ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

Διπλωματική Εργασία

Τίτλος Διπλωματικής Εργασίας

Όνομα Επίθετο

Ιωάννινα, Μήνας 2018

Εξεταστική Επιτροπή:

- **Όνομα Επώνυμο**, Καθηγητής, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων (Επιβλέπων)
- **Όνομα Επώνυμο**, Καθηγητής, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων
- **Όνομα Επώνυμο**, Καθηγητής, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Η εργασία αυτή εστίασε στα BFT πρωτόκολλα που όπως αναφέρθηκε παραπάνω τα περισσότερα permissioned blockchains χρησιμοποιούν για μηχανισμό συμφωνίας. Για παράδειγμα, το Hyperledger της εταιρείας IBM [1] χρησιμοποιεί το Practical Byzantine Fault Tolerance (PBFT) [2] για μηχανισμό συμφωνίας. Παρόλο που το PBFT μπορεί να επιτύχει μεγαλύτερη απόδοση από το μηχανισμό του Bitcoin ακόμη δεν μπορεί να φτάσει τις αποδόσεις των υπαρχόντων μεθόδων συναλλαγής (π.χ η Visa διαχειρίζεται δεκάδες χιλιάδες συναλλαγές το δευτερόλεπτο). Επιπλέον το PBFT μπορεί να επεκταθεί μόνο σε μερικές δεκάδες κόμβων αφού ανταλλάσει $O(n^2)$ μηνύματα για την επίτευξη συμφωνίας για μία λειτουργία μεταξύ n διακομιστών. Έτσι, η βελτίωση της επεκτασιμότητας και της απόδοσης των πρωτοκόλλων BFT είναι απαραίτητη για την εξασφάλιση της καθιέρωσης τους σε υπάρχουσες λύσεις blockchain.

Σε αυτή την εργασία, βασιστήκαμε σε μία υπάρχουσα υλοποίηση του BFT, στον αλγόριθμο MinBFT [3]. Είναι μεταγενέστερο του PBFT και βελτιώνει στο ότι χρειάζεται λιγότερους διακομιστές καθώς και ότι ανταλλάσει λιγότερα μηνύματα. Το MinBFT είναι ένας αλγόριθμος βυζαντινής συμφωνίας ο οποίος απαιτεί $2f + 1$ διακομιστές για f ελαττωματικούς (Βυζαντινούς) διακομιστές. Οι BFT αλγόριθμοι τυπικά χρειάζονται $3f + 1$ διακομιστές όμως με την βοήθεια ενός μηχανισμού ασφαλείας που υπάρχει σε κάθε υπολογιστή που εκτελεί το MinBFT ο αριθμός των διακομιστών μπορεί να πέσει απο $3f + 1$ σε $2f + 1$. Αυτός ο μηχανισμός ασφαλείας πρέπει να λειτουργεί σωστά ακόμα και αν όλο το σύστημα έχει παραβιαστεί. Το βασικό πρόβλημα είναι ότι όλοι οι διακομιστές θα πρέπει να εκτελέσουν την ίδια ακολουθία λειτουργιών. Ο μηχανισμός ασφαλείας θα πρέπει

να παρέχει μια υπηρεσία η οποία θα ορίζει την ακολουθία των λειτουργιών που εκτελούν, με τέτοιο τρόπο όπου ένας κακόβουλος διακομιστής δεν θα μπορεί να κάνει τους άλλους σωστούς διακομιστές να εκτελέσουν μια άλλη λειτουργία αντί για την κ-οστη λειτουργία. Αυτό μπορεί να υλοποιηθεί με την βοήθεια ενός έμπιστου μονοτονικού μετρητή που θα αντιστοιχίζει ακολουθιακούς αριθμούς σε κάθε λειτουργία. Στην αρχική έκδοση του MinBFT [3] είχε χρησιμοποιηθεί ο μηχανισμός ασφαλείας Atmel TPM 1.2 για την υλοποίηση της υπηρεσίας USIG (βλ. ενότητα 2.7) η οποία είναι υπεύθυνη για την δημιουργία του μονοτονικού μετρητή.

1.1 Στόχοι

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

1.2 Δομή της Διπλωματικής Εργασίας

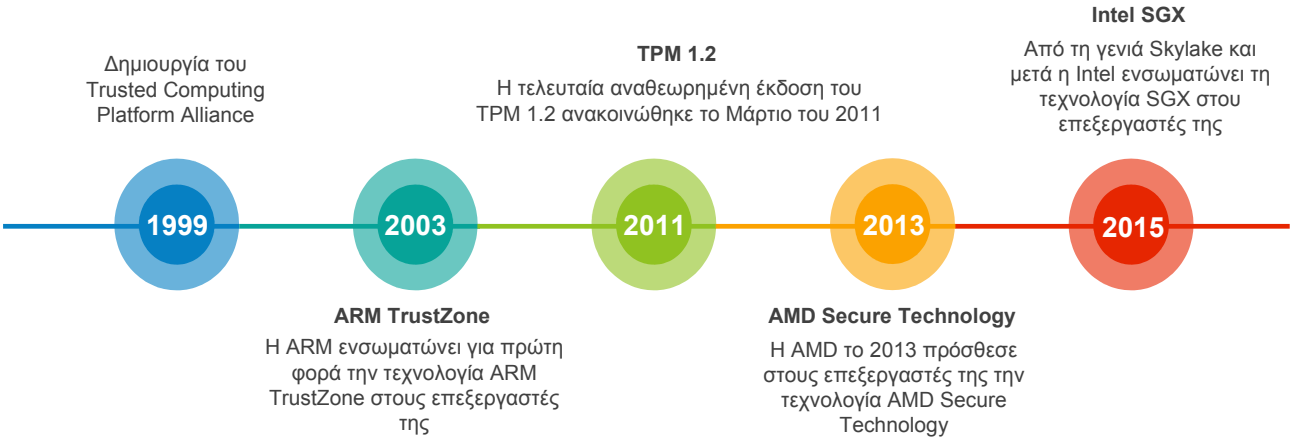
Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Κεφάλαιο 2

Υπόβαθρο

2.1 Προστατευόμενο Περιβάλλον Εκτέλεσης

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε



Σχήμα 2.1: Ιστορική εξέλιξη των μηχανισμών ασφαλείας υλικού

2.2 Intel Software Guard Extensions - SGX

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest

gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language [4].

2.2.1 Enclave

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: To Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

