Μελέτη Πρωτοκόλλων Ασφαλούς Υπολογισμού Πολλών Μερών (Study on Secure Multi Party Computation Protocols)

Θεόδωρος Συμεωνίδης
Α.Μ. 1064870

Τμήμα Μηχανικών Η/Υ και Πληροφορίκής
Πανεπιστήμιο Πατρών

1 Οκτωβρίου 2022

Εισαγωγή

Το αντικείμενο αυτής της διπλωματικής εργασίας :

- Μελέτη Πρωτοκόλλων Ασφαλούς Υπολογισμού Πολλών Μερών (Secure Multi Party Computation (SMPC) Protocols)
- Εφαρμογή τους στην κατασκευή μιας Ασφαλούς Υπολογισμού Δύο Μερών BLAS Level-1 Βιβλιοθήκης, για πρώτη φορά στη βιβλιογραφία

Για να επιτευχθεί το παραπάνω έγινε εισαγωγή και αναπτύχθηκε το εξής θεωρητικό υπόβαθρο :

- Μαθηματικά της κρυπτογραφίας
- Σύγχρονα κρυπτογραφικά εργαλεία που χρησιμοποιούνται σε πρωτόκολλα SMPC
- Θεωρητικά αποδείξιμη ασφάλεια

Theorem

Ασφαλής Υπολογισμός *: Όλες οι υπολογιστικές μέθοδοι που επιτρέπουν τον υπολογισμό επάνω σε δεδομένα κρατώντας τα δεδομένα μυστικά. Τα δεδομένα, δηλαδή, ένα μέρος των δεδομένων μπορεί να ανήκουν σε κάποιον τρίτο ο οποίος δε θέλει να μας φανερώσει τις τιμές τους.*

Παραδείγματα :

- Υπολογισμός του εσωτερικού γινομένου δυό διανυσμάτων από ένα άτομο ενώ τα διανύσματα παρέχονται από κάποιο άλλο άτομο. Το πρώτο άτομο μπορεί να υπολογίσει το αποτέλεσμα χωρίς να μάθει τις πραγματικές τιμές των διανυσμάτων αυτών του δεύτερου ατόμου.
- Υπολογισμός της Ευκλείδιας Νόρμας ενός διανύσματος από το οποίο κάθε άτομο μιας ομάδας άτομο διαθέτει ένα στοιχείο

Theorem

Επαληθεύσιμος Υπολογισμός *: Όλες οι υπολογιστικές μέθοδοι που επιτρέπουν την επαλήθευση πως η έξοδος του υπολογισμού που εξήγαγαν είναι πράγματι η έξοδος του υπολογισμού και των δεδομένων που τους δόθηκαν.*

Κατηγορίες Ασφαλούς και Επαληθεύσιμου Υπολογισμού

- Ασφαλής Αναθέσιμος Υπολογισμός : Μη διαδραστικά πρωτόκολλα υπολογισμού (π.χ. HE, SWHE, FHE)
- **Ασφαλής Υπολογισμός Πολλών Μερών** : Διαδραστικά πρωτόκολλα υπολογισμού

Ασφαλής Υπολογισμός Πολλών Μερών (Secure Multi Party Computation (SMPC) Protocols

Theorem

Ασφαλής Υπολογισμός Πολλών Μερών (Secure Multi Party Computation ή SMPC) *: Είναι μια κλάση διαδραστικών κρυπτογραφικών σχημάτων που επιτρέπουν σε συμμετέχοντες* $P_1, P_2, \ldots, P_n$ *με αντίστοιχες εισόδους* $x_1, x_2, \ldots, x_n$ *να υπολογίσουν τη συνάρτηση την έξοδο* y *της συνάρτησης* f*, ως* $y = f(x_1, x_2, \ldots, x_n)$ *και ικανοποιούν τις παρακάτω ιδιότητες :*

- *Ορθότητα : Η έξοδος* y *είναι η σωστή έξοδος της συνάρτησης για τις δεδομένες εισόδους*
- *Ιδιωτικότητα : Η έξοδος* y *είναι η μόνη πληροφορία που φανερώνει το πρωτόκολλο σε έναν αντίπαλο που ελέγχει κάποιους διεφθαρμένους συμμετέχοντες.*

Εφαρμογές Ασφαλούς Υπολογισμού Πολλών Μερών :

- Κατανεμημένες ψηφοφορίες και ιδιωτικές ψήφους
- Μηχανική Μάθηση που Διατηρεί την Ιδιωτικότητα (Privacy Preserving Machine Learning)
- Στατιστική που Διατηρεί την Ιδιωτικότητα (Privacy Preserving Statistics)
- Ψηφιακές Υπογραφές Κατωφλίου (Threshold Digital Signatures) με μοναδικό ιδιωτικό κλειδί το οποίο διανέμεται σε κατάλληλη μορφή στους συμμετέχοντες ώστε αν ο κατάλληλος αριθμός συμμετεχόντων συνεργαστεί να μπορεί να παράξει μια υπογραφή χωρίς να χρειάζεται να ανακατασκευαστεί το κλειδί.

