ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΟΜΕΑΣ ΜΑΘΗΜΑΤΙΚΩΝ

E-governement και πρωτόκολλο ασφαλείας SSL.

Μοντελοποίηση με χρήση της αλγεβρικής μεθόδου προδιαγραφών CafeOBJ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ειρήνη Ε. Παναγιάρη

Εξεταστική επιτροπή: Αλέξανδρος Παπαϊωάννου, Αναπληρωτής Καθηγητής ΕΜΠ

Γεώργιος Κολέτσος, Καθηγητής ΕΜΠ

Πέτρος Στεφανέας, Λέκτορας ΕΜΠ (επιβλέπων)

Αθήνα, Μάρτιος 2011

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΟΜΕΑΣ ΜΑΘΗΜΑΤΙΚΩΝ

E-governement και πρωτόκολλο ασφαλείας SSL.

Μοντελοποίηση με χρήση της αλγεβρικής μεθόδου προδιαγραφών CafeOBJ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ειρήνη Ε. Παναγιάρη

Επιβλέπων: Πέτρος Στεφανέας

Λέκτορας ΕΜΠ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την

……	……..	……..
Αλέξανδρος Παπαϊωάννου	Γεώργιος Κολέτσος	Πέτρος Στεφανέας
Αναπλ.Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ	Λέκτορας ΕΜΠ

Αθήνα, Μάρτιος 2011

……

Ειρήνη Ε. Παναγιάρη

Διπλωματούχος Σχολής Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών, Ε.Μ.Π.

ΠΕΡΙΛΗΨΗ

Το e-government αποτελεί τη νέα υπηρεσία διακυβέρνησης για όλες τις ανεπτυγμένες και αναπτυσσόμενες χώρες. Απλοποιεί τις διαδικασίες διενέργειας συναλλαγών μεταξύ των πολιτών, των επιχειρήσεων και της κυβέρνησης.

Για να προστατευθούν οι συναλλαγές και οι πληροφορίες που μεταφέρονται μέσω του διαδικτύου, αναπτύχθηκαν το πρωτόκολλα ασφαλείας. Το πιο διαδεδομένο πρωτόκολλο ασφαλείας είναι το SSL(Secure Sockets Layer), το οποίο έχει υιοθετηθεί από τους περισσότερους WWW browsers. Επομένως είναι σημαντικό το SSL να είναι πράγματι ασφαλές.

Στην παρούσα διπλωματική εργασία, χρησιμοποιείται η αλγεβρική γλώσσα προδιαγραφών CafeOBJ και η μέθοδος ΠΣΜ (Παρατηρήσιμων Συστημάτων Μετάδοσης) με σκοπό τη μοντελοποίηση κάποιων συστημάτων. Πρώτη εφαρμογή γίνεται στο πρωτόκολλο χειραψίας του SSL. Το σημαντικό αποτέλεσμα αυτής της ανάλυσης, είναι ότι τα προ-κύρια μυστικά (pre-master secrets) κατά τη διαδικασία χειραψίας του SSL, δε μπορούν να διαρρεύσουν όταν ο πελάτης και ο εξυπηρετητής έχουν διαπραγματευθεί μια ακολουθία Cipher και τις παραμέτρους ασφαλείας. Η δεύτερη εφαρμογή πραγματεύεται το πώς μπορεί να χρησιμοποιηθεί η μέθοδος CafeOBJ/ΠΣΜ κατά τον σχεδιασμό ενός γενικού μοντέλου e-government. Αναδεικνύεται η χρησιμότητά της για τη διατήρηση της λογικής άρα και της αποφυγής σφαλμάτων κατά την ανάλυση και τον σχεδιασμό σημαντικών προγραμμάτων.

Η δομή της διπλωματικής εργασίας είναι η εξής: Το πρώτο κεφάλαιο αποτελεί μια εισαγωγή στο e-government και τα πρωτόκολλα ασφαλείας που χρησιμοποιούνται σ΄αυτήν. Στο δεύτερο κεφάλαιο αναλύεται λεπτομερώς ο τρόπος λειτουργίας του SSL. Στο τρίτο κεφάλαιο περιγράφεται η αλγεβρική γλώσσα CafeOBJ και η μέθοδος ΠΣΜ. Το τέταρτο κεφάλαιο παρουσιάζει το πώς επαληθεύονται τα ΠΣΜ και πώς κατασκευάζονται τα Proof Scores. Το πέμπτο κεφάλαιο αποτελεί μια εφαρμογή της μεθόδου ΠΣΜ για το πρωτόκολλο χειραψίας του SSL. Τέλος, στο έκτο κεφάλαιο εφαρμόζεται η CafeOBJ/ΠΜΣ στον σχεδιασμό ενός απλού μοντέλου e-government όπως είναι η Δημόσια Διοίκηση και ειδικότερα μιας συναλλαγής δημόσιας υπηρεσίας, της απόδοσης πιστοποιητικού ποινικού μητρώου .

Λέξεις κλειδιά: CafeOBJ, OTS, e-government, SSL, αλγεβρικές γλώσσες προδιαγραφών, μοντελοποίηση, παρατηρήσιμα συστήματα μετάδοσης

ABSTRACT

E-government is the new service of governance in all developed and developing countries. It simplifies the procedures of many of the transitions taking place between the citizens, the enterprises and the government.

A large number of security protocols have been developed so as to protect and secure data and transactions exchanged in the internet. The most widely deployed security protocol amongst them is SSL (Secure Sockets Layer). Most of the WWW browsers adopt SSL. So, it is of high importance to prove that SSL is really secure.

In this graduation dissertation we use the algebraic specification language CafeOBJ and especially the OTS (Observation Transition System) method in order to formalize certain systems. Firstly, CafeOBJ/OTS is applied on the handshake protocol of SSL. The most important result of this analysis is that pre-master secrets cannot be leaked, when a client and the server have negotiated and agreed on a cipher suite and the security parameters. The second application is how CafeOBJ/OTS can be used in the designing of a general e-government model.It is underlined that CafeOBJ/OTS is useful in order to maintain the logic and avoid mistakes while analyzing and designing important applications.

The structure of the dissertation is as follows: The first chapter is an introduction to e-government and the security protocols which are used in it. In the second chapter, the way that SSL operates is thoroughly analyzed. In the third chapter, CafeOBJ and the OTS method are presented. The fourth chapter shows how the OTSs are verified and how the Proof Scores are built. The fifth chapter is an application of the OTS method for the handshake protocol of SSL. Finally, in the sixth chapter we apply CafeOBJ/OTS on a simple e-government model, such as Public Administration domain and especially on a public service, such as the certification of a citizen's criminal record.

Keywords: CafeOBJ, OTS, e-government, SSL, algebraic specification languages, formalization.

Ευχαριστίες

Πρωτίστως θα ήθελα να ευχαριστήσω την οικογένειά μου, τον σύζυγό μου, τους γονείς μου και τον αδελφό μου, για την αμέριστη συμπαράσταση και στήριξή τους κατά τη διάρκεια των σπουδών μου. Η οικογενειακή ηρεμία που μου εξασφάλισαν με βοήθησε να παραμένω συγκεντρωμένη στους στόχους μου.

Επίσης, θα ήθελα να αποδώσω τις θερμότερες ευχαριστίες μου στον επιβλέποντα καθηγητή μου κ.Πέτρο Στεφανέα, ο οποίος με έφερε σε επαφή με ένα τόσο ενδιαφέρον γνωστικό αντικείμενο και μου έδωσε τη δυνατότητα να ασχοληθώ μ'αυτό. Η βοήθεια και η καθοδήγησή του ήταν πάντα άμεση και αποτελεσματική.

Πίνακας περιεχομένων

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.ΤΙ ΕΙΝΑΙ ΤΟ e-government

α) Γενικά

Το **e-government (ηλεκτρονική διακυβέρνηση)** είναι ο όρος που χρησιμοποιείται για την αναφορά στην χρήση της τεχνολογίας πληροφοριών και επικοινωνιών η οποία έχει σκοπό να παρέχει και να βελτιώσει τις παροχές της κυβέρνησης και τις συναλλαγές με τους πολίτες.

Τα **πρωτογενή μοντέλα του e-government** χωρίζονται στις εξής κατηγορίες:

- Κυβέρνηση-προς-πολίτη (government-to-citizen)
- Κυβέρνηση-προς-επιχειρήσεις (government-to-enterprises)
- Κυβέρνηση-προς-κυβέρνηση (government-to-government)
- Κυβέρνηση-προς-εργαζομένους (government-to-employees)

Σε κάθε έναν από τους παραπάνω τομείς αλληλεπίδρασης, πραγματοποιούνται τέσσερα είδη δραστηριοτήτων.

- **Παροχή πληροφοριών μέσω του Διαδικτύου,** π.χ. υπηρεσίες κανονισμών, ζητήματα δημόσιας ακρόασης, κοινοποιήσεις κλπ.
- **Αμφίδρομη επικοινωνία μεταξύ των οργανισμών και του πολίτη, της επιχείρησης ή άλλου δημοσίου φορέα.** Σε αυτό το μοντέλο, οι χρήστες μπορούν να συμμετάσχουν σε διάλογο με τους οργανισμούς και να τους αποστέλλουν σχόλια ή αιτήματα.
- **Διενέργεια συναλλαγών,** π.χ. υποβολή φορολογικών δηλώσεων, πληρωμές φόρων, υποβολή αιτήσεων για επιχορηγήσεις.
- **Διακυβέρνηση,** π.χ. απευθείας διεξαγωγή ψηφοφορίας, προεκλογική εκστρατεία.

Ο όρος e-government παραπέμπει κυρίως σε ηλεκτρονική διακυβέρνηση μέσω διαδικτύου. Παρ' όλα αυτά υπάρχουν πολλές ηλεκτρονικές, μη διαδικτυακές τεχνολογίες οι οποίες χρησιμοποιούνται για να επιτευχθεί ηλεκτρονική διακυβέρνηση. Κάποιες τέτοιες φόρμες περιλαμβάνουν επικοινωνία μέσω τηλεφώνου, φαξ, μηνυμάτων μέσω κινητού τηλεφώνου, ασύρματων δικτύων και υπηρεσιών, συστήματα παρακολούθησης, ταυτότητες και «έξυπνες κάρτες», exit polls, διαχείριση της οδικής κυκλοφορίας κλπ.

Τα προσδοκώμενα οφέλη της ηλεκτρονικής διακυβέρνησης, είναι η αποδοτικότητα, η βελτίωση των υπηρεσιών, η βελτίωση της προσβασιμότητας των δημόσιων υπηρεσιών και η μεγαλύτερη διαφάνεια των διαδικασιών.

Οι πολέμιοι του e-government υποστηρίζουν ότι ένας τέτοιος τύπος διακυβέρνησης περιθωριοποιεί ένα ποσοστό του πληθυσμού το οποίο δεν δύναται να έχει πρόσβαση στο διαδίκτυο. Επίσης οι αμφισβητίες θεωρούν πως χάνεται η αίσθηση της ιδιωτικότητας και δεν διαφυλάσσονται επαρκώς τα προσωπικά δεδομένα των χρηστών. Σε όποια κατηγορία κι αν ανήκουμε, των υποστηρικτών ή των πολέμιων, οφείλουμε να λάβουμε σοβαρά υπόψιν έναν σημαντικό παράγοντα, την ανάγκη για **διασφάλιση των συναλλαγών** που πραγματοποιούνται μέσω διαδικτύου.

Επομένως, κρίνεται επιτακτικό να δημιουργηθεί ένα **υψηλό επίπεδο ασφαλείας** των συναλλαγών το οποίο θα ελαχιστοποιεί τον κίνδυνο από τις επιθέσεις στον κυβερνοχώρο.

β) Το e-government στην Ελλάδα

‘Ερευνες των μεταπτυχιακών φοιτητών του τμήματος Οικονομικών και Κοινωνικών Επιστημών του Πανεπιστημίου Μακεδονίας, οδήγησαν στα εξής στοιχεία: Η αρχική ενέργεια για εφαρμογή e-government στην Ελλάδα έγινε το 1994 με το πρόγραμμα «Κλεισθένης», το οποίο εισήγαγε νέες τεχνολογίες στο δημόσιο τομέα. Η «ΣΥΖΕΥΞΙΣ», το Εθνικό Δίκτυο Δημόσιας Διοίκησης, ξεκίνησε το 2001 και σταδιακά συνδέθηκε με το Ελληνικό Δίκτυο Έρευνας και Τεχνολογίας και το ευρωπαϊκό ασφαλές δίκτυο TESTA. Την περίοδο 2000-2009 δρομολογήθηκαν αρκετά νέα έργα. Το πρόγραμμα «ΑΡΙΑΔΝΗ» στοχεύει στην αξιολόγηση, απλούστευση και ψηφιοποίηση των διοικητικών διαδικασιών, το «ΠΟΛΙΤΕΙΑ» το οποίο εγκαθίδρυσε τις πραγματικές ανάγκες της δημόσιας διοίκησης και το «TAX-ISNET» το οποίο προσφέρει στους πολίτες μέσω διαδικτύου φορολογικές και τελωνειακές υπηρεσίες που περιλαμβάνουν τη διαχείριση του ΦΠΑ, τη δήλωση φορολογίας εισοδήματος, την καταχώρηση οχημάτων κλπ. Το 2009, η Εθνική Πύλη Δημόσιας Διοίκησης «ΕΡΜΗΣ» εξασφάλισε την ασφαλή συναλλαγή της πληροφόρησης του κοινού. Από την έναρξη της Εθνικής Ψηφιακής Στρατηγικής 2006-2013 – η οποία εισήχθει στη δεύτερη φάση της το 2009 – η Ελλάδα επέδειξε σημαντική πρόοδο στον τομέα της πληροφόρησης και των τεχνολογιών επικοινωνίας.

Η Λευκή Βίβλος που δημοσιεύθηκε το 1999 και επικαιροποιήθηκε το 2002, στόχευε να τονίσει την ανάγκη για δημόσιες υπηρεσίες ποιότητας. Για την περίοδο 2006-2013, ένα νέο στρατηγικό σχέδιο, η Ψηφιακή Στρατηγική 2006-2013, υιοθετήθηκε ώστε να χαρτογραφηθεί η Εθνική Ψηφιακή Πορεία. Το σχέδιο αυτό δεν έχει επικεντρωθεί σε συγκεκριμένα σχέδια για κάθε οργανισμό, σκοπός του ήταν η βελτίωση της παραγωγικότητας της ελληνικής οικονομίας και της ποιότητας ζωής του πολίτη. Σύμφωνα με το Εθνικό Στρατηγικό Πλαίσιο Αναφοράς 2007-2013, η οργάνωση της δημόσιας διοίκησης έχει ως στόχο να βελτιωθεί μέσω του Δημοσίου Επιχειρησιακού Προγράμματος Μεταρρύθμισης.

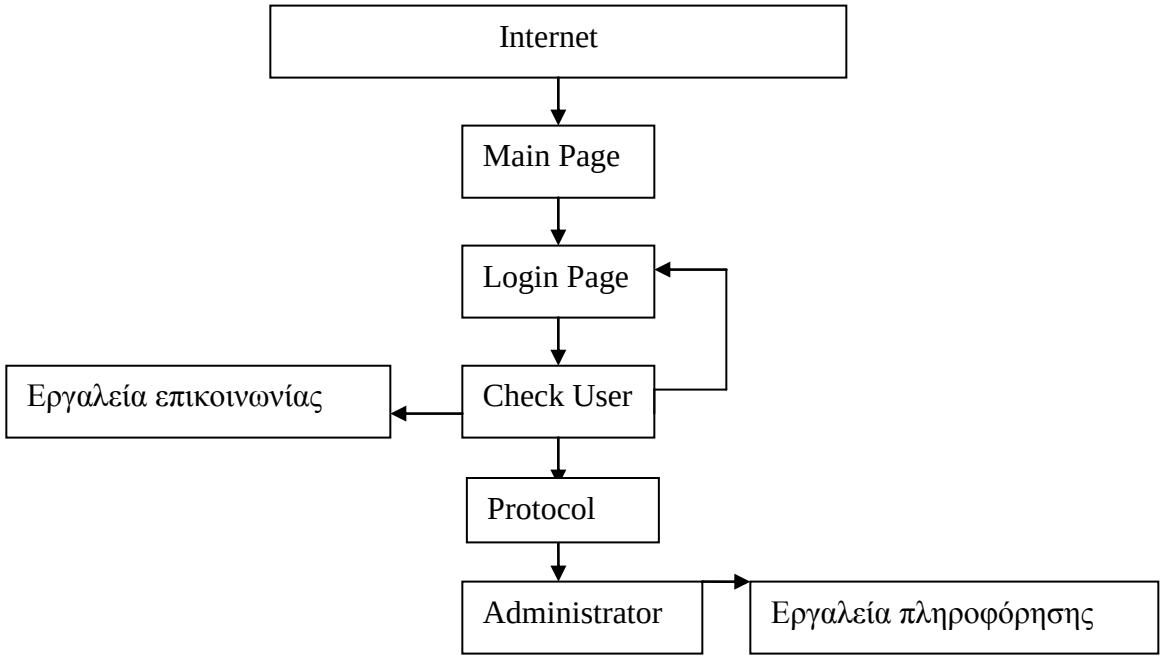
Το Ελληνικό Σύνταγμα εγγυάται τις θεμελιώδεις αρχές του δικαιώματος πρόσβασης στην πληροφόρηση, τη συμμετοχή όλων στην κοινωνία της πληροφόρησης και την υποχρέωση της πολιτείας να απαντά στα αιτήματα των πολιτών για παροχή πληροφοριών σε εύθετο χρόνο. Οι εργασίες μέλους για την ηλεκτρονική διακυβέρνηση ελέγχονται από το Ελληνικό Ελεγκτικό Συνέδριο. Περαιτέρω νομικές οντότητες που έχουν υιοθετηθεί είναι οι εξής:

- Νόμος για την προστασία των φυσικών προσώπων έναντι της επεξεργασίας δεδομένων προσωπικού χαρακτήρα. Προστατεύει το δικαίωμα των πολιτών στην ιδιωτική ζωή.
- Νόμος Τηλεπικοινωνιών. Ελέγχει τις ηλεκτρονικές επικοινωνίες.
- Προεδρικά διατάγματα. Κάνουν απλούστερες τις διαδικασίες δημοσίων συμβάσεων και καθιερώνουν μια διαδικασία ηλεκτρονικών συμβάσεων.

Για την ηλεκτρονική διακυβέρνηση στην Ελλάδα, είναι υπεύθυνο το Υπουργείο Εσωτερικών και πιο συγκεκριμένα , η Γενική Γραμματεία Δημόσιας Διοίκησης και Ηλεκτρονικής Διακυβέρνησης. Επίσης, η Ειδική Γραμματεία Ψηφιακού Σχεδιασμού, Υπουργείο Οικονομίας και Οικονομικών, έχει ως κύριο έργο της την εφαρμογή της συνολικής στρατηγικής για την Κοινωνία της Πληροφορίας.

γ)Μια απλή σχηματική απεικόνιση του σχεδιασμού και της ανάπτυξης μιας εφαρμογής e-government

Στο παρακάτω σχήμα απεικονίζεται ένα απλουστευμένο μοντέλο μιας εφαρμογής e-government. Το κάτωθι περιβάλλον αποτελείται από εργαλεία επικοινωνίας βασισμένα στο web και ένα ηλεκτρονικό πρωτόκολλο. Τα εργαλεία χωρίζονται σε «εργαλεία ενημέρωσης» και «εργαλεία επικοινωνίας» και η πρόσβαση σε κάθε ένα από αυτά εξαρτάται από τα δικαιώματα του χρήστη. Στο επίπεδο του πρωτοκόλλου πραγματοποιούνται όλες οι συναλλαγές.



Όλες οι συναλλαγές του χρήστη με το σύστημα πραγματοποιούνται στο επίπεδο του πρωτοκόλλου και στο επίπεδου που ελέγχεται η ταυτότητα του χρήστη. Οι συναλλαγές πραγματοποιούνται υπό ασφαλείς επικοινωνίες.

2.ΑΣΦΑΛΕΙΑ ΚΑΙ e-government

Η ανάπτυξη του e-government, η οποία βασίζεται στο διαδίκτυο, έρχεται αντιμέτωπη με πολύ σοβαρά προβλήματα ασφαλείας λόγω της πολυπλοκότητας και της ευστάθειας του δικτύου. Σύμφωνα με το άρθρο «Τι είναι λοιπόν αυτή η ηλεκτρονική δημοκρατία;» της εφημερίδας «ΤΟ ΒΗΜΑ», οι κίνδυνοι ασφαλείας που αντιμετωπίζει το e-government περιλαμβάνουν τις ακόλουθες πτυχές:

i) Παρακολούθηση πληροφοριών. Οι χρήστες του e-government ή κάποιοι εισβολείς, συλλαμβάνουν ή κλέβουν ηλεκτρονικές πληροφορίες της κυβέρνησης ή άλλων χρηστών.

ii) Παραποίηση πληροφοριών. Οι επιτιθέμενοι στο διαδίκτυο παραποιούν, εισάγουν ή διαγράφουν τα αρχικά δεδομένα με διάφορες τεχνικές μεθόδους και τις διαβιβάζουν εκ νέου στο δίκτυο, με σκοπό να βλάψουν την πληρότητα των δεδομένων.

iii) Άρνηση υπηρεσιών. Είναι η πλήρης ακύρωση του συστήματος δικτύου ή του εξυπηρετητή του συστήματος σε κάποια χρονική περίοδο. Προέρχεται κυρίως από τις επιθέσεις των χάκερ ή ιών καθώς και από καταστροφή των συσκευών από φυσικά πρόσωπα.

iv) Κλοπή πόρων του συστήματος. Είναι πολύ συχνή η κλοπή πόρων του συστήματος στο περιβάλλον του συστήματος του δικτύου.

v) Ψευδείς πληροφορίες. Αυτό σημαίνει ότι οι επιτιθέμενοι γνωρίζουν τους κανόνες των δεδομένων στις πληροφορίες του δικτύου ή ότι αποκωδικοποιούν τις κυβερνητικές πληροφορίες. Έτσι, προσποιούνται τους νόμιμους χρήστες ή δίνουν ψευδείς πληροφορίες για να εξαπατήσουν τους άλλους χρήστες. Η συνηθέστερη τακτική είναι ότι προσποιούμενοι τους χρήστες, παίρνουν παράνομες πιστοποιήσεις, έγγραφα, υποκλέπτουν στοιχεία λογαριασμών ηλεκτρονικού ταχυδρομείου κλπ.

Οι πιθανές πηγές απειλών σε μια εφαρμογή e-government, συνοψίζονται στον ακόλουθο πίνακα:

Απειλή	Πιθανή Πηγή
Εκ προθέσεως απειλές	Τρομοκράτες
	Δυσαρεστημένοι με τον οργανισμό ή άνθρωποι με νοητική αστάθεια
	Εγκληματίες
	Χάκερς
	Διαχειριστές του συστήματος που συνεργάζονται με εξωτερικούς εχθρούς
Ακούσιες απειλές	Λανθασμένες ενέργειες χρηστών
	Λανθασμένες ενέργειες διαχειριστών ή χειριστών
Φυσικές απειλές	Σεισμοί
	Ηφαιστειακές εκρήξεις
	Τυφώνες
	Πλημμύρες
	Κεραυνοί

Για τις «εκ προθέσεως» απειλές, τα κίνητρα μπορεί να είναι τα εξής:

- Η απόκτηση δικαιωμάτων πρόσβασης σε απόρρητα ή ευαίσθητα δεδομένα.
- Η παρακολούθηση της λειτουργίας του «υπό επίθεση» συστήματος.
- Η διαταραχή της λειτουργίας του στόχου.
- Η κλοπή χρημάτων, αντικειμένων ή υπηρεσιών.
- Η χρησιμοποίηση πηγών (υπολογιστών, ιστοσελίδων κλπ) χωρίς άδεια.
- Η τεχνολογική πρόκληση.
- Η περιέργεια.

