$: Player i executes the sampling algorithm of the accumulator W to obtain $(sk_i, pk_i) = ((e_1, e_2), (2e_1e_2 + 1))$.
  Then computes a ZK proof $PoK\{sk_i : (pk_i, sk_i) \in \mathcal{R}\}$ and sends pk_i, π to the CA of the PKI.
  If CA verifies the proof sends a certificate for pk_i and stores it to the list.
- $Sign_{sk_i}(Ring, param, m)$: is executed by player i.
  1. Compute $w = f(u, pk_{[n]} \setminus \{pk_i\}) = u^{pk_1 \dots pk_{i-1} pk_{i+1} \dots pk_n}$ and $v = f(u, pk_{[n]}) = w^{pk_i}$
  2. Compute:

$$\sigma' = SPK \left\{ \begin{matrix} (w, (e_1, e_2), pk_i) : pk_i = 2e_1e_2 + 1 \ \wedge \ |pk_i - w^{\ell}| < 2^{\mu} \ \wedge \\ \wedge \ |e_1 - 2^{\ell/2}| < 2^{\mu} \ \wedge \ w^{pk_i} = v \wedge \\ \wedge \ tok = \tilde{g}^{e_1 + e_2} \end{matrix} \right\} (m)$$

  output: The signature is $\sigma = (v, tok, \sigma')$
- $Verify_{pk_{[n]}, \tilde{g}}(m, \sigma)$: Checks if $v = u^{pk_1 \dots, pk_n}$ and if the proof σ' is valid for m.
- $Link_L(m_1, \sigma_1, m_2, \sigma_2)$: are linked iff $tok_1 = tok_2$

Figure 5.6: Short Linkable Ring Signature scheme as was constructed in [5]

5.4.3 Security of SLRS

The security properties that the above scheme has are Unforgeability, Anonymity, Linkability and Non-Slanderability. We briefly discuss these. For more details we refer to the original paper [5].

The assumptions that we are gonna make are DDH, Strong RSA (SRSA) and Link Decisional RSA (LD-RSA) assumption. With $QR(N)$ we notate the set of quadratic residues modulo N.

Definition 5.8. *(Link Decisional RSA Assumption) Let an λ-bit safe prime product N, a generator g of $QR(N)$, $n_0 = p_0 q_0$ and $n_1 = p_1 q_1$, where p_0, q_0, p_1, q_1 are primes of $poly(\lambda)$-size. No PPT algorithm given N, g, n_0, n_1 and $g^{p_b + q_b}$, where $b \in \{0, 1\}$ random coin, can output $b' = b$ with probability non-negligibly close to* $1/2$.

Unforgeability An external adversary is given a list of public keys $Ring$. Then she makes signing queries freely and additionally gets secret keys of some users of her choice in the ring. Yet she fails to forge a new ring signature (as long as she did not get it form the oracle or it is not a signature from a secret key she obtained).

Theorem 5.9. *Given that DDH assumption for $QR(N)$, LD-RSA assumption and Strong RSA assumption hold the above SLRS scheme is unforgeable in the random oracle model.*

Anonymity For Anonymity we consider an experiment where the adversary $\mathcal{A}$ receives a Ring of public keys and has access to signing oracle $Sign_{sk}(\cdot, \cdot, \cdot)$ for any sk and for any $Ring$ as long as long as it contains sk. Also $\mathcal{A}$ can demand secret keys of some users and she will get them. Then she chooses two indices i_0, i_1 and a random bit b is generated. $\mathcal{A}$ receives $\sigma = Sign_{sk_{i_b}}(Ring, param, m)$. Afterwards, she accesses the $Sign$ oracle again and demand secret keys again, exactly as before. Finally adversary outputs a bit b'. $\mathcal{A}$ wins if $b = b'$ and she never queried $Sign_{sk_{i_0}}$ or $Sign_{sk_{i_1}}$ for the same parameters or asked the secret keys sk_{i_0}, sk_{i_1}.

The short linkable ring signature scheme is anonymous if any PPT adversary wins the game with a probability negligibly close to $1/2$.