Πρωτόκολλο GMW

Πρωτόκολλο GMW

Πρωτόκολλο GMW

BLAS (Basic Linear Algebra Subroutines)

Theorem

BLAS *: Είναι μια διεπαφή/πρότυπο συναρτήσεων και ρουτινών γραμμικής που επιτρέπει σε έναν κατασκευαστή υλικού να δημιουργήσει μια δική του υλοποίηση του προτύπου είναι παραμετροποιημένη για το συγκεκριμένο υλικό. Χωρίζεται σε τρία επίπεδα.*

- *Level-1 : Πράξεις μεταξύ διανυσμάτων*
- *Level-2 : Πράξεις πινάκων με διανύσματα*
- *Level-3 : Πράξεις μεταξύ πινάκων*

Η διεπαφή της BLAS υπάρχει για δύο γλώσσες προγραμματισμού :

- FORTAN και αποκαλείται BLAS
- **C/C++ και αποκαλείται CBLAS**

Η BLAS διεπαφή χρησιμοποιείται σχεδόν από κάθε πρόγραμμα που κάνει πράξεις γραμμικής άλγεβρας, ενδεικτικά αναφέρουμε τα εξής :

- Βιβλιοθήκες επιστημονικού υπολογισμού όπως η NumPy και η Sci-Py
- Βιβλιοθήκες μηχανικής μάθησης όπως η Tensorflow και η PyTorch
- Προγράμματα επιστημονικού υπολογισμού όπως το MATLAB

Παραδείγματα προτύπου CBLAS

```
float   cblas_sdot(const CBLAS_INT N,
                   const float   *X, const CBLAS_INT incX,
                   const float   *Y, const CBLAS_INT incY);
```

```
void cblas_saxpy(const CBLAS_INT N,
                 const float alpha,
                 const float *X, const CBLAS_INT incX,
                 float *Y, const CBLAS_INT incY);
```

Περιγραφή προβλήματος

Οι BLAS βιβλιοθήκες χρησιμοποιούνται σχεδόν παντού, από τον Επιστημονικό Υπολογισμό μέχρι την Μηχανική Μάθηση κσι σε τομείς που η ιδωτικότητα των επεξεργαζόμενων δεδομένων είναι ιδιαίτερα επιθυμητή. Ωστόσο, δεν υπάρχει στη βιβλιογραφία, τουλάχιστον στη δική μας γνώση, κάποια βιβλιοθήκη που να επιτρέπει τη χρήση BLAS συναρτήσεων για Ασφαλή Υπολογισμό Πολλών Μερών και να μπορεί να λειτουργήσει ως σχεδόν drop-in αντικατάστατο μιας κανονικής βιβλιοθήκης.

Επίλυση προβλήματος

Εφαρμογή του SMPC πρωτοκόλλου GMW στην κατασκευή της βιβλιοθήκης MPC-BLAS, μιας Ασφαλούς Υπολογισμού Δύο Μερών BLAS Level-1 βιβλιοθήκης, η οποία κατά τη γνώση μας, παρουσιάζεται πρώτη φορά στη βιβλιογραφία.

Η βιβλιοθήκη MPC-BLAS

Περιγραφή :

- Ένα πρόγραμμα που είναι χρησιμοποιεί τη διεπαφή CBLAS απαιτεί ελάχιστες αλλαγές στον πηγαίο του κώδικα για να χρησιμοποιήσει την διεπαφή MPC-BLAS
- Απαιτεί δύο διεργασίες, μία για κάθε συμμετέχοντα οι οποίες μπορούν να τρέχουν στο ίδιο μηχάνημα ή σε διαφορετικό
- Υποστηρίζει μόνο το λειτουργικό σύστημα Linux, διότι είναι το μοναδικό που υποστηρίζεται από την βιβλιοθήκη ABY. Έχει αναπτυχθεί και ελεγχθεί σε Ubuntu 22.04 LTS
- Υλοποιεί όλες τις Level-1 BLAS ρουτίνες εκτός από τις xROTx, διότι η βιβλιοθήκη ABY έχει πολύ περιορισμένη υποστήριξη για πράξεις πινάκων.
-

Η βιβλιοθήκη MPC-BLAS

Τεχνικές Προδιαγραφές :

- Γραμμένη σε C++20 και χτισμένη με CMake
- Βασισμένη στην βιβλιοθήκη ABY για τις κρυπτογραφικές πράξεις και την εκτέλεση του πρωτοκόλλου GMW.
- Απαιτεί στατική σύνδεση (static linking), λόγω περιορισμών της βιβλιοθήκης ABY

Προσαρμογή προγράμματος από CBLAS σε MPC-BLAS 1

```
float result_1 =
    cblas_sdot(4,
               test_vector_1,1,
               test_vector_2,1);
```

```
mpcblas_initialize(SERVER,
                   "127.0.0.1",
                   7766,
                   bitlen);

float result_1 =
    mpcblas_sdot(4,
                 test_vector_1,1,
                 MPCBLAS_IGNORE,1)
          ->get_value();

mpcblas_uninitialize();
```

Προσαρμογή προγράμματος από CBLAS σε MPC-BLAS 2

```
cblas_saxpy(4,
    cblas_sdot(4,
                test_vector_1,1,
                test_vector_2,1),
            test_vector_1,1,
            test_vector_2,1);
float result =
    cblas_snrm2(4,
                test_vector_2,1);
```

```
mpcblas_initialize(CLIENT,
                   "127.0.0.1",
                   7766,
                   bitlen);
auto result y =
mpcblas_saxpy(4,
    mpcblas_sdot(4,
                 MPCBLAS_IGNORE,1,
                 test_vector_2,1),
             MPCBLAS_IGNORE,1,
             test_vector_2,1);
float result_2 =
    mpcblas_snrm2(4, y, 1)
          ->get_value();
mpcblas_uninitialize();
```

Τέλος

Τέλος

Ευχαριστώ