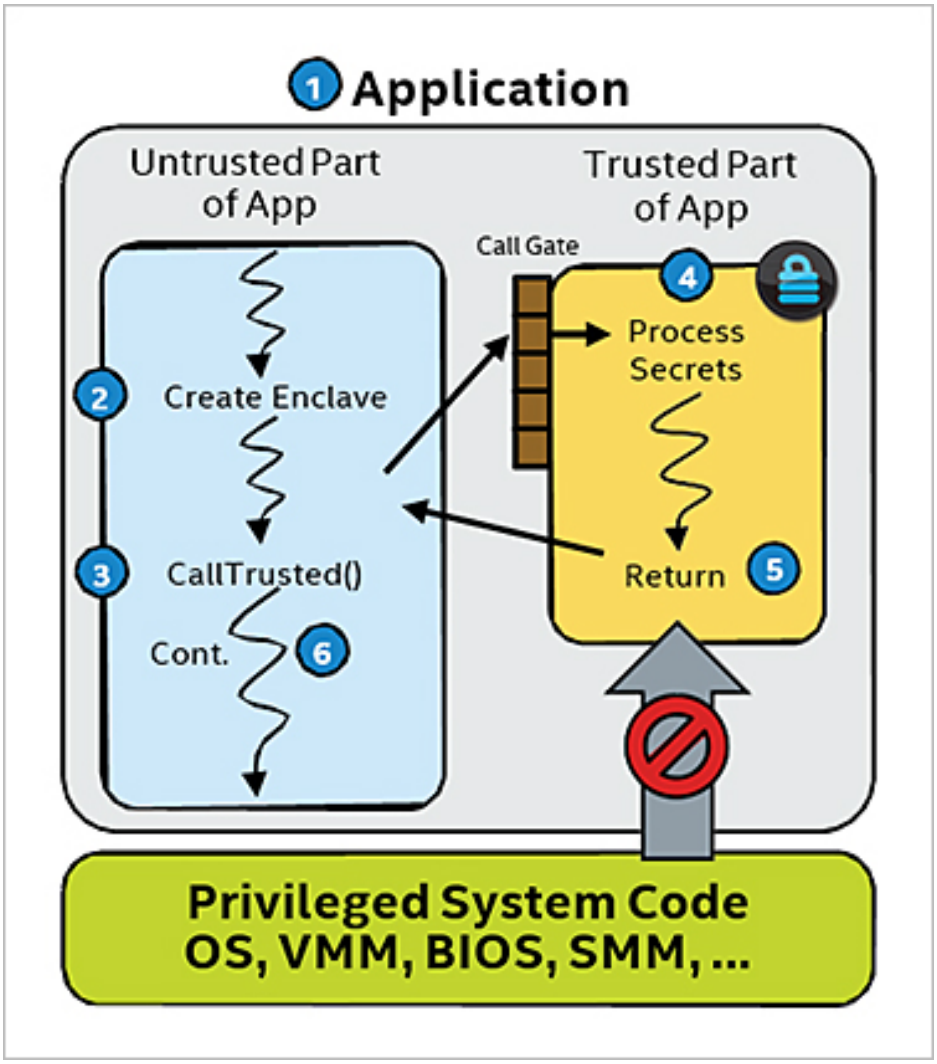
Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: To Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: To Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: To Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: To Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: To Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.



Σχήμα 2.2: Τα βήματα που ακολουθούνται στο enclave κατά τη διάρκεια εκτέλεσης. Πηγή: software.intel.com

1. Η εφαρμογή δημιουργείται με trusted και untrusted μέρη
2. Η εφαρμογή εκτελείται και δημιουργεί το enclave, το οποίο είναι σε ασφαλή απομονωμένη μνήμη
3. Καλείται η trusted μέθοδος και η εκτέλεση μεταφέρεται στο enclave
4. Το enclave βλέπει όλα τα δεδομένα τη διεργασίας, η πρόσβαση από έξω στα δεδομένα του enclave απαγορεύεται.
5. Η trusted function επιστρέφει τα δεδομένα του enclave
6. Η εφαρμογή συνεχίζει την κανονική της εκτέλεση

2.2.2 Attestation

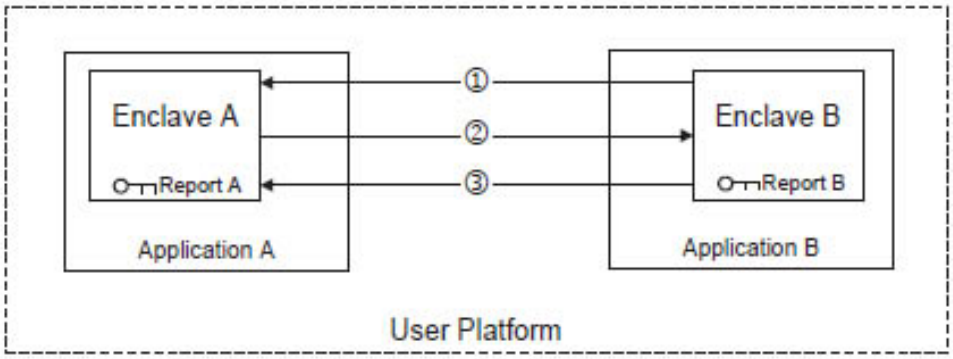
Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: Το Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το

SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

Local attestation

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: Το Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: Το Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.



Σχήμα 2.3: Διαδικασία τοπικής βεβαίωσης (Local attestation). Πηγή: software.intel.com

Το σχήμα 2.3 δείχνει ένα παράδειγμα ροής για το πως δύο enclave στην ίδια πλατφόρμα πιστοποιούν το ένα το άλλο.

1. Η εφαρμογή Α έχει το enclave Α και η εφαρμογή Β έχει το enclave Β. Αφού πρώτα η κάθε εφαρμογή δημιούργησε μια διαδρομή επικοινωνίας με το enclave της, το enclave Β στέλνει την ταυτότητα (MRENCLAVE) του στο enclave Α.

2. Το enclave Α ζητάει από το hardware να δημιουργήσει μία αναφορά προορισμένη για το enclave Β χρησιμοποιώντας το MRENCLAVE που έλαβε από το enclave Β. Το enclave Α στέλνει την αναφορά του στο enclave Β μέσω της untrusted εφαρμογής. Σαν μέρος της αναφοράς, το enclave Α μπορεί επίσης να στείλει δεδομένα στον Β.

3. Μόλις λάβει το enclave Β την αναφορά απο το enclave Α, το enclave Β ζητάει από το hardware να πιστοποιήσει την αναφορά για να επιβεβαιώσει το enclave Β ότι το enclave Α είναι στην ίδια πλατφόρμα με αυτό. Το enclave Β μπορεί τώρα να απαντήσει χρησιμοποιώντας την αναφορά του enclave Α, χρησιμοποιώντας την

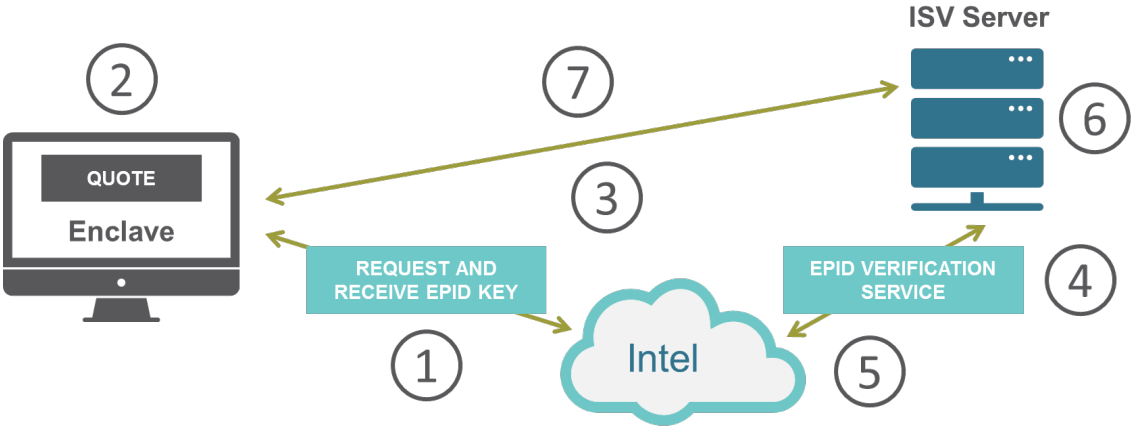
MRENCLAVE τιμή από την αναφορά που μόλις έλαβε. Το enclave B στέλνει την αναφορά του στο enclave A.

4. Το enclave A τότε πιστοποιεί την αναφορά για να επιβεβαιωθεί ότι το enclave B ανήκει στην ίδια πλατφόρμα με το enclave A.

Remote attestation

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: Το Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: Το Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.



