



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ
ΕΠΙΣΤΗΜΩΝ

Αξιολόγηση σχημάτων
κρυπτογράφησης με δυνατότητα
αναζήτησης

EVALUATION AND VULNERABILITY ANALYSIS OF
SEARCHABLE ENCRYPTION SCHEMES

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Αλέξανδρου Μπάκα

Επιβλέπων: Παγουρτζής Αριστείδης, Αναπληρωτής Καθηγητής
Ε.Μ.Π

Αθήνα, Ιούλιος 2017

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

Αξιολόγηση σχημάτων
κρυπτογράφησης με δυνατότητα
αναζήτησης

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: Παγουρτζής Αριστείδης

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 3η Ιουλίου 2017.

(Υπογραφή)

(Υπογραφή)

(Υπογραφή)

.....
Παγουρτζής Αριστείδης
Αναπληρωτής Καθηγητής
Ε.Μ.Π

.....
Στάθης Ζάχος
Καθηγητής
Ε.Μ.Π.

.....
Στεφανέας Πετρος
Επίκουρος Καθηγητής
Ε.Μ.Π.

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

Copyright ©2017 – All rights reserved Αλέξανδρος Μπάκας
Με την επιφύλαξη παντός δικαιώματος.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή με σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα.

Το περιεχόμενο αυτής της εργασίας δεν απηχεί απαραίτητα τις απόψεις του Τμήματος, του Επιβλέποντα, ή της επιτροπής που την ενέκρινε

Υπεύθυνη Δήλωση

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της διπλωματικής εργασίας, και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται σε αυτήν. Επίσης, έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης, βεβαιώνω ότι η παρούσα εργασία προετοιμάστηκε ειδικά για τις απαιτήσεις του προγράμματος σπουδών της σχολής Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών του Εθνικού Μετσόβιου Πολυτεχνείου.

(Υπογραφή)

.....
Μπάκας Αλέξανδρος

*Στον Κώστα,
που η φιλία μας είναι σαν τα ψηλά βουνά*

Περίληψη

Το παρών κείμενο έχει σκοπό να καταδείξει τεχνολογίες αιχμής, όπως είναι το cloud computing και στη συνέχεια παρουσιάζονται περιγραφικά πρότασεις σύμφωνα με τις οποίες ένας απλός καθημερινός χρήστης μπορεί να χρησιμοποιεί αυτές τις τεχνολογίες χωρίς να κάνει συμβιβασμούς, όσον αφορά το την ασφάλεια και την ιδιωτικότητα του. Πιο συγκεκριμένα μελετάμε συστήματα τα οποία μας επιτρέπουν να αποθηκεύουμε αρχεία στο cloud χωρίς όμως ο provider να γνωρίζει τι αρχεία έχουμε. Αυτό δίνει στον χρήστη μία αίσθηση ασφάλειας όταν τοποθετεί αρχεία στο cloud.

Αναλυτικότερα, δίνονται περιγραφές τέτοιων συστημάτων, τόσο συμμετρικού όσο και δημόσιου κλειδιού, όπου το βασικό σενάριο είναι ένας χρήστης να ανεβάζει κρυπτογραφημένα αρχεία στο cloud και στη συνέχεια μέσω κάποιων token αναζήτησης, ψάχνει στα κρυπτογραφημένα αρχεία, για αυτά που περιέχουν κάποιο keyword που σχετίζεται με το token αναζήτησης. Εξετάζουμε στατικά τέτοια σχήματα, οποία είναι πιο εύκολα για implementation αλλά λιγότερο πρακτικά καθώς οι πιο πολλές βάσεις δεδομένων θέλουμε να έχουν τη δυνατότητα να ανανεώνονται, αλλά και δυναμικά, που αν και είναι πιο πρακτικά διαρρέουν μεγαλύτερο πόσο πληροφορίας. Στη συνέχεια βλέπουμε πως τέτοια συστήματα δημόσιου κλειδιού συνδέονται με άλλες τεχνικές κρυπτογράφησης, όπως το functional encryption και το identity based encryption. Τα συστήματα αυτά αξιολογούνται με βάση κάποιες παραμέτρους, όπως είναι η αποδοτικότητα, η ασφάλεια και ο χρόνος. Το κείμενο κλείνει με μία φιλολογική ενότητα περί ιδιωτικότητας, το οποίο είναι φλέγον ζήτημα παγκοσμίως μετά την έξαρση των τρομοκρατικών χτυπημάτων τον τελευταίο καιρο.

Λέξεις κλειδιά: Searchable encryption, security, cloud storage, cloud.

Abstract

The present text aims to demonstrate cutting-edge technologies, such as cloud computing, and then to show how a simple day-to-day user can use these technologies without making compromises in terms of security and privacy. More specifically, we are studying systems that allow us to store files in the cloud in such a way that the provider learns nothing about the stored files. This gives the user a sense of security when he places files in the cloud. More precisely, descriptions of such symmetric and public key systems are given, where the main scenario is a user uploading encrypted files to the cloud and then through a search token, searches in the encrypted data base for those that contain a keyword associated with the search token. We are examining both static searchable encryption schemes, which are easier to implement, but less practical, as we want most databases to be able to be updated, and dynamics, which, although more practical, leak much more information. We then see how such public key systems are associated with other encryption techniques, such as functional encryption and identity based encryption. These schemes are evaluated on the basis of some parameters, such as efficiency, safety and running time. The text ends with a literary section on privacy, which is a global issue after the outbreak of terrorist attacks lately.

Keywords: Searchable encryption, security, cloud storage, cloud.

Ευχαριστίες

Θέλω να ευχαριστήσω την οικογένεια μου για τη συνεχή στήριξη όλα αυτά τα χρόνια και την εμπιστοσύνη που μου δείχνουν. Επίσης θέλω να ευχαρίστησω και τους φίλους μου οι οποίοι ήταν πάντα εκεί όταν τους χρειαζόμουν ακόμα και αν βρισκόντουσαν πολύ μακριά. Ιδιαίτερες ευχαριστίες στη Μαρία. Τελός, θέλω να ευχαριστήσω τον κύριο Παγουρτζή για την καθοδήγηση του σε αυτή τη διπλωματική, καθώς και τον κύριο Στεφανέα για τη συνέχη στήριξη προς το πρόσωπό μου.

Περιεχόμενα

Εισαγωγή	1
1 Γιατί το Searchable Encryption είναι ιδανικό για το Cloud Computing	5
1.1 Ορολογία και συμβολισμός	6
1.2 Έννοιες ασφάλειας	8
2 Γενικό μοντέλο κρυπτογράφησης με δυνατότητα αναζήτησης, με κρυπτογραφία συμμετρικού κλειδιού.	13
3 Σχήματα που επιτρέπουν την κρυπτογράφηση με δυνατότητα αναζήτησης.	19
3.1 Two-Layered Encryption Scheme	19
3.1.1 The basic Scheme	19
3.1.2 The Final Scheme	21
3.2 Forward Index	22
3.3 Hierarchical Structure of Logarithmic Levels	24
3.4 Inverted Index	24
3.4.1 Επιτυχάνοντας Δυναμικότητα	26
3.5 Keyword Red-Black Tree	28
3.6 Blind Storage	30
4 Περί Ιδιωτικότητας (Privacy Issues)	33
4.1 Η ιδιωτικότητα σε static sse	33
4.2 Η ιδιωτικότητα σε dynamic sse	34

5	Αποδοτικότητα	37
6	Γενικό μοντέλο κρυπτογράφησης με δυνατότητα αναζήτησης, με κρυπτογραφία δημόσιου κλειδιού Public Key encryption with keyword search-PEKS	39
7	Σχήματα δημοσίου κλειδιού που επιτρέπουν την κρυπτογράφηση με δυνατότητα αναζήτησης (PEKS)	43
7.1	Κατασκευή με bilinear map	43
7.2	Κατασκευή με trapdoors permutations	44
8	Μειονεκτήματα του PEKS	47
8.1	Keywords που ανανεώνονται.	48
8.2	Αφαίρεση ασφαλούς καναλιού	49
8.3	Πολλαπλά Keywords	52
9	Functional Encryption και Identity Based Encryption	55
9.1	Functional Encryption	55
9.2	Identity Based Encryption	56
9.2.1	Ασφάλεια IBE	57
10	Άλλες προσεγγίσεις.	61
10.1	PEKS implies IBE	61
10.2	Multiple writers/multiple readers	62
10.2.1	Ντετερμινιστική κρυπτογράφηση	63
10.2.2	Proxy-re encryption	63
11	Μια γενικότερη συζήτηση περί ιδιωτικότητας	67
12	Επίλογος και συμπεράσματα	71
12.1	Συμπεράσματα	72
12.2	Ερευνητικές κατευθύνσεις	74
	Βιβλιογραφία	77

Κατάλογος Πινάκων

Κατάλογος Σχημάτων

3.1	20
3.2	22
3.3	26
3.4	30
3.5	31
3.6	32

Εισαγωγή

Cloud Storage

Τα τελευταία χρόνια, παρατηρούμε μια εκπληκτικά μεγάλη αύξηση στη χρήση του cloud computing. Αξιοποιώντας τις παρόχες ενός τέτοιου συστήματος και εξαλείφοντας προβλήματα όπως ο ανενεργός χρόνος του υπολογιστή, έχουμε ένα επιχειρηματικό μοντέλο το οποίο είναι οικονομικά επωφελές. Μαζί με το γεγονός ότι η ποσότητα των δεδομένων αυξάνεται συνεχώς, το cloud computing παρέχει ένα οικονομικό κίνητρο για πολλές εταιρείες και προσωπικούς χρήστες να επιλέξουν την ανάθεση της αποθήκευσης των αρχείων τους σε κάποιον Cloud Service Provider (CSP). Αυτή η τάση όμως εγείρει ένα ζήτημα ασφάλειας, δεδομένου ότι πολλοί πελάτες θέλουν να διατηρήσουν τα αρχεία τους εμπιστευτικά. Η λύση είναι η κρυπτογράφηση των αρχείων πριν αυτά σταλούν στο CSP.

Υπάρχουν δύο φαινομενικά αντιφατικοί στόχοι που πρέπει να επιτύχει ένα σύστημα κρυπτογράφησης για να είναι χρήσιμο σε αυτό το σενάριο. Από τη μία πλευρά, η κρυπτογράφηση πρέπει να ικανοποιεί μια ισχυρή έννοια της ασφάλειας για να κρατηθούν τα δεδομένα κρυμμένα από τον CSP. Από την άλλη πλευρά, το σύστημα πρέπει να επιτρέπει στους πελάτες να συνεχίζουν αποτελεσματικά τις δραστηριότητές τους, δηλαδή με χρόνο και υπολογιστικό κόστος συγκρίσιμο με το αν η αποθήκευση των αρχείων ήταν τοπική.

Μια βασική εφαρμογή για πολλούς χρήστες είναι το 'search'. Ως εκ τούτου, είναι σημαντικό να αναπτυχθούν και να χρησιμοποιηθούν συστήματα κρυπτογράφησης που να επιτρέπουν την αποδοτική αναζήτηση στα δεδομένα που αποθηκεύονται στο cloud. Εάν οι πελάτες πρέπει να κάνουν λήψη ολόκληρου του συνόλου δεδομένων και να εκτελέσουν την αναζήτηση τοπικά, τότε το σύστημα

είναι εντελώς άχρηστο.

Για παράδειγμα ας φανταστούμε ένα σενάριο στο οποίο ο χρήστης ανεβάζει στο Dropbox 10Gb. Πριν το upload θα κρυπτογραφήσει τα αρχεία του.

Στη συνέχεια εάν θέλει να βρει ένα αρχείο απο αυτά είτε θα πρέπει να κάνει download όλα τα encrypted files, να τα κάνει decrypt και να βρει αυτό που θέλει ή να μπορεί να ψάξει online αυτό που θέλει, χωρίς όμως ο CSP (Dropbox) να 'μάθει' το περιεχόμενο του αρχείου.

Searchable Encryption

(Κρυπτογράφηση με δυνατότητα αναζήτησης)

Η κρυπτογράφηση με δυνατότητα αναζήτησης (SE) είναι μια ενισχυμένη τεχνική κρυπτογράφησης που επιτρέπει την κρυπτογράφηση παράλληλα με την αναζήτηση λέξεων-κλειδιών στα κρυπτογραφημένα δεδομένα (όπως θα ήταν δυνατό σε απλά κείμενα). Η βασική εφαρμογή της είναι η αποθήκευση στο cloud. Στα searchable encryption schemes θα πρέπει να είναι δυνατή για τον CSP, με τη βοήθεια κάποιου token αναζήτησης που αποστέλλεται από τον πελάτη, να εκτελέσει ορισμένες λειτουργίες και στη συνέχεια να στείλει τα σχετικά δεδομένα στον πελάτη. Τα σχετικά δεδομένα θα πρέπει να είναι τέτοια ώστε, αφενός να περιέχουν τα αντίστοιχα έγγραφα (δηλαδή τα έγγραφα που περιέχουν τη λέξη-κλειδί της αναζήτησης) και αφετέρου το μέγεθος τους να μην είναι πολύ μεγαλύτερο από αυτό των αντίστοιχων εγγράφων. (ο Provider δεν μπορεί απλά να στέλνει τη μισή database σε κάθε αναζήτηση). Φυσικά ο CSP δεν πρέπει να μάθει τη λέξη-κλειδί της αναζήτησης, αλλιώς μαθαίνει μερικές πληροφορίες σχετικά με τα έγγραφα.

Στον τομέα της κρυπτογράφησης με δυνατότητα αναζήτησης υπάρχουν σχέσεις συμβιβασμού μεταξύ της αποδοτικότητας, της λειτουργικότητας και της ασφάλειας.

Για την αποδοτικότητα, είναι επιθυμητό να μειωθεί όσο το δυνατόν περισσότερο ο αριθμός των πράξεων που εκτελείται από το διακομιστή κατά τη διάρκεια μιας αναζήτησης. Είναι επίσης πολύ σημαντικό να κάνουμε αυτές τις λειτουργίες παραλληλισμένες (προκειμένου να βελτιωθεί η απόδοση εισόδου/εξόδου),

έτσι ώστε ο χρόνος αναζήτησης να βελτιωθεί.

Από την σκοπιά της λειτουργικότητας, μια σημαντική παράμετρος είναι η 'εκφραστικότητα' του ερωτήματος. Ένα σύστημα SE θα πρέπει να υποστηρίζει όσο το δυνατόν πιο δυνατά ερωτήματα, αυξάνοντας έτσι τη χρησιμότητα του προγράμματος στους πελάτες. Άλλες σημαντικές παράμετροι είναι εάν ένας ή πολλοί πελάτες αποθηκεύουν δεδομένα στο Cloud και αν ένας ή πολλοί πελάτες θα πρέπει να μπορούν να διαβάζουν τα δεδομένα. Επιπλέον, θα πρέπει τα συστήματα να είναι δυναμικά. Θα πρέπει δηλαδή να επιτρέπεται η ενημέρωση της βάση δεδομένων χωρίς πρόσθετη διαρροή πληροφοριών.

Από την άποψη της ασφάλειας, είναι σημαντικό να μειωθεί η διαρροή πληροφοριών που προκαλείται απ' όλες τις λειτουργίες όσο το δυνατόν περισσότερο.

Ανάλογα με τις απαιτήσεις του εκάστοτε συστήματος, είναι δυνατόν να χρησιμοποιηθεί είτε κρυπτογραφία δημόσιου κλειδιού ή κρυπτογράφηση συμμετρικού κλειδιού, αλλά γενικά, τα public key encryption schemes with keyword search (PEKS) δεν αποδίδουν όσο καλά θα θέλαμε καθώς έχουν χρόνο αναζήτησης γραμμικό στον αριθμό των εγγράφων.

Εισαγωγή στην ιδέα του PEKS έκανε ο Boneh, ο οποίος έφτιαξε ένα scheme που επιτρέπει σε πολλούς χρήστες να κρυπτογραφούν αρχεία σε μία βάση δεδομένων, με ένα δημόσιο κλειδί, τα οποία όμως μπορούν να αποκρυπτογραφηθούν μόνο από τον 'κάτοχο' της βάσης με το μυστικό του κλειδί. Η αποδοτικότητα ενός τέτοιου σχήματος περιορίζεται από το κόστος των λειτουργιών του δημοσίου κλειδιού.

Η συμμετρική κρυπτογράφηση με δυνατότητα αναζήτησης, εισήχθη από τον Song, ο οποίος παρουσιάζει ένα scheme το οποίο, επιτρέπει τον γραμμικό χρόνο αναζήτησης (στον αριθμό των εγγράφων) από το διακομιστή. Δυστυχώς το scheme τους δεν επιτυγχάνει μια ισχυρή έννοια ασφάλειας: Δεν έχει καμία εγγύηση ασφάλειας που σχετίζεται με τη διαρροή που μπορεί να προκαλείται από τη χρήση των αναγνωριστικών αναζήτησης (search tokens) που δίνονται στο διακομιστή, ώστε να επιτρέπεται η αναζήτηση που εκτελείται από την πλευρά του διακομιστή. Ο Goh εισήγαγε την προσέγγιση της χρήσης ασφαλών indexes για να επιτευχθεί γραμμικός χρόνος αναζήτησης με ισχυρότερες εγγυήσεις ασφαλείας. Δυστυχώς ο χρόνος αναζήτησης αυτής της προσέγγισης είναι εγγενώς γραμμικός στον αριθμό των αρχείων. Ο Curtmola παρουσίασε το πρώτο

ασφαλές σχήμα με sublinear search time χρησιμοποιώντας αντεστραμμενους indexes (χρησιμοποιώντας τις λέξεις -κλειδιά ως ευρετήριο) και εισήγαγε επίσης ένα ισχυρό μοντέλο ασφάλειας για κρυπτογράφηση με δυνατότητα αναζήτησης που έθεσε τα 'standards' στην ασφαλεία των searchable encryption schemes τα τελευταία χρόνια.

Η προσέγγιση του αντεστραμμένου index είναι αρκετά αποδοτική και είναι στην πραγματικότητα βέλτιστη ως προς τον αριθμό των λειτουργιών που πρέπει να εκτελέσει ο διακομιστής κατά τη διάρκεια μιας αναζήτησης. Ένας περιορισμός αυτής της μεθόδου είναι ότι είναι εγγενώς διαδοχική και επομένως είναι δύσκολο να εκμεταλλευτούμε τον παραλληλισμό για να βελτιώσουμε τις επιδόσεις του. Ένα άλλο πρόβλημα είναι ότι δεν είναι κατάλληλη για δυναμικές βάσεις δεδομένων, όπως συμβαίνει με τις περισσότερες εφαρμογές. Πιο πρόσφατες εργασίες έχουν κάνει βήματα προς την κατεύθυνση των δυναμικών βάσεων δεδομένων.

Η συμμετρική κρυπτογράφηση με δυνατότητα αναζήτησης ταιριάζει απόλυτα στο σενάριο όπου υπάρχει μόνο ένας χρήστης που γράφει και διαβάζει από τη βάση δεδομένων.

Όσον αφορά την εκφραστικότητα των ερωτημάτων, τα περισσότερα συμμετρικά συστήματα με δυνατότητα αναζήτησης βασίζονται σε ερωτήματα ισότητας, αλλά όπως δείχνουν μερικά πρόσφατα αποτελέσματα, είναι εφικτό να επεκταθούν οι δομές δεδομένων που χρησιμοποιούνται, για συμμετρική κρυπτογράφηση με δυνατότητα αναζήτησης, με σκοπό την αντιμετώπιση πιο περίπλοκων ερωτημάτων, όπως ερωτήματα για συνδυασμούς λέξεων-κλειδιών.

Παρακάτω θα ασχοληθούμε με όλα τα συστήματα που αναφέρθηκαν, PEKS και symmetric, καθώς και με κάποιες παραλλαγές τους. Πριν από αυτό όμως, θα εξετάσουμε σε βάθος γιατί η κρυπτογράφηση με δυνατότητα αναζήτησης ταιριάζει πάρα πολύ με το cloud computing.

Κεφάλαιο 1

Γιατί το Searchable Encryption είναι ιδανικό για το Cloud Computing

Όπως είναι λογικό, παράλληλα με την αυξανόμενη χρήση του Cloud Computing τα τελευταία χρόνια, ακόμα και από απλούς χρήστες σε καθημερινή βάση (Dropbox,i-cloud), άρχισαν να γίνονται και οι πρώτες επιθέσεις οι οποίες είχαν στόχο την υποκλοπή δεδομένων. Η επιτυχία αυτών των επιθέσεων είχε ως αποτέλεσμα τη δυσπιστία πολλών δυνητικών χρηστών απέναντι στο cloud. Οι πιο πολλοί από αυτούς φοβούνται να εμπιστευθούν τα ευαίσθητα αρχεία τους σε κάποιον τρίτο για αποθήκευση αφού χωρίς ισχυρά πιστοποιητικά ασφάλειας τα αρχεία είναι πιθανό να πέσουν στα χέρια τρίτων.

Η κοινή λογική λέει πως η λύση είναι η κρυπτογράφηση των αρχείων όταν αυτά βρίσκονται online. Αυτό όμως εγείρει ακόμα ένα ζήτημα ασφάλειας καθώς όλη η εμπιστοσύνη για την ασφάλεια των δεδομένων μας περνά στα χέρια του διακομιστή. Με άλλα λόγια όταν ο διακομιστής γίνεται υπεύθυνος για την κρυπτογράφηση των δεδομένων μας, έχει στα χέρια του το κλειδί κρυπτογράφησης/αποκρυπτογράφησης, δημιουργώντας έτσι αμφιβολίες για την ασφάλεια των δεδομένων μας σε περιπτώσεις ενός κακόβουλου Provider ή διαχειριστή. Εδώ έρχεται να μπει στο puzzle η κρυπτογράφηση με δυνατότητα αναζήτησης.