Theorem 5.10. *Given that DDH assumption for $QR(N)$ and LD-RSA assumption hold the above SLRS scheme is Anonymous in the random oracle model.*

Linkability An external adversary is given a list of public keys $Ring$. Then she makes signing queries freely and additionally gets secret keys of some users of her choice in the ring, who are from now on considered corrupted. Then she outputs two signatures on (possibly) different rings $(m_1, \sigma_1, Ring_1)$ and $(m_2, \sigma_2, Ring_2)$. Adversary wins if these signatures are both verified and are not linked and
$\#\{\text{corrupted users } \in Ring_1 \cup Ring_2\} + \#\{\text{added } pk\text{'s to } Ring\} \leq 1$.

Theorem 5.11. *Given that DDH assumption for $QR(N)$, LD-RSA assumption and Strong RSA assumption hold the above SLRS scheme is Linkable in the random oracle model.*

Non-Slanderability This is similar property with Exculpability of Traceable Ring Signatures. What it states is that no adversary can entrap a player who signed by providing a signature that is linked to the one that the player provided.

An external adversary is given a list of public keys $Ring$. Then she makes signing queries freely and additionally gets secret keys of some users of her choice in the ring, who are from now on considered corrupted. Then she outputs two signatures where the first came from the oracle and the second was not (it is forged). Adversary wins the game if both signatures are verified and are linked.

Theorem 5.12. *Given that DDH assumption for $QR(N)$, LD-RSA assumption and Strong RSA assumption hold the above SLRS scheme is Non-slanderable in the random oracle model.*

5.5 A proposed Ad-hoc Survey Scheme from Short Linkable Ring Signatures

5.5.1 Scheme Definition

Now we propose a new Ad-hoc Survey Scheme based Short Linkable Ring Signatures. As said before, the main motivation was to improve efficiency of the DTRS-based ad-hoc survey scheme of section 5.3. The only change that we make is that wherever there is a zero knowledge proof using Fiat-Shamir we replace it with Pass transformation from section 4.2. So the signature and the proof on key generation phase are Σ-protocols transformed to oSE NIZK proofs with a Pass transformation.

The general scheme is quite similar with the one proposed before but instead of linear size DTR signatures it utilizes constant size SLR signatures. So basically only the submission phase is different.

We define again the seven algorithms of the scheme:

System Setup Registration Authority's role is to maintain a PKI. So it is the Certificate Authority of the PKI . So RA generates a public/secret key pair, with which is going to sign the public keys in the registration phase.

$$(sk_{RA}, vk_{RA}) \leftarrow \underline{GenRA(1^{\lambda})}$$

Also chooses the accumulator parameters $desc$ and publishes them.

User Registration Phase User is registered to the PKI.

$\underline{RegUser^{\mathcal{U}}(1^n, vk_{RA}, id, desc)}$: chooses at random a secret-public key pairs by executing $Key - Gen(1^{\lambda, desc})$. Then sets up a secure session with RA and sends pk_{id}, π (see fig.5.6). Afterwards checks if RA put a valid signature with $Ver_{vk_{RA}}$ algorithm of the Signature Scheme.

$\underline{RegUser^{\mathcal{U}}(1^n, sk_{RA}, id)}$: Checks if the ZK proof π is valid and if so signs pk_{id} with sk_{RA} and stores it to the public key list of PKI.

Survey Creation Anyone can be a SA. SA creates a pair of public/secret keys (sk_{SA}, vk_{SA}) of a digital signature scheme.

$\underline{GenSurvey(1^n, L)}$: SA takes the public keys that correspond to users of her choice from PKI and put them on a the list of participants. Finally with sk_{SA} she signs all public keys of the list. The public keys in the list form a ring. More public keys-participants can later be added.

Also generates a unique $\tilde{g} \in QR(N)$ for the Survey. $\tilde{g}$ is the base of θ bijective map (see section 5.4.2 and fig.5.6) characterizes the Survey.

So the output is the public key list and $\tilde{g}$.

Survey Submission The user id checks if she is authorized to submit answer to the survey.