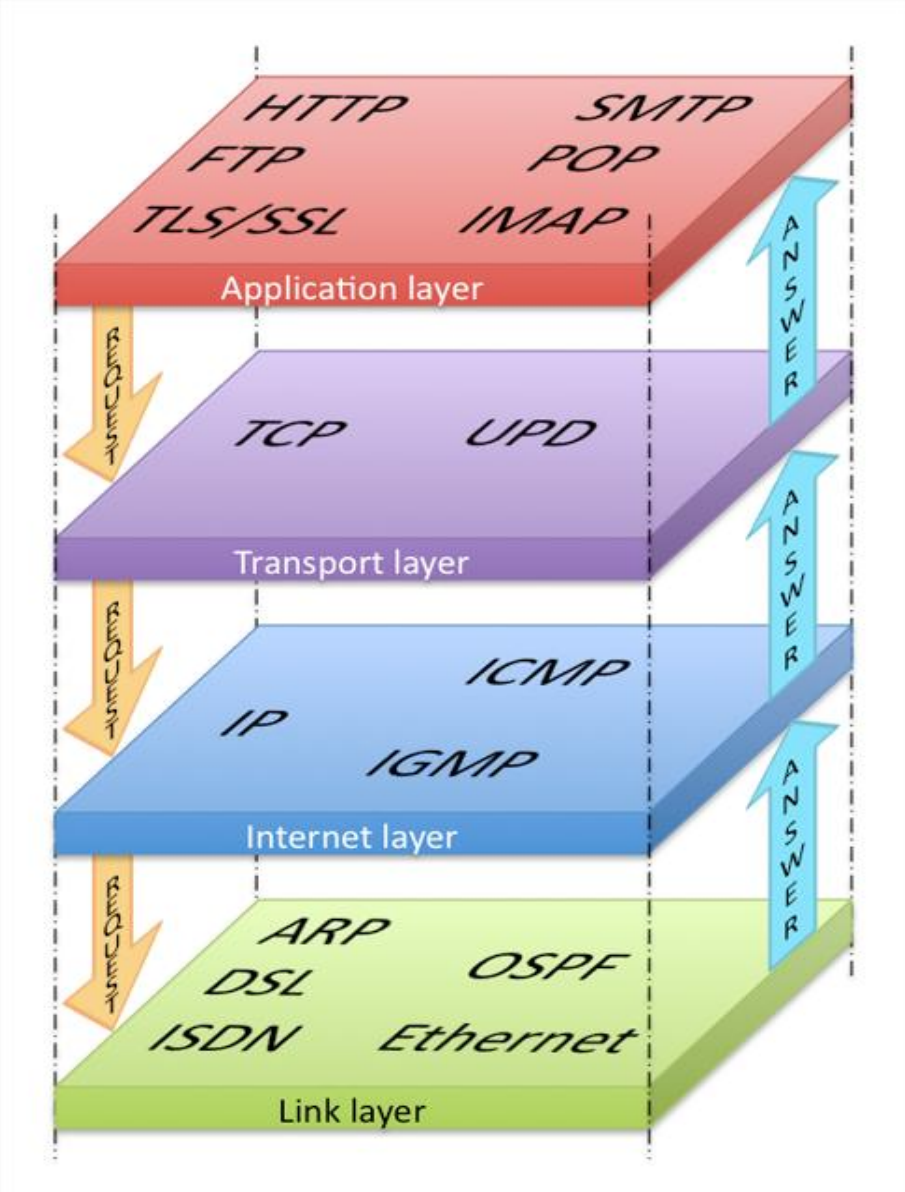
Όλες οι παραπάνω απειλές είναι πραγματικές και συχνές και επομένως είναι πρωταρχική ανάγκη να βρεθούν τρόποι οι οποίοι θα διασφαλίζουν όλες τις διαδικτυακές συναλλαγές που πραγματοποιούνται σε ένα σύστημα ηλεκτρονικής διακυβέρνησης.

Για τη διασφάλιση των ηλεκτρονικών συναλλαγών, αναπτύχθηκαν τα πρωτόκολλα ασφαλείας.

3.ΠΡΩΤΟΚΟΛΛΑ ΑΣΦΑΛΕΙΑΣ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΣΤΟ e-government

Μια εφαρμογή e-government στηρίζεται στο διαδίκτυο. Μια διαδικτυακή εφαρμογή αποτελείται από τέσσερα στρώματα (layers). Η γενική ιδέα είναι ότι πρέπει να διατηρείται ο έλεγχος των δεδομένων που ανταλλάσσονται μέχρι το τέλος μιας σύνδεσης. Κατά τη διάρκεια ανταλλαγής δεδομένων, οι εξυπηρετητές του δικτύου δεν ασχολούνται με το είδος των δεδομένων που μεταφέρουν, απλά χρησιμοποιούν την ενέργειά τους για να στείλουν και να λάβουν δεδομένα. Όλοι οι έλεγχοι για τα δεδομένα που ανταλλάσσονται, πραγματοποιούνται στο «στρώμα εφαρμογών» (application layer), στο «στρώμα μεταφοράς» (transport layer) και στο «στρώμα δικτύου» (internet layer).

Σχηματικά, μια εφαρμογή internet απεικονίζεται ως εξής:



Στο «στρώμα εφαρμογών» χρησιμοποιούνται τα πρωτόκολλα ασφαλείας, τα ονομαζόμενα κρυπτογραφικά πρωτόκολλα.

Ένα πρωτόκολλο κρυπτογράφησης μπορεί να είναι αφηρημένο ή συγκεκριμένο και εγγυάται την ασφάλεια, εφαρμόζοντας κρυπτογραφικές μεθόδους. Το πρωτόκολλο κρυπτογράφησης περιγράφει πώς θα πρέπει να χρησιμοποιούνται οι αλγόριθμοι. Ένα ικανοποιητικά λεπτομερές πρωτόκολλο περιλαμβάνει λεπτομερή στοιχεία σχετικά με τις δομές και τις παραστάσεις δεδομένων και χρησιμοποιείται για την υλοποίηση πολλών και διαλειτουργικών εκδόσεων ενός προγράμματος.

Τα πρωτόκολλα κρυπτογράφησης ενσωματώνουν τουλάχιστον ορισμένες από τις παρακάτω πτυχές:

- Συμφωνία ή εγκατάσταση κλειδιού.
- Έλεγχο γνησιότητας.
- Συμμετρική κρυπτογράφηση και μηνύματα ταυτοποίησης κατασκευής.
- Ασφαλή μεταφορά σε επίπεδο εφαρμογής των δεδομένων.
- Μεθόδους μη-αποκύρηξης.

Τα δύο πιο διαδεδομένα πρωτόκολλα κρυπτογράφησης που εγγυώνται την ασφάλεια σε εφαρμογές Web είναι το **TLS (Transport Layer Security)** και ο προκάτοχός του, **SSL (Secure Socket Layer).**

Τόσο το SSL όπως και το TLS παρέχουν ασφαλή μεταφορά και σύνδεση μεταξύ των εφαρμογών, χρησιμοποιώντας τα παρακάτω υποπρωτόκολλα:

- Πρωτόκολλο χειραψίας
    - Διαπραγμάτευση των αλγορίθμων και παραμέτρων ασφαλείας.
    - Ανταλλαγή κλειδιού.
    - Ταυτοποίηση του εξυπηρετητή και προαιρετικά, ταυτοποίηση του πελάτη.
- Πρωτόκολλο εγγραφών
    - Κατακερματισμός
    - Συμπίεση
    - Ταυτοποίηση μηνύματος και προστασία της ακεραιότητας.
    - Κρυπτογράφηση
- Πρωτόκολλο συναγερμού
    - Μηνύματα σφάλματος
- Πρωτόκολλο αλλαγής προδιαγραφών
    - Μήνυμα που υποδεικνύει το τέλος της χειραψίας

Το TLS μπορεί να θεωρηθεί ως η έκδοση 3.1 του SSL, το οποίο θα αναπτυχθεί αναλυτικά σε επόμενο κεφάλαιο.

Το πρωτόκολλο **SSH(Secure Shell)** είναι ένα πρωτόκολλο δικτύου στο στρώμα εφαρμογών, που διασφαλίζει την ανταλλαγή δεδομένων μέσω ενός ασφαλούς καναλιού και χρησιμοποιεί κρυπτογράφηση δημοσίου κλειδιού. Χρησιμοποιείται κυρίως σε λειτουργικά συστήματα LINUX και UNIX.

Ένα άλλο πρωτόκολλο ασφαλείας που χρησιμοποιείται σε μια πλατφόρμα e-government, κυρίως εκεί που υπάρχουν οικονομικές συναλλαγές μέσω διαδικτύου, είναι το **SET (Secure Electronic Transaction).**

Το SET είναι ένα πρωτόκολλο που εξασφαλίζει τις συναλλαγές μέσω πιστωτικών καρτών στο διαδίκτυο. Δεν αποτελεί από μόνο του ένα σύστημα πληρωμών αλλά ένα σύνολο πρωτοκόλλων ασφαλείας που δίνει τη δυνατότητα στους χρήστες να χρησιμοποιούν την υπάρχουσα υποδομή της πιστωτικής κάρτας σε ένα ανοιχτό δίκτυο, με ασφαλή τρόπο.

Αναπτύχθηκε από την SETco, με επικεφαλής τις VISA και MASTERCARD (και εμπλέκοντας άλλες εταιρείες όπως οι GTE, IBM, Microsoft, Netscape, RSA και VeriSign) και ξεκίνησε το 1996. Επιτρέπει στις δύο πλευρές που συναλλάσσονται, να πιστοποιήσουν κρυπτογραφικά τις ταυτότητές τους και να ανταλλάξουν ασφαλώς πληροφορίες. Το SET χρησιμοποιεί έναν αλγόριθμο, που στην ουσία, επιτρέπει να αντικατασταθεί ο αριθμός πιστωτικής κάρτας του χρήστη με ένα πιστοποιητικό. Επομένως, με τη χρήση του SET δεν αποκαλύπτεται ο αριθμός της πιστωτικής κάρτας του χρήστη και έτσι, παρέχονται εξακριβωμένα καλές πληρωμές και προστατεύονται οι χρήστες και οι εταιρείες πιστοληπτικής ικανότητας από απάτες.

Το SET αποσκοπούσε στο να αποτελέσει τη μόνη ή έστω τη συχνότερη μέθοδο συναλλαγών μέσω διαδικτύου αλλά παρ' όλη τη δημοσιότητα που έλαβε, απέτυχε να κερδίσει αυτό το σημαντικό κομμάτι της αγοράς. Οι λόγοι που οδήγησαν στην αποτυχία ήταν οι εξής:

- Το κόστος και η πολυπλοκότητα για να παρέχεται υποστήριξη, ειδικά συγκρινόμενα με το χαμηλό κόστος και την απλότητα του υπάρχοντος SSL.
- Το πιστοποιητικό διανομής απ' την πλευρά του πελάτη
- Η απαίτηση εγκατάσταση λογισμικού πελάτη.

Ένα ακόμα πρωτόκολλο του «στρώματος εφαρμογής» που χρησιμοποιείται σε εφαρμογές e-government για να διασφαλίσει τα ηλεκτρονικά ταχυδρομεία είναι το **PGP (Pretty Good Privacy).**

Το PGP αναπτύχθηκε το 1991 από τον Philip Zimmermann και χρησιμοποιείται για να κρυπτογραφεί και να αποκρυπτογραφεί μηνύματα ηλεκτρονικού ταχυδρομείου. Η PGP κρυπτογράφηση χρησιμοποιεί ένα σειριακό συνδυασμό συμπίεσης δεδομένων, συμμετρικό κλειδί κρυπτογράφησης και κρυπτογραφία δημοσίου κλειδιού. Κάθε βήμα χρησιμοποιεί έναν από τους υποστηριζόμενους αλγορίθμους. Κάθε δημόσιο κλειδί είναι συνδεδεμένο με ένα όνομα χρήστη και/ή μια διεύθυνση ηλεκτρονικού ταχυδρομείου. Νέες εκδόσεις του PGP απελευθερώνονται περιοδικά και οι προγραμματιστές συνεχώς βελτιώνουν τα τρωτά σημεία του πρωτοκόλλου.

Στο επίπεδο του «στρώματος διαδικτύου», η ασφάλεια διασφαλίζεται με χρήση του πρωτοκόλλου **IPsec (Internet Protocol Security)**. Το IPsec διασφαλίζει τις επικοινωνίες επικυρώνοντας και κρυπτογραφώντας κάθε πακέτο κατά τη διάρκεια μιας συνόδου επικοινωνίας. Επίσης, περιλαμβάνει πρωτόκολλα για την ίδρυση αμοιβαίου ελέγχου ταυτότητας κατά την έναρξη της συνόδου και για τη διαπραγμάτευση των κλειδιών κρυπτογράφησης.

Μπορεί να χρησιμοποιηθεί για την προστασία της ροής δεδομένων μεταξύ ενός ζεύγους υποδοχέων (host-to-host) , μεταξύ ενός ζεύγους πυλών ασφαλείας(network-to-network) ή μεταξύ ενός υποδοχέα και μιας πύλης ασφαλείας (host-to-network). Αυτό που, στην ουσία, κάνει το IPsec είναι να προστατεύει την κίνηση των εφαρμογών σε ένα δίκτυο IP. Οι ίδιες οι εφαρμογές δεν χρησιμοποιούν IPsec. Η ασφάλεια των εφαρμογών πρέπει να διασφαλίζεται από τα πρωτόκολλα ασφαλείας SSL ή TLS όπως αναφέρθηκε παραπάνω.

Εκτός από τα παραπάνω πρωτόκολλα ασφαλείας που χρησιμοποιούνται σε μια εφαρμογή e-government, αλλά και γενικότερα σε μια εφαρμογή Internet, πρέπει να λαμβάνουμε και συμπληρωματικά μέτρα.

Μια εφαρμογή e-government οφείλει να χρησιμοποιεί συστήματα Firewall και συστήματα ανίχνευσης εισβολέων. Επιπλέον, για τα προγράμματα που κατασκευάζονται για να εξυπηρετήσουν το e-government πρέπει να χρησιμοποιούνται υψηλού επιπέδου γλώσσες προγραμματισμού, όπως η Java και η C.

Αντικείμενο αυτής της εργασίας, αποτελεί η επαλήθευση ιδιοτήτων του πρωτοκόλλου SSL χρησιμοποιώντας την αλγεβρική γλώσσα προδιαγραφών CafeOBJ. Ας εξετάσουμε λοιπόν αναλυτικά το πρωτόκολλο ασφαλείας SSL.

ΚΕΦΑΛΑΙΟ 2:
SSL (Secure Sockets Layer)

Πριν ξεκινήσουμε την ανάλυση του πρωτοκόλλου SSL, θα ήταν χρήσιμο να αναφερθούμε στην κρυπτογράφηση, δεδομένου ότι το SSL είναι ένα κρυπτογραφικό πρωτόκολλο.

1. ΚΡΥΠΤΟΓΡΑΦΗΣΗ

Η **κρυπτογράφηση** εφαρμόζει μια μαθηματική συνάρτηση σ' ένα απλό κείμενο και το μετατρέπει σε κρυπτογραφημένο κείμενο.

Ο αλγόριθμος που εκτελεί μια διαδικασία κρυπτογράφησης ονομάζεται **cipher.**

- Μια cipher εξαρτάται συνήθως από ένα κομμάτι βοηθητικών πληροφοριών που ονομάζεται κλειδί.
- Το **κλειδί**, το οποίο μπορεί να ποικίλει, αλλάζει τη λειτουργία του αλγορίθμου.
- Το κλειδί πρέπει να επιλέγεται πριν χρησιμοποιηθεί μια cipher για την κρυπτογράφηση ενός μηνύματος.
- Χωρίς το κατάλληλο κλειδί, θα ήταν δύσκολο, αν όχι αδύνατον, να αποκρυπτογραφήσουμε το κρυπτογραφημένο κείμενο και να το μετατρέψουμε σε απλό.

Η κρυπτογράφηση είναι απαραίτητη κατά την επικοινωνία μέσω του διαδικτύου. Χρειάζεται να διασφαλίσει τα εξής:

- Πιστοποίηση
- Ιδιωτικότητα
- Ακεραιότητα
- Υπευθυνότητα

Η **πιστοποίηση** αποδεικνύει την ταυτότητα. Η ορθή εφαρμογή της κρυπτογράφησης αποδεικνύει την ταυτότητα. Το όνομα χρήστη και ο κωδικός πρόσβασης που παρέχονται για τη δημιουργία μιας σύνδεσης, αποτελούν δύο μοναδικά κλειδιά.

Λέγοντας **ιδιωτικότητα**, εννοούμε ότι η μετάδοση δεδομένων διατηρείται ιδιωτική. Η κρυπτογράφηση καθιστά σχεδόν αδύνατο για τα παράνομα μέρη να αποκρυπτογραφήσουν οποιεσδήποτε υποκλεμμένες μεταδόσεις.

Η **ακεραιότητα** ορίζεται ως μια κατάσταση ευρωστίας και ολότητας. Όταν ένα έγγραφο έχει υπογραφεί ψηφιακά με μια κρυπτογραφημένη υπογραφή, δε μπορεί να τροποποιηθεί χωρίς την αλλαγή της ψηφιακής υπογραφής. Επομένως, μια ψηφιακή υπογραφή εγγυάται ότι ένα έγγραφο ή αρχείο δεν έχει τροποποιηθεί και συνεπώς, ενεργεί ως σφραγίδα του εγγράφου.

Η **υπευθυνότητα** είναι συνώνυμη με τη μη-αποκύρηξη. Οι ψηφιακές υπογραφές παρέχουν τη μη-αποκύρηξη των εγγράφων. Εάν ο συγγραφέας υπέγραψε το έγγραφο ή το μήνυμα, δε μπορεί αργότερα να αρνηθεί τη δημιουργία αυτού και επίσης, κανείς άλλος δε μπορεί να διεκδικήσει τη συγγραφή αυτού του εγγράφου.

Υπάρχουν δύο κύρια είδη κρυπτογράφησης κλειδιού:

1. **Συμμετρική, γνωστή και ως κρυπτογράφηση ιδιωτικού κλειδιού.**

   Χρησιμοποιούνται συμμετρικά, ή πανομοιότυπα, κλειδιά για την κρυπτογράφηση και την αποκρυπτογράφηση των μηνυμάτων. Ο αποστολέας και ο παραλήπτης έχουν λάβει εκ των προτέρων ένα κοινό κλειδί που είναι κρυφό από όλα τα άλλα μέρη.

   Υπάρχουν δύο κατηγορίες κρυπτογράφησης συμμετρικού κλειδιού:

   - **Αλγόριθμοι Block**. Οι αλγόριθμοι Block έχουν τα εξής χαρακτηριστικά:

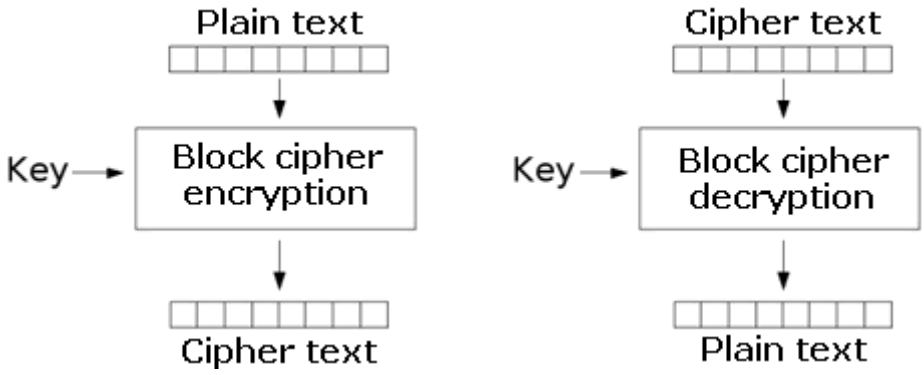
     i) Λειτουργούν με καθορισμένου μήκους ομάδες bits, που ονομάζονται blocks, με αναλλοίωτο μετασχηματισμό.

     ii) Αποτελούνται από δύο ζεύγη αλγορίθμων, έναν για την κρυπτογράφηση και έναν για την αποκρυτογράφηση.

     Οι αλγόριθμοι DES (Data Encryption Standard), AES (Advanced Encryption Standard) και IDEA είναι αλγόριθμοι Block.

   - **Αλγόριθμοι Stream**. Στους αλγορίθμους stream είναι κρυπτογραφημένος ένας χαρακτήρας, ή ένα byte, των δεδομένων κάθε στιγμή. Για το λόγο αυτό, οι αλγόριθμοι stream ονομάζονται και αλγόριθμοι κατάστασης.

2. **Ασύμμετρη, γνωστή και ως κρυπτογράφηση δημοσίου κλειδιού.**



Επιτρέπει στους χρήστες να επικοινωνούν με ασφάλεια χωρίς να προαπαιτείται πρόσβαση σε ένα διαμοιραζόμενο μυστικό κλειδί. Χρησιμοποιεί ένα ζευγάρι κρυπτογραφικών κλειδιών

που σχετίζονται μαθηματικά. Το ένα ορίζεται ως ιδιωτικό κλειδί και παραμένει μυστικό και το άλλο ορίζεται ως δημόσιο κλειδί και διανέμεται ελεύθερα.

Υπάρχουν αρκετοί τύποι κρυπτογράφησης δημοσίου κλειδιού. Κάποιοι απ' αυτούς είναι οι εξής:

- **Κρυπτογράφηση δημοσίου κλειδιού.** Διατηρεί ένα μήνυμα κρυφό από οποιονδήποτε δεν κατέχει ένα συγκεκριμένο ιδιωτικό κλειδί. Ο αλγόριθμος RSA είναι ένας ευρέως χρησιμοποιούμενος αλγόριθμος ο οποίος βασίζεται στην παραγοντοποίηση πρώτων αριθμών.
- **Δημόσιο κλειδί με ψηφιακή υπογραφή**. Επιτρέπει σε οποιονδήποτε να επαληθεύσει ότι ένα μήνυμα δημιουργήθηκε με συγκεκριμένο ιδιωτικό κλειδί.
- **Συμφωνία κλειδιού**. Επιτρέπει στα δύο μέρη που αρχικά δε μοιράζονται ένα μυστικό κλειδί, να συμφωνήσουν σε ένα. Το πρωτόκολλο συμφωνίας Diffie-Hellman χρησιμοποιείται ευρέως.

Ένα μεγάλο μειονέκτημα των αλγορίθμων συμμετρικού κλειδιού είναι η απαίτηση ενός κοινού μυστικού κλειδιού με ένα αντίγραφο. Δεδομένου ότι τα κλειδιά υπόκεινται σε πιθανή ανακάλυψη από έναν κρυπτογραφικό αντίπαλο, πρέπει να αλλάζονται συχνά και να φυλάσσονται ασφαλή κατά τη διανομή και κατά την πραγματοποίηση της υπηρεσίας. Η επιλογή, διανομή και αποθήκευση των κλειδιών χωρίς λάθη και χωρίς απώλειες, είναι δύσκολο να επιτευχθούν αξιόπιστα.

Να σημειώθει επίσης ότι οι αλγόριθμοι ασύμμετρου κλειδιού είναι πιο αργοί από τους αλγορίθμους συμμετρικού κλειδιού.

Τέλος, αναφέρουμε ότι όσο το μέγεθος του κλειδιού κρυπτογράφησης αυξάνεται, τόσο αυξάνεται και η δυσκολία αποκρυπτογράφησης.

Ο αλγόριθμος ασύμμετρου κλειδιού **RSA** εφευρέθη από τους Ronald L.Rivest. Adi Shamir και Leonard Adleman το 1997. Χρησιμοποιεί παραγοντοποίηση πρώτων αριθμών. Απαιτεί δύο μεγάλους πρώτους αριθμούς, έστω p και q, ένα ιδιωτικό κλειδί δ, ένα δημόσιο κλειδί ε και χρησιμοποιεί τη συνάρτηση του Euler.

Μια ψηφιακή υπογραφή είναι η σύνοψη ενός κρυπτογραφημένου μηνύματος. Η σύνοψη αυτή δημιουργείται ως εξής:

Λαμβάνεται ένα μήνυμα και χρησιμοποιείται ένας αλγόριθμος κατακερματισμού για να υπολογιστεί μια ακολουθία χαρακτήρων μήκους 128 bits. Αυτή η ακολουθία αποτελεί τη σύνοψη του μηνύματος και είναι κρυπτογραφημένη με το ιδιωτικό κλειδί του αποστολέα.