Σχήμα 2.4: Διαδικασία απομακρυσμένης βεβαίωσης (Remote attestation)

1. Μετά την εγκατάσταση του λογισμικού Intel SGX Platform (PSW) στον υπολογιστή-πελάτη, το PSW ζητά από την υπηρεσία παροχής υπηρεσιών SGX της Intel να παράσχει το κλειδί EPID (Enhanced Privacy ID) του υπολογιστή. Αν η ανταλλαγή πρωτοκόλλων μεταξύ της υπηρεσίας Intel SGX PSW και της υπηρεσίας παροχής υπηρεσιών Intel είναι επιτυχής - κάτι που απαιτεί γνήσια CPU εξοπλισμένη με Intel SGX, τότε η συσκευή ή ο διακομιστής θα έχει το κλειδί EPID.

2. Το enclave παράγει ένα quote. Ένα quote είναι απλώς μια υπογεγραμμένη αναφορά των τρεχουσών τιμών του Platform Configuration Registers (PCRs) στον υπολογιστή. Ο επεξεργαστής του υπολογιστή που τρέχει το enclave δημιουργεί αυτομάτως αυτό το quote.

3. Ο υπολογιστής που τρέχει το enclave στέλνει τότε το quote στον ανεξάρτητο προμηθευτή λογισμικού (Independent software vendor, ISV) που έχει ένα μυστικό που χρειάζεται η εφαρμογή του enclave.

4. Ο ISV με τη σειρά του στέλνει το quote στην Intel για επαλήθευση.

5. Η Intel επαληθεύει το quote για τον ISV. Ωστόσο, η Intel δεν παρέχει άλλες πληροφορίες (όπως ποιος δημιούργησε το quote). Στην περίπτωση αυτή, η Intel είναι σαν συμβολαιογράφος: το μόνο που μπορεί να πει είναι ότι το κρυπτογραφικά υπογεγραμμένο quote προέρχεται από έγκυρο enclave, αλλά τίποτα περισσότερο.

6. Στη συνέχεια, ο ISV είναι υπεύθυνος για την επιθεώρηση του quote. Για παράδειγμα, το ISV μπορεί να επιβεβαιώσει ότι το κλειδί είναι δικό του.

7. Αφού ο ISV ικανοποιηθεί με το quote, ο διακομιστής του ISV και το enclave μπορούν τώρα να παράξουν ένα κλειδί κρυπτογράφησης από το quote. Αυτό είναι ένα συμμετρικό κλειδί μεταξύ του διακομιστή και του enclave που χρησιμοποιεί τον αλγόριθμο Diffie-Hellman.

Αυτό που επιτρέπει στους επεξεργαστές που είναι εξοπλισμένοι με το SGX να παρέχουν αυτές τις ισχυρές εγγυήσεις βασίζεται στα δύο ακόλουθα hardware: Το Processor Reserved Memory (PRM) που είναι ένα συνεχές μπλοκ μνήμης που προορίζεται για το SGX, απρόσιτο από το μη αξιόπιστο λογισμικό και ακόμη και από το υλικό. Η λειτουργία Enclave είναι μια λειτουργία υπό την οποία ένας επεξεργαστής αποκτά πρόσβαση στο PRM.

2.2.3 Monotonic counters

Μία από τις λειτουργίες που παρέχει η Intel SGX, η οποία είναι ιδιαίτερα σημαντική για αυτήν την εργασία, είναι η πρόσβαση σε αξιόπιστους μονοτονικούς μετρητές. Όπως υποδεικνύεται από το όνομα, οι μονοτονικοί μετρητές είναι ακέραιοι μετρητές, οι οποίοι μπορούν μόνο να αυξηθούν.

Μία από τις λειτουργίες που παρέχει η Intel SGX, η οποία είναι ιδιαίτερα σημαντική για αυτήν την εργασία, είναι η πρόσβαση σε αξιόπιστους μονοτονικούς μετρητές. Όπως υποδεικνύεται από το όνομα, οι μονοτονικοί μετρητές είναι ακέραιοι μετρητές, οι οποίοι μπορούν μόνο να αυξηθούν.

Το Intel SGX παρέχει τέσσερις κλήσεις σχετικές με τον μονοτονικό μετρητή:

- Η κλήση $sgx\_create\_monotonic\_counter$ δημιουργεί έναν μονοτονικό μετρητή με την προεπιλεγμένη πολιτική κατόχου 0x1, που σημαίνει ότι τα enclaves με το ίδιο κλειδί υπογραφής έχουν πρόσβαση στον μονοτονικό μετρητή.
- Η κλήση $sgx\_increment\_monotonic\_counter$ αυξάνει την τιμή ενός μετρητή κατά 1.
- Η κλήση $sgx\_read\_monotonic\_counter$ επιστρέφει την τιμή του μετρητή.
- Η κλήση $sgx\_destroy\_monotonic\_counter$ καταστρέφει τον μετρητή.

2.3 Blockchain

Μία από τις λειτουργίες που παρέχει η Intel SGX, η οποία είναι ιδιαίτερα σημαντική για αυτήν την εργασία, είναι η πρόσβαση σε αξιόπιστους μονοτονικούς μετρητές. Όπως υποδεικνύεται από το όνομα, οι μονοτονικοί μετρητές είναι ακέραιοι μετρητές, οι οποίοι μπορούν μόνο να αυξηθούν.

2.3.1 Τι είναι Blockchain

Ένα blockchain είναι μια ανοιχτή βάση δεδομένων που διατηρεί ένα κατανεμημένο λογιστικό κατάλογο που συνήθως αναπτύσσεται μέσα σε ένα δίκτυο peer-to-peer. Αποτελείτε από μια συνεχώς αυξανόμενη λίστα εγγραφών που ονομάζονται block, τα οποία περιέχουν συναλλαγές. Τα block προστατεύονται από παραβιάσεις με την χρήση κρυπτογραφικών hashes και με μηχανισμό συμφωνίας.

Ένα blockchain είναι μια ανοιχτή βάση δεδομένων που διατηρεί ένα κατανεμημένο λογιστικό κατάλογο που συνήθως αναπτύσσεται μέσα σε ένα δίκτυο peer-to-peer. Αποτελείτε από μια συνεχώς αυξανόμενη λίστα εγγραφών που ονομάζονται block, τα οποία περιέχουν συναλλαγές. Τα block προστατεύονται από παραβιάσεις με την χρήση κρυπτογραφικών hashes και με μηχανισμό συμφωνίας.

Ένα blockchain είναι μια ανοιχτή βάση δεδομένων που διατηρεί ένα κατανεμημένο λογιστικό κατάλογο που συνήθως αναπτύσσεται μέσα σε ένα δίκτυο peer-to-peer. Αποτελείτε από μια συνεχώς αυξανόμενη λίστα εγγραφών που ονομάζονται block, τα οποία περιέχουν συναλλαγές. Τα block προστατεύονται από παραβιάσεις με την χρήση κρυπτογραφικών hashes και με μηχανισμό συμφωνίας.

Ένα blockchain είναι μια ανοιχτή βάση δεδομένων που διατηρεί ένα κατανεμημένο λογιστικό κατάλογο που συνήθως αναπτύσσεται μέσα σε ένα δίκτυο peer-to-peer. Αποτελείτε από μια συνεχώς αυξανόμενη λίστα εγγραφών που ονομάζονται block, τα οποία περιέχουν συναλλαγές. Τα block προστατεύονται από παραβιάσεις με την χρήση κρυπτογραφικών hashes και με μηχανισμό συμφωνίας.

Ένα blockchain είναι μια ανοιχτή βάση δεδομένων που διατηρεί ένα κατανεμημένο λογιστικό κατάλογο που συνήθως αναπτύσσεται μέσα σε ένα δίκτυο peer-to-peer. Αποτελείτε από μια συνεχώς αυξανόμενη λίστα εγγραφών που ονομάζονται block, τα οποία περιέχουν συναλλαγές. Τα block προστατεύονται από παραβιάσεις με την χρήση κρυπτογραφικών hashes και με μηχανισμό συμφωνίας.