Πιο συγκεκριμένα οι χρήστες μπορούν να κρυπτογραφήσουν τα δεδομένα

τους πριν τα ανεβάσουν σε μια online βάση δεδομένων και έπειτα μπορούν να κάνουν αναζήτηση στη βάση για κάποιο αρχείο, παρακάμπτοντας έτσι το βήμα της αποκρυπτογράφησης. Μπορούν δηλαδή να κάνουν κρυπτογράφηση τοπικά, πριν ανεβάσουν τα δεδομένα τους στο cloud και μετά μπορούν να αναζητήσουν απευθείας τα κρυπτογραφημένα δεδομένα, τα οποία είναι αποθηκευμένα στο cloud. Με αυτόν τον τρόπο οι χρήστες θα είναι σίγουροι ότι τα αποθηκευμένα δεδομένα τους θα είναι ασφαλή, ακόμα και στην περίπτωση κακόβουλου provider, αφού οι ίδιοι θα είναι οι μόνοι που θα γνωρίζουν το κλειδί κρυπτογράφησης. Με άλλα λόγια, ακόμα και ο αν ο provider αποφασίσει να παραβιάσει το απόρρητο των χρηστών, δε θα είναι σε θέση να βρει καμία πληροφορία, εφόσον το κρυπτοσύστημα είναι ασφαλές.

Γενικά, τα searchable encryption schemes προσπαθούν να παρέχουν στους χρήστες εμπιστοσύνη, διατηρώντας παράλληλα τα πλεονεκτήματα της αποθήκευσης στο cloud (διαθεσιμότητα ανα πάσα στιγμή, κοινή χρήση δεδομένων κτλ), δημιουργώντας τις κατάλληλες προϋποθέσεις ασφάλειας. Ωστόσο, μέχρι και σήμερα δεν υπάρχει κάποιος φορέας παροχής υπηρεσιών cloud που να υποστηρίζει τέτοια συστήματα. Αυτό οφείλεται, κυρίως, στην πρόσθετη έρευνα που απαιτείται για τη δημιουργία μίας αποτελεσματικής και αξιόπιστης εφαρμογής.

1.1 Ορολογία και συμβολισμός

Πριν συνεχίσουμε θα δώσουμε κάποιους ορισμούς, οι οποίοι θα είναι ιδιαίτερα χρήσιμοι παρακάτω.

Ορισμός 1.1.1. (*Random Oracle*) *Random Oracle* λέμε ένα 'μαντείο', το οποίο απαντάει σε κάθε ερώτηση με μία τυχαία απάντηση. Αν η ίδια ερώτηση επαναληφθεί, η απάντηση θα είναι πάντα η ίδια.

Ορισμός 1.1.2. Έστω $A : \{0, 1\}^n \rightarrow \{0, 1\}$ ένας αυθαίρετος αλγόριθμος και έστω X και Y τυχαίες μεταβλητές κατανομημένες στο $\{0, 1\}^n$. Ορίζουμε ως πλεονέκτημα του A την ποσότητα

$$Adv A = |Pr[A(X) = 1] - Pr[A(Y) = 1]|$$

Ορισμός 1.1.3. (*Pseudorandom Generator*) Ένας *Pseudorandom Generator* G είναι ένα *stream cipher*. Λέμε ότι ο G είναι (t, e) -ασφαλής αν για κάθε αλγόριθμο με *running time* το πολύ t έχει *advantage* $AdvA < e$.

Ορισμός 1.1.4. (*pseudorandom function*) Λέμε ότι μια *pseudorandom function* $F : K \times X \rightarrow Y$ είναι (t, q, e) -ασφαλή αν κάθε αλγόριθμος A στο *random oracle model* που κάνει το πολύ q ερωτήματα και έχει *running time* το πολύ t έχει *advantage*

$$AdvA < e$$

Ορισμός 1.1.5. (*Pseudorandom Permutation*) Μία *Pseudorandom Permutation* E είναι ένα *block cipher*. Λέμε ότι $E : K \times Z \rightarrow Z$ είναι (t, q, e) -ασφαλή *pseudorandom function* αν κάθε αλγόριθμος A στο *random oracle model* που κάνει το πολύ q ερωτήματα και έχει *running time* το πολύ t έχει *advantage* $AdvA < e$

Ορισμός 1.1.6. (*Bloom Filters*) *Probabilistic* δομή δεδομένων που χρησιμοποιείται για να ελέγξει εάν ένα στοιχείο ανήκει σε κάποιο σύνολο. Υποστηρίζει 2 λειτουργίες, *test* και *add*. Η *test* είναι *boolean* και χρησιμοποιείται για να ελέγξουμε εάν ένα δοθέν στοιχείο ανήκει στο σύνολο. Εάν επιστρέψει :

- (i) *false*, τότε το στοιχείο σίγουρα δεν ανήκει στο σύνολο.
- (ii) *true*, τότε το στοιχείο μάλλον ανήκει στο σύνολο.

Ορισμός 1.1.7. (*Διγραμμική απεικόνιση*) Έστω G_1, G_2 ομάδες με τάξη έναν πρώτο αριθμό p και μία διγραμμική απεικόνιση $e : G_1 \times G_1 \rightarrow G_2$. Η απεικόνιση ικανοποιεί τις επόμενες ιδιότητες:

- (i) Είναι υπολογίσιμο. Δεδομένων g, h που ανήκουν στο G_1 υπάρχουν πολωνιμικού χρόνου αλγόριθμοι που υπολογίζουν αν το $e(g, h)$ ανήκει στο G_2 .
- (ii) είναι διγραμμική. Για κάθε ακεραίους x, y στο $[0, 1]$ ισχύει : $e(g^x, g^y) = e(g, g)^{xy}$.
- (iii) *Non-degenerate*. Αν g είναι ένας γεννήτορας του G_1 , τότε το $e(g, g)$ είναι γεννήτορας του G_2 .

Ορισμός 1.1.8. (Ομομορφική Κρυπτογράφηση) Μία μορφή κρυπτογράφησης η οποία επιτρέπει υπολογισμούς σε κρυπτοκείμενο χωρίς να χρειάζεται η αποκρυπτογράφηση του.

Ορισμός 1.1.9. *Negligible function* Με τον όρο *negligible function* θα αναφερόμαστε σε μια συνάρτηση $f : [0, 1] \rightarrow \mathbb{R}$ για την οποία θα ισχύει:

$$f < \frac{1}{g(s)}$$

όπου g είναι *polynomial* και το s αρκετά μεγάλο.

1.2 Έννοιες ασφάλειας

Το πρώτο σχήμα κρυπτογράφησης με δυνατότητα αναζήτησης Song (2000), που είναι και το πρώτο σχήμα που μελετάει αυτό το κείμενο, δεν όριζε την ασφάλεια για τις ανάγκες ενός τέτοιου σχήματος. Ωστόσο, αποδείχτηκε ότι το σχήμα ήταν ένας ασφαλής pseudorandom generator . Αργότερα αποδείχθηκε ότι ήταν και indistinguishable against chosen plaintext attacks (IND-CPA) ασφαλές. Γενικά ένα σχήμα κρυπτογράφησης, είναι IND-CPA ασφαλές αν ένας αντίπαλος A δεν μπορεί να ξεχωρίσει τις κρυπτογραφήσεις δύο τυχαίων plaintext (τα οποία έχει επιλέξει ο A) ακόμα και αν ο A έχει το δικαίωμα να κάνει ερωτήματα σε ένα encryption oracle. Αυτό σημαίνει ότι ένα σχήμα είναι IND-CPA ασφαλές αν τα κρυπτοκείμενα δεν διαρρέουν ούτε μερική πληροφορία για τα plaintext. Αυτός ο ορισμός σιγουρεύει ότι δε θα υπάρχει διαρροή πληροφορίας. Βέβαια σε searchable encryption σχήματα η κύρια διαρροή πληροφορίας έρχεται απο ερωτήματα προς τον server για τα search tokens ή trapdoors, κάτι το οποίο δεν λαμβάνεται υπ όψιν στην IND-CPA ασφάλεια. Για το λόγο αυτό, η IND-CPA ασφάλεια δεν μπορεί να είναι η σωστή έννοια της ασφάλειας για ένα searchable encryption σχήμα.

Η πρώτη έννοια ασφάλειας για το searchable encryption ήρθε το 2003 απο τον Goh ο οποίος όρισε την ασφάλεια για indexes ως semantic security (indistinguishability) against adaptive chosen keywords attacks (IND1-CKA). Η IND1-CKA ασφάλεια σιγουρεύει οτι ο αντίπαλος A δεν μπορεί να συμπεράνει

το περιεχόμενο ενός αρχείου από το index του αρχείου. Ένα IND1-CKA ασφαλές σχήμα παράγει indexes που περιέχουν τον ίδιο αριθμό από λέξεις για ίδιου μεγέθους αρχεία. Αυτό σημαίνει ότι δεδομένων δύο κρυπτογραφημένων αρχείων ίδιου μεγέθους και ενός index, ο αντίπαλος A δεν μπορεί να ξεχωρίσει ποιο αρχείο είναι κωδικοποιημένο στο index.

Το 2005 οι Chan και Mitzenmacher εισήγαγαν μία πιο ισχυρή έννοια ασφάλειας (πιο ισχυρή μορφή της IND1-CKA), με την έννοια ότι ο αντίπαλος A δεν μπορούσε να ξεχωρίσει indexes ούτε από ανόμοιου μεγέθους αρχεία. Αυτό βέβαια προϋποθέτει ότι ανόμοιου μεγέθους αρχεία έχουν indexes που περιέχουν τον ίδιο αριθμό λέξεων.

Αργότερα ο Goh εισήγαγε τον ορισμό της IND2-CKA ασφάλειας που να μην προστατεύει το μέγεθος των αρχείων αλλά όπως και τα προηγούμενα αποτυγχάνει να παραδώσει ασφάλεια όσον αφορά τα ερωτήματα για τα search tokens. Τα IND1/2-CK θεωρούνται αρκετά ασθενή όσον αφορά την ασφάλεια ενός searchable encryption σχήματος από την άποψη ότι δεν εγγυούνται την ασφάλεια των search tokens. Με άλλα λόγια δεν εγγυώνται ότι ο server δεν μπορεί να ανακτήσει πληροφορίες για τις λέξεις προς αναζήτηση από τα search tokens

Ο Curtmola το 2006 προσέγγισε διαφορετικά το ζήτημα της ασφάλειας και αποφάνθηκε πως η ασφάλεια των indexes και αυτή των Search tokens συνδέονται με άμεσο τρόπο. Εισήγαγε λοιπόν, δύο νέα μοντέλα ασφάλειας, ένα nonadaptive (IND-CKA1) και ένα adaptive (IND-CKA2) τα οποία ακόμα και σήμερα χρησιμοποιούνται ως το ελάχιστο standard για την ασφάλεια ενός searchable encryption σχήματος. Τα δύο αυτά μοντέλα απαιτούν να μην διαρρέεται καμία πληροφορία εκτός από το αποτέλεσμα της αναζήτησης και το search pattern των ερωτημάτων. Οι IND-CKA1/2 ορισμοί, παρέχουν ασφάλεια για τα search tokens και εγγυώνται πως τα search tokens δε διαρρέουν πληροφορίες για τα keywords (εκτός από τις πληροφορίες που διαρρέουν από το search pattern και το access pattern. Ο nonadaptive ορισμός εγγυάται την ασφάλεια ενός σχήματος, αν ο χρήστης κάνει όλα τα ερωτήματα μονομιάς, κάτι που δεν είναι εφικτό σε πολλά σενάρια. Ο adaptive ορισμός επιτρέπει στον αντίπαλο A να διαλέγει τα ερωτήματα που θα κάνει ως συνάρτηση κάποιων search tokens που έχει ήδη στην κατοχή του. Ο IND-CKA2 ορισμός παρέχει μία ισχυρή έννοια

ασφάλειας για symmetric searchable encryption σχήματα.

Στην περίπτωση της μη συμμετρικής κρυπτογράφησης, (Boneh 2004) τα σχήματα είναι πολύ δύσκολο να παρέχουν ασφάλεια για τα search tokens μίας και αυτά δημιουργούνται από ένα δημόσιο κλειδί. Σε αυτή την περίπτωση ο ορισμός εγγυάται ότι καμία πληροφορία δεν διαρρέεται για ένα keyword, εκτός και αν είναι ήδη γνωστό το search token για αυτό. Ένας αντίπαλος δηλαδή, δεν θα μπορεί να ξεχωρίσει δύο encryptions από δύο keywords ακόμα και αν έχει στην κατοχή του search tokens για όποιο άλλο keyword θέλει.

Ντετερμινιστική κρυπτογράφηση: Η ντετερμινιστική κρυπτογράφηση δεν έχει καθόλου τυχαιότητα και συνεπώς πάντα παράγει το ίδιο κρυπτοκείμενο για ένα plaintext και ένα κλειδί. Στην περίπτωση το δημόσιου κλειδιού, αυτό σημαίνει ότι μία ντετερμινιστική κρυπτογράφηση δεν μπορεί ποτέ να είναι IND-CPA ασφαλή, αφού ένας αντίπαλος θα μπορεί να τρέξει brute force attacks προσπαθώντας να κατασκευάσει όλα τα πιθανά ζεύγη plaintexts ciphertexts χρησιμοποιώντας την συνάρτηση κρυπτογράφησης. Η ντετερμινιστική κρυπτογράφηση παράγει πιο αποδοτικά σχήματα τα οποία όμως είναι λιγότερο ασφαλή από τα σχήματα που κρυπτογραφούνται με probabilistic κρυπτογράφηση. Τα σχήματα ντετερμινιστικής κρυπτογράφησης προσεγγίζουν το πρόβλημα της αναζήτησης σε κρυπτογραφημένα αρχεία από μία πρακτική σκοπιά με στόχο την καλύτερη απόδοση των σχημάτων. Ένα παράδειγμα της αδυναμίας τους, όσον αφορά την ασφάλεια, είναι ότι απο τη φύση της, η ντετερμινιστική κρυπτογράφηση, διαρρέει ισότητα μηνυμάτων. Ο Bellare όρισε την ασφάλεια για searchable encryption σχήματα δημόσιου κλειδιού με τρόπο παρόμοιο της IND-CPA ασφάλειας, με κάποιες όμως διαφορές, όπως για παράδειγμα τα plaintext να έχουν μεγάλη min entropy. Η μεγάλη ελάχιστη εντροπία, εξασφαλίζει ότι δε θα είναι εύκολο για τον αντίπαλο να κάνει btur force attacks. Γενικά η ντετερμινιστική κρυπτογράφηση δεν είναι αρκετά καλή για διάφορες πρακτικές εφαρμογές, καθώς διαρρέονται παρα πολλές πληροφορίες.

Random oracle model VS standard model : Τα searchable encryption σχήματα αποδυναμώνονται ασφαλή είτε στο Random oracle model ή στο standard model. Το standard model είναι ένα υπολογιστικό μοντέλο στο οποίο ένας αντίπαλος περιορίζεται μόνο απο τα resources που έχει (χρόνος και υπολογιστική δύναμη). Αυτό σημαίνει ότι χρειάζονται μόνο computationals assumptions

για να αποδειχθεί ένα σχήμα ασφαλές. Το Random oracle model από την άλλη αντικαθιστά κρυπτογραφικά primitives με ιδανικές εκδοχές (για παράδειγμα αντικαθιστά μια hash function με μία πραγματικά random συνάρτηση). Λύσεις στο random oracle model είναι γενικά πιο αποδοτικές αλλά έχουν τις επιπλέον υποθέσεις για τα κρυπτογραφικά primitives

Κεφάλαιο 2

Γενικό μοντέλο κρυπτογράφησης με δυνατότητα αναζήτησης, με κρυπτογραφία συμμετρικού κλειδιού.

Η κρυπτογράφηση με δυνατότητα αναζήτησης επιτρέπει στους χρήστες να παράγουν τα λεγόμενα token αναζήτησης τα οποία στέλνονται στον CSP για να μπορέσει αυτός να ψάξει στα κρυπτογραφημένα δεδομένα.

Θεωρούμε τα δεδομένα σαν μια συλλογή:

$F = (f_1, f_2, \dots, f_n)$ από n αρχεία όπου κάθε f_i είναι μία ακολουθία λέξεων (w_1, w_2, \dots, w_n) από κάποιο keyword space. Επιπλέον κάθε αρχείο f_i έχει έναν μοναδικό identifier: $\text{id}(f_i)$.

Αν το searchable encryption scheme είναι δυναμικό, δηλαδή να υποστηρίζει προσθήκες και διαγραφές από τη βάση δεδομένων, ο χρήστης θα κάνει generate και add/delete tokens τα οποία θα στέλνει στον provider με σκοπό την ανανέωση της βάσης.

Παρακάτω δίνεται ο ορισμός ενός δυναμικού symmetric index-based searchable encryption σχήματος (dynamic index based sse).

Ορισμός 2.0.1. Ένα *dynamic index based sse* αποτελείται από 9 αλγόριθμους $SSE=(Gen, Enc, SearchToken, AddToken, DeleteToken, Search, Add, Delete, Dec)$

όπου :

- (i) *Gen (Generate)*: Probabilistic key-generator αλγόριθμος. Δέχεται σαν input μία security parameter και βγάζει σαν output το secret key K . (χρησιμοποιείται από τον χρήστη για τη δημιουργία του Secret key K).
- (ii) *Enc (Encrypt)*: Probabilistic αλγόριθμος ο οποίος δέχεται σαν input ένα secret key K και μία ακολουθία από αρχεία f_i και βγάζει σαν output έναν κρυπτογραφημένο δείκτη (encrypted index) γ και μία ακολουθία από ciphertexts c_i (χρησιμοποιείται από τον χρήστη για να παράξει ciphertexts που αντιστοιχούν στα plaintexts αρχεία του, καθώς και έναν encrypted index, τα οποία μετά τα στέλνει στο server).
- (iii) *SearchToken*: Probabilistic αλγόριθμος. Δέχεται σαν input το secret key K και ένα keyword w και βγάζει ένα output search token $t_s(w)$ για τη λέξη w . (χρησιμοποιείται από τον χρήστη για τη δημιουργία token για μία συγκεκριμένη λέξη, το οποίο μετά στέλνεται στο server).
- (iv) *AddToken*: Probabilistic αλγόριθμος. Δέχεται input το secret key K και ένα file f και βγάζει output ένα add token $t_a(f)$ και ένα ciphertext c_f . (χρησιμοποιείται από τον χρήστη με σκοπό τη δημιουργία ενός add token για νέο αρχείο καθώς και την κρυπτογράφηση του αρχείου, τα οποία μετά αποστέλλονται στο server).
- (v) *DeleteToken*: Probabilistic αλγόριθμος. Δέχεται input το secret key K και ένα file f και βγάζει output ένα delete token $t_d(f)$. (χρησιμοποιείται από τον χρήστη για τη δημιουργία ενός delete token για κάποιο αρχείο και εν συνεχεία αποστέλλεται στο server).
- (vi) *Search*: Deterministic αλγόριθμος. Δέχεται σαν input έναν encrypted index γ , μία ακολουθία από ciphertexts και ένα search token $t_s(w)$ και δίνει output μια ακολουθία από file identifiers I_w . (χρησιμοποιείται από το server μόλις αυτός παραλάβει ένα search token με σκοπό να ψάξει στην

κρυπτογραφημένη βάση δεδομένων και να αποφασίσει ποια *ciphertexts* αντιστοιχούν στη λέξη κλειδί που αναζητείται, με σκοπό να τα στείλει πίσω στον χρήστη).

- (vii) *Add: Deterministic* αλγόριθμος. Δέχεται *input* έναν *encrypted index* γ , μία ακολουθία από *ciphertexts* c και ένα *add token* $t_a(f)$ και δίνει *output* έναν νέο *encrypted index* γ' και μία νέα ακολουθία από *ciphertexts* c' . (χρησιμοποιείται απο το *server* μόλις λάβει ενα *add token* με σκοπό την ανανέωση της βάσης δεδομένων).
- (viii) *Delete: Deterministic* αλγόριθμος. Δέχεται σαν *input* έναν *encrypted index* γ , μία ακολουθία από *ciphertexts* c και ένα *delete token* $t_d(w)$ και δίνει *output* έναν νέο *encrypted index* γ' και μία νέα ακολουθία απο *ciphertexts* c' . (χρησιμοποιείται από τον *server* μόλις λάβει ένα *delete token* με σκοπό να διαγράψει από τη βάση τα αρχεία που αντιστοιχούν στο *delete token*).
- (ix) *Dec (Decrypt): Deterministic* αλγόριθμος. Δέχεται σαν *input* το *secret key* K και ένα *ciphertext* c και δίνει *output* ένα *plaintext* αρχείο f . (χρησιμοποιείται από τον χρήστη για να αποκρυπτογραφήσει τα *ciphertexts* που λαμβάνει από το *server*).

Ένα στατικό *index based sse* μπορεί να οριστεί αν παραλείψουμε τους αλγορίθμους 4, 5, 7, 8 δηλαδή τους *AddToken, DeleteToken, Add* και *Delete*. Σχετικά με την ασφάλεια, μία διαισθητική πρώτη προσέγγιση είναι να ορίσουμε ως ασφαλές ένα σύστημα στο οποίο καμία πληροφορία δεν διαρρέεται στον *server* πέρα απο το αποτέλεσμα της αναζήτησης (γνωστό και ως *access pattern*), όπως για παράδειγμα οι *identifiers* των αρχείων που περιέχουν το *keyword* που αναζητήθηκε. Δυστυχώς όμως σε ένα *sse* διαρρέουν πιο πολλές πληροφορίες, όπως για παράδειγμα εάν έγιναν δύο αναζητήσεις για το ίδιο *keyword*, το λεγόμενο *search pattern*. Το *search pattern* π.χ. διαρρέει εάν χρησιμοποιήθουν ντετερμινιστικά *search tokens*, κάτι που συμβαίνει στις πιο αποδοτικές, από άποψη χρόνου, περιπτώσεις.