Το πρωτόκολλο ανταλλαγής κλειδιού Diffie-Hellman επιτρέπει στις δύο πλευρές, που δεν έχουν καμία προηγούμενη γνώση, να δημιουργήσουν από κοινού ένα μυστικό κλειδί σε ένα μη-ασφαλές κανάλι επικοινωνίας. Αυτό το κλειδί μπορεί να χρησιμοποιηθεί και σε μεταγενέστερες επικοινωνίες.

Θα πρέπει να εξασφαλίζεται ότι είναι δύσκολο για τον αποστολέα να λύσει το ιδιωτικό κλειδί του παραλήπτη και αντίστροφα. Εάν το κλειδί μπορεί να αποκρυπτογραφηθεί εύκολα, τότε ένας εισβολέας

μπορεί να το υποκαταστήσει με δικό του ζεύγος ιδιωτικού/δημοσίου κλειδιού ή να συνδέσει το δημόσιο κλειδί του παραλήπτη με το δικό του ιδιωτικό κλειδί ή να δημιουργήσει ένα ψευδές μυστικό κλειδί ή να λύσει αποκωδικοποιήσει το ιδιωτικό κλειδί του παραλήπτη.

Η απλούστερη εφαρμογή του πρωτοκόλλου λειτουργεί ως εξής:

- Ο αποστολέας και ο παραλήπτης συμφωνούν σε ένα πεπερασμένο σύνολο αριθμών G το οποίο παράγει ένα στοιχείο $g \in G$. Το g είναι γνωστό σε όλους, ακόμα και στον εισβολέα.
- Ο αποστολέας επιλέγει έναν τυχαίο αριθμό a και στέλνει το g^a στον παραλήπτη.
- Ο παραλήπτης επιλέγει έναν τυχαίο αριθμό b και στέλνει το g^b στον αποστολέα.
- Ο αποστολέας υπολογίζει το $(g^b)^a$.
- Ο παραλήπτης υπολογίζει το $(g^a)^b$.
- Τόσο ο αποστολέας, όσο και ο παραλήπτης έχουν τώρα στην κατοχή τους το στοιχείο g^{ab} που αποτελεί το κοινό μυστικό κλειδί.

2. ΑΝΑΛΥΣΗ ΤΟΥ SSL

α) Ιστορική ανασκόπηση

Το πρωτόκολλο **SSL (Secure Sockets Layer)** αναπτύχθηκε το 1994 από την Netscape για να απαντήσει στην αυξανόμενη ανησυχία για την ασφάλεια στο Διαδίκτυο.
Αρχικά αναπτύχθηκε για τη διασφάλιση των επικοινωνιών μεταξύ ενός Web server και των Web browsers, ανεξαρτήτως του λειτουργικού συστήματος.
Το SSL λειτουργεί στο στρώμα Μεταφορών και συνοδεύει το Μοντέλο Διασύνδεσης OSI (Open System Interconnection) για τη στήριξη του στρώματος εφαρμογών.
Ο σχεδιασμός των προδιαγραφών του είχε σκοπό να επιτρέψει πρωτόκολλα υψηλότερου επιπέδου, όπως τα FTP, HTTP και Telnet να χρησιμοποιούν το SSL.
Το SSL κατά το σχεδιασμό του, επωφελήθηκε από τα συστήματα κρυπτογράφησης.

Τον Νοέμβριο του 1995, η Netscape δημοσιοποίησε το SSL 3.0

Η έκδοση 3.0 είναι η έκδοση που χρησιμοποιούν οι διακομιστές Web σήμερα. Είναι καλύτερο από την έκδοση 2.0 διότι:

- Μειώνει την πιθανότητα επίθεσης από κάποιον εισβολέα κατά τη διαδικασία μιας χειραψίας SSL.
- Υποστηρίζει cipher.

Ένας διακομιστής Web που εκτελεί το πρωτόκολλο SSL παρέχει:

- Προστασία των Προσωπικών Δεδομένων και άρα, πρόληψη των υποκλοπών.
- Προστασία της ταυτότητας του χρήστη και επομένως πρόληψη πλαστοπροσωπίας.
- Ακεραιότητα, που σημαίνει την πρόληψη τροποποίησης ενός μηνύματος.

Το SSL χρησιμοποιεί διαφορετικούς αλγόριθμους, καθένας απ' τους οποίους με έχει διαφορετικό κρυπτογραφικό κλειδί για την επίτευξη και των τριών παραπάνω χαρακτηριστικών ασφαλείας. Χρησιμοποιεί δηλαδή τα εξής:

- Κρυπτογράφηση ιδιωτικού κλειδιού στις συνόδους κρυπτογράφησης.
- Κρυπτογράφηση δημοσίου κλειδιού για την εξακρίβωση της γνησιότητας των clients και servers.
- Κρυπτογράφηση ιδιωτικού κλειδιού για τα δεδομένα της εφαρμογής.

β) Μοντέλο OSI και η θέση του SSL μέσα στο OSI

Το **Μοντέλο Διασύνδεσης OSI (Open System Interconnection)** ορίζει ένα πλαίσιο δικτύωσης για την εφαρμογή πρωτοκόλλων σε επτά στρώματα. Το μοντέλο OSI βασίζεται σε εργασίες που πραγματοποιήθηκαν στην IBM το 1974.

Εφαρμογή (Στρώμα 7)	Αυτό το στρώμα υποστηρίζει τις διαδικασίες εφαρμογής και του τελικού χρήστη. Οι επικοινωνιακοί εταίροι προσδιορίζονται, η ποιότητα των υπηρεσιών εντοπίζεται, η ταυτοποίηση του χρήστη και της ιδιωτικότητας λαμβάνονται υπόψιν, και προσδιορίζονται οι ενδεχόμενοι περιορισμοί στη σύνταξη των δεδομένων. Τα πάντα σε αυτό το στρώμα είναι για εφαρμογές. Υπηρεσίες εφαρμογών όπως το FTP, το telnet, και το ηλεκτρονικό ταχυδρομείο, πραγματοποιούνται αποκλειστικά σε επίπεδο εφαρμογής.
Παρουσίαση (Στρώμα 6)	Αυτό το στρώμα παρέχει ανεξαρτησία από τις διαφορές στην αναπαράσταση δεδομένων μεταφράζοντας από μορφή εφαρμογής σε μορφή δικτύου, και αντιστρόφως. Το στρώμα παρουσίασης εργάζεται για να μετατρέψει τα δεδομένα στη μορφή που το επίπεδο εφαρμογών μπορεί να δεχθεί. Αυτή η μορφή, κρυπτογραφεί τα δεδομένα που αποστέλλονται μέσω ενός δικτύου, απελευθερώνοντας τα από προβλήματα συμβατότητας. Μερικές φορές αποκαλείται στρώμα σύνταξης.
Σύνοδος (Στρώμα 5)	Αυτό το στρώμα καθορίζει, συντονίζει, και ολοκληρώνει την επικοινωνία μεταξύ εφαρμογών σε κάθε άκρο. Ασχολείται με τον συντονισμό της συνόδου και της σύνδεσης.
Μεταφορά (Στρώμα 4)	Αυτό το στρώμα παρέχει μεταφορά δεδομένων με διαφάνεια μεταξύ τελικών συστημάτων, ή υποδοχέων, και είναι υπεύθυνο για την αναφορά σφαλμάτων και τον έλεγχο της ροής. Εξασφαλίζει τη μεταφορά δεδομένων. Το πιο γνωστό πρωτόκολλο του στρώματος 4 είναι το TCP.
Δίκτυο (Στρώμα 3)	Αυτό το στρώμα παρέχει μεταγωγή και δρομολόγηση. Δημιουργεί εικονικά κυκλώματα για τη διαβίβαση δεδομένων από τον κόμβο σε κόμβο. Η δρομολόγηση και η προώθηση αποτελούν λειτουργίες αυτού του στρώματος, καθώς και η διευθυνσιοδότηση, η διαδικτύωση, η αντιμετώπιση των λαθών, ο έλεγχος συμφόρησης και η αλληλουχία πακέτων. Το πιο γνωστό πρωτόκολλο του στρώματος 3 είναι το Internet Protocol.
Ζεύξη Δεδομένων (Στρώμα 2)	Σε αυτό το στρώμα, τα πακέτα δεδομένων κωδικοποιούνται και αποκωδικοποιούνται σε bits. Γέφυρες και διακόπτες λειτουργούν σε αυτό το στρώμα. Προσκομίζεται η μετάδοση του πρωτοκόλλου γνώσης και διαχείρισης και χειρίζονται τα σφάλματα στο φυσικό στρώμα, ελέγχεται η ροή και συγχρονίζονται τα πλαίσια. Το στρώμα ζεύξης δεδομένων χωρίζεται σε δύο υποστρώματα: Media Access Control (MAC) και Logical Link Control (LLC). Το υπόστρωμα MAC ελέγχει πώς ένας υπολογιστής δικτύου αποκτά πρόσβαση στα δεδομένα και το δικαίωμα να τα μεταδώσει. Το στρώμα LLC ελέγχει το συγχρονισμό πλαισίου, τη ροή και τα σφάλματα.
Φυσικό	Αυτό το στρώμα μεταφέρει τη ροή bit, μέσω του δικτύου σε ηλεκτρολογικό και

(Στρώμα 1)	μηχανολογικό επίπεδο. Παρέχει το υλικό μέσω της αποστολής και λήψης δεδομένων σε έναν φορέα. Το Ethernet και το ATM είναι πρωτόκολλα με συστατικά στο φυσικό στρώμα.

Το SSL τοποθετείται στα ανώτερα στρώματα του μοντέλου OSI.

Στρώμα Εφαρμογής		
Change Cipher Specification	**SSL Alert Protocol**	**SSL Handshake Protocol**
SSL Record Protocol		
TCP Protocol		
IP Protocol		

Οι παραπάνω πίνακες προέκυψαν κατά τη μελέτη Σχεδίασης Εικονικών Δικτύων από το Τμήμα Πληροφορικής και Τεχνολογίας Υπολογιστών του ΤΕΙ Λαμίας.

γ) Βασικές αρχές λειτουργίας του SSL και υπο-πρωτόκολλα

Το πρωτόκολλο SSL έχει δύο στρώματα.

1. **Πρωτόκολλο Εγγραφής (Record Protocol)**

   Το SSL Record Protocol επιτρέπει την ενθυλάκωση των πρωτοκόλλων υψηλότερου επιπέδου, όπως:

   - Του πρωτοκόλλου Χειραψίας
   - Του SSL Alert πρωτοκόλλου
   - Του HTTP

   Το SSL Record Protocol αποτελεί το θεμέλιο για ολόκληρη τη μεταφορά των δεδομένων. Χτίζει τη διαδρομή δεδομένων μεταξύ αποστολέα και παραλήπτη. Επίσης, πριν σταλούν τα δεδομένα, κρυπτογραφεί τη διαδρομή τους, η οποία ξεκινά χωρίς κρυπτογράφηση.

2. **Πρωτόκολλο Χειραψίας (Handshake Protocol)**

   Το πρωτόκολλο χειραψίας χρησιμοποιεί το Record πρωτόκολλο για την ανταλλαγή μιας σειράς μηνυμάτων μεταξύ μιας ενός server και ενός client- στους οποίους είναι ενεργοποιημένο το

SSL- και συνδέονται για πρώτη φορά μέσω μιας SSL σύνδεσης.
Αυτή η ανταλλαγή μηνυμάτων είναι σχεδιασμένη έτσι ώστε να ενεργοποιήσει τις ακόλουθες δράσεις:

- Να επικυρώσει το διακομιστή στον πελάτη.
- Να επιτρέψει στον client και στον server για να επιλέξουν κρυπτογραφικούς αλγορίθμους, ή ciphers, που και οι δύο υποστηρίζουν.
- Προαιρετικά να ταυτοποιήσει τον πελάτη με τον server.
- Να δημιουργήσει κοινά μυστικά κλειδιά με χρήση κρυπτογράφησης δημόσιου κλειδιού.
- Να δημιουργήσει μια κρυπτογραφημένη σύνδεση SSL.

Συνοπτικά, τα βήματα του πρωτοκόλλου SSL είναι τα εξής:

1. Μια σύνοδος SSL ξεκινά όταν ένας πελάτης συνδέεται με ένα διακομιστή.

- Μια χειραψία SSL ξεκινά κάθε φορά που μια σύνοδος SSL ξεκινά.
- Τα πρωτόκολλα που χρησιμοποιούνται για την επικοινωνία εγκαθίστανται.
- Επιλέγεται ο κρυπτογραφικός αλγόριθμος.
- Επικυρώνονται ο πελάτης και ο διακομιστής.
- Ένα κύριο μυστικό δημιουργείται με κρυπτογράφηση δημοσίου κλειδιού.

2. Ένα κύριο μυστικό δημιουργείται από ένα προ-κύριο μυστικό το οποίο αποστέλλεται από τον πελάτη. Παράγονται τέσσερα κλειδιά συνόδου.

- Ένα κλειδί κρυπτογράφησης για τα δεδομένα που αποστέλλονται από τον client στον server.
- Ένα κλειδί κρυπτογράφησης για τα δεδομένα που αποστέλλονται από τον server στον client.
- Ένα κλειδί ταυτοποίησης για τα δεδομένα που αποστέλλονται από τον client στον server.
- Ένα κλειδί ταυτοποίησης για τα δεδομένα που αποστέλλονται από τον server στον client.

Ας εξετάσουμε τώρα λεπτομερώς τα βήματα του πρωτοκόλλου χειραψίας του SSL.

1. Ο πελάτης στέλνει στο διακομιστή τον αριθμό έκδοσης του SSL που χρησιμοποιεί, τις ρυθμίσεις κρυπτογράφησης, τα τυχαία παραγόμενα δεδομένα, καθώς και άλλες πληροφορίες που ο διακομιστής χρειάζεται για να επικοινωνήσει με τον πελάτη με χρήση SSL.

2. Ο διακομιστής στέλνει στον πελάτη τον αριθμό έκδοσης του SSL που χρησιμοποιεί, τις ρυθμίσεις κρυπτογράφησης, τα τυχαία παραγόμενα δεδομένα, καθώς και άλλες πληροφορίες που χρειάζεται ο πελάτης για να επικοινωνήσει με τον διακομιστή μέσω SSL. Ο διακομιστής στέλνει επίσης το δικό του ψηφιακό πιστοποιητικό και, αν απαιτείται έλεγχος ταυτότητας του πελάτη, ζητά το ψηφιακό πιστοποιητικό του πελάτη.

3. Ο πελάτης χρησιμοποιεί τις πληροφορίες που αποστέλλονται από το διακομιστή για να τον ταυτοποιήσει. Εάν ο διακομιστής δεν μπορεί να πιστοποιηθεί, ο χρήστης θα ειδοποιηθεί για το πρόβλημα ότι δεν μπορεί να καθοριστεί μια κρυπτογραφημένη και πιστοποιημένη σύνδεση. Εάν ο διακομιστής ταυτοποιηθεί με επιτυχία, τότε συνεχίζει.

4. Χρησιμοποιώντας όλα τα δεδομένα που παράγονται κατά τη χειραψία μέχρι στιγμής, ο πελάτης δημιουργεί το προ-κύριο μυστικό για τη σύνοδο, το κρυπτογραφεί με το δημόσιο κλειδί του server (που λαμβάνεται από το ψηφιακό πιστοποιητικό του server), και στέλνει το κρυπτογραφημένο προ-κύριο μυστικό στο διακομιστή.

5. Εάν ο διακομιστής έχει ζητήσει ταυτότητας πελάτη (προαιρετικό βήμα κατά τη χειραψία), ο πελάτης υπογράφει ένα ακόμα κομμάτι των δεδομένων που είναι μοναδικό σε αυτή το χειραψία και είναι γνωστό τόσο στον πελάτη, όσο και στον διακομιστή. Στην περίπτωση αυτή, ο πελάτης στέλνει το υπογεγραμμένο κομμάτι δεδομένων και την ψηφιακό πιστοποιητικό του στον server, μαζί με το κρυπτογραφημένο προ-κύριο μυστικό.

6. Εάν ο διακομιστής έχει ζητήσει ταυτότητας πελάτη, ο διακομιστής προσπαθεί να ταυτοποιήσει τον πελάτη. Αν ο πελάτης δεν μπορεί να πιστοποιηθεί, η σύνοδος τερματίζεται. Αν ο ταυτοποιηθεί επιτυχώς, ο διακομιστής χρησιμοποιεί το ιδιωτικό κλειδί του για να αποκρυπτογραφήσει το προ-κύριο μυστικό, στη συνέχεια, εκτελεί μια σειρά βημάτων που ο πελάτης εκτελεί επίσης, ξεκινώντας από το ίδιο προ-κύριο μυστικό για να δημιουργήσουν το κύριο μυστικό.

7. Τόσο ο πελάτης όσο και ο διακομιστής χρησιμοποιούν το κύριο μυστικό για να παράγουν τα κλειδιά συνόδου που είναι συμμετρικά κλειδιά που χρησιμοποιούνται για την κρυπτογράφηση και την αποκρυπτογράφηση των πληροφοριών που ανταλλάσσονται κατά τη διάρκεια της συνόδου SSL και να επιβεβαιώσει την ακεραιότητά της.

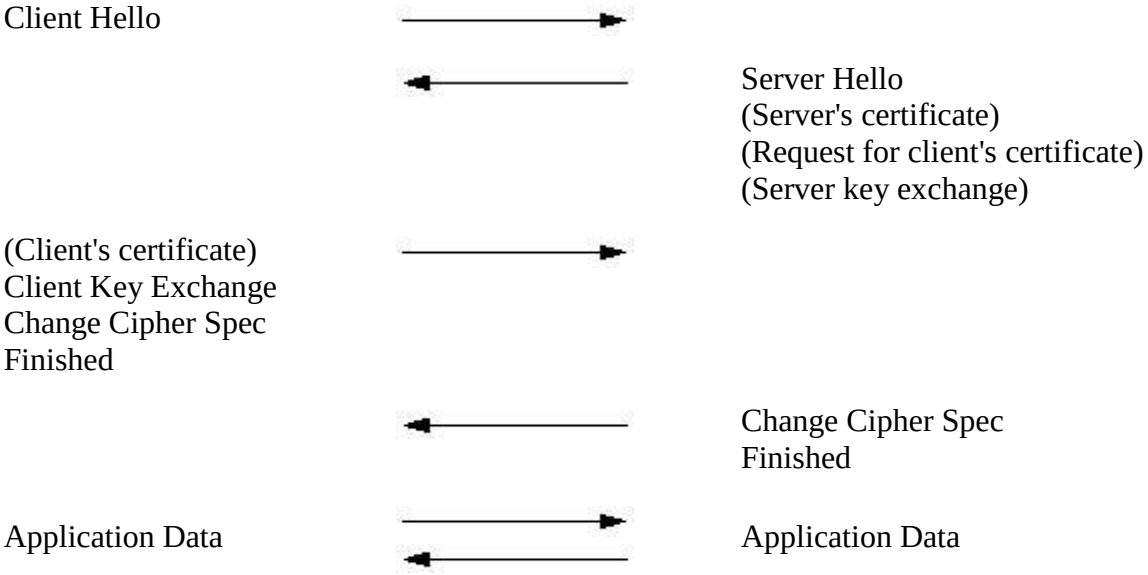
8. Ο πελάτης ενημερώνει τον server ότι μελλοντικά του μηνύματα θα είναι κρυπτογραφημένα με το κλειδί συνόδου. Στη συνέχεια, στέλνει ένα ξεχωριστό κρυπτογραφημένο μήνυμα που δηλώνει ότι το τμήμα της χειραψίας έχει τελειώσει από την πλευρά του.

9. Ο server ενημερώνει τον πελάτη ότι μελλοντικά μηνύματα του θα είναι κρυπτογραφημένα με το κλειδί συνόδου. Στη συνέχεια, στέλνει ένα ξεχωριστό κρυπτογραφημένο μήνυμα που δηλώνει ότι το τμήμα της χειραψίας έχει τελειώσει από την πλευρά του.

10. Η χειραψία SSL έχει πλέον ολοκληρωθεί, και η σύνοδος SSL έχει αρχίσει. Ο πελάτης και ο διακομιστής χρησιμοποιούν τα κλειδιά συνόδου για την κρυπτογράφηση και

αποκρυπτογράφηση των δεδομένων που στέλνουν ο ένας στον άλλο ώστε να επικυρώσουν την ακεραιότητά τους.

Σχηματικά, η διαδικασία χειραψίας είναι ως εξής:



Τα μηνύματα στις παρενθέσεις είναι προαιρετικά και απαιτούνται μόνο εάν ο server δεν έχει πιστοποιητικό ή το πιστοποιητικό του δεν υποστηρίζει τον αλγόριθμο Diffie-Hellman.

Το **SSL Alert πρωτόκολλο** ενημερώνει για τα προβλήματα που προκύπτουν σε μια σύνοδο SSL. Τα μηνύματα προειδοποίησης αναφέρουν τη σοβαρότητα του σφάλματος του μηνύματος και το περιγράφουν.

Κατά τη διαβίβαση ή την παραλαβή ενός μήνυματος σφάλματος, τα δύο μέρη που επικοινωνούν κλείνουν αμέσως τη σύνδεση.

Επομένως, ο πελάτης και ο διακομιστής πρέπει να ανακοινώνουν ότι η σύνδεση τελειώνει ώστε να αποφευχθεί μια απότομη αποκοπή. Οποιοδήποτε μέρος μπορεί να ξεκινήσει το κλείσιμο της ανταλλαγής μηνυμάτων. Ένας ομαλός τερματισμός επιτυγχάνεται όταν αποστέλλεται το μήνυμα close_notify. Αυτό το μήνυμα ενημερώνει τον παραλήπτη ότι ο αποστολέας δεν θα στείλει άλλα μηνύματα σε αυτό το πλαίσιο. Η σύνοδος δεν συνοψίζεται εάν η σύνδεση τερματιστεί χωρίς ένα σωστό close_notify μήνυμα.

Κάποια σημαντικά μηνύματα προειδοποίησης είναι τα παρακάτω:

- `unexpected_message`
  Ένα ακατάλληλο μήνυμα ελήφθη. Η ειδοποίηση αυτή είναι πάντοτε μοιραία και δεν πρέπει ποτέ να λαμβάνεται σε μια καλή επικοινωνία μεταξύ των εφαρμογών.

- `bad_record_mac`
  Η ειδοποίηση αυτή επιστρέφεται αν μια εγγραφή έχει ληφθεί με λάθος κωδικό ταυτότητας μηνυμάτων. Αυτό το μήνυμα είναι πάντοτε μοιραίο.
- `decompression_failure`
  Η λειτουργία αποσυμπίεσης έλαβε μη αποδεκτά δεδομένα (π.χ. δεδομένα που θα επεκταθούν σε υπερβολικό μήκος). Αυτό το μήνυμα είναι πάντοτε μοιραίο.
- `handshake_failure`
  Δηλώνει ότι ο αποστολέας δεν ήταν σε θέση να διαπραγματευτεί μια αποδεκτή σειρά των παραμέτρων ασφαλείας από τις διαθέσιμες επιλογές. Αυτό είναι ένα μοιραίο σφάλμα.
- `no_certificate`
  Μπορούν να αποσταλεί σε απάντηση αίτησης πιστοποίησης, εφόσον δεν υπάρχει διαθέσιμο το κατάλληλο πιστοποιητικό.
- `bad_certificate`
  Το πιστοποιητικό ήταν διεφθαρμένο, πιθανότατα περιείχε μια ψηφιακή υπογραφή που δεν ταυτοποιήθηκε σωστά σωστά.
- `unsupported_certificate`
  Το πιστοποιητικό ήταν τύπου που δεν υποστηρίζεται.
- `certificate_revoked`
  Το πιστοποιητικό ανακλήθηκε από τον υπογράφοντά του.
- `certificate_expired`
  Το πιστοποιητικό έχει λήξει ή δεν είναι επί του παρόντος ισχύον.
- `certificate_unknown`
  Κάποιο απροσδιόριστο ζήτημα ανέκυψε κατά την επεξεργασία του πιστοποιητικού, το οποίο το καθιστά μη αποδεκτό.
- `illegal_parameter`
  Ένα πεδίο στη χειραψία ήταν εκτός της εμβέλειας ή ασυμβίβαστο με άλλους τομείς. Αυτό είναι πάντοτε μοιραίο.