$\underline{Authorized(vid, vk_{vid}, id)}$: checks if she is on the participants list and sees if the corresponding signature is valid $Ver_{vk_{SA}}(pk_{id}, \sigma_{id}) = 1$.

Then completes the Survey. Completion is a message m. Then uses the short linkable ring signature scheme to sign the message.

$\underline{SubmitSurvey(1^n, L, m, sk_{id})}$: $Sub = Sign_{sk_i}(Ring, param, m)$

Validation Finally anyone can check if a submission is valid by checking the validity of the corresponding linkable ring signature. Furthermore anyone can check that each public key of the list correspond to the declared id by checking the PKI and check the validity the corresponding signature. id contains the e-mail of a user anyone can check by e-mailing id that RA didn't cheat in storing a false pk_{id} for id.

Anyone can execute $Link$ of DTRS protocol to see if two submissions came from the same person.

$\underline{Check(L, Sub)}$: $Verify_L(m, \sigma)$

5.5.2 Security

For security, everything that was discussed for the DTRS based scheme on 5.3.2 is applicable to the above scheme as well. That's because traceable and linkable ring signatures are very similar primitives in general. Furthermore, if one looks closely the security properties of two schemes she will notice that they have the same impact on anonymity and authenticity of the corresponding ad-hoc survey schemes.

5.5.3 Efficiency of SLRS based scheme

Again freedom of choice of signature scheme for RA and SA signatures allows us to have a very efficient user registration and survey creation phase.

Now each submission has the size of the corresponding ZK proof of the signature. Normally the ZK proof would have constant size but our choice of oSE NIZK give us an $\omega(log\lambda)$ overhead. For example λ-size. This is the same size as Anonize. Furthermore in the submission each user may first check if all public keys of the ring have valid certificate from CA, which would require $O(n)$ time (if signature verification for each user is constant).

The validation procedure at the end requires $O(n^2)$ runs of $Link$ algorithm, which takes constant time (checks just an equality). Furthermore, verifications of all submissions would require $n \cdot \omega(log\lambda)$.

5.5.4 Discussion on SLRS based scheme

We claim that this scheme, also, defends against 4.3.1 authenticity attack, as again to verify a signature one must feed all public keys of the ring to the algorithm. So as each public key corresponds to one secret key there cannot be malicious submissions from people outside the ring (even if the collude with RA)

5.6 An idea for receipt-freeness with constraints

5.6.1 The idea

In section 4.3.2 we talked about necessity of receipt-freeness and vulnerability of Anonize to vote-buying. Afterwards, we presented a motivation from real world. Now, we give an abstract idea on how to overcome this problem considering, though, constrains similar to the specific application.

What is interesting in 4.3.2 scenario is that Brave browser, i.e. the Survey Collector is the one insulted from vote-selling. We described that vote-seller (user) and, even more, vote-buyer (malicious website) gain from lack of receipt-freeness but Brave actually loses, as the designed system does not function as intended.

To make it more general we are interested in situations where Survey-Collector (it does not have to be the same as survey initiator) is honest for receipt-freeness. But, it may still be malicious regarding authenticity and anonymity. That is, we introduce an authority called Survey-Collector (SC) that has to be honest regarding receipt-freeness. However, in Authenticity and Anonymity security Experiments, SC is considered (possibly) malicious. Survey Authority is responsible for the initiation of the Survey.

Furthermore, we consider simple Surveys that have relatively small number of questions that can be answers with a number and that the result is deduced with simple linear operations.

The idea is this:

1. User sends the answer m to the Survey Collector using Anonize $Sub = (tok, m, \pi)$.
2. SC commits to the message m, $c \leftarrow Com(m)$ and calculates an designated verifier oSE ZK proof π' that $Com(m) = c$.
3. User uses Anonize again to but now to submit the ciphertext c' as answer $Sub = (tok, c', \pi'')$
4. If tok is the same and c' is not $c = Com(m)$ then SC rejects. Otherwise accepts and publishes the Submission.