Με αυτά σαν δεδομένα μπορούμε να πούμε οτι ένα *sse* θα είναι ασφαλές εάν δεν διαρρέεται τίποτα παραπάνω απο το *search pattern* και το *access pattern*.

Παρακάτω δίνεται ο ορισμός της ασφάλειας ενάντια σε επιθέσεις επιλεγμένων keyword. (Security against adaptive chosen keyword attacks(CKA2-Security)).

Ως L_i, L_a, L_s και L_d ονομάζουμε τις leakage functions που σχετίζονται με τη δημιουργία του index, την αναζήτηση, την πρόσθεση και τη διαγραφή με βάση ένα παιχνίδι, όπως συχνά γίνεται στην κρυπτογραφία.

Ορισμός 2.0.2. Έστω $SSE=(Gen, Enc, SearchToken, AddToken, DeleteToken, Search, Add, Delete, Dec)$ ένα *dynamic index based symmetric searchable encryption scheme* και L_i, L_s, L_a, L_d οι *leakage functions*. Πραγματοποιούμε το παρακάτω πείραμα.

1. $Real_A(\lambda)$: Το secret key K δημιουργείται τρέχοντας τον αλγόριθμο $Gen(\lambda)$. Ο αντίπαλος διαλέγει μια συλλογή αρχείων f και έναν encrypted index γ μαζί με μία συλλογή αρχείων c έτσι ώστε $(\gamma, c) \leftarrow Enc(K, f)$.

Ο αντίπαλος ξεκινάει να κάνει ερωτήματα για να πάρει *search, add* και *delete tokens*. Τα *tokens* δημιουργούνται από τους αλγορίθμους του SSE και στέλνονται στον αντίπαλο. Στο τέλος ο αντίπαλος δίνει output ένα bit b καταδεικνύοντας εάν πιστεύει πως είναι το πραγματικό ή το ιδεατό παιχνίδι.

2. $Ideal_{A,S}(\lambda)$: Ο αντίπαλος διαλέγει μία συλλογή αρχείων f . Ένας simulator παίρνει την $L_i(f)$ και κάνει *simulate* έναν encrypted index και ciphertexts c τα οποία στέλνει στον αντίπαλο. Ο αντίπαλος έχει και πάλι το δικαίωμα να κάνει ερωτήματα για να πάρει *search, add* και *delete tokens* αλλά αυτή τη φορά ο simulator παράγει τα *tokens* που θα στείλει στον αντίπαλο χρησιμοποιώντας μόνο την διαρροή από τις L_s, L_a και L_d . Στο τέλος ο αντίπαλος δίνει output ένα bit b καταδεικνύοντας εάν πιστεύει πως είναι το πραγματικό ή το ιδεατό παιχνίδι.

Ένα SSE είναι L_i, L_s, L_a, L_d -ασφαλές ενάντια σε *chosen keyword attacks* εάν για όλους τους *probabilistic polynomial time* αντιπάλους A , υπάρχει ένας *probabilistic polynomial time simulator* τέτοιος ώστε :

$$|Pr[Real_A(\lambda) = 1] - Pr[Ideal_{A,S}(\lambda) = 1]| \leq negl(\lambda).$$

Με άλλα λόγια, αυτός ο ορισμός μας λέει ότι αν κανένας αντίπαλος δεν μπορεί να ξεχωρίσει αν τα encrypted index, ciphertexts και tokens που του δοθήκαν, δημιουργήθηκαν από τα αληθινά δεδομένα και το SSE ή απο έναν simulator, ο οποίος επεξεργάζεται μόνο πληροφορίες απο τις leakage functions, τότε το SSE διαρρέει μόνο πληροφορίες που σχετίζονται με τις leakage functions.

Με βάση τα παραπάνω μπορούμε να δεχτούμε σαν καλή περίπτωση αυτήν όπου,

- Η L_i διαρρέει μόνο τον αριθμό των αρχείων, τα keywords, τους identifiers των αρχείων και το μέγεθος των αρχείων.
- Η L_s διαρρέει μόνο το search pattern και το access pattern.
- Η L_a διαρρέει μόνο το μέγεθος και το identifier του προστιθέμενου αρχείου καθώς και τον καινούριο αριθμό των keywords.
- Η L_d διαρρέει μόνο τον καινούριο αριθμό των keywords.

Κεφάλαιο 3

Σχήματα που επιτρέπουν την κρυπτογράφηση με δυνατότητα αναζήτησης.

3.1 Two-Layered Encryption Scheme

Η βασική ιδέα εδώ είναι να κρυπτογραφείται το κάθε keyword ξεχωριστά με ντετερμινιστικό τρόπο και μετά με τη χρήση ενός stream cipher να γίνεται η δεύτερη κρυπτογράφηση.

Για μία λέξη w υπολογίζουμε ντετερμινιστική κρυπτογράφηση της, εστώ $x = E(w)$. Στη συνέχεια η κρυπτογραφημένη λέξη χωρίζεται σε δύο μέρη $x = x_r || x_l$. Το x_r χρησιμοποιείται για τη δημιουργία ενός κλειδιού k μίας hash function h και παράλληλα ένα τυχαίο seed s γίνεται XOR με το x_l και υπολογίζουμε το $h(k, s)$ το οποίο στη συνέχεια γίνεται XOR με το x_r . Το Search Token για μία λέξη w είναι $T(w) = E(w)$ και το κλειδί k είναι αυτό που παράχθηκε από το x_a . Ο server έτσι μπορεί να πραγματοποιήσει την αναζήτηση ελέγχοντας για κάθε ciphertext αν το $c \oplus x$ είναι της μορφής $s || h(k, s)$.

3.1.1 The basic Scheme

Η αρχική ιδέα έχει ως εξής : Έστω η Alice θέλει να κρυπτογραφήσει μια ακολουθία λέξεων $W_1, W_2 \dots W_l$. Διαισθητικά το scheme δουλεύει υπολογίζοντας

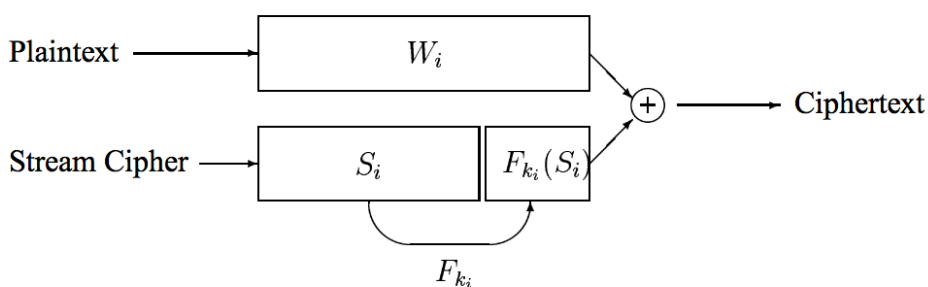
το XOR του plaintext με μία ακολουθία από pseudorandom bits τα οποία έχουν μία συγκεκριμένη δομή. Αυτή η δομή θα επιτρέψει την αναζήτηση χωρίς να αποκαλύπτεται τίποτα παραπάνω για το plaintext.

Πιο συγκεκριμένα το βασικό σχήμα έχει ως εξής : Η Alice παράγει μία ακολουθία από pseudorandom τιμές S_1, S_2, \dots, S_l χρησιμοποιώντας έναν pseudorandom generator. Κάθε S_i μέτρο $n - m$ bits. Για την κρυπτογράφηση μίας n -bits λέξης W_i που εμφανίζεται στη θέση i η Alice παράγει ένα pseudorandom stream T_i ως εξής: $T_i = S_i || F_k(S_i)$ (όπου F_k είναι μια pseudorandom function) και τελικώς παράγει το ciphertext ως $C_i = W_i + T_i$.

Ας παρατηρήσουμε εδώ πως μόνο η Alice μπορεί να παράξει το pseudorandom stream T_i , οπότε κανένας άλλος δεν μπορεί να το αποκρυπτογραφήσει.

Το βασικό αυτό σχήμα υποστηρίζει την αναζήτηση σε κρυπτογραφημένο κείμενο ως εξής: Αν η Alice θέλει να ψάξει τη λέξη W , μπορεί να πει στον Bob τη λέξη W και το k_i που αντιστοιχεί σε κάθε θέση i που μπορεί να εμφανίζεται η λέξη W . Ο Bob τότε μπορεί να ψάξει για τη λέξη W στο κρυπτοκείμενο ελέγχοντας εάν το $C_i \oplus W_i$ είναι της μορφής $s || F_{k_i}(s)$ για κάποιο s . Μία τέτοια αναζήτηση μπορεί να γίνει σε γραμμικό χρόνο. Στις θέσεις που ο Bob δεν γνωρίζει τα k_i δεν μαθαίνει τίποτα για το plaintext.

Παρακάτω βλέπουμε σχηματικά τη λειτουργία του βασικού αυτού σχήματος.



Σχήμα 3.1

Με αυτό το βασικό σχήμα, βέβαια δεν έχουμε καταφέρει και τίποτα σπουδαίο.

Αν η Alice θέλει να βοηθήσει τον Bob να βρει μία λέξη W τότε η Alice είτε πρέπει να αποκαλύψει στον Bob όλα τα k_i (και συνεπώς να δώσει την

ευκαιρία στον Bob να αποκρυπτογραφήσει όλο το κείμενο) ή αλλιώς θα πρέπει να θυμάται όλες τις θέσεις που εμφανίζεται η λέξη W . Παρακάτω θα δούμε το final σχήμα και πως το η extra ‘στρωση’ κρυπτογράφησης θα λύσει αυτά τα προβλήματα.

3.1.2 The Final Scheme

Σε αυτό το σχήμα η Alice αρχικά κρυπτογραφεί με ντετερμινιστικό τρόπο τις λέξεις W ως $X = E(W)$. Στη συνέχεια, χωρίζει την πρώτη αυτή κρυπτογράφηση σε 2 μέρη ως $X_i = L_i || R_i$ (όπου το L_i είναι τα πρώτα $n - m$ bits και το R_i τα τελευταία m bits του X_i). Αντι η Alice να παράξει τα κλειδια k ως $k_i = f_k(E(W_i))$, τα παράγει ως $k_i = f_k(L_i)$.

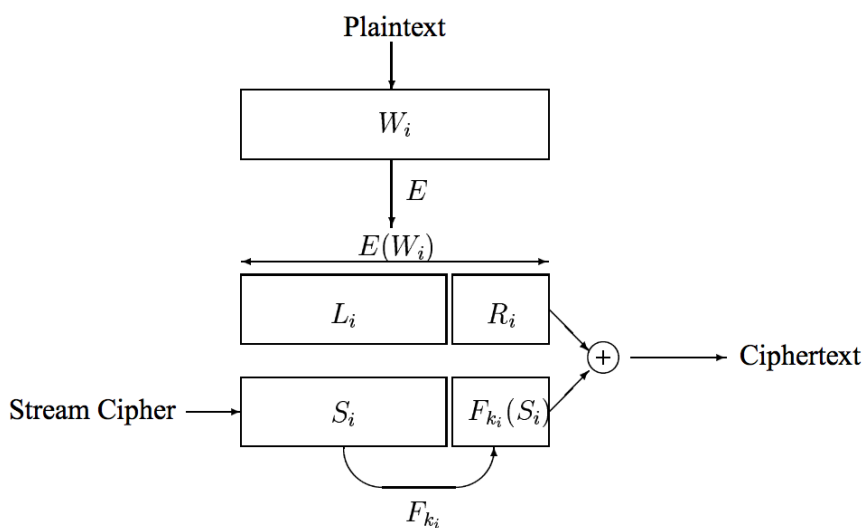
Για την αποκρυπτογράφηση η Alice, παράγει τα S_i από κάποιον pseudorandom generator και με τα S_i μπορεί να ανακτήσει τα L_i κάνοντας XOR το S_i με τα πρώτα $(n - m)$ bits του C_i . Τελικως, με τη γνώση του L_i η Alice μπορεί πλέον να υπολογίσει το k_i και έτσι να τελειώσει την αποκρυπτογράφηση. Η συγκεκριμένη διόρθωση δεν θα ήταν ασφαλής εάν οι λέξεις δεν είχαν κρυπτογραφηθεί αρχικά αφού είναι πολύ πιθανό διαφορετικές λέξεις να έχουν τα ίδια $(n - m)$ πρώτα bits. Η κρυπτογράφηση στο πρώτο στρώμα ουσιαστικά εξαφανίζει αυτό το πρόβλημα, αφού με μεγάλη πιθανότητα όλα τα L_i είναι διαφορετικά μεταξύ τους. (Αν υποθέσουμε πως η αρχική κρυπτογράφηση E είναι μία pseudorandom permutation, τότε σύμφωνα με το παράδοξο των γενεθλίων η πιθανότητα σύγκρουσης είναι πολυ μικρή). Παρακάτω (Σχήμα 3.2) θα δούμε σχηματικά το Final Scheme.

Μετά την λέξη προς λέξη κρυπτογράφηση, η Alice μπορεί να κάνει re-order το ciphertext χρησιμοποιώντας καποιά pseudorandom permutation. Με αυτόν τον τρόπο όταν ο Bob αναζητά μία λέξη, δε θα γνωρίζει σε ποια θέση εμφανίζεται η λέξη στο αρχικό plaintext. Τα προβλήματα του σχήματος αυτού είναι οτι απο τη μία χρησιμοποιεί fixed size keywords και απο την άλλη ότι διαρρέει τη θέση ενός keyword μέσα στο κείμενο. Επιπλέον ο χρόνος αναζήτησης είναι γραμμικός στο συνολικό αριθμό των λέξεων που περιέχονται στα αρχεία.

Αποδοτικότητα : Η πολυπλοκότητα των αλγορίθμων κρυπτογράφησης και αναζήτησης είναι γραμμική στο συνολικό αριθμό των λέξεων ανα κείμενο (η χει-

ρότερη δηλαδή περίπτωση). Για την κρυπτογράφηση, μία κρυπτογράφηση, ένα XOR και δύο pseudo-random συναρτήσεις πρέπει να υπολογιστούν, ανά λέξη ανά αρχείο. Για το search token χρειάζεται μία κρυπτογράφηση και μία pseudo-random συνάρτηση. Η αναζήτηση χρειάζεται ένα XOR και μία pseudorandom συνάρτηση ανά λέξη ανά κείμενο.

Ασφάλεια : Το σχήμα αυτό είναι IND-CPA ασφαλές, υπό την προϋπόθεση ότι οι pseudorandom συναρτήσεις είναι ασφαλείς. Η IND-CPA ασφάλεια δεν περιλαμβάνει ερωτήματα και έτσι δεν είναι ιδιαίτερα σημαντική όσον αφορά το searchable encryption. Επίσης το σχήμα αυτό διαρρέει τις θέσεις των keywords σε ένα αρχείο. Αν πολλά ερωτήματα πραγματοποιηθούν, τότε υπάρχει πιθανότητα να είναι ευάλωτο όλο το αρχείο, απλά με στατιστική ανάλυση.



Σχήμα 3.2

3.2 Forward Index

Σε αυτό το σχήμα για κάθε αρχείο υπάρχει μια κρυπτογραφημένη δομή δεδομένων που χρησιμοποιείται για την αναζήτηση των keywords. Συγκεκριμένα χρησιμοποιείται ένα Bloom Filter. Τα Bloom Filters είναι δομές δεδομένων οι οποίες ελέγχουν εάν ένα στοιχείο ανήκει σε ένα σύνολο. Χρησιμοποιεί ένα

διάνυσμα μήκους n του οποίου αρχικά όλα τα bits είναι 0. Για κάθε λέξη W που θα μπει στο σύνολο, υπολογίζουμε t ανεξάρτητα hashes της λέξης (t διαφορετικές hash functions), όπου κάθε h_i κάνει hash στο σύνολο $\{1, 2, \dots, n\}$ και τα bits $h_i(w)$ γίνονται 1.

Με αυτόν τον τρόπο είναι δυνατό να ελέγξουμε εάν ένα keyword ανήκει στο αρχείο ή όχι απλά ελέγχοντας εάν τα bits που δίνει το $h_i(w)$ είναι 0 ή 1.

Ερησιμοποιώντας ένα bloom filter ανά αρχείο, ο χρόνος αναζήτησης γίνεται γραμμικός στον αριθμό των αρχείων. Ένα σημαντικό πρόβλημα των bloom filters είναι ότι παράγουν false positives. Υπάρχει όμως τρόπος να ξεπεραστεί αυτό το πρόβλημα (σε κάποιο βαθμό τουλάχιστον). Η ιδέα είναι να χρησιμοποιείται μία pseudorandom function δύο φορές και μετά να γίνεται Insert στο bloom filter. Τη δεύτερη φορά, η pseudorandom function, θα παίρνει input, το output της πρώτης pseudorandom function αλλά και ένα unique identifier, κάτι που θα κάνει όλα τα bloom filters να είναι διαφορετικά, ακόμα για αρχεία με τα ίδια keywords.

Αποδοτικότητα : Πρέπει να παραχθεί ένα bloom filter ανά αρχείο. Έτσι ο αλγόριθμος θα είναι γραμμικός στον αριθμό των διακριτών λέξεων ανά κείμενο. Η αναζήτηση μέσω bloom filter είναι λειτουργία σταθερού χρόνου και πρέπει να γίνει για κάθε αρχείο. Έτσι η αναζήτηση είναι ανάλογη του αριθμού των αρχείων και όχι στον αριθμό των λέξεων όπως πριν. Το μέγεθος του index είναι ανάλογο στον αριθμό των διακριτών λέξεων στο αρχείο. Από τη στιγμή μάλιστα, που χρησιμοποιείται ένα bloom filter, οι ασυμπτωτικές σταθερές είναι μικρές (λίγα bits)

Ασφάλεια : Το σχήμα είναι IND-CKA ασφαλές. Η συγκεκριμένη ασφάλεια βέβαια δεν εγγυάται την ασφάλεια των search tokens, δηλαδή δεν μας εγγυάται ότι ο Server δεν μπορεί να ανακτήσει πληροφορίες για τις λέξεις για τις οποίες έχει ερωτηθεί, από τα Search tokens. Ένα από τα μειονεκτήματα των bloom filter είναι ότι ο αριθμός των 1 εξαρτάται από τα entries, σε αυτή την περίπτωση δηλαδή στον αριθμό των διακριτών keywords σε κάθε αρχείο. Σαν συνέπεια, το σχήμα διαρρέει τον αριθμό των Keywords σε κάθε αρχείο.

3.3 Hierarchical Structure of Logarithmic Levels

Το επόμενο σχήμα που θα μελετήσουμε υποστηρίζει dynamic symmetric searchable encryption schemes και χρησιμοποιεί μία ιεραρχική δομή απο λογαριθμικά επίπεδα. Για n ζεύγη απο αρχεία/keywords ο server αποθηκεύει μία δομη δεδομένων που έχει $\log(n + 1)$ επίπεδα. Κάθε επίπεδο m μπορεί να αποθηκεύσει μέχρι και 2^m entries, όπου κάθε entry κρυπτογραφεί τις πληροφορίες για ένα keyword, έναν identifier ενός αρχείου f που περιέχει τη λέξη w , τον τύπο της λειτουργίας (add ή delete), καθώς και τον αριθμό των φορών που εμφανίζεται το keyword w στο επίπεδο m . Το σχήμα αυτό διασφαλίζει οτι σε κάθε επίπεδο αποθηκεύεται μόνο μια λειτουργία (add ή delete) για κάθε ζεύγος αρχείου/keyword.

Για την αναζήτηση, χρησιμοποιείται ένα search token ανά επίπεδο. Σε αυτό το σχήμα κάθε ανανέωση (προσθήκη ή διαγραφή) προκαλεί ανασύσταση των επιπέδων της δομής.

Αυτό το σχήμα έχει σχετικά μικρή διαφορά σχετικά με άλλα sse schemes και επιπλέον τόσο η αναζήτηση όσο και η ανανέωση της βάσης γίνονται σε ημι-γραμμικό χρόνο. Σε αντίθεση με άλλα σχήματα προσφέρει μία καλύτερη αίσθηση ασφάλειας. Ένα search token που έχει χρησιμοποιηθεί στο παρελθόν δεν μπορεί να χρησιμοποιηθεί ξανά για τα αρχεία που προστίθενται στη βάση. Αυτό συμβαίνει επειδή κάθε φορά που γίνεται ανασύσταση ενός επιπέδου, ένα νέο κλειδί χρησιμοποιείται για την κρυπτογράφηση των entries σε αυτό το επίπεδο.

3.4 Inverted Index

Όπως αναφέρθηκε και στην εισαγωγή η ιδέα του αντίστροφου δείκτη είναι optimal

Η βασική ιδέα εδώ είναι να χρησιμοποιηθεί ένας δείκτης για κάθε keyword αντί για κάθε αρχείο. Το αποτέλεσμα είναι ο χρόνος αναζήτησης να περιορίζεται από γραμμικός στον αριθμό των αρχείων σε γραμμικό στον αριθμό των αρχείων που περιέχουν την προς αναζήτηση λέξη, που είναι optimal.

Για κάθε keyword w , υπάρχει μια linked list L_w που περιέχει identifiers για

τα αρχεία που περιέχουν το w . Οι linked lists δεν μπορούν να είναι σε plaintext, αφού έτσι θα υπήρχε διαρροή πληροφορίας. Οπότε, όλοι οι κόμβοι της linked list αποθηκεύονται σε διάνυσμα A , μη ταξινομημένοι και κρυπτογραφημένοι. Το plaintext κάθε κόμβου περιέχει τρία πράγματα :

- (i) Identifier ενός αρχείου.
- (ii) Το encryption key του επόμενου κόμβου.
- (iii) Έναν Pointer στον επόμενο κόμβο.

