ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ

ΣΧΟΛΗ ΟΙΚΟΝΟΜΙΑΣ, ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Σχεδίαση και ανάπτυξη βιβλιοθήκης λογισμικού για τη διαχείριση δομών πληροφορίας μορφότυπων περιγραφής συμβάντων ασφάλειας

Αθανασίου Αριστείδης

Α.Μ. 2024200800021

Επιβλέποντες: Τσελίκας Νικόλαος, Επίκουρος Καθηγητής ΠΑ.ΠΕΛ.

Λιουδάκης Γεώργιος, Ερευνητικός Συνεργάτης Ε.Μ.Π. (Λέκτορας βάσει Π.Δ. 407/80 για τα ακ. έτη 2008-2009, 2009-2010, 2010-2011)

ΤΡΙΠΟΛΗ

ΜΑΡΤΙΟΣ 2014

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά την οικογένειά μου που με στηρίζει όλα αυτά τα χρόνια ψυχολογικά και οικονομικά ώστε να πραγματοποιήσω τις σπουδές μου. Επίσης, ευχαριστώ ειλικρινά τους επιβλέποντες της πτυχιακής κ. Γιώργο Λιουδάκη και κ. Νικόλαο Τσελίκα, οι οποίοι μου έδωσαν την ευκαιρία να ασχοληθώ με ένα θέμα το οποίο με ενδιέφερε ιδιαιτέρως και χωρίς την πολύτιμη βοήθεια των οποίων, δεν θα είχε ολοκληρωθεί αυτή η πτυχιακή εργασία. Τέλος, ευχαριστώ τις Μαρίζα Κουκοβίνη και Ευγενία Παπαγιαννακοπούλου για την πολύτιμη βοήθειά τους κατά την εκπόνηση της πτυχιακής.

Περίληψη

Η διάδοση και ανταλλαγή με έναν ενιαίο τρόπο πληροφοριών αναφορικά με συμβάντα που αφορούν στην ασφάλεια (security incidents) αποκτά ολοένα και μεγαλύτερη σημασία στα κατανεμημένα περιβάλλοντα, δεδομένης της ετερογένειας των συστημάτων και της ανάγκης για συνεργατικότητα στην αντιμετώπιση των εκάστοτε απειλών. Σε αυτή την κατεύθυνση η Τακτική Δύναμη Μηχανικών Διαδικτύου (Internet Engineering Task Force IETF) έχει προτείνει δύο πρωτόκολλα βασισμένα σε XML, τα οποία αποτελούν σημεία αναφοράς: το IDMEF (Intrusion Detection Message Exchange Format) και το IODEF (Incident Object Description Exchange Format). Το πρώτο, για το οποίο έχουν ήδη αναπτυχθεί αρκετές βιβλιοθήκες, ορίζει μορφότυπους αναπαράστασης δεδομένων και διαδικασίες ανταλλαγής πληροφοριών που αφορούν σε επιθέσεις (alerts) μεταξύ συστημάτων ανίχνευσης εισβολών (intrusion detection systems), συστημάτων απόκρισης (response systems) και λοιπών συστημάτων διαχείρισης (management systems). Το IODEF ορίζει έναν μορφότυπο για την περιγραφή, αποθήκευση και ανταλλαγή πληροφορίας μεταξύ των Security Incident Response Teams (CSIRTS). Στο πλαίσιο της διπλωματικής εργασίας πραγματοποιήθηκε μελέτη των συγκεκριμένων μορφότυπων και πρωτοκόλλων, ενώ σχεδιάστηκε και αναπτύχθηκε βιβλιοθήκη λογισμικού, η οποία αναλαμβάνει την αποθήκευση συμβάντων ασφαλείας τα οποία ακολουθούν τον μορφότυπο του IDMEF στην οντολογία που αναπτύχθηκε για τις ανάγκες της πτυχιακής εργασίας. Πέραν της αποθήκευσης των συμβάντων ασφαλείας η βιβλιοθήκη παρέχει πρόσθετες λειτουργικότητες, όπως τη διαχείριση των αναλυτών του συστήματος και την εξαγωγή γνώσης από την οντολογία. Η ανάπτυξη της βιβλιοθήκης λογισμικού πραγματοποιήθηκε με χρήση της γλώσσας προγραμματισμού Java ενώ η οντολογία αναπτύχθηκε σε OWL DL.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ : Ασφάλεια υπολογιστικών συστημάτων και δικτύων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ : IDMEF, IODEF, XML, OWL, Java, Οντολογίες

Abstract

Considering that alert information is inherently heterogeneous as well as the need of collaboration between different systems in order to effectively confront potential threats, the exchange of information regarding security incidents over a distributed system has become extremely important over the past years. Internet Engineering Task Force has proposed two protocols based on XML that are considered a touchstone; IDMEF (Intrusion Detection Message Exchange Format) and IODEF (Incident Object Description Exchange Format). IDMEF is intended to be a standard data format that can be used by automated intrusion detection systems, response systems as well as management systems. IODEF is a format for describing, archiving and exchanging security information commonly sent between Security Incident Response Teams (CSIRTs). In the context of this thesis, a software system has been developed that parses alerts expressed in Intrusion Detection Message Exchange Format (IDMEF) and appropriately fills the underlying ontology with useful information contained therein. Apart from that, the system provides extra functionalities for efficient management of analyzers as well as for knowledge extraction. The implementation is a Java software library, while the ontology was created in OWL DL.

SUBJECT AREA: Computer and network security

KEYWORDS : IDMEF, IODEF, XML, OWL, Java, Ontologies

Πίνακας Περιεχομένων

Λίστα Σχημάτων

Διάρθρωση της Πτυχιακής Εργασίας

Η παρούσα εργασία περιλαμβάνει 5 κεφάλαια.

Στο Κεφάλαιο 1 γίνεται μία σύντομη περιγραφή των δύο προτύπων που ορίζει η IETF για την αναπαράσταση δεδομένων και διαδικασιών ανταλλαγής πληροφοριών που αφορούν σε επιθέσεις μεταξύ συστημάτων ανίχνευσης εισβολών, του IDMEF και του IODEF.

Στο Κεφάλαιο 2 ορίζεται η έννοια του Σημασιολογικού Ιστού και παρουσιάζονται οι κυριότερες τεχνολογίες που τον απαρτίζουν. Ιδιαίτερη έμφαση δίνεται στις γλώσσες XML, RDF, RDF-S και OWL καθώς αυτές κυρίως θα μας απασχολήσουν στα πλαίσια της εργασίας.

Στο Κεφάλαιο 3 παρουσιάζεται η οντολογία που αναπτύχθηκε με σκοπό την αποθήκευση συμβάντων ασφαλείας που ακολουθούν τον μορφότυπο του IDMEF. Επίσης, παρέχεται αναλυτική περιγραφή των σχετικών οντολογικών κλάσεων και ιδιοτήτων.

Στο Κεφάλαιο 4 περιγράφεται η υλοποίηση του συστήματος λογισμικού που αναπτύχθηκε στα πλαίσια της πτυχιακής εργασίας. Συγκεκριμένα, περιγράφονται η ανάλυση του συστήματος, ο σχεδιασμός και τέλος η υλοποίηση του.

Τέλος, στο Κεφάλαιο 5 παρουσιάζονται τα τελικά συμπεράσματα καθώς και πιθανές μελλοντικές κατευθύνσεις βελτιστοποίησης του συστήματος.

1 Μορφότυποι Περιγραφής Συμβάντων Ασφάλειας

1.1 Το IDMEF

Το IDMEF (Intrusion Detection Message Exchange Format) αποτελεί πρότυπο της IETF[1] (Τακτική Δύναμη Μηχανικών Διαδικτύου, Internet Engineering Task Force) (RFC:4765, H. Debar, D. Curry, B. Feinstein) και αναπτύχθηκε με σκοπό τη δημιουργία ενός συγκεκριμένου μορφότυπου, το οποίο θα μπορούν να χρησιμοποιήσουν τα συστήματα ανίχνευσης εισβολών (intrusion detection systems) για να αναφέρουν συμβάντα ασφαλείας (security incidents). Η χρήση του προαναφερθέντος μορφότυπου επιτρέπει τη διαλειτουργικότητα (interoperability) μεταξύ διαφορετικών συστημάτων ανίχνευσης εισβολής επιτρέποντας με αυτό τον τρόπο στους χρήστες την εγκατάσταση διαφορετικών συστημάτων σύμφωνα με τις ανάγκες και επιτυγχάνοντας τη βέλτιστη υλοποίηση.

Η υλοποίηση του IDMEF λαμβάνει χώρα στο κανάλι επικοινωνίας μεταξύ των ανιχνευτών/αναλυτών (sensors/analyzers) και του κεντρικού συστήματος στο οποίο αναφέρουν τα συμβάντα ασφαλείας. Εκτός από τη διαλειτουργικότητα που προαναφέρθηκε, το IDMEF επεκτείνει τις δυνατότητες του ευρύτερου συστήματος ασφαλείας. Ενδεικτικά:

- Η ύπαρξη μίας οικουμενικής βάσης δεδομένων στην οποία θα αποθηκεύονται τα συμβάντα ασφαλείας που αναφέρονται απ' όλα τα συστήματα ανίχνευσης εισβολών, καθιστά την ανάλυση των δεδομένων με σκοπό την εξαγωγή πρόσθετης γνώσης ευκολότερη και αποτελεσματικότερη, αφού πραγματοποιείται σε έναν πολύ μεγαλύτερο όγκο δεδομένων. Είναι γνωστό από τη βιβλιογραφία πως, όσο μεγαλύτερος είναι ο όγκος των δεδομένων που αναλύονται, τόσο πιο αξιόπιστα και αποτελεσματικά μπορεί να γίνει η εξαγωγή γνώσης απ' αυτά.
- Ένα σύστημα συσχέτισης γεγονότων (event correlator), το οποίο δέχεται δεδομένα από μία πλειάδα διαφορετικών συστημάτων ανίχνευσης εισβολών, έχει τη δυνατότητα να πραγματοποιήσει πιο αξιόπιστη και εξελιγμένη διασταύρωση γεγονότων και πληροφορίας σε σχέση με ένα σύστημα το οποίο δέχεται ως είσοδο δεδομένα από ένα και μοναδικό σύστημα ανίχνευσης εισβολής.
- Η ύπαρξη ενός γραφικού περιβάλλοντος (graphical user interface), το οποίο έχει τη δυνατότητα να παρουσιάζει στο χρήστη συμβάντα ασφαλείας προερχόμενα από πολλά διαφορετικά συστήματα ανίχνευσης εισβολών, δίνει τη δυνατότητα στον διαχειριστή να παρακολουθεί ταυτόχρονα όλα τα συστήματα σε μία και μόνο οθόνη. Με αυτόν τον τρόπο, η διαχείριση του δικτύου καθίσταται πολύ πιο εύκολη, καθώς δεν απαιτείται από το διαχειριστή η εξοικείωσή του με πλήθος διεπαφών παρά μόνο με μία.

[1] http://www.ietf.org

- Η ύπαρξη ενός συγκεκριμένου μορφότυπου ανταλλαγής δεδομένων καθιστά πολύ απλούστερη την επικοινωνία καθώς και την ανταλλαγή δεδομένων μεταξύ των διαφορετικών οργανισμών.
- Ο μορφότυπος δεδομένων του IDMEF αποτελεί μία αντικειμενοστραφή αναπαράσταση των συμβάντων ασφαλείας που δημιουργούνται και στέλνονται από τους αναλυτές (sensors-analyzers), στο κεντρικό σύστημα διαχείρισης στο οποίο αναφέρουν (manager).

Η χρήση του μορφότυπου του IDMEF δίνει λύσεις σε σοβαρά προβλήματα που σχετίζονται με την αναπαράσταση συμβάντων ασφαλείας. Είναι γνωστό πως η πληροφορία σχετικά με τα συμβάντα ασφαλείας είναι εγγενώς ανομοιογενής [1] [2]. Ορισμένα συμβάντα ασφαλείας παρέχουν μόνο πολύ βασικές πληροφορίες όπως την προέλευση, το στόχο, ένα όνομα και την ώρα του συμβάντος. Επιπλέον, υπάρχουν συμβάντα ασφαλείας τα οποία περικλείουν πολύ περισσότερη και πιο εξειδικευμένη πληροφορία, όπως πόρτες (ports), υπηρεσίες, διεργασίες, πληροφορίες για τους χρήστες και ούτω καθεξής. Ο μορφότυπος δεδομένων του IDMEF είναι σχεδιασμένος ώστε να είναι αρκετά ευέλικτος και να μπορεί να καλύψει τις διαφορετικές ανάγκες του κάθε συστήματος. Το μοντέλο του IDMEF ως αντικειμενοστραφές, εξ' ορισμού είναι επεκτάσιμο μέσω δημιουργίας συνενώσεων και υποκλάσεων. Σε μία υλοποίηση του μορφότυπου όπου αυτό επεκτείνεται είτε μέσω συνενώσεων είτε μέσω υποκλάσεων, ακόμα και το αν στο σύστημα δεν είναι γνωστές οι επεκτάσεις που έχουν γίνει στο μοντέλο, θα μπορεί να καταλάβει τις πληροφορίες που περιέχουν αυτές οι επεκτάσεις. Με τον προαναφερθέντα τρόπο επέκτασης διατηρείται η επεκτασιμότητα καθώς και η συνεκτικότητα του αρχικού μοντέλου.

Τα περιβάλλοντα των συστημάτων ανίχνευσης εισβολών είναι διαφορετικά. Ορισμένοι αναλυτές ανιχνεύουν εισβολές πραγματοποιώντας ανάλυση της δικτυακής κίνησης, άλλοι μέσω επεξεργασίας των αρχείων καταγραφής (log files) του συστήματος και άλλοι βάσει εντοπισμού ίχνους (audit trail) [1]. Γίνεται λοιπόν αντιληπτό ότι ακόμα και συμβάντα ασφαλείας που αντιστοιχούν στην ίδια επίθεση θα περιέχουν διαφορετική πληροφορία αν οι αναλυτές που τα δημιούργησαν χρησιμοποιούν διαφορετικά περιβάλλοντα ανίχνευσης. Ο μορφότυπος του IDMEF δίνει λύση στο παραπάνω πρόβλημα καθώς υπάρχουν ειδικά διαμορφωμένες κλάσεις που εξυπηρετούν τις διαφορετικές ανάγκες των περιβαλλόντων τα οποία χρησιμοποιούνται από τα συστήματα ανίχνευσης. Πιο συγκεκριμένα, ο προσδιορισμός της πηγής και του στόχου της επίθεσης αναπαριστάται με έναν συνδυασμό πληροφοριών σχετικών με τούς κόμβους (nodes), τους χρήστες (users), τις διεργασίες (processes) και τις υπηρεσίες (services) που εμπλέκονται.

Οι δυνατότητες ανάλυσης των διαφόρων συστημάτων ανίχνευσης είναι σαφώς διαφορετικές. Ανάλογα με τις ανάγκες που υπάρχουν σε κάθε περιβάλλον, εγκαθίσταται και το κατάλληλο σύστημα ανίχνευσης. Έτσι, υπάρχουν συστήματα με χαμηλές απαιτήσεις σε πόρους (επεξεργαστική ισχύ, μνήμη) τα οποία παράγουν συμβάντα ασφαλείας με περιορισμένη πληροφορία, αλλά και σύνθετα συστήματα με υψηλές απαιτήσεις, τα οποία όμως παρέχουν περισσότερες και αναλυτικότερες πληροφορίες σχετικά με το συμβάν. Επίσης, συχνά υπάρχει η ανάγκη μετατροπής των συμβάντων ασφαλείας σε μορφή

κατανοητή από εργαλεία ανάλυσης και εξαγωγής γνώσης, ώστε να είναι δυνατή η περαιτέρω επεξεργασία τους. Το μοντέλο δεδομένων του IDMEF ορίζει τις κατάλληλες επεκτάσεις στο Document Type Definition (DTD) που επιτρέπουν την περιγραφή απλών, αλλά και πιο σύνθετων συμβάντων ασφαλείας. Η επέκταση του μοντέλου γίνεται με επέκταση ή συσχέτιση των υπαρχόντων κλάσεων.

Λόγω ύπαρξης διαφορετικών λειτουργικών συστημάτων, τυχόν επιθέσεις παρατηρούνται και συνεπώς, αναφέρονται με διαφορετικά χαρακτηριστικά. Στο IDMEF υπάρχουν οι κατάλληλες κλάσεις (Node και Service) οι οποίες παρέχουν την απαραίτητη ευελιξία στην αναφορά συμβάντων ασφαλείας από διαφορετικά λειτουργικά συστήματα. Αν υπάρξει ανάγκη για μοντελοποίηση ακόμα περισσότερης πληροφορίας, δύνανται να οριστούν οι κατάλληλες υποκλάσεις για την επέκταση του μοντέλου.

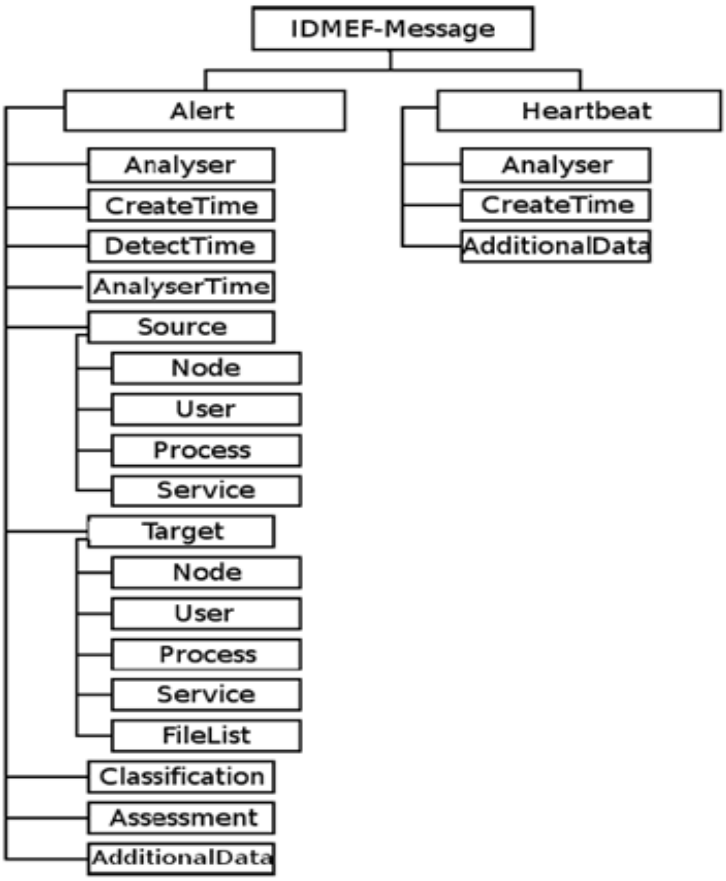
Τα εμπορικά συστήματα ανίχνευσης εισβολών ποικίλουν και μπορεί να έχουν σημαντικές διαφοροποιήσεις, τόσο όσον αφορά τα συμβάντα που μπορούν να ανιχνεύσουν όσο και στον τρόπο με τον οποίο τα αναφέρουν. Το αντικειμενοστραφές μοντέλο του IDMEF είναι αρκετά ευέλικτο ώστε να μπορεί να εξυπηρετεί τις ανάγκες όλων των συστημάτων ανίχνευσης, ενώ οι κανόνες που αφορούν τον τρόπο με τον οποίον γίνεται η επέκταση του μοντέλου διατηρούν την ακεραιότητα του.

Στο Σχήμα 1 βλέπουμε τη πληροφορία που μπορεί να φέρει ένα συμβάν ασφαλείας εκφρασμένο σε IDMEF.

Βλέπουμε την κεντρική κλάση IDMEF-Message η οποία αναπαριστά όλα τα IDMEF μηνύματα. Ένα μήνυμα μπορεί να είναι τύπου Alert ή Heartbeat.

Οι αναλυτές χρησιμοποιούν μηνύματα τύπου heartbeat για να ενημερώνουν περιοδικά τον διαχειριστή στον οποίον αναφέρουν για την κατάσταση τους. Τα μηνύματα στέλνονται σύγχρονα (δεν απαιτείται η ύπαρξη κάποιου συμβάντος για την αποστολή τους) ανά τακτά χρονικά διαστήματα. Οι αναλυτές θα πρέπει να στέλνουν περιοδικά heartbeat μηνύματα στον διαχειριστή, ώστε αυτός να γνωρίζει ότι είναι ενεργοί και λειτουργικοί. Η απουσία κάποιου heartbeat, σημαίνει ότι κάποιος αναλυτής έχει σταματήσει να λειτουργεί ή υπάρχει κάποια βλάβη στο δίκτυο. Σ' αυτήν την περίπτωση ο διαχειριστής θα πρέπει να πραγματοποιήσει τις απαραίτητες ενέργειες για την αποκατάσταση της βλάβης. Στα πλαίσια της εργασίας θα ασχοληθούμε μόνο με την κλάση Alert καθώς η κλάση Heartbeat δεν παρουσιάζει κάποιο ιδιαίτερο ερευνητικό ενδιαφέρον.

Όλες οι υπόλοιπες κλάσεις χρησιμοποιούνται για την αναπαράσταση πληροφοριών σχετικά με το συμβάν ασφάλειας. Οι σημαντικότερες κλάσεις είναι η `Analyzer`, η `Source` και η `Target`, στο Κεφάλαιο 3 παρέχεται μια αναλυτική περιγραφή σχετικά με την πληροφορία που αναπαριστούν οι εν λόγω κλάσεις καθώς και οι υπόλοιπες.



Σχήμα 1: Το Μοντέλο δεδομένων του IDMEF.

1.2 Το IODEF

Το άλλο πρότυπο της IETF σχετικά με τον ορισμό μορφότυπων αναπαράστασης συμβάντων ασφαλείας αποτελεί το IODEF (Incident Object Description and Exchange Format RFC: 5070, Danyliw, et al.). Το IODEF είναι ένα πρότυπο για την αναπαράσταση πληροφορίας σχετικής με την ασφάλεια υπολογιστικών συστημάτων η οποία ανταλλάσσεται μεταξύ διαφόρων ομάδων αντιμετώπισης περιστατικών ασφάλειας (Computer Security Incident Response Teams, CSIRTs). Παρέχει ένα XML Schema για την αναπαράσταση και μετάδοση συμβάντων ασφαλείας μεταξύ των αρχών οι οποίες είναι υπεύθυνες για την ασφάλεια ενός τομέα. Το μοντέλο δεδομένων του IODEF κωδικοποιεί πληροφορία σχετικά με:

- Εξυπηρετητές
- Δίκτυα
- Υπηρεσίες που τρέχουν σε αυτά τα συστήματα
- Μεθοδολογία επιθέσεων και στοιχεία με βάση τα οποία μπορούν να αναγνωριστούν
- Επίπτωση της επίθεσης στο σύστημα

- Περιορισμένη δυνατότητα περιγραφής μίας ροής εργασιών (workflow).

Σκοπός του IODEF είναι να ενισχύσει τις δυνατότητες των CSIRTs. Η υιοθέτησή του από την κοινότητα θα συμβάλλει στην αποτελεσματικότερη διευθέτηση συμβάντων ασφαλείας απλοποιώντας σημαντικά τη συνεργασία και την ανταλλαγή πληροφοριών μεταξύ των CSIRTs. Ο μορφότυπος που ορίζει το IODEF επιτρέπει:

- Αυτοματοποίηση της επεξεργασίας των δεδομένων σχετικά με τα συμβάντα ασφαλείας από τους αναλυτές καθώς τα δεδομένα θα ακολουθούν έναν συγκεκριμένο μορφότυπο και δεν θα είναι σε ελεύθερη μορφή (free-form textual documents).
- Ευκολότερη κανονικοποίηση παρόμοιων δεδομένων τα οποία προέρχονται από διαφορετικές πηγές και είναι πολύ πιθανό να ακολουθούν διαφορετικούς μορφότυπους.
- Έναν κοινό μορφότυπο με βάση το οποίο θα αναπτυχτούν διαλειτουργικά εργαλεία για τον χειρισμό περιστατικών και την εκτενέστερη ανάλυση δεδομένων τα οποία προέρχονται από διαφορετικές αρχές.

Ο συντονισμός μεταξύ διαφορετικών CSIRTs δεν αποτελεί αποκλειστικά ένα τεχνικό πρόβλημα. Υπάρχουν θέματα διαδικαστικά, εμπιστοσύνης και νομικών υποχρεώσεων, τα οποία μπορούν να αποτρέψουν έναν οργανισμό από το να διαμοιραστεί δεδομένα. Προφανώς και το IODEF δεν μπορεί να διευθετήσει τα παραπάνω προβλήματα, παρ' όλα αυτά παρέχει μία καλή λύση όσον αφορά το τεχνικό κομμάτι της επικοινωνίας μεταξύ των οργανισμών.