Το μήνυμα **Change Cipher Specification** αποστέλλεται και από τον πελάτη και από τον server ώστε και τα δύο συμβαλλόμενα μέρη να ειδοποιηθούν ότι οι επόμενες εγγραφές θα προστατεύονται βάσει της παρελθούσας διαπραγμάτευσης CipherSpec και των παρελθόντων κλειδιών.
Υπάρχει για να ενημερώνει την cipher που χρησιμοποιείται στη σύνδεση. Επιτρέπει αλλαγές κατά τη διάρκεια μιας συνόδου SSL χωρίς να χρειάζεται να επαναδιαπραγματευθεί η σύνδεση. Το μήνυμα που αποστέλλεται, αποτελείται από ένα μόνο byte ίσο με 1.

Υπάρχουν δύο καταστάσεις για τον αλγόριθμο αλλαγής προδιαγραφών.

- ο Διάβασε τις τρέχουσες
- ο Διάβασε την αναμονή

Το μήνυμα Change Cipher Specification συνήθως αποστέλλεται στο τέλος της χειραψίας SSL.

δ) Ακολουθία cipher

Μια ακολουθία cipher είναι, στην ουσία, ένα πρόγραμμα κρυπτογράφησης. Δηλαδή, ένα σύνολο κρυπτογραφικών αλγορίθμων.

Το πρωτόκολλο χειραψίας SSL καθορίζει το πώς ο πελάτης και ο διακομιστής διαπραγματεύονται τις ακολουθίες cipher που θα ακολουθήσουν.

Οι πιο συχνά χρησιμοποιούμενες ακολουθίες κρυπτογράφησης είναι οι εξής:

- DES (Data Encryption Standard)
- DSA (Digital Signature Algorithm)
- KEA (αλγόριθμος Exchange)
- MD5
- RC2
- RC4
- RSA
- SHA-1 (Secure Hash Algorithm)
- SKIP JACK
- Triple DES

ΚΕΦΑΛΑΙΟ 3
CafeOBJ/OTS METHOD

1.ΓΕΝΙΚΑ

Οι υψηλού επιπέδου γλώσσες προγραμματισμού δεν είναι αποτελεσματικές στην κατανόηση ενός τομέα. Χρησιμοποιώντας μια υψηλού επιπέδου γλώσσα προγραμματισμού είναι δύσκολο να αναγνωρίσουμε τα λάθη στον σχεδιασμό και οι παραδοχές δεν είναι ρητές, αλλά είναι εξαρτώμενες απ' την εφαρμογή. Εστιάζουν στο **πώς** εκτελούνται οι διαδικασίες στο σύστημα. Δε μας επιτρέπουν να περιγράψουμε **τι** κάνει το σύστημα. Στην έρευνα και τη βιομηχανία χρησιμοποιούμε τις αλγεβρικές γλώσσες προδιαγραφών σαν μια απάντηση στο παραπάνω ερώτημα.

Η CafeOBJ είναι μια αλγεβρική γλώσσα προδιαγραφών νέας γενιάς, διάδοχος της OBJ, που αναπτύχθηκε στην Ιαπωνία υπό την καθοδήγηση του καθηγητή Kokichi Futatsugi. Έχει κληρονομήσει τα βασικά χαρακτηριστικά της OBJ3 και έχει εμπλουτιστεί με νέα με σκοπό να χρησιμοποιηθεί για την τυπική προδιαγραφή και επαλήθευση συστημάτων, τη γρήγορη δημιουργία προτοτύπου, τον προγραμματισμό κ.α. Ο όρος OBJ αναφέρεται σε μια οικογένεια γλωσσών και οι OBJ2, OBJ3, CafeOBJ, BOBJ αποτελούν μέλη της οικογένειας OBJ. Όλες οι παραπάνω γλώσσες είναι αλγεβρικές γλώσσες προγραμματισμού και προδιαγραφών, οι οποίες βασίζονται σε μια ταξινομημένη εξισωτική λογική και αποδεικνύουν την ισχύ του παραμετροποιημένου προγραμματισμού. Όλες οι OBJ γλώσσες βασίζονται σε ένα λογικό σύστημα, είναι δηλαδή γλώσσες λογικής με την έννοια ότι τα προγράμματα που είναι γραμμένα σ'αυτές αποτελούν σύνολα προτάσεων λογικών συστημάτων. Όλες οι πρόσφατες OBJ γλώσσες χρησιμοποιούν ταξινομημένη άλγεβρα, η οποία παρέχει μια ισχυρή βάση για τη δημιουργία υπο-τύπων (sub-types) που ορίζονται από τους χρήστες, για τη διαχείρηση των εξαιρέσεων(exception handling), για την διατήρηση της κληρονομικότητας(inheritance), για πολλαπλές αναπαραστάσεις και για διορθώσεις.

Ειδικά η OBJ3 χρησιμοποιήθηκε επιτυχώς στην έρευνα και στην διδασκαλία σχεδιασμού και προδιαγραφών λογισμικού (software design and specification), στην αποδεικτική διαδικασία θεωρημάτων και στην επαλήθευση του hardware (hardware verification). Ήταν η πρώτη γλώσσα που εφάρμοσε πλήρως τον παραμετροποιημένο προγραμματισμό (parameterized programming). Στη συνέχεια παρουσιάζονται σύντομα το λογικό υπόβαθρο, τα κύρια χαρακτηριστικά και το βασικό συντακτικό της CafeOBJ.

2.ΕΙΣΑΓΩΓΗ ΣΤΗΝ CafeOBJ

α) Λογικό υπόβαθρο

Η CafeOBJ είναι εκτελέσιμη γλώσσα. Η εκτελεσιμότητα αυτή καθιστά δυνατό το να επαληθευτεί ότι κάποιες ιδιότητες ισχύουν και για τις προδιαγραφές ή ότι οι προδιαγραφές «απολαμβάνουν» κάποιες ιδιότητες.

Η λειτουργική σημασιολογία της CafeOBJ δίνεται από ένα σύστημα αναγραφής όρων με διατεταγμένους τύπους. Μια αλγεβρική προδιαγραφή είναι ένα κείμενο, γραμμένο σ' ένα τυπικό συντακτικό που υποδηλώνει ένα αλγεβρικό σύστημα αποτελούμενο από τύπους (sorts) και τελεστές (operators) πάνω σ' αυτούς τους τύπους.

Σύμφωνα με τη διδασκαλία της CafeOBJ στο πανεπιστήμιο JAIST, από τον καθηγητή Kazuhiro Ogata, παραθέτουμε τα σημαντικά χαρακτηριστικά και το βασικό συντακτικό της CafeOBJ.

β) Σημαντικά χαρακτηριστικά

Τα σημαντικά χαρακτηριστικά της CafeOBJ είναι:

- Ο αλγεβρικός προγραμματισμός. Επιτρέπεται η δήλωση ιδιοτήτων στις εξισώσεις (μεταθετικότητα, προσεταιριστικότητα, ουδέτερα στοιχεία).
- Η συμπεριφοριακή προδιαγραφή. Περιγράφονται οι συμπεριφορές συστημάτων και αντικειμένων και όχι ο τρόπος υλοποίησης.
- Η προδιαγραφή με τη λογική της αναγραφής. Η χρήση της επιτρέπει την προδιαγραφή και επαλήθευση συστημάτων μετάδοσης.
- Σύστημα ενοτήτων προδιαγραφών (System modules specification). Επιτρέπει τον παραμετροποιημένο προγραμματισμό και ενοποιεί τις προδιαγραφές σε CafeOBJ με εκτελέσιμο κώδικασε μια χαμηλότερου επιπέδου γλωσσα.
- Ισχυρό σύστημα τύπων. Επιτρέπει τη χρήση υποτύπων με βάση την άλγεβρα με διατεταγμένους τύπους.

γ) Βασικό συντακτικό

Οι βασικές έννοιες στην CafeOBJ είναι οι εξής:

- Άλγεβρες, ενότητες (modules) , τύποι (sorts), τελεστές (operators), μεταβλητές (variables).
- Όροι (terms), τελεστές mixfix (mixfix operators), εξισώσεις (equations), αναγωγή (reduction).
- Χαρακτηριστικά τελεστών, «υπό όρους» εξισώσεις.
- Εισαγωγή ενοτήτων (module importation).

Οι **άλγεβρες** θέτουν συναρτήσεις ή τελεστές στα σύνολα. Π.χ. το σύνολο των φυσικών αριθμών {0,1,….,} και οι πράξεις «πρόσθεση (+)» και «πολλαπλασιασμός (*)» αποτελούν μια άλγεβρα.

Το σύστημα ενός υπολογιστή μπορεί φυσικά να θεωρηθεί ως μια άλγεβρα, αφού αποτελείται από δεδομένα και όλα όσα «συναλλάσσονται» μ' αυτά τα δεδομένα.

Οι **ενότητες(modules)** είναι η βασική μονάδα στις προδιαγραφές CafeOBJ. Μια ενότητα δηλώνεται ως εξής:

mod! ModuleName{….}

και σ' αυτήν ορίζονται οι τύποι, οι τελεστές, οι μεταβλητές και οι εξισώσεις. Υπάρχει η δυνατότητα να εισάγονται ενότητες σε μια ενότητα. Η εισαγωγή γίνεται ως εξής:

```
mod! M {
  pr (N)
}
```

Τύποι (sorts) ονομάζονται τα ονόματα που δίνονται στα σύνολα. Π.χ. Nat είναι ο τύπος για το σύνολο των φυσικών αριθμών {0,1,….}. Οι τύποι δηλώνονται ως εξής στο εσωτερικό μιας ενότητας.

[SortName]

Οι τύποι μπορεί να είναι ορατοί (visible) ή κρυφοί (hidden). Οι ορατοί τύποι αντιστοιχούν στις τιμές των δεδομένων και οι κρυφοί τύποι αντιστοιχούν στις καταστάσεις ή στους χώρους καταστάσεων σε μια μηχανή κατάστασης.

Οι **τελεστές (operators)** έχουν το ρόλο των συναρτήσεων και δηλώνονται σε μια ενότητα ως εξής:

$$\mathbf{op\ f : S_1 \ldots S_n \rightarrow S}$$

Η ακολουθία τύπων $\mathbf{S_1 \ldots S_n}$ ονομάζεται **arity της f** και ο τύπος S ονομάζεται **coarity της f.** Το ζεύγος arity και coarity αποτελεί το **βαθμό της f.** Όταν το arity είναι κενό (n=0) , ο τελεστής ονομάζεται σταθερά. Οι συναρτήσεις ίδιου βαθμού μπορούν να δηλωθούν ταυτόχρονα:

$$\mathbf{ops\ f_1 \ldots f_m : S_1 \ldots S_n \rightarrow S}$$

Στους τελεστές μπορούν να αποδοθούν χαρακτηριστικά, **operator attributes**. Δύο απ' αυτά είναι τα prec n και comm. Το prec n είναι η προτεραιότητα του τελεστή (όσο μικρότερο το n, τόσο μεγαλύτερη η προτεραιότητα) και δηλώνεται ως:

op s : Nat Nat→Nat {prec : 20}

και το comm υποδεικνύει πως ο τελεστής είναι αντιμεταθετικός και δηλώνεται ως εξής:

op = : Nat Nat→Bool {comm}

Στην περίπτωση που το arity του τελεστή περιλαμβάνει κρυφούς τύπους (hidden sorts), ο τελεστής ονομάζεται **τελεστής συμπεριφοράς (behavioral operator)**.

Αν το arity του τελεστή συμπεριφοράς περιέχει ακριβώς ένα κρυφό τύπο και το coarity του είναι ορατός τύπος, τότε ονομάζεται **συνάρτηση παρατήρησης (observation function)**.

Αν το arity του τελεστή συμπεριφοράς περιέχει ακριβώς έναν κρυφό τύπο και το coarity του είναι ο ίδιος κρυφός τύπος, τότε ονομάζεται **συνάρτηση μετάβασης (transition function).**

Οι τελεστές συμπεριφοράς δηλώνονται ως : **bop f : S_1 …..S_n → S**

Οι **μεταβλητές (variables)** ορίζονται ως εξής : **var X : S**

Η δήλωση αυτή υποδεικνύει ότι η X είναι μεταβλητή τύπου S. Μεταβλητές ίδιου τύπου μπορούν να δηλωθούν ταυτόχρονα **vars X_1 …X_m : S**

Οι **όροι (terms)** ορίζονται επαγωγικά ως εξής:

(i) μια μεταβλητή X τύπου S είναι όρος τύπου S.

(ii) Μια συνάρτηση $f(t_1,….,t_n)$: S_1 …..S_n → S όπου $t_1,….,t_n$ είναι όροι τύπου S_1 …..S_n , αποτελεί όρο τύπου S.

Οι **mixfix τελεστές (mixfix operators)** είναι συναρτήσεις οι οποίες ορίζονται ελεύθερα αφού προσδιορίσουμε τους κανόνες που αυτές ακολουθούν. Δηλώνονται ως εξής:

op + : Nat Nat → Nat (infix)

op ~_ : Bool→Bool (prefix)

op _! : Nat→Nat (postfix)

op if_then_else_fi : Bool Nat Nat→ Nat

if ~B then x+y else Z! fi (mixfix)

Η _ υποδηλώνει πού τοποθετείται το αντικείμενο και επομένως ο αριθμός των _ πρέπει να συμπίπτει με τον αριθμό των αντικειμένων.

Οι **εξισώσεις (equations)** χρησιμοποιούνται για να ορίσουμε συναρτήσεις και ιδιότητες των συναρτήσεων. Δηλώνονται ως εξής:

eq term$_1$ = term$_2$

Οι **term$_1$, term$_2$** πρέπει να είναι όροι του ίδιου τύπου, ο **term$_1$** να μην είναι μόνο μεταβλητή και όλες οι μεταβλητές του **term$_2$** πρέπει να εμφανίζονται στον **term$_1$.**

Οι εξισώσεις θεωρούνται από «αριστερά προς τα δεξιά» γραμμένοι κανόνες για να μειώσουμε, να υπολογίσουμε ή να απλουστεύσουμε έναν δοσμένο όρο.

Επομένως χρησιμοποιούμε τη **αναγωγή (reduction)** για να πάρουμε έναν δοσμένο όρο στην κανονική του μορφή. Δηλώνεται ως εξής:

red in ModuleName : term

Χρησιμοποιώντας την εντολή "set trace on" στην αναγωγή, εμφανίζονται όλα τα βήματα που πραγματοποιήθηκαν μέχρι το τέλος της διαδικασίας. Με την "set trace off" εμφανίζεται μόνο το τελικό αποτέλεσμα.

Οι εξισώσεις μπορεί να είναι «υπό όρους». Να υπόκεινται δηλαδή σε περιορισμούς. Οι **«υπό όρους» εξισώσεις (conditional equations)** δηλώνονται ως εξής:

ceq term$_1$ = term$_2$ if cond

Η παραπάνω εξίσωση θα χρησιμοποιηθεί μόνο αν η cond πάρει την τιμή true. Πρέπει να σημειωθεί ότι όλες οι μεταβλητές της cond πρέπει να εμφανίζονται στον **term$_1$.**

Στο παρακάτω παράδειγμα φαίνεται μια απλή ενότητα, στο σώμα της οποίας ορίζονται όσα χαρακτηριστικά του βασικού συντακτικού της CafeOBJ αναφέρθηκαν παραπάνω.

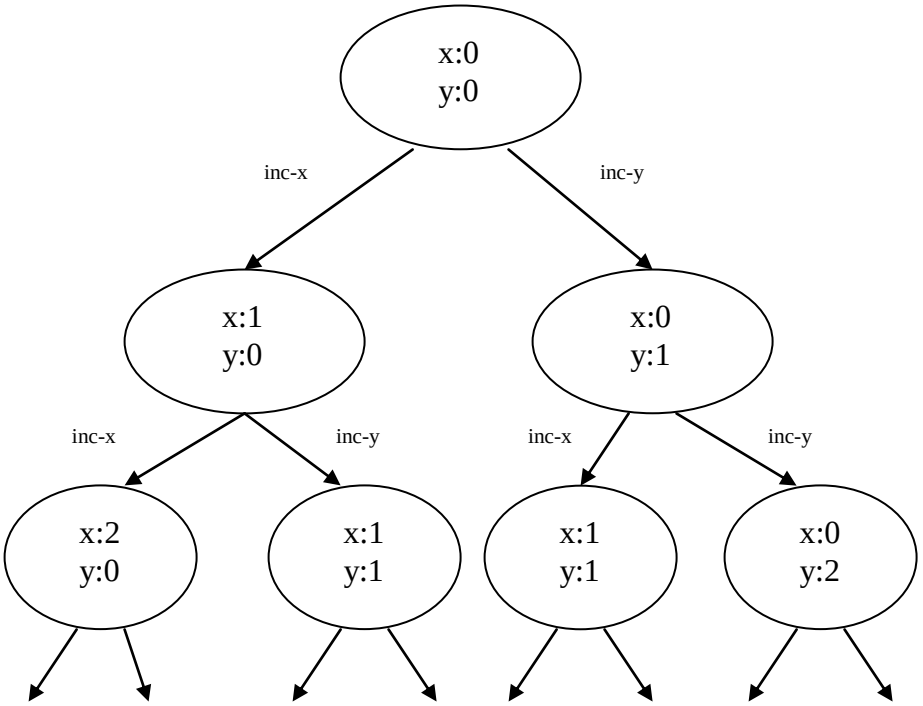
```
mod! PNAT{
    [Nat]
    op o :    →Nat
    op s : Nat→Nat
    op + : Nat Nat → Nat
    vars X,Y : Nat
    eq 0+Y=Y
    eq s(X)+Y= s(X,Y)
}
```

3.ΠΑΡΑΤΗΡΗΣΙΜΑ ΣΥΣΤΗΜΑΤΑ ΜΕΤΑΒΑΣΗΣ, ΠΣΜ(OBSERVATION TRANSITION SYSTEMS, OTS)

α) Εισαγωγή

Πριν ορίσουμε τα Παρατηρήσιμα Συστήματα Μετάβασης ΠΣΜ , είναι χρήσιμο να ορίσουμε τις **μηχανές κατάστασης** και τις **προδιαγραφές συμπεριφοράς**.

Μια απλή μηχανή κατάστασης INCXY και το διάγραμμα μετάβασης των καταστάσεών της, απεικονίζεται σχηματικά ως εξής:



Η παραπάνω μηχανή κατάστασης INCXY προσδιορίζεται σε CafeOBJ ως εξής :

```
mod* INCXY { pr (NAT)
        [Sys]
        op init :    → Sys
        bops x y : Sys → Nat
        bops inc-x  inc-y : Sys → Sys
        var S : Sys
        -          init
        eq  x(init) = 0
        eq  y(init) = 0
        -          for inc-x
        eq  x(inc-x(S)) = x(S) + 1
        eq  y(inc-y(S)) = y(S)
        -          for inc-y
        eq  x(inc-x(S)) = x(S)
        eq  y(inc-y(S)) = y(S) + 1
}
```

Εδώ το Nat αντιστοιχεί στο σύνολο των φυσικών αριθμών και αποτελεί ορατό τύπο (visible sort) και το Sys αντιστοιχεί στο χώρο κατάστασης της μηχανής INCXY και αποτελεί κρυφό τύπο (hidden sort). Επίσης, τα x και y είναι συναρτήσεις παρατήρησης, ενώ τα inc-x και inc-y είναι συναρτήσεις μετάβασης.

Οι **προδιαγραφές συμπεριφοράς (behavioral specifications)** είναι αλγεβρικές προδιαγραφές συστημάτων.

- Όλος ο χώρος καταστάσεων συμβολίζεται ως κρυφός τύπος.
- Οι καταστάσεις χαρακτηρίζονται από συναρτήσεις παρατήρησης.
- Οι καταστάσεις μετάβασης αντιστοιχούν σε συναρτήσεις μετάβασης.

Σε κάθε περίπτωση, οι προδιαγραφές συμπεριφοράς γενικεύουν ένα συμβατικό σύστημα παρατήρησης ως προς το ότι ακόμα κι αν κάθε συνάρτηση παρατήρησης επιστρέφει την ίδια τιμή για δύο καταστάσεις, οι καταστάσεις αυτές μπορούν να είναι διαφορετικές.

β) Παρατηρήσιμα συστήματα μετάβασης ΠΣΜ

Με βάση τα παραπάνω, καταλήγουμε στο ότι τα Παρατηρήσιμα συστήματα μετάβασης (ΠΣΜ) είναι μαθηματικά μοντέλα (συστήματα μετάβασης ή μηχανές κατάστασης) συστημάτων.

Επίσης, τα ΠΣΜ μπορούν να θεωρηθούν ως μια υπο-κλάση των συμπεριφοριακών προδιαγραφών, που αντιστοιχεί σε συμβατικά συστήματα μετάβασης.

Όταν κάθε παρατήρηση επιστρέφει την ίδια τιμή για δύο διαφορετικές καταστάσεις σε ένα ΠΣΜ, οι δύο αυτές καταστάσεις είναι ίσες με βάση το ΠΣΜ.

Τώρα μπορούμε να ορίσουμε τα ΠΣΜ.

ΟΡΙΣΜΟΣ

Ένα **ΠΣΜ S(O,I,T)** αποτελείται από τα ακόλουθα :

(i) **O : Ένα σύνολο συναρτήσεων παρατήρησης (παρατηρητές).**

Κάθε $o \in O$ είναι μια συνάρτηση παρατήρησης $o : Y \rightarrow D$,

Y: σύνολο καταστάσεων, D: σύνολο τύπων δεδομένων κάθε παρατηρούμενης τιμής.

Δύο καταστάσεις $u_1, u_2 \in Y$ είναι ίσες στο ΠΣΜ S , $u_1 =_s u_2$, αν $\forall\ o \in O$, $o(u_1)=o(u_2)$.

(ii) **I : σύνολο αρχικών καταστάσεων της αφηρημένης μηχανής $I \subset Y$.**

(iii) **T : σύνολο κανόνων μετάβασης.** Κάθε $\tau \in T$ είναι μια συνάρτηση μετάβασης $T: Y/_{=_s} \rightarrow Y/_{=_s}$. Για κάθε κατάσταση $u \in Y$, το τ(u) είναι η επόμενη κατάσταση του u. Οι συναρτήσεις μετάβασης περιέχουν μια συνθήκη c_τ η οποία είναι ένα κατηγόρημα στις καταστάσεις του Y και καλείται **αποτελεσματική συνθήκη.** Αν η c_τ είναι ψευδής, τότε $\tau(u) =_s u$.

Μια εκτέλεση της αφηρημένης μηχανής S είναι μία άπειρη ακολουθία καταστάσεων $u_0, u_1, \ldots$ που ικανοποιούν τα εξής :

- Αρχικοποίηση : $u_0 \in I$
- Συνέπεια : Για κάθε $i \in \{0,1,\ldots\}$, $u_{i+1} =_s \tau(u_i)$ για κάποιο $\tau \in T$.

Μια κατάσταση καλείται **προσιτή(reachable)** για το ΠΣΜ S, αν και μόνο αν μπορεί να εμφανιστεί ως μια εκτέλεση της αφηρημένης μηχανής S.

Σε αυτό το κεφάλαιο θα ασχοληθούμε μόνο με αποδείξεις ιδιοτήτων οι οποίες θεωρούνται **invariant (αμετάβλητες)** , δηλαδή που ισχύουν σε κάθε κατάσταση του OTS S.

4. ΜΕΤΑΦΟΡΑ ΤΩΝ ΠΣΜ ΣΤΗΝ CAFEOBJ

Η μεταφορά ενός ΠΣΜ σε CafeOBJ γίνεται με φυσικό τρόπο. Έστω λοιπόν το σύνολο καταστάσεων Υ, το οποίο δηλώνουμε με ένα hidden sort- έστω Η- και έστω οι τύποι δεδομένων D_k $(k=i_1,\ldots,i_m)$ οι οποίοι δηλώνονται με τις visible sorts V_k $(k=i_1,\ldots,i_m)$.