Για την αναζήτηση λοιπόν μίας λέξης w , χρειαζόμαστε το encryption key του πρώτου κόμβου της L_w και έναν pointer στη θέση του στο A . Όλες αυτές οι πληροφορίες αποθηκεύονται κρυπτογραφημένες σε μία pseudorandom position σε ένα look up table T . Ο χρήστης, δηλαδή, παράγει τα A και T με βάση τη συλλογή των plaintext αρχείων που έχει, και τα αποθηκεύει στον server μαζί με τα encrypted αρχεία.

Το search token αποτελείται από τη θέση στο T για το keyword w μαζί με το κλειδί που χρησιμοποιήθηκε για το encryption της εισαγωγής στο T .

Ένα μεγάλο μειονέκτημα ενός τέτοιου σχήματος είναι ότι δεν είναι παντα εφικτό να είναι δυναμικά. Τα διανύσματα θα πρέπει να ανανεώνονται σε κάθε add ή delete. Επίσης, μίας και οι encrypted indexes που χρησιμοποιούνται αποθηκεύουν δεδομένα σε random θέσεις και η θέση του επόμενου κομβου μαθαίνεται μόνο αφού τα δεδομένα ενός κόμβου έχουν διαβαστεί, οι λειτουργίες δεν μπορούν να γίνουν παράλληλες.

Παρακάτω θα δούμε έναν τρόπο για να ξεπεράσουμε το πρόβλημα της δυναμικότητας.

Ας δούμε όμως πρώτα ένα παράδειγμα για να δούμε πως δουλεύει το σύστημα με τους inverted indexes.

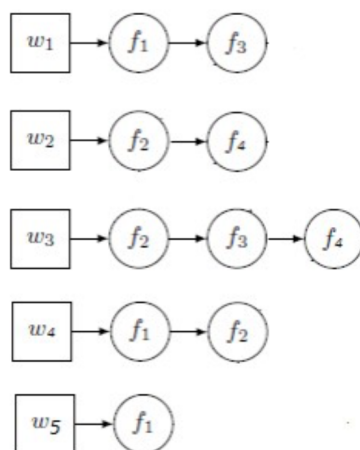
Εστω ότι έχουμε 4 αρχεία $f = (f_1, f_2, f_3, f_4)$ και 5 keywords $w = (w_1, w_2, w_3, w_4, w_5)$. Το κάθε file περιέχει τα εξής keywords:

$$f_1\{w_1, w_4, w_5\}$$

$$f_2\{w_2, w_3, w_4\}$$

$$f_3\{w_1, w_3\}$$

$$f_4\{w_2, w_3\}$$



Σχήμα 3.3

Έστω τώρα ότι ψάχνουμε για ένα keyword, έστω το w_4 , ο server θα μάθει τη θέση του πρώτου tuple που περιέχει το keyword, (που περιέχει το identifier του f_1) και στη συνέχεια θα ανακαλύψει και το δεύτερο tuple που περιέχει το identifier του f_2 .

Αποδοτικότητα : Το σχήμα αυτό είναι το πρώτο σχήμα με βέλτιστο χρόνο αναζήτησης. Η παραγωγή των index είναι γραμμική στον αριθμό των διακριτών λέξεων ανά κείμενο. Οι υπολογισμοί του server ανά αναζήτηση είναι ανάλογοι με τον αριθμό των αρχείων που περιέχουν ένα keyword. Οι δομές δεδομένων που χρησιμοποιεί αυτό το σχήμα περιορίζουν το look-up time σε $O(1)$. Η ανανέωση της βάσης είναι δύσκολη εξαιτίας της αναπαράστασης των δεδομένων. Έτσι, το σχήμα αυτό είναι πιο αποδοτικό για στατικές βάσεις δεδομένων.

Ασφάλεια : Το σχήμα είναι συνεπές με την IND-CKA1 ασφάλεια.

3.4.1 Επιτυχάνοντας Δυναμικότητα

Το βασικό πρόβλημα στην προηγούμενη προσέγγιση είναι ότι όταν ένα αρχείο προστίθεται ή αφαιρείται από τη βάση δεδομένων, οι κόμβοι στο διάγραμμα A

πρέπει να ανανεωθούν. Πιο συγκεκριμένα όταν ένα αρχείο f αφαιρείται από τη βάση, οι κόμβοι του A που αναφέρονται σε αυτό το αρχείο πρέπει να ‘καθαριστούν’. Επιπλέον, όταν ένα αρχείο προστίθεται στη βάση πρέπει κάποιοι δείκτες στη linked list να ανανεωθούν (οι οποίοι, όμως, είναι κρυπτογραφημένοι).

Για να ξεπεραστεί αυτό το πρόβλημα θα γίνει χρήση ενός deletion array και θα συμπληρώσουμε τα εξής βήματα στο προηγούμενο construction.

- (i) Ένα deletion array θα παρακολουθεί τις θέσεις του search array A στις οποίες οι κόμβοι θα πρέπει να τροποποιηθούν σε περίπτωση διαγραφής ενός αρχείου f . Η πρόσβαση σε αυτό το array μπορεί να γίνει μέσω ενός token που παράγεται από το χρήστη.
- (ii) Υπάρχει μία επιπλέον λίστα από άδειους κόμβους που ‘παρακολουθεί’ τις άδειες θέσεις στο διάνυσμα αναζήτησης A . Αυτή η λίστα μπορεί να χρησιμοποιηθεί από το server σε περίπτωση που θέλουμε να προσθέσουμε ένα αρχείο f στη βάση δεδομένων μας.
- (iii) Οι δείκτες κρυπτογραφούνται με ομομορφική κρυπτογράφηση προκειμένου να επιτρέπονται αλλαγές στο κρυπτοκείμενο χωρίς να είναι αναγκαία η αποκρυπτογράφηση. Πιο συγκεκριμένα, η κρυπτογράφηση γίνεται κάποιοντας XOR το μήνυμα με το αποτέλεσμα μία pseudorandom function.

Το μειονέκτημα πλέον του σχήματος αυτού είναι το τι διαρρέει κατά τη διαδικασία πρόσθεσης και αφαίρεσης αρχείων. Η leakage function που συνδέεται με τις λειτουργίες add και delete διαρρέει αρκετή πληροφορία.

Πιο συγκεκριμένα, διαρρέει τα search tokens που αντιστοιχούν στα keywords που περιέχονται στα added/deleted αρχεία.

Αποδοτικότητα : Το σχήμα επιτυγχάνει optimal χρόνο αναζήτησης ενώ παράλληλα χειρίζεται με ευκολία ανανεώσεις στη βάση δεδομένων. Η παραγωγή των index χρειάζονται οχτώ pseudorandom functions ανά keyword. Για την αναζήτηση, ο server χρησιμοποιεί ένα look up table για τον πρώτο κόμβο και στη συνέχεια εκτελούνται XOR operations ανά κόμβο. Κάθε κόμβος αντιπροσωπεύει ένα αρχείο που περιέχει το προς αναζήτηση keyword

Ασφάλεια : Το σχήμα είναι IND-CKA2 ασφαλές. Οι ανανεώσεις της βάσης διαρρέουν πληροφορίες για τα Search tokens των keywords σε ανανεωμένα αρχεία. Η ασφάλεια του σχήματος αποδεικνύεται στο random oracle model.

3.5 Keyword Red-Black Tree

Η επόμενη μέθοδος που θα μελετήσουμε είναι πιο εναλλακτική και χρησιμοποιεί μία διαφορετική δομή δεδομένων. Είναι μία δομή αρκετά παρεμφερής με τα red-black trees και για αυτό και ονομάστηκε keyword red black tree. Το keyword red-black tree κρυπτογραφείται με pseudorandom functions και pseudorandom permutations και το random oracle. Το τελικό σχήμα έχει ασυμπτωτικά την ίδια αποδοτικότητα με ένα unencrypted red-black tree.

Το keyword red-black tree είναι μία δυαδική tree-based δομή δεδομένων. Υποθέτουμε ότι το universe των keywords είναι σταθερό (έστω m σε σύνολο) και πολύ μικρότερο από τον αριθμό των αρχείων (ο οποίος μπορεί να μεγαλώσει). Επιπλέον θεωρούμε και μία ταξινόμηση των αρχείων $f = (f_1, f_2, \dots, f_n)$ με βάση τους identifiers τους. Στα φύλλα του δέντρου, έχουμε pointers για το κάθε αρχείο. Σε κάθε εσωτερικό κόμβο u του δέντρου αποθηκεύουμε ένα διάνυσμα από m bits, $d_u = d_{u,1} \dots d_{u,m}$ όπου το κάθε $d_{u,i}$ αντιστοιχεί στο i -οστό keyword w_i του universe. Το bit $d_{u,i}$ είναι 1, αν και μόνο αν, ένα από τα αρχεία που συσχετίζονται με τα παιδιά του u , περιέχει το keyword w_i . Αυτό είναι σχετικά εύκολο να υπολογιστεί ξεκινώντας από τα φύλλα του δέντρου και μετά για τους εσωτερικούς κόμβους υπολογίζουμε το d_i ως το bitwise OR των τιμών των δύο παιδιών του.

Για να ψάξουμε για ένα keyword w_i , απλά ξεκινάμε από τη ρίζα και συνεχίζουμε αναδρομικά μέχρις ότου είτε φτάσουμε σε κόμβο στον οποίο $d_{u,i} = 0$ (δεν υπάρχει δηλαδή αρχείο που να σχετίζεται με τα παιδιά του κόμβου και να περιέχει το w_i), ή φτάσουμε σε ένα φύλλο για το οποίο το συσχετισμένο αρχείο περιέχει το w_i .

Ένας από τους λόγους που αυτό το σχήμα είναι χρήσιμο, είναι ότι υποστηρίζει και keyword based operations (αν ακολουθήσουμε μονοπάτι από τη ρίζα προς τα φύλλα) αλλά και file based operations (αν ακολουθήσουμε μονοπάτι από τα φύλλα προς τη ρίζα). Αυτό διευκολύνει πολύ την ανανέωση της βάσης

δεδομένων.

Η ιδέα για την κρυπτογράφηση της δομής είναι η εξής :

Για κάθε keyword w_i υπάρχει ένα διακριτο key το οποίο χρησιμοποιείται για την κρυπτογράφηση των bits $d_{u,i}$ (για όλα τα u). Το encrypted bit $d_{u,i}$ αποθηκεύεται τότε σε ένα hash table (που συνδέεται με τον κόμβο u) σε μία pseudorandom position. Ένα άλλο hash table θα έχει μία random τιμή στη σχετική θέση. Η επιλογή του hash table στο οποίο θα αποθηκευτεί το bit $d_{u,i}$ εξαρτάται από το output ενός random oracle. Για να ανανεώσουμε τη βάση, ο server πραγματοποιεί ένα structure update στο keyword red-black tree το οποίο περιλαμβάνει και τα απαραίτητα rotations που πραγματοποιούνται κατά το update ενός red-black tree (με σκοπό να διατηρήσουμε λογαριθμικό ύψος). Για να πραγματοποιηθεί αυτή η λειτουργία χρειαζόμαστε μόνο το file identifier. Ο server τότε στέλνει στον χρήστη το κομμάτι του δέντρου που χρειάζεται ανανέωση, και ο χρήστης απαντάει με ένα add/delete token που επιτρέπει στον server να κάνει το απαιτούμενο update των τιμών σε αυτές τις θέσεις.

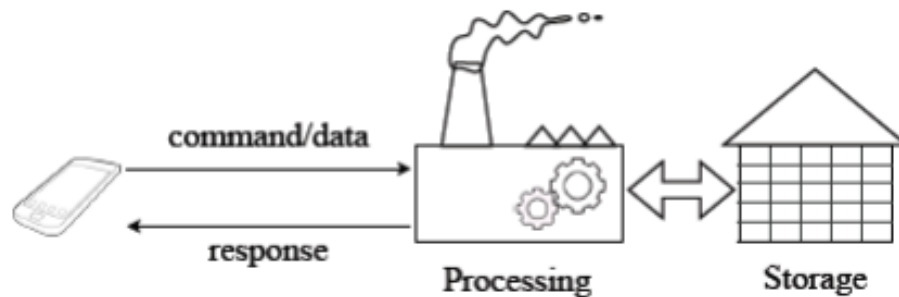
Τα updates της βάσης δεν διαρρέουν έξτρα πληροφορίες εκτός από αυτές που μπορούν να συναχθούν από προηγούμενα search tokens. Επιπλέον, τα updates της βάσης μπορούν να εκτελεστούν αρκετά αποδοτικά, αφού όλη η πληροφορία για ένα αρχείο f μπορεί να βρεθεί και να ανανεωθεί σε χρόνο $O(\log|f|)$ αλλά χρειάζεται ενάμιση γύρο διαδραστικότητας. Ο συνολικός χρόνος αναζήτησης εάν χρησιμοποιηθεί ένας αρκετά μεγάλος αριθμός επεξεργαστών είναι $O(\log|f|)$. Το μεγάλο μειονέκτημα του σχήματος αυτού είναι ότι η δομή δεδομένων που χρησιμοποιείται έχει μέγεθος $O(m|f|)$ και συνήθως οι σταθερές είναι αρκετά μεγάλες.

Αποδοτικότητα : Αρχικά χρειάζονται $2n-1$ κρυπτογραφήσεις (όσοι είναι δηλαδή και οι κόμβοι του δέντρου), όπου n είναι ο αριθμός των αρχείων. Για την αναζήτηση χρειάζονται $D(w)\log n$ αποκρυπτογραφήσεις, όπου με D συμβολίζουμε τα αρχεία που περιέχουν το προς αναζήτηση keyword

Ασφάλεια : Το σχήμα έχει αποδειχθεί IND-CKA2 ασφαλές στο Random oracle model.

3.6 Blind Storage

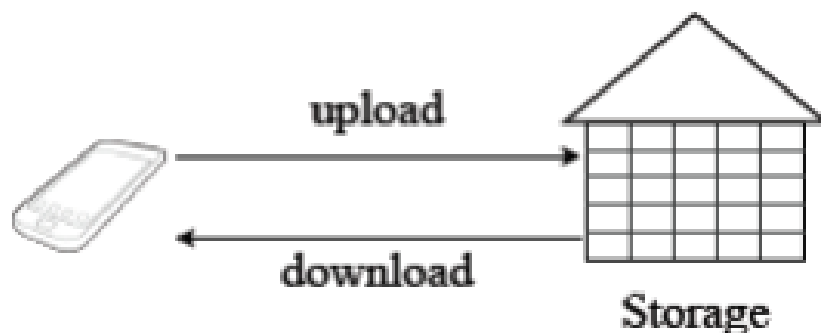
Το σχήμα που θα δούμε σε αυτήν την ενότητα επιτρέπει στον χρήστη να αποθηκεύει μία συλλογή από αρχεία στο server με τέτοιον τρόπο, όπου όλες οι πληροφορίες για αυτά είναι μυστικές μέχρι να γίνουν accessed. Ακόμα και πληροφορίες όπως ο αριθμός των αρχείων ή το μέγεθος τους. Όταν ένα αρχείο γίνει accessed, ο server μαθαίνει την ύπαρξη του και το μέγεθος του, αλλά ούτε το όνομα του, ούτε το περιεχόμενο. Ο server έχει επίσης τη δυνατότητα να παρατηρεί αν το ίδιο αρχείο έχει γίνει accessed πολλές φορές. Η λειτουργία των symmetric searchable encryptions schemes που έχουμε δει μέχρι τώρα σε αντιπαράβολή με αυτήν την τυπική blind storage φαίνεται στα παρακάτω σχήματα :



Σχήμα 3.4

Το βασικό σχήμα Blind Storage, που ονομάζεται SCATTERSTORE, κατασκευάζεται με τη χρήση μίας απλής αλλά ισχυρής τεχνική. Κάθε αρχείο αποθηκεύεται ως μια συλλογή μπλοκ που διατηρούνται σε pseudorandom positions. Στο Σχήμα 3.6 μπορείτε να δείτε πώς μετατρέπεται ένα αρχείο σε μια συλλογή μπλοκ. Ο διακομιστής βλέπει μόνο ένα υπερ-σύνολο των θέσεων όπου βρίσκονται τα μπλοκ του αρχείου και όχι το ακριβές σύνολο τοποθεσιών. Η βασική έννοια ασφαλείας από την σκοπιά του διακομιστή, είναι ότι κάθε αρχείο συσχετίζεται με ένα σύνολο από τοποθεσίες οι οποίες είναι ανεξάρτητες από τα άλλα αρχεία του συστήματος. Από την άλλη πλευρά, τα σύνολα των τοποθεσιών για δύο αρχεία μπορούν να επικαλύπτονται.

Υπάρχει ένα άνω φράγμα N στον αριθμό των μπλοκ που μπορούν να αποθηκευτούν. Δεδομένου ενός αρχείου f με n blocks, an τοποθεσίες στο set



Σχήμα 3.5

$\{1, \dots, N\}$ διαλέγονται με τη χρήση ενός pseudorandom number generator και τα n μπλοκ του f αποθηκεύονται σε αυτές τις θέσεις. Ο λογος που διαλέξαμε an και όχι n blocks για την αποθήκευση του f είναι οτι πιθανώς να υπάρχουν συγκρούσεις με τις θέσεις άλλων αρχείων. Επομένως οι an τοποθεσίες που επιστρέφονται απο τον server για να κάνουμε access το f διαλέγονται ανεξάρτητα απο των άλλων αρχείων, και έτσι η f αποθηκεύεται encrypted σε n απο αυτές τις θέσεις. Ένα θέμα είναι οτι ο χρήστης πρέπει να γνωρίζει τον αριθμό των μπλοκ του αρχείου f .

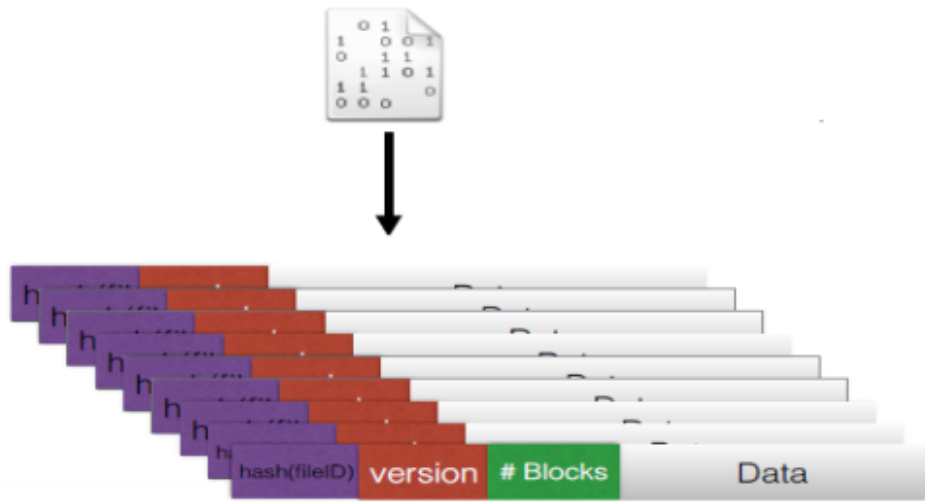
Η ιδέα για να φτιάξουμε το sse σχήμα απο το blind storage είναι να αποθηκεύουμε για όλα τα keywords, τους search index entries (which lists όλα τα αρχεία που περιέχουν το keyword) σαν ξεχωριστά αρχεία στο blind storage.

Η θετική πλευρά, σε αυτό το σχήμα είναι οτι ο διακομιστής δεν χρειάζεται να πραγματοποιήσει κανένα υπολογισμό, αλλά μόνο να παρέχει interfaces για τη μεταφόρτωση και τη λήψη αρχείων, γεγονός που καθιστά το σύστημα διαφανές για χρήση σε περιβάλλον cloud. Επιπλέον η ασφάλεια του σχήματος αυτού έγκειται στο γεγονός οτι ο server δεν κάνει ούτε ένα decrypt.

Η αρνητική πλευρά είναι οτι δεν παρέχεται το ίδιο ισχυρό security στα αρχικά files και σε αυτά που προστέθηκαν αργότερα ενώ, οι ανανεώσεις διαρρέουν μία deterministic συνάρτηση των keywords με συνέπεια οι εγγυήσεις ασφάλειας να είναι πολυ μικρότερες για αρχεία που δεν ήταν απο την αρχή στη βάση. Αυτό

32 · ΣΧΗΜΑΤΑ ΠΟΥ ΕΠΙΤΡΕΠΟΥΝ ΤΗΝ ΚΡΥΠΤΟΓΡΑΦΗΣΗ ΜΕ ΔΥΝΑΤΟΤΗΤΑ ΑΝΑΖΗΤΗΣΗΣ.

είναι ιδιαίτερα ανησυχητικό για βάσεις δεδομένων που ξεκινάνε από άδειες και γεμίζουν σιγά σιγά, που είναι και η συνηθέστερη περίπτωση.



Σχήμα 3.6

Κεφάλαιο 4

Περί Ιδιωτικότητας (Privacy Issues)

4.1 Η ιδιωτικότητα σε static sse

Δυστυχώς, πρέπει να γίνουν αρκετοί συμβιβασμοί ούτως ώστε να επιτύχουμε λειτουργικότητα σε ένα symmetric searchable encryption σχήμα. Το καλύτερο σενάριο, όσον αφορά τη διαρροή πληροφορίας σε ένα SSE θα ήταν να διαρρέεται μόνο το αποτέλεσμα της αναζήτησης, (οι identifiers των αρχείων που περιέχουν το keyword που αναζητήθηκε) δηλαδή το access pattern. Το να προσπαθήσουμε να αποκρύψουμε το access pattern οδηγεί σε πολύ ακριβές λύσεις, οι οποίες δεν είναι ιδιαίτερα αποδοτικές.