Η υλοποίηση του IODEF σε XML (όπως και αυτή του IDMEF) παρέχει ένα πλήθος πλεονεκτημάτων. Η επεκτασιμότητά του το κάνει ιδανικό για τον προσδιορισμό ενός πλαισίου εργασίας το οποίο θα υποστηρίζει διαφορετικές κωδικοποιήσεις χαρακτήρων. Ακόμα η ύπαρξη πολλών παρόμοιων τεχνολογιών (EXtensible Stylesheet Language XSL, XPath, XML-Signature) απλοποιεί την αξιοποίηση των δεδομένων. Κρίνεται σκόπιμο πως πρέπει να τονιστούν τα εξής:

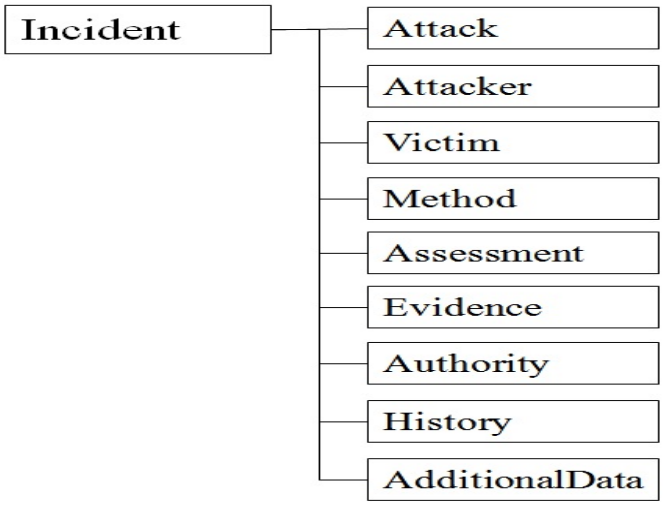
- Το μοντέλο δεδομένων του IODEF έχει δημιουργηθεί για τη μεταφορά πληροφορίας μεταξύ των οργανισμών και όχι για τη μακροχρόνια αποθήκευση και ταξινόμηση των δεδομένων.
- Εφόσον μέχρι στιγμής δεν υπάρχει κάποιος σαφής ορισμός ευρείας κοινής αποδοχής σχετικά με το τι είναι ένα συμβάν ασφαλείας, το IODEF δεν επιχειρεί να προσδιορίσει έναν, μέσω της υλοποίησης του.
- Η δυνατότητα περιγραφής ενός οποιουδήποτε περιστατικού ασφαλείας με βάση τα όσα υπάρχουν στη βιβλιογραφία θα έκανε το μοντέλο δεδομένων εξαιρετικά πολύπλοκο. Γι' αυτό το λόγο το IODEF αναπτύχθηκε έτσι ώστε να μπορεί να περιγράψει την πληροφορία που ανταλλάσσεται πιο συχνά μεταξύ των οργανισμών. Παρ' όλα αυτά το μοντέλο του διασφαλίζει ότι θα υπάρχουν οι απαραίτητοι μηχανισμοί για την επέκτασή του, έτσι ώστε να μπορεί να καλύψει τις εξειδικευμένες ανάγκες του κάθε οργανισμού, καθώς και τεχνικές για την αναφορά

συμβάντων των οποίων οι πληροφορίες δεν μπορούν να ενσωματωθούν στο προτεινόμενο μοντέλο.

- Ο τομέας της ασφάλειας δεν είναι απολύτως τυποποιημένος (standardised) - τουλάχιστον προς το παρόν - και έτσι συχνά η περιγραφή συμβάντων ασφαλείας βασίζεται στη χρήση κειμένου ελευθέρας μορφής. Το IODEF επιχειρεί να βρει τη χρυσή τομή μεταξύ της περιγραφής των συμβάντων σε ελεύθερη μορφή και της αυτοματοποιημένης επεξεργασίας τους.

Το IODEF είναι μέρος μίας μεγάλης προσπάθειας από την IETF για την τυποποίηση των δεδομένων που σχετίζονται με τον τομέα της ασφάλειας. Στα πλαίσια αυτής της προσπάθειας, έχουν αναπτυχθεί διάφορες τυποποιήσεις με στόχο να είναι συμβατές και συμπληρωματικές μεταξύ τους. Το ήδη υπάρχον μοντέλο δεδομένων του IDMEF επηρέασε ιδιαίτερα την ανάπτυξη του μοντέλου δεδομένων του IODEF. Στο Σχήμα 2 βλέπουμε τη πληροφορία που μπορεί να φέρει ένα συμβάν ασφαλείας εκφρασμένο σε IODEF.

Η κεντρική κλάση είναι η Incident η οποία αναπαριστά συμβάντα ασφάλειας και όλες οι υπόλοιπες κλάσεις χρησιμοποιούνται για τη μοντελοποίηση πληροφορίας που τα περιγράφει. Το πρότυπο επιτρέπει την αναλυτική περιγραφή πληροφορίας σχετικά με έναν συμβάν ασφαλείας μέσω των κλάσεων `Attack`, `Attacker`, `Victim`, `Method`, `Assessment`, `Evidence`, `Authority` και `History`. Οι τρείς πρώτες προαναφερθείσες κλάσεις, οι οποίες αναπαριστούν την επίθεση αυτή καθ' αυτή, τον επιτιθέμενο και το θύμα της, θεωρούνται και οι πλέον σημαντικές.

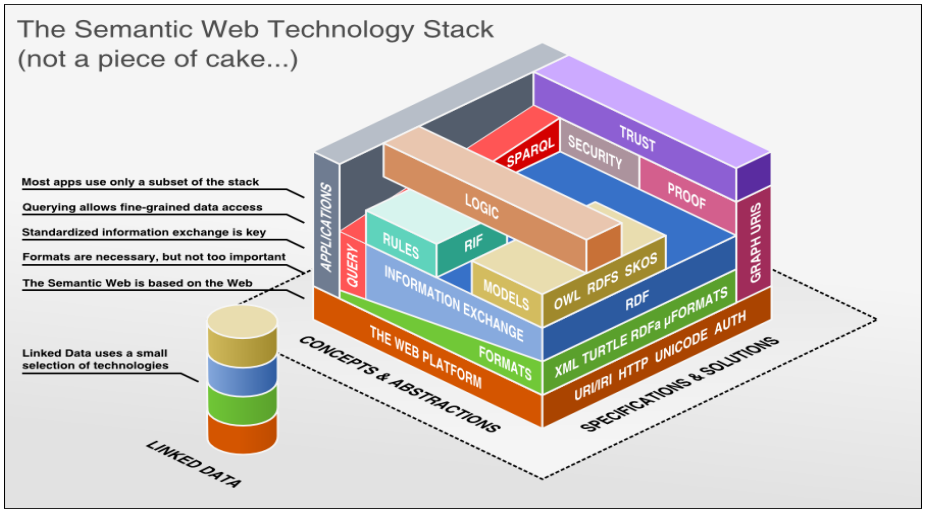


Σχήμα 2: Το μοντέλο δεδομένων του IODEF.

2 Ο Σημασιολογικός Ιστός

2.1 Εισαγωγή στο Σημασιολογικό Ιστό

Ο Σημασιολογικός Ιστός (Semantic Web) αποτελεί ουσιαστικά την εξέλιξη του Παγκόσμιου Ιστού (World Wide Web) και σχεδιάστηκε με σκοπό τη μοντελοποίηση των δεδομένων του Παγκόσμιου Ιστού ώστε να γίνονται αντιληπτά από μηχανές, καθώς και την αυτοματοποίηση λειτουργιών και εφαρμογών του διαδικτύου, όπως είναι οι μηχανές αναζήτησης και οι πράκτορες (agents). Ενώ ο Παγκόσμιος Ιστός είχε ως σκοπό την ανταλλαγή εγγράφων, ο Σημασιολογικός Ιστός στοχεύει στη δημιουργία μορφότυπων για την ενοποίηση και τον συνδυασμό δεδομένων που προέρχονται από διαφορετικές πηγές. Για την ανάπτυξη του Σημασιολογικού Ιστού δεν αρκεί απλά η δημοσίευση πληροφοριών στο Διαδίκτυο θα πρέπει ο τεράστιος όγκος πληροφορίας ο οποίος βρίσκεται δημοσιευμένος και αποθηκευμένος αυτή τη στιγμή στον Παγκόσμιο Ιστό να αποκτήσει τυπικό νόημα (formal meaning), σημασιολογία (semantics) και να δομηθεί με κατάλληλο τρόπο, έτσι ώστε να γίνει αντιληπτός από τις μηχανές που τον επεξεργάζονται (machine understandable). Στη συνέχεια, θα πρέπει να δημιουργηθούν κατάλληλοι πράκτορες (agents), οι οποίοι θα πραγματοποιούν επερωτήσεις και θα ενοποιήσουν αυτά τα δεδομένα. Από τη στιγμή που η πληροφορία θα είναι δομημένη με έναν σημασιολογικά πλούσιο τρόπο, αυτομάτως ενισχύεται ο διαμοιρασμός (sharing) και η επαναχρησιμοποίησή (reusability) της, επιτυγχάνοντας μ’ αυτόν τον τρόπο διαλειτουργικότητα (interoperability) και συνδεσιμότητα (interconnectivity) ετερογενών (heterogeneous) συστημάτων και εφαρμογών. Προκειμένου η γνώση και η πληροφορία να περιγραφεί με έναν τυπικό (formal) τρόπο, ο οποίος θα δηλώνει τη σημασία της, θα πρέπει να χρησιμοποιηθούν γλώσσες αναπαράστασης γνώσης. Η ανάπτυξη του Σημασιολογικού Ιστού γίνεται σε επίπεδα (layers), με το κάθε στρώμα να υλοποιεί μία λειτουργικότητα (functionality), χρησιμοποιώντας και επεκτείνοντας τη λειτουργικότητα και τις τεχνολογίες που παρέχονται από τα χαμηλότερα στρώματα.



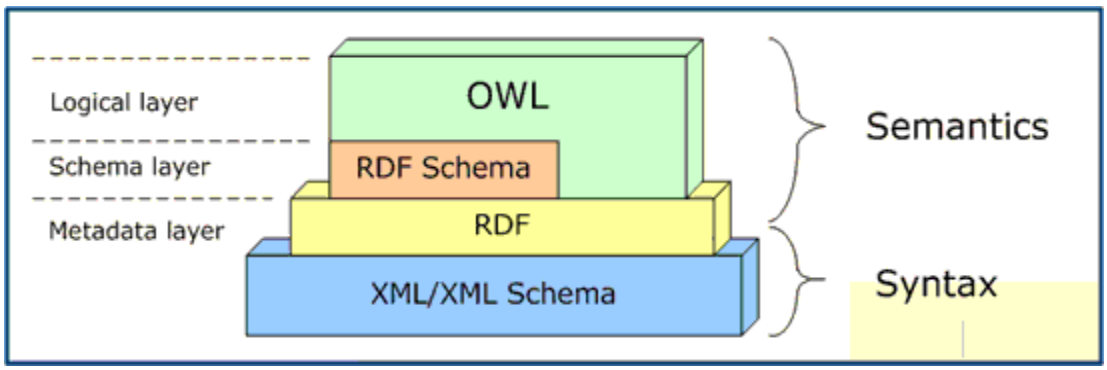
Σχήμα 3: Τριδιάστατη απεικόνιση του Layer Cake του Σημασιολογικού Ιστού

Στο Σχήμα 3 βλέπουμε μία τριδιάστατη αναπαράσταση του περίφημου “Semantic Web Layer Cake”. Έτσι, λοιπόν, στα χαμηλότερα επίπεδα υλοποιούνται λειτουργίες οι οποίες είναι πολύ κοντά στον Παγκόσμιο Ιστό και στις μηχανές, ενώ, καθώς ανεβαίνουμε στην ιεραρχία των επιπέδων, διατρέχουμε επίπεδα τα οποία υλοποιούν λειτουργικότητες αναπαράστασης γνώσης, πολύπλοκης συλλογιστικής και εμπιστοσύνης πλησιάζοντας στην ανθρώπινη γνώση και σκέψη. Το όραμα για τον Σημασιολογικό Ιστό είναι ότι κάποτε:

- Η διαδικτυακή πληροφορία θα είναι κατανοητή και θα μπορεί εύκολα να υποστεί επεξεργασία από υπολογιστικά συστήματα
- Τα υπολογιστικά συστήματα θα μπορούν να ενσωματώνουν με τρόπο αυτοματοποιημένο πληροφορία από τον Παγκόσμιο Ιστό.

2.2 Τα 4 Επίπεδα του Σημασιολογικού Ιστού

Όπως είδαμε ο Σημασιολογικός Ιστός χωρίζεται σε διάφορα επίπεδα· στα πλαίσια της εργασίας θα μας απασχολήσουν περισσότερο η XML (eXtensible Markup Language), με χρήση της οποίας γίνεται η αναπαράσταση των συμβάντων ασφαλείας στο IDMEF και στο IODEF καθώς και η OWL (Web Ontology Language), επειδή το λογισμικό που αναπτύχθηκε αποθηκεύει τα συμβάντα ασφαλείας σε μία OWL οντολογία. Στη συνέχεια της εργασίας περιγράφονται εκτενέστερα η XML και η OWL, καθώς και οι RDF και RDFS διότι όπως θα δούμε αποτελούν τη βάση της OWL. Στο Σχήμα 4 βλέπουμε τα τέσσερα προαναφερθέντα επίπεδα.



Σχήμα 4: Τα XML, RDF,RDFS και OWL επίπεδα του Σημασιολογικού Ιστού.

Επίπεδο 1. **Επίπεδο μεταδεδομένων (Metadata layer)**

Στο επίπεδο των μεταδεδομένων εισάγεται μία απλή μεν, πολύ βασική δε, γλώσσα αναπαράστασης γνώσης για τον Παγκόσμιο Ιστό η RDF (Resource Description Framework) (Lassila O., Swick R., 1999). Το μοντέλο της γλώσσας αυτής προσφέρει ουσιαστικά μόνο τη δυνατότητα δημιουργίας ισχυρισμών (assertions) για τα στοιχεία του διαδικτύου. Οι έννοιες που εισάγονται είναι αυτές του πόρου (resource) και της ιδιότητας (property), οι οποίες χρησιμοποιούνται για την περιγραφή των μετα-δεδομένων. Για παράδειγμα, μπορούμε να περιγράψουμε κάποιον πόρο αποδίδοντάς του μία ή περισσότερες ιδιότητες και προσδιορίζοντας τις τιμές που παίρνουν αυτές οι ιδιότητες.

Επίπεδο 2. **Επίπεδο σχήματος (Schema layer)**

Στο επίπεδο σχήματος εισάγονται κάποια επιπλέον βασικά στοιχεία για την περιγραφή γνώσης στο Σημασιολογικό Ιστό. Η γλώσσα η οποία υλοποιεί το επίπεδο αυτό είναι η γλώσσα RDF-S (RDF - Schema) (Brickey D., Guha R.V., 2000). Στην RDF-S εισάγονται για πρώτη φορά οι έννοιες της κλάσης (class) καθώς και της ιεραρχίας κλάσεων (class hierarchy) και ιδιοτήτων (properties) μέσω των super και sub properties. Για να οριστούν αυτές οι έννοιες χρησιμοποιείται η λειτουργικότητα που παρέχεται από το κατώτερο επίπεδο των μεταδεδομένων.

Επίπεδο 3. **Λογικό επίπεδο (Logical layer)**

Στο λογικό επίπεδο υλοποιούνται περισσότερο εκφραστικές γλώσσες αναπαράστασης γνώσης. Η γλώσσα η οποία υλοποιεί τη λειτουργικότητα του επιπέδου αυτού είναι η OWL (Bechhofer et al., 2004). Η OWL χρησιμοποιείται για την επέκταση του επιπέδου σχήματος, παρέχοντας περισσότερες εκφραστικές δυνατότητες και χωρίζεται σε τρεις υπο-γλώσσες με βάση με τη δύναμη εκφραστικότητας που προσφέρει η καθεμία.

Επίπεδο 4. **Επίπεδο κανόνων (Rules layer)**

Στο επίπεδο αυτό η λειτουργικότητα των γλωσσών του λογικού επιπέδου επεκτείνεται ακόμη περισσότερο παρέχοντας τη δυνατότητα καταγραφής κανόνων. Στη συνέχεια αναλύονται εκτενέστερα η σύνταξη και η σημασιολογία των προαναφερθεισών γλωσσών, με σκοπό την καλύτερη κατανόηση των δυνατοτήτων και των περιορισμών που έχει η κάθε γλώσσα.

2.3 Η XML (eXtensible Markup Language)

Η XML αποτελεί πρότυπο-σύσταση της W3C (World Wide Web Consortium) από τον Φεβρουάριο του 1998, δεν είναι μία γλώσσα προγραμματισμού, αλλά μία «γλώσσα αναπαράστασης και περιγραφής» δεδομένων. Σχεδιάστηκε με σκοπό να εστιάσει κυρίως στον προσδιορισμό του είδους των δεδομένων. Όπως και η HTML (HyperText Markup Language) είναι μία γλώσσα σήμανσης και όχι προγραμματισμού και στηρίζεται στις ετικέτες (tags). Σε αντίθεση με την HTML όπου τα tagsXM είναι προδιαγεγραμμένα, τα tags που χρησιμοποιούνται στην XML πρέπει να προσδιοριστούν από τον χρήστη, καθώς η γλώσσα δεν διαθέτει προκαθορισμένα tags (με εξαίρεση το tag <XML>). Ο προσδιορισμός των tags γίνεται με χρήση ενός XML σχήματος (XML Schema) ή ενός DTD (Document Type Definition). Ο συνδυασμός του αρχείου XML με το XML Schema (ή το DTD) αποτελούν την αυτοπεριγραφή των δεδομένων. Η XML δεν αντικαθιστά την HTML καθώς η καθεμία σχεδιάστηκε για διαφορετικούς σκοπούς. Πιο συγκεκριμένα:

- XML: Να μεταφέρει και να αποθηκεύει δεδομένα και να εστιάζει στην απάντηση του ερωτήματος «τι είδους δεδομένα είναι αυτά»
- HTML: Να εμφανίζει δεδομένα και να εστιάζει στο «πώς μπορώ να παρουσιάσω/εμφανίσω τα δεδομένα αυτά».

Όπως αναφέρει χαρακτηριστικά και η ίδια η W3C, η XML κατ' ουσίαν δεν κάνει τίποτα (XML Does Not DO Anything). Παραθέτουμε το κλασσικό παράδειγμα ενός σημειώματος από τον Tove στη Jani το οποίο είναι αποθηκευμένο σε XML μορφή.

```
<note>
   <to>Tove</to>
   <from>Jani</from>
   <heading>Reminder</heading>
   <body>Don't forget me this weekend!</body>
</note>
```

Το παραπάνω σημείωμα είναι αυτοπροσδιοριστικό. Το σημείωμα αποτελείται από το κυρίως σώμα (body) και από μία επικεφαλίδα για το θέμα του σημειώματος (heading) και περιέχει πληροφορία για τον αποστολέα (sender) και τον παραλήπτη (receiver). Παρ' όλα αυτά η XML δεν κάνει κατ' ουσίαν τίποτα και για την περαιτέρω επεξεργασία της πληροφορίας (αποστολή, λήψη και εμφάνιση) απαιτείται η συγγραφή πρόσθετου κώδικα. Όπως προαναφέραμε στην XML δεν υπάρχουν προκαθορισμένα tags, ο χρήστης έχει την ελευθερία να καθορίσει από μόνος του τη δομή των δεδομένων που θέλει να περιγράψει προσδιορίζοντας τις κατάλληλες ετικέτες. Αυτή είναι και μεγάλη δύναμη της XML καθώς ο κάθε χρήστης έχει τη δυνατότητα να επεκτείνει τα tags του όπως ο ίδιος επιθυμεί.

Στη συνέχεια αναφέρουμε μερικές από τις χρήσεις της XML:

- Διαχωρίζει τα δεδομένα από τον HTML κώδικα

Σε περιπτώσεις που απαιτείται η προβολή δυναμικών δεδομένων (dynamic data) αν χρησιμοποιούσαμε μόνο HTML κάθε φορά που τα δεδομένα θα άλλαζαν θα έπρεπε να αλλάξουμε το HTML αρχείο. Με χρήση της XML μπορεί να απλοποιηθεί η σύζευξη μεταξύ των πραγματικών δεδομένων (XML) και του πώς θα εμφανίζονται τα δεδομένα (HTML/CSS).

- Απλοποιεί τον διαμοιρασμό και την αποθήκευση των δεδομένων

Τα υπολογιστικά συστήματα καθώς και οι βάσεις δεδομένων, διατηρούν πληροφορία σε διαφορετικές μορφές, οι οποίες συνήθως δεν είναι συμβατές μεταξύ τους. Τα XML δεδομένα αποθηκεύονται σε ένα απλό αρχείο κειμένου (plaintext format), παρέχοντας έτσι μία μέθοδο αποθήκευσης ανεξάρτητη οποιουδήποτε λογισμικού ή υλικού. Το γεγονός αυτό καθιστά την ανταλλαγή δεδομένων μεταξύ διάφορων εφαρμογών πολύ πιο απλή.

- Ανταλλαγή πληροφορίας

Καθώς η XML αποτελεί ένα διεθνές και κοινώς αποδεκτό πρότυπο χρησιμοποιείται ευρέως για την ανταλλαγή δεδομένων μεταξύ ετερογενών συστημάτων. Η μετατροπή των δεδομένων σε XML μορφή ουσιαστικά ομογενοποιεί την ανταλλασσόμενη πληροφορία. Δίνει λύση σε ένα πολύπλοκο και χρονοβόρο πρόβλημα όπως η ανταλλαγή πληροφορίας μεταξύ ασύμβατων συστημάτων. Τέλος, η XML τείνει να γίνει το πρότυπο στην ανταλλαγή οικονομικών δεδομένων (Business to Business, B2B).

- Ευκολότερη μετάβαση σε νέα πλατφόρμα

Οι αναβαθμίσεις σε νέα συστήματα (λογισμικού ή υλικού) είναι διαδικασίες δύσκολες, χρονοβόρες και συχνά έχουν ως αποτέλεσμα την απώλεια δεδομένων. Η χρήση της XML καθιστά τις αναβαθμίσεις συστημάτων πολύ πιο ομαλές και απλές διαδικασίες χάρις στην απλότητα της.

- Καθιστά τα δεδομένα ευπρόσιτα

Διαφορετικές εφαρμογές, πέρα από σελίδες HTML, έχουν τη δυνατότητα πρόσβασης στην πληροφορία, η οποία καθίσταται πλέον αντιληπτή από πολλά διαφορετικά συστήματα υλικού και λογισμικού όπως κινητά τηλέφωνα, ταμπλέτες και news feeds.

- Η XML έχει χρησιμοποιηθεί για τη δημιουργία νέων Διαδικτυακών Γλωσσών.

  Μερικές από τις γλώσσες οι οποίες δημιουργήθηκαν με χρήση της XML είναι:

  - XHTML (Extensible HyperText Markup Language)
  - WSDL (Web Service Definition Language) για την περιγραφή Υπηρεσιών Ιστού (Web Services)
  - WAP (Wireless Application Protocol) και WML (Wireless Markup Language) γλώσσες σήμανσης για handheld συσκευές
  - RSS (Really Simple Syndication) γλώσσες για news feeds
  - SMIL (Synchronized Multimedia Integration Language) για περιγραφή πολυμέσων στο Παγκόσμιο Ιστό
  - RDF (Resource Description Framework) και OWL (Web Ontology Language) για την περιγραφή αντικειμένων και οντολογιών.

2.4 Το RDF (Resource Description Framework)

Όπως και η XML το RDF αποτελεί και αυτό πρότυπο-σύσταση της W3C από τον Φεβρουάριο του 2004 και δημιουργήθηκε με σκοπό την αναπαράσταση δεδομένων και ανταλλαγής γνώσης στο Διαδίκτυο, συχνά χρησιμοποιείται για την περιγραφή πληροφορίας σχετικής με τον τίτλο, τον συγγραφέα, το περιεχόμενο και την πνευματική ιδιοκτησία Διαδικτυακών σελίδων. Είναι σχεδιασμένο έτσι ώστε να είναι αναγνώσιμο και αντιληπτό από υπολογιστικά συστήματα και όχι για την προβολή του σε ανθρώπους. Το RDF, όπως προαναφέραμε, είναι γραμμένο σε XML και η γλώσσα που χρησιμοποιεί ονομάζεται RDF/XML. Ακόμα, αποτελεί θεμελιώδες κομμάτι του Σημασιολογικού Ιστού. Είναι ανεξάρτητο από το πεδίο εφαρμογής καθώς και μπορεί να ιδωθεί σαν ένας κατευθυνόμενος γράφος με ετικέτες. Η ανταλλαγή RDF πληροφορίας μπορεί να γίνει εύκολα ακόμα και ανάμεσα σε υπολογιστικά συστήματα με διαφορετικό υλικό και λογισμικό. Το μοντέλο δεδομένων του RDF είναι ένα αφηρημένο, εννοιολογικό επίπεδο ανεξάρτητο από την XML, έτσι η XML μπορεί μεν να παρέχει τη σύνταξη για το RDF, δεν αποτελεί όμως συστατικό του.

Στο RDF χρησιμοποιούνται συμβολοσειρές (Uniform Resource Identifiers - URIs) που ταυτοποιούν μοναδικά πόρους (resources) όπως ένα Web site, μία ιδιότητα, έναν άνθρωπο κλπ. Το RDF περιγράφει τις οντότητες μέσω ιδιοτήτων (properties) και τιμών (property values). Πιο συγκεκριμένα:

- Ως resource ορίζεται οτιδήποτε στο οποίο μπορεί να αντιστοιχηθεί ένα URI. Για παράδειγμα: http://www.dit.uop.gr/index
- Ως property ορίζεται ένα resource το οποίο έχει ένα όνομα, όπως «homepage»

Ως property value ορίζεται η «τιμή» που παίρνει ένα property. Ένα resource μπορεί να χρησιμοποιηθεί και ως property value.