At a point when the Survey Collector wants to calculate the result all answers are added homomorphically and then the result is opened:

$$Res = Open(c_1 \cdot ... \cdot c_n) = Open(Com(m_1) \cdot ... \cdot Com(m_n)) = Dec(Com(m_1 + ... + m_n)) = \sum_{i=1}^{n} m_i$$

We note that the commitment scheme should be additively homomorphic and there should exist ZK proof of correct commitment. For example Pedersen commitment scheme is additively homomorphic. Also, say $c = g^m h^r$ then Survey collector can provide a simple Schnorr [26] PoK of exponent π' that ZK$\{r : \frac{c}{g^m} = h^r\}$.

Finally, the proof π' should be designated verifier, as user should not be able to use it to prove that $c = Com(m)$ to a vote-buyer. One suggestion would be Pass' deniable Concurrent zero knowledge proof of knowledge [20], though other designated verifier ZK proofs would also work, as long as they can be transformed to oSE ZK proofs.

Receipt-freeness As long as SC does not reveal the initial submission $Sub = (tok, m, \pi)$ no user can sell her vote. This is because even if she provide proof for her submission, from hiding property of commitment scheme the vote-buyer cannot gain any information from c about m. And the only published submissions are on commitments. Furthermore, the designated verifier proof cannot convince the vote-buyer that $\frac{c}{g^m} = h^r$.

On the other hand, the use of Anonize preserves Anonymity and Authenticity even from SC, so there is no need to be honest regarding these properties.

5.6.2 Discussion

The above partially solves the lack of receipt-freeness of Anonize for simple Surveys. We mention that instead of Anonize, any ad-hoc survey Scheme can be chosen. Thus, our proposed systems from ring signatures presented in previous chapters may be utilized. Then, both the attack of section 4.3.1 is deterred and receipt-freeness (under constraints) is added.

Chapter 6

Conclusion

6.1 Summary

To sum up, in this thesis we studied the problem of Anonymous data Collection from cryptographic perspective. We studied in depth Anonize, a concrete existing scheme and the corresponding generic cryptographic primitive defined Anonymous Ad-Hoc Surveys. We made some observation on Anonize's security strength and afterwards proposed two new Anonymous Ad-Hoc Survey Schemes, both based on ring signature variants.

After presenting Traceable Ring Signatures we proposed an extension, which we called Dynamic Traceable Ring Signatures, that allows dynamic formation of groups. Furthermore, we proved that security properties of Traceable Ring Signatures hold in our proposed new primitive. Finally, we proposed an Ad-hoc Survey Scheme which we claim that strengthens the security of the existing. In this scheme the size of each Submission is linear in participants' number.

To improve Submission's size we presented Short Linkable Ring Signatures and base a new scheme on them. This scheme has constant size submissions as Anonize and we, further, believe that offers a security strengthening.

Finally, we presented an idea of how to perform receipt-free data Collection, under constraints, with Ad-Hoc Survey Schemes.

6.2 Future Work

There are many directions of future study and work:

1. **Receipt-Free Ad-hoc Surveys:** We are highly interested in notion of Receipt-Freeness. Ways of adding it without considering trusted parties and for general purpose Surveys, is directions of high priority. Specifically, we are looking to adjusting receipt-free Voting System of Panagiotis Grontas, Aris Pagourtzis, Alexandros Zacharakis and Bingsheng Zhang [9] to Ad-Hoc Surveys.

2. **Utilizing Ad-hoc Survey Scheme to Voting:** Another direction is study how Ad-Hoc Schemes can be used to perform secure elections.

3. **Surveys with Submissions as ciphertexts:** We pointed out the need to submit answers as ciphertexts instead of plaintexts in section 4.3.5. We briefly gave an idea in the last section (5.6) for simple results. Though, we are interested in study more complex operations that

can be performed homomorphically over ciphertexts. This would enable us submit data as ciphertexts and add extra anonymity to the system.

4. **Accumulator for Dynamic Traceable Ring Signature:** We may examine if Accumulators can be used to reduce signature size of Dynamic Traceable Ring Signatures. With this, we would get constant size Dynamic Traceable Ring Signatures. So we seek to theoretically study if it is possible to find an appropriate Accumulator that fits our needs.