Όμως το access pattern δεν είναι η μοναδική διαρροή που έχουμε. Οι πιο αποδοτικές προσεγγίσεις μέχρι σήμερα χρησιμοποιούν ντετερμινιστικά token αναζήτησης (search tokens), πράγμα το οποίο διαρρέει και το search pattern. (Δηλαδή αν δύο ερωτήσεις ήταν για το ίδιο keyword ή όχι).

Εκτός από το access pattern και το search pattern πολλά sse σχήματα διαρρέουν ακόμα πιο πολλά πράγματα όπως για παράδειγμα τον αριθμό των αρχείων που υπάρχουν στη βάση δεδομένων, τον αριθμό των keywords που έχουν χρησιμοποιηθεί, ακόμα και τον αριθμό ζευγαρίων αρχείο-keyword. Βέβαια, όλα αυτά είναι προβλήματα με τα οποία πρέπει να συμβιβαστούμε καθώς αυτή θεωρείται η καλή περίπτωση καθώς το είδος της πληροφορίας που διαρρέεται δεν θεωρείται

ιδιαίτερα σημαντικό.

4.2 Η ιδιωτικότητα σε dynamic sse

Στα dynamic symmetric searchable encryption schemes το πρόβλημα γίνεται ακόμα μεγαλύτερο καθώς πέρα από τη διαρροή των static sse, η οποία προφανώς υπάρχει και έδω, διαρρέουν ακόμα πιο πολλά πράγματα κατά τη διάρκεια της πρόσθεσης και της αφαίρεσης αρχείων στη βάση δεδομένων. Αυτό σημαίνει ότι τα δυναμικά sse σχήματα είναι ακατάλληλα για βάσεις δεδομένων όπου τα αρχεία προστίθενται σταδιακά, που είναι και η συνηθέστερη περίπτωση. Αν τα deterministic search tokens συνεχίσουν να είναι valid στο μέλλον (κάτι που συμβαίνει στα πιο πολλά σχήματα), τότε ένας κακόβουλος διακομιστής ή διαχειριστής θα μπορεί να ψάχνει αν στα προστιθέμενα files υπάρχει κάποιο keyword, το οποίο είχε αναζητηθεί στο παρελθόν, και δεν μπορεί κανείς να κάνει κάτι για αυτό.

Γενικότερα, ένα searchable encryption σχήμα θα διαρρέει πληροφορίες, οι οποίες μπορούν να διαιρεθούν σε τρεις κατηγορίες. Index information, search pattern και access pattern.

- (i) Οι πληροφορίες για τον index αναφέρονται σε πληροφορίες για τα keywords που περιέχονται στον index. Αυτού του είδους η πληροφορία διαρρέεται από τα αποθηκευμένα στο server ciphertexts και indexes. Οι πληροφορίες αυτές περιέχουν στοιχεία όπως τον αριθμό των keywords ανά αρχείο, τον αριθμό των αρχείων, το μέγεθος των αρχείων, τους identifiers των αρχείων καθώς και την ομοιότητα των κειμένων.
- (ii) Το search pattern αναφέρεται σε πληροφορίες που μπορούν να εξαχθούν με την ακόλουθη έννοια: Δεδομένου ότι δύο αναζητήσεις επιστρέψουν το ίδιο αποτέλεσμα, καθόρισε αν οι δύο αναζητήσεις ήταν για το ίδιο keyword. Η χρήση ντετερμινιστικών search tokens διαρρέει άμεσα το search pattern. Η πρόσβαση στο search pattern επιτρέπει στον server να κάνει στατιστική ανάλυση και συνεπώς να καθορίσει πληροφορίες για τα keywords.

(iii) Το access pattern αναφέρεται στην πληροφορία που υπονοείται από τα αποτελέσματα των ερωτημάτων. Για παράδειγμα εάν μία ερώτηση επιστρέφει ένα x αρχείο, ενώ μία άλλη επιστρέφει το x και άλλα δύο αρχεία, τότε ο server καταλαβαίνει ότι το πρώτο ερώτημα ήταν πιο restrictive από το δεύτερο

Η γενική παραδοχή είναι ότι δε πρέπει να διαρρέεται τίποτε από τα αποθηκευμένα αρχεία και τους indexes, εκτός από το αποτέλεσμα και το pattern της αναζήτησης. Τα searchable encryption σχήματα δεν πρέπει να διαρρέουν τα keywords σε plaintext μορφή. Όλα τα σχήματα που συζητήθηκαν διαρρέουν τουλάχιστον τα search και access patterns.

Κεφάλαιο 5

Αποδοτικότητα

Υπάρχουν αρκετές παράμετροι που πρέπει να ελεγχθούν ουτως ώστε ένα sse να είναι αποδοτικό. Στο παρόν κείμενο δεν γίνεται αναφορά σε παραμέτρους όπως το software και το hardware ενός υπολογιστή.

Αρχικά, το σημαντικότερο όλων είναι η πολυπλοκότητα του χρόνου αναζήτησης. Χρόνος γραμμικός στον αριθμό των κειμένων δεν είναι καθόλου αποδοτικός μίας και συνήθως μιλάμε για βάσεις οι οποίες έχουν πολυ μεγάλο αριθμό αρχείων. Ο optimal χρόνος αναζήτησης είναι ο ημιγραμμικός, όπου το search περιορίζεται στον αριθμό των αρχείων που περιέχουν το keyword που αναζητείται.

Στη συνέχεια μία ακόμη βασική παράμετρος που πρέπει να ελεγχθεί είναι το κατα πόσον οι λειτουργίες μπορούν να γίνουν παράλληλες. Σχήματα που υποστηρίζουν αυτή την παράμετρο είναι ιδανικά για χρήση στο cloud.

Επιπλέον το μέγεθος των δομών δεδομένων που θα χρησιμοποιούνται απο τον server, αλλά και από τον χρήστη, πρέπει να είναι όσο το δυνατόν μικρότερο γίνεται.

Τέλος η αλληλεπίδραση μεταξύ του χρήστη και του server πρέπει να είναι μικρή για να περιορίζεται οσον το δυνατό περισσότερο ούτως ώστε να ελαχιστοποιείται το network delay.

Γενικότερα,κάποια σχήματα κρυπτογραφούν απευθείας το plaintext με τέτοιο τρόπο όπου το προκύπτον κρυπτοκείμενο να μπορεί να ερωτηθεί για keywords. Αυτό έχει ως αποτέλεσμα να έχουμε χρόνο αναζήτησης γραμμικό στον

μέγεθος των δεδομένων που είναι αποθηκευμένα στον server. Αν δηλαδή για παράδειγμα έχουμε n αρχεία με w keywords τότε η πολυπλοκότητα είναι γραμμική στον αριθμό των keywords ανά αρχείο, $O(nw)$, αφού για κάθε keyword πρέπει να βρεθεί ένα ταίριασμα.

Από την άλλη, άλλα σχήματα, προκειμένου να επιταχύνουν τη διαδικασία χρησιμοποιούν ένα συνηθισμένο εργαλείο των βάσεων δεδομένων, έναν index, ο οποίος παράγεται με βάση τα plaintext αρχεία. Η χρήση ενός index μπορεί να ελατώσει σημαντικά τη πολυπλοκότητα αναζήτησης και συνεπώς να αυξήσει την αποδοτικότητα της αναζήτησης. Η αυξημένη απόδοση αναζήτησης έρχεται με το κόστος ενός προπαρασκευαστικού βήματος. Από τη στιγμή που ο index κατασκευάζεται με βάση τα plaintext αρχεία, η παραγωγή του δεν είναι πάντοτε εφικτή και εξαρτάται σε μεγάλο βαθμό από τα αρχεία προς κρυπτογράφηση. Οι δύο κύριες προσεγγίσεις για την κατασκευή ενός index είναι οι επόμενες.

- (i) Ένας forward index είναι ένας Index ανά αρχείο και όπως είναι φυσικό περιορίζει τον χρόνο αναζήτησης σε $O(n)$. Αυτό οφείλεται στο γεγονός ότι ένας index ανά έγγραφο πρέπει να υποβληθεί σε επεξεργασία κατά τη διάρκεια ενός ερωτήματος.
- (ii) Αυτή τη στιγμή, η συνηθέστερη μέθοδος για την επίτευξη sublinear search time είναι η χρήση ενός ανεστραμμένου index, το οποίο είναι ένας δείκτης ανά λέξη-κλειδί στη βάση δεδομένων. Ανάλογα με το πόση πληροφορία είμαστε διατεθειμένοι να αφήσουμε να διαρρεύσει, η χρονική πολυπλοκότητα μπορεί να μειωθεί σε $O(\log w')$ (πχ με χρήση ενός hash tree) ή ακόμα και $O(|D(w)|)$ (που είναι και η optimal case), όπου $\Delta(w)$ είναι ο αριθμός των αρχείων που περιέχουν το keyword w .

Όλα τα σχήματα που έχουμε δει κάνουν χρήση index εκτός από το two layered encryption scheme, το οποίο είναι το μοναδικό σχήμα που κρυπτογραφεί τα αρχεία με τέτοιο τρόπο όπου το προκύπτον κρυπτοκείμενο είναι απευθείας searchable και decryptable

Κεφάλαιο 6

Γενικό μοντέλο

κρυπτογράφησης με

δυνατότητα αναζήτησης, με

κρυπτογραφία δημόσιου

κλειδιού Public Key encryption

with keyword search-PEKS

Εδώ θα δούμε ένα άλλο παράδειγμα το οποίο θα βοηθήσει στην καλύτερη κατανόηση του ζητήματος.

Υποθέτουμε ότι η Alice έχει στην κατοχή της διάφορες συσκευές (laptop, desktop, κινητό) στις οποίες μπορεί να διαβάσει τα e-mail της. Ας κάνουμε μία επιπλέον υπόθεση θεωρώντας ότι η Alice είναι ένα πολύ κοινωνικό άτομο και οι φίλοι και οι συνάδελφοι της την 'σπαμάρουν' συνεχώς με e-mail. Η Alice για να αποφύγει αυτό το spam, θέλει στο κινητό της (που το έχει πάντα μαζί της) να της πηγαίνουν μόνο τα σημαντικά e-mail τα οποία και πρέπει να διαβάσει αμέσως. Ενώ, τα υπόλοιπα e-mail να πηγαίνουν μόνο στο desktop της για να τα διαβάσει αργότερα. Για παράδειγμα, εάν ο Bob θέλει να στείλει στην Alice ένα e-mail με το keyword 'επείγον' αυτό να πάει απευθείας στο κινητό της, ενώ

αν της στείλει "lunch", να πάει στο desktop της για να το διαβάσει αργότερα. Καταλαβαίνουμε λοιπόν ότι κάθε e-mail θα πρέπει να περιέχει ένα μικρό αριθμό από keywords.

Τώρα ας υποθέσουμε ότι ο Bob, ο οποίος δεν είναι πολύ αφελής, στέλνει στην Alice ένα κρυπτογραφημένο e-mail χρησιμοποιώντας το δημόσιο κλειδί της Alice. Τόσο το περιεχόμενο του e-mail, όσο και τα keywords είναι encrypted. Σε αυτήν την περίπτωση ο mail server δεν μπορεί να αποφασίσει πως θα κάνει route το e-mail. Στόχος τώρα είναι λοιπόν να δώσουμε στον provider τη δυνατότητα να ελέγξει εάν η λέξη 'επείγον' είναι keyword στο e-mail ή όχι, χωρίς όμως ο provider να μάθει τίποτε άλλο για το e-mail.

Για να το πετύχουμε αυτό ο Bob αρχικά κρυπτογραφεί το e-mail του χρησιμοποιώντας κάποιο public key system και στη συνέχεια προσθέτει στο κρυπτοκείμενο του ένα PEKS για κάθε keyword W .

Ο Bob δηλαδή στέλνει:

$$E_{A_{pub}}(M) || (PEKS(A_{pub}, W_1) || \dots || PEKS(A_{pub}, W_n))$$

Όπου :

M : Plaintext κείμενο

W_1, \dots, W_n : Τα keywords

A_{pub} : Το δημόσιο κλειδί της Alice.

Η ιδέα είναι ότι με αυτή την κρυπτογράφηση η Alice δίνει εάν trapdoor T_w για το keyword w και αυτός ελέγχει για το w .

Δηλαδή, δεδομένου του PEKS (A_{pub}, W') και του T_w ο provider μπορεί να ελέγξει αν $W = W'$. Αν δεν ισχύει η ισότητα ο provider δεν μαθαίνει τίποτα παραπάνω για το e-mail.

Ας ορίσουμε τώρα το PEKS με βάση τα παραπάνω.

Ορισμός 6.0.1. (PEKS) Ένα σχήμα κρυπτογράφησης δημόσιου κλειδιού αποτελείται από μία τετράδα αλγορίθμων πολυωνυμικού χρόνου $(KeyGen, S = Peks(A_{pub}, W), T_w = Trapdoor(A_{priv}, W), Test(A_{pub}, S, T_w))$ όπου:

- (i) $KeyGen(s)$: Δέχεται σαν input μία παράμετρο s και δίνει output ένα ζευγάρι Δημόσιου και Ιδιωτικού κλειδιού A_{pub}/A_{priv} .

- (ii) $S = PEKS(A_{pub}, W)$: Δέχεται σαν *input* ένα *public key* και μία λέξη W και σαν *output* δίνει *searchable encryption* του W .
- (iii) $T_w = Trapdoor(A_{priv}, W)$: Δέχεται σαν *input* ένα ιδιωτικό κλειδί και μία λέξη W και δίνει *output* ένα *trapdoor* T_w του W .
- (iv) $Test(A_{pub}, S, T_w)$: Δέχεται *input* ένα δημόσιο κλειδί, ένα *searchable encryption* του W' και ένα *trapdoor* για το W . Για *output*, απαντάει ?YES? αν $W' = W$ και ?NO? αλλιώς.

Με άλλα λόγια, η Alice τρέχει τον *KeyGen* και παράγει το ζεύγος των κλειδίων της. Χρησιμοποιεί τον *trapdoor* για να δημιουργήσει *trapdoors* T_w για τα *keywords* W που θέλει, και ο *Mail server* τρέχει τον *test* με *input* τα *trapdoors* για να αποφασίσει εάν ένα *e-mail* περιέχει τα *keywords* που ορίστηκαν από την Alice.

Παρακάτω θα δώσουμε έναν ορισμό για την ασφάλεια του *PEKS* με βάση πάλι ένα παιχνίδι. Πρέπει να είμαστε σίγουροι ότι το $PEKS(A_{pub}, W)$ δεν διαρρέει καμία πληροφορία για το W χωρίς να γνωρίζουμε το T_w . Θα ορίσουμε την ασφάλεια με βάση ένα παιχνίδι, όπου ο *attacker* A θα γνωρίζει *trapdoors* T_w για λέξεις W της επιλογής του. Ακόμα και σε μία τέτοια περίπτωση ο *attacker* A δεν θα πρέπει να μπορεί να ξεχωρίσει το *encryption* ενός *keyword* W_0 από το *encryption* ενός *keyword* W_1 για την οποία δεν γνωρίζει το *trapdoor*.

Ας δούμε το *PEKS security game* μεταξύ ενός *challenger* και ενός *attacker*:

- (i) Ο *challenger* τρέχει τον $KeyGen(s)$, παράγει το ζεύγος ιδιωτικού και δημόσιου κλειδιού, και δίνει το δημόσιο κλειδί στον *attacker*.
- (ii) Ο *attacker* έχει το δικαίωμα να ζητάει *trapdoors* T_w για κάθε *keyword* W που θέλει.
- (iii) Κάποια στιγμή, ο *attacker* στέλνει στον *challenger* 2 λέξεις W_0 και W_1 στις οποίες θέλει να γίνει *challenged*. Ο μόνος περιορισμός εδώ είναι ότι προηγουμένως δεν είχε ζητήσει τα *trapdoors* T_{w_0} και T_{w_1} . Ο *challenger* διαλέγει ένα τυχαίο b και δίνει στον *attacker* το $C = PEKS(A_{pub}, W_b)$, όπου C είναι το *PEKS* του *challenger*.

(iv) Ο *attacker* μπορεί να συνεχίσει να ζητάει *trapdoors* για *keywords* W , αρκεί να μην ζητήσει για τα W_0 και W_1 .

(v) Τελικά ο *attacker* δίνει *output* ενα b' και κερδίζει το παιχνίδι αν $b = b'$.

Με άλλα λόγια ο *attacker* κερδίζει το παιχνίδι εαν μπορεί να να μαντέψει σωστά εάν του δόθηκε το *PEKS* για το W_0 ή για το W_1 . Ορίζουμε το πλεονέκτημα του *attacker* A για να σπάσει το *PEKS* ως

$$Adv_A(s) = |Pr\{b = b'\} - \frac{1}{2}|$$

Ορισμός 6.0.2. Λέμε ότι ένα *PEKS* είναι ασφαλές απέναντι σε μία επίθεση διαλεγμένων *keywords* όταν για κάθε *polynomial time attacker* A , το $Adv_A(s)$ είναι μία *negligible function*.

Κεφάλαιο 7

Σχήματα δημοσίου κλειδιού που επιτρέπουν την κρυπτογράφηση με δυνατότητα αναζήτησης (PEKS)

7.1 Κατασκευή με bilinear map

Για την πρώτη προσέγγιση θα χρειαστούμε ένα bilinear map e και δύο ομάδες G_1, G_2 με τάξη έναν πρώτο αριθμό p . Το μέγεθος των G_1 και G_2 θα καθορίζεται από την Security parameter του αλγορίθμου KeyGen. Επίσης θα χρειαστούμε δύο Hash Functions $H_1 : \{0, 1\}^* \rightarrow G_1$ και $H_2 : G_2 \rightarrow \{0, 1\}^{logp}$.

Το PEKS δουλεύει ως εξής:

- KeyGen: Η security parameter καθορίζει το μέγεθος p των ομάδων G_1 και G_2 . Ο αλγόριθμος διαλέγει ένα τυχαίο a και έναν γεννήτορα g του G_1 . Το output του αλγορίθμου είναι :

$$A_{pub} = [g, h = g^a] \text{ και } A_{priv} = a$$

- PEKS(A_{pub}, W) : Υπολογίζει το $t = e(H_1(W), h^r) \in G_2$. Το output του αλγορίθμου είναι :

$$\text{PEKS}(A_{pub}, W) = [g^r, H_2(t)]$$

- $Trapdoor(A_{priv}, W)$: δίνει output :

$$T_w = H_1(W)^a (\in G_1).$$

- $Test(A_{pub}, S, T_w)$: Έστω $S = [A, B]$. Έλεγχξε αν $H_2(e(T_w, A)) = B$.

Εάν ισχύει : output ?yes? Εάν όχι : output ?no?

Η ασφάλεια του παραπάνω σχήματος οφείλεται στη δυσκολία του Bilinear Diffie-Hellman problem (BDH) το οποίο διατυπώνεται παρακάτω.

Bilinear Diffie-Hellman Problem (BDH) : Έστω g ένας γεννήτορας του G_1 . Το BDH έχει ως εξής : Δεδομένων των $g, g^a, g^b, g^c \in G_1$, να υπολογιστεί το $e(g, g)^{(abc)} \in G_2$.

Λέμε ότι το BDH είναι ανυπέρβλητο πρόβλημα εάν όλοι οι polynomial time αλγόριθμοι έχουν αμελητέο πλεονέκτημα στο να λύσουν το BDH.

7.2 Κατασκευή με trapdoors permutations

Η δεύτερη προσέγγιση για το PEKS βασίζεται σε μεταθέσεις των trapdoors, υποθέτωντας ότι ο συνολικός αριθμός των λέξεων είναι φραγμένος από μία polynomial συνάρτηση. Επιπλέον θα χρειαστούμε μία οικογένεια κρυπτογραφήσεων για τις οποίες θα ισχύει ότι : δεδομένου ενός ciphertext θα είναι υπολογιστικά δύσκολο να πούμε με ποιο public key έχει κρυπτογραφηθεί το κείμενο. Ένα τέτοιο σύστημα κρυπτογράφησης δημόσιου κλειδιού θα το λέμε source-indistinguishable. Πιο συγκεκριμένα το source-indistinguishability για ένα σχήμα κρυπτογράφησης (G, E, D) θα οριστεί μέσω του επόμενου παιχνιδιού μεταξύ ενός challenger και ενός attacker (G είναι ο key generation αλγόριθμος, E είναι ο αλγόριθμος κρυπτογράφησης και D είναι ο αλγόριθμος αποκρυπτογράφησης).

Η παραμετρος ασφαλείας s δίνεται και στους 2 παίκτες.

Source-Indistinguishability security game

- (i) Ο challenger τρεχει τον $G(s)$ δύο φορές και παράγει 2 ζεύγη από δημόσια και ιδιωτικά κλειδιά $(PK_0, Priv_0)$ και $(PK_1, Priv_1)$.

- (ii) Ο challenger διαλέγει ένα τυχαίο $M \in \{0, 1\}^s$ και ένα τυχαίο $b \in \{0, 1\}$ και στη συνέχεια υπολογίζει μια κρυπτογράφηση $C = PK_b(M)$. Ο challenger δίνει στον attacker το ζεύγος (M, C)
- (iii) Ο attacker βγάζει ένα output b' και κερδίζει το παιχνίδι αν $b = b'$.