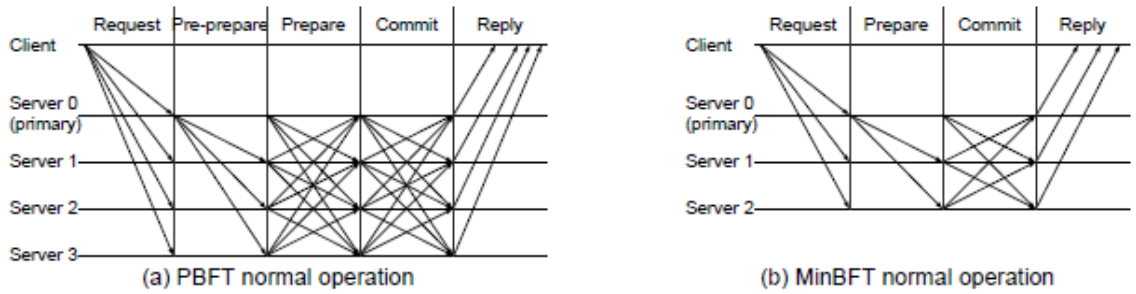
2.3.2 Το πρωτόκολλο BFT στην τεχνολογία Blockchain

Η αναπαραγωγή του blockchain πάσχει από βραχυπρόθεσμη ασυνέπεια. Ακόμα κι αν υποθέσουμε άπειρη εκτέλεση και είμαστε πρόθυμοι να υποθέσουμε ότι ένα μπλοκ που είναι θαμμένο από κρυπτογραφικά παζλ αξίας μιας ώρας είναι απολύτως ασφαλές, αυτό σημαίνει ότι πρέπει να περιμένουμε μια ώρα! Στην πραγματικότητα ένα από τα μεγαλύτερα πλεονεκτήματα των πρωτοκόλλων BFT είναι ότι παρέχουν ”άμεση οριστικότητα”.

Η αναπαραγωγή του blockchain πάσχει από βραχυπρόθεσμη ασυνέπεια. Ακόμα κι αν υποθέσουμε άπειρη εκτέλεση και είμαστε πρόθυμοι να υποθέσουμε ότι ένα μπλοκ που είναι θαμμένο από κρυπτογραφικά παζλ αξίας μιας ώρας είναι απολύτως ασφαλές, αυτό σημαίνει ότι πρέπει να περιμένουμε μια ώρα! Στην πραγματικότητα ένα από τα μεγαλύτερα πλεονεκτήματα των πρωτοκόλλων BFT είναι ότι παρέχουν ”άμεση οριστικότητα”.

2.4 Πρωτόκολλο BFT - Byzantine fault tolerance

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε



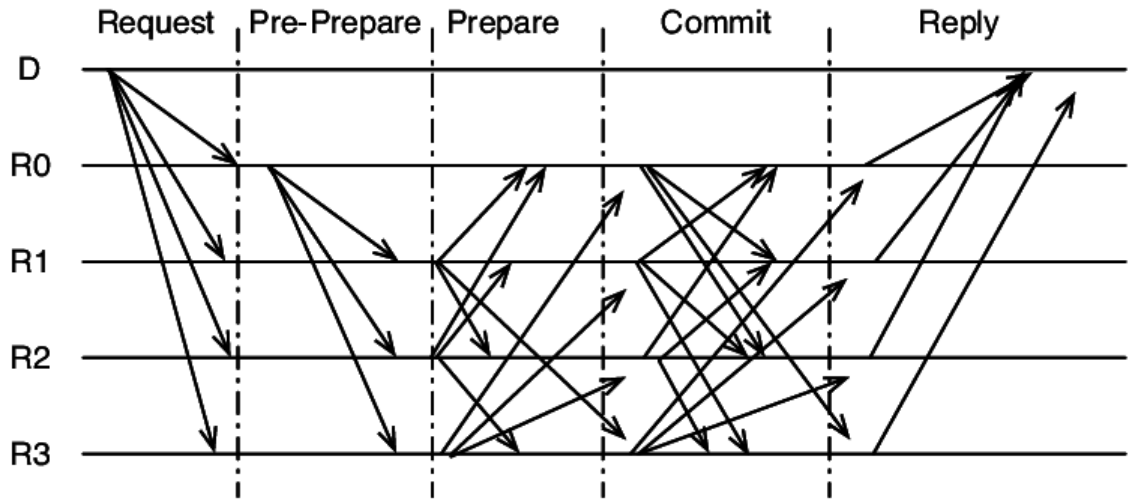
Σχήμα 2.5: Ακολουθία μηνυμάτων των PBFT και MinBFT (Figure 1 από [3]).

καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Οι αλγόριθμοι BFT τυπικά απαιτούν $3f + 1$ servers (ή replicas) για να ανεχθούν f βυζαντινούς (ή ελαττωματικούς) διακομιστές. Βασίζονται στην ιδέα ότι τα σωστά αντίγραφα μπορούν να ξεπεράσουν τα ελαττωματικά αντίγραφα με μια σειρά ψήφων και, για να συμβεί αυτό, θα χρειαστούν τουλάχιστον $2f + 1$ σωστά αντίγραφα, σε σύνολο εξυπηρετητών $3f + 1$. Ωστόσο, αυτοί οι αλγόριθμοι απαιτούν περισσότερους διακομιστές από αυτό το ελάχιστο.

2.5 Practical BFT - PBFT

Το Practical Byzantine Fault Tolerance (PBFT) είναι ένας αλγόριθμος BFT που δημοσιεύθηκε το 1999 στην εργασία ”Practical Byzantine Fault Tolerance” [2]. Στοχεύει να βελτιώσει τον αυθεντικό BFT μηχανισμό συμφωνίας και υλοποιήθηκε σε πολλά μοντέρνα κατανεμημένα υπολογιστικά συστήματα, συμπεριλαμβανομένων και μερικών γνωστών blockchain. Το PBFT απαιτεί $3f + 1$ servers (ή replicas) για να ανεχθεί f βυζαντινούς (ή ελαττωματικούς) διακομιστές. Ουσιαστικά, όλοι οι κόμβοι του μοντέλου PBFT οργανώνονται σε μια ακολουθία με έναν κόμβο να είναι ο πρωτεύον κόμβος (leader) και οι άλλοι να είναι εφεδρικοί κόμβοι. Όλοι οι κόμβοι του συστήματος επικοινωνούν μεταξύ τους και ο στόχος είναι η επίτευξη συμφωνίας για την κατάσταση του συστήματος μέσω πλειοψηφίας.



Σχήμα 2.6: Κανονική λειτουργία του PBFT

Κάθε γύρος στο PBFT για την επίτευξη συμφωνίας αποτελείτε από 4 φάσεις. Αυτό το μοντέλο είναι περισσότερο «Διοικητή και υποδιοικητή» από ένα καθαρό πρόβλημα βυζαντινών στρατηγών (BFT), όπου όλοι οι στρατηγοί είναι ίσοι, λόγω της παρουσίας ενός κόμβου ηγέτη. Οι φάσεις είναι οι εξής:

1. Ένας client στέλνει ένα αίτημα στον πρωτεύον διακομιστή.
2. Ο πρωτεύον κόμβος διανέμει το αίτημα στους εφεδρικούς κόμβους.
3. Οι κόμβοι εκτελούν το αίτημα και στη συνέχεια στέλνουν μια απάντηση στον client.
4. Ο client αναμένει $f + 1$ (το f αντιπροσωπεύει το μέγιστο αριθμό κόμβων που ενδέχεται να είναι ελαττωματικοί) από διαφορετικούς κόμβους.