Η συνάρτηση παρατήρησης που αντιστοιχεί στις παρατηρούμενες τιμές $o_{i_1},\ldots,o_{i_m} \in O$ δηλώνεται ως εξής:

bop o : H V_{i_1} ,…..,V_{i_m} → V

Κάθε αρχική κατάσταση του συνόλου Ι, έστω init, δεν έχει ορίσματα και δηλώνεται ως εξής:

op init : → H

Αρχικοποιώντας τα $o_{i_1},\ldots,o_{i_m}$ δηλώνουμε:

eq o(init, X_{i_1} ,….,X_{i_m})=f(X_{i_1} ,….,X_{i_m})

X: μεταβλητή τύπου X_k ,

$f(X_1,\ldots,X_m)$: όρος που αντιστοιχεί στο $f(i_1,\ldots,i_m)$.

Έστω οι τύποι δεδομένων D_k $(k=j_1,\ldots,j_n)$ και οι visible sorts V_k $(k=j_1,\ldots,j_n)$ ‘όπως προηγουμένως. Μια συνάρτηση μετάβασης δηλώνεται ως:

bop a : H V_{j_1} ,…..,V_{j_n} → H

Οι συναρτήσεις μετάβασης τ_{j_1} ,….,τ_{j_n} εφαρμόζονται στις καταστάσεις του συνόλου καταστάσεων Υ. Για να είναι αποτελεσματικές και να επιτύχουμε αλλαγή της τιμής των παρατηρητών $o_{i_1},\ldots,o_{i_m}$ πρέπει να ικανοποιούνται οι αποτελεσματικές συνθήκες τους. Αυτό, δηλώνεται στην CafeOBJ ως εξής:

ceq o(a(W,X_{j_1} ,…..,X_{j_n}), X_{i_1} ,….,X_{i_m})=e-a(W, X_{j_1} ,…..,X_{j_n} , X_{i_1} ,….,X_{i_m})

if c-a(W, X_{j_1} ,…..,X_{j_n})

όπου W: μια μεταβλητή για το Η,

X_k : μεταβλητή για τον τύπο δεδομένων V_k ,

$a(W,X_{j_1},\ldots,X_{j_n})$: η επόμενη κατάσταση της W,

e-a(W, X_{j_1} ,…..,X_{j_n} , X_{i_1} ,….,X_{i_m}) : πιθανή νέα τιμή του παρατηρητή,

c-a(W, X_{j_1} ,…..,X_{j_n}) : η αποτελεσματική συνθήκη

Αν η αποτελεσματική συνθήκη δεν ισχύει, τότε η τιμή του παρατηρητή δεν αλλάζει.

ΚΕΦΑΛΑΙΟ 4
ΕΠΑΛΗΘΕΥΣΗ ΕΝΟΣ ΠΣΜ

1.ΣΥΝΘΕΤΙΚΗ ΑΠΟΔΕΙΞΗ ΤΩΝ ΑΜΕΤΑΒΛΗΤΩΝ ΚΑΤΑΣΤΑΣΕΩΝ

Έστω ότι θέλουμε να αποδείξοθμε ότι ένα σύστημα έχει μια αμετάβλητη κατάσταση (invariant property). Αρχικά μοντελοποιούμε το σύστημα ως ΠΣΜ σε CafeOBJ.

Έστω H το hidden sort που δηλώνει το σύνολο καταστάσεων Y και έστω $pred_1(s,x_1)$ η αμετάβλητη κατάσταση, όπου s: ελεύθερη μεταβλητή για τις καταστάσεις και x_1 οι υπόλοιπες ελεύθερες μεταβλητές. Συχνά είναι αδύνατον να αποδείξουμε ότι η $pred_1(s,x_1)$ είναι αμετάβλητη, ελέγχοντας την μόνη της.

Υποθέτουμε λοιπόν ότι είναι δυνατόν να αποδείξουμε ότι η $pred_1(s,x_1)$ μαζί με τα n-1 κατηγορήματα είναι αμετάβλητα στο ΠΣΜ και ότι $pred_2(s,x_2)$,…., $pred_n(s,x_n)$ είναι τα υπόλοιπα n-1 κατηγορήματα. Αποδεικνύουμε ότι η

$pred(s,x_1,\ldots,x_n) = pred_1(s,x_1) \wedge \ldots \wedge pred_n(s,x_n)$ είναι αμετάβλητη, άρα και η $pred_1(s,x_1)$.

Παρόλο που κάποιες φορές οι αμετάβλητες καταστάσεις αποδεικνύονται με αναγωγή και/ή ανάλυση κατά περίπτωση, συχνά χρειάζεται να χρησιμοποιήσουμε επαγωγή, ιδιαίτερα στον αριθμό των μεταβατικών κανόνων που εφαρμόζονται ή εκτελούνται. Θεωρούμε ότι η $pred(s,x_1,\ldots,x_n)$ αποδεικνύεται με επαγωγή στον αριθμό των κανόνων μετάβασης που χρησιμοποιούνται. Επίσης, θεωρούμε ένα επαγωγικό βήμα στο οποίο ισχύει ότι κάθε μεταβατικός κανόνας που δηλώνεται από μια μεταβατική συνάρτηση a σε CafeOBJ διατηρεί την $pred(s,x_1,\ldots,x_n)$. Επομένως είναι αρκετό να δείξουμε ότι:

$$\mathbf{pred(s,x_1,\ldots,x_n) \Rightarrow pred(a(s,y),x_1,\ldots,x_n)} \qquad \mathbf{(1)}$$

όπου y: ορίσματα της συνάρτησης μετάβασης εκτός του s.

Συχνά δε μπορούμε να αποδείξουμε το παραπάνω γιατί η επαγωγική υπόθεση $pred(s,x_1,\ldots,x_n)$ είναι πολύ αδύναμη. Σ'αυτή την περίπτωση ενδυναμώνουμε την υπόθεση προσθέτοντας τον τύπο SIH :

$pred(s,t_1^1,\ldots,t_n^1) \wedge \ldots \wedge pred(s,t_1^m,\ldots,t_n^m)$ όπου $t_1^i,\ldots,t_n^i$ (i=1,….,m) είναι λίστα όρων. Τότε η (1) αντικαθίσταται με την απόδειξη του :

$$\mathbf{(SIH \wedge pred(s,x_1,\ldots,x_n)) \Rightarrow pred(a(s,y),x_1,\ldots,x_n)}$$

Ο παραπάνω τύπος μπορεί να αποδεχθεί συνθέτοντας τις αποδείξεις των ακόλουθων τύπων:

$$\mathbf{(SIH \wedge pred(s,x_1,\ldots,x_n)) \Rightarrow pred_1(a(s,y),x_1)}$$

$$\mathbf{.} \qquad \mathbf{(2)}$$

$$\mathbf{.}$$

$$\mathbf{(SIH \wedge pred(s,x_1,\ldots,x_n)) \Rightarrow pred_n(a(s,y),x_n)}$$

Ή αρκεί να δείξουμε τους

$$\mathbf{pred_1(s,x_1) \Rightarrow pred_1(a(s,y),x_1)}$$

$$\mathbf{.} \qquad \mathbf{(3)}$$

.

$\mathbf{pred}_n\ \mathbf{(s,x}_n\mathbf{)} \Rightarrow \mathbf{pred}_n\ \mathbf{(a(s,y),\ x}_n\mathbf{)}$

επειδή ο i-τύπος των (2) μπορεί να συναχθεί από τον i-τύπο του (3) με $1 \leq i \leq n$.

Έστω ότι ο τύπος $\mathbf{pred}_i\ \mathbf{(s,x}_i\mathbf{)} \Rightarrow \mathbf{pred}_i\ \mathbf{(a(s,y),x}_i\mathbf{)}$, $1 \leq i \leq n$, δε μπορεί να αποδεχθεί στη μορφή που ε΄ναι γιατί η επαγωγική του υπόθεση είναι πολύ αδύναμη και έστω ότι η $\text{pred}_j(s,u_j)$, $1 \leq j \leq n$ ενισχύει την επαγωγική υπόθεση. Αρκεί να δείξουμε τον :

$\mathbf{(pred}_j\mathbf{(s,u}_j\mathbf{)} \wedge \mathbf{pred}_i\ \mathbf{(s,x}_i\mathbf{)) \Rightarrow pred}_i\ \mathbf{(a(s,y),x}_i\mathbf{)}$

Γενικά το $\text{pred}_{j_1}(s,u_{j_1}) \wedge \ldots.. \wedge \text{pred}_{j_k}(s,u_{j_k})$, $1 \leq j_1,\ldots,j_k \leq n$ ενισχύει την επαγωγική υπόθεση. Έστω ότι ο SIH_i είναι ο ενισχυτικός τύπος για την επαγωγική υπόθεση $\text{pred}_i(s,x_i)$. Τότε η απόδειξη του i-τύπου της (3) αντικαθίσταται με την απόδειξη του :

$\mathbf{(SIH}_i \wedge \mathbf{pred}_i\ \mathbf{(s,x}_i\mathbf{)) \Rightarrow pred}_i\ \mathbf{(a(s,y),x}_i\mathbf{)}$ **(4)**

Στη συνέχεια «σπάμε» σε l υποπεριπτώσεις για να αποδείξουμε την (4). Οι l υποπεριπτώσεις δηλώνονται με l τύπους $\text{case}_1^i,\ldots,\text{case}_l^i$ που πρέπει να ικανοποιούν την :

$\mathbf{(case}_1^i \vee \mathbf{\ldots.} \vee \mathbf{case}_l^i\mathbf{)=true}$

Τότε η απόδειξη της (4) αντικαθίστατι με τις αποδείξεις των ακόλουθων l τύπων:

$\mathbf{(case}_1^i \wedge \mathbf{SIH}_i \wedge \mathbf{pred}_i\ \mathbf{(s,x}_i\mathbf{)) \Rightarrow pred}_i\ \mathbf{(a(s,y),x}_i\mathbf{)}$

. **(5)**

.

$\mathbf{(case}_l^i \wedge \mathbf{SIH}_i \wedge \mathbf{pred}_i\ \mathbf{(s,x}_i\mathbf{)) \Rightarrow pred}_i\ \mathbf{(a(s,y),x}_i\mathbf{)}$

Συνεπώς, από τα παραπάνω συμπεραίνουμε ότι οι n αμετάβλητες καταστάσεις μπορούν να αποδειχθούν συνθετικά, ακόμα κι αν εξαρτώνται η μια από την άλλη, δεδομένου ότι η i-αμετάβλητη κατάσταση χρησιμοποιείται για να ενισχύσει την επαγωγική υπόθεση της j-αμετάβλητης κατάστασης και αντιστρόφως. Η αρχική αμετάβλητη κατάσταση $\text{pred}_1(s,x_1)$ που θέλαμε να αποδείξουμε, διαιρείται σε επιμέρους αμετάβλητες καταστάσεις.

Τα Proof Scores στα ΠΣΜ/CafeOBJ βασίζονται στα παραπάνω και επομένως, μπορούμε να γράψουμε Proof Scores των n αμετάβλητων καταστάσεων μεμονωμένα.

2. PROOF SCORES

Έστω ότι γράφουμε Proof Scores για τις n αμετάβλητες καταστάσεις που περιγράφηκαν στην προηγούμενη κατάσταση.

Πρώτα γράφουμε ένα module, έστω INV, όπου η $pred_i$ (s,x_i) (i=1,…,n) εκφράζεται σε CafeOBJ ως εξής:

op inv_1 : H V_1 → Bool
….
op inv_n : H V_n → Bool
eq inv(W,X_1) = $pred_1$ (W,X_1) .
……
eq inv_n (W,X_n) = $pred_n$ (W,X_n)

V_i (i=1,…,n) : λίστα visible sorts που αντιστοιχούν στα X_i,

W : μεταβλητή για το hidden sort H,

X_i : λίστα μεταβλητών για τα V_i.

Ο όρος $pred_i$ (W,X_i) δηλώνει την $pred_i$ (s,x_i) (i=1,…,n).

Στο module, δηλώνουμε επίσης τις σταθερές X_i για τα V_i. Στα Proof Scores, μια μεταβλητή που δεν περιορίζεται, χρησιμοποιείται για τη δήλωση ενός αυθαίρετου αντικειμένου συγκεκριμένου τύπου. Για παράδειγμα, αν δηλώσουμε μια σταθερά x για Nat τότε είναι το visible sort για τους φυσικούς αριθμούς και το x δηλώνει έναν αυθαίρετο φυσικό αριθμό. Τέτοιες σταθερές περιορίζονται με εξισώσεις, οι οποίες μας δίνουν τη δυνατότητα να διαιρέσουμε το σύνολο καταστάσεων. Υποθέτουμε ότι μια κατάσταση διαιρείται στα δύο: μία όπου το x είναι ίσο με 0 και μια άλλη όπου το x είναι διάφορο του μηδενός και όπου το x είναι παραπάνω από 0. Τα παραπάνω εκφράζονται με τις εξής εξισώσεις:

eq x=0

eq(x>0)=true

Θα περιγράψουμε κυρίως το Proof Score της i-αμετάβλητης κατάστασης. Έστω init η αρχική κατάσταση του συστήματος. Για να δείξουμε ότι η $pred_i$ (s,x_i) ισχύει σε κάθε αρχική κατάσταση, γράφουμε:

open INV
 red inv_i (init, x_i) .
close

Στη συνέχεια γράφουμε ένα module, έστω ISTEP, όπου οι δύο σταθερές s,s’ δηλώνονται και υποδηλώνουν κάθε κατάσταση και κάθε επόμενη κατάσταση μετά την εφαρμογή του κανόνα μετάβασης

στην κατάσταση. Τα κατηγορήματα προς απόδειξη σε κάθε επαγωγική περίπτωση, εκφράζονται σε CafeOBJ ως εξής:

op $istep_1$: V_1 -> Bool
….
op $istep_n$: V_n -> Bool
eq $istep_1$ (X) = inv_1 (s,X_1) implies inv_1 (s',X_1) .
…..
eq $istep_n$ (X) = invn(s,X_n) implies inv_n (s',X_n)

(Αντιστοιχούν στην (3) της προηγούμενης παραγράφου).

Κάθε επαγωγική περίπτωση, συνήθως διαιρείται σε υποπεριπτώσεις με τα βασικά κατηγορήματα να δηλώνονται σε CafeOBJ προδιαγραφές. Έστω ότι αποδεικνύουμε πως κάθε κανόνας μετάβασης που υποδηλώνεται από μια συνάρτηση μετάβασης a σε CafeOBJ διατηρεί το $pred_i$ (s,x_i). Διαιρούμε σε l υποπεριπτώσεις $case^i_1$,….,$case^i_l$. Τότε ο κώδικας σε CafeOBJ που δείχνει ότι ο κανόνας μετάβασης διατηρεί το $pred_i$ (s,x_i) για την περίπτωση $case^i_j$ (j=1,…,l) είναι ο εξής:

open ISTEP
Declare constants denoting arbitrary objects.
Declare equations denoting $case^i_j$.
Declare equations denoting facts if necessary.
eq s' = a(s, y) .
red $istep_i$ (x_i) .
close

όπου y: λίστα σταθερών που χρησιμοποιούνται ως ορίσματα στη συνάρτηση μετάβασης a.

Αν το $istep_i$ (x_i) αναχθεί σε true, αποδεικνύεται ότι ο κανόνας μετάβασης διατηρεί το $pred_i$ (s,x_i) στην υποπερίπτωση j, η οποία αντιστοιχεί στην απόδειξη του j-τύπου της (5) της προηγούμενης παραγράφου. Το $istep_i$ (x_i) ανάγεται στο εξής:

(SIH_i and inv_i (s, x_i)) implies inv_i (s',x_i)

ή

SIH_i implies $istep_i$ (x_i).

Οι παραπάνω πληροφορίες αντλήθηκαν από το άρθρο "Proof Scores in the OTS/CafeOBJ method" των K.Ogata και K.Futatsugi.

ΚΕΦΑΛΑΙΟ 5
ΕΦΑΡΜΟΓΗ 1: ΤΟ ΠΡΩΤΟΚΟΛΛΟ SSL

1.ΑΝΑΛΥΣΗ ΤΟΥ ΠΡΩΤΟΚΟΛΛΟΥ ΧΕΙΡΑΨΙΑΣ ΤΟΥ SSL

Μια σχηματική περίληψη του πρωτοκόλλου χειραψίας του SSL είναι η εξής:

ClientHello	A→B	$Rand_A$, ListOfChoices
ServerHello	B→A	$Rand_B$, SID, Choice
Certificate	B→A	$Cert_B$
KeyExchange	A→B	E_{K_B} (PMS)
ClientFinished	A→B	$E_{Clientkey}$ (ClientFinish)
ServerFinished	B→A	$E_{ServerKey}$(ServerFinish)
ClientHello2	A→B	$Rand_A$, SID
ServerHello2	B→A	$Rand_B$, SID, Choice
ServerFinished2	B→A	$E_{ServerKey}$(ServerFinish2)
ClientFinished2	A→B	$E_{Clientkey}$ (ClientFinish2)

Στο παραπάνω, το A υποδηλώνει τον πελάτη και το B τον εξυπηρετητή. Οι έξι πρώτες ανταλλαγές μηνυμάτων πραγματοποιούνται για την πλήρη διαπραγμάτευση της cipher και των παραμέτρων ασφαλείας και οι υπόλοιπες για την επανάληψη μιας προηγούμενης συνόδου ή για την επικάλυψη της τρέχουσας συνόδου.

Τα κρυπτογραφικά αρχέτυπα που χρησιμοποιούνται στο πρωτόκολλο είναι η H(.) (μια μονόπλευρη συνάρτηση κατακερματισμού), η $E_K(.)$ (μια κρυπτογρφική συνάρτηση συμμετρικού ή ασύμμετρου κλειδιού) και η $S_X(.)$ (μια συνάρτηση ψηφιακής υπογραφής με αρχέτυπο το ιδιωτικό κλειδί του X). Βασικές ποσότητες οι οποίες προκύπτουν στο πρωτόκολλο είναι η $Rand_X$ (ένας τυχαίος αριθμός που δημιουργείται από το αρχέτυπο X), η ListOfChoices (μια λίστα chipper ακολουθιών), η Choice (μια ακολουθία chipper), το SID (ένας αναγνωριστικό της συνόδου), το PMS (προ-κύριο μυστικό) και το K_X (το αρχέτυπο του δημοσίου κλειδιού του X). Τα σύνθετα δεδομένα που προκύπτουν στο πρωτόκολλο είναι τα εξής:

$Cert_X$: $X,K_X,S_{CA}(x,K_X)$
ClientKey : H(A,PMS, $Rand_A$, $Rand_B$)
ServerKey : H(B,PMS, $Rand_A$, $Rand_B$)
ClientFinish : H(“client”,A,B,SID,ListOfChoices,Choice, $Rand_A$, $Rand_B$,PMS)
ServerFinish : H(“server”, ,B,SID,ListOfChoices,Choice, $Rand_A$, $Rand_B$,PMS)
ClientFinish2 : H(“client”,A,B,SID,ListOfChoices,Choice, $Rand_A$, $Rand_B$,PMS)

ServerFinish2 : H(“server”, ,B,SID,ListOfChoices,Choice, $Rand_A$, $Rand_B$,PMS)

Η Choice δεν είναι μόνο μια ακολουθία cipher, αλλά περιέχει και το νούμερο της έκδοσης και έναν αλγόριθμο συμπίεσης.

Υποθέτουμε τα ακόλουθα: ο εξυπηρετητής στέλνει το πιστοποιητικό του στον πελάτη μόνο όταν η χειραψία έχει πραγματοποιηθεί, ο εξυπηρετητής δεν στέλνει ούτε ServerkeyExchange ούτε CertificateRequest μηνύματα, τα Certificate μηνύματα του εξυπηρετητή έχουν το ρόλο των ServeHelloDone μηνυμάτων, ο πελάτης δεν στέλνει ούτε Certificate ούτε CertificateVerify μηνύματα, τα ChangeCipherSpec μηνύματα υπονοούνται, υπάρχει μόνο ένα αξόπιστο πιστοποιητικό που υποδηλώνεται από το CA, η μέθοδος που χρησιμοποιείται για την ανταλλαγή των προ-κύριων μυστικών είναι μόνο η RSA, το περιεχόμενο ένος μηνύματος ολοκλήρωσης δεν αποτελεί τον κατακερματισμό των παραμέτρων ασφαλείας και των μηνυμάτων χειραψίας που ανταλλάχθηκαν. Τα ολοκληρωμένα μηνύματα που ανταλλάχθηκαν σε μια σύντομη διαδικασία χειραψίας στέλνονται με σαφήνεια.

2.ΠΑΡΑΔΟΧΕΣ

Υποθέτουμε ότι εκτός από αξιόπιστους εντολείς, υπάρχουν και αρκετοί κακόβουλοι και ότι το κρυπτογρφικό σύστημα που χρησιμοποιείται είναι τέλειο. Οι αξιόπιστοι εντολείς ακολουθούν το πρωτόκολλο απακριβώς ενώ οι κακόβουλοι σκοπεύουν να δράσουν ενάντια στο πρωτόκολλο. Οι εισβολείς πιθανώς να κάνουν τα εξής:

- Να κρυφακούσουν τα μηνύματα που ρέουν στο δίκτυο.
- Να περισυλλέξουν οποιαδήποτε πληροφορία συμπεριλαμβάνεται στο μήνυμα. (Παρόλο που ο εισβολέας μπορεί να αποκρυπτογραφήσει την cipher μόνο αν ξέρει το κλειδί και δε μπορεί να υπολογίσει τις proimages του κατακερματισμού αν δεν τις γνωρίζει.)
- Να εξαπατήσει και να στείλει μηνύματα βασισμένος στις πληροφορίες που περισυνέλλεξε.(Παρόλο που μπορεί να κρυπτογραφήσει και/ή να υπογράψει κάτι μόνο αν ξέρει το κρυπτογραφικό κλειδί και/ή την υπογραφή και δε μπορεί να μαντέψει τις άγνωστες μυστικές αξίες.)