Παραθέτουμε ένα απλό παράδειγμα RDF στο οποίο για λόγους απλότητας παραλείπονται τα namespaces.

```
<?xml version="1.0"?>


<RDF>
  <Description about="http://www.w3schools.com/rdf">
    <author>Jan Egil Refsnes</author>
    <homepage>http://www.w3schools.com</homepage>
  </Description>
</RDF>
```

Να σημειωθεί ότι η ενδεδειγμένη κατάληξη για αρχεία είναι «.rdf» παρ' όλα αυτά συνήθως χρησιμοποιείται η κατάληξη «.xml» για λόγους συμβατότητας με παλαιότερους xml parsers.

Ο συνδυασμός ενός Resource, ενός Property και ενός Property Value δημιουργούν ένα Statement. Οι δηλώσεις καλούνται και τριάδες (triples) καθώς αποτελούνται από τρία στοιχεία. Τα statements αποτελούνται από:

- Υποκείμενο (subject): Ταυτοποιεί αυτό στο οποίο αναφέρεται μία δήλωση
- Κατηγόρημα (predicate): Ταυτοποιεί την ιδιότητα ή το χαρακτηριστικό του υποκειμένου το οποίο καθορίζει τη δήλωση.
- Αντικείμενο (object): Ταυτοποιεί την τιμή της ιδιότητας.

Για παράδειγμα στη δήλωση «Ο συγγραφέας τους http://www.1984.com/rdf είναι ο George Orwell.»

Έχουμε:

- Subject: http://www.1984.com/rdf.
- Predicate: συγγραφέας.
- Object: George Orwell.

Οι δηλώσεις μπορούν να απεικονιστούν με έναν γράφο όπου οι κόμβοι αναπαριστούν resources και property values ενώ οι ακμές αντιπροσωπεύουν properties. Το αποτέλεσμα είναι ένας γράφος όπως αυτός που απεικονίζεται στο Σχήμα 5.

Σχήμα 5: Αναπαράσταση RDF Statement με κατευθυνόμενο γράφο.

2.5 Το RDF-S (Resource Description Framework Schema)

Αποτελεί επέκταση του RDF και παρέχει αρχές μοντελοποίησης για την οργάνωση πόρων του Παγκόσμιου Ιστού σε ιεραρχίες. Ακόμα παρέχει μηχανισμούς για την περιγραφή συλλογών από ομοειδείς πόρους καθώς και των σχέσεων ανάμεσα στους πόρους αυτούς. Βασικά στοιχεία του είναι οι κλάσεις (classes) και οι ιδιότητες (properties), ενώ επιπλέον υπάρχουν οι έννοιες της υποκλάσης (subclass) και της υπο-ιδιότητας (sub property). Ακόμα υπάρχει η δυνατότητα ορισμού περιορισμών πεδίου ορισμού (domain) και συνόλου τιμών (range) μίας ιδιότητας. Το RDFS ως επέκταση του RDF βασίζεται σ' αυτό. Μπορεί να θεωρηθεί ως μία πρωταρχική γλώσσα για τη σύνταξη οντολογιών, όμως συνήθως απαιτείται η χρήση πιο ισχυρών γλωσσών οντολογιών (με πιο ισχυρά semantics) όπως η OWL.

Κλάσεις: Οι πόροι είναι δυνατό να χωριστούν σε ομάδες που ονομάζονται κλάσεις (classes). Τα μέλη μίας κλάσης ονομάζονται στιγμιότυπα (instances) της κλάσης. Οι κλάσεις είναι και αυτές πόροι. Στο RDFS υπάρχει σαφής διάκριση μεταξύ μίας κλάσης και των στιγμιοτύπων της. Με κάθε κλάση συνδέεται ένα σύνολο το οποίο ονομάζεται επέκταση της κλάσης (class extension), το οποίο αποτελείται από το σύνολο των στιγμιοτύπων της κλάσης αυτής. Δύο κλάσεις μπορεί ενώ έχουν το ίδιο σύνολο στιγμιοτύπων να είναι παρ' όλα αυτά διαφορετικές κλάσεις.

Υποκλάσεις: Η ιδιότητα rdfs:subClassOf μπορεί να χρησιμοποιηθεί για να δηλώσει ότι μία κλάση είναι υποκλάση μίας άλλης κλάσης. Η ιδιότητα της υποκλάσης ισχύει μεταβατικά δηλαδή αν μία κλάση C αποτελεί υποκλάση της κλάσης B, τότε όλα τα στιγμιότυπα της C είναι επίσης στιγμιότυπα της κλάσης B.

Υπερκλάσεις: Ο όρος υπερκλάση (super-class) χρησιμοποιείται σαν ο ανάστροφος του όρου υποκλάση. Η μεταβατικότητα ισχύει και σ' αυτήν την περίπτωση έτσι αν μία κλάση B είναι υπερκλάση μίας κλάσης C, τότε όλα τα στιγμιότυπα της C είναι επίσης και στιγμιότυπα της B.

Όλα τα αντικείμενα τα οποία περιγράφονται με RDFS ονομάζονται πόροι, και είναι στιγμιότυπα της κλάσης rdfs:Resource. Αυτή η κλάση περιλαμβάνει τα πάντα και όλες οι άλλες κλάσεις αποτελούν υποκλάσεις της.

2.6 Οι Οντολογίες

Η λέξη οντολογία έχει ρίζες στα αρχαία Ελληνικά και προέρχεται από τις λέξεις ον και λόγος. Σύμφωνα με τον T.Gruber (1993) μία οντολογία αποτελεί μία τυπική, καλά ορισμένη προδιαγραφή μίας κοινής εννοιολογικής θεώρησης ενός φαινομένου. Η εννοιολογική θεώρηση είναι μία αφηρημένη, απλοποιημένη όψη του κόσμου που θέλουμε να αναπαραστήσουμε. Στον Σημασιολογικό Ιστό οι οντολογίες συνήθως εκφράζονται σε γλώσσες όπως η RDFS και η OWL οι οποίες βασίζονται σε αντικείμενα (objects/individuals), έννοιες (concepts/classes), συσχετίσεις (relations) και αξιώματα (axioms). Υπάρχουν πολλαπλοί λόγοι για την ανάπτυξη μίας οντολογίας όπως:

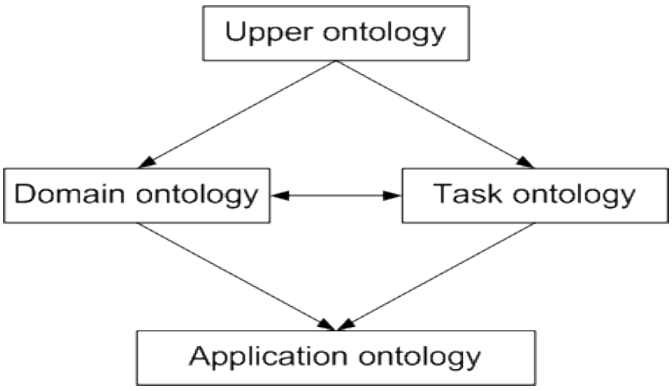
Ο διαμοιρασμός μίας κοινής κατανόησης της δομής της πληροφορίας σχετικής με έναν τομέα μεταξύ ανθρώπων ή πρακτόρων λογισμικού, αποτελεί έναν από τους πιο σημαντικούς λόγους για την ανάπτυξη μίας οντολογίας (Mussen 1992, Gruber 1993). Για παράδειγμα, ας υποθέσουμε ότι υπάρχει ένας αριθμός από ιστοσελίδες οι οποίες παρέχουν πληροφορίες σχετικές με την υγεία. Αν όλες οι σελίδες χρησιμοποιούν μία κοινή οντολογία στην οποία περιέχονται οι όροι που χρησιμοποιούν, τότε με τη χρήση υπολογιστικών πρακτόρων μπορούν είτε να εξάχθουν πρόσθετες είτε να συνενώσουν υπάρχουσες πληροφορίες ώστε να απαντάνε αποδοτικά σε ερωτήματα χρηστών ή να χρησιμοποιήσουν τα «ενοποιημένα» δεδομένα ως είσοδο σε άλλες εφαρμογές.

Η επαναχρησιμοποίηση γνώσης σχετικής με κάποιον τομέα είναι ένας εξίσου σημαντικός λόγος για την ανάπτυξη μίας οντολογίας. Για παράδειγμα, σε πολλές περιπτώσεις απαιτείται η μοντελοποίηση της έννοιας του χρόνου. Αν μία ομάδα ερευνητών αναπτύξει προσεχτικά και με λεπτομέρεια μία τέτοια οντολογία, τότε αυτή μπορεί να χρησιμοποιηθεί αυτούσια από άλλους επιστήμονες για να εξυπηρετήσει τις ανάγκες τους.

Επίσης υπάρχει η δυνατότητα συνένωσης οντολογιών με σκοπό τη δημιουργία μίας μεγαλύτερης, γεγονός πολύ χρήσιμο καθώς επιτρέπει την αποσύνθεση ενός τομέα σε μικρότερους, την ανάπτυξη ξεχωριστών μικρότερων οντολογιών και έπειτα τη συνένωσή τους σε μία μεγάλη οντολογία. Δηλαδή, εφαρμόζοντας την τεχνική «διαίρει και βασίλευε» επιτυγχάνουμε την κατάτμηση ενός γνωστικού τομέα σε περισσότερους, καθιστώντας έτσι ευκολότερη τη μοντελοποίηση τους.

Τέλος ακόμα και αν μία υπάρχουσα οντολογία για έναν γνωστικό τομέα δεν καλύπτει τις ανάγκες μας στο έπακρο, υπάρχει πάντα η δυνατότητα επέκτασης της υπάρχουσας οντολογίας προσθέτοντας νέες κλάσεις ή συσχετίσεις μεταξύ αυτών. Πολλές φορές η ανάπτυξη μίας οντολογίας δεν είναι αυτοσκοπός, συχνά αναπτύσσονται οντολογίες με σκοπό την οργάνωση των δεδομένων που δέχεται κάποιο πρόγραμμα στην είσοδο του. Αναφέρουμε ότι μία οντολογία η οποία περιέχει στιγμιότυπα (instances) των κλάσεων αποτελεί μία γνωσιακή βάση.

Για την απλοποίηση των οντολογιών αλλά και την αποδοτικότερη επαναχρησιμοποίησή τους ενδείκνυται η ανάπτυξη τους σε τμήματα. Χρησιμοποιείται δηλαδή η έννοια της κληρονομικότητας στην ανάπτυξη οντολογιών. Έτσι, προκύπτει η ιεραρχία οντολογιών που απεικονίζεται στο Σχήμα 6:



Σχήμα 6: Η ιεραρχία τμηματοποίησης των οντολογιών.

Αναλυτικότερα τα τμήματα της ιεραρχίας οντολογιών είναι:

- **Upper Ontology**: Οντολογία υψηλότερου επίπεδου, η οποία περιγράφει γενικές έννοιες όπως χώρος και χρόνος
- **Domain Ontology**: Οντολογίες οι οποίες περιγράφουν πιο συγκεκριμένους τομείς όπως ο ιατρικός ή ακόμα και πιο συγκεκριμένους τομείς όπως είναι η φαρμακοβιομηχανία
- **Task Ontology**: Πρόκειται για οντολογία που περιγράφει τον ακριβή τρόπο με τον οποίο πραγματοποιείται μία συγκεκριμένη εργασία όπως, για παράδειγμα, η συναρμολόγηση ενός αυτοκινήτου
- **Application Ontology**: Συνιστά το κατώτερο μέρος της ιεραρχίας. μία application ontology έχει αναπτυχθεί για μία πολύ συγκεκριμένη εφαρμογή. Όπως βλέπουμε στο Σχήμα 6 μία application ontology μπορεί να έχει προκύψει από συνένωση μίας domain και μίας task οντολογίας.

Οι οντολογίες στα κατώτερα τμήματα της ιεραρχίας κληρονομούν στοιχεία από τις οντολογίες που βρίσκονται σε υψηλότερο τμήμα, ενώ ταυτόχρονα τις επεκτείνουν προσθέτοντας πιο εξειδικευμένη γνώση. Όπως βλέπουμε και στο σχήμα οι domain και task οντολογίες προκύπτουν επεκτείνοντας μία upper οντολογία ενώ μία application ontology μπορεί να έχει προκύψει από συνένωση μίας domain και μίας task οντολογίας. Οι upper οντολογίες είναι οι πλέον επαναχρησιμοποιήσιμες ενώ οι application συνήθως δεν επαναχρησιμοποιούνται ποτέ. Σημειώνεται ότι η οντολογία που αναπτύχθηκε για τις ανάγκες αυτής της εργασίας κατατάσσεται στη κατηγορία των application οντολογιών.

Επίσης είναι δυνατή η χρησιμοποίηση πολλαπλών οντολογιών από ένα σύστημα, περίπτωση στην οποία ίσως απαιτούνται κάποιες πρόσθετες διεργασίες για να γίνει δυνατή η ταυτόχρονη χρήση τους. Αναφέρουμε ενδεικτικά μερικές απ' αυτές χωρίς να τις αναλύουμε περαιτέρω.

- Merge
- Mapping
- Alignment
- Refinement
- Unification
- Integration
- Inheritance

Σε γενικές γραμμές οι παραπάνω διεργασίες είναι εξαιρετικά δύσκολες, ενώ δεν μπορούν να πραγματοποιηθούν πλήρως αυτοματοποιημένα.

2.7 Η OWL (Web Ontology Language)

Η OWL έχει αρκετά κοινά στοιχεία με το RDF, αλλά πρόκειται για μία πολύ πιο εκφραστική γλώσσα, με πιο ισχυρή σημασιολογία (semantics) και ερμηνευσιμότητα (interpretability) από το RDF-S. Διαθέτει πολύ μεγαλύτερο λεξιλόγιο καθώς και ισχυρότερους συντακτικούς κανόνες. Όπως και στο RDF έτσι και στην OWL υπάρχει η δυνατότητα δημιουργίας τριάδων (triples) οι οποίες απαρτιζόμενων από subject, predicate και object. Ακόμα στην OWL υπάρχει η δυνατότητα να οριστεί ότι δύο διαδικτυακά αντικείμενα αντιστοιχούν/δείχνουν στο ίδιο φυσικό αντικείμενο, όπως για παράδειγμα ότι το www.wikipedia.org/wiki/Michael_Jackson αντιστοιχεί στο ίδιο αντικείμενο/φυσικό πρόσωπο με το www.poplegends.com/Michael_Jackson. Αυτό είναι πολύ σημαντικό καθώς δίνει τη δυνατότητα συνένωσης δεδομένων τα οποία εκφράζονται σε διαφορετικά σχήματα. Η OWL επεκτείνει τα RDF και RDFS και δημιουργήθηκε με σκοπό να προσδώσει στον Σημασιολογικό Ιστό τεράστια εκφραστική και συλλογιστική δύναμη (expressive and reasoning power). Ο όρος συλλογιστική δύναμη (reasoning power) έχει να κάνει με την επεξεργασία δεδομένων με σκοπό την εξαγωγή συμπερασμάτων καθώς και νέας γνώσης. Τα θεμέλια της OWL βρίσκονται στον τομέα των Περιγραφικών Λογικών (Description Logic, SHOIN/SROIQ). Η OWL υποστηρίζεται ιδιαίτερα από την κοινότητα της πληροφορικής και έτσι γύρω της έχουν αναπτυχθεί μία σειρά εργαλείων:

- APIS (e.g., OWL API, Thea, OWLink, Jena)
- Περιβάλλοντα ανάπτυξης οντολογιών (Protégé, Swoop, TopBraid Composer, Neon)
- Reasoners (Pellet, Fact++, Racer, HermiT, Quonto).

Για την επίτευξη διαστρωμάτωσης μέσα στην OWL δημιουργήθηκαν τρεις εκφραστικές υπο-γλώσσες της OWL (OWL Lite, OWL Full, OWL DL) σχεδιασμένες για διαφορετική χρήση η κάθε μία.

OWL Lite: Η OWL Lite δεν χρησιμοποιεί όλες της εκφραστικές δυνατότητες που παρέχει η OWL και έχει περισσότερους περιορισμούς στη χρήση της σε σχέση με τις OWL Full και OWL DL. Απευθύνεται σε χρήστες οι οποίοι επιθυμούν να χρησιμοποιήσουν την OWL για την περιγραφή γνώσης σε εφαρμογές που δεν έχουν μεγάλες απαιτήσεις όσον

αφορά την εκφραστική δύναμη της γλώσσας. Το γεγονός ότι είναι σχετικά περιορισμένη δίνει τη δυνατότητα ανάπτυξης εξειδικευμένων εργαλείων και μηχανισμών εξόρυξης γνώσης τα οποία θα λειτουργούν πιο γρήγορα και αποτελεσματικά σε σχέση με εργαλεία τα οποία υλοποιούν τις πιο εκφραστικές υπο-γλώσσες της OWL. Μιλώντας με όρους Description Logics, η γλώσσα παρέχει την ίδια εκφραστική δυνατότητα με τη γλώσσα SHIF(D).

OWL DL: Η OWL DL παρέχει στον χρήστη τη μέγιστη εκφραστική δυνατότητα που προσφέρεται από την OWL, διατηρώντας όμως τις καλές υπολογιστικές και συλλογιστικές ιδιότητές της. Αυτό σημαίνει ότι η OWL DL είναι αποφασίσιμη (decidable) δηλαδή υπάρχει η δυνατότητα χρήσης ενός reasoner για την εξαγωγή συμπερασμάτων και πρόσθετης γνώσης κάτι που δεν ισχύει για την OWL Full. Συγκριτικά με τις Description Logics η OWL DL παρέχει την ίδια εκφραστικότητα με την περιγραφική λογική SHOIN(D).

OWL Full: Η OWL Full προσφέρει το ίδιο λεξιλόγιο με την OWL DL, επιπρόσθετα όμως παρέχει τη συντακτική ελευθερία και τα χαρακτηριστικά του RDF και πιο συγκεκριμένα τη δυνατότητα της μετα-μοντελοποίησης. Στην OWL Full μία κλάση μπορεί να αντιμετωπιστεί ταυτόχρονα ως μία συλλογή στιγμιοτύπων (individuals) αλλά και ως ένα στιγμιότυπο από μόνη της κάτι το οποίο δεν επιτρέπεται στην OWL DL. Επίσης επιτρέπεται η επαύξηση της σημασιολογίας του προκαθορισμένου λεξιλογίου που παρέχεται από RDF και OWL. Η OWL Full είναι μη αποφασίσιμη (undecidale) (Boris 2005). Από τη μία παρέχει συμβατότητα με το μοντέλο και τη σημασιολογία του RDF, από την άλλη η μεγάλη εκφραστική δύναμή της δεν επιτρέπει τη δημιουργία reasoner, καθώς μέχρι σήμερα δεν είναι γνωστός κανένας αλγόριθμος εξαγωγής συμπερασμάτων γι' αυτήν.

Γίνεται αντιληπτό ότι υπάρχει ένα tradeoff μεταξύ της εκφραστικότητας της γλώσσας και της δυνατότητας εξαγωγής συμπερασμάτων και πρόσθετης γνώσης. Γι' αυτό το λόγο η ομάδα εργασίας της OWL (OWL Working Group OWG) δημιούργησε τις OWL Lite και OWL DL για τις οποίες οι αλγόριθμοι εξαγωγής συμπερασμάτων προϋπήρχαν από το πεδίο των Description Logics.

Τα βασικά δομικά στοιχεία της OWL είναι τα αντικείμενα(individuals/objects), οι έννοιες (concepts/classes), οι συσχετίσεις (relations) και τα αξιώματα (axioms).

2.7.1 Οντολογικές Κλάσεις

Στην OWL οι κλάσεις ορίζονται ως συλλογές αντικειμένων. Περιγράφονται φορμαλιστικά με εκφράσεις που δηλώνουν επ' ακριβώς τις απαιτήσεις για να είναι ένα αντικείμενο μέλος μίας κλάσης. Οι κλάσεις μπορούν να οργανωθούν σε μία ιεραρχία από υπερ-κλάσεις και υπο-κλάσεις (superclass/subcass hierarchy) το οποίο είναι γνωστό και ως ταξινομία (taxonomy). Οι υπο-κλάσεις αποτελούν εξειδικεύσεις των υπερ-κλάσεων. Το στιγμιότυπο μίας υποκλάσης μπορεί να θεωρηθεί ότι ανήκει και στην υπερκλάση. Αν θεωρήσουμε την υπερκλάση «Ζώο» και την υποκλάση της «Γάτα», τότε όλα τα στιγμιότυπα της κλάσης «Γάτα» θα ανήκουν και στην κλάση «Ζώο». Να σημειωθεί ότι ένα στιγμιότυπο μπορεί να ανήκει σε παραπάνω από μία κλάση. Δύο κλάσεις μπορούν να οριστούν ως αμοιβαίως αποκλειόμενες (disjoint), υποδηλώνοντας ότι ένα στιγμιότυπο δεν

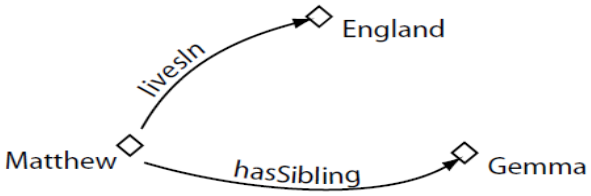
μπορεί να ανήκει και στις δύο κλάσεις. Για παράδειγμα ένα στιγμιότυπο της κλάσης «`Αγόρι`» δεν μπορεί να ανήκει στην κλάση «`Κορίτσι`».

2.7.2 Στιγμιότυπα Κλάσης

Όλα τα μέλη μίας κλάσης αναφέρονται ως στιγμιότυπά της κλάσης. Η OWL δίνει τη δυνατότητα ορισμού των προϋποθέσεων που θα πρέπει να ισχύουν για ένα στιγμιότυπο ώστε να γίνει μέλος μίας κλάσης.

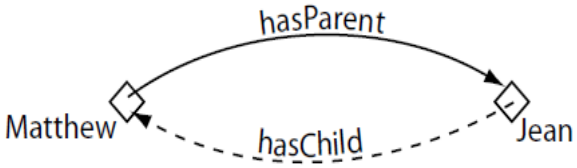
2.7.3 Αντικειμενικές Ιδιότητες

Οι αντικειμενικές ιδιότητες χρησιμοποιούνται για τη συσχέτιση στιγμιοτύπων μεταξύ τους. Το στιγμιότυπο Matthew συσχετίζεται με τα στιγμιότυπα «`England`» και «`Gemma`» μέσω των αντικειμενικών ιδιοτήτων «`livesIn`» και «`hasSibling`» αντίστοιχα.



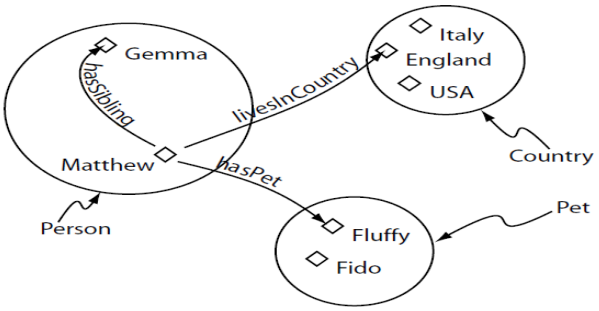
Σχήμα 7: Παράδειγμα αντικειμενικής ιδιότητας μεταξύ στιγμιότυπων.

Μια αντικειμενική ιδιότητα μπορεί να έχει και μία αντίστοιχη αντίστροφη (inverse) ιδιότητα.



Σχήμα 8: Παράδειγμα αντίστροφης αντικειμενικής ιδιότητας μεταξύ στιγμιότυπων.

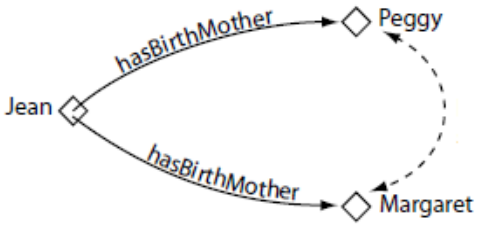
Οι αντικειμενικές ιδιότητες μπορούν να συνδέουν είτε στιγμιότυπα που ανήκουν στην ίδια κλάση είτε στιγμιότυπα που ανήκουν σε διαφορετικές κλάσεις.



Σχήμα 9: Παράδειγμα αντικειμενικών ιδιοτήτων μεταξύ στιγμιοτύπων διαφορετικών κλάσεων.

2.7.3.1 Λειτουργικές Ιδιότητες

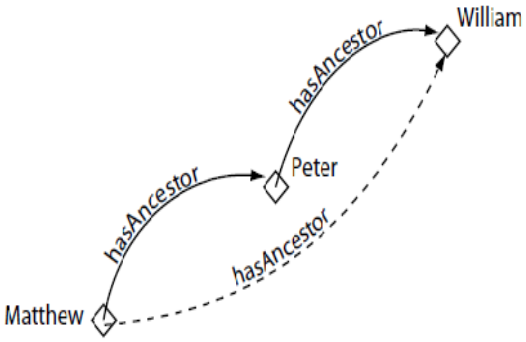
Μία λειτουργική ιδιότητα (functional property) δεν μπορεί να λάβει παραπάνω από ένα στιγμιότυπο ως τιμή. Στο παρακάτω παράδειγμα εφόσον η ιδιότητα «`hasBirthMother`» έχει οριστεί ως λειτουργική μπορούμε να συμπεράνουμε ότι τα στιγμιότυπα «`Margaret`» και «`Peggy`» είναι το ίδιο πρόσωπο.