Οι τρεις φάσεις είναι οι pre-prepare, prepare, και commit. Οι φάσεις pre-prepare και prepare χρησιμοποιούνται για να διατάξουν αιτήματα που στάλθηκαν στον ίδιο γύρο ακόμα και αν ο πρωτεύων διακομιστής είναι ελαττωματικός, που σημαίνει η διάταξη των αιτημάτων είναι λανθασμένη. Οι φάσεις prepare και commit χρησιμοποιούνται για να βεβαιωθεί ότι τα αιτήματα που γίνονται commit είναι πλήρως διατεταγμένα σε κάθε γύρο.

Είναι απαραίτητο οι κόμβοι να είναι ντετερμινιστικοί και να ξεκινούν από τη ίδια κατάσταση. Το τελικό αποτέλεσμα είναι όλοι οι ειλικρινής κόμβοι να έρθουν σε συμφωνία, είτε αποδέχονται είτε απορρίπτουν ένα αίτημα.

2.6 MinBFT

Το MinBFT, είναι ένας $2f + 1$ αλγόριθμος συμφωνίας με βυζαντινά σφάλματα (BFT). Ο αλγόριθμος MinBFT ακολουθεί ένα πρότυπο ανταλλαγής μηνυμάτων παρόμοιο με το PBFT (βλ. Σχήμα 2.5). Οι διακομιστές οργανώνονται σε ομάδες που ονομάζονται $views$. Κάθε $view$ έχει έναν πρωτεύον διακομιστή και όλοι οι υπόλοιποι είναι εφεδρικά αντίγραφα (backups). Οι clients είναι αυτοί που δημιουργούν τα αιτήματα που ενσωματώνουν κάποιες λειτουργίες για να εκτελεστούν από τους διακομιστές.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το

πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Clients. Ένας client c για να ζητήσει την εκτέλεση μιας λειτουργίας op στέλνει ένα μήνυμα της μορφής <REQUEST, c, seq, op> σ_c σε όλους τους διακομιστές. Ο αριθμός seq είναι ο αναγνωριστικός κωδικός της αίτησης που χρησιμοποιείται για να εξασφαλιστεί η μοναδικότητα. Οι διακομιστές αποθηκεύουν σε ένα πίνακα V_{req} τον αριθμό seq του τελευταίου αιτήματος που έχουν εκτελέσει για κάθε πελάτη, απορρίπτουν αιτήματα με αναγνωριστικό κωδικό seq μικρότερο από το τελευταίο εκτελεσμένο (για να αποφευχθεί η εκτέλεση του ίδιου αιτήματος δύο φορές) και τέλος τυχόν αιτήματα που λαμβάνονται όσο το προηγούμενο είναι υπό επεξεργασία. Τα αιτήματα υπογράφονται με το ιδιωτικό κλειδί του client και αιτήματα με μη έγκυρη υπογραφή c απλά απορρίπτονται. Μετά από την αποστολή ενός αιτήματος, ο client περιμένει για $f + 1$ απαντήσεις της μορφής <REPLY, s, seq, res> από διαφορετικούς διακομιστές s με αντίστοιχα αποτελέσματα res, γεγονός που εξασφαλίζει ότι τουλάχιστον μία απάντηση θα προέρχεται από σωστό διακομιστή. Στην περίπτωση που ο client δεν λάβει αρκετές απαντήσεις κατά τη διάρκεια ενός χρονικού διαστήματος που διαβάζεται από το τοπικό ρολόι του, ξαναστέλνει το αίτημα. Σε περίπτωση που το αίτημα αυτό έχει ήδη υποβληθεί σε επεξεργασία, οι διακομιστές θα ξαναστείλουν την αποθηκευμένη απάντηση τους.

Διακομιστές. Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Η ιδέα είναι ότι ένα αίτημα m στέλνεται από το πρωτεύον διακομιστή s_i σε όλους τους διακομιστές σε ένα μήνυμα τύπου $< PREPARE, v, s_i, m, UI_i >$, και κάθε server s_j το ξαναστέλνει σε όλους τους άλλους σε ένα μήνυμα τύπου $< COMMIT, v, s_j, s_i, m, UI_i, UI_j >$, όπου το UI_j λαμβάνεται καλώντας το $createUI$. Κάθε μήνυμα που αποστέλλεται, είτε είναι $PREPARE$ είτε $COMMIT$, έχει ως εκ τούτου ένα μοναδικό αναγνωστικό UI που αποκτάται καλώντας τη συνάρτηση $createUI$, έτσι ώστε να μην υπάρχουν δύο μηνύματα με το ίδιο αναγνωριστικό. Οι διακομιστές ελέγχουν εάν τα αναγνωριστικά των μηνυμάτων που λαμβάνουν ισχύουν για αυτά τα μηνύματα, χρησιμοποιώντας μια λειτουργία επαλήθευσης.

Στην περίπτωση που ένας διακομιστής s_k δεν έλαβε κάποιο μήνυμα PREPARE αλλά έλαβε ένα μήνυμα COMMIT με έγκυρο αναγνωριστικό, τότε στέλνει το δικό του COMMIT. Αυτό μπορεί να συμβεί αν ο αποστολέας είναι ελαττωματικός και δεν στέλνει το μή-

νυμα PREPARE στον server s_k (αλλά το στέλνει σε άλλους διακομιστές) ή εάν το μήνυμα PREPARE απλώς καθυστερεί και λαμβάνεται μετά τα μηνύματα COMMIT. Ένα αίτημα m γίνεται αποδεκτό από έναν διακομιστή που εκτελεί τον αλγόριθμο εάν ο διακομιστής λάβει $f + 1$ έγκυρα μηνύματα $COMMIT$ από διαφορετικούς διακομιστές για το αίτημα m.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

2.7 Η υπηρεσία USIG

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος. Το interface της υπηρεσίας έχει δύο λειτουργίες:

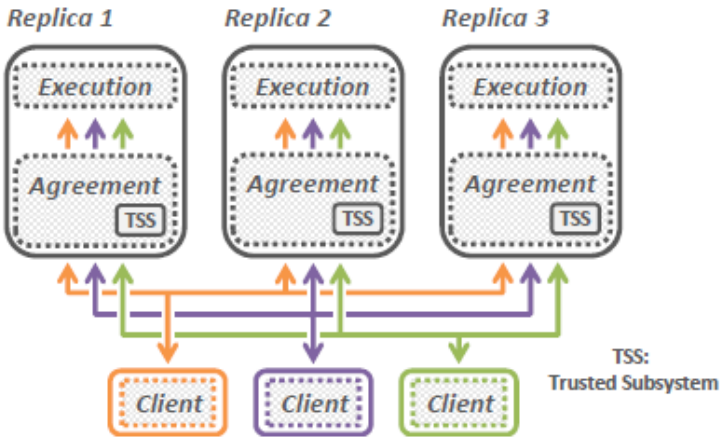
- createUI (m) - επιστρέφει ένα πιστοποιητικό USIG που περιέχει ένα μοναδικό αναγνωριστικό UI και πιστοποιεί ότι αυτό το UI δημιουργήθηκε από την αξιόπιστη και απαραβίαστη μονάδα για το μήνυμα m. Το μοναδικό αναγνωριστικό είναι ουσιαστικά μια ανάγνωση του μονοτονικού μετρητή, ο οποίος αυξάνεται κάθε φορά που καλείται η $createUI$.