3. ΜΟΝΤΕΛΟΠΟΙΗΣΕΙΣ

(α) Μοντελοποίηση μηνυμάτων

Πριν μοντελοποιήσουμε τα μηνύματα που ανταλλάσονται στο πρωτόκολλο, μοντελοποιούμε τις ποσότητες που απαρτίζουν τα μηνύματα. Δηλώνουμε τις ακόλουθες visible sorts και τους ανταποκρινόμενους σ'αυτά κατασκευαστές δεδομένων γι'αυτές τις ποσότητες:

- **Principal** υποδηλώνει τους εντολείς. Υπάρχουν δύο εντολείς: ο εισβολέας που δηλώνεται ως **intruder** και η αρχή πιστοποίησης που δηλώνεται ως **ca**. Υποθέτουμε ότι οι intruder και ca δεν είναι ποτέ ίσοι. Το **Rand** δηλώνει τυχαίους αριθμούς. Το **Choice** δηλώνει ακολουθίες cipher. Το **Sid** δηλώνει την ταυτότητα της συνόδου. Το **ListOfChoices** δηλώνει λίστες ακολουθιών cipher. Ο τελεστής **_\in_** είναι το κατηγόρημα ιδιότητας για τις λίστες.
- Το **Secret** υποδηλώνει τις μυστικές αξίες που καθιστούν το προ-κύριο μυστικό καθολικά μοναδικό και να μη μπορεί να μαντευθεί. Το **pms** δηλώνει τα προ-κύρια μυστικά. Δοσμένων δύο εντολέων a,b και μια τιμή μυστικού s, ένα προ-κύριο μυστικό που γεννάται από τον πελάτη a για τον εξυπηρετητή b, δηλώνεται ως **pms(a,b,s)**.
- Το **PubKey** δηλώνει τα δημόσια κλειδιά. Το δημόσιο κλειδί του εντολέα a δηλώνεται ως k(a).
- Το **Sig** δηλώνει ψηφιακές υπογραφές ζευγαριών εντολέα και δημοσίου κλειδιού. Η ψηφιακή υπογραφή ενός ζεύγους του εντολέα b και του δημοσίου κλειδιού k που έχει υπογραφεί από τον εντολέα a, δηλώνεται ως sig(a,k,b).
- Το **Cert** δηλώνει τα πιστοποιητικά των δημοσίων κλειδιών. Δοσμένου του εντολέα a, του δημοσίου κλειδιού k και της υπογραφής g, το πιστοποιητικό δηλώνεται ως cert(a,k,g).
- Το **Key** δηλώνει τους κατακερματισμούς που χρησιμοποιούνται ως συμμετρικά κλειδιά για να κρυπτογραφηθούν τα ολοκληρωμένα μηνύματα. Δοσμένου του εντολέα a, ενός προ-κύριου μυστικού pms και δύο τυχαίων αριθμών r1,r2, ο κατακερματισμός αυτών των ποσοτήτων δηλώνεται ως k(a,pms,r1,r2).
- Το **CFinish** δηλώνει το ClientFinish. Δοθέντων δύο εντολέων a,b , μια ταυτότητα συνόδου i, μιας λίστας ακολουθιών cipher l, μιας ακολουθίας cipher c, δύο τυχαίων αριθμών r1,r2 και ενός προ-κύριου μυστικού s, το αντίτοιχο ClientFinish δηλώνεται ως cfin(a,b,i,l,c,r1,r2,s).
- Το **SFinish** δηλώνει το ServerFinish. Δοθέντων δύο εντολέων a,b , μια ταυτότητα συνόδου i, μιας λίστας ακολουθιών cipher l, μιας ακολουθίας cipher c, δύο τυχαίων αριθμών r1,r2 και ενός προ-κύριου μυστικού s, το αντίτοιχο ServerFinish δηλώνεται ως sfin(a,b,i,l,c,r1,r2,s).
- Το **CFinish2** δηλώνει το ClientFinish2. Δοθέντων δύο εντολέων a,b , μια ταυτότητα συνόδου i, μιας λίστας ακολουθιών cipher l, μιας ακολουθίας cipher c, δύο τυχαίων αριθμών r1,r2 και ενός προ-κύριου μυστικού s, το αντίτοιχο ClientFinish2 δηλώνεται ως cfin2(a,b,i,l,c,r1,r2,s).
- Το **SFinish2** δηλώνει το ServerFinish2. Δοθέντων δύο εντολέων a,b , μια ταυτότητα συνόδου i, μιας λίστας ακολουθιών cipher l, μιας ακολουθίας cipher c, δύο τυχαίων

αριθμών r1,r2 και ενός προ-κύριου μυστικού s, το αντίτοιχο ServerFinish2 δηλώνεται ως sfin2(a,b,i,l,c,r1,r2,s).

- Το **EncPms** δηλώνει τα προ-κύρια μυστικά (pre-master secrets) που κρυπτογραφούνται από τα δημόσια κλειδιά. Ένα προ-κύριο μυστικό pms που κρυπτογραφήθηκε από ένα δημόσιο κλειδί k δηλώνεται ως epms(k,pms).
- Το **EncCFin** δηλώνει το ClientFinish που κρυπτογραφείται από συμμετρικά κλειδιά. Το ClientFinish f που έχει κρυπτογραφηθεί από το συμμετρικό κλειδί k δηλώνεται ως ecfin(k,f).
- Το **EncSFin** δηλώνει το ServerFinish που κρυπτογραφείται από συμμετρικά κλειδιά. Το ServerFinish f που έχει κρυπτογραφηθεί από το συμμετρικό κλειδί k δηλώνεται ως esfin(k,f).
- Το **EncCFin2** δηλώνει το ClientFinish2 που κρυπτογραφείται από συμμετρικά κλειδιά. Το ClientFinish2 f που έχει κρυπτογραφηθεί από το συμμετρικό κλειδί k δηλώνεται ως ecfin2(k,f).
- Το **EncSFin2** δηλώνει το ServerFinish2 που κρυπτογραφείται από συμμετρικά κλειδιά. Το ServerFinish2 f που έχει κρυπτογραφηθεί από το συμμετρικό κλειδί k δηλώνεται ως esfin2(k,f).
- Το **Session** δηλωνει τον τετραπλασιασμό μιας ακολουθίας cipher, δύο τυχαίων αριθμών και ενός προ-κύριου μυστικού. Ένας τετραπλασιασμός της ακολουθίας cipher c, δύο τυχαίοι αριθμοί r1,r2 και ένα προ-κύριο μυστικό pms, δηλώνονται ως st(c,r1,r2,pms).

Για κάθε κατασκευαστή δεδομένων όπως ο pms, ορίζονται επίσης και οι τελεστές προβολής όπως οι client,server kai secret οι οποίοι επιστρέφουν ορίσματα. Για παράδειγμα, client(pms(a,b,s))=a, server(pms(a,b,s))=b και secret(pms(a,b,s))=s.

Αφού έχουμε υποθέσει ότι το κρυπτογραφικό μας σύστημα είναι τέλειο, μπορούμε να υποθέσουμε ότι, δοσμένων δύο διαφορετικών τιμών, τα δύο αποτελέσματα των συναρτήσεων κατακερματισμού θα είναι διαφορετικά. Συνεπώς,χρησιμοποιούμε πέντε διαφορετικά visible sorts Key,CFinish, SFinish, CFinish2, SFinish2 για τα πέντε είδη κτακερματισμού. Για αντίστοιχο λόγο, χρησιμοποιούμε τέσσερα διαφορετικά visible sorts EncCFin, EncSFin, EncCFin2, EncSFin2 για τα τέσσερα είδη cipher που κρυπτογραφούνται από τα συμμετρικά κλειδιά.

Έχουμε 10 τελεστές (κατασκευαστές δεδομένων) για να δηλώσουμε τους 10 τύπους μηνυμάτων. Οι κατασκευαστές είναι οι εξής:

```
op ch : Prin Prin Prin Rand ListOfChoices -> Msg
op sh : Prin Prin Prin Rand Sid Choice    -> Msg
op ct : Prin Prin Prin Cert               -> Msg
op kx : Prin Prin Prin EncPms             -> Msg
op cf : Prin Prin Prin EncCFin            -> Msg
op sf : Prin Prin Prin EncSFin          -> Msg
op ch2 : Prin Prin Prin Rand Sid        -> Msg
op sh2 : Prin Prin Prin Rand Sid Choice -> Msg
op cf2 : Prin Prin Prin EncCFin2        -> Msg
op sf2 : Prin Prin Prin EncSFin2        -> Msg
```

Msg είναι το visible sort που υποδηλώνει τα μηνύματα. Για τον κατασκευαστή δεδομένων x (x=ch,sh,ct,kx,cf,sf,ch2,sh2,sf2,cf2), το κατηγόρημα x προσδιορίζεται, το οποίο ελέγχει αν το x είναι είναι πράγματι μήνυμα. Δοσμένου ενός όρου που υποδηλώνει ένα μήνυμα, οι προβολείς crt,src,dst επιστρέφουν το πρώτο,το δεύτερο και το τρίτο όρισμα του τύπου αντιστοίχως και οι προβολείς rand, list, choice, sid, cert, epms, ecfin, esfin, ecfin2, esfin2 επιστρέφουν άλλα ορίσματα αντιστοίχως,αν υπάρχουν.

Το πρώτο, δεύτεο και τρίτο όρισμα κάθε κατασκευαστή δεδομένων εννοεί τον πραγματικό αποστολέα, τον εμφανιζόμενο ως αποστολέα και τον παραλήπτη του αντίστοιχου μηνύματος. Το πρώτο όρισμα είναι μετά-πληροφορία και είναι διαθέσιμη μόνο στον εξωτερικό παρατηρητή και στον εντολέα του μηνύματος και δε μπορεί να το πλαστογραφήσει ο εισβολέας. Τα υπόλοιπα ορίσματα μπορούν να πλαστογραφηθούν από τον εισβολέα. Συνεπώς, θεωρούμε ότι υπάρχει ένα μήνυμα στο δίκτυο. Πράγματι, ο εντολέας που υποδηλώνεται από το πρώτο όρισμα έχει στείλει το μήνυμα. Αν το πρώτο όρισμα είναι ο εισβολέας και το δεύτερο δεν είναι ο εισβολέας, τότε το μήνυμα έχει πλαστογραφηθεί από τον εισβολέα.

Αυτή η μοντελοποίηση των μηνυμάτων μας επιτρέπει να περιγράψουμε ιδότητες όπως το ότι ένα μήνυμα που παραλήφθηκε από έναν εντολέα πράγματι προέρχεται από τον εμφανιζόμενο ως αποστολέα. Για παράδειγμα, υποθέτουμε ότι ο εντολέας a λαμβάνει ένα μήνυμα που δηλώνεται ως sf(b2,b1,a,esfin). Αφού τα μηνύματα δε διαγράφονται ποτέ από το δίκτυο, τότε το μήνυμα sf(b2,b1,a,esfin) που υπάρχει στο δίκτυο, πράγματι προέρχεται από τον b1.

(β) Μοντελοποίηση του Δικτύου

Το δίκτυο μοντελοποιείται ως ένα υπερσύνολο μηνυμάτων που χρησιμοποιείται ως μια αποθήκη που ο εισβολέας μπορεί να χρησιμοποιήσει. Επίσης, το δίκτυο χρησιμοποιείται ως η ιδιωτική μνήμη κάθε εντολέα που του υπενθυμίζει να στέλνει μηνύματα, των οποίων το πρώτο όρισμα να δηλώνει τον εντολέα. Κάθε μήνυμα που έχει σταλεί ή έχει μπει στο δίκτυο έστω μια φορά, υποτίθεται ότι δε μπορεί ποτέ να διαγραφεί γιατί ο εισβολέας μπορεί να αναπαράγει το μήνυμα επεναλαμβανόμενα, πορόλο που δε μπορεί να πλαστογραφήσει το πρώτο όρισμα. Συνεπώς, αν το δίκτυο είναι άδειο, σημαίνει ότι κανένα μήνυμα δεν έχει αποσταλεί.

Ο εισβολέας προσπαθεί να περισυλλέξει επτά είδη πληροφοριών από το δίκτυο. Οι πληροφορίες που τον ενδιαφέρουν είναι τα προ-κύρια μυστικά, οι ψηφιακές υπογραφές και πέντε είδη cipher.

Οι τυχαίοι αριθμοί, η ταυτότητα της συνόδου και τα δημόσια κλειδιά μπορούν εύκολα να υποκλαπούν αφού αποστέλλονται με σαφήνεια. Αφού τα συμμετρικά κλειδιά ClientKeys και ServerKeys δεν περιλαμβάνονται ποτέ σε κανένα μήνυμα, δε μπορούν να υποκλαπούν , εκτός από αυτά που υπολογίζονται από τα προ-κύρια μυστικά τα οποία γνωρίζει ο εισβολέας. Όμως δεν είναι απαραίτητο στον εισβολέα να γνωρίζει τα συμμετρικά κλειδιά.

Οι συλλογές των επτά πληροφοριών που υποκλέπτονται από το δίκτυο, δηλώνονται με τους παρακάτω τελεστές:

```
op cpms : Network -> ColPms
op csig : Network -> ColSig
op cepms : Network -> ColEncPms
op cecfin : Network -> ColEncCFin
op cesfin : Network -> ColEncSFin
op cecfin2 : Network -> ColEncCFin2
```

```
op cesfin2 : Network -> ColEncSFin2
```

Το **Network** είναι visble sort που υποδηλώνει τα δίκτυα. Το **ColX** είναι visible sort που δηλώνει τις συλλογές των ποσοτήτων που δηλώνονται από την visible sort X.

Οι τελεστές δηλώνονται με εξισώσεις. Παρακάτω φαίνονται οι εξισώσεις που δηλώνουν το cpms.

```
eq PMS \in cpms(void) = (client(PMS) = intruder) .
ceq PMS \in cpms(M,NW) = true
   if (kx?(M) and owner(k(epms(M))) = intruder
               and PMS = pms(epms(M))) .
ceq PMS \in cpms(M,NW) = PMS \in cpms(NW)
   if not(kx?(M) and owner(k(epms(M))) = intruder
                 and PMS = pms(epms(M))).
```

Η σταθερά void υποδηλώνει τον άδειο σάκο. Ο τελεστής _\in_ είναι το κατηγόρημα ιδιότητας των συλλογών. Ο τελεστής _,_ των M, NW είναι ο κατασκευαστής δεδομένων του σάκου. Η πρώτη εξίσωση εννοεί ότι το προ-κύριο μυστικό που παράγεται από τον εισβολέα, είναι πάντα διαθέσιμο σ'αυτόν και κάθε άλλο προ-κύριο μυστικό δεν ανήκει στην αρχική κατάσταση. Τα μηνύματα από τα οποία μπορούν να υποκλαπούν τα προ-κύρια μυστικά είναι μόνο τα Certificate μηνύματα. Αν υπάρχει στο δίκτυο ένα Certificate μήνυμα και η cipher του είναι κρυπτογραφημένη με το δημόσιο κλειδί του εισβολέα, τότε ο εισβολέας μπορεί να μάθει το προ-κύριο μυστικό του μηνύματος. Αυτό δηλώνεται στη δεύτερη εξίσωση. Η τρίτη εξίσωση λέει ότι κανένα προ-κύριο μυστικό δε μπορεί να υποκλαπεί από κάθε μη-Certificate μήνυμα και κάθε Certificate μήνυμα του οποίου οι cipher δεν είναι κρυπτογραφημένες με το δημόσιο κλειδί του εισβολέα. Οι υπόλοιποι τελεστές ορίζονται παρομοίως.

(γ) Μοντελοποίηση των αξιόπιστων εντολέων

Πριν μοντελοποιήσουμε τη συμπεριφορά των αξιόπιστων εντολέων, θα περιγράψουμε τις παρατηρούμενες τιμές εξωτερικά του πρωτοκόλλου. Υποθέτουμε ότι το δίκτυο, οι καταστάσεις συνόδου ματαξύ των δύο εντολέων, το σύνολο των τυχαίων αριθμών, το σύνολο των χρησιμοποιούμενων ταυτοτήτων συνόδων και το σύνολο των χρησιμοποιούμενων μυστικών είναι παρατηρήσιμα. Οι παρατηρητές δηλώνονται με τους τελεστές παρατήρησης σε CafeOBJ nw, ss, ur, ui και us αντιστοίχως και δηλώνονται ως εξής:

```
bop nw : Protocol -> Network
bop ss : Protocol Prin Prin Sid -> Session
bop ur : Protocol -> URand
bop ui : Protocol -> USid
bop us : Protocol -> USecret
```

Το Protocol είναι hidden sort που υποδηλώνει το χώρο καταστάσεων. Τα URand, USid και USecret είναι visible sorts που υποδηλώνουν σύνολα τυχαίων αριθμών, ταυτοτήτων συνόδων και μυστικών. Έστω p μια κατάσταση του πρωτοκόλλου. Το nw(p) δηλώνει το δίκτυο, το ur(p) το σύνολο των χρησιμοποιούμενων τυχαίων αριθμών, το ui(p) το σύνολο των χρησιμοποιούμενων ταυτοτήτων συνόδων και το us(p) το σύνολο χρησιμοποιούμενων μυστικών στην κατάσταση. Επιπλέον, έστω a,b οι εντολείς και i μια ταυτότητα συνόδου. Το ss(p,a,b,i) δηλώνει την κατάσταση συνόδου για τον εντολέα a, οποία αναγνωρίζεται απ'τον b με την ταυτότητα συνόδου i.

Η συμπεριφορά των αξιόπιστων εντολέων μοντελοποιείται με 12 είδη μεταβάσεων. 10 από αυτά αντιστοιχούν στην αποστολή των 10 τύπων μηνυμάτων. Τα υπόλοιπα 2 αντιστοιχούν στο λαμβανόμενο μήνυμα από τον πελάτη ServerFinished και στο λαμβανόμενο από τον εξυπηρετητή ClientFinished2 μήνυμα, αντιστοίχως. Τα 12 είδη μεταβάσεων δηλώνονται με τους τελεστές της CafeOBJ chello, shello, cert, kexch, cfin, sfin, compl, chello2, shello2, sfin2, cfin2 και compl2 ως εξής:

```
bop chello : Protocol Prin Prin Rand
                              ListOfChoices ->Protocol
bop shello : Protocol Prin Rand Sid Choice
                                       Msg -> Protocol
bop cert : Protocol Prin Msg Msg           -> Protocol
bop kexch : Protocol Prin Secret Msg Msg
                                      Msg -> Protocol
bop cfin : Protocol Prin Secret Msg Msg
                                   Msg Msg -> Protocol
bop sfin : Protocol Prin Msg Msg Msg Msg
                                      Msg -> Protocol
bop compl : Protocol Prin Secret Msg Msg
                            Msg Msg Msg Msg -> Protocol
bop chello2 : Protocol Prin Prin Secret Rand
                                      Sid -> Protocol
bop shello2 : Protocol Prin Rand Msg       -> Protocol
bop sfin2 : Protocol Prin Msg Msg          -> Protocol
bop cfin2 : Protocol Prin Secret Msg Msg
```

```
                                        Msg -> Protocol
bop compl2 : Protocol Prin Msg Msg Msg Msg -> Protocol
```

Οι 12 τελεστές μετάβασης ορίζονται με εξισώσεις. Οι εξισώσεις για το cert δηλώνονται ως εξής:

```
op c-cert : Protocol Prin Msg Msg -> Bool
eq c-cert(P,B,M1,M2) = (M1 \in nw(P) and M2 \in nw(P)
     and ch?(M1) and sh?(M2) and dst(M1) = B and
     crt(M2) = B and src(M2) = B and src(M1) = dst(M2)
     and choice(M2) \in list(M1)) .
ceq nw(cert(P,B,M1,M2))
    = ct(B,B,dst(M2),cert(B,k(B),sig(ca,B,k(B))))
    , nw(P) if c-cert(P,B,M1,M2) .
eq ss(cert(P,B,M1,M2),A2,B2,I2) = ss(P,A2,B2,I2) .
eq ur(cert(P,B,M1,M2)) = ur(P) .
eq ui(cert(P,B,M1,M2)) = ui(P) .
eq us(cert(P,B,M1,M2)) = us(P) .
ceq cert(P,B,M1,M2) = P if not c-cert(P,B,M1,M2).
```

Το ότι ο εντολέας a έστειλε το μήνυμα m1 στον εντολέα b, δηλώνεται από το ότι το m1 υπάρχει στο δίκτυο. Τα crt(m1) και src(m1) είναι ίσα με το a και το dst(m1) είναι ίσο με το b. Για να λαμβάνει ο a το μήνυμα m2 από τον b, πρέπει το m2 να υπάρχει στο δίκτυο έτσι ώστε το src(m2) να είναι ίσο με το b και το dst(m2) να είναι ίσο με το a. Αλλά, το crt(m2) μπορεί να μην είναι ίσο με το b αλλά με τον intruder. Οι υπόλοιποι τελεστές μετάβασης ορίζονται παρομοίως.

(δ) Μοντελοποίηση του εισβολέα

Μέρος του εισβολέα μοντελοποιήθηκε ως δίκτυο. Έχουμε ήδη ορίσει ποιες πληροφορίες μπορεί να υποκλέψει ο εισβολέας από το δίκτυο. Θα περιγράψουμε στη συνέχεια τι ψευδή μηνύματα μπορεί να μεταδώσει ο εισβολέας βασισμένος στις πληροφορίες που υπέκλεψε.

Έχουμε 15 είδη μεταβάσεων που δηλώνουν τα ψευδή μηνύματα του εισβολέα, τα οποία δηλώνονται με 15 τελεστές μετάβασης σε CafeOBJ. Η αποτελεσματική συνθήκη κάθε μετάβασης είναι ότι η απαραίτητη πληροφορία είναι διαθέσιμη στον εισβολέα.

Δύο από τους τελεστές μετάβασης για τα ψευδή ServerFinished μηνύματα, δηλώνονται ως εξής:

```
bop fakeSfin1 : Protocol Prin Prin EncSFin      -> Protocol
bop fakeSfin2 : Protocol Prin Prin Sid ListOfChoices
                           Choice Rand Rand Pms -> Protocol
```

Οι 15 τελεστές μετάβασης ορίζονται με εξισώσεις. Οι παρακάτω εξισώσεις δηλώνουν τις εξισώσεις για το fakeSfin2:

```
op c-fakeSfin2 : Protocol Prin Prin Sid ListOfChoices
                          Choice Rand Rand Pms -> Bool
eq c-fakeSfin2(P,B,A,I,L,C,R1,R2,PMS)
   = PMS \in cpms(nw(P)) .
```

```
ceq nw(fakeSfin2(P,B,A,I,L,C,R1,R2,PMS))
    = sf(intruder,B,A,esfin(k(B,PMS,R1,R2),
         sfin(A,B,I,L,C,R1,R2,PMS))) , nw(P)
    if c-fakeSfin2(P,B,A,I,L,C,R1,R2,PMS) .
eq ss(fakeSfin2(P,B,A,I,L,C,R1,R2,PMS),B2,A2,I2)
   = ss(P,B2,A2,I2) .
eq ur(fakeSfin2(P,B,A,I,L,C,R1,R2,PMS)) = ur(P) .
eq ui(fakeSfin2(P,B,A,I,L,C,R1,R2,PMS)) = ui(P) .
eq us(fakeSfin2(P,B,A,I,L,C,R1,R2,PMS)) = us(P) .
ceq fakeSfin2(P,B,A,I,L,C,R1,R2,PMS)
    = P if not c-fakeSfin2(P,B,A,I,L,C,R1,R2,PMS) .
```

Οι υπόλοιποι τελεστές μετάβασης ορίζονται αντιστοίχως.

4. ΑΝΑΛΥΣΗ

(α) Επαληθευμένες ιδιότητες

Ας δώσουμε πρώτα τις ανεπίσημες περιγραφές των επαληθευμένων ιδιοτήτων του πρωτοκόλλου.

- Τα προ-κύρια μυστικά δε μπορούν να διαρεύσουν.
- Αν ένας αξιόπιστος πελάτης λάβει ένα ServerFinished μήνυμα που ανταποκρίνεται στο πρωτόκολλο και φαίνεται να έχει σταλεί από τον εξυπηρετητή, τότε πράγματι προέρχεται από τον εξυπηρετητή.
- Αν ένας αξιόπιστος πελάτης λάβει ένα ServerFinished2 μήνυμα που ανταποκρίνεται στο πρωτόκολλο και φαίνεται να έχει σταλεί από τον εξυπηρετητή, τότε πράγματι προέρχεται από τον εξυπηρετητή.
- Αν ένας αξιόπιστος πελάτης λάβει ένα ServerHello μήνυμα, ένα Certificate μήνυμα και ένα ServerFinished μήνυμα που ανταποκρίνονται στο πρωτόκολλο και φαίνεται να έχουν σταλεί από έναν εξυπηρετητή, τότε τα ServerHello και Certificate μηνύματα, πράγματι προέρχονται από τον εξυπηρετητή.
- Αν ένας αξιόπιστος πελάτης λάβει ένα ServerHello2 μήνυμα και ένα ServerFinished2 μήνυμα που ανταποκρίνονται στο πρωτόκολλο και φαίνεται να έχουν σταλεί από έναν εξυπηρετητή, τότε το ServerHello2 μήνυμα, πράγματι προέρχεται από τον εξυπηρετητή.

Τα προ-κύρια μυστικά είναι οι πιο θεμελιώδεις παράμετροι στο πρωτόκολλο γιατί όλες οι παράμετροι ασφαλείας υπολογίζονται βασιζόμενοι σ'αυτά. Η πρώτη ιδιότητα εξασφαλίζει στους χρήστες του πρωτοκόλλου ότι οι παράμετροι ασφαλείας είναι πράγματι μυστικές. Η δεύτερη ιδιότητα εγγυάται ότι όταν ένας πελάτης διαπραγματεύεται μια ακολουθία cipher και τις παραμέτρους ασφαλείας με τον αντίστοιχο εξυπηρετητή μέσω μιας ολοκληρωμένης διαδικασίας χειραψίας, ο εξυπηρετητής έχει πράγματι συμφωνήσει μ'αυτά. Η τρίτη ιδιότητα εγγυάται ότι όταν ένας πελάτης διαπραγματεύεται μια ακολουθία cipher και τις παραμέτρους ασφαλείας με τον αντίστοιχο εξυπηρετητή μέσω μιας σύντομης

διαδικασίας χειραψίας, ο εξυπηρετητής έχει πράγματι συμφωνήσει μ'αυτά. Η τέταρτη ιδιότητα εγγυάται ότι όταν ένας πελάτης διαπραγματεύεται μια ακολουθία cipher και τις παραμέτρους ασφαλείας με τον αντίστοιχο εξυπηρετητή μέσω μιας ολοκληρωμένης διαδικασίας χειραψίας, η ακολουθία cipher και κάποιες από τις παραμέτρους ασφαλείας έχουν προταθεί από τον εξυπηρετητή. Η πέμπτη ιδιότητα εγγυάται ότι όταν ένας πελάτης διαπραγματεύεται μια ακολουθία cipher και τις παραμέτρους ασφαλείας με τον αντίστοιχο εξυπηρετητή μέσω μιας σύντομης διαδικασίας χειραψίας, η ακολουθία cipher και κάποιες από τις παραμέτρους ασφαλείας έχουν προταθεί από τον εξυπηρετητή.