Σχήμα 10: Παράδειγμα λειτουργικής ιδιότητας.

2.7.3.2 Μεταβατικές Ιδιότητες

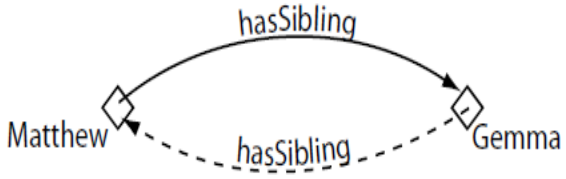
Εάν μία αντικειμενική ιδιότητα έχει οριστεί ως μεταβατική (transitive property) και συνδέει δύο στιγμιότυπα a και b αλλά και το στιγμιότυπο b με ένα στιγμιότυπο c, τότε και το a συνδέεται με το c μέσω αυτής της ιδιότητας. Στο Σχήμα 11 η «`hasAncestor`» είναι μία μεταβατική ιδιότητα αντικειμένων.



Σχήμα 11: Παράδειγμα μεταβατικής ιδιότητας.

2.7.3.3 Συμμετικές Ιδιότητες

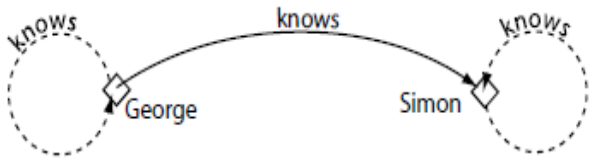
Εάν μία αντικειμενική ιδιότητα έχει οριστεί ως συμμετρική (symmetric property), και ένα στιγμιότυπο a συνδέεται μέσω αυτής με κάποιο άλλο στιγμιότυπο b, τότε και το b συνδέεται με το a μέσω της ίδιας ιδιότητας. Στο Σχήμα 12 η ιδιότητα αντικειμένων «`hasSibling`» είναι συμμετρική.



Σχήμα 12: Παράδειγμα συμμετρικής ιδιότητας.

2.7.3.4 Ανακλαστικές Ιδιότητες

Αν μία ιδιότητα η οποία ορίζεται ως ανακλαστική (reflexive property) θα πρέπει να ισχύει και για το ίδιο το στιγμιότυπο που την έχει ως subject. Στο Σχήμα 13 η ιδιότητα «knows» είναι ανακλαστική.



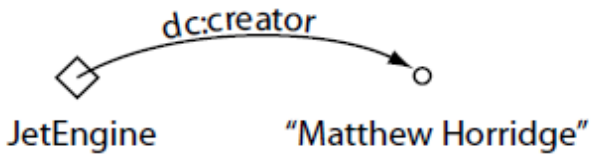
Σχήμα 13: Παράδειγμα ανακλαστικής ιδιότητας.

2.7.3.5 Πεδίο Ορισμού και Σύνολο Τιμών των Ιδιοτήτων

Οι αντικειμενικές ιδιότητες δύνανται να παίρνουν κλάσεις ως πεδίο ορισμού και σύνολο τιμών. Αυτό σημαίνει ότι το στιγμιότυπο από το οποίο ξεκινάει η ιδιότητα θα πρέπει να ανήκει στην κλάση η οποία έχει οριστεί ως πεδίο ορισμού της ιδιότητας και ότι το στιγμιότυπο στο οποίο δείχνει η ιδιότητα θα πρέπει να ανήκει στην κλάση η οποία έχει οριστεί ως σύνολο τιμών της ιδιότητας.

2.7.4 Ιδιότητες Τύπου Δεδομένων

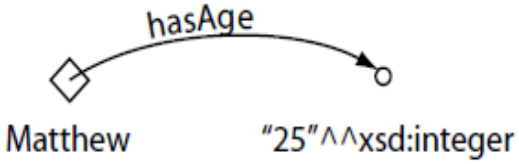
Οι ιδιότητες τύπου δεδομένων συνδέουν ένα στιγμιότυπο με έναν XML Schema τύπο δεδομένων ή ένα literal. Με άλλα λόγια περιγράφουν τις σχέσεις μεταξύ στιγμιοτύπων και τιμών δεδομένων.



Σχήμα 14: Παράδειγμα ιδιότητας τύπου δεδομένων.

2.7.5 Ιδιότητες Επισημείωσης

Η OWL επιτρέπει τη συσχέτιση κλάσεων, στιγμιοτύπων και ιδιοτήτων με διάφορα κομμάτια πληροφορίας ή/και μεταδεδομένα (meta-data) μέσω των ιδιοτήτων επισημείωσης (annotation properties). Μπορούν να χρησιμοποιηθούν για την περιγραφή μίας τιμής που συσχετίζεται με ένα στιγμιότυπο, καθώς και για την προσθήκη πληροφορίας όπως συγγραφέας, ημερομηνία και σχόλια. Ενώ στην OWL Full δεν υπάρχουν περιορισμοί όσον αφορά τις ιδιότητες επισημείωσης στην OWL DL υπάρχει μία σειρά αυστηρών περιορισμών όσον αφορά τη χρήση τους.



Σχήμα 15: Παράδειγμα ιδιότητας επισημείωσης.

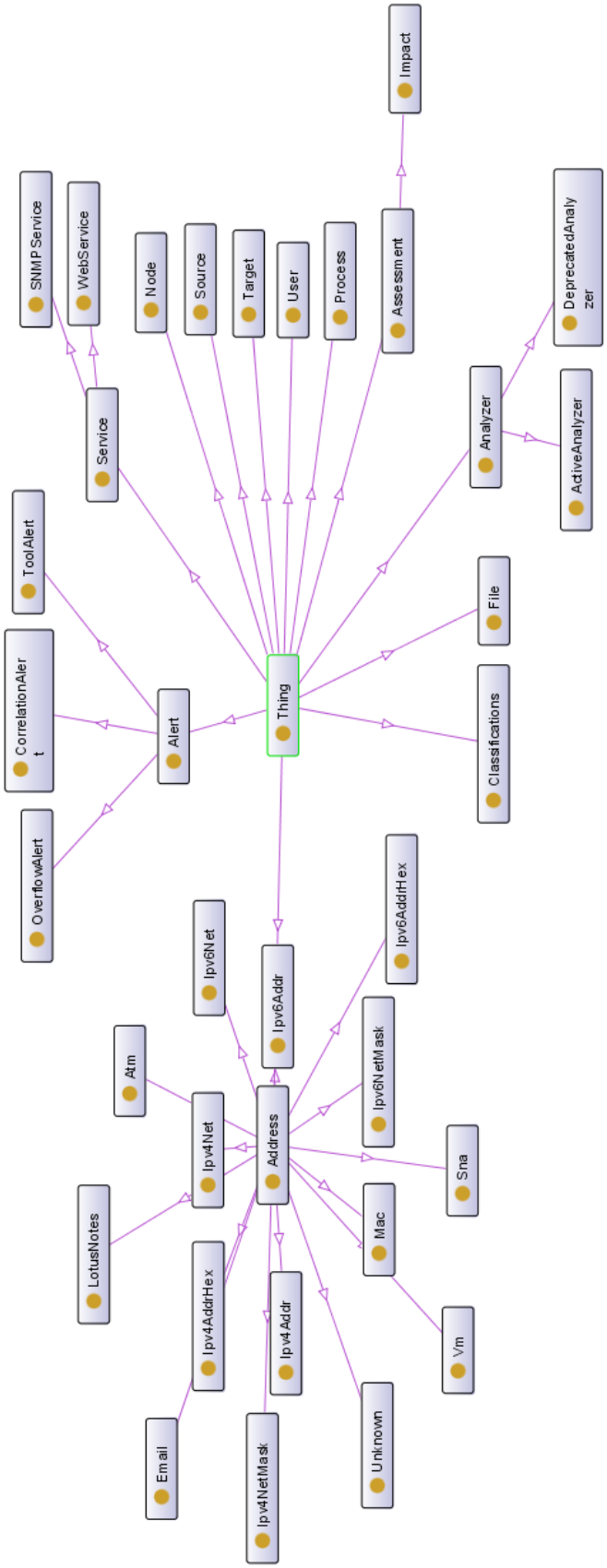
3 Προτεινόμενη Οντολογία για Αναπαράσταση IDMEF

3.1 Εισαγωγή

Για τις ανάγκες της εργασίας δημιουργήθηκε μία οντολογία στη γλώσσα OWL. Πιο συγκεκριμένα για την ανάπτυξη της οντολογίας χρησιμοποιήθηκε η γλώσσα OWL DL, καθώς οι καλές υπολογιστικές και συλλογιστικές δυνατότητες που διαθέτει κρίθηκαν απαραίτητες. Αναλυτικότερα, αν η ανάπτυξη της οντολογίας γινόταν με OWL Full αυτό θα σήμαινε ότι δεν θα ήταν αποφασίσιμη και δεν θα υπήρχε η δυνατότητα εξαγωγής συμπερασμάτων και πρόσθετης γνώσης από αυτήν. Αυτό πρακτικά θα ακύρωνε τον σκοπό αυτής της εργασίας, ο οποίος δεν είναι άλλος από την εξαγωγή πρόσθετης γνώσης σχετικά με περιστατικά ασφαλείας μέσω της αποθήκευσης τους σε μία οντολογία. Για τη δημιουργία της οντολογίας χρησιμοποιήθηκε το περιβάλλον Protégé έκδοση 4.3, το οποίο είναι ένα εργαλείο ανοιχτού κώδικα που παρέχει τα απαραίτητα μέσα για τη δημιουργία μίας οντολογίας, καθώς και την εξαγωγή πρόσθετης γνώσης απ' αυτή μέσω των ενσωματωμένων μηχανών εξαγωγής συμπερασμάτων που διαθέτει (Fact++,Hermit, RacerPro TG, Pellet). Αξίζει να σημειωθεί ότι το Protégé παρέχει τη δυνατότητα «οπτικοποίησης» της οντολογίας και αναπαράστασή της με ένα γράφημα.

Η ανάπτυξη της οντολογίας έγινε έτσι ώστε να υπάρχει η δυνατότητα αναπαράστασης οποιασδήποτε πληροφορίας μπορεί να φέρει ένα συμβάν ασφαλείας το οποίο ακολουθεί τον μορφότυπο του IDMEF. Επίσης, έχουν δημιουργηθεί οι κατάλληλες κλάσεις έτσι ώστε να είναι δυνατή η διαχείριση των αναλυτών. Αυτό επιτεύχθηκε μέσω του διαχωρισμού τους σε ενεργούς και μη-ενεργούς αναλυτές.

Κατά την ανάπτυξη της οντολογίας ιδιαίτερη έμφαση δόθηκε στον προσεκτικό προσδιορισμό κλάσεων, ιδιοτήτων, στιγμιοτύπων καθώς και στην ταξινομία τους έτσι ώστε πιθανά ερωτήματα που θα γίνονται στην οντολογία να μπορούν να απαντηθούν σε βέλτιστο χρόνο. Χωρίς να απαιτείται δηλαδή η διάσχιση τεραστίων γράφων, όποτε τουλάχιστον αυτό ήταν εφικτό.



Σχήμα 16: Η προτεινόμενη οντολογία.

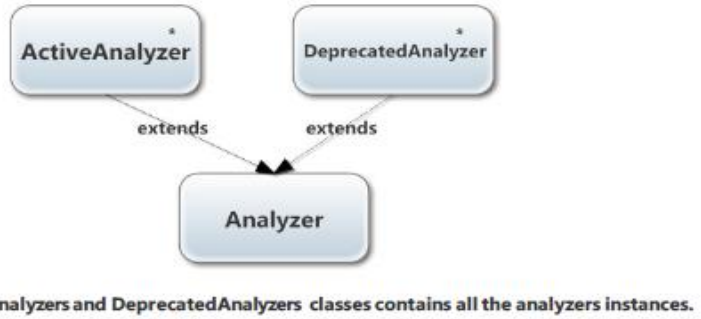
3.2 Ανάλυση της Οντολογίας

Αρχικά θα αναλύσουμε τις οντολογικές κλάσεις που δημιουργήθηκαν και θα αναφέρουμε τι αντιπροσωπεύουν. Το Σχήμα 16 απεικονίζει την οντολογία IDMEF.

`Analyzer`: Αναπαριστά τους αναλυτές/σένσορες οι οποίοι είναι υπεύθυνοι για την ανίχνευση εισβολών και επιθέσεων, τη δημιουργία των συμβάντων ασφαλείας και την αποστολή τους στο κεντρικό σύστημα. Χωρίζεται σε δύο υποκλάσεις, την `ActiveAnalyzer` και την `DeprecatedAnalyzer`. Είναι μία αφηρημένη (abstract) οντολογική κλάση καθώς δεν υπάρχει η δυνατότητα δημιουργίας στιγμιοτύπων της, παρά μόνον η δημιουργία στιγμιοτύπων των υποκλάσεων της. Ωστόσο, χρησιμεύει στην ομαδοποίηση όλων των αναλυτών σε μία οντολογική κλάση και στην απάντηση ερωτημάτων όπως «Εμφάνισε όλους του αναλυτές ενεργούς και ανενεργούς».

Η `ActiveAnalyzer` αναπαριστά τους αναλυτές οι οποίοι έχουν ορισθεί ως ενεργοί και περιέχει ένα στιγμιότυπο για κάθε αναλυτή, ο οποίος τη δεδομένη χρονική στιγμή έχει οριστεί ως ενεργός. Οποιοδήποτε μια ειδοποίηση για ένα συμβάν ασφάλειας φτάσει στο σύστημα, το οποίο δεν έχει προέλθει από αναλυτή που να αντιστοιχεί σε κάποιο στιγμιότυπο της `ActiveAnalyzer` θα απορρίπτεται ως μη έγκυρο.

Από την άλλη πλευρά, η υποκλάση `DeprecatedAnalyzer` αντιπροσωπεύει τα μηχανήματα τα οποία ήταν στο παρελθόν ενεργά όμως τώρα έχουν είτε αφαιρεθεί είτε απενεργοποιηθεί προσωρινά. Περιέχει ένα στιγμιότυπο για κάθε τέτοιο μηχάνημα. Αν μία ειδοποίηση που φτάνει στο σύστημα, φαίνεται να έχει προέλθει από έναν αναλυτή ο οποίος αντιστοιχεί σε στιγμιότυπο της `DeprecatedAnalyzer` θα πρέπει να απορρίπτεται ως ύποπτο. Είναι σημαντικό ένας αναλυτής να διατηρείται αποθηκευμένος ακόμα και μετά την κατάργηση του καθώς σε διαφορετική περίπτωση οι ειδοποιήσεις οι οποίες είχαν προέλθει απ' αυτόν δεν θα περιείχαν πληροφορία σχετικά με το ποιος τα δημιούργησε.



Σχήμα 17: Οι οντολογικές κλάσεις `ActiveAnalyzer` και `DeprecatedAnalyzer`.

Οι δύο υποκλάσεις είναι μεταξύ ξένες μεταξύ τους που σημαίνει ότι ένας αναλυτής δεν μπορεί να είναι και ενεργός και ανενεργός την ίδια χρονική στιγμή. Οι παραπάνω κλάσεις πέραν από την αναπαράσταση της πληροφορίας σχετικά με το ποιος δημιούργησε μία ειδοποίηση, είναι ιδιαίτερα σημαντικής σημασίας και για τη διαχείριση των αναλυτών.

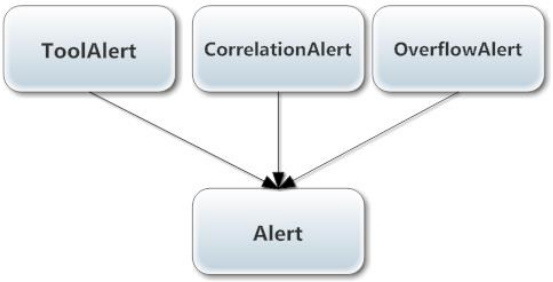
`Alert`: Αντιπροσωπεύει τις ειδοποιήσεις τις οποίες δημιουργούν οι αναλυτές όταν ανιχνεύσουν ύποπτη δραστηριότητα και στέλνουν στο κεντρικό σύστημα. Ανάλογα με τον τύπο του αναλυτή μια ειδοποίηση μπορεί να αντιστοιχεί είτε σε έναν μόνο συμβάν ασφάλειας είτε σε περισσότερα. Οι ειδοποιήσεις δημιουργούνται ασύγχρονα όποτε αυτό κρίνεται σκόπιμο. Η Alert χωρίζεται σε τρεις υποκλάσεις `ToolAlert`, `CorrelationAlert` και `OverflowAlert` η καθεμία από τις οποίες εξυπηρετεί διαφορετικούς σκοπούς.

Η `ToolAlert` αντιπροσωπεύει τα συμβάντα ασφαλείας τα οποία φέρουν πρόσθετη πληροφορία σχετικά με τη χρήση κακόβουλου λογισμικού ή εργαλείων επίθεσης όπως ένας Δούρειος Ίππος (Trojan Horse) και χρησιμοποιείται από έναν αναλυτή ο οποίος έχει τη δυνατότητα να ανιχνεύσει πότε έχει γίνει χρήση τους. Χρησιμοποιείται για την ομαδοποίηση ενός ή περισσότερων ειδοποιήσεων οι οποίες έχουν σταλεί σε κάποια προγενέστερη χρονική στιγμή ώστε να δηλωθεί ότι οι παραπάνω ειδοποιήσεις προκλήθηκαν ως αποτέλεσμα της χρήσης ενός συγκεκριμένου εργαλείου.

Η `CorrelationAlert` αντιπροσωπεύει τα συμβάντα ασφαλείας που στέλνονται από κάποιον αναλυτή με σκοπό να συσχετίσει τις ειδοποιήσεις οι οποίες έχουν σταλεί σε προηγούμενη χρονική στιγμή. Αυτό αποσκοπεί στο να δηλωθεί ότι κάποιες ειδοποιήσεις που στάλθηκαν δεν είναι «ανεξάρτητες» αλλά είναι πολύ πιθανό να είναι αποτέλεσμα της ίδιας επίθεσης.

Η `OverflowAlert` αντιπροσωπεύει τις ειδοποιήσεις τις οποίες στέλνει ένας αναλυτής όταν ανιχνεύσει ότι πραγματοποιείται μία επίθεση υπερχείλισης κάποιου buffer. Χρησιμοποιείται για την αναλυτικότερη περιγραφή τέτοιου είδους επιθέσεων.

Οι τέσσερις προαναφερθείσες κλάσεις `Alert`, `ToolAlert`, `CorrelationAlert` και `OverflowAlert` είναι ξένες μεταξύ τους· αυτό σημαίνει ότι κάθε συμβάν ασφαλείας ανήκει σε μία και μόνο από αυτές τις κατηγορίες.



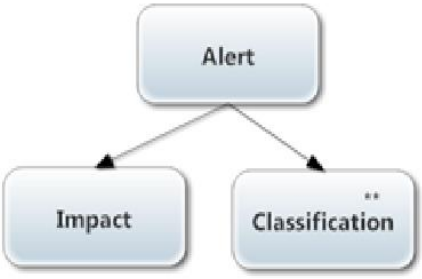
Σχήμα 18: Οι οντολογικές κλάσεις `ToolAlert`, `CorrelationAlert` και `OverflowAlert`.

Μια ειδοποίηση μπορεί να περιέχει πληροφορία σχετικά με το ποιος ήταν ο αντίκτυπος της επίθεσης στο σύστημα (impact) καθώς και πληροφορία κατηγοριοποίησης που υποδηλώνει πως η εν λόγω ειδοποίηση ήταν αποτέλεσμα μίας κατηγορίας επιθέσεων ή εκμετάλλευση κάποιου γνωστού κενού ασφαλείας (classification).

Με βάση τα παραπάνω, η κλάση `Impact` χρησιμοποιείται για την αναπαράσταση πληροφορίας την οποία περιέχει μια ειδοποίηση σχετικά με το αντίκτυπο που είχε μία επίθεση στο σύστημα. Μπορεί να περιέχει πληροφορία σχετικά με το αν η επίθεση ήταν

επιτυχής ή όχι, για τη σοβαρότητα της κατάστασης, καθώς και μία μετρική για το πόσο σίγουρος είναι ο αναλυτής για την εκτίμησή του.

Η κλάση `Classification` χρησιμοποιείται για την αναπαράσταση των κατηγοριοποιήσεων οι οποίες είναι δυνατόν να αφορούν σε κενά ασφαλείας (vulnerabilities-security exploits) και σε γνωστές «οικογένειες» επιθέσεων. Η εν λόγω κλάση περιέχει στιγμιότυπα τέτοιων κατηγοριοποιήσεων. Έτσι, όταν φτάνει στο σύστημα μια ειδοποίηση η οποία περιέχει σχετική πληροφορία, δημιουργείται μία συσχέτιση (μέσω μιας αντικειμενικής ιδιότητας) μεταξύ της ειδοποίησης και του σχετικού στιγμιοτύπου της `Classification` στην οντολογία. Ο διαχειριστής του συστήματος είναι υπεύθυνος για την ενημέρωση της κλάσης με νέα στιγμιότυπα σχετικά με νέες επιθέσεις, κενά ασφαλείας και απειλές οι οποίες έχουν γίνει γνωστές. Για την ενημέρωσή της, ο διαχειριστής θα πρέπει να συμβουλευτεί αξιόπιστες πηγές όπως: το SecurityFocus, το Common Vulnerabilities and Exposures και το Open Source Vulnerability Database.

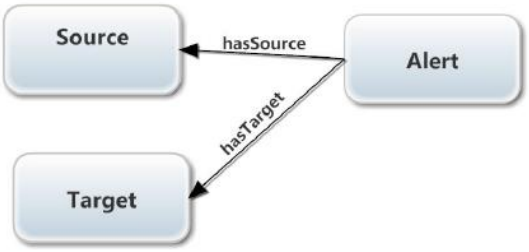


Σχήμα 19: Οι οντολογικές κλάσεις `Impact` και `Classification`.

Οι οντολογικές κλάσεις `Source` και `Target` χρησιμοποιούνται για την αναπαράσταση πληροφορίας σχετικά με το ποιος προκάλεσε το συμβάν ασφάλειας και τον στόχο της επίθεσης.

Η οντολογική κλάση `Source` αναπαριστά την πληροφορία σχετική με τις πιθανές πηγές που μπορεί να είναι πίσω από το συμβάν που προκάλεσε την ειδοποίηση. Ένα συμβάν μπορεί να έχει παραπάνω από μία πηγές, για παράδειγμα σε μία κατανεμημένη επίθεση άρνησης υπηρεσιών (distributed denial of service attack ddos).

Η οντολογική κλάση `Target` αναπαριστά την πληροφορία σχετική με τους πιθανούς στόχους της επίθεσης που προκάλεσε την ειδοποίηση. Ένα συμβάν μπορεί να έχει παραπάνω από έναν στόχους όπως για παράδειγμα στην περίπτωση ενός port sweep.



Σχήμα 20: Οι οντολογικές κλάσεις `Source` και `Target`.

Για την αναπαράσταση διευκρινιστικής πληροφορίας σχετικά με τους αναλυτές, τις πηγές και τους στόχους του συμβάντος, που προκάλεσε μια ειδοποίηση χρησιμοποιούνται οι κλάσεις `Node`, `Process`, `User` και `Service`.

Η οντολογική κλάση `Node` αναπαριστά πληροφορία σχετικά με τον κόμβο, ο οποίος είτε προκαλεί είτε δέχεται επίθεση. Συγκεκριμένα, περιέχει πληροφορία (μέσω ιδιοτήτων τύπου δεδομένων) η οποία αφορά το πού βρίσκεται ο κόμβος, το όνομα του, τη διαδικτυακή διεύθυνση κλπ. Σημειώνεται ότι η κλάση δεν περιέχει από την αρχή στιγμιότυπα καθώς κάτι τέτοιο θα απαιτούσε την ύπαρξη ενός στιγμιοτύπου για κάθε κόμβο του Διαδικτύου. Όταν μία νέα ειδοποίηση φθάνει στο σύστημα, ελέγχεται αν οι κόμβοι που ορίζει ως επιτιθέμενους (source) ή ως θύματα (target) υπάρχουν ήδη στην οντολογία και τότε εισάγονται οι κατάλληλες αντικειμενικές ιδιότητες μεταξύ κόμβου και ειδοποίησης. Αν διαπιστωθεί ότι ο κόμβος δεν υπάρχει στην οντολογία, τότε δημιουργείται ένα νέο στιγμιότυπο της κλάσης και στη συνέχεια προστίθενται οι κατάλληλες ιδιότητες.

Η οντολογική κλάση `Process` αναπαριστά πληροφορία σχετικά με τις διεργασίες οι οποίες μπορούν είτε να εκτελούνται από το θύμα της επίθεσης, ή από τον επιτιθέμενο, είτε να «τρέχουν» στον αναλυτή. Περιέχει πληροφορία (μέσω ιδιοτήτων τύπου δεδομένων) σχετική με το όνομα, το process id, το μονοπάτι αλλά και το περιβάλλον της διεργασίας.