5. **Implementation:** An implementation, of course, is always desired for cryptographic schemes as much for testing as for practical use of a system. This would enable us to test in practice the efficiency of our proposed schemes.

Bibliography

[1] A. Rosen, *Concurrent Zero-Knowledge: With Additional Background by Oded Goldreich*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[2] O. Goldreich and H. Krawczyk, “On the composition of zero-knowledge proof systems,” *SIAM Journal on Computing*, vol. 25, no. 1, pp. 169–192, 1996.

[3] V. Shoup and R. Gennaro, “Securing threshold cryptosystems against chosen ciphertext attack,” in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1998, pp. 1–16.

[4] D. Bernhard, M. Fischlin, and B. Warinschi, “Adaptive proofs of knowledge in the random oracle model,” in *Public-Key Cryptography – PKC 2015*, J. Katz, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 629–649.

[5] M. H. Au, S. S. Chow, W. Susilo, and P. P. Tsang, “Short linkable ring signatures revisited,” in *European Public Key Infrastructure Workshop*. Springer, 2006, pp. 101–115.

[6] S. Hohenberger, S. Myers, R. Pass, and a. shelat, “Anonize: A large-scale anonymous survey system,” in *Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE, 2014, pp. 375–389.

[7] A. Zacharakis, P. Grontas, and A. Pagourtzis, “Conditional blind signatures,” in *7th International Conference on Algebraic Informatics (short version), 2017*. Full version available on: http://eprint.iacr.org/2017/682

[8] P. Grontas, A. Pagourtzis, and A. Zacharakis, “Coercion resistance in a practical secret voting scheme for large scale elections,” in *Proceedings of ISPAN-FCST-ISCC 2017*, 2017, pp. 514–519.

[9] P. Grontas, A. Pagourtzis, A. Zacharakis, and B. Zhang, “Towards everlasting privacy and efficient coercion resistance in remote electronic voting,” in *3rd Workshop on Advances in Secure Electronic Voting, in assocation with Financial Cryptography 2018.*

[10] S. Zachos, A. Pagourtzis, and P. Grontas, *Computational Cryptography [ebook]*. Athens:Hellenic Academic Libraries. Available Online at: http://hdl.handle.net/11419/5439, 2015.

[11] E. Fujisaki and K. Suzuki, “Traceable ring signature,” in *International Workshop on Public Key Cryptography*. Springer, 2007, pp. 181–200.

[12] P. P. Tsang and V. K. Wei, "Short linkable ring signatures for e-voting, e-cash and attestation," in *International Conference on Information Security Practice and Experience*. Springer, 2005, pp. 48–60.

[13] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of computer and system sciences*, vol. 28, no. 2, pp. 270–299, 1984.

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

[15] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proceedings of the 1st ACM conference on Computer and communications security*. ACM, 1993, pp. 62–73.

[16] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," *Journal of the ACM (JACM)*, vol. 51, no. 4, pp. 557–594, 2004.

[17] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct randolli functions," in *Foundations of Computer Science, 1984. 25th Annual Symposium on*. IEEE, 1984, pp. 464–479.

[18] D. Catalano and I. Visconti, "Hybrid commitments and their applications to zero-knowledge proof systems," *Theoretical Computer Science*, vol. 374, no. 1-3, pp. 229–260, 2007.

[19] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Annual International Cryptology Conference*. Springer, 1991, pp. 129–140.

[20] R. Pass, "On deniability in the common reference string and random oracle model," in *Annual International Cryptology Conference*. Springer, 2003, pp. 316–337.

[21] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 281–308, 1988.

[22] J. Katz and Y. Lindell, *Introduction to modern cryptography*. CRC press, 2014.

[23] D. Chaum, "Blind signatures for untraceable payments," in *Advances in cryptology*. Springer, 1983, pp. 199–203.

[24] J. L. Camenisch, J.-M. Piveteau, and M. A. Stadler, "Blind signatures based on the discrete logarithm problem," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1994, pp. 428–432.

[25] E. Mohammed, A.-E. Emarah, and K. El-Shennaway, "A blind signature scheme based on elgamal signature," in *Radio Science Conference, 2000. 17th NRSC'2000. Seventeenth National*. IEEE, 2000, pp. C25–1.