Οι ιδιότητες έχουν μοντελοποιηθεί ως όροι σε CafeOBJ ώστε να επαληθευθούν επισήμως. Η πρώτη ιδιότητα δηλώνεται σε CafeOBJ ως εξής:

```
op inv1 : Protocol Pms -> Bool
eq inv1(P,PMS) = (PMS \in cpms(nw(P)) implies
   (client(PMS) = intruder or server(PMS) = intruder)) .
```

Ο όρος λέει ότι αν ένα προ-κύριο μυστικό είναι διαθέσιμο στον εισβολέα, τότε έχει παραχθεί από τον εισβολέα ή το έχει παράξει ο πελάτης κατά τη διάρκεια μιας συνόδου με τον εισβολέα. Αυτό υπαγορεύει ότι ο εισβολέας δε μπορεί να αποκτήσει κανένα προ-κύριο μυστικό αν δεν έχει εμπλακεί στη σύνοδο.

Η δεύτερη ιδιότητα δηλώνεται σε CafeOBJ ωε εξής:

```
op inv2 : Protocol Prin Prin Prin Rand Rand
             ListOfChoices Choice Sid Secret -> Bool
eq inv2(P,A,B,B1,R1,R2,L,C,I,S)
   = (not(A = intruder) and
      sf(B1,B,A,esfin(k(B,pms(A,B,S),R1,R2),
         sfin(A,B,I,L,C,R1,R2,pms(A,B,S)))) \in nw(P)
      implies
      sf(B,B,A,esfin(k(B,pms(A,B,S),R1,R2),
         sfin(A,B,I,L,C,R1,R2,pms(A,B,S)))) \in nw(P)) .
```

Ο όρος λέει ότι αν ένας αξιόπιστος πελάτης- δηλαδή ο πελάτης δεν είναι εισβολέας -λάβει ένα Server-Finished μήνυμα που φαίνεται να έχει σταλεί από έναν εξυπηρετητή- δηλαδή ο πραγματικός αποστολέας B1 μπορεί να είναι ο εισβολέας - που ανταποκρίνεται στο πρωτόκολλο και χρησιμοποιεί ένα προ-κύριο μυστικό παραγόμενο από τον πελάτη για τον εξυπηρετητή, τότε το μήνυμα πράγματι προέρχεται από τον εξυπηρετητή.

Χρειαζόμαστε ακόμα 13 ιδιότητες για να αποδείξουμε τις 5 κύριες ιδιότητες. Πέντε από τις ιδιότητες, συμπεριλαμβανομένων και της τέταρτης και πέμπτης, έχουν αποδειχθεί με ανάλυση κατά περίπτωση με άλλες ιδότητες και οι υπόλοιπες αποδεκνύονται με επαγωγή στον αριθμό των μεταβάσεων που εφαρμόζονται.

(β) Επαλήθευση

Περιγράφουμε το Proof Score για το inv2. Δηλώνουμε τον τελεστή που υποδηλώνει το βασικό τύπο για να αποδείξουμε σε κάθε επαγωγική περίπτωση ως εξής:

```
op istep2 : Prin Prin Prin Rand Rand ListOfChoices
                                Choice Sid Secret -> Bool
eq istep2(A,B,B1,R1,R2,L,C,I,S)
   = inv2(p,A,B,B1,R1,R2,L,C,I,S)
   implies inv2(p',A,B,B1,R1,R2,L,C,I,S) .
```

Τα p, p' είναι σταθερές hidden sort Protocol. Το p υποδηλώνει μια αυθαίρετη κατάσταση και το p' είναι η επόμενη κατάσταση της p.

Τα sfin1,sfin2,sfin3 έστω ότι είναι οι ακόλουθοι τύποι αντιστοίχως :

```
sf(intruder,b10,a10,esfin(k(b10,pms10,r10,r20),
            sfin(a10,b10,i10,l10,c10,r10,r20,pms10)))
sf(b1,b,a,esfin(k(b,pms(a,b,s),r1,r2),
          sfin(a,b,i,l,c,r1,r2,pms(a,b,s))))
sf(b,b,a,esfin(k(b,pms(a,b,s),r1,r2),
         sfin(a,b,i,l,c,r1,r2,pms(a,b,s))))
```

όπου όλες οι σταθερές εκτός από την intruder δηλώνουν αυθαίρετες τιμές των αντίστοιχων sorts. Για παράδειγμα, τα r1,r2,r10,r20 δηλώνουν αυθαίρετες σταθερές της visible sort Rand.

Ας θεωρήσουμε την επαγωγική περίπτωση στην οποία ο τελεστής μετάβασης fakeSfin2 διατηρεί το inv2. Όταν η αποτελεσματική συνθήκη του fakeSfin2 είναι true, η περίπτωση διαιρείται στις παρακάτω 5 υποπεριπτώσεις:

1. (sfin1≠sfin2)∧(sfin1≠sfin3)
2. (sfin1≠sfin2) ∧(sfin1= sfin3)
3. (sfin1= sfin2) ∧ (b= intruder)
4. (sfin1= sfin2) ∧ (b≠ intruder) ∧ (a=intruder)
5. (sfin1= sfin2) ∧ (b≠ intruder) ∧ (a≠intruder)

Οι πρώτες 4 υποπεριπτώσεις δε χρειάζονται κανένα επιπλέον κατηγόρημα για να ενισχύσει την επαγωγική υπόθεσή τους αλλά η πέμπτη υποπερίπτωση χρειάζεται το inv1 για να ενισχύσει την επαγωγική του υπόθεση.

Δείχνουμε το αποδεικτικό μονοπάτι που αντιστοιχεί στην πέμπτη υποπερίπτωση:

```
open ISTEP
-- arbitrary objects
 ops a10 b10 : -> Prin .       op i10 : -> Sid .
 op l10 : -> ListOfChoices .   op c10 : -> Choice .
 ops r10 r20 : -> Rand .       op pms10 : -> Pms .
-- assumptions
 eq pms(a,b,s) \in cpms(nw(p)) = true .
 --
 eq b1 = intruder . eq r10 = r1 . eq i10 = i .
 eq l10 = l . eq c10 = c . eq pms10 = pms(a,b,s) .
 eq r20 = r2 . eq a10 = a . eq b10 = b .
 --
```

```
 eq (b = intruder) = false . eq (a = intruder) = false .
-- successor state
 eq p' = fakeSfin2(p,b10,a10,
                   i10,l10,c10,r10,r20,pms10) .
-- check if the predicate is true.
 red inv1(p,pms(a,b,s))
     implies istep2(a,b,b1,r1,r2,l,c,i,s) .
close
```

Σταθερές όπως οι r1,r2 οι οποίες δε δηλώνονται σ'αυτή την απόδειξη, δηλώνονται σε ένα module, έστω INV, που εισάγεται από το ISTEP. Η πρώτη εξίσωση σημαίνει ότι η αποτελεσματική συνθήκη είναι true και η δεύτερη, διαμέσου της δέκατης, σημαίνει ότι το sfin1 είναι ίσο με το sfin2. Η ισότητα sfin1=sfin2 μπορεί να αναχθεί από τις 9 εξισώσεις με αναγραφή αλλά οι 9 εξισώσεις δε μπορούν να αναχθούν από τη μία με αναγραφή. Γι'αυτό χρησιμοποιούμε τις 9 εξισώσεις. Σ'αυτό το αποδεικτικό πέρασμα, το inv1(p,pms(a,b,s)) χρησιμοποιείται για να ενισχύσει την επαγωγική υπόθεση inv2(p,a,b,b1,r1,r2,l,c,i,s).

(γ) Συμπεράσματα

Στο κεφάλαιο αυτό έγινε μια προσπάθεια να μοντελοποιηθούν και να επαληθευθούν κάποιες ιδιότητες του πρωτοκόλλου χειραψίας του SSL χρησιμοποιώντας την CafeOBJ/ ΠΣΜ Method και τη διαδικασία των Proof Scores.

Σύμφωνα με τη σύντομη σχηματική απεικόνιση του πρωτοκόλλου χειραψίας όπως παρουσιάζεται στην αρχή του κεφαλαίου 5, επαληθεύσαμε ότι ένα ServerFinished2 μήνυμα προχωρά σε ένα ClientFinished2 μήνυμα. Δηλαδή ότι μια σύνοδος ανταλλαγής μηνυμάτων μεταξύ server και client, τερματίζεται μόνο αφού τα μηνύματα έχουν ανταλαχθεί ασφαλώς και αφού και οι δύο πλευρές έχουν ενημερωθεί για την επιτυχή ανταλλαγή μηνυμάτων. Επίσης επαληθεύσαμε ότι οι πέντε ιδιότητες που περιγράφηκαν στην προηγούμενη παράγραφο ισχύουν στο πρωτόκολλο όπου ένα ClientFinished2 μήνυμα προχωρά σε ένα ServerFinished2 μήνυμα.

Δεν επεκταθήκαμε στην απόδειξη όλων των ιδιοτήτων λόγω του ότι τα Proof Scores της μεθόδου CafeOBJ/ΠΣΜ έχουν την ευελιξία να προσαρμόζονται εύκολα σε κάθε μικρή αλλαγή των προδιαγραφών και ο σκοπός της έρευνας επετεύχθη με τα όσα μελετήθηκαν μέχρι στιγμής.

Μια πρόταση για περαιτέρω μελέτη είναι να πιστοποιηθούν και άλλα υπο-πρωτόκολλα του SSL, όπως το Alert protocol.

ΚΕΦΑΛΑΙΟ 6
ΕΦΑΡΜΟΓΗ 2: ΕΝΑ ΓΕΝΙΚΟ ΜΟΝΤΕΛΟ e-government

Εκτός από τη μοντελοποιήση του πρωτοκόλλου χειραψίας του SSL που έγινε στο προηγούμενο κεφάλαιο, θα παρουσίαζε ενδιαφέρον να μοντελοποιήσουμε σε CafeOBJ/ΠΣΜ και ένα απλό γενικό μοντέλο e-government όπως η **δημόσια διοίκηση.** Η χρήση της γλώσσας αλγεβρικών προδιαγραφών για τον σχεδιασμό ενός τομέα, τον καθιστά πιο περίπλοκο αλλά μειώνει τις ελλείψεις και τα σφάλματα.

1. Ο ΤΟΜΕΑΣ ΔΗΜΟΣΙΑΣ ΔΙΟΙΚΗΣΗΣ

Με τον όρο **Δημόσια Διοίκηση** εννοούμε όλες τις δραστηριότητες κυβερνητικού χαρακτήρα. Δηλαδή, δραστηριότητες νομοθετικές, φορολογικές, εθνικής άμυνας, δημόσιας τάξης και ασφάλειας, υπηρεσίες μετανάστευσης, εξωτερικών υποθέσεων, διεθνούς βοήθειας και διαχείρησης κυβερνητικών προγραμμάτων. Η **Δημόσια Διοίκηση** αποτελεί το σύνολο μέσων και ενεργειών που αποσκοπούν στην επίτευξη συγκεκριμένου αποτελέσματος για την ικανοποίηση του γενικού συμφέροντος των πολιτών ενός κράτους. Η Δημόσια Διοίκηση μιας Χώρας αποτελεί μέρος του κρατικού μηχανισμού της και ειδικότερα της εκτελεστικής εξουσίας. Υπό την ευρεία έννοια η Δημοσια Διοίκηση περιλαμβάνει δύο επιμέρους έννοιες, αυτή της εκάστοτε Κυβέρνησης και "υπό στενή έννοια" αυτή της Διοίκησης. Σημειώνεται ότι υπό τη στενή έννοια των πολιτικών αυτών όρων η Κυβέρνηση διαφέρει της Διοίκησης στο ότι κινείται με απόλυτη (νόμιμη) πρωτοβουλία και είναι αυτή που χαράσσει τις κατευθυντήριες γραμμές, ενώ η δεύτερη περιορίζεται στην εφαρμογή και παρακολούθηση της κυβερνητικής θέλησης. Δημόσια Διοίκηση υφίσταται σε όλες τις Χώρες του κόσμου, ανεξάρτητα του πολιτεύματός των.

Θα μελετήσουμε τον σχεδιασμό της δημόσιας διοίκησης ως ένα σύνολο οντοτήτων (entities), συναρτήσεων-λειτουργιών (functions), γεγονότων (events) και συμπεριφορών (behaviors).

Ως **οντότητα** αντιλαμβανόμαστε κάτι που ακόμα κι αν υποστεί αλλαγές, δε μεταβάλλεται η υπόστασή του και παραμένει το ίδιο. Οντότητες στον φυσικό κόσμο του τομέα δημόσιας διοίκησης είναι οι Κοινοβουλευτικές επιτροπές που κατασκευάζουν τους νόμους, τα δικαστήρια που τους εφαρμόζουν και οι πολίτες που συναλλάσσονται με τους παραπάνω.

Ως **συνάρτηση** θεωρούμε μια μαθηματική λειτουργία, η οποία αν εφαρμοστεί σε ένα όρισμα, παράγει ένα αποτέλεσμα. Μπορούμε να εντοπίσουμε τις εξής συναρτήσεις στον τομέα δημόσιας διοίκησης :

- Έγγραφα λειτουργίας: Τα έγγραφα δημιουργούνται, επεξεργάζονται, διαβάζονται, αντιγράφονται και μπορούν να αναζητηθούν από τους φορείς.

  Operation_Doc: Agent x Action x Document → Document

- Έγγραφα έγκρισης: Οι φορείς μπορούν να τα επεξεργαστούν, αναγνώσουν, αντιγράψουν ποικιλοτρόπως και να τα διανέμουν μεταξύ τους.

  Authorisation_Doc: Agent x Action x Agent x Document → Document

Τα ορίσματα στις παραπάνω συναρτήσεις είναι τα:

- Agent: όλες οι οντότητες που περιγράφηκαν παραπάνω.
- Action: δημιουργία, επεξεργασία, ανάγνωση, αντιγραφή, αναζήτηση, διανομή
- Document: έγγραφο νόμου, φόρμες συμπλήρωσης, αιτήσεις, παράπονα, απαντήσεις.

Ως **συμπεριφορά** αντιλαμβανόμαστε μια σειρά ενεργειών. Για παράδειγμα, μια συμπεριφορά είναι η ακολουθία δράσεων για την εφαρμογή του νόμου ή τη θέσπισή του.

Ως **γεγονός** θεωρούμε κάτι που ενεργοποιεί ή ενεργοποιείται από μια ενέργεια και πιθανά αλλάζει τη συμπεριφορά. Πιθανά γεγονότα είναι τα εξής: «Ο νόμος (δεν) πέρασε», «Πολίτης παρέβει το νόμο», «Πολίτης εφάρμοσε μια παρεχόμενη δημόσια υπηρεσία».

2. ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΤΟΥ ΤΟΜΕΑ ΤΟΠΙΚΗΣ ΑΥΤΟΔΙΟΙΚΗΣΗΣ ΣΕ CafeOBJ

Οντότητες

Ταξινομήσαμε προηγουμένως τις οντότητες σε τρία είδη: τους κατασκευαστές νόμων **(LawMaker)**, τους εφαρμογείς νόμων **(LawImplement)** και τους πολίτες **(Citizen)**.Οι παραπάνω θα καλούνται φορείς**(Actor)** και σε κάθε έναν φορέα οφείλουμε να αποδώσουμε κάποια χαρακτηριστικά και ιδιότητες όπως: το όνομά του **(ActorName)**, την κατάστασή του **(ActorStatus)**, τον τύπο του **(ActorType)**, τις δραστηριότητές του **(OperationSet)** και το για ποιες δράσεις είναι εξουσιοδοτημένος **(Authorization-Set)**. Η απεικόνιση των παραπάνω σε CafeOBJ είναι η εξής:

```
mod! ACTOR {
     pr(EQL + NAME + ACTORSTATUS + ACTORTYPE + AUTHORIZATIONSET + OP-
     ERATIONSET )
     [Actor]
     op mk-actor : ActorName ActorStatus ActorType OperationSet Au-
     thorizationSet -> Actor
     op actorname : Actor -> ActorName
     op actorstatus : Actor -> ActorStatus
     op actortype : Actor -> ActorType
     op operationrecord : Actor -> OperationSet
     op authorizationset : Actor -> AuthorizationSet


     --

     var AN : ActorName
     var AS : ActorStatus
```

```
     var AT : ActorType
     var OS : OperationSet
     var AUS : AuthorizationSet

     --

     eq actorname(mk-actor(AN,AS,AT,OS,AUS )) = AN .
     eq actorstatus(mk-actor(AN,AS,AT,OS,AUS)) = AS .
     eq actortype(mk-actor(AN,AS,AT,OS,AUS))= AT .
     eq operationrecord(mk-actor(AN,AS,AT,OS,AUS))=OS.
     eq authorizationset(mk-actor(AN,AS,AT,OS,AUS))=AUS.

     }


mod! ACTORTYPE {
     pr(EQL)
     [ActorType]
     ops lawmaker lawimplement citizen : -> ActorType
     eq (lawmaker = lawimplement) = false.
     eq (lawmaker = citizen) = false.
     eq (lawimplement = citizen) = false.
     }
```

Συναρτήσεις

Επίσης, στην προηγούμενη παράγραφο ορίσαμε δύο συναρτήσεις: Operation_Doc και Authorisation_Doc. Τα ορίσματα των δύο συναρτήσεων είναι τα Agent, Document και Action. Η μοντελοποίηση του Agent είναι η μοντελοποίηση των φορέων (Actor) που περιγράφηκε παραπάνω. Ας μοντελοποιήσουμε λοιπόν και τα Document και Action.

Στο Document αποδίδουμε τα χαρακτηριστικά DocName(όνομα του εγγράφου), ActorName(όνομα του φορέα του εγγράφου), DocStatus(η κατάσταση του εγγράφου), DocType(ο τύπος του εγγράφου), Info(περιεχόμενο του εγγράφου), OperationList(η λειτουργία που έχει εφαρμοστεί στο έγγραφο) και AuthRecord(τα δικαιώματα μεταβίβασης του εγγράφου μεταξύ των φορέων).

Το Action έχει τις ιδιότητες Create(δημιουργία), Edit(επεξεργασία), Read(ανάγνωση), Copy(αντιγραφή), Search(αναζήτηση) και Distribute(διανομή).

Η απεικόνιση σε CafeOBJ είναι η εξής:

```
mod! DOCUMENT {
     pr(EQL + DOCUMENTSTATUS + NAME + DOCUMENTTYPE +
     INFORMATION + AUTHORIZATIONLIST + OPERATIONLIST)
     [Document]
     op mk-document : DocName ActorName DocStatus DocType
                      Info OperationList AuthRecord
                                          -> Document
```

```
     op _=_ : Document Document -> Bool
     --
     op docname : Document -> DocName
     op doccreator : Document -> ActorName
     op docstate : Document -> DocStatus
     op doctype : Document -> DocType
     op docinfo : Document -> Info
     op doctrace : Document -> OperationList
     op authrecord : Document -> AuthRecord
     }


mod! OPERATION {
     pr(EQL)
     [Operation]
     ops create edit read copy search distribute:
                                              -> Operation
     }
```

Με βάση τα παραπάνω, μπορούμε να προχωρήσουμε στον ορισμό των συναρτήσεων. Στόχος μας είναι να καταγράψουμε ποιος εφάρμοσε τη δράση και πώς αυτή επιδρά στο έγγραφο. Η δομή δεδομένων για τη δημιουργία αυτού του ζεύγους πληροφοριών είναι η εξής:

```
-- used for describing document trace
-- and operation record

mod! PAIR(M :: TRIV, N :: TRIV) {
     pr(EQL)
     [Pair]

     op [_,_] : Elt.M Elt.N -> Pair
     op left : Pair -> Elt.M
     op right : Pair -> Elt.N

     var P : Pair
     vars X1 X2 : Elt.M
     vars Y1 Y2 : Elt.N

     eq left([X1, Y1]) = X1 .
     eq right([X1, Y1]) = Y1 .
     eq ([X1, Y1] = [X2, Y2]) = (X1 = X2) and (Y1 = Y2) .
     }

-- used for describing actor authorization
-- and document authorization record
```

```
mod! TPAIR( M :: TRIV, N :: TRIV, O :: TRIV ) {
     pr(EQL)
     [TPair]
     op < _, _, _ > : Elt.M Elt.N Elt.O -> TPair
     op hed : TPair -> Elt.M
     op mid : TPair -> Elt.N
     op lst : TPair -> Elt.O

     var P : TPair
     vars X1 X2 : Elt.M
     vars Y1 Y2 : Elt.N
     vars Z1 Z2 : Elt.O

     eq hed( < X1, Y1, Z1 > ) = X1 .
     eq mid( < X1, Y1, Z1 > ) = Y1 .
     eq lst( < X1, Y1, Z1 > ) = Z1 .
     eq ( < X1, Y1, Z1 > = < X2, Y2, Z2 > )
     = ( X1 = X2 ) and ( Y1 = Y2 ) and ( Z1 = Z2 ) .
     }
```

Βασιζόμενοι στους παραπάνω τύπους δεδομένων, μπορούμε να περιγράψουμε το σύστημα δημόσιας διοίκησης και όλες τις λειτουργίες που πραγματοποιούνται σ'αυτό. Μπορούμε να το θεωρήσουμε ως Παρατηρήσιμο Σύστημα Μετάβασης. Το μοντελοποιούμε ως μια hidden sort PA. Υπάρχουν δύο παρατηρητές για την κατάσταση των οντοτήτων Actor και Document και οι συναρτήσεις Operation_Doc και Authorization_Doc αποτελούν τις συναρτήσεις μετάβασης του ΠΣΜ. Το σύστημα περιγράφεται σε CafeOBJ ως εξής:

```
mod* PA {
     pr (ACTOR + DOCUMENT + OPERATION)
     pr (EQL)
     *[System]*
-- any initial state
     op init : -> System
-- observations in PA System
     bop actor : System ActorName -> Actor
     bop document : System DocName -> Document
-- actions
     bop authorize : System ActorName ActorName
     DocName Operation -> System
     bop operate : System ActorName DocName
     Operation -> System
     }
```

Γεγονότα

Στα παραπάνω, η init αποτελεί την αρχική κατάσταση του συστήματος. Υποθέτουμε ότι στην αρχική κατάσταση του συστήματος, ένας αυθαίρετος φορέας, έστω A, έχει κατάσταση exiting, τύπο citizen και τα Operation και AuthRecord είναι κενά. Ένα αυθαίρετο έγγραφο, έστω D, στην αρχική του κατάσταση είναι νόμος(lawdoc), έχει περιεχόμενο law και προτείνεται από μια κοινοβουλευτική επιτροπή(committee).

```
-- initial state
eq actor(init,A)=mk-actor(A,exiting,citizen,em-os,em-au).
eq document(init, D)=mk-document(D,committee,idled,lawdoc,
                                          law,nil-os,nil-au).
```

Η εφαρμογή των λειτουργιών(συναρτήσεις μετάβασεις) στην αρχική κατάσταση αλλάζει την κατάσταση του συστήματος και τις τιμές των παρατηρητών. Ως γεγονός, λαμβάνουμε τις αποτελεσματικές συνθήκες των συναρτήσεων. Οι αποτελεσματικές συνθήκες προσδιορίζονται με τις παρακάτω εξισώσεις:

```
-- effective condition of authorize
   op c-authorize: System ActorName ActorName DocName Operation
                                                        -> Bool
   eq c-authorize(S,A1,A2,D,O)
                =actorstatus(actor(S,A1))=exiting
                and actorstatus(actor(S,A2))=exiting
                and ((A1=doccreator(document(S,D)))
                or(<D,O,A1>/in authorizationset(actor(S,A1)))).
-- effective condition of operate
   op c-operate : System ActorName DocName Operation -> Bool
   eq c-operate(S,A,D,O)
              =<D,O,A>/in authorizationset(actor(S,A))
              and (docstate(document(S,D))=idled)or(O=creat).
```

Συμπεριφορά

Μια ακολουθία εφαρμογών των συναρτήσεων μετάβασης στο σύστημα PA αποτελεί τη συμπεριφορά του συστήματος.

```
open PA
ops applicant officer: -> ActorName.
op application: -> DocName.
op s: -> System.
  eq s=operate (authorize (authorize (operate (init ,applicant ,
               application,create),applicant, officer,
               application,read),applicant, officer,application,
```

```
                edit),officer, application, edit).
close
```

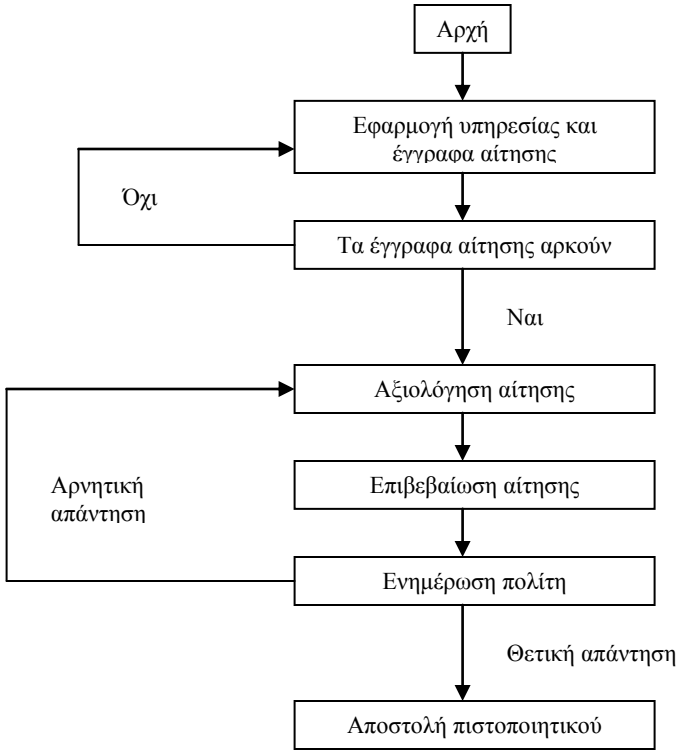
Ο παραπάνω κώδικας με την εντολή open, ανοίγει το module PA. Υποθέτουμε ότι υπάρχουν δύο φορείς, έστω applicant και officer, και ένα έγγραφο application. Δοσμένης μιας αυθαίρετης κατάστασης s, ξεκινά μια εφαρμογή υπηρεσίας. Αλλεπάλληλες εφαρμογές των συναρτήσεων μετάβασης περιγράφουν μια συμπεριφορά.