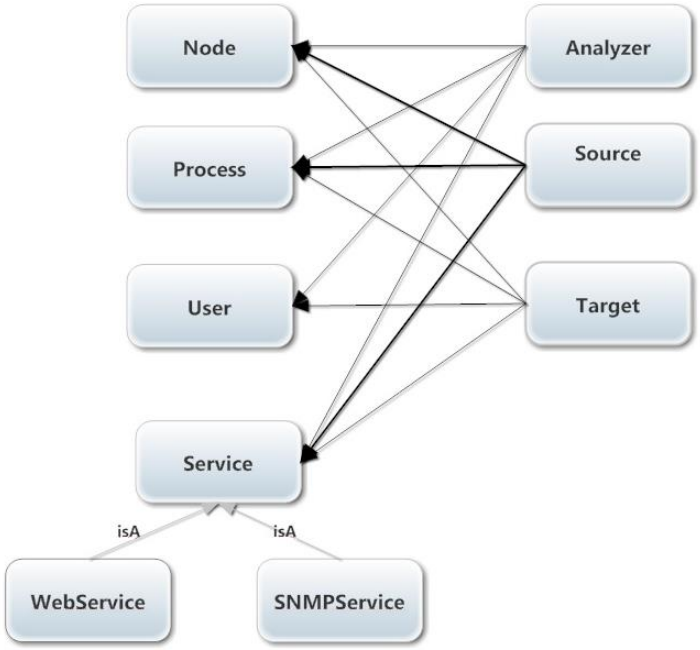
Η οντολογική κλάση `User` αναπαριστά πληροφορία σχετικά με χρήστες και μπορεί να αναφέρεται είτε σε χρήστες οι οποίοι φαίνεται να προκαλούν ένα συμβάν, είτε σε θύματα μίας επίθεσης. Ακόμη χρησιμοποιείται για αναφορά στο πρόσωπο/υπεύθυνο που χειρίζεται έναν αναλυτή. Τέλος μπορεί να αναφαίρετε είτε σε χρήστη μίας συγκεκριμένης εφαρμογής είτε στον χειριστή μίας συσκευής ή ενός λειτουργικού συστήματος.

Η οντολογική κλάση `Service` αναπαριστά πληροφορία σχετικά με τις δικτυακές υπηρεσίες σε πηγές και στόχους της επίθεσης. Συγκεκριμένα, μέσω ιδιοτήτων τύπου δεδομένων αναφέρεται στο όνομα, το port και το πρωτόκολλο της υπηρεσίας. Όταν ένα στιγμιότυπο της κλάσης `Service` συνδέεται με ένα στιγμιότυπο της κλάσης `Source` αναφέρεται σε υπηρεσία η οποία προκαλεί ή χρησιμοποιείται για την πραγματοποίηση μίας επίθεσης. Στην περίπτωση που το στιγμιότυπο της κλάσης `Service` συνδέεται με ένα στιγμιότυπο της κλάσης `Target` αυτό αναφέρεται στην υπηρεσία η οποία δέχεται την επίθεση. Και στις δύο περιπτώσεις θεωρούμε ότι οι υπηρεσίες «τρέχουν» στους κόμβους (`Node`) που ορίζονται είτε στο `Source` (επιτιθέμενη υπηρεσία) είτε στο `Target` (υπηρεσία θύμα επίθεσης). Η κλάση `Service` χωρίζεται σε δύο υποκλάσεις την `WebService` και την `SNMPService`.

Η `WebService` αναπαριστά πληροφορία που σχετίζεται με διαδικτυακή κίνηση. Μέσω data properties μπορεί να φέρει πληροφορία σχετική με το URL, το CGI (common gateway interface) και μεθόδους HTTP (HyperText Transfer Protocol).

Η `SNMPService` αναπαριστά πληροφορία αναφορικά με SNMP (Simple Network Management Protocol) κίνηση δηλαδή κίνηση σχετικά με τη διαχείριση ενός δικτύου. Περιέχει πληροφορίες σχετικά με το μοντέλο ασφαλείας, το επίπεδο ασφαλείας και τον object identifier.

Η `SNMPService` και η `WebService` είναι αμοιβαίως αποκλειόμενες υποδηλώνοντας ότι μία υπηρεσία δεν μπορεί να ανήκει και στις δύο κλάσεις ταυτόχρονα.



Σχήμα 21: Οι οντολογικές κλάσεις `Node`, `Process`, `User` και `Service`.

Η οντολογική κλάση `Address` χρησιμοποιείται για την αναπαράσταση πληροφορίας αναφορικά με τις διευθύνσεις δικτύου, υλικού και εφαρμογών. Χωρίζεται σε δεκαπέντε υποκλάσεις οι οποίες αναπαριστούν διευθύνσεις διαφορετικού είδους. Αν δεν υπήρχε ο παραπάνω διαχωρισμός, θα έπρεπε η κλάση `Address` να περιέχει στιγμιότυπα διαφορετικών τύπων το οποίο δεν ήταν επιθυμητό καθώς θα αύξανε σημαντικά την πολυπλοκότητα. Κάθε φορά που θα έφτανε μία νέα ειδοποίηση στο σύστημα και θα έπρεπε να ελεγχθεί το εάν μία διεύθυνση υπάρχει ήδη στην οντολογία, θα απαιτούνταν να διασχιστούν όλα τα περιεχόμενα της κλάσης `Address` ακόμα και διευθύνσεις διαφορετικού τύπου. Οι δεκαπέντε υποκλάσεις της `Address` είναι οι:

`Unknown`: Για την αναπαράσταση διευθύνσεων αγνώστου τύπου.

Atm: Για την αναπαράσταση ATM (Asynchronous Transfer Mode) δικτυακών διευθύνσεων.

`Email`: Για την αναπαράσταση διευθύνσεων ηλεκτρονικού ταχυδρομείου.

`LotusNotes`: Για την αναπαράσταση διευθύνσεων ηλεκτρονικού ταχυδρομείου τύπου `Lotus` Notes.

`Mac`: Για την αναπαράσταση MAC (Media Access Control) διευθύνσεων.

`Sna`: Για την αναπαράσταση SNA(IBM Shared Network Architecture) διευθύνσεων.

`Vm`: Για την αναπαράσταση διευθύνσεων ηλεκτρονικού ταχυδρομείου τύπου IBM VM («PROFS»).

`Ipv4Addr`: Για την αναπαράσταση IP έκδοσης 4 (Internet Protocol version 4) διευθύνσεων σε δεκαδική μορφή.

`Ipv4AddrHex`: Για την αναπαράσταση IP έκδοσης 4 (Protocol version 4) διευθύνσεων σε δεκαεξαδική μορφή.

`Ipv4Net`: Για την αναπαράσταση IP έκδοσης 4 (Protocol version 4) διευθύνσεων δικτύου σε δεκαδική μορφή όπου θα ορίζεται και το μέγεθος του υποδικτύου.

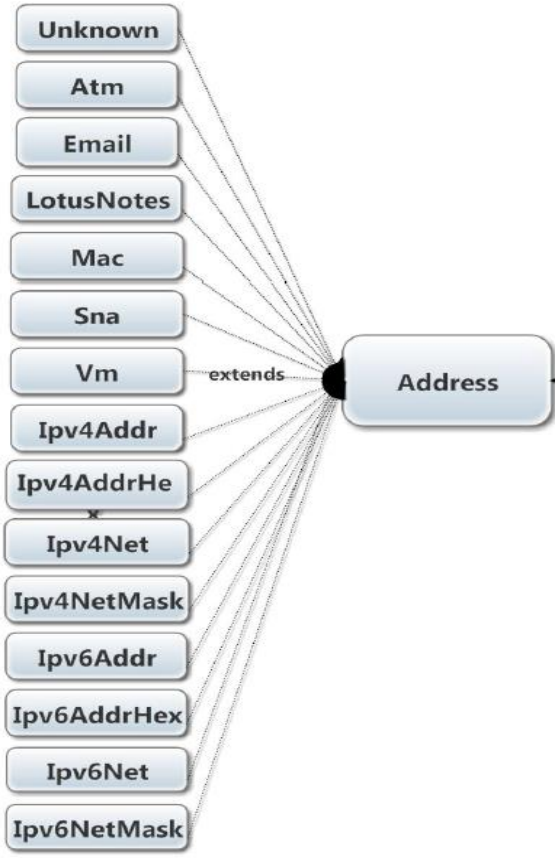
`Ipv4NetMask`: Για την αναπαράσταση της μάσκας υποδικτύου (subnet mask) σε δεκαδική μορφή μίας διεύθυνσης IP version 4.

`Ipv6Addr`: Για την αναπαράσταση IP έκδοσης 6 (Internet Protocol version 6) διευθύνσεων σε δεκαδική μορφή.

`Ipv6AddrHex`: Για την αναπαράσταση IP έκδοσης 6 (Protocol version 6) διευθύνσεων σε δεκαεξαδική μορφή.

`Ipv6Net`: Για την αναπαράσταση IP έκδοσης 6 (Protocol version 6) διευθύνσεων δικτύου σε δεκαδική μορφή στο οποίο θα ορίζεται και το μέγεθος του υποδικτύου.

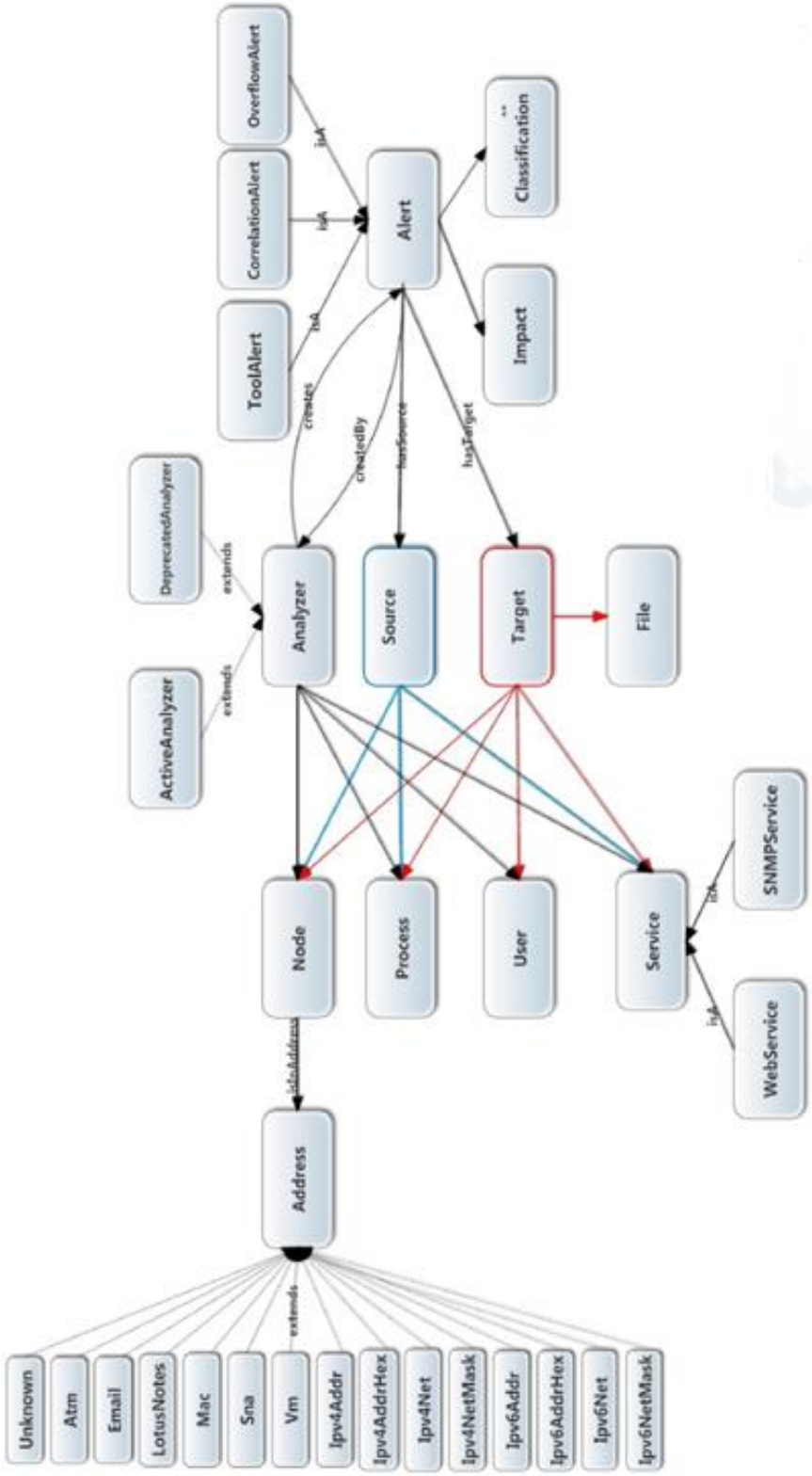
`Ipv6NetMask`: Για την αναπαράσταση IP έκδοσης 6 (Internet Protocol version 6) διευθύνσεων σε δεκαδική μορφή.



Σχήμα 22: Η οντολογική κλάση `Address` και οι υποκλάσεις της.

Η κλάση `Address` είναι αφηρημένη που σημαίνει ότι δεν είναι δυνατόν μία διεύθυνση να ανήκει στην κλάση `Address`, αλλά θα πρέπει να προσδιορίζεται ως στιγμιότυπο κάποιας από τις υποκλάσεις της. Σε περίπτωση που δεν είναι γνωστός ο τύπος της διεύθυνσης χρησιμοποιείται η υποκλάση `Unknown`.

Προφανώς όλες οι παραπάνω κλάσεις είναι αμοιβαίως αποκλειόμενες και ένα στιγμιότυπο δεν μπορεί να μέλος σε πάνω από μία απ' αυτές. Για παράδειγμα, δεν μπορεί μία διεύθυνση email ναι είναι ταυτόχρονα και διεύθυνση ATM. Στο Σχήμα 23 απεικονίζονται όλες οι κλάσεις καθώς και οι συσχετίσεις μεταξύ τους για καλύτερη κατανόηση του αναγνώστη.



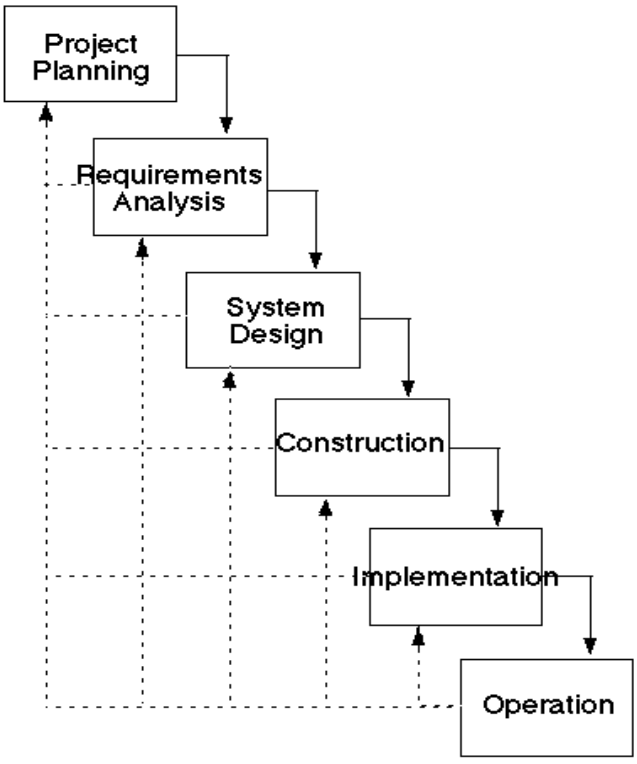
Σχήμα 23: Η αναπαράσταση του συνόλου των οντολογικών κλάσεων.

4 Ανάλυση, Σχεδίαση και Υλοποίηση Συστήματος

Στο πλαίσιο της πτυχιακής εργασίας αναπτύχθηκε ένα σύστημα λογισμικού, το οποίο δέχεται ως είσοδο συμβάντα ασφαλείας που ακολουθούν το μορφότυπο που έχει ορίσει το IDMEF και τα τοποθετεί καταλλήλως στην οντολογία που έχει αναπτυχθεί. Η ανάπτυξη του συστήματος ακολούθησε τον πλήρη κύκλο ζωής ενός έργου τεχνολογίας λογισμικού.

4.1 Το Μοντέλο Ανάπτυξης Λογισμικού που Χρησιμοποιήθηκε

Για την ανάπτυξη του λογισμικού ακολουθήθηκε το επαυξητικό μοντέλο, κατά το οποίο το λογισμικό παραδίδεται σταδιακά με επαυξήσεις (increments) οι οποίες εμπλουτίζουν τη λειτουργικότητά του, με στόχο να καλυφθεί σταδιακά η συνολική λειτουργικότητα και να μειωθεί η πιθανότητα να αποκλίνει το τελικό προϊόν από το επιθυμητό. Το συγκεκριμένο μοντέλο προσφέρει ευελιξία, αρκετά καλή διαχείριση κινδύνων και αξιολόγηση της πορείας του έργου. Επίσης το επαυξητικό μοντέλο είναι γραμμικό δηλαδή υπάρχει μία γραμμική διαδοχή των δραστηριοτήτων ανάπτυξης, γεγονός που το καθιστά πιο εύκολο στην εκτέλεση. Για τους παραπάνω λόγους και με δεδομένο ότι πρόκειται για ένα σύστημα σχετικά μικρό σε έκταση, κρίθηκε ότι το επαυξητικό μοντέλο είναι το πλέον κατάλληλο για την ανάπτυξη του συστήματος.



Σχήμα 24: Το επαυξητικό μοντέλο.

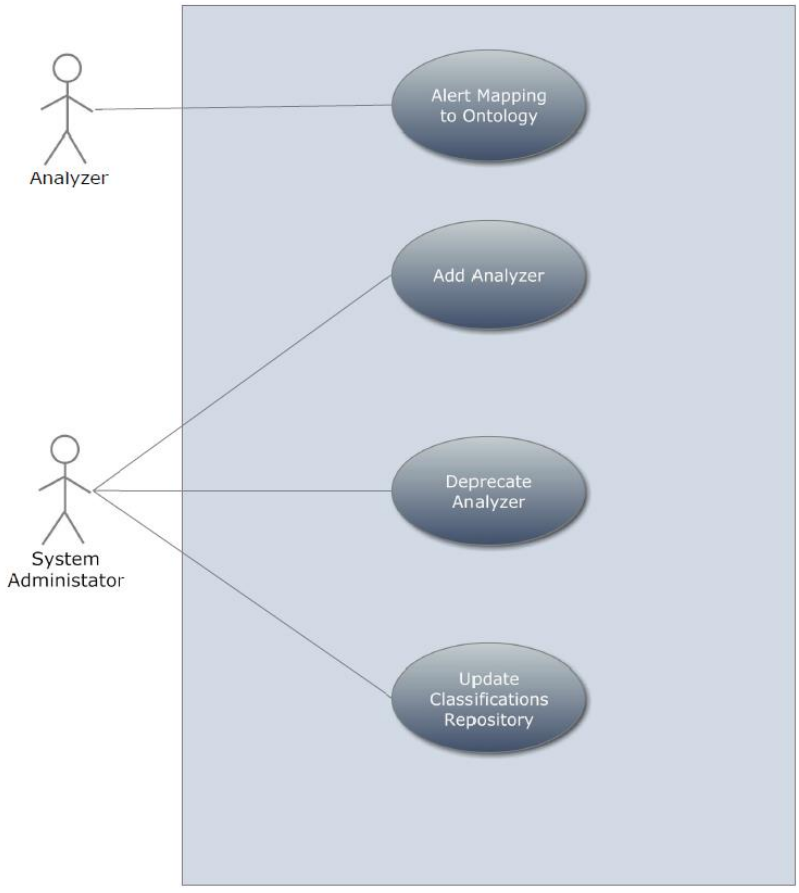
Έτσι, για την ανάπτυξη του λογισμικού, πέραν της υλοποίησης αυτής κάθε αυτής, πραγματοποιήθηκαν τα εξής στάδια:

- Προσδιορισμός περιπτώσεων χρήσης (use cases definition)
- Εκμαίευση και εκλέπτυνση απαιτήσεων (requirements elicitation)
- Σχεδιασμός συστήματος (system design)
- Έλεγχος (testing).

4.2 Περιπτώσεις Χρήσης

Οι βασικές περιπτώσεις χρήσης του συστήματος είναι οι εξής:

- Η αποθήκευση μιας ειδοποίησης που ανέφερε ένας αναλυτής στην οντολογία
- Η προσθήκη ενός νέου αναλυτή στο σύστημα
- Η κατάργηση ενός αναλυτή από το σύστημα
- Η ενημέρωση της οντολογικής κλάσης `Classification` η οποία περιέχει γνωστές αδυναμίες (vulnerabilities) και επιθέσεις συστημάτων.



Σχήμα 25: Διάγραμμα περιπτώσεων χρήσης.

Στη συνέχεια παρέχεται μία αναλυτικότερη περιγραφή των περιπτώσεων χρήσης.

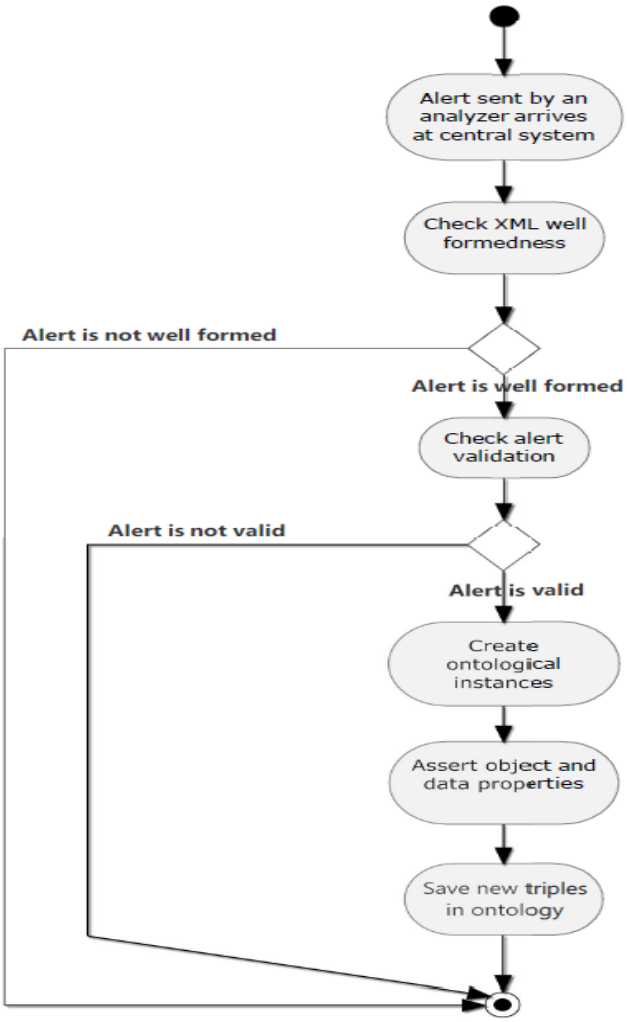
Τοποθέτηση μιας ειδοποίησης που ανέφερε ένας αναλυτής στην οντολογία

Πρωτεύων actor: Αναλυτής

Συνθήκες εισόδου: Ένας αναλυτής έχει στείλει μια ειδοποίηση στο κεντρικό σύστημα.

Συνθήκες εξόδου: Η ειδοποίηση έχει αποθηκευτεί επιτυχώς στην οντολογία.

Ένας αναλυτής ανιχνεύει ότι υπάρχει επίθεση σε κάποιον από τους κόμβους για τους οποίους είναι υπεύθυνος ή κρίνει πως ένα συμβάν είναι ύποπτο. Δημιουργεί μια ειδοποίηση η οποία ακολουθεί το μορφότυπο που ορίζει το IDMEF και τη στέλνει στον διαχειριστή στον οποίο αναφέρει. Το σύστημα ελέγχει την εγκυρότητα της ειδοποίησης καθώς και το εάν η ειδοποίηση είναι καλά ορισμένη (well formed). Στη συνέχεια, ελέγχει τα XML πεδία της ειδοποίησης και την τοποθετεί κατάλληλα στην οντολογία. Αυτό γίνεται δημιουργώντας τα κατάλληλα στιγμιότυπα των οντολογικών κλάσεων της οντολογίας που έχει δημιουργηθεί. Επίσης, προστίθενται οι κατάλληλες αντικειμενικές ιδιότητες καθώς και οι ιδιότητες τύπου δεδομένων, έτσι ώστε να περιγράφεται πλήρως η ειδοποίηση μέσω της οντολογίας. Στο Σχήμα 26 βλέπουμε το διάγραμμα δραστηριότητας (activity diagram) της τοποθέτησης μιας ειδοποίησης στην οντολογία.



Σχήμα 26: Διάγραμμα δραστηριότητας τοποθέτησης μιας ειδοποίησης στην οντολογία.