- verifyUI (PK, UI, m) - επαληθεύει εάν το μοναδικό αναγνωριστικό UI είναι έγκυρο για το μήνυμα m, δηλαδή εάν το πιστοποιητικό USIG ταιριάζει με το μήνυμα και τα υπόλοιπα δεδομένα στο UI.

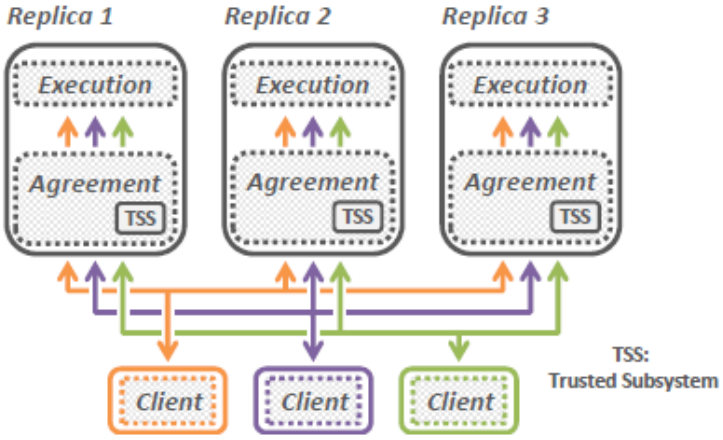
Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

2.8 Σύγκριση με άλλες ερευνητικές εργασίες

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

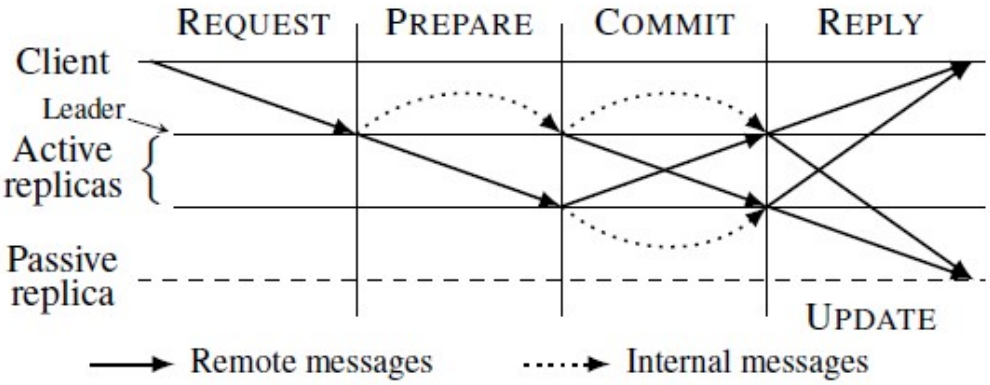


Σχήμα 2.7: Hybrid BFT state-machine replication (Figure 1 από [6])



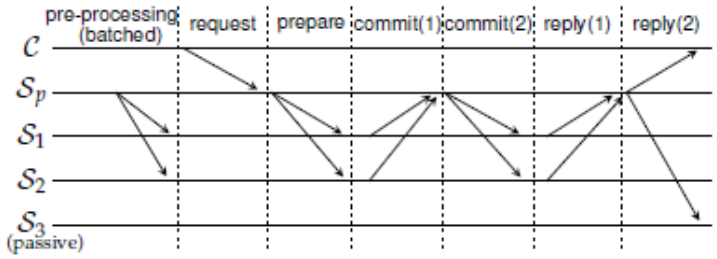
Σχήμα 2.8: Η ιδέα της παραλληλοποίησης του υβριδικού πρωτοκόλλου (Figure 2 από [6])

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.



Σχήμα 2.9: Ακολουθία μηνυμάτων του CheapBFT (Figure 4 από [8]).

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.



Σχήμα 2.10: Ακολουθία μηνυμάτων του FastBFT (Figure 2 από [9]).

Ομοίως, το FastBFT [9] Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Κεφάλαιο 3

Υλοποίηση

Στην εργασία αυτή βασίστηκα σε κώδικα ο οποίος είχε αναπτυχθεί για την εργασία Efficient Byzantine Fault Tolerance [10]. Στην εργασία αυτή είχε αναπτυχθεί ο κώδικας για τον αλγόριθμο MinBFT, που αναφέραμε στην ενότητα 2.6. Το MinBFT για την λειτουργία του χρησιμοποιεί το interface USIG (βλ. ενότητα 2.7). Στην δουλεία εκείνη για την ανάπτυξη του USIG χρησιμοποιήθηκε η τεχνολογία Trusted Platform Module (TPM).

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

Υπενθυμίζουμε ότι ένα blockchain απαιτεί ένα μηχανισμό για την επίτευξη κατανεμημένης συμφωνίας ή την επικύρωση και την κανονικοποίηση ενός μόνο καταλόγου. Το πρωτόκολλο BFT αναφέρεται στο χαρακτηριστικό των κατανεμημένων συστημάτων που τους επιτρέπει να φτάσουν σε συμφωνία ενάντια στα βυζαντινά σφάλματα, δηλαδή σε καταστάσεις όπου τα συστατικά του συστήματος θα αποτύχουν - αλλά όχι μόνο να αποτύχουν - οι εσφαλμένοι βυζαντινοί κόμβοι θα ενεργήσουν αυθαίρετα και συχνά παρουσιάζουν αντικρουόμενες πληροφορίες σε διαφορετικούς κόμβους του συστήματος.

3.1 Ενεργοποιώντας το Intel SGX

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς

PSW)[11].

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

3.2 Trusted Monotonic Counter

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

Στο **LibSgxJni** η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

```
enclave {
    from "sgx_tae_service.edl" import *;

    /* enum definition */
    enum TEE_ERROR {
        TEE_ERROR_INVALID_SIGNATURE = 0,
        TEE_ERROR_INVALID_COUNTER = 1,
        TEE_ERROR_INVALID_SECRET = 2
    };

    trusted {
        /* define ECALLs here. */
        public uint32_t create_counter(void);
        public uint32_t read_counter([out] uint32_t* ctr);
        public uint32_t increment_counter(void);
        public uint32_t destroy_counter(void);
    };

};
```

Snippet 3.1: *Enclave.edl*, η διεπαφή μεταξύ του enclave και του αναξιόπιστου κώδικα

Η δομή που δημιουργήθηκε για τα δεδομένα μονοτονικού μετρητή ονομάζεται monotonic counter (Snippet 3.1) και είναι προσβάσιμη μόνο από το $enclave$. Αποτελείται από την τρέχουσα τιμή του μετρητή, καθώς και το μοναδικό αναγνωριστικό του μονοτονικού μετρητή, τα οποία χρησιμοποιεί το SGX για την πρόσβαση στην κατάλληλη μνήμη.

```
typedef struct _monotonic_counter
{
sgx_mc_uuid_t mc;
uint32_t mc_value;
} monotonic_counter;
```