[26] C.-P. Schnorr, "Efficient identification and signatures for smart cards," in *Conference on the Theory and Application of Cryptology*. Springer, 1989, pp. 239–252.

[27] M. Abe and E. Fujisaki, "How to date blind signatures," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 1996, pp. 244–251.

[28] M. Stadler, J.-M. Piveteau, and J. Camenisch, "Fair blind signatures," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1995, pp. 209–219.

[29] A. Juels, M. Luby, and R. Ostrovsky, "Security of blind digital signatures," in *Annual International Cryptology Conference*. Springer, 1997, pp. 150–164.

[30] B. Schoenmakers, "Lecture notes cryptographic protocols," 2018.

[31] D. Pointcheval and J. Stern, "Provably secure blind signature schemes," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 1996, pp. 252–265.

[32] D. Schröder and D. Unruh, "Security of blind signatures revisited," in *International Workshop on Public Key Cryptography*. Springer, 2012, pp. 662–679.

[33] D. Schröder, "On the complexity of blind signatures," Ph.D. dissertation, Technische Universität, 2010.

[34] M. Abe and T. Okamoto, "Provably secure partially blind signatures," in *Annual International Cryptology Conference*. Springer, 2000, pp. 271–286.

[35] D. Chaum and E. Van Heyst, "Group signatures," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1991, pp. 257–265.

[36] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2001, pp. 552–565.

[37] A. Bender, J. Katz, and R. Morselli, "Ring signatures: Stronger definitions, and constructions without random oracles," in *Theory of Cryptography Conference*. Springer, 2006, pp. 60–79.

[38] 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.

[39] I. F. Blake, G. Seroussi, N. P. Smart *et al.*, *Advances in elliptic curve cryptography*. Cambridge University Press, 2005, vol. 317.

[40] A. J. Menezes, T. Okamoto, and S. A. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field," *iEEE Transactions on information Theory*, vol. 39, no. 5, pp. 1639–1646, 1993.

[41] A. Joux and K. Nguyen, "Separating decision diffie–hellman from computational diffie–hellman in cryptographic groups," *Journal of cryptology*, vol. 16, no. 4, pp. 239–247, 2003.

[42] A. Joux, "A one round protocol for tripartite diffie–hellman," in *International algorithmic number theory symposium*. Springer, 2000, pp. 385–393.

[43] D. Boneh and X. Boyen, "Efficient selective-id secure identity-based encryption without random oracles," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 223–238.

[44] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Annual international cryptology conference*. Springer, 2001, pp. 213–229.

[45] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Workshop on the theory and application of cryptographic techniques*. Springer, 1984, pp. 47–53.

[46] C. Cocks, "An identity based encryption scheme based on quadratic residues," in *IMA International Conference on Cryptography and Coding*. Springer, 2001, pp. 360–363.

[47] 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.

[48] O. Goldreich, *Foundations of Cryptography: Volume 1*. New York, NY, USA: Cambridge University Press, 2006.

[49] O. Goldreich and Y. Oren, "Definitions and properties of zero-knowledge proof systems," *Journal of Cryptology*, vol. 7, no. 1, pp. 1–32, 1994.

[50] D. Boneh and V. Shoup, "A graduate course in applied cryptography," *Version 0.1, from http://cryptobook. net*, 2008.

[51] 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.

[52] M. Bellare and O. Goldreich, "On defining proofs of knowledge," in *Annual International Cryptology Conference*. Springer, 1992, pp. 390–420.

[53] C. Hazay and Y. Lindell, *Efficient secure two-party protocols: Techniques and constructions*. Springer Science & Business Media, 2010.

[54] I. Damgård, "Efficient concurrent zero-knowledge in the auxiliary string model," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2000, pp. 418–430.

[55] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge and its applications," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*. ACM, 1988, pp. 103–112.

[56] 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*. Springer, 1986, pp. 186–194.