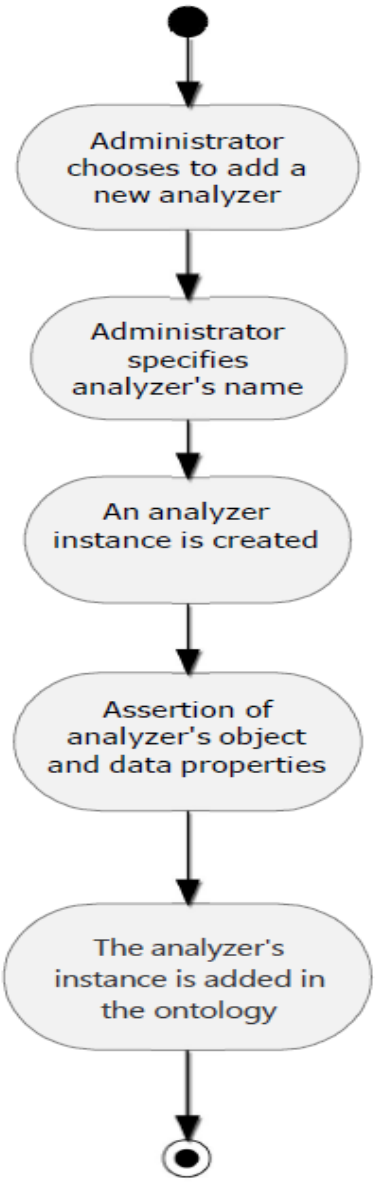
Προσθήκη ενός νέου αναλυτή στο σύστημα

Πρωτεύων actor: Διαχειριστής Συστήματος (System administrator)

Συνθήκες εισόδου: Ένας αναλυτής έχει προστεθεί στο ευρύτερο σύστημα.

Συνθήκες εξόδου: Ένα στιγμιότυπο της κλάσης `ActiveAnalyzer` έχει προστεθεί στην οντολογία.

Ο διαχειριστής του συστήματος θέλει να προσθέσει έναν νέο αναλυτή στο σύστημα. Δημιουργεί λοιπόν ένα νέο στιγμιότυπο της οντολογικής κλάσης `ActiveAnalyzer` και προσδιορίζει τις αντικειμενικές ιδιότητες καθώς και τις ιδιότητες τύπου δεδομένων που χρειάζονται (π.χ. προσδιορισμός `Address`). Το σύστημα επιστρέφει μήνυμα επιτυχίας για την ορθή εισαγωγή του νέου αναλυτή στην οντολογία. Στο Σχήμα 27 βλέπουμε το διάγραμμα δραστηριότητας της προσθήκης νέου αναλυτή.



Σχήμα 27: Διάγραμμα δραστηριότητας προσθήκης ενός αναλυτή.

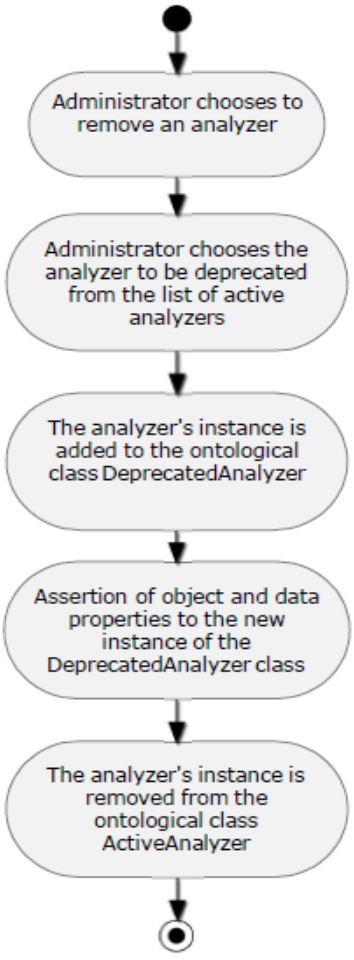
Κατάργηση ενός αναλυτή από το σύστημα

Πρωτεύων actor: Διαχειριστής Συστήματος (System administrator)

Συνθήκες εισόδου: Ένας αναλυτής έχει αφαιρεθεί από το ευρύτερο σύστημα.

Συνθήκες εξόδου: Ένα στιγμιότυπο της κλάσης `ActiveAnalyzer` έχει καταργηθεί και ένα στιγμιότυπο της κλάσης `DeprecatedAnalyzer` έχει προστεθεί στην οντολογία.

Ο διαχειριστής του συστήματος αποφασίζει ότι θέλει να καταργήσει έναν αναλυτή από το σύστημα. Ζητάει λοιπόν από το σύστημα να του επιστρέψει όλα τα στιγμιότυπα της κλάσης `ActiveAnalyzer` και επιλέγει αυτόν που επιθυμεί να καταργήσει. Το σύστημα δημιουργεί ένα νέο στιγμιότυπο της κλάσης `DeprecatedAnalyzer` στο οποίο προστίθενται οι αντικειμενικές ιδιότητες καθώς και οι ιδιότητες τύπου δεδομένων του στιγμιοτύπου της `ActiveAnalyzer` που επιλέχθηκε. Για παράδειγμα, αν το στιγμιότυπο προς κατάργηση σχετίζεται μέσω μιας αντικειμενικής ιδιότητας με το στιγμιότυπο της κλάσης `Address` 192.168.1.254 τότε και το στιγμιότυπο της `DeprecatedAnalyzer` που δημιουργήθηκε θα συσχετιστεί μέσω της ίδιας αντικειμενικής ιδιότητας με το στιγμιότυπο 192.168.1.254. Στη συνέχεια, το σύστημα καταργεί το στιγμιότυπο που αντιστοιχούσε στον ενεργό αναλυτή. Στο Σχήμα 28 βλέπουμε το διάγραμμα δραστηριότητας της κατάργησης ενός αναλυτή.



Σχήμα 28: Διάγραμμα δραστηριότητας κατάργησης ενός αναλυτή.

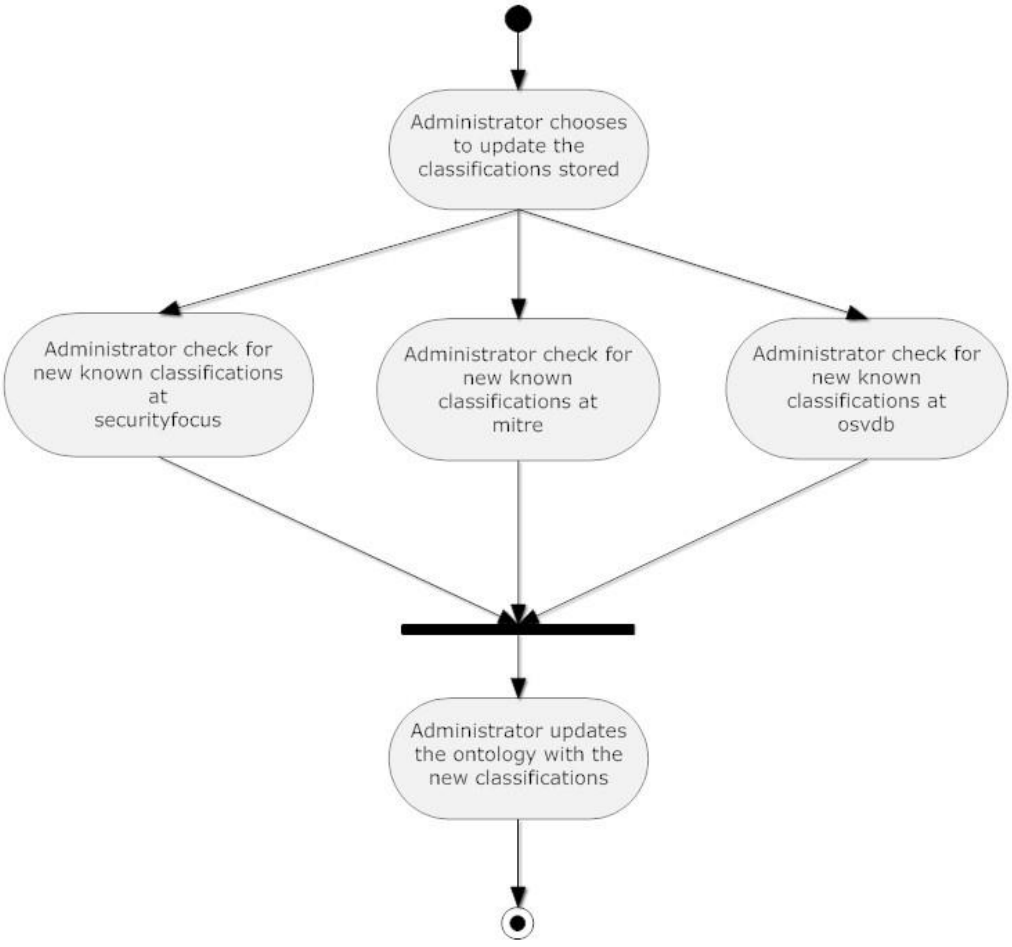
Ενημέρωση της οντολογικής κλάσης `Classification`

Πρωτεύων actor: Διαχειριστής Συστήματος (System administrator)

Συνθήκες εισόδου: Νέες επιθέσεις και αδυναμίες έχουν γίνει γνωστές και έχουν δημοσιευθεί.

Συνθήκες εξόδου: Οι νέες επιθέσεις έχουν καταχωρηθεί στην οντολογία μέσω της οντολογικής κλάσης `Classification`.

Ο διαχειριστής του συστήματος επιθυμεί να ενημερώσει το σύστημα για νέες απειλές, επιθέσεις και κενά ασφαλείας που έχουν γίνει γνωστά. Στην οντολογική κλάση `Classification` υπάρχουν στιγμιότυπα γνωστών επιθέσεων. Ο διαχειριστής αναλαμβάνει να ενημερώσει το σύστημα παίρνοντας πληροφορίες από πηγές όπως είναι η βάση αναγνώρισης αδυναμιών SecurityFocus («Bugtraq») (www.securityfocus.com/bid), το λεξικό γνωστών κενών ασφαλείας Common Vulnerabilities and Exposures (CVE) (www.cve.mitre.org) και η ανοιχτού κώδικα βάση αδυναμιών Open Source Vulnerability Databse www.osvdb.org. Ο διαχειριστής αφού εντοπίσει κάποια νέα απειλή ελέγχοντας τις παραπάνω σελίδες, για την οποία θεωρεί πως πρέπει να υπάρχει στο σύστημα, δημιουργεί ένα νέο στιγμιότυπο της κλάσης `Classification` στην οντολογία. Στο Σχήμα 29 βλέπουμε το διάγραμμα δραστηριότητας.



Σχήμα 29: Διάγραμμα δραστηριότητας για την ενημέρωση του συστήματος σχετικά με νέες επιθέσεις και απειλές.

4.3 Απαιτήσεις Συστήματος

Από τις παραπάνω περιπτώσεις χρήσης προέκυψαν οι παρακάτω απαιτήσεις:

1. Απαιτείται όλοι οι αναλυτές, ενεργοί και μη, να είναι αποθηκευμένοι στην οντολογία μαζί με τα στοιχεία που τους χαρακτηρίζουν (μέσω ιδιοτήτων).

2. Οι αναλυτές θα πρέπει να είναι μοναδικά αναγνωρίσιμοι. Αυτό με βάση το standard επιτυγχάνεται μέσω των ζευγών alertident – analyzerid τα οποία και θα πρέπει να είναι μοναδικά.

3. Θα πρέπει να υπάρχει δυνατότητα διαχείρισης των αναλυτών, όπως προσθήκη και κατάργηση ενός αναλυτή και αυτόματη ενημέρωση της οντολογίας.

4. Θα πρέπει οι ειδοποιήσεις που φτάνουν στο σύστημα κατόπιν επεξεργασίας και έλεγχο των XML ετικετών τους να αποθηκεύονται κατάλληλα στην οντολογία μέσω της δημιουργίας στιγμιοτύπων και της προσθήκης των κατάλληλων ιδιοτήτων.

5. Απαιτείται η ύπαρξη αποθηκευμένων γνωστών επιθέσεων και κενών ασφαλείας ώστε να μπορεί να πραγματοποιηθεί η κατηγοριοποίηση και η ταξινόμηση των ειδοποιήσεων. Ο διαχειριστής του συστήματος θα πρέπει να μπορεί να ενημερώνει τα αποθηκευμένα λήμματα με βάση τις πηγές που έχουν οριστεί (securityfocus, mitre, osvdb).

6. Το σύστημα θα πρέπει να απορρίπτει οποιοδήποτε ειδοποίηση φαίνεται να έχει προέλθει είτε από αναλυτή ο οποίος δεν υπάρχει αποθηκευμένος στο σύστημα είτε υπάρχει αλλά έχει οριστεί ως ανενεργός (deprecated).

7. Το σύστημα θα πρέπει να απορρίπτει ως μη έγκυρη κάθε ειδοποίηση που δεν φέρει πληροφορία τουλάχιστον για την ώρα που δημιουργήθηκε και από ποιο αναλυτή και δεν φέρει τα μοναδικά αναγνωριστικά.

8. Ο αναλυτής θα πρέπει να έχει τη δυνατότητα να συσχετίζει ειδοποιήσεις μεταξύ τους, όσον αφορά το malware/attack tool που χρησιμοποιήθηκε. Καθώς και την ακριβή ενέργεια που επιχειρήθηκε μέσω του κακόβολου λογισμικού.

9. Όταν στο σύστημα φτάνει ένα `CorrelationAlert` θα πρέπει οι ειδοποιήσεις τις οποίες συσχετίζει να αποκτήσουν συσχέτιση και σε επίπεδο οντολογίας μέσω της προσθήκης των κατάλληλων ιδιοτήτων.

10. Όταν στο σύστημα φτάνει ένα `ToolAlert` το οποίο προσδιορίζει το εργαλείο κακόβουλου λογισμικού (malicious software) όπως για παράδειγμα ένας δούρειος ίππος (Trojan horse) με χρήση του οποίου πραγματοποιήθηκαν επιθέσεις για τις οποίες είχαν αποσταλεί προηγουμένως οι αντίστοιχες ειδοποιήσεις, το σύστημα θα πρέπει να προσθέτει και στην οντολογία τις κατάλληλες ιδιότητες στα στιγμιότυπα των ειδοποιήσεων.

11. Όλες οι οντότητες του συστήματος θα πρέπει να είναι χρονικά συγχρονισμένες, ώστε να έχουν νόημα οι ιδιότητες αντικειμένων που ορίζουν τιμές για την ώρα δημιουργίας της ειδοποίησης (CreateTime), την ώρα ανίχνευσής του (DetectTime) καθώς και την τοπική ώρα του αναλυτή (AnalyzerTime).

12. Το σύστημα θα πρέπει να πραγματοποιεί έλεγχο για το αν οι ειδοποιήσεις που έλαβε είναι καλώς ορισμένες (well formed) όσον αφορά την XML, αν είναι δηλαδή σε συμφωνία με το XML Schema το οποίο έχει οριστεί αλλά και το αν είναι έγκυρες (valid), αν δηλαδή πληρούν όλους τους περιορισμούς που έχουν ορισθεί όσον αφορά τις τιμές των πεδίων τους. Σε περίπτωση που μια ειδοποίηση δεν είναι έγκυρη αλλά ακολουθεί τον μορφότυπο που έχει ορίσει το IDMEF δεν θα πρέπει να απορρίπτεται αλλά θα πρέπει να υπόκειται σε επεξεργασία καθώς κάτι τέτοιο κρίνεται ύποπτο.

13. Θα πρέπει να υπάρχει μηχανισμός αμοιβαίας αυθεντικοποίησης (mutual authentication mechanism) μεταξύ των αναλυτών και του διαχειριστή στον οποίον αναφέρουν. Αν μια ειδοποίηση προέρχεται από έναν αναλυτή ο οποίος δεν έχει αυθεντικοποιηθεί από το σύστημα, τότε η ειδοποίηση θα πρέπει να απορρίπτεται.

14. Τα μηνύματα που στέλνονται μεταξύ των οντοτήτων του συστήματος θα πρέπει να είναι κρυπτογραφημένα. Θα πρέπει επιπλέον να παρέχεται ένα πλήθος διαφορετικών κρυπτογραφήσεων για λόγους προσαρμοστικότητας.

15. Κάθε μήνυμα του IDEMF θα πρέπει να είναι μοναδικά αναγνωρίσιμο.

16. Το σύστημα θα πρέπει να κρατάει αρχείο καταγραφής (log file) στο οποίο θα αποθηκεύονται όλες οι αλλαγές που πραγματοποιούνται στην οντολογία.

17. Το σύστημα θα πρέπει να διαθέτει γραφικό περιβάλλον (Graphical User Interface GUI) στο οποίο θα παρουσιάζονται τα συμβάντα ασφαλείας, δίνοντας έτσι στον διαχειριστή τη δυνατότητα να παρακολουθεί ταυτόχρονα όλο το ευρύτερο σύστημα σε μία μόνο οθόνη, διευκολύνοντας παράλληλα την ενημέρωσή του όποτε αυτό κρίνεται αναγκαίο. Επίσης, θα πρέπει να υπάρχει κατάλληλο γραφικό περιβάλλον για την ευκολότερη διαχείριση των αναλυτών καθώς και ολόκληρης της οντολογίας.

4.4 Σχεδιασμός Συστήματος

Μέσω της διαδικασίας του σχεδιασμού συστήματος το μοντέλο ανάλυσης μετασχηματίστηκε σε μοντέλο σχεδιασμού συστήματος. Κατά το στάδιο του σχεδιασμού επιλέχθηκαν στρατηγικές για:

- την ανάπτυξη του συστήματος
- την επιλογή των τεχνολογιών υλικού και λογισμικού που θα χρησιμοποιηθούν
- το μοντέλο δεδομένων (data model)
- τη διαχείριση των μόνιμα αποθηκευμένων δεδομένων
- τη συνολική ροή ελέγχου
- το χειρισμό των οριακών περιπτώσεων.

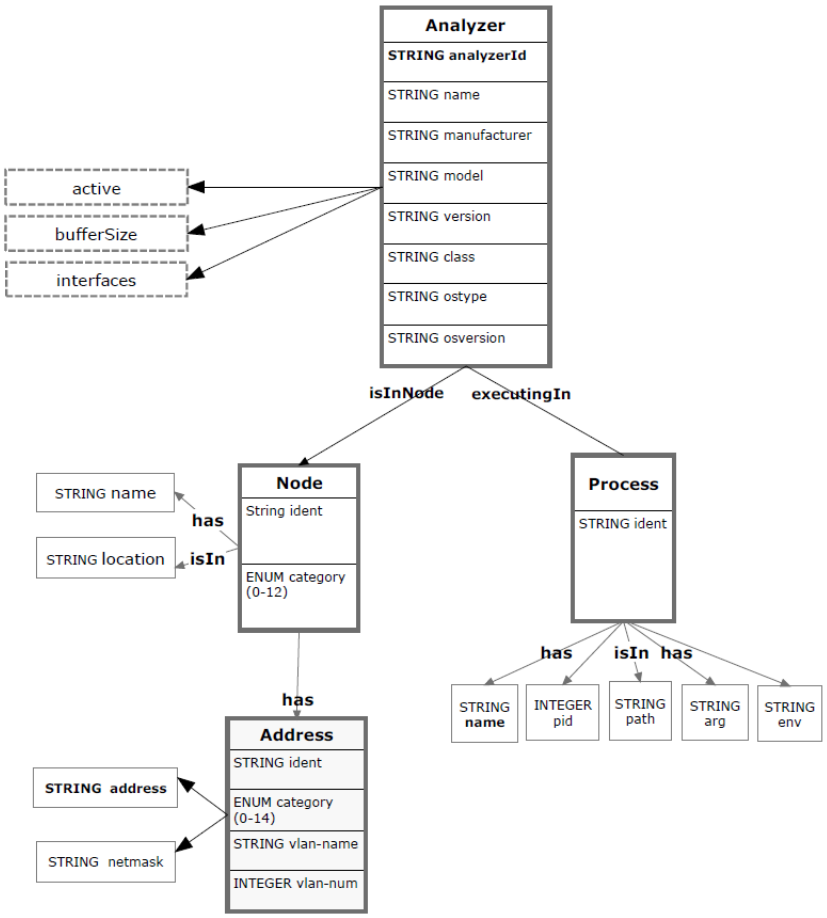
Η διαδικασία του σχεδιασμού έγινε ακολουθώντας μία top-down προσέγγιση. Έτσι, πρώτο στάδιο αποτέλεσε η αρχική αποσύνθεση του συστήματος, κατά το οποίο το σύστημα αποσυντέθηκε σε μικρότερα τμήματα με βάση τις περιπτώσεις χρήσης που ορίστηκαν στο στάδιο της ανάλυσης. Ακολούθησε η περαιτέρω αποσύνθεση σε

υποσυστήματα, έως ότου καλύφθηκαν όλοι οι σχεδιαστικοί στόχοι. Η αποσύνθεση του συστήματος σε μικρότερα υποσυστήματα εξυπηρέτησε την καλύτερη διαχείριση της πολυπλοκότητας, καθώς κάθε υποσύστημα αναπτύχθηκε αυτόνομα.

Το αποτέλεσμα είναι ένα μοντέλο το οποίο περιλαμβάνει την αρχιτεκτονική του συστήματος καθώς και την αποσύνθεση σε υποσυστήματα και μία σαφή περιγραφή των στρατηγικών.

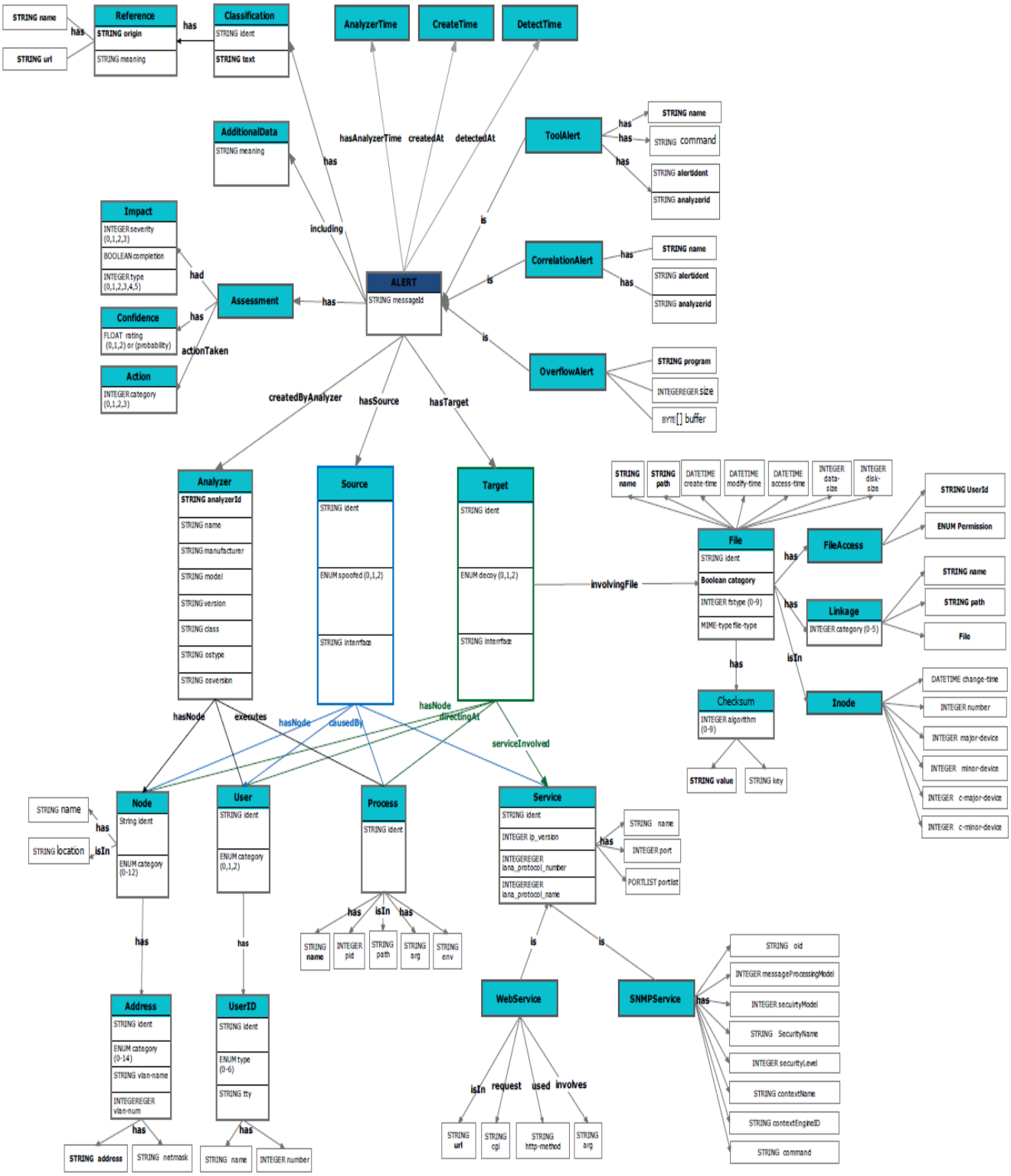
4.5 Το Μοντέλο Δεδομένων

Το μοντέλο δεδομένων αναπαριστά τα δεδομένα που διαχειρίζεται το σύστημα. Η δημιουργία του κρίθηκε πολύ βασική για την ανάπτυξη του λογισμικού, καθώς δομεί τα δεδομένα παρέχοντας συσχετίσεις μεταξύ τους και περιγράφοντας τον τύπο τους. Η ανάπτυξη αυτού του μοντέλου βοηθά στη συμβατότητα (compatibility) τόσο μεταξύ των υποσυστημάτων του λογισμικού όσο και του υπό ανάπτυξη συστήματος με άλλα συστήματα, όπως για παράδειγμα προϋπάρχοντα συστήματα (legacy systems). Είναι από τα σημαντικότερα μοντέλα για συστήματα που διαχειρίζονται μεγάλο όγκο δεδομένων, όπως στην περίπτωσή μας. Για τις ανάγκες της εργασίας κρίθηκε σκόπιμη η δημιουργία δύο μοντέλων δεδομένων. Το πρώτο μοντέλο που εξυπηρετεί τη διαχείριση των αναλυτών και αντιμετωπίζει του αναλυτές ως υποσυστήματα (components) απεικονίζεται στο Σχήμα 30.