Με άλλα λόγια, ο attacker κερδίζει εαν μπορεί να μαντέψει αν του δόθηκε η κρυπτογράφηση του M υπό το κλειδί PK_0 ή απο το PK_1 . Το πλεονεκτημα του attacker ορίζεται ως :

$$AdvSI_A(s) = |Pr[b = b'] - \frac{1}{2}|$$

Ορισμός 7.2.1. Λέμε οτι ενα *public key encryption* σχήμα είναι *source indistinguishable* αν για κάθε *polynomial time attacker* A , το $AdvSI_A(s)$ είναι μία *negligible function*.

Τώρα μπορούμε να δούμε πως δουλεύει ένα PEKS απο trapdoor permutations.

Όταν το keyword space Σ έχει polynomial size είναι σχετικά εύκολόνα κατασκευάσουμε ενα PEKS απο οποιοδήποτε source indistinguishable public key system (G, E, D) . Έστω s η παράμετρος ασφαλείας.

- **KeyGen** : Για κάθε $W \in \Sigma$ τρέχουμε τον $G(s)$ για να παράξουμε ένα ζεύγος δημοσιου/ιδιωτικού κλειδιου $PK_w/Priv_w$ για το source indistinguishable σύστημα κρυπτογράφησης. Έχουμε

$$A_{pub} = \{PK_w | W \in \Sigma\} \text{ και } A_{priv} = Priv_w | W \in \Sigma$$

- **PEKS** (A_{pub}, W) : Διάλεξε τυχαίο $M \in \{0, 1\}^s$. Υπολόγισε το:
 $PEKS(A_{pub}, W) = (M, E[PK_w, M])$ (δηλαδή κρυπτογράφησε το M χρησιμοποιώντας το δημόσιο κλειδί PK_w)

- **Trapdoor** (A_{priv}, W) : Το trapdoor για μία λέξη W είναι απλά :

$$T_w = Priv_w$$

- $Test(A_{pub}, S, T_w)$: Έλεγε αν η αποκρυπτογράφηση $D[T_w, S] = 0^s$.
Εάν ισχύει : output "yes"
Εάν όχι : output "no".

Η ασφάλεια του συγκεκριμένου PEKS ορίζεται ακριβώς όπως στον ορισμό 6.0.2 με τη μόνη διαφορά να είναι ότι εδώ ο attacker δεν έχει το δικαίωμα να κάνει ερωτήσεις για τα keywords.

Κεφάλαιο 8

Μειονεκτήματα του PEKS

Πέρα απο τον παράγοντα του χρόνου, που είναι γραμμικός (έχουμε ήδη δει κατασκευες που υποστηρίζουν ήμιγραμμικούς χρόνους), θα αναφερθούμε σε τρία επιπλέον μειονεκτήματα που παρουσιάζει το PEKS.

- (i) Για να μπορέσει ο server να τρέξει τον αλγόριθμο Test όταν λάβει ένα ciphertext η Alice πρέπει να δώσει στον server κάποιες πληροφορίες σχετικά με τα trapdoors. Το PEKS από τη φύση του, διασφαλίζει ότι χωρίς την πληροφορία για τα trapdoors, ο server δεν μαθαίνει τίποτα σχετικά με το e-mail. Επιπλέον το trapdoor για ένα keyword δεν αποκαλύπτει καμία πληροφορία για τα υπόλοιπα keywords. Στην πράξη, ένα τέτοιο σύστημα θα χρησιμοποιηθεί πολλές φορές. Στα σχήματα PEKS που έχουμε περιγράψει, ο server έχει τη δυνατότητα να κρατάει keywords και να τα χρησιμοποιεί για να μάθει πληροφορίες για μελλοντικά e-mails. Με άλλα λόγια, το PEKS που έχουμε περιγράψει είναι ένα one-way system. Το να υποθέσουμε ότι ο server δεν μπορεί να 'θυμάται' trapdoors είναι πολύ ισχυρή υπόθεση και πολύ δύσκολο να γίνεται implement.
- (ii) Το σύστημα χρησιμοποιεί ένα ασφαλές (κρυπτογραφημένο και πιστοποιημένο) κανάλι μεταξύ της Alice και του server. Προφανώς αυτό δεν είναι πάντοτε εφικτό, καθώς η κατασκευή ενός τέτοιου καναλιού είναι συνήθως αρκετά ακριβή.
- (iii) Σε πολλές περιπτώσεις η αναζήτηση μπορεί να γίνει σε πολλαπλά keyw-

ords τα οποία συνδέονται με λογικές πράξεις. Για παράδειγμα θα μπορούσε κάποιος να ψάξει ένα e-mail το οποίο περιέχει τις λέξεις ‘επείγον’ και/ή ‘γεύμα’. Το PEKS που έχουμε ορίσει όμως δεν μπορεί να υποστηρίξει τέτοια ερωτήματα.

Παρακάτω θα παρουσιάσουμε λύσεις για τα προβλήματα 2 και 3 μέσω παραλλαγών του PEKS ενώ θα συζητήσουμε και για το πρόβλημα 1.

8.1 Keywords που ανανεώνονται.

Στο PEKS, τα trapdoors πρέπει να παράγονται για κάθε keyword και αν ένα trapdoor δεν δοθεί στον server τότε αυτός δε θα μπορεί να αποφανθεί πιο PEKS ciphertext κρυπτογραφεί ποια λέξη. Αν παραλείψουμε αυτή τη συνθήκη, θα μπορούσαμε απλά να κρυπτογραφήσουμε ένα keyword χρησιμοποιώντας το Public key του χρήστη και ο χρήστης θα μπορούσε μετά να στείλει το αντίστοιχο ιδιωτικό κλειδί (μέσω ενός secure channel) στον server.

Ωστόσο, εδώ προκύπτει μία αντιφατική κατάσταση ακόμα και αν το PEKS που χρησιμοποιούμε είναι IND-CKA secure. Ας δούμε σαν παράδειγμα μία περίπτωση όπου τρία keywords, έστω “high priority”, “normal” και “less priority” εμφανίζονται συχνά στο σύστημα. Όταν τώρα ο χρήστης στέλνει ένα trapdoor, ο server το αποθηκεύει στην μνήμη του. Τότε, κάποια στιγμή ο server θα έχει πάρει trapdoors για όλα τα keywords που έχουν χρησιμοποιηθεί και θα μπορεί να αποφασίσει ποιο PEKS κρυπτοκείμενο κρυπτογραφεί ποια λέξη χωρίς να λάβει επιπλέον trapdoors από τον χρήστη. Από τη στιγμή που η χωρητικότητα των υπολογιστών αυξάνεται συνεχώς, ακόμα και αν χρησιμοποιήσουμε πιο πολλά keywords, ο server θα μπορεί να αποθηκεύει όλα τα trapdoors που θα λαμβάνει και συνεπώς θα μπορεί να διεξάγει την αναζήτηση για ένα keyword χωρίς την εντολή του χρήστη.

Θεωρητικά, κάποιος θα μπορούσε να παρακάμψει αυτό το πρόβλημα απλά χρησιμοποιώντας ένα keyword μόνο μία φορά. Αυτό όμως δεν είναι καθόλου πρακτικό μιας και σε πολλές περιπτώσεις οι χρήστες θα θέλουν να ξαναχρησιμοποιήσουν το ίδιο Keyword. Για παράδειγμα η Alice θέλει να κρυπτογραφεί ένα keyword “TARDIS” κάθε φορά που στέλνει ένα μήνυμα στον Bob. Μία

άλλη εναλλακτική θα ήταν ο sender να κρυπτογραφήσει ένα keyword με ένα nonce, αλλά αυτό καθιστά αδύνατο για τον receiver να παράξει ένα trapdoor εάν ο sender δε δώσει στον receiver το nonce που χρησιμοποιήθηκε όταν το PEKS ciphertext δημιουργήθηκε. Από τη στιγμή κιόλας που ένας από τους στόχους του PEKS είναι να κάνει το Keyword search δυνατό χωρίς την αλληλεπίδραση sender και receiver, η λύση με τα nonces δεν μπορεί να θεωρηθεί σωστή. Παρόλα αυτά, μία πιθανή λύση για το παραπάνω πρόβλημα θα ήταν η ανανέωση των συχνά χρησιμοποιούμενων keywords συνδέοντας τα με χρονική πληροφορία. Για παράδειγμα ένα keyword $w = \text{TheFlash}$, τώρα γίνεται $w' = \text{TheFlash}||23/06/2017$ όπου το 23/06/2017 υποδηλώνει '23 Ιουνίου 2017'. Ας παρατηρήσουμε εδώ ότι όσο πιο λεπτή είναι η χρονική περίοδος, τόσο περισσότερο αυξάνεται η ασφάλεια. Σε αυτό το παράδειγμα όταν ο receiver δώσει στον Server ένα trapdoor, ο Server θα μπορεί να ψάχνει τα PEKS ciphertexts που αντιστοιχούν στο w' , χωρίς να λάβει κάποιο επιπλέον trapdoor, μέχρι το τέλος της ημέρας. Αν βέβαια το time period είναι σχετικά μικρό, π.χ 5 ώρες, ο server θα μπορεί να ψάχνει σε αυτό το διάστημα των 5 ωρών.

Με αυτόν τον τρόπο το μέγεθος του συνόλου των keywords γίνεται άπειρο και συνεπώς δεν υπάρχει κανένα νόημα ο server να κρατάει trapdoors.

8.2 Αφαίρεση ασφαλούς καναλιού

Η βασική ιδέα είναι ο server να έχει το δικό του ζεύγος από δημόσιο και ιδιωτικό κλειδί. Για να δημιουργήσει ο sender ένα PEKS ciphertext χρησιμοποιεί και το δημόσιο κλειδί του server και το δημόσιο κλειδί του receiver. Ο receiver μπορεί τότε να στείλει ένα trapdoor για να ανακτήσει πληροφορίες που σχετίζονται με το κρυπτογραφημένο keyword ως συνήθως, αλλά αυτή τη φορά μπορεί να τα στείλει μέσω ενός δημόσιου καναλιού. Όταν ο Server λάβει το trapdoor μπορεί να ελέγξει αν ένα PEKS ciphertext κάνει match με το trapdoor χρησιμοποιώντας το ιδιωτικό του κλειδί.

Παρακάτω ορίζουμε αυτό το μοντέλο.

Ορισμός 8.2.1. Ένα *Secure Channel Free* κρυπτοσύστημα δημόσιου κλειδιού που επιτρέπει τη δυνατότητα αναζήτησης, αποτελείται από τέσσερις αλγό-

ριθμούς.

- (i) $KeyGen(k)$: Δέχεται μία παράμετρο ασφάλειας $k \in N$ σαν *input*, και παράγει μία *common parameter* cp .
- (ii) $KeyGenS(cp)$: Δέχεται σαν *input* το cp και παράγει ένα ζεύγος απο δημόσιο και ιδιωτικό κλειδί (skS, pkS) για τον *server*.
- (iii) $KeyGenR(cp)$: Δέχεται σαν *input* το cp και παράγει ένα ζεύγος απο δημόσιο και ιδιωτικό κλειδί (skR, pkR) για τον *receiver*.
- (iv) $SCF - PEKS(cp, pkS, pkR, w)$: Δέχεται για *input* το cp , τα δημόσια κλειδιά του *server* και του *receiver* και ένα *keyword* w . Επιστρέφει ένα *PEKS ciphertext* S που είναι το *searchable encryption* του w . $S = SCF - PESKS(cp, pkS, pkR, w)$.
- (v) $Trapdoor(cp, skR, w)$: Δέχεται *input* το cp , το ιδιωτικό κλειδί του *receiver* και ένα *keyword* w και παράγει ένα *trapdoor* T_w για το w .
- (vi) $Test(cp, Tw, skS, S)$: Δέχεται *input* το cp , το T_w για ένα *keyword* w , το ιδιωτικό κλειδί του *server* και ένα $PEKSS = (SCF - PEKS(pkS, pkR, w'))$. Ο αλγόριθμος επιστρέφει 'correct' αν $w = w'$, αλλιώς επιστρέφει 'incorrect'.

Παρακάτω θα ορίσουμε την ασφάλεια για το SCF-PEKS (IND-SCF-CKA). Πρακτικά αυτή η μορφή ασφάλειας εγγυάται πως αν ο *server* δεν έχει στην κατοχή του *trapdoors* για δεδομένα *keywords*, δεν μπορεί να ξεχωρίσει ποιο *PEKS ciphertext* κρυπτογραφεί ποια λέξη. Επίσης, μας εγγυάται ότι ένας τρίτος *attacker*, ο οποίος δεν έχει στην κατοχή του το ιδιωτικό κλειδί του *Server* δεν μπορεί να πάρει πληροφορίες για τα *PEKS ciphertexts* ακόμα και αν έχει στην κατοχή του όλα τα *trapdoors*.

Ορισμός 8.2.2. (*IND-SCF-CKA*) *Game 1* : Ο *attacker* A είναι ο *server*.

- *Phase 1-1* : Ο *challenger* τρέχει τους $KeyGen(k)$, $KeyGenR(k)$ και $KeyGenS(k)$. Παράγει τα cp και τα δημόσια και ιδιωτικά κλειδιά για τον *receiver* και τον *server*. Τα δίνει όλα στον A εκτός από το skR .

- *Phase 1-2*: Ο A ζητά *trapdoors* T_w για *keywords* w .
- *Phase 1-3*: Ο A δίνει ένα *target keyword pair* (w_0^*, w_1^*) . Ο *Challenger* τότε διαλέγει ένα τυχαίο $b \in \{0, 1\}$ και φτιάχνει ένα *PEKS ciphertext* ως εξής : $S^* = SCF - PEKS(cp, pkS, pkR, w_b^*)$ και το δίνει στον A .
- *Phase 1-4* : Ο A συνεχίζει τα ερωτήματα όπως στο *Phase 1-2* μόνο που τώρα δεν επιτρέπεται να ζητήσει *trapdoors* για τα w_0^* και w_1^*
- *Phase 1-5* : Ο A καταλήγει σε ένα $b' \in \{0, 1\}$ και κερδίζει το παιχνίδι αν $b = b'$

Ορίζουμε την επιτυχία του A να είναι :

$$Succ_A(k) = 2Pr[b' = b] - 1.$$

Game 2 : Ο *attacker* είναι ένας εξωτερικός *attacker*.

- *Phase 2-1* : Ο *challenger* τρέχει τους $KeyGen(k), KeyGenR(k)$ και $KeyGenS(k)$. Παράγει τα cp και τα δημόσια και ιδιωτικά κλειδιά για τον *receiver* και τον *server*. Τα δίνει όλα στον a εκτός από το skS .
- *Phase 2-2*: Ο a ζητά *trapdoors* T_w για *keywords* w .
- *Phase 2-3*: Ο a δίνει ένα *target keyword pair* (w_0^*, w_1^*) . Ο *Challenger* τότε διαλέγει ένα τυχαίο $b \in \{0, 1\}$ και φτιάχνει ένα *PEKS ciphertext* ως εξής : $S^* = SCF - PEKS(cp, pkS, pkR, w_b^*)$ και το δίνει στον a .
- *Phase 2-4* : όπως η *Phase 2-2*.
- *Phase 2-5* : Ο A καταλήγει σε ένα $b' \in \{0, 1\}$ και κερδίζει το παιχνίδι αν $b = b'$

Ορίζουμε την επιτυχία του a να είναι : $Succ_a(k) = 2Pr[b' = b] = 1$.

Το *SCF-PEKS* είναι *IND-SCF-CKA* ασφαλές αν $Succ_A(k)$ και $Succ_a(k)$ είναι και τα 2 *negligible functions*.

8.3 Πολλαπλά Keywords

Σύμφωνα με όσα έχουμε πει μέχρι τώρα, θα περίμενε κανείς ένα σχήμα PEKS να υποστηρίζει πολλαπλά keywords με τον εξής τρόπο. Έστω ότι η Alice θέλει να στείλει στον Bob ένα μήνυμα M με πολλαπλά keywords. Το μήνυμα της Alice θα έχει τη μορφή:

$$E(pkR, M) || PEKS(pkR, w_1) || \dots || PEKS(pkR, w_n)$$

όπου E είναι μία ασφαλή function δημόσιου κλειδιού.

Θα δούμε πως αυτό είναι εφικτό και ασφαλές παραλληλα.

Ορισμός 8.3.1. Ένα κρυπτοσύστημα δημόσιου κλειδιού που επιτρέπει τη δυνατότητα αναζήτησης για πολλαπλά keywords αποτελείται από τέσσερις αλγόριθμους.

- (i) $KeyGen(k)$: Ίδιος αλγόριθμος με τον $KeyGen$ του PEKS.
- (ii) $MPEKS(pk, w)$: Δέχεται σαν input το δημόσιο κλειδί του receiver και ένα multiple keyword $w = (w_1, \dots, w_n)$. Επιστρέφει ένα MPEKS ciphertext S που είναι το searchable encryption του w . Γράφουμε $S = MPEKS(pk, w)$.
- (iii) $Trapdoor(sk, w)$: Ίδιος αλγόριθμος με τον $Trapdoor$ του PEKS.
- (iv) $Test(Tw, S)$: Ίδιος αλγόριθμος με τον $Test$ του PEKS. Τώρα θα ορίσουμε μία έννοια ασφάλειας για το MPEKS, το (IND-MK-CKA).

Ορισμός 8.3.2. ((IND-MK-CKA) Έστω attacker A . Θεωρούμε το παρακάτω παιχνίδι.

- *Phase 1* : Ο challenger τρέχει τον $KeyGen(k)$ και παράγει ένα ζεύγος δημόσιου και ιδιωτικού κλειδιού (sk, pk) . Στη συνέχεια δίνει το pk στον A .
- *Phase 2*: Ο A ζητά trapdoors T_w για keywords w .

- *Phase 3:* Ο A δίνει ένα διανυσματικό ζεύγος (w_0^*, w_1^*) όπου $w_0^* = (w_{01}^*, \dots, w_{0n}^*)$ και $w_1^* = (w_{11}^*, \dots, w_{1n}^*)$, από τα οποία τίποτα δεν είχε ζητηθεί στο *phase 2*. Ο *challenger* διαλέγει ένα τυχαίο $b \in \{0, 1\}$ και δημιουργεί το $MPEKS(pk, wb^*)$ και το επιστρέφει στον A .
- *Phase 4:* Ο A συνεχίζει τα ερωτήματα όπως στο *Phase 1-2* μόνο που τώρα δεν επιτρέπεται να ζητήσει *trapdoors* για τα w_0^* και w_1^* .
- *Phase 5 :* Ο A καταλήγει σε ένα $b' \in \{0, 1\}$ και κερδίζει το παιχνίδι αν $b = b'$

Ορίζουμε την επιτυχία του A να είναι : $Succ_A(k) = 2Pr[b' = b]?1$.

Το $MPEKS$ είναι $IND-MK-CKA$ ασφαλές αν το $Succ_A(k)$ *negligible functions*.

Κεφάλαιο 9

Functional Encryption και Identity Based Encryption

Παρακάτω θα δούμε άλλα δύο κρυπτογραφικά συστήματα τα οποία θα μας βοηθήσουν να προσεγγίσουμε διαφορετικά το PEKS. Πιο συγκεκριμένα θα ορίσουμε το Functional Encryption και το Identity Based Encryption, και στο επόμενο κεφάλαιο θα δούμε πως συνδέονται.

9.1 Functional Encryption

Αρχικά θα δώσουμε τον συντακτικό ορισμό του functional encryption (FE) για μία functionality F . Η Functionality F περιγράφει τις συναρτήσεις ενός plaintext που μπορούμε να πάρουμε από το ciphertext.

Ορισμός 9.1.1. Ως *functionality* F στο (K, X) ορίζουμε μία συνάρτηση $F : K \times X \rightarrow \{0, 1\}^*$ η οποία περιγράφεται ως μία *deterministic Turing Machine*. Το σύνολο K είναι το σύνολο των κλειδίων και το το σύνολο X είναι το σύνολο των *plaintexts*. Θα εφοδιάσουμε το σύνολο K με ένα ειδικό κλειδί, το οποίο θα ονομάσουμε e .

Ορισμός 9.1.2. Ένα *functional encryption* σχήμα (FE) για μία *functionality* F στο (K, X) είναι μία τετράδα αλγορίθμων (*setup, keygen, enc, dec*) για τους οποίους για κάθε $k \in K$ και $\chi \in X$ ισχύει :

$setup(1^l) \rightarrow (pk, mk)$ (Δέχεται σαν input μία security parameter και δίνει output ένα δημόσιο και ένα master key).

$keygen(mk, k) \rightarrow sk$ (Δέχεται σαν input ένα master key και ένα k ανήκει K και δίνει output ένα secret key για το k).

$enc(pk, x) \rightarrow c$ (Δέχεται σαν input ένα δημόσιο κλειδί k και ένα plaintext x και παράγει μία κρυπτογράφηση c του x).

$dec(sk, c) \rightarrow y$ (Δέχεται σαν input ένα secret key k και ένα ciphertext c και υπολογίζει μία συνάρτηση του plaintext c $F(k, x)$).

Αξίζει εδώ να αναφέρουμε ότι το standard public key encryption είναι μία απλή περίπτωση functional encryption. Εύκολα γίνεται αντιληπτό από τα παρακάτω :

Πρόταση 9.1.3. *Το functional encryption είναι γενίκευση του standard public key encryption system :*

Έστω $L = \{1, e\}$ και θεωρούμε την παρακάτω functionality F ορισμένη στο (K, X) όπου X είναι ένα σύνολο από Plaintexts.

$$F(k, x) = \begin{cases} x & \text{if, } k = 1 \\ len(x) & \text{if, } k = e \end{cases}$$

Ένα secret key για $k = 1$, αποκρυπτογραφεί πλήρως τα ciphertexts ενώ το empty key e γυρίζει το bit length του plaintext.

9.2 Identity Based Encryption

Η βασική ιδέα πίσω από το identity based encryption είναι να χρησιμοποιείται ένα αυθαίρετο δημόσιο κλειδί. Κίνητρο για μία τέτοια ιδέα ήταν η απλοποίηση του certificate management στα e-mails.