Snippet 3.2: Η δομή monotonic_counter στο Enclave.cpp

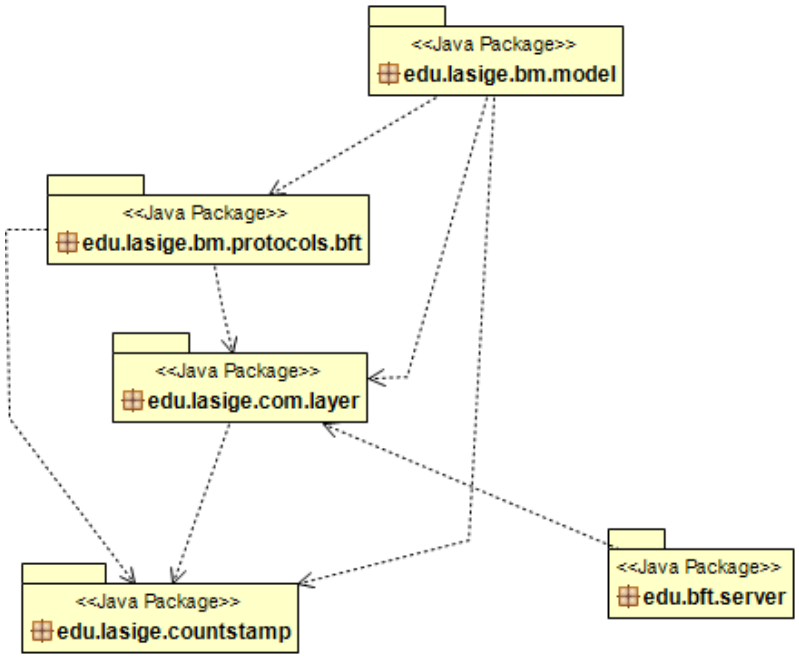
Κατά τη δημιουργία και την πρόσβαση στους μονοτονικούς μετρητές, το SGX πρέπει να δημιουργήσει μια σύνδεση με το Platform Services Enclave (PSE), το οποίο είναι ένα προστατευόμενο enclave που είναι προσβάσιμο μόνο από άλλα τοπικά enclaves. Το PSE παρέχει πρόσβαση σε προστατευμένες λειτουργίες, όπως μονότονοι μετρητές, σφράγιση και βεβαίωση. Μετά την καθιέρωση αυτής της σύνδεσης, η δημιουργία ενός μονοτονικού μετρητή είναι απλά μια κλήση στο $sgx\_create\_monotonic\_counter$. Όταν έχει δημιουργηθεί ο μονοτονικός μετρητής, το struct είναι σφραγισμένο και διαβιβάζεται στον μη αξιόπιστο κώδικα. Εδώ είναι σημαντικό να σημειώσουμε ότι η μνήμη που περιέχει τα μη σφραγισμένα δεδομένα πρέπει να καθαριστεί πριν επιστρέψουν τα σφραγισμένα δεδομένα στον μη αξιόπιστο κώδικα, για να αποφευχθεί η διαρροή δεδομένων.

Η αύξηση ενός μονοτονικού μετρητή είναι πολύ παρόμοια με τη δημιουργία του. Λόγω του γεγονότος ότι ο μη αξιόπιστος κώδικας έχει μόνο πρόσβαση σε σφραγισμένες δομές, τα δεδομένα πρέπει πρώτα να αποσφραγιστούν. Αυτό επιτυγχάνεται με μια κλήση στο sgx_unseal_data. Επίσης, τα δεδομένα επαληθεύονται (δηλ. Ότι η τιμή του μονοτονικού μετρητή είναι όπως αναμενόταν), για να διασφαλιστεί ότι δεν έχουν παραποιηθεί τα δεδομένα. Η επαλήθευση δεν εγγυάται ότι μια άλλη οντότητα του enclave δεν έχει αποσφραγίσει τα δεδομένα και έχει αυξήσει τον μετρητή. Ωστόσο, πρέπει να είναι μια άλλη οντότητα του ακριβώς ίδιου enclave, καθώς η διαδικασία σφράγισης είναι κρυπτογραφημένη με ένα Seal Key, το οποίο είναι μοναδικό για το enclave και την πλατφόρμα[12].

3.3 MinBFT-SGX

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

Το package **edu.lasige.com.layer** είναι υπεύθυνο για την επικοινωνία των διακομιστών και την διαχείρηση των μυνημάτων που λαμβάνει ο κάθε διακομιστής. Οι δύο βασικές κλάσεις μεσα στο πακέτο ειναι η $CommunicationSystem$ και η $BatchControl$. Η κλάση $CommunicationSystem$ αναλαμβάνει την λήψη και την μετάδοση των μυ-



Σχήμα 3.1: Σχεδιάγραμμα των βασικών πακέτων του MinBFT.

νημάτων, από και προς το $BatchControl$. Στην κλάση $BatchControl$ προωθείται καθε μήνυμα που λαμβάνει ο διακομιστής και ανάλογα με τον τύπο του μηνύματος κάνει τις αντίστοιχες λειτουργίες.

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

3.4 Πώς τρέχει το MinBFT-SGX

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11]. Αφού έχουμε κάνει όλα τα παραπάνω, εκτελούμαι την παρακάτω εντολή σε

κάθε διακομιστή:

```
java -cp dist/LASIGE-BFT-Protocols.jar:lib/* edu.bft.server.
   ServerCS [server id] [protocol] [size] [delay attack] [
   batchSize]
```

Το όρισμα $[serverid]$ δηλώνει το αναγνωριστικό του διακομιστή, το $[protocol]$ είναι κάθε φορά MinBFT, το $[size]$ δηλώνει το μέγεθος του μηνύματος που στέλνει για απάντηση ο διακομιστής, στο $[delayattack]$ αν δώσουμε τιμή μεγαλύτερη από το 0 δηλώνει σε πόσο χρόνο αυτός ο διακομιστής θα καταρρεύσει και τέλος το $[batchSize]$ το οποίο δηλώνει πόσα μηνύματα θα επιβεβαιώνει ταυτόχρονα.

3.5 Προγράμματα Αξιολόγησης (Benchmarks)

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

Σύνοψη προγραμμάτων για benchmark	
Πρόγραμμα	Περιγραφή
SgxTimeTest	Υπολογισμός μέσης διάρκειας για την αύξηση ενός μονοτονικού μετρητή με το Intel SGX
ThroughputClient	Μέτρηση του throughput των διακομιστών
ClientThread	Μέτρηση της μέσης καθυστέρησης για ένα σύνολο μηνυμάτων
LatencyClient	Μέτρηση της μέσης καθυστέρησης για ένα μήνυμα

Πίνακας 3.1: Σύνοψη και περιγραφή των προγραμμάτων αξιολόγησης που χρησιμοποιήθηκαν

3.5.1 Πώς τρέχουν οι clients

Η τεχνολογία Intel SGX στους υπολογιστές από προεπιλογή είναι απενεργοποιημένη. Για να χρησιμοποιήσει κανείς το Intel SGX πρέπει να το ενεργοποιήσει αρχικά μέσω του BIOS. Αυτό απαιτεί ένα BIOS όπου από τον κατασκευαστή του υποστηρίζει ρητά το

Intel SGX. Η υποστήριξη που παρέχεται από το BIOS μπορεί να ποικίλει μεταξύ κατασκευαστών. Για να λειτουργήσει το Intel SGX, αφού ενεργοποιήσουμε στο BIOS, πρέπει να εγκατασταθεί στο σύστημα το πακέτο λογισμικού Intel SGX Platform (ή αλλιώς PSW)[11].

```
java -cp dist/LASIGE-BFT-Protocols.jar:lib/* edu.bft.client.
   ThroughputClient [client id] [request] [concurrent] [interval
   ]
```

Κεφάλαιο 4

Αξιολόγηση

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT). Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT). Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

4.1 Micro-Benchmarks

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας

υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

4.2 End-to-End Benchmarks (Latency)

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

Μετρήσεις Latency			
Req/Res	JPBFT	MinBFT-SGX Hardware Mode	MinBFT-SGX Simulation Mode
0/0	0.55 ms	1339.46 ms	1.3 ms
4K/0	0.55 ms	1432.68 ms	1.40 ms
0/4k	0.57 ms	1546.67 ms	1.45 ms