[57] C. Dwork, M. Naor, and A. Sahai, “Concurrent zero-knowledge,” in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, ser. STOC ’98. New York, NY, USA: ACM, 1998, pp. 409–418. [Online]. Available: http://doi.acm.org/10.1145/276698.276853

[58] J. Kilian, E. Petrank, and C. Rackoff, “Lower bounds for zero knowledge on the internet,” in *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*. IEEE, 1998, pp. 484–492.

[59] R. Richardson and J. Kilian, “On the concurrent composition of zero-knowledge proofs,” in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1999, pp. 415–431.

[60] A. Rosen, “A note on the round-complexity of concurrent zero-knowledge,” in *Annual International Cryptology Conference*. Springer, 2000, pp. 451–468.

[61] R. Canetti, J. Kilian, E. Petrank, and A. Rosen, “Black-box concurrent zero-knowledge requires\tilde {Ω}(log n) rounds,” in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. ACM, 2001, pp. 570–579.

[62] M. Prabhakaran, A. Rosen, and A. Sahai, “Concurrent zero knowledge with logarithmic round-complexity,” in *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*. IEEE, 2002, pp. 366–375.

[63] O. Goldreich, “Concurrent zero-knowledge with timing, revisited,” in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. ACM, 2002, pp. 332–340.

[64] R. Pass, “Alternative variants of zero-knowledge proofs,” Ph.D. dissertation, 2004.

[65] D. Bernhard, M. Fischlin, and B. Warinschi, “Adaptive proofs of knowledge in the random oracle model,” *IET Information Security*, vol. 10, no. 6, pp. 319–331, 2016.

[66] D. Bernhard, N. K. Nguyen, and B. Warinschi, “Adaptive proofs have straightline extractors (in the random oracle model),” in *International Conference on Applied Cryptography and Network Security*. Springer, 2017, pp. 336–353.

[67] R. Canetti, O. Goldreich, S. Goldwasser, and S. Micali, “Resettable zero-knowledge,” in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*. ACM, 2000, pp. 235–244.

[68] M. Fischlin, “Communication-efficient non-interactive proofs of knowledge with online extractors,” in *Annual International Cryptology Conference*. Springer, 2005, pp. 152–168.

[69] Y. Desmedt, C. Goutier, and S. Bengio, “Special uses and abuses of the fiat-shamir passport protocol,” in *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 1987, pp. 21–39.

[70] D. Dolev, C. Dwork, and M. Naor, “Non-malleable cryptography,” in *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing*, ser. STOC

’91. New York, NY, USA: ACM, 1991, pp. 542–552. [Online]. Available: http://doi.acm.org/10.1145/103418.103474

[71] A. Sahai, “Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security,” in *Foundations of Computer Science, 1999. 40th Annual Symposium on*. IEEE, 1999, pp. 543–553.

[72] R. Pass and A. Rosen, “New and improved constructions of nonmalleable cryptographic protocols,” *SIAM Journal on Computing*, vol. 38, no. 2, pp. 702–752, 2008.

[73] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai, “Robust non-interactive zero knowledge,” in *Annual International Cryptology Conference*. Springer, 2001, pp. 566–598.

[74] Y. Dodis and A. Yampolskiy, “A verifiable random function with short proofs and keys,” in *International Workshop on Public Key Cryptography*. Springer, 2005, pp. 416–431.

[75] J. Benaloh and D. Tuinstra, “Receipt-free secret-ballot elections,” in *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*. ACM, 1994, pp. 544–553.

[76] S. Delaune, S. Kremer, and M. D. Ryan, “Receipt-freeness: Formal definition and fault attacks,” in *Proceedings of the Workshop Frontiers in Electronic Elections (FEE 2005), Milan, Italy*, 2005.

[77] S. Delaune, S. Kremer, and M. Ryan, “Coercion-resistance and receipt-freeness in electronic voting,” in *Computer Security Foundations Workshop, 2006. 19th IEEE*. IEEE, 2006, pp. 12–pp.

[78] H. L. Jonker and E. P. de Vink, “Formalising receipt-freeness,” in *International Conference on Information Security*. Springer, 2006, pp. 476–488.

[79] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup, “Anonymous identification in ad hoc groups,” in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 609–626.