Όταν δηλαδή η Alice στέλνει mail στον Bob στο bob@ntua.gr για παράδειγμα, να κρυπτογραφεί το mail της χρησιμοποιώντας σαν δημόσιο κλειδί το "bob@ntua.gr".

Με αυτόν τον τρόπο η Alice δεν έχει ανάγκη να έχει το πιστοποιητικό του δημόσιου κλειδιού του Bob. Όταν ο Bob λάβει το e-mail της Alice, επικοινωνεί με ένα third party, θα το πούμε Private Key Generator (PKG). Ο Bob γίνεται

authenticate στο PKG, με τον ίδιο τρόπο που θα γινόταν σε ένα CA λαμβάνει το private key του και τελικά αποκρυπτογραφεί το e-mail του και μπορεί να το διαβάσει.

Ορισμός 9.2.1. Ένα *identity based encryption* σχήμα E ορίζεται από 4 αλγόριθμους : *Setup*, *Extract*, *Encrypt*, *Decrypt*

- (i) *Setup* : Παίρνει μία παράμετρο ασφαλείας k και γυρνάει το *params* (παράμετροι του συστήματος) και ένα *master key*. Οι παράμετροι του συστήματος περιλαμβάνουν ένα πεπερασμένο σύνολο μηνυμάτων M και μία περιγραφή ενός πεπερασμένου συνόλου από *ciphertexts* C . Διαισθητικά οι παράμετροι του συστήματος θα είναι δημόσια γνωστοί, ενώ το *master key* θα είναι γνωστό μόνο στον "Private Key Generator" (PKG).
- (ii) *Extract*: Παίρνει σαν *Input* τα: *params*, *master key* και ένα αυθαίρετο *stringID* $\in \{0,1\}^*$, και γυρνάει ένα ιδιωτικό κλειδί d . Εδώ το αυθαίρετο *string* θα χρησιμοποιείται σαν δημόσιο κλειδί και το d θα είναι το αντίστοιχο ιδιωτικό κλειδί αποκρυπτογράφησης. Ο αλγόριθμος *Extract* παράγει ένα ιδιωτικό κλειδί από ένα δοθέν δημόσιο κλειδί.
- (iii) *Encrypt* : Δέχεται σαν *input* *params*, *ID* και m . Επιστρέφει ένα *ciphertext* $c \in C$.
- (iv) *Decrypt* : Δέχεται σαν *input* *params*, $c \in C$ και ένα ιδιωτικό κλειδί και επιστρέφει $m \in M$

Αυτοί οι αλγόριθμοι πρέπει να είναι συνεπείς. Δηλαδή όταν το private key d παράγεται από τον *Extract* αλγόριθμο όταν αυτός δέχεται ένα *ID* σαν δημόσιο κλειδί, τότε :

$$\forall m \in M : \text{Decrypt}(\text{params}, C, d) = m \text{ όπου } C = \text{Encrypt}(\text{params}, ID, M).$$

9.2.1 Ασφάλεια IBE

Chosen ciphertext security. Το chosen ciphertext security (IND-CCA) είναι το στάνταρ ασφαλείας για ένα κρυπτοσύστημα δημόσιου κλειδιού. Οπότε απαιτούμε και από το Identity Based Encryption σχήμα μας να ικανοποιεί αυτήν

την παράμετρο. Βέβαια στη συγκεκριμένη περίπτωση πρέπει να ενισχυθεί λίγο το IND-CCA. Ο λόγος για αυτό είναι ότι αν ένας attacker κάνει επίθεση σε ένα δημόσιο κλειδί ID, σε ένα IBE σύστημα, μπορεί ήδη να γνωρίζει τα private keys των users $ID_1 \dots ID_n$ της επιλογής του. Οπότε ο ορισμός της ασφάλειας (IND-CCA) πρέπει να επιτρέπει στον attacker να έχει στην κατοχή του τα ID_i που θέλει (εκτός προφανώς από το public key ID στο οποίο γίνεται η επίθεση). Θα αναφερόμαστε σε αυτά τα queries ως private key exchange queries. Άλλη μία μεγάλη διαφορά σε σχέση με τα standard public key cryptosystems είναι ότι εδώ ο attacker γίνεται challenged σε δημόσιο κλειδί ID της επιλογής του και όχι σε ένα τυχαίο δημόσιο κλειδί.

Λέμε ότι ένα Identity based encryption scheme, είναι (IND- ID -CCA) ασφαλές εάν κανένας polynomially bounded attacker A , δεν έχει μη αμελητέο πλεονέκτημα έναντι του challenger στο παρακάτω IND-ID-CCA παιχνίδι :

Setup : Ο Challenger διαλέγει μία παράμετρο ασφαλείας k και τρέχει τον Setup αλγόριθμο. Δίνει στον attacker τις προκύπτουσες παραμέτρους του συστήματος (params) και κρατάει το master key για τον εαυτό του,

Phase 1 : Ο attacker θέτει ερωτήματα q_1, \dots, q_m όπου κάθε q_i είναι ένα από τα επόμενα :

- Extraction query $[ID_i]$. Σε αυτήν την περίπτωση ο challenger απαντάει τρέχοντας τον αλγόριθμο Extract για να παράξει private key d_i που αντιστοιχεί στο public key ID_i . Στέλνει στον attacker το d_i .

-Decryption query $[ID_i, C_i]$. Ο challenger τώρα απαντάει τρέχοντας τον αλγόριθμο Extract και παράγει ένα private key d_i που αντιστοιχεί στο ID_i . Μετά, τρέχει τον αλγόριθμο Decrypt για να αποκρυπτογραφήσει το ciphertext C_i χρησιμοποιώντας το private key d_i . Στέλνει στον attacker το plaintext που θα προκύψει.

Challenge : Όταν ο attacker αποφασίσει πως έχει τελειώσει το Phase 1 δίνει output δυο, ίσου μήκους plaintexts $m_0, m_1 \in M$, και ένα ID στο οποίο επιθυμεί να γίνει challenged. Ο μοναδικός περιορισμός είναι να μην έχει εμφανιστεί το ID στη phase 1. Ο challenger διαλέγει ένα τυχαίο $b \in \{0, 1\}$ και θέτει : $C = \text{Encrypt}(params, ID, M_b)$. Στέλνει το C στον attacker σαν το challenge.

Phase 2 : Ο attacker συνεχίζει να κάνει ερωτήματα $q_{m+1} \dots q_n$ όπου κάθε

q_i είναι ένα απο τα επόμενα:

- Extraction query $[ID_i]$. Ο challenger απαντάει όπως στη Phase 1.
- Decryption query $[ID_i, C_i]$. Ο challenger απαντάει όπως στη Phase 1.

Guess : Τελικά ο attacker δίνει output ένα $b' \in \{0, 1\}$ και κερδίζει το game εάν $b = b'$

Τον attacker A θα τον ονομάσουμε IND-ID-CCA attacker. Σαν πλεονέκτημα του A όταν επιτίθεται στο IBE scheme E είναι η ποσότητα :

$$Adv_{E,A}(k) = |Pr[b = b'] - \frac{1}{2}|$$

όπου k είναι η παράμετρος ασφαλείας του αλγορίθμου Setup.

Χρησιμοποιώντας αυτό το IND-ID-CCA παιχνίδι μπορούμε να ορίσουμε την ασφάλεια σε ένα Identity Based Encryption scheme E απέναντι σε chosen ciphertext attack.

Ορισμός 9.2.2. Λέμε ότι ένα IBE system είναι ασφαλές απέναντι σε chosen ciphertext attacks αν για κάθε polynomial time IND-ID-CCA attaker A, η συνάρτηση $Adv_{E,A}(k)$ είναι negligible.

Πλέον και αφού έχουμε ορίσει και το Functional Encryption αλλά και το Identity Based Encryption, είναι πολυ εύκολο να δει κανείς ότι το FE είναι μία γενίκευση του IBE. Πιο συγκεκριμένα έχουμε ήδη πει οτι για μία functionality F το functional encryption scheme είναι:

- (i) $setup(1^l) \rightarrow (pk, mk)$
- (ii) $keygen(mk, k) \rightarrow sk$
- (iii) $enc(pk, x) \rightarrow c$
- (iv) $dec(sk, c) \rightarrow y = F(k, x)$

Αν λοιπόν ορίσουμε την Functionality $F(k,x)$ να είναι ίση με x όταν το k αντιστοιχεί σε ένα id που επιτρέπεται να αποκρυπτογραφήσουμε και (αναποδο T) σε κάθε άλλη περίπτωση τότε έχουμε ένα IBE σχήμα.

Κεφάλαιο 10

Άλλες προσεγγίσεις.

10.1 PEKS implies IBE

Ένα σύστημα κρυπτογράφησης δημόσιου κλειδιού που επιτρέπει την αναζήτηση συνδέεται με την identity based encryption. Η κατασκευή ενός PEKS φαντάζει να είναι δυσκολότερη από αυτήν ενός IBE σχήματος, Πραγμάτι, το παρακάτω λήμμα μας δείχνει ότι η δημιουργία ενός PEKS υπονοεί τη δημιουργία ενός Identity Based Encryption. Το ανάποδο μάλλον δεν ισχύει. Η ασφάλεια ενός IBE (IND-ID-CCA) είναι όπως έχει οριστεί σε προηγούμενο κεφάλαιο.

Λήμμα 10.1.1. Ένα PEKS το οποίο είναι ασφαλές απέναντι σε *chosen keyword attack* δημιουργεί ένα (IND-ID-CCA) secure IBE system.

Δεδομένου ενός PEKS (KeyGen, PEKS, Trapdoor, Test) το IBE system έχει ως εξής.

(i) Setup : Τρέξε τον KeyGen αλγόριθμο του PEKS για να παράξεις 2 κλειδιά A_{pub}/A_{priv} . Οι system parameter του IBE θα είναι το A_{pub} και το master key θα είναι το A_{priv} .

(ii) KeyGen : Το private key του identity based encryption που συνδέεται με ένα $X \in \{0, 1\}^*$ είναι :

$$d_X = [Trapdoor(A_{priv}, X||0), Trapdoor(A_{priv}, X||1)]$$

όπου το $||$ δηλώνει την παράθεση.

- (iii) Encrypt : Κρυπτογράφησε ένα $bitb \in \{0, 1\}$ χρησιμοποιώντας ένα δημόσιο κλειδί $X \in \{0, 1\}^*$ ως :

$$CT = PEKS(A_{pub}, X||b).$$

- (iv) Decrypt : Για την αποκρυπτογράφηση του $CT = PEKS(A_{pub}, X||b)$ χρησιμοποίησε το ιδιωτικό κλειδί $D_x = (D_0, D_1)$. Δώσε output '0' εάν:

$$Test(A_{pub}, CT, D_0) = 'yes'$$

Δώσε output '1' εάν:

$$Test(A_{pub}, CT, D_1) = 'yes'.$$

Τα παραπάνω μας δείχνουν ότι η κατασκευή ενός public key encryption scheme with keyword search είναι τουλάχιστον όσο δύσκολη είναι και η κατασκευή ενός Identity based encryption scheme.

Το ανάποδο (IBE implies PEKS) δεν έχει αποδειχτεί αλλά θεωρώ πως ένα καλό πρώτο βήμα θα ήταν να ορίσουμε :

$$PEKS(A_{pub}, W) = E_w[0^k].$$

Δηλαδή να κρυπτογραφήσουμε ένα string απο k μηδενικά χρησιμοποιώντας το public key του IBE : W ανήκει $\{0, 1\}^*$. Ο Test αλγόριθμος προσπαθεί να αποκρυπτογραφήσει το $E_w[0]$ και ελέγχει ότι το plaintext που προκύπτει είναι 0^k .

Αυτή η προσέγγιση δυστυχώς δεν εγγυάται ένα ασφαλές PEKS. Το πρόβλημα είναι ότι το ciphertext CT μπορεί να κάνει expose το δημόσιο κλειδί που χρησιμοποιήθηκε για την κρυπτογράφηση του. Γενικά ένα κρυπτοσύστημα δημόσιου κλειδιού, δεν κρύβει το δημόσιο κλειδι που χρησιμοποιεί για τη δημιουργία ενός ciphertext κάτι το οποίο είναι ουσιώδες για ένα PEKS για το οποίο θα ισχυεί ότι: $PEKS(A_{pub}, W) = E_w[0^k]$.

10.2 Multiple writers/multiple readers

Τα σχήματα που έχουμε μελετήσει μέχρι τώρα αναφέρονται στα μοντέλα single writer/single reader και multiwriter/single reader, τα οποία είναι και τα πιο δι αδεδομένα απο ερευνητική σκοπιά στον τομέα του searchable encryption.

Παρακάτω θα δούμε περιγραφικά κάποια σχήματα από τα μοντέλα `multiwriter/multireader`

10.2.1 Ντετερμινιστική κρυπτογράφηση

Η ιδέα του σχήματος αυτού είναι να γίνει το searchable encryption σχήμα πιο αποδοτικό, χρησιμοποιώντας ντετερμινιστική κρυπτογράφηση, θυσιάζοντας έτσι όμως την ασφάλεια. Το σχήμα αυτό, σε αντίθεση με τα υπόλοιπα σχήματα δημόσιου κλειδιού που είδαμε, αναφέρεται στο μοντέλο `multiwriter/multireader`. Ο κρυπτογραφημένος Index είναι άμεσα ευάλωτος σε dictionary attacks. Για να γίνει το ciphertext, searchable, ένα ντετερμινιστικό hash του keyword προσαρμόζεται στην κρυπτογράφηση του keyword.

Αποδοτικότητα : Το σχήμα αυτό μπορεί να χρησιμοποιήσει οποιοδήποτε σχήμα κρυπτογράφησης δημόσιου κλειδιού σε συνδυασμό με μία ντετερμινιστική hash function. Η αναζήτηση είναι πρακτικά αναζήτηση στη βάση δεδομένων για την τιμή της hash function

Ασφάλεια : Το σχήμα παρέχει μία έννοια semantic-security ασφάλειας που ονομάζεται PRIV security. Είναι παρεμφερές με τον στάνταρ ορισμό της IND-CPA ασφάλειας με δύο διαφορές. Το σχήμα πρέπει να έχει μεγάλη ελάχιστη εντροπία και τα plaintexts πρέπει να είναι ανεξάρτητα από το public key. Η ασφάλεια του σχήματος αποδεικνύεται στο random oracle model.

10.2.2 Proxy-re encryption

Εδώ έχουμε δύο σχήματα (θα τα ονομάσουμε PR-1 και PR-2 όπου κάθε χρήστης έχει το δικό του κλειδί για να κρυπτογραφεί, να ψάχνει και να αποκρυπτογραφεί δεδομένα. Και τα δύο σχήματα χρειάζονται έναν trusted key management server για να διαχειρίζεται τα κλειδιά.

Η ιδέα του PR-1 είναι να γίνει χρήση ενός RSA-based proxy re-encryption σχήματος. Το proxy re-encryption μπορεί να χρησιμοποιηθεί σε διάφορα κρυπτοσυστήματα. Πρακτικά, επιτρέπει στον server να μετασχηματίζει μία κρυπτογράφηση από το κλειδί ενός χρήστη σε μία κρυπτογράφηση από ένα άλλο κλειδί χωρίς να διαρρέει πληροφορία για το plaintext. Με αυτόν τον τρόπο, κρυπτοκείμενα από διαφορετικούς χρήστες μπορούν να μετασχηματιστούν σε

κρυπτοκείμενα από το κλειδί του server, πράγμα που επιτρέπει σε πολλαπλούς χρήστες να δημιουργήσουν searchable encrypted) δεδομένα. Με τον ίδιο τρόπο δημιουργούνται και τα trapdoors. Ένας χρήστης δημιουργεί ένα trapdoor για ένα keyword κρυπτογραφώντας το keyword με το κλειδί του. Ο server επανακρυπτογραφεί το trapdoor και μπορεί έτσι να ψάξει στην κρυπτογραφημένη βάση δεδομένων. Επιπλέον, η αποκρυπτογράφηση απαιτεί ένα re-encryption βήμα με στόχο να μετασχηματιστεί το κρυπτοκείμενο από το κλειδί του server σε κρυπτοκείμενο από το κλειδί του παραλήπτη. Το σχήμα αυτό χρησιμοποιεί μία semantically secure συμμετρική κρυπτογράφηση για την κρυπτογράφηση των δεδομένων, αλλά για την αναζήτηση χρησιμοποιείται μόνο μία hash function η οποία καθιστά τα αρχεία searchable, αλλά δεν είναι semantically secure.

Το PR-2 επίσης χρησιμοποιεί ένα RSA-based proxy re-encryption σχήμα. Η ιδέα εδώ είναι να χρησιμοποιηθεί OAEP (optimal asymmetric encryption padding) για να γίνουν τα κρυπτοκείμενα indistinguishable. Το RSA-OAEP είναι ασφαλές, όσο είναι ασφαλές και το RSA. Η κύρια διαφορά είναι στην κρυπτογράφηση των keywords. Η proxy re-encryption που χρησιμοποιείται για τα keywords είναι ντετερμινιστική.

Αποδοτικότητα : Το PR-1 υπολογίζει $u+1$ exponentiations για τη δημιουργία του index, όπου το u είναι ο αριθμός των διακριτών keywords ανά αρχείο. Για την αναζήτηση, ο Server επανακρυπτογραφεί (ένα exponentiation) ένα trapdoor και ελέγχει κάθε keyword ανά index για ισότητα.

Το PR-2 Το PR-1 υπολογίζει $4u+1$ exponentiations για τη δημιουργία του index, όπου το u είναι ο αριθμός των διακριτών keywords ανά αρχείο. Για την αναζήτηση, ο Server επανακρυπτογραφεί (ένα exponentiation) ένα trapdoor και στη συνέχεια πρέπει να υπολογίσει άλλα $4u$ exponentiations

Ασφάλεια : Και τα δύο σχήματα είναι IND-CKA1 ασφαλή (ο αντίπαλος έχει πρόσβαση στις δημόσιες παραμέτρους). Επιπλέον τα σχήματα είναι One-Way ασφαλή, εφόσον ισχύει η RSA assumption στο random oracle. Η One-Way ασφάλεια εγγυάται ότι είναι δύσκολο για έναν αντίπαλο να αντιστρέψει ένα κρυπτοκείμενο που έχει κρυπτογραφηθεί με το κλειδί ενός χρήστη και να μάθει το keyword, ακόμα και αν ο αντίπαλος έχει τις δημόσιες παραμέτρους και όλα τα ζεύγη κλειδιών χρηστών-server, χωρίς όμως να ξέρει το ζευγος κλειδιών του συγκεκριμένου χρήστη.

Τα κύρια προβλήματα τπυ σχήματος αυτού είναι οτι απαιτεί έναν trusted server καθώς και ότι στα proxy re-encryption σχήματα είναι εφικτή μία collusion attack που επιτρέπει σε έναν αντίπαλο και έναν χρήστη να μάθουν τα master keys, αν ο αντίπαλος γνωρίζει όλα τα server-side κλειδιά.

Κεφάλαιο 11

Μια γενικότερη συζήτηση περί ιδιωτικότητας

Τον τελευταίο καιρό μάλιστα σε παγκόσμιο επίπεδο ένας πόλεμος μεταξύ της εθνικής ασφαλείας και της ιδιωτικότητας. Κυβερνήσεις ανά τον κόσμο επιμένουν να παραβιάζουν προσωπικά δεδομένα χρηστών, είτε αυτά είναι αρχεία αποθηκευμένα στο cloud ή ακόμα και συνομιλίες σε εφαρμογές chat. Είναι ηθικά σωστό να κρύβουμε τα αρχεία μας, τα δεδομένα μας και τις συνομιλίες από τους servers;

Μπορούμε να δούμε πιο συγκεκριμένα την ιστορία με το whatsapp (εφαρμογή ανταλλαγής μηνύματων) το οποίο ναι μεν δεν συμπεριλαμβάνεται στην κατηγορία του searchable encryption αλλά η λογική είναι στο ίδιο πνεύμα. Αποκρύπτουμε τις πληροφορίες μας από τον server provider, όπως ακριβώς γίνεται και με τα sse σχήματα και με τα PEKS.

Πριν απαντήσουμε στο ερώτημα κατά πόσον είναι σωστή η αποκρύψη των δεδομένων μας ας δούμε κάποια βασικά πράγματα.

Πως δουλεύει το encryption;

Η κρυπτογραφία είναι γνωστή και ως η επιστήμη της ασφαλούς επικοινωνίας. Οι αλγόριθμοι κρυπτογράφησης είναι σαν ένα κουτί με δύο κλειδαριές. Για παράδειγμα, εάν ένας χρήστης που ονομάζεται Alice θέλει να στείλει στον φίλο

της ένα ασφαλές μήνυμα, το βάζει στο κουτί και το κλειδώνει με το κλειδί. Στη συνέχεια, στέλνει το κλειδωμένο κουτί στον φίλο της Bob, ο οποίος μπορεί μόνο να ανοίξει το κιβώτιο και να διαβάσει το μήνυμα της Alice αν έχει το δικό του έγκυρο κλειδί.

Αλλά για να είναι εφικτή η επικοινωνία με νέους χρήστες, χρειαζόμαστε έναν τρόπο να μοιράζουμε κλειδιά που είναι ακόμα ασφαλή. Για να ξεπεραστεί αυτό, κάθε χρήστης έχει αυτό που ονομάζεται δημόσιο κλειδί που είναι διαθέσιμο σε οποιονδήποτε και αποδεικνύει την ταυτότητα του χρήστη και ένα ιδιωτικό κλειδί που παραμένει με τον χρήστη. Η Alice χρησιμοποιεί το δημόσιο κλειδί του Bob για να κλειδώσει το κουτί, αλλά αυτό μπορεί να ξεκλειδωθεί μόνο με το ιδιωτικό κλειδί του Bob.