Σχήμα 30: Το μοντέλο δεδομένων σχετικά με τη διαχείριση των αναλυτών.

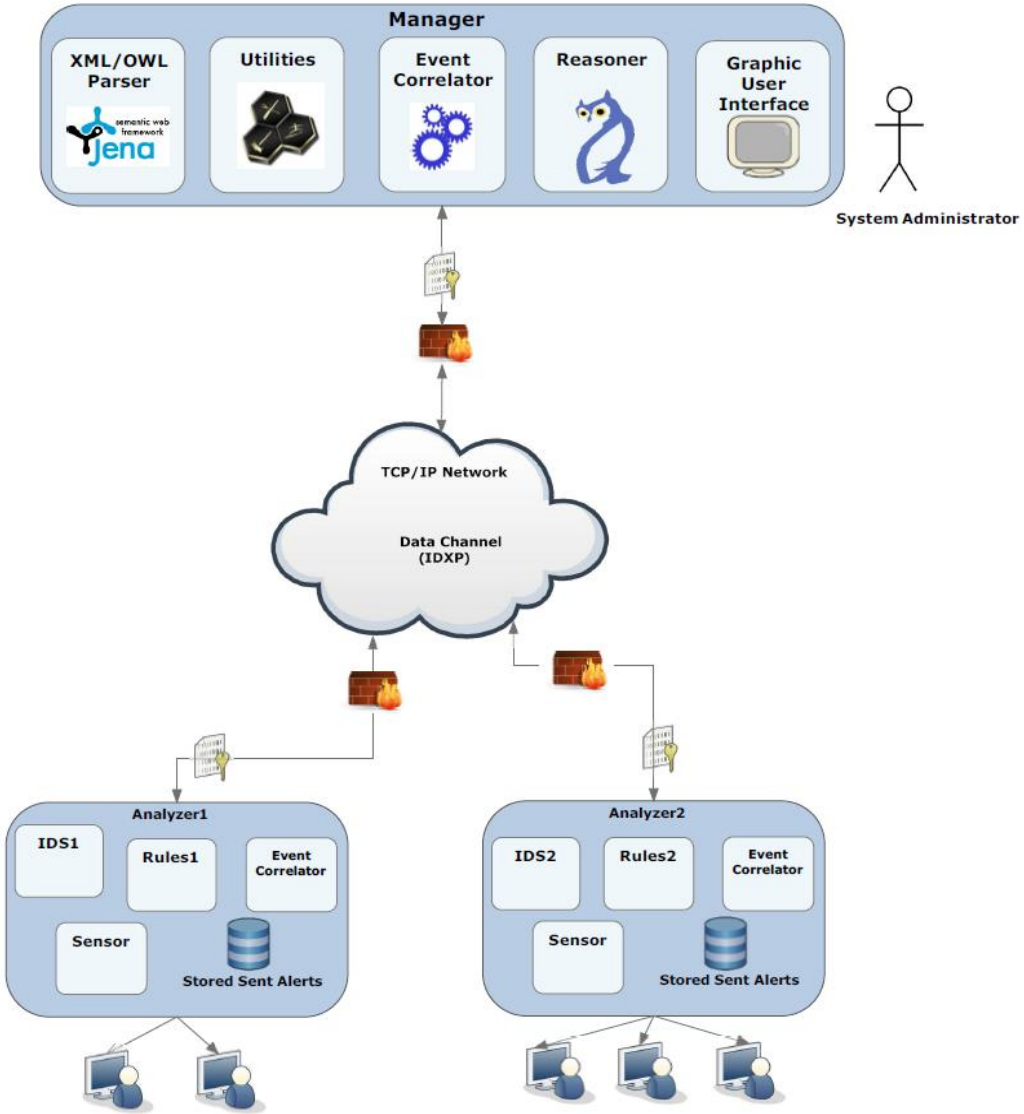
Το δεύτερο μοντέλο δεδομένων δεν αντιμετωπίζει τους αναλυτές ως υποσυστήματα, αλλά ως δεδομένα, και περιέχει όλα τα πιθανά δεδομένα με βάση την πληροφορία που μπορεί να φέρει μια ειδοποίηση. Οι μεγάλες εκφραστικές δυνατότητες που παρέχει το standard καθιστούν αναπόφευκτα το μοντέλο αρκετά πολύπλοκο και εκτενές. Το Σχήμα 31 απεικονίζει το εν λόγω μοντέλο.



Σχήμα 31: Το μοντέλο δεδομένων ενός συμβάντος ασφάλειας που ακολουθεί τον μορφότυπο του IDMEF.

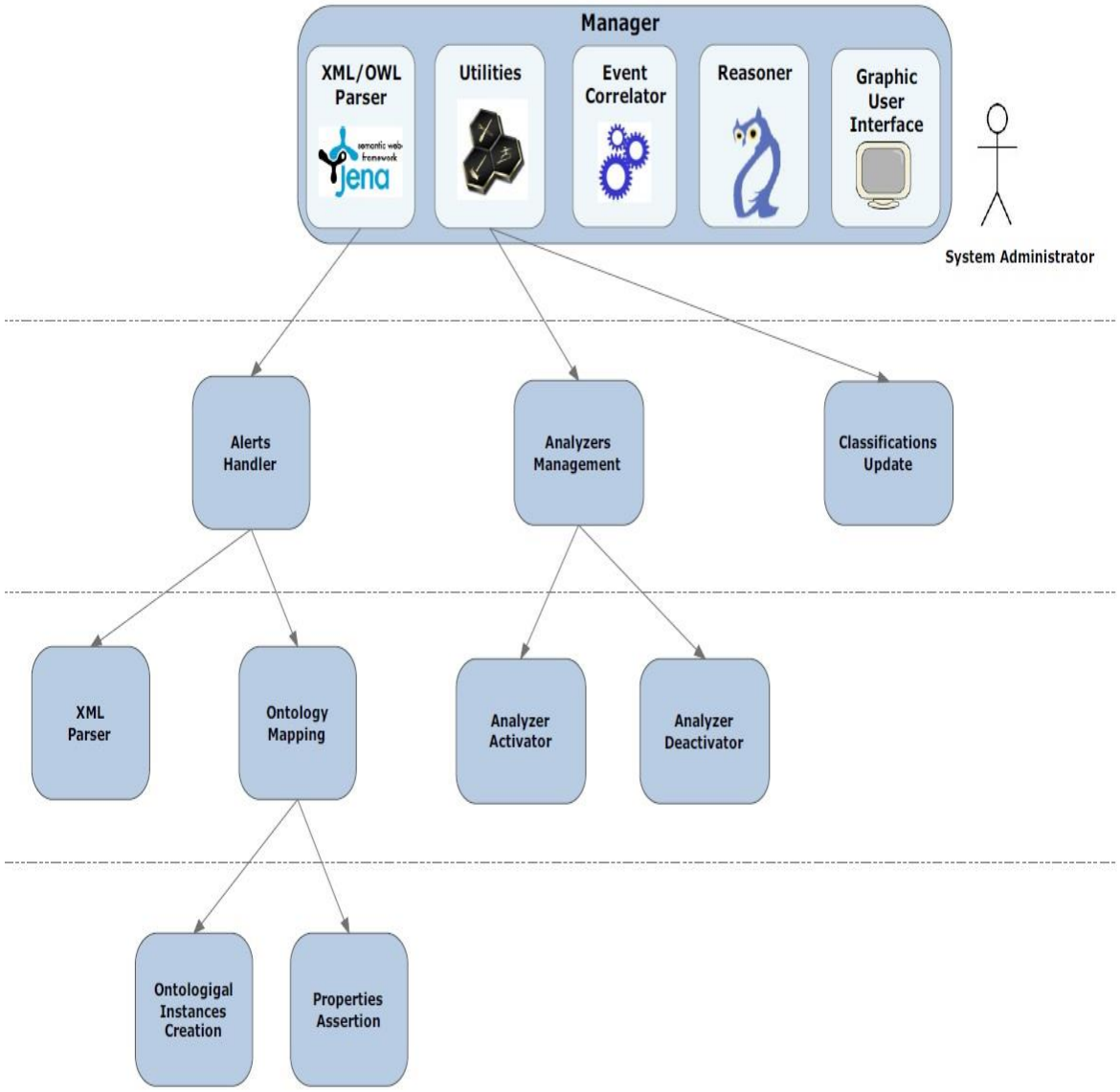
4.6 Αρχιτεκτονική Συστήματος

Ιδιαίτερη έμφαση δόθηκε στην αποσύνθεση του συστήματος σε μικρότερα υποσυστήματα. Ένα υποσύστημα είναι ένα αντικαταστάσιμο τμήμα του συστήματος με καλά καθορισμένες διεπαφές που ενθυλακώνει την κατάσταση και τη συμπεριφορά των κλάσεων που περιέχει. Η αποσύνθεση σε υποσυστήματα επιτρέπει την παράλληλη ανάπτυξη ολόκληρου του συστήματος. Η αποσύνθεση έγινε με τρόπο top-down, δηλαδή αρχικά ορίστηκαν τα υποσυστήματα σε υψηλό επίπεδο και στη συνέχεια μελετήθηκε η συμπεριφορά και οι λειτουργίες του καθενός σε βάθος. Τα υποσυστήματα ορίστηκαν βάσει των υπηρεσιών και λειτουργιών που προσφέρουν, δηλαδή η αποσύνθεση που έγινε ήταν λειτουργική. Χάρη στη λειτουργική αποσύνθεση (functional decomposition) επιτεύχθηκε τόσο η χαλαρή σύζευξη (loose coupling) μεταξύ των υποσυστημάτων, όσο και η υψηλή συνεκτικότητα (high cohesion) του κάθε υποσυστήματος. Στο Σχήμα 32 βλέπουμε την αρχιτεκτονική ολόκληρου του συστήματος σε υψηλό επίπεδο.



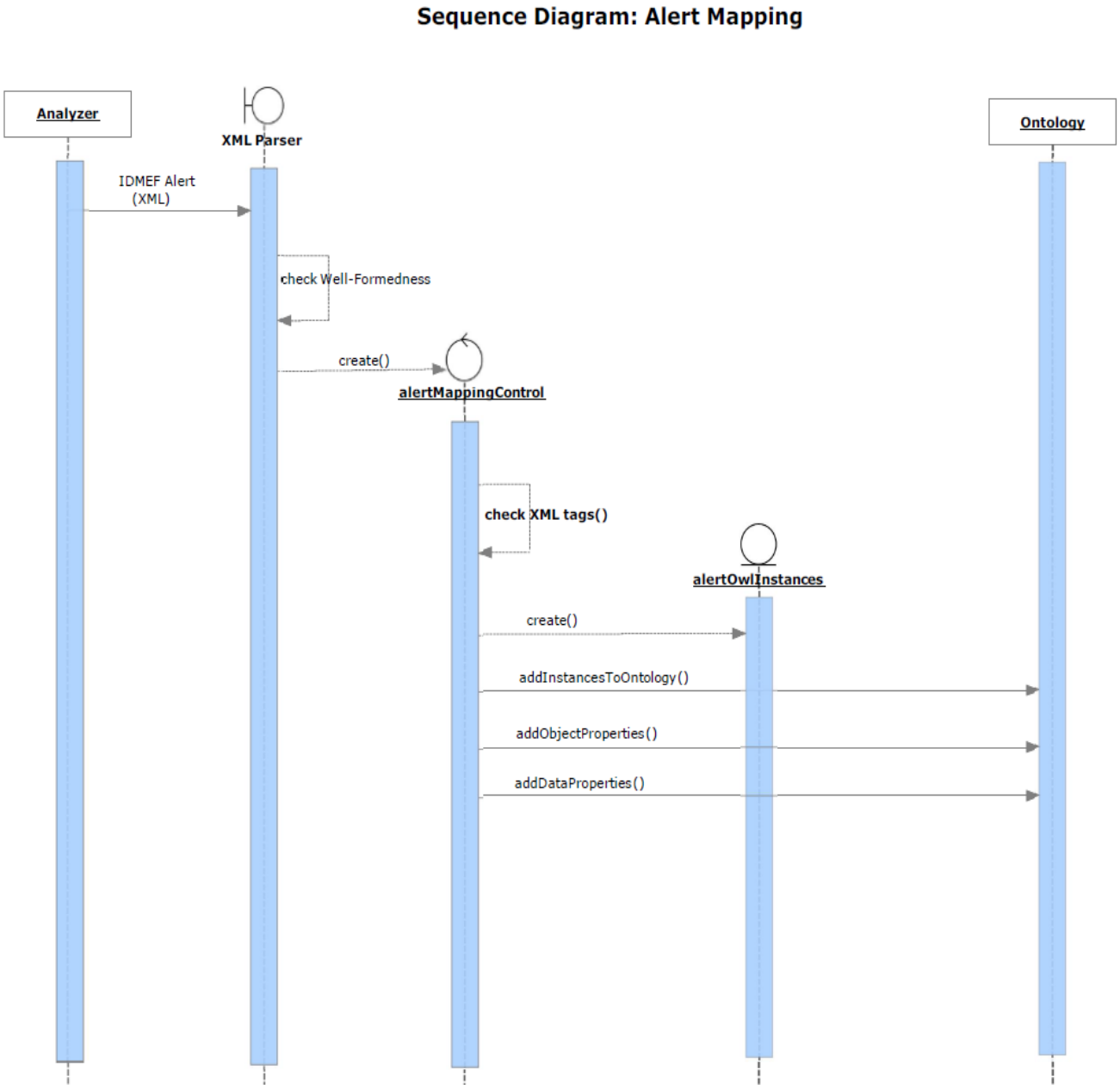
Σχήμα 32: Η αρχιτεκτονική του συστήματος σε υψηλό επίπεδο.

Στη συνέχεια ακολούθησε η ανάλυση των παραπάνω υποσυστημάτων. Πιο συγκεκριμένα, πραγματοποιήθηκε η ανάλυση του XML/OWL Parser που αποτελεί την «καρδιά» ολόκληρου του συστήματος, καθώς αναλαμβάνει να διαβάσει τις ειδοποιήσεις IDMEF που είναι εκφρασμένες σε XML και ακολουθούν το XSD του IDMEF και στη συνέχεια να τις αποθηκεύσει στην οντολογία. Επίσης, αναλύθηκε το υποσύστημα `Utilities` που παρέχει λειτουργικότητες για τη διαχείριση των αναλυτών, την ενημέρωση των `Classifications` καθώς και για την αλληλεπίδραση με την οντολογία (ομαδοποίηση ειδοποιήσεων, επισκόπηση των επιθέσεων που πραγματοποιήθηκαν μέσα σε ένα συγκεκριμένο χρονικό διάστημα κλπ).

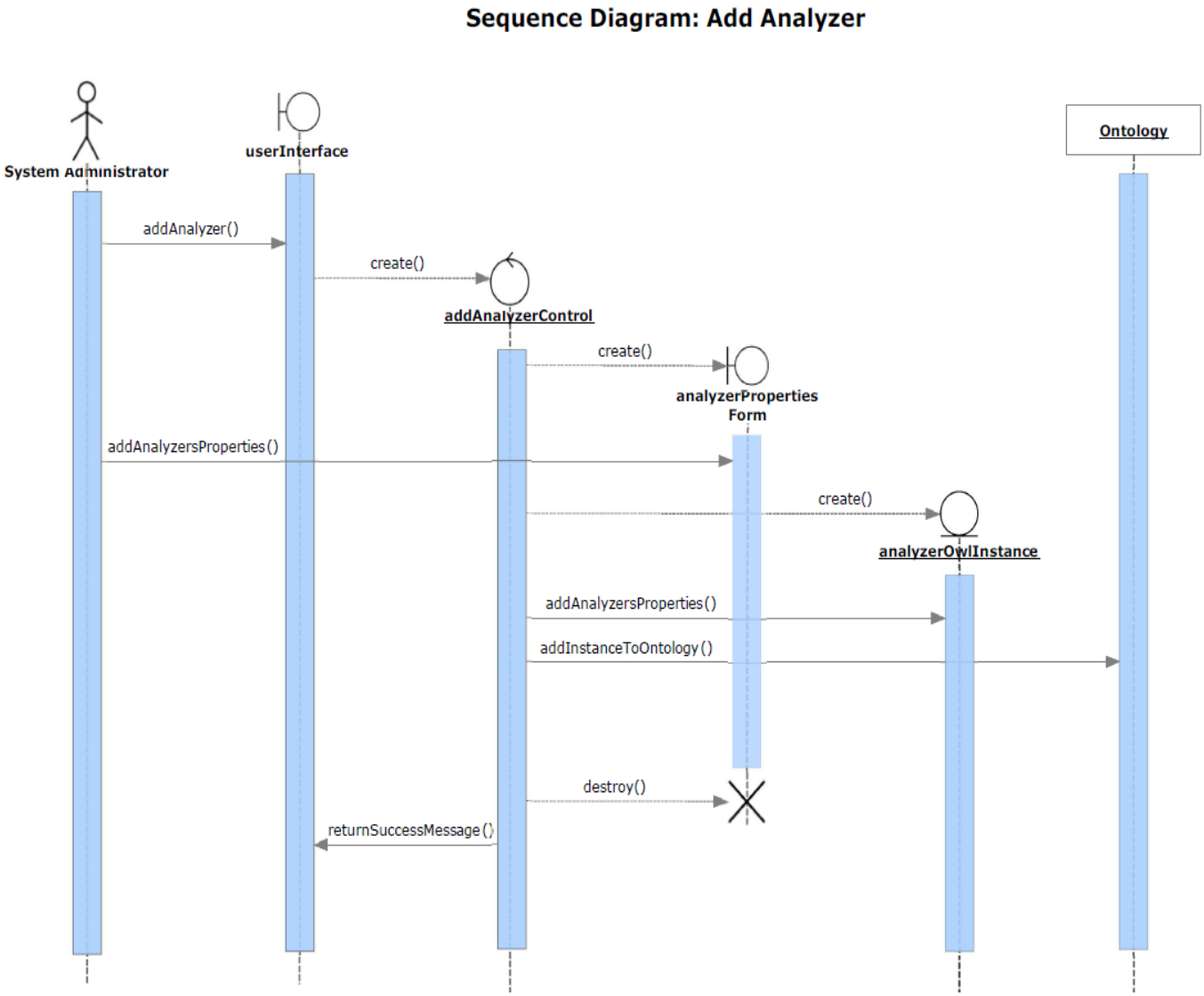


Σχήμα 33: Η αποσύνθεση των υποσυστημάτων XML/OWL Parser και Utilities.

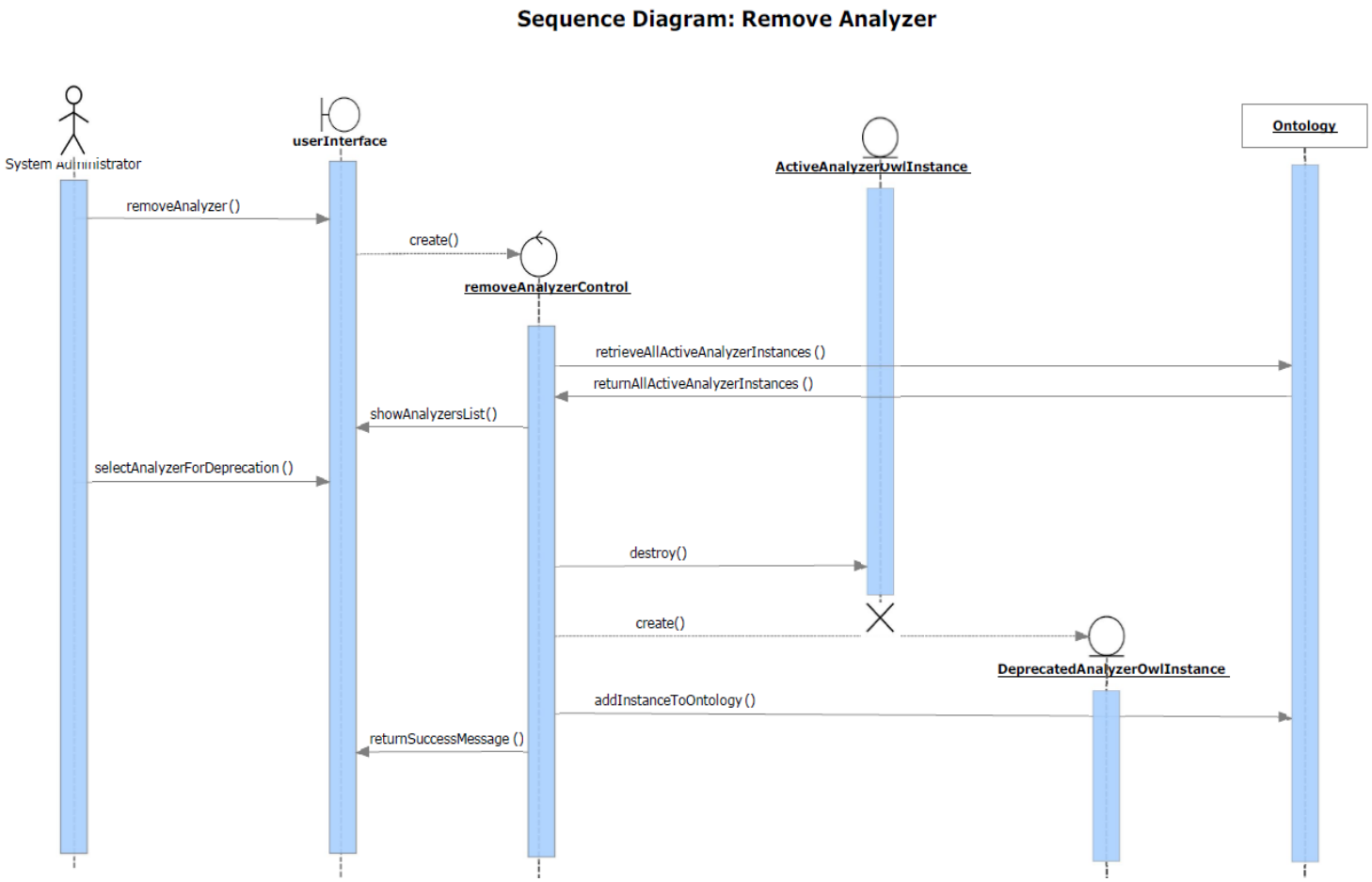
4.7 Διαγράμματα Ακολουθίας



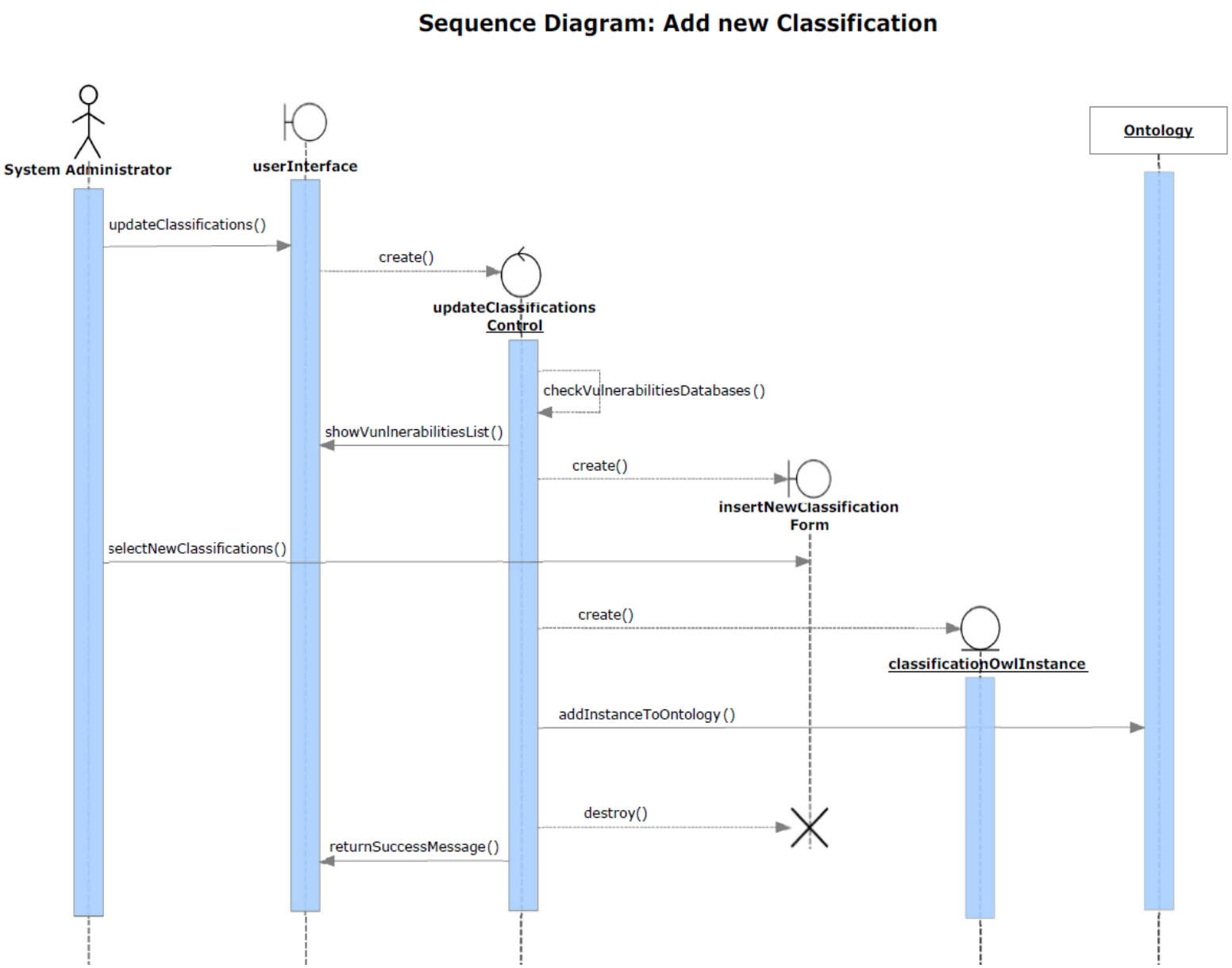
Σχήμα 34: Διάγραμμα ακολουθίας για την τοποθέτηση ενός συμβάν ασφάλειας στην οντολογία.