3.Παράδειγμα: ΑΙΤΗΣΗ ΠΟΛΙΤΗ ΓΙΑ ΑΠΟΚΤΗΣΗ ΠΙΣΤΟΠΟΙΗΤΙΚΟΥ ΠΟΙΝΙΚΟΥ ΜΗΤΡΩΟΥ

Ένα σύστημα e-government αποσκοπεί κυρίως στην παροχή υπηρεσιών στους πολίτες. Το μοντέλο που θα αναλυθεί παρακάτω, εμπλέκει μια Εισαγγελική Αρχή και έναν πολίτη, οι οποίοι συναλλάσσονται και αλληλεπιδρούν μέσω διαδικτύου ώστε να αποκτήσει ο δεύτερος(citizen) το πιστοποιητικό ποινικού μητρώου(document) από τον πρώτο(lawenforcer). Οι δράσεις που πραγματοποιούνται μέχρι να τελειώσει η συναλλαγή είναι οι εξής:

1. Εφαρμογή της δημόσιας υπηρεσίας: ο πολίτης αιτείται την απόκτηση του εγγράφου.
2. Αξιολόγηση της εφαρμογής: η Εισαγγελική Αρχή λαμβάνει την αίτηση του πολίτη και ζητά-αν χρειάζεται- συμπληρωματικά έγγραφα.
3. Επιβεβαίωση της αξιολόγησης: η Εισαγγελική Αρχή επιβεβαιώνει ότι έχει όλα τα απαραίτητα έγγραφα και σε περίπτωση που δεν τα έχει, επιστρέφει στη δράση 2.
4. Κοινοποίηση της απόφασης: η Εισαγγελική Αρχή ενημερώνει τον πολίτη για το αποτέλεσμα της αίτησής του.
5. Εκτέλεση απόφασης: η Εισαγγελική Αρχή αποστέλλει το έγγραφο στον πολίτη.

Σχηματικά, τα παραπάνω απεικονίζονται ως εξής:



Οι βασικοί τύποι δεδομένων που χρησιμοποιούνται για τη δημιουργία CafeOBJ/ΠΣΜ μοντέλου για την παραπάνω συναλλαγή είναι :

Για το έγγραφο:

DocName: όνομα εγγράφου που αποτελεί το αναγνωριστικό του

DocType: τύπος του εγγράφου που μπορεί να είναι αίτηση του πολίτη, έντυπο απόφασης, συμπληρωματικά έγγραφα που ζητήθηκαν

Actor: ο δημιουργός του εγγράφου

Info: περιεχόμενο του εγγράφου

DTrace: ζεύγος φορέα-εγγράφου για λόγους ιστορικού

Doc: έγγραφο στο οποίο βασιζόμαστε για την παραγωγή του τελικού εγγράφου

Dlabel: κατάσταση του εγγράφου που μπορεί να είναι applied, evaluated, confirmed, notified, accepted, unsatisfied, executed

DocAuth: λίστα που υποδεικνύει ποιος έχει την αρμοδιότητα να κάνει τι

Ο κατασκευαστής **mk-doc(N:Dname, T:Dtype, DC:Actor, DI:Info, DT:Dtrace, R:Doc, DL:Dlabel, DA:DocAuth)**

Για τον φορέα (Εισαγγελική Αρχή ή πολίτης) **:**

ActorName: όνομα που αποτελεί το αναγνωριστικό του φορέα

ActorStatus: πιθανές καταστάσεις στο φορέα handling/idling

DocNameSet: ποια έγγραφα έχει το δικαίωμα να εξετάζει κάθε φορέας

DocTrace: ο φορέας μπορεί να παρακολουθεί την πορεία του εγγράφου του

Ο κατασκευαστής **mk-actor(N:Aname, AS:Astatus, T:Atype, AP:Opset, AO:Opset)** κατασκευάζει τον φορέα.

Επίσης, πρέπει να κατασκευάσουμε έξι τελεστές (οι οποίοι έχουν ως ορίσματα τα DocName, Actor) : **create, edit, read, copy, shred, authorization.** Χρησιμοποιούμε τα κατηγορήματα **iscreate, isedit, isread, iscopy, isshred, isauth** για να ελέγξουμε την απόδοση του κάθε τελεστή.

Δεδομένων των παραπάνω τελεστών, ορίζονται και οι τελεστές προβολής:

docofo: επιστρέφει το έγγραφο στο οποίο γίνεται η δράση

actofo: επιστρέφει τον φορέα που πραγματοποιεί η δράση

infoofo και **editofo:** επιστρέφουν την πληροφορία που δημιουργείται ή επεξεργάζεται αντιστοίχως κατά τη δράση

timeofo: επιστρέφει το πόσες φορές μπορεί να εφαρμοστεί η δράση στο έγγραφο

typeofo: επιστρέφει τα είδη εξουσιοδότησης για τη δράση

attiofo: επιστρέφει αν ο φορέας συμφωνεί ή διαφωνεί με το έγγραφο

Τώρα μπορούμε να δημιουργήσουμε το ΠΣΜ μοντέλο PA μέσα σ'ένα module, το οποίο κατ'αρχάς εισάγει όλους τους παραπάνω τύπους δεδομένων.

Οι **παρατηρητές** actor και doc στο ΠΣΜ δηλώνονται ως εξής:

```
bop actor : Sys ActorName -> Actor
bop doc : Sys DocName -> Doc
```

Στη συνέχεια αρχικοποιείται το σύστημα:

```
-- [0] equations of initial state
   eq actor(init,C)
      = mk-actor(C,idling,basicLawSet,ndt).
   eq doc(init,D)
      = (if D /in basicLawSet
         then basicLaw else emptyDoc fi).
```

Έπειτα, δημιουργούνται οι συναρτήσεις μετάβασης και οι αποτελεσματικές συνθήκες:

```
bop apply : Sys ActorName DocName Info -> Sys
bop evaluate: Sys ActorName DocName DocNameSet
                                          Info -> Sys
bop confirm : Sys ActorName DocName Info -> Sys
```

```
bop notify : Sys ActorName DocName Result-> Sys
bop execute : Sys ActorName DocName -> Sys
bop track : Sys ActorName DocName -> Sys


-- [3] evaluate
  op c-evaluate :Sys ActorName DocName Info -> Bool
  eq c-evaluate(S,C,D,I)
     = C /in agenciesSet and
       not (C = docCreator(doc(S,D))) and
       docType(doc(S,D)) = pdoc and
       docLabel(doc(S,D)) = submited and
      (auth(docCreator(doc(S,D)),evaluate,agenciesSet)
      /in docAuth(doc(S,D))).
```

Τελικά, εφαρμόζουμε αναγωγή για να «ιχνηλατήσουμε» την πορεία του εγγράφου.

```
open PA
 op a : -> ActorName .
 op d : -> DocName .
 op s : -> Sys .
 red (docCreator(doc(s,d)) = a) implies c-track(s,a,d) .
 red docLabel(basicLaw) = issused
 and c-  track(s,a,docName(basicLaw)).
close
```

Αν από την τελική αναγωγή το αποτέλεσμα είναι True, τότε ο πολίτης λαμβάνει το πιστοποιητικό ποινικού μητρώου.

Μετά την ανάλυση ροής για το παραπάνω μοντέλο, η οποία περιγράφει και επεξηγεί τα σημαντικότερα σημεία, παραθέτουμε ολόκληρο τον κώδικα σε CafeOBJ:

```
mod* PA {
  pr (ACTOR + DOCUMENT + AUTHSET + OPSET + ACTORROLESET + RESULT)
  pr (EQL)
 *[Sys]*

-- initial state
  op init : -> Sys

-- observer
  bop actor : Sys ActorName -> Actor
  bop doc : Sys DocName -> Doc



-- actions
```

```
  bop apply : Sys ActorName DocName Info -> Sys
  bop evaluate : Sys ActorName DocName DocNameSet Info -> Sys
  bop confirm : Sys ActorName DocName Info -> Sys
  bop notify : Sys ActorName DocName Result -> Sys
  bop execute : Sys ActorName DocName -> Sys
  bop track : Sys ActorName DocName -> Sys

-- variable
  var S : Sys
  vars C C1 C2 : ActorName
  vars D D1 D2 : DocName
  vars DS DS1 DS2 : DocNameSet
  var I : Info
  var R : Result

-- added operation
  op sumInfo : DocNameSet -> Info

--
  eq sumInfo(add(D,emptyDocNameSet)) = docInfo(doc(S,D)) .
  eq sumInfo(add(D,DS)) = (docInfo(doc(S,D)) @ sumInfo(DS)) .

--> *** added end
-- [0] equations of initial state
  eq actor(init,C)
     = mk-actor(C,idling,basicLawSet,issused) .
  eq doc(init,D) = (if D /in basicLawSet then basicLaw else         emptyDoc
fi).

-- [1] apply
  op c-apply : Sys ActorName DocName Info -> Bool
  eq c-apply(S,C,D,I) = doc(S,D) = emptyDoc and C /in       applicantSet and
not (D /in basicLawSet) .

--
 ceq actor(apply(S,C,D,I),C1)
     = (if not(C1 = C) then actor(S,C1)
        else          mk-                                                ac-
tor(C,handling,actorKnow(actor(S,C)),actorTrack(actor(S,C)))     fi)if     c-
apply(S,C,D,I) .
  ceq doc(apply(S,C,D,I),D1)
      = (if not(D1 = D)
         then doc(S,D1)
         else mk-doc(D,C,operating,pdoc,I,
                     add([C,apply],ndt),
                     docAuth(doc(S,D)),
                     actorKnow(actor(S,C)),applied) fi)
         if c-apply(S,C,D,I) .
  ceq apply(S,C,D,I) = S if not c-apply(S,C,D,I) .

-- [2] evaluate
```

```
  op c-evaluate : Sys ActorName DocName DocNameSet Info -> Bool
  eq c-evaluate(S,C,D,DS,I)
     = C /in agenciesSet and not (C = docCreator(doc(S,D))) and
     docType(doc(S,D)) = pdoc and docLabel(doc(S,D)) = applied
     and  (auth(docCreator(doc(S,D)),evaluate,agenciesSet)  /in           do-
cAuth(doc(S,D))) .

--
  ceq actor(evaluate(S,C,D,DS,I),C1)
      = (if not (C1 = C) then actor(S,C1)
         else   mk-actor(C,   handling,   actorKnow(actor(S,C)),          ac-
torTrack(actor(S,C))) fi)
if c-evaluate(S,C,D,DS,I) .
  ceq doc(evaluate(S,C,D,DS,I),D1)
      = (if not(D1 = D) then doc(S,D1)
         else                                                             mk-
doc(D,docCreator(doc(S,D)),docStatus(doc(S,D)),docType(doc(S,D)),
(I @ docInfo(doc(S,D))),add([C,evaluate],docTrace(doc(S,D))),
add(auth(C,confirm,agenciesSet),docAuth(doc(S,D))),
DS U docRefer(doc(S,D)),evaluated) fi)
if c-evaluate(S,C,D,DS,I) .
  ceq evaluate(S,C,D,DS,I) = S if not c-evaluate(S,C,D,DS,I) .
-- [3] confirm
  op c-confirm : Sys ActorName DocName Info -> Bool
  eq c-confirm(S,C,D,I)
     = C /in agenciesSet and
     not      (C      =      docCreator(doc(S,D)))      and      not(C      =
left(getTerm(evaluate,docTrace(doc(S,D))))) and
docType(doc(S,D)) = pdoc and docLabel(doc(S,D)) = evaluated and
(auth(left(getTerm(evaluate,docTrace(doc(S,D)))),confirm,agenciesSet)    /in
docAuth(doc(S,D))) .

--
  ceq actor(confirm(S,C,D,I),C1)
      = (if not(C1 = C) then actor(S,C1)
         else    mk-actor(C,handling,actorKnow(actor(S,C)),               ac-
torTrack(actor(S,C))) fi)
if c-confirm(S,C,D,I) .
  ceq doc(confirm(S,C,D,I),D1)
       = (if not(D1 = D)
          then doc(S,D1)
          else                                                            mk-
doc(D,docCreator(doc(S,D)),docStatus(doc(S,D)),docType(doc(S,D)),
(I @ docInfo(doc(S,D))),add([C,confirm],docTrace(doc(S,D))),
del(auth(left(getTerm(evaluate,docTrace(doc(S,D)))),evaluate,agenciesSet),
(if I = noenough then docAuth(doc(S,D))
else add(auth(C,notify,agenciesSet),docAuth(doc(S,D))) fi)),
(actorKnow(actor(S,C)) U docRefer(doc(S,D))),
(if I = noenough then applied else confirmed fi)) fi)
if c-confirm(S,C,D,I) .
  ceq confirm(S,C,D,I) = S if not c-confirm(S,C,D,I) .
```

```
-- [4] notify
  op c-notify : Sys ActorName DocName Result -> Bool
  eq c-notify(S,C,D,R)
     = C /in agenciesSet and not(C = docCreator(doc(S,D))) and
not(C = left(getTerm(confirm,docTrace(doc(S,D))))) and
docLabel(doc(S,D)) = confirmed and
auth(left(getTerm(confirm,docTrace(doc(S,D)))),notify,agenciesSet)  /in  do-
cAuth(doc(S,D)) .

--
  ceq actor(notify(S,C,D,R),C1)
      = (if not(C1 = C) then (if C1 = docCreator(doc(S,D))
         then                                                          mk-
actor(C1,idling,add(D,actorKnow(actor(S,C1))),actorTrack(actor(S,C1)))
else actor(S,C1) fi)
else mk-actor(C,handling,actorKnow(actor(S,C)), actorTrack(actor(S,C))) fi)
if c-notify(S,C,D,R) .
  ceq doc(notify(S,C,D,R),D1)
      = (if not(D1 = D)
         then doc(S,D1)
else mk-doc(D,docCreator(doc(S,D)),docStatus(doc(S,D)),ddoc,
(docInfo(doc(S,D))),add([C,confirm],docTrace(doc(S,D))),
del(auth(left(getTerm(confirm,docTrace(doc(S,D)))),notify,agenciesSet),
(if R = unpass then docAuth(doc(S,D))
else add(auth(C,execute,agenciesSet),docAuth(doc(S,D))) fi)),
docRefer(doc(S,D)),notified) fi)
if c-notify(S,C,D,R) .
  ceq notify(S,C,D,R) = S if not c-notify(S,C,D,R) .

-- [5] execute
  op c-execute : Sys ActorName DocName -> Bool
  eq c-execute(S,C,D)
     = C /in agenciesSet and not (C = docCreator(doc(S,D))) and
not(C = left(getTerm(confirm,docTrace(doc(S,D))))) and
docType(doc(S,D)) = ddoc and docLabel(doc(S,D)) = notified and
auth(left(getTerm(confirm,docTrace(doc(S,D)))),execute,agenciesSet)  /in  do-
cAuth(doc(S,D)) .

--
  ceq actor(execute(S,C,D),C)
      = (if not(C1 = C) then actor(S,C)
         else     mk-actor(C,handling,actorKnow(actor(S,C)),              ac-
torTrack(actor(S,C))) fi)
if c-execute(S,C,D) .
  ceq doc(execute(S,C,D),D1)
      = (if not(D1 = D)
          then doc(S,D)
else mk-doc(D,docCreator(doc(S,D)),exiting,docType(doc(S,D)),
docInfo(doc(S,D)),add([C,execute],docTrace(doc(S,D))),
del(auth(left(getTerm(notify,docTrace(doc(S,D)))),execute,agenciesSet),
```

```
docAuth(doc(S,D))), docRefer(doc(S,D)),executed) fi)
if c-execute(S,C,D) .
  ceq execute(S,C,D) = S if not c-execute(S,C,D) .

-- [6] track
  op c-track : Sys ActorName DocName -> Bool
  eq c-track(S,C,D)
     = ((C = docCreator(doc(S,D))) and not (doc(S,D) = emptyDoc)) or (D /in
basicLawSet) .

--
  ceq actor(track(S,C,D),C1)
      = (if not(C1 = C) then actor(S,C1)
         else      mk-actor(C1,handling,actorKnow(actor(S,C1)),      docLa-
bel(doc(S,D))) fi)
if c-track(S,C,D) .
  ceq doc(track(S,C,D),D1)
      = (if not (D = D1) then doc(S,D1)
         else mk-doc(D1, C, docStatus(doc(S,D1)),docType(doc(S,D1)),
docInfo(doc(S,D1)),add([C,track], docTrace(doc(S,D1))),docAuth(doc(S,D1)),
docRefer(doc(S,D1)), docLabel(doc(S,D1))) fi)
if c-track(S,C,D) .
  ceq track(S,C,D) = S if not c-track(S,C,D) .
}

-- 1. trivial property of transparency: creator of the document can
track his/her application any time

open PA
 op a : -> ActorName .
 op d : -> DocName .
 op s : -> Sys .
 red (docCreator(doc(s,d)) = a) implies c-track(s,a,d) .
 red docLabel(basicLaw) = issused and c-track(s,a,docName(basicLaw)) .
close
```

4.ΣΥΜΠΕΡΑΣΜΑΤΑ

Στο πέμπτο κεφάλαιο είδαμε ότι οι αλγεβρικές γλώσσες προδιαγραφών, και συγκεκριμένα η CafeOBJ που χρησιμοποιήθηκε, μας βοηθούν να πιστοποιήσουμε την ασφάλεια ενός συστήματος. Στο παρόν κεφάλαιο διαπιστώσαμε πως οι τυπικές προδιαγραφές μπορούν να συμβάλλουν στην αναπτυξιακή διαδικασία των συστημάτων που απαιτούν σχεδιασμό υψηλού επιπέδου. Επίσης, χρησιμοποιώντας τυπικές προδιαγραφές, μπορούμε να πραγματοποιήσουμε εφαρμογές της λογικής. Για παράδειγμα, δοσμένης μιας ακολουθίας δράσεων, μπορούμε να διερευνήσουμε αν μια δράση είναι δυνατόν να εφαρμοστεί στην τρέχουσα κατάσταση του συστήματος. Η CafeOBJ επιστρέφει μια τιμή Boolean (true/false) ανάλογα με το αν επιτρέπεται να συνεχιστεί η δράση ή όχι.

Συμπερασματικά, από τα πρακτικά κομμάτια αυτής της διπλωματικής εργασίας, είναι προφανές ότι οι αλγεβρικές γλώσσες προδιαγραφών είναι σκόπιμο να χρησιμοποιούνται πριν τον προγραμματισμό κατά τον σχεδιασμό σημαντικών εφαρμογών για να αποφεύγονται τα λογικά σφάλματα, να ελαχιστοποιούνται οι πιθανές διαρροές πληροφοριών και να διασφαλίζεται η διαφάνεια των συναλλαγών.

ΒΙΒΛΙΟΓΡΑΦΙΑ-ΠΗΓΕΣ

1. Mary Maureen Brown. "Electronic Government". Encyclopedia of Public Administration and Public Policy, Marcel Dekker
2. Shailendra C. Jain Palvia and Sushil S. Sharma. "E-Government and E-Governance: Definitions/Domain Framework and Status around the World"
3. Atkinson, Robert D.; Castro, Daniel , Digital Quality of Life, The Information Technology and Innovation Foundation
4. Becker, Shirley "Bridging Literacy, Language, and Cultural Divides to Promote Universal Usability of E-GovernmentWebsites".Northern Arizona University.
5. " e-government structure for e-protocol,e-application submission and internal organizational and operational support", A.S Drigas,L.Koukianakis,S.Domoxoudis
6. wikipedia
7. Andrew S. Tanenbaum, "Computer Networks"
8. W. Richard Stevens, "The Protocols"
9. The SSL Protocol: Version 3.0 Netscape's final SSL 3.0
10. "e-Government in Europe". European Commission.
11. "Europe 2020: A strategy for smart, sustainable and inclusive growth". European Commission, Information Society.
12. Study on the E-government Security Risk Management, Zhitian Zhou, Congyang Hu
13. Internet Security Architectures and Protocols , Laboratory of Cryptography and Systems Security (CrySyS), Department of Telecommunications, Budapest University of Technology and Economics
14. http://en.wikipedia.org/wiki/Cryptographic_protocol
15. Network Security protocols, Radia Perlman
16. Analyzing Internet Security Protocols Alec Yasinsac, Justin Childs
17. Challenges in e-government and security of information Min-Shiang Hwang,Chun-Ta Li,Jau-Ji Shen,Yen-Ping Chu
18. Secure Sockets Layer,John Michael Pierobon
19. R. Diaconescu and K. Futatsugi. CafeOBJ report.
20. D. Dolev and A. C. Yao. On the security of public key protocols
21. J. Hsiang and N. Dershowitz. Rewrite methods for clausal and nonclausal theorem proving.
22. K. Ogata and K. Futatsugi. Proof scores in the OTS/CafeOBJ method
23. Towards Transparent E-Government Systems, a View from Formal Methods, Xiaoyi CHEN
24. Αλγεβρικές προδιαγραφές αλγορίθμων κωδικοποίησης βίντεο : MPEG-2, Ξύστρα Ελ. Κατερίνα