Το σύστημα της WhatsApp προσθέτει ένα επιπλέον επίπεδο κρυπτογράφησης, γνωστό ως "perfect forward secrecy". Αυτό είναι σαν μια δεύτερη κλειδαριά με ένα κλειδί που αλλάζει για κάθε συνεδρία μηνυμάτων. Όταν η Alice επιθυμεί να στείλει ένα μήνυμα στον Bob, δημιουργεί πρώτα ένα νέο session key, το τοποθετεί στο κουτί και χρησιμοποιεί το δημόσιο κλειδί του Bob για να το κλειδώσει. Τότε το στέλνει στον Bob, ο οποίος χρησιμοποιεί το ιδιωτικό του κλειδί για να αποκτήσει πρόσβαση στο session key. Οι δύο τους μπορούν στη συνέχεια να αρχίσουν να επικοινωνούν με ασφάλεια χρησιμοποιώντας το session key που είναι γνωστό μόνο σε αυτούς για να κρυπτογραφήσουν τα μηνύματά τους.

Αυτό το σύστημα εγγυάται ότι δεν υπάρχει κανένα κλειδί που θα παρέχει πρόσβαση σε όλα τα δεδομένα που στέλνονται μεταξύ της Alice και του Bob είτε στο παρελθόν είτε στο μέλλον. Με άλλα λόγια, ακόμα και αν το κλειδί είναι συμβιβασμένο, θα ξεκλειδώσει μόνο μερικά μηνύματα πριν γίνει άχρηστο.

Ωστόσο, το προηγούμενο σύστημα της WhatsApp σήμαινε ότι η εταιρεία ήταν σε θέση να έχει πρόσβαση στα κλειδιά και έτσι θεωρητικά θα μπορούσε εύκολα να ξεκλειδώσει τα μηνύματα, παραβιάζοντας το απόρρητο της Alice. Πέρυσι, η εταιρεία εισήγαγε την αποκαλούμενη κρυπτογράφηση "end-to-end", η οποία φαίνεται να έχει λύσει αυτό το πρόβλημα. Η Alice και ο Bob χρησιμοποιούν τώρα κλειδιά για τα οποία η WhatsApp δεν κρατάει πληροφορίες, πράγμα που σημαίνει πως μόνο η Alice και ο Bob μπορούν να ξεκλειδώσουν τα μηνύματά τους.

Backdoor

Ο τρόπος με τον οποίο λειτουργεί τώρα το WhatsApp καθιστά αδύνατο για ένα τρίτο μέρος να ξεκλειδώσει τα μηνύματα που ανταλλάσσουν η Alice και η Bob. Ο μόνος τρόπος με τον οποίο ένα τρίτο μέρος, όπως η αστυνομία ή οι υπηρεσίες πληροφοριών, έχει πρόσβαση στα μηνύματα των χρηστών είναι εάν το WhatsApp καταργήσει την end to end κρυπτογράφηση και επιστρέψει στην παλιά έκδοση του λογισμικού του ή εάν έχει εγκατασταθεί ένα backdoor.

Ένα backdoor μπορεί να θεωρηθεί λογισμικό ενσωματωμένο στην εφαρμογή. Στην περίπτωση του WhatsApp, αυτό θα ήταν λογισμικό που παρέχει πρόσβαση σε όλα τα κλειδιά που δημιουργούν οι χρήστες. Οι κυβερνήσεις υποστηρίζουν ότι ένας τέτοιος μηχανισμός θα ενεργοποιείται μόνο εάν υπήρχε ένταλμα πρόσβασης στα μηνύματα ενός ύποπτου χρήστη. Με την ενεργοποίηση του backdoor, το WhatsApp θα ανακτήσει τα κλειδιά που δημιουργήθηκαν από τους αντίστοιχους χρήστες προκειμένου να αποκρυπτογραφήσουν τα μηνύματά τους. Αυτό θα επέτρεπε στο WhatsApp να αποκαλύψει όλες τις πληροφορίες που έστειλε ένας συγκεκριμένος χρήστης.

Όμως, τα backdoors δημιουργούν επίσης ένα θέμα ευπάθειας στο λογισμικό ή στο σύστημα που μπορούν να χρησιμοποιήσουν οι εισβολείς για να αποκτήσουν πρόσβαση ή στα ιδιωτικά δεδομένα των χρηστών. Ως αποτέλεσμα, αν το WhatsApp προσθέσει ένα backdoor στο λογισμικό τότε δεν θα είναι πλέον ένας ασφαλής τρόπος επικοινωνίας και έτσι θα χάσει ένα από τα πιο βασικά χαρακτηριστικά του. Είναι επίσης πιθανό ότι η εγκατάσταση ενός τέτοιου μηχανισμού θα σήμαινε ότι οι υπηρεσίες πληροφοριών θα χτυπούν καθημερινά την πόρτα της WhatsApp ζητώντας τους να αποκαλύψουν μηνύματα κάποιου.

Τελικά, αν κάποιος νομίζει ότι η κατάργηση της κρυπτογράφησης WhatsApp θα είναι η λύση, τότε δεν καταλαβαίνουν το πραγματικό πρόβλημα. Ακόμα κι αν αφαιρεθεί η end to end κρυπτογράφηση από το WhatsApp, οι εγκληματίες θα μπορούσαν να δημιουργήσουν το δικό τους παρόμοιο λογισμικό που θα τους επιτρέψει να επικοινωνούν με ασφάλεια, ενώ οι απλοί χρήστες θα χάσουν τη δυνατότητα να στείλουν πραγματικά ιδιωτικά μηνύματα.

Όλα τα παραπάνω προφανώς, δεν έχουν να κάνουν με το Whatsapp συγκεκριμένα, αλλά με το ότι όσο πιο εύκολο είναι να παραβιάσσει το Privacy

τόσο περισσότερο θα προσπαθούν να συμβεί. Η φιλοσοφία του searchable encryption είναι ακριβώς αυτή. Το πως δηλαδή μπορεί ένας χρήστης να εκμεταλλευτεί τη τεχνολογία χωρίς να θυσιάσει καθόλου από την ιδιωτικότητα του.

Κεφάλαιο 12

Επίλογος και συμπεράσματα

Η αποθήκευση στο Cloud αυξάνεται εκθετικά λόγω της αποτελεσματικότητας του κόστους, αλλά φέρνει μαζί της νέες ανησυχίες για την ασφάλεια. Από τη μία, πρέπει να αναπτυχθούν λύσεις που να προστατεύουν τα αρχεία των χρηστών και από την άλλη πρέπει αυτές οι λύσεις να είναι πρακτικές και εύχρηστες. Τα δυναμικά σχήματα κρυπτογράφησης με δυνατότητα αναζήτησης είναι πολύ χρήσιμα αλλά δυστυχώς χρειάζονται αρκετή έρευνα ακόμη ούτως ώστε να γίνουν ασφαλή. Τα σχήματα δημόσιου κλειδιού, ενώ καθιστούν πολύ εύκολη την πρόσβαση στη βάση δεδομένων από πολλούς διαφορετικούς χρήστες υστερούν όσον αφορά τον χρόνο. Όπως δείξαμε και στο κείμενο ένα πρόβλημα που παρουσιάζει ενδιαφέρον είναι να καταφέρουμε να δείξουμε πως η κατασκευή ενός Identity Based Encryption σχήματος, συνεπάγεται τη δημιουργία ενός PEKS.

Από τα πρώτα κιόλας βήματα του searchable encryption, η έρευνα προσανατολίζεται σε τρεις κατευθύνσεις: Την εκφραστικότητα των ερωτημάτων προς τους σερβερς, την αποδοτικότητα και την ασφάλεια. Είναι εύκολο να δει κάποιος τα tradeoffs που γίνονται στις επόμενες τρεις περιπτώσεις: (1) ασφάλεια εναντίον αποδοτικότητας, (2) ασφάλεια εναντίον εκφραστικότητας ερωτημάτων και (3) αποδοτικότητα εναντίον εκφραστικότητας ερωτημάτων. Όταν ένα σχήμα προσπαθεί να βελτιωθεί σε κάποιον από αυτούς τους τομείς πρέπει να θυσιάσει κάποιον άλλο. Ένα καλό παράδειγμα για να καταδείξουμε αυτό το tradeoff ειδικά για την περίπτωση (1) είναι η χρήση ντετερμινιστικής κρυπτογράφησης. Η ντετερμινιστική κρυπτογράφηση κάνει ένα σχήμα σαφώς πιο αποδοτικό, αλλά

παράλληλα διαρρέει περισσότερη πληροφορία . Στη συγκεκριμένη κιάλας περίπτωση, το ίδιο το κρυπτοκείμενο διαρρέει πληροφορίες (πχ ομοιότητα των keywords) και την ίδια στιγμή διαρρέεται και το search pattern. Στην περίπτωση της ντετερμινιστικής κρυπτογράφησης δημόσιου κλειδιού που χρησιμοποιεί γνωστα keywords ο server μπορεί να κάνει brute force attack κρυπτογραφώντας όλα τα πιθανά keywords με το δημόσιο κλειδί και ελέγχοντας τις κρυπτογραφήσεις ενάντια στα κρυπτοκείμενα.

Τα περισσότερα μοντέλα πέφτουν στην κατηγορία single writer/single reader και multiwriter/ single reader και αυτό οφείλεται κυρίως στα συμμετρικά και μη συμμετρικά μοντέλα κρυπτογράφησης που χρησιμοποιούνται. Αν και το μοντέλο single writer/ multireader δεν έχει ερευνηθεί σε βάθος έχει αρχίσει να είναι πολύ σύννητες, μιας και αναφέρεται σε βάσεις δεδομένων όπου κάποιος ανεβάζει ένα αρχείο και πολλοί μπορούν να το δουν. Τέλος στο μοντέλο multiwriter/ multireader δεν υπάρχει έντονη ερευνητική δραστηριότητα, καθώς δεν έχουμε πρακτικές εφαρμογές. Τα περισσότερα implementations δεν είναι δημοσίως ελεύθερα, πράγμα που καθιστά πολυ δύσκολη τη συγκριση των σχημάτων χρησιμοποιώντας το ίδιο hardware. Επιπλέον είναι επίσης δύσκολη η απευθείας σύγκριση αφού τα πρωτόκολλα για διάφορα searchable encryption σχήματα αναφέρονται σε διαφορετικά σενάρια και threat models.

12.1 Συμπεράσματα

Αν και η έρευνα πάνω στην κρυπτογράφηση με δυνατότητα αναζήτησης διεξάγεται πάνω από μία δεκαετία τώρα, υπάρχουν ακόμα μεγάλα περιθώρια βελτίωσης όσον αφορά τις τρεις κύριες ερευνητικές κατευθύνσεις.

Εκφραστικότητα Ερωτημάτων : Πως δηλαδή ένα searchable encryption σχήμα θα υποστηρίζει πιο σύνθετα ερωτήματα από ερωτήματα απλής ισότητας, όπως για παράδειγμα λογικές εκφράσεις. Τα public key searchable encryption σχήματα φαίνεται να μπορούν πιο εύκολα να προσαρμοστούν για να υποστηρίζουν τέτοια ερωτήματα αλλά όπως έχουμε ήδη αναφέρει, υστερούν στον τομέα της αποδοτικότητας.

Αποδοτικότητα : Υπάρχει ένα αρκετά σημαντικό πρόβλημα που πρέπει να λυθεί στο multiuser μοντέλο, ούτως ώστε να διαδοθεί η χρήση του searchable

encryption. Τα αποδοτικά σχήματα με optimal χρόνο αναζήτησης που επιτυγχάνουν IND/PK-CKA2 ασφάλεια, υπάρχουν μόνο στο μοντέλο single writer/single reader. Στα υπολοιπά μοντέλα, σχήματα που επιτυγχάνουν IND/PK-CKA2 ασφάλεια δεν είναι αποδοτικά(χρόνος αναζήτησης γραμμικός στον αριθμό των αρχείων) και συνεπώς δεν ανταποκρίνονται καλά για μεγάλες βάσεις δεδομένων, πράγμα που τα καθιστά μη πρακτικά για καθημερινή χρήση. Υπάρχουν δύο βασικοί λόγοι που είναι αναγκαία η κατασκευή τέτοιων σχημάτων.

- (i) Μία σειρά απο κυβερνητικούς κανονισμούς, όπως το Health Insurance Portability and Accountability Act προβλέπουν ότι οι οργανισμοί θα πρέπει να κρυπτογραφούν τα δεδομένα τους. Την ίδια στιγμή όμως αυτοί οι οργανισμοί θα πρέπει να μπορούν να ψάξουν τα κρυπτογραφημένα δεδομένα τους με ασφάλεια. Έτσι λοιπον, γίνεται σαφές οτι είναι όλο και πιο σημαντικό να υπάρξουν λύσεις που να ανταποκρίνονται στις απαιτήσεις πραγματικών σεναρίων.
- (ii) Στο παρελθόν πολλές εταιρίες βασιζόντουσαν κυρίως σε ενα offline περιβάλλον για να κρατάνε τα αρχεία τους. Με τη χρήση όμως του cloud όπως αυτή γίνεται σήμερα περιορίζει τις εταιρίες να βασίζονται μόνο στην κρυπτογράφηση για να ασφαλίσουν τα αρχεία τους και να αντιμετωπίσουν απειλές όπως το business espionage

Ασφάλεια : Όλα τα σχήματα που συζητήθηκαν παρέχουν μία ασφάλεια. Πρέπει όλα τα searchable encryption σχήματα να παρέχουν μία αίσθηση ασφαλείας, αλλιώς η κρυπτογράφηση θα ήταν περιττή. Ακόμα και έτσι όμως, είναι αρκετά δύκολα να αποφανθούμε για το ποια σχήματα παρέχουν την καλύτερη ασφάλεια, καθώς τα περισσότερα από αυτά αποδεικνύονται ασφαλή σε διαφορετικά security models. Είναι ωστόσο κοινά αποδεκτό πως αν ένα σχήμα είναι IND-CKA2ασφαλές, τότε θεωρείται ασφαλές.

Βέβαια αυτός ο ορισμός απο μόνος του δεν είναι αρκετός, αφού τα searchable encryption σχήματα πρέπει να λαμβάνουν υπόψιν τους και τις πληροφορίες που διαρρέονται οι οποίες μπορούν να οδηγήσουν σε επιθέσεις στατιστικής ανάλυσης (π.χ.search pattern).

Κάποια σχήματα μπορεί να επιτρέπουν τη διαρροή του search pattern, ενώ άλλα να προσπαθούν να κρύψουν όσο περισσότερη πληροφορία γίνεται. Η διαρροή του search pattern μπορεί να μην προκαλεί προβλήματα για κάποια σχήματα, ενώ για άλλα να είναι καταστροφική. Για παράδειγμα, για μία ιατρική βάση δεδομένων η διαρροή του search pattern μπορεί να σημαίνει και τη διαρροή πολύ μεγάλου ποσού πληροφορίας που μπορεί να συσχετιστεί με άλλες δημόσιες βάσεις δεδομένων. Στις πιο πολλές περιπτώσεις πάντως είναι αποδεκτό για ένα σχήμα να διαρρεί το access pattern. Ωστόσο, για εφαρμογές ύψιστης ασφάλειας, θα πρέπει να παραμένει κρυφό. Όσο γνωρίζουμε, δεν υπάρχει πρακτικό searchable encryption σχήμα, που να κρύβει το access pattern

12.2 Ερευνητικές κατευθύνσεις

Η μελλοντική έρευνα στον τομέα του searchable encryption θα πρέπει να επικεντρωθεί κυρίως στην εκφραστικότητα των ερωτημάτων και στην αποδοτικότητα των searchable encryption σχημάτων στα μοντέλα single writer/multireader, multiwriter/single reader και multiwriter/multireader. Η έρευνα για την εκφραστικότητα των ερωτημάτων θα πρέπει να επικεντρωθεί στο πως μπορεί να καμφθεί το κενό μεταξύ των searchable encryption σχημάτων και της αναζήτησης σε ένα plaintext κείμενο. Αυτό σημαίνει ότι τα σχήματα θα πρέπει να υποστηρίζουν τουλάχιστον αναζήτηση φράσεων, λογικών εκφράσεων και παρεμφερή αναζήτηση. Ειδικά για multireader σενάρια, τα οποία είναι και αποτελούν τυπικά σενάρια data sharing η μεγαλύτερη εκφραστικότητα των ερωτημάτων είναι άκρως επιθυμητή. Η πιο ενδιαφέρουσα ερευνητική προσέγγιση είναι σίγουρα μία problem driven προσέγγιση, να αναγνωρίσουμε δηλαδή πρώτα πραγματικά προβλήματα, απαιτήσεις και ανάγκες των χρηστών και στη συνέχεια να δούμε πως το searchable encryption μπορεί να αποτελέσει ασφαλή και πρακτική λύση για καθημερινά σενάρια, όπως η αναζήτηση σε online databases, το e-mail routing και η αναζήτηση σε κρυπτογραφημένα e-mails. Για τη διάδοση των searchable encryption σχημάτων είναι αναγκαία η αποδοτικότητα των multiuser σχημάτων σε μεγάλες βάσεις δεδομένων.

Συνολικά, η αναζήτηση με δυνατότητα κρυπτογράφησης έχει πολλές προοπτικές για την αύξηση της ασφάλειας των λύσεων σε cloud, επομένως είναι μία

πολύ ενδιαφέρουσα ερευνητική κατεύθυνση με σκοπό να αποκτήσουμε πιο πρακτικές λύσεις που μπορούν να χρησιμοποιηθούν στην πραγματικές εφαρμογές και όχι μόνο σε ιδεατά σενάρια.

Βιβλιογραφία

- [1] Στάθης Ζάχος, Αρης Παγουρτζής. Υπολογιστική Κρυπτογραφία. Σημειώσεις ΕΜΠ.
- [2] Δημήτριος Μ. Πουλάκης. Κρυπτογραφία, η επιστήμη της ασφαλούς επικοινωνίας.
- [3] Αλέξανδρος Παπαιωάννου, Χρήστος Κουκουβίνος. Κρυπτογραφία.Σημειώσεις ΕΜΠ.
- [4] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, Advances in Cryptology ? EUROCRYPT 2004, volume 3027 of Lecture Notes in Computer Science, pages 506?522, Interlaken, Switzerland, May 2?6, 2004. Springer, Berlin, Germany. (Cited on page 2.)
- [5] Dawn Xiaodong Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In 2000 IEEE Symposium on Security and Privacy, pages 44?55, Oakland, California, USA, May 2000. IEEE Computer Society Press. (Cited on page 1, 3, 5.)
- [6] Rafael Dowsley, Antonis Michalás , Matthias Nagel . A Report on Design and Implementation of Protected Searchable Data in IaaS.
- [7] Eu-Jin Goh. Secure indexes. Cryptology ePrint Archive, Report 2003/216, 2003. <http://eprint.iacr.org/2003/216>. (Cited on page 1, 6.)

- [8] Reza Curtmola, Juan A. Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, ACM CCS 06: 13th Conference on Computer and Communications Security, pages 79?88, Alexandria, Virginia, USA, October 30 ? November 3, 2006. ACM Press. (Cited on page 1, 2, 3, 5, 6, 7.)
- [9] Seny Kamara, Charalampos Papamanthou, and Tom Roeder. Dynamic searchable symmetric encryption. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, ACM CCS 12: 19th Conference on Computer and Communications Security, pages 965?976, Raleigh, NC, USA, October 16?18, 2012. ACM Press. (Cited on page 1, 2, 3, 5, 7, 8, 9, 11.)
- [10] Kaoru Kurosawa and Yasuhiro Ohtaki. UC-secure searchable symmetric encryption. In Angelos D. Keromytis, editor, FC 2012: 16th International Conference on Financial Cryptography and Data Security, volume 7397 of Lecture Notes in Computer Science, pages 285?298, Kralendijk, Bonaire, February 27 ? March 2, 2012. Springer, Berlin, Germany. (Cited on page 1, 7.)
- [11] Emil Stefanov, Charalampos Papamanthou, and Elaine Shi. Practical dynamic searchable encryption with small leakage. In ISOC Network and Distributed System Security Symposium ? NDSS 2014, San Diego, California, USA, February 23?26, 2014. The Internet Society. (Cited on page 10, 11, 12.)
- [12] Muhammad Naveed, Manoj Prabhakaran, and Carl A. Gunter. Dynamic searchable encryption via blind storage. In 2014 IEEE Symposium on Security and Privacy, pages 639?654, Berkeley, California, USA, May 18?21, 2014. IEEE Computer Society Press. (Cited on page 2, 3, 10, 14, 15, 18.)
- [13] D. Boneh and M. Franklin, Identity-based Encryption from the Weil Pairing. Appears in SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615,

2003. An extended abstract of this paper appears in the Proceedings of Crypto 2001, volume 2139 of Lecture Notes in Computer Science, pages 213-229, Springer-Verlag, 2001.

- [14] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, TCC 2007: 4th Theory of Cryptography Conference, volume 4392 of Lecture Notes in Computer Science, pages 535-554, Amsterdam, The Netherlands, February 21-24, 2007. Springer, Berlin, Germany. (Cited on page 2.).
- [15] Baek J., Safavi-Naini R., Susilo W. (2008) Public Key Encryption with Keyword Search Revisited. In: Gervasi O., Murgante B., Lagan? A., Taniar D., Mun Y., Gavrilova M.L. (eds) Computational Science and Its Applications • ICCSA 2008. ICCSA 2008. Lecture Notes in Computer Science, vol 5072. Springer, Berlin, Heidelberg
- [16] Douglas R. Stinson . Cryptography, theory and practice, 2nd edition.