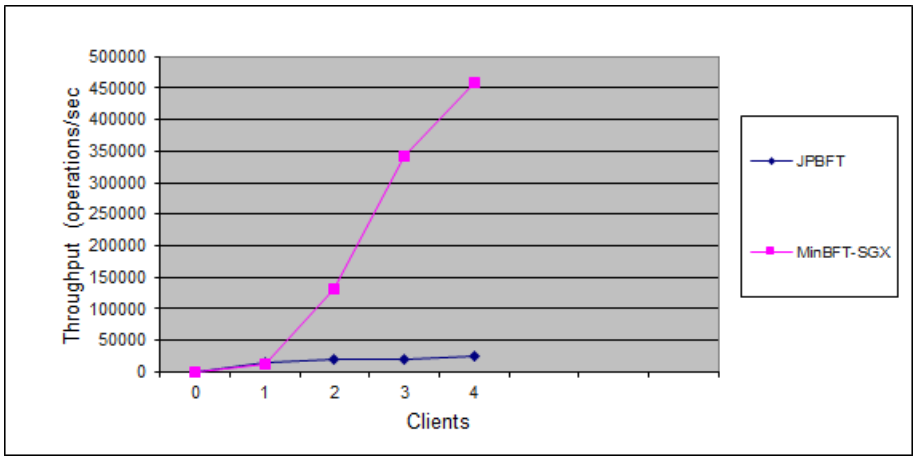
Πίνακας 4.1: Καθυστέρηση για διάφορα μεγέθη Αίτησης και Απάντησης για το JPBFT και το MinBFT-SGX

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

4.3 Αξιολόγηση του MinBFT-SGX (Throughput)

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT). Το Σχήμα 4.1 δείχνει ότι τα λιγότερα βήματα επικοινωνίας και ο αριθμός των αντιγράφων στο MinBFT-SGX αντικατοπτρίζεται σε υψηλότερη απόδοση, επιτυγχάνοντας περίπου 30.000 λειτουργίες ανά δευτερόλεπτο. Είναι ενδιαφέρον να παρατηρήσουμε ότι ο μειωμένος αριθμός σταδίων επικοινωνίας και αντιγράφων κάνει τα αντίγραφα(διακομιστές) να επεξεργάζονται λιγότερα μηνύματα (λιγότερα I/O), πράγμα που αυξάνει την απόδοση.

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των



Σχήμα 4.1: Απόδοση για 0/0 λειτουργίες για το MinBFT και το JPBFT

εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

Κεφάλαιο 5

Συμπεράσματα

5.1 Συμπεράσματα

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

- **Αύξηση Απόδοσης**
  Η Intel από το 2015 που ανακοίνωσε την τεχνολογία Software Guard Extensions, από την έκτη γενιά επεξεργαστών που βασίζεται στην μικροαρχιτεκτονική Intel Skylake και μετέπειτα την ενσωματώνει δωρεάν σε κάθε νέα γενιά επεξεργαστών. Αυτό σημαίνει πως η τεχνολογία αυτή βρίσκεται σε κάθε προσωπικό υπολογιστή και τα οφέλη της υλοποίησης μας είναι προσβάσιμα από όλους.

- **Διαθεσιμότητα Intel SGX**, Η Intel από το 2015 που ανακοίνωσε την τεχνολογία Software Guard Extensions, από την έκτη γενιά επεξεργαστών που βασίζεται στην μικροαρχιτεκτονική Intel Skylake και μετέπειτα την ενσωματώνει δωρεάν σε κάθε νέα γενιά επεξεργαστών. Αυτό σημαίνει πως η τεχνολογία αυτή βρίσκεται σε κάθε προσωπικό υπολογιστή και τα οφέλη της υλοποίησης μας είναι προσβάσιμα από όλους.

5.2 Μελλοντική Δουλειά

Αυτή η ενότητα παρουσιάζει τα αποτελέσματα απόδοσης του αλγορίθμου MinBFT-SGX με τη χρήση micro-benchmarks. Μετρήσαμε την καθυστέρηση και την απόδοση των εφαρμογών MinBFT με μηδενικές λειτουργίες. Το PBFT θεωρείται συχνά ως η γραμμή βάσης για τους αλγόριθμους BFT, επομένως μας ενδιαφέρει η σύγκριση της δικής μας υλοποίησης του αλγορίθμου με την εφαρμογή που είναι διαθέσιμη στο web. Για να συγκρίνουμε αυτή την εφαρμογή με τον αλγόριθμο MinBFT-SGX, χρησιμοποίησα την υλοποίηση της κανονικής λειτουργίας του PBFT στην Java (JPBFT).

Κεφάλαιο 6

Παράρτημα

6.1 Monotonic Counter code

```
#include <iostream>
using namespace std;

int main()
{
    int n, t1 = 0, t2 = 1, nextTerm = 0;

    cout << "Enter the number of terms: ";
    cin >> n;

    cout << "Fibonacci Series: ";

    for (int i = 1; i <= n; ++i)
    {
        // Prints the first two terms.
        if(i == 1)
        {
            cout << " " << t1;
            continue;
        }
        if(i == 2)
        {
            cout << t2 << " ";
            continue;
        }
        nextTerm = t1 + t2;
        t1 = t2;
        t2 = nextTerm;

        cout << nextTerm << " ";
    }
    return 0;
}
```

Snippet 6.1: Δημιουργία monotonic counter στο Enclave.cpp

Βιβλιογραφία

[1] *IBM blockchain*. http://www.ibm.com/blockchain/. Accessed: 2018-09-15.

[2] Miguel Castro and Barbara Liskov. *Practical Byzantine Fault Tolerance*. URL: http://pmg.csail.mit.edu/papers/osdi99.pdf.

[3] Giuliana Santos Veronese, Miguel Correia, Alysson Neves Bessani, Lau Cheuk Lung, and Paulo Verissimo. *Efficient Byzantine Fault Tolerance*. URL: https://ieeexplore.ieee.org/document/6081855/.

[4] Gopinath Nirmala and Rakesh. *Improving the Security and Efficiency of Blockchain-based Cryptocurrencies*. URL: https://aaltodoc.aalto.fi/handle/123456789/27919.

[5] João Sousa, Alysson Bessani, and Marko Vukolić. *A Byzantine Fault-Tolerant Ordering Service for the Hyperledger Fabric Blockchain Platform*. URL: https://arxiv.org/abs/1709.06921.

[6] Johannes Behl, Tobias Distler, and Rudiger Kapitza. *Hybrids on Steroids: SGX-Based High Performance BFT*. URL: https://www4.cs.fau.de/Publications/2017/behl_17_eurosys.pdf.

[7] J. Behl, T. Distler, and R. Kapitza. *Consensus-oriented parallelization: How to earn your first million.* URL: https://www4.cs.fau.de/Publications/2015/behl_15_mw.pdf.

[8] R. Kapitza, J. Behl, C. Cachin, T. Distler, S. Kuhnle, S. V. Mohammadi, W. Schroder-Preikschat, and K. Stenge. *CheapBFT: resource-efficient byzantine fault tolerance*. URL: https://dl.acm.org/citation.cfm?id=2168866.

[9] Jian Liu, Wenting Li, Ghassan O. Karame, and N. Asokan. *Scalable Byzantine Consensus via Hardware-assisted Secret Sharing*. URL: https://arxiv.org/pdf/1612.04997v4.pdf.

[10] *The source code used in Efficient Byzantine Fault Tolerance paper*. http://www.gsd.inesc-id.pt/~mpc/software/minimal/index.html. Accessed: 2018-06-21.

[11] *Intel(R) Software Guard Extensions for Linux* OS*. https://github.com/intel/linux-sgx. Accessed: 2018-06-21.

[12] *Intel(R) Software Guard Extensions Developer Guide*. https://software.intel.com/en-us/documentation/sgx-developer-guide. Accessed: 2018-06-21.