Σχήμα 35: Διάγραμμα ακολουθίας για την προσθήκη ενός νέου αναλυτή στο σύστημα.



Σχήμα 36: Διάγραμμα ακολουθίας για την κατάργηση ενός νέου αναλυτή από το σύστημα.



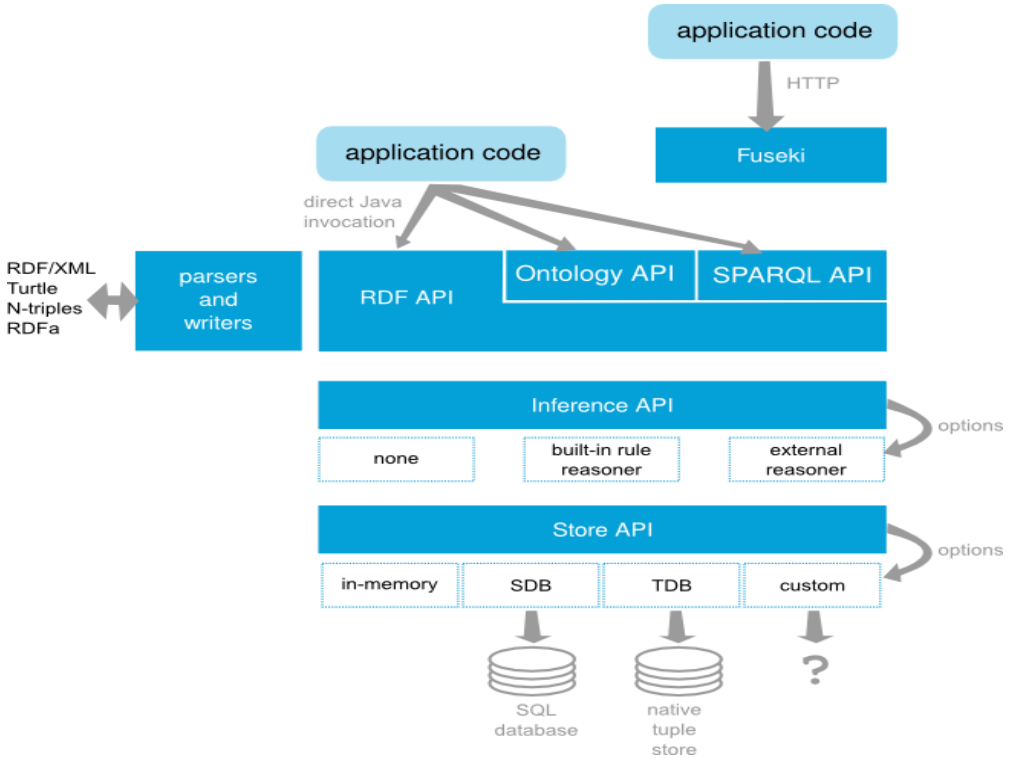
Σχήμα 37: Διάγραμμα ακολουθίας για την ενημέρωση του συστήματος σχετικά με νέες απειλές και επιθέσεις που έχουν γίνει γνωστές.

4.8 Υλοποίηση της Βιβλιοθήκης Λογισμικού

Στο πλαίσιο της διπλωματικής εργασίας σχεδιάστηκε και αναπτύχθηκε βιβλιοθήκη λογισμικού η οποία αναλαμβάνει την αποθήκευση των συμβάντων ασφαλείας που ακολουθούν τον μορφότυπο που ορίζει το IDMEF στην οντολογία. Η ανάπτυξη της βιβλιοθήκης πραγματοποιήθηκε στη γλώσσα προγραμματισμού Java. Πιο συγκεκριμένα, το λογισμικό που αναπτύχθηκε κάνει χρήση της βιβλιοθήκης λογισμικού Jena.

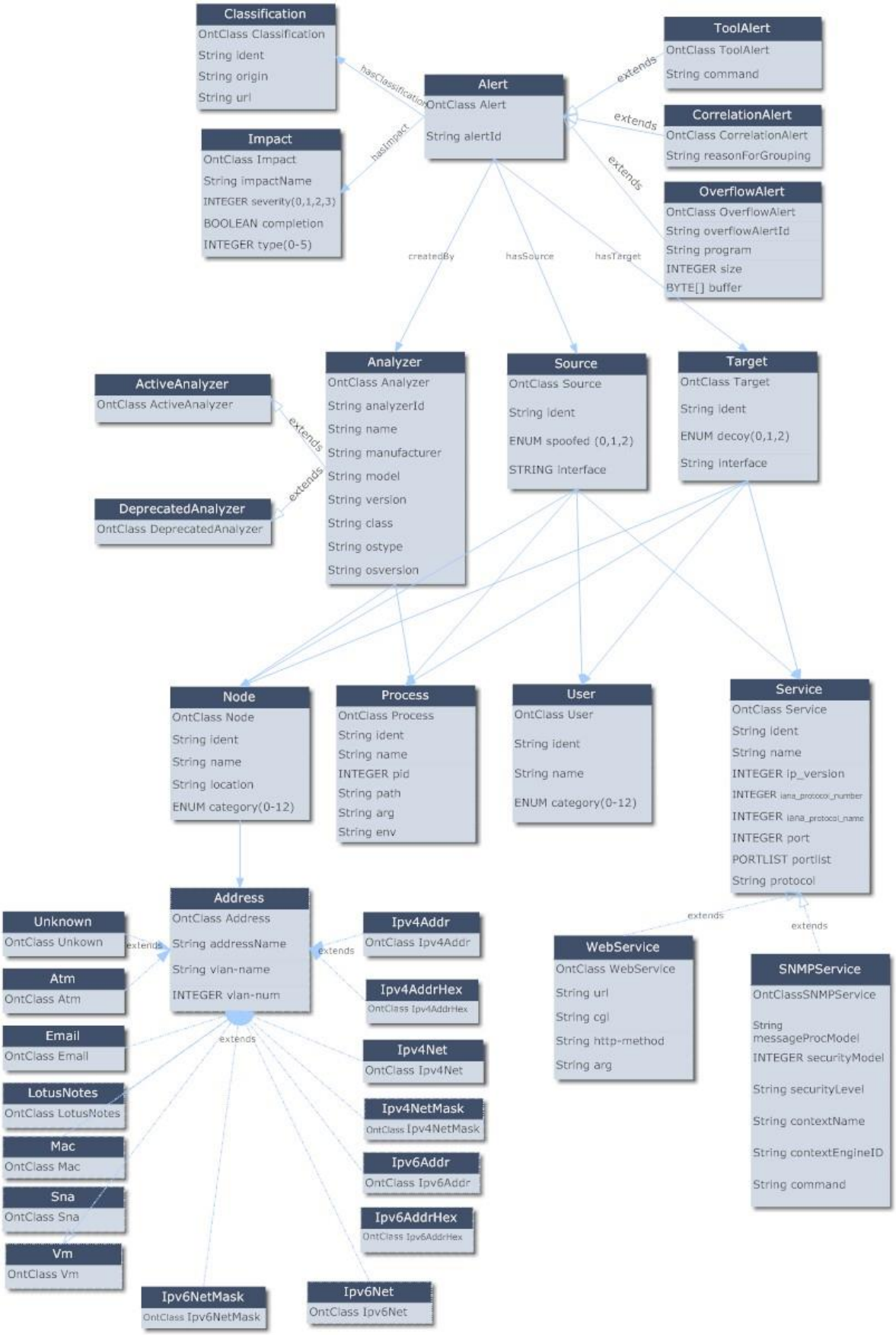
Η βιβλιοθήκη Jena είναι γραμμένη σε Java και έχει χρησιμοποιηθεί ευρέως από την κοινότητα της πληροφορικής για την ανάπτυξη εφαρμογών οι οποίες αλληλεπιδρούν με οντολογίες ή RDF σχήματα. Παρέχει μία προγραμματιστική διεπαφή για την ανάπτυξη

εφαρμογών ανεξαρτήτως της γλώσσας στην οποία έχει δημιουργηθεί η οντολογία. Οι γλώσσες στις οποίες μπορεί να είναι γραμμένη μία οντολογία με σειρά εκφραστικής δύναμης είναι OWL Full, OWL DL, OWL Lite και RDFS. Η Jena ενσωματώνει όμως εσωτερικά τους κατάλληλους μηχανισμούς ώστε να χειρίζεται οντολογίες γραμμένες σε διαφορετικές γλώσσες με διαφορετικό τρόπο, καθώς, όπως έχει προαναφερθεί σε προηγούμενο εδάφιο, ανάλογα με τη γλώσσα που χρησιμοποιείται, τίθενται και οι ανάλογοι περιορισμοί. Για παράδειγμα, ενώ σε όλες τις υπο-γλώσσες της OWL υπάρχει η δυνατότητα δημιουργίας μιας αντικειμενικής ιδιότητας, αν η οντολογία είναι γραμμένη σε RDFS κάτι τέτοιο δεν είναι εφικτό, καθώς στο RDFS δεν ορίζονται αντικειμενικές ιδιότητες. Αξίζει να αναφερθεί ότι κατά την επεξεργασία της οντολογίας μέσω της Jena, δεν αλλάζει η RDF αναπαράστασή της, αλλά διατηρείται η μορφή των τριάδων. Επίσης, σημειώνεται και η ύπαρξη ενσωματωμένων reasoners για την εξαγωγή πρόσθετης γνώσης και συμπερασμάτων από την οντολογία. Η Jena αποτελεί το αποτέλεσμα του συνδυασμού διαφορετικών τεχνολογιών καθώς ενσωματώνει μία σειρά από APIs τα οποία αλληλεπιδρούν εσωτερικά.



Σχήμα 38: Η δομή του Jena API.

Για όλους τους παραπάνω λόγους προτιμήθηκε η χρήση της συγκεκριμένης βιβλιοθήκης μεταξύ πολλών που υπάρχουν διαθέσιμες (OWL API, Jastor, JAOB, Sommer). Αρχικά αναλύουμε τις κλάσεις που δημιουργήθηκαν και τις λειτουργίες που ενσωματώνουν και στη συνέχεια δίνουμε ορισμένα παραδείγματα σχετικά με το πώς υλοποιούνται οι βασικότερες περιπτώσεις χρήσης. Αναλυτικότερα δημιουργήθηκε μία Java κλάση για κάθε οντολογική κλάση που υπάρχει στην οντολογία. Έτσι, δημιουργήθηκαν οι κλάσεις που φαίνονται στο Σχήμα 39:



Σχήμα 39: Διάγραμμα κλάσεων του λογισμικού.

Για λόγους απλότητας και οικονομίας χώρου στο Σχήμα 39 παραλείπονται οι μέθοδοι των κλάσεων.

Κάθε κλάση διαθέτει απαραιτήτως έναν ή περισσότερους κατασκευαστές για τη δημιουργία των αντίστοιχων αντικειμένων (Java objects). Η κατασκευή ενός αντικειμένου δεν ισοδυναμεί με τη δημιουργία ενός στιγμιοτύπου της αντίστοιχης οντολογικής κλάσης. Η δημιουργία των στιγμιοτύπων μίας οντολογικής κλάσης γίνεται μέσω των ειδικών μεθόδων που διαθέτει η κάθε αντίστοιχη Java κλάση. Παρ' όλα αυτά, κάθε αντικείμενο διαχειρίζεται μέχρι ένα στιγμιότυπο, έτσι ώστε να υπάρχει μία αντιστοίχιση μεταξύ αντικειμένων – στιγμιότυπων.

Επίσης, έχουν υλοποιηθεί οι κατάλληλες μέθοδοι για την προσθήκη των αντικειμενικών ιδιοτήτων μεταξύ των στιγμιοτύπων όπως αυτές ορίζονται στην οντολογία. Πιο συγκεκριμένα, έχουν αναπτυχθεί δύο μέθοδοι για κάθε ιδιότητα αντικειμενική ή τύπου δεδομένων που υπάρχει στην οντολογία, μία για την προσθήκη της ιδιότητας και μία για την κατάργησή της. Όπως προαναφέρθηκε, η αντιστοίχιση αντικειμένων και στιγμιοτύπων είναι 1 προς 1 έτσι όταν ένα αντικείμενο καλεί μία μέθοδο προσθήκης κάποιας ιδιότητας περνώντας ως όρισμα ένα άλλο αντικείμενο, δε δημιουργείται σύγχυση. Σε κάθε άλλη περίπτωση, όπου ένα αντικείμενο θα αντιστοιχούσε σε παραπάνω από ένα οντολογικά στιγμιότυπα, όταν περνούσαμε ένα αντικείμενο ως παράμετρο σε μία μέθοδο δεν θα ήταν σαφές για ποιο από όλα τα στιγμιότυπα, τα οποία χειρίζεται αυτό το αντικείμενο, προοριζόταν η προσθήκη της ιδιότητας. Με βάση τα παραπάνω, έχει γίνει η εξής θεώρηση: το στιγμιότυπο που αντιστοιχεί στο αντικείμενο που καλεί τη μέθοδο αποτελεί το subject της συσχέτισης. Η αντικειμενική ιδιότητα την οποία αναλαμβάνει να προσθέσει η καλούμενη μέθοδος αποτελεί το predicate και τέλος το στιγμιότυπο στο οποίο αντιστοιχεί το αντικείμενο που περάστηκε ως όρισμα στη συνάρτηση είναι το object της τριάδας.

Ακόμα, όταν προστίθεται μία αντικειμενική ιδιότητα μεταξύ δύο στιγμιότυπων, θα πρέπει να ελέγχεται αν υπάρχει στην οντολογία η αντίστροφη ιδιότητά της (inverse property) και τότε να προστίθεται και αυτή μεταξύ των στιγμιότυπων. Αυτό γίνεται ως εξής: δημιουργούμε μία νέα συσχέτιση θέτοντας ως subject το στιγμιότυπο που προηγουμένως είχε οριστεί ως object, και ως object το στιγμιότυπο που είχε οριστεί προηγουμένως ως subject. Στη συνέχεια, ανακτούμε από την οντολογία την αντίστροφη ιδιότητα απ' αυτή που προστέθηκε αρχικά και την ορίζουμε ως predicate μεταξύ των στιγμιότυπων.

Πέρα από τις παραπάνω Java κλάσεις, οι οποίες αντιστοιχούν στις οντολογικές κλάσεις, αναπτύχθηκε και η κλάση `Utilities` η οποία είναι μία βοηθητική κλάση. Περιέχει χρήσιμες μεθόδους για τον χειρισμό της οντολογίας, καθώς και για τη διαχείριση των αναλυτών. Ορισμένες από τις λειτουργίες τις οποίες περιέχει η `Utilities` είναι οι εξής:

- Προσθήκη αναλυτή
- Κατάργηση αναλυτή
- Εμφάνιση όλων των στιγμιοτύπων τα οποία ανήκουν σε μία οντολογική κλάση

- Εμφάνιση όλων των ιδιοτήτων (αντικειμενικών και τύπου δεδομένων) ενός στιγμιοτύπου μαζί με τις τιμές τους
- Εμφάνιση των ειδοποιήσεων οι οποίες έχουν προστεθεί στην οντολογία μεταξύ δύο χρονικών στιγμών
- Αποκοπή του namespace
- Ενημέρωση της οντολογικής κλάσης `Classifications`.

Οι παραπάνω μέθοδοι είναι στατικές (static) και δεν απαιτείται η δημιουργία αντικειμένου της κλάσης Utilities για την κλήση τους.

Πριν αρχίσουμε να έχουμε οποιαδήποτε αλληλεπίδραση με την οντολογία απαιτείται να «φορτώσουμε» την οντολογία στο λογισμικό ώστε να μπορεί να τη διαχειριστεί. Αρχικά απαιτείται ο ορισμός του namespace το οποίο θα έχουν όλοι οι RDF κόμβοι όπως αυτό ορίστηκε και στην οντολογία που δημιουργήσαμε. Το εν λόγω namespace είναι το "http://www.semanticweb.org/ontologies/2014/04/IDMEF.owl#". Στη συνέχεια παρουσιάζουμε αναλυτικά πώς υλοποιούνται οι βασικότερες από τις περιπτώσεις χρήσης.

4.8.1 Αποθήκευση Ειδοποίηση για Συμβάν Ασφάλειας στην Οντολογία

Αποτελεί τη βασικότερη περίπτωση χρήσης του συστήματος που αναπτύχθηκε. Το σύστημα διαβάζει και επεξεργάζεται μια ειδοποίηση η οποία δεν είναι παρά ένα XML αρχείο και το αποθηκεύει στην οντολογία.

Το πρώτο βήμα είναι το διάβασμα του XML αρχείου, το οποίο γίνεται με χρήση του SAX (Simple API for XML). Το SAX παρέχει έναν μηχανισμό για το διάβασμα δεδομένων από ένα XML αρχείο και αποτελεί εναλλακτική επιλογή αντί του DOM (Document Object Model). Προτιμήθηκε λόγω της απλούστερης διεπαφής του, αλλά και του γεγονότος ότι είναι event based σε αντίθεση με το DOM. Επίσης, με χρήση του SAX δεν απαιτείται η αποθήκευση του XML αρχείου στην προσωρινή μνήμη του συστήματος, γεγονός πολύ σημαντικό αν αναλογιστούμε ότι στο κεντρικό σύστημα μπορούν να φτάνουν την ίδια χρονική στιγμή πολλές ειδοποιήσεις οι οποίες πρέπει να υποστούν επεξεργασία και να αποθηκευτούν.

Αφού λοιπόν διαβαστεί και επεξεργαστεί το XML αρχείο και είναι πλέον γνωστό ποιες ετικέτες (tags) από αυτά που ορίζονται στο XSD του IDMEF περιέχει ακολουθεί η διαδικασία αποθήκευσής του στην οντολογία.

Αρχικά το σύστημα δημιουργεί τα στιγμιότυπα των οντολογικών κλάσεων που απαιτούνται για την αποθήκευση του συμβάντος ασφαλείας. Αυτό γίνεται δημιουργώντας τα Java αντικείμενα των κατάλληλων κλάσεων και καλώντας τις μεθόδους που περιέχουν για τη δημιουργία στιγμιότυπων.

Το επόμενο βήμα είναι η προσθήκη των αντικειμενικών ιδιοτήτων μεταξύ των στιγμιοτύπων που δημιουργήθηκαν. Καλούνται λοιπόν οι απαραίτητες μέθοδοι των

αντικειμένων που δημιουργήθηκαν, οι οποίες αναλαμβάνουν να προσθέσουν τις object properties καθώς και τις inverse τους.

Τέλος προστίθενται οι απαραίτητες ιδιότητες τύπου δεδομένων, σε όποια στιγμιότυπα αυτό απαιτείται.

4.8.2 Προσθήκη Νέου Αναλυτή

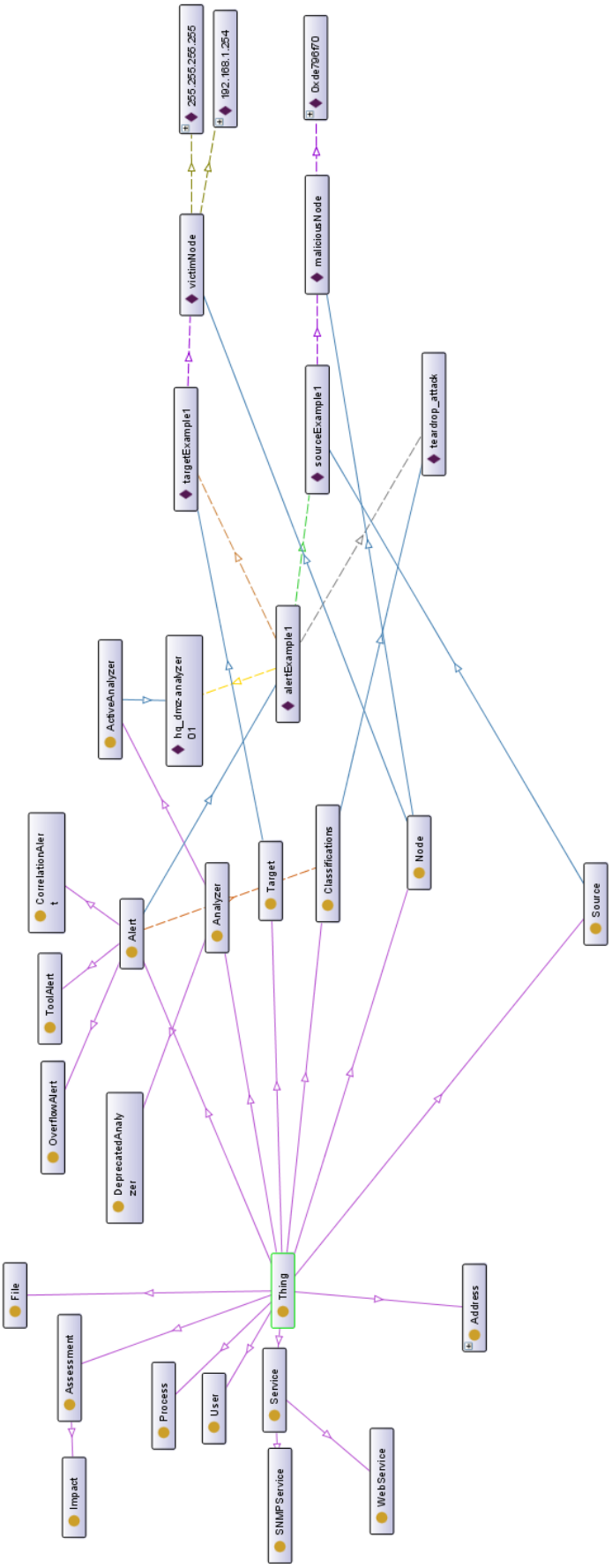
Πρόκειται για μία από τις δύο λειτουργίες που αφορούν στη διαχείριση των αναλυτών και περιέχεται στην κλάση `Utilities`. Ο διαχειριστής επιθυμεί να προσθέσει έναν νέο αναλυτή στο σύστημα και καλεί τη μέθοδο `addAnalyzer(String analyzer)` της κλάσης `Utilities`. Η μέθοδος παίρνει ως όρισμα το όνομα του αναλυτή που πρόκειται να προστεθεί, στη συνέχεια το σύστημα κατασκευάζει ένα Java αντικείμενο της κλάσης `ActiveAnalyzer` το οποίο χρησιμοποιείται για τη δημιουργία ενός στιγμιοτύπου της οντολογικής κλάσης `ActiveAnalyzer`. Τέλος, ο διαχειριστής προσθέτει όποιες ιδιότητες αυτός επιθυμεί και το στιγμιότυπο αποθηκεύεται στην οντολογία.

4.8.3 Κατάργηση Ενός Αναλυτή

Η έτερη λειτουργία σχετικά με τη διαχείριση των αναλυτών περιέχεται και αυτή στην κλάση `Utilities`. Ο διαχειριστής επιθυμεί να καταργήσει έναν αναλυτή ο οποίος δεν χρησιμοποιείται πλέον, έτσι καλεί τη μέθοδο της `Utilities listAllActiveAnalyzers`() η οποία του επιστρέφει όλους τους αναλυτές που είναι εκείνη τη χρονική στιγμή ενεργοί. Ο διαχειριστής διαλέγει τον αναλυτή που επιθυμεί να καταργήσει και τον περνάει ως όρισμα στη μέθοδο `deprecateAnalyzer(String analyzer`). Το σύστημα κατασκευάζει ένα Java αντικείμενο το οποίο χρησιμοποιείται για τη δημιουργία ενός στιγμιοτύπου της κλάσης `DeprecatedAnalyzer` με το ίδιο όνομα που έχει και ο αναλυτής προς κατάργηση. Στη συνέχεια, βρίσκει όλες τις ειδοποιήσεις οι οποίες είχαν προέλθει από τον προς κατάργηση αναλυτή και δημιουργεί τις κατάλληλες αντικειμενικές ιδιότητες μεταξύ αυτών και του στιγμιοτύπου της κλάσης `DeprecatedAnalyzer` που μόλις φτιάχτηκε. Τέλος, δημιουργεί ένα αντικείμενο της κλάσης `ActiveAnalyzer` για να καλέσει τη μέθοδο διαγραφής του εν λόγω στιγμιοτύπου από την οντολογία.

Στο Σχήμα 40 βλέπουμε πώς διαμορφώνεται η οντολογία μετά την αποθήκευση της παρακάτω ειδοποίησης.

```
<idmef:Alert messageid="alertExample1">
           <idmef:Analyzer analyzerid="hq-dmz-analyzer01">
           </idmef:Analyzer>
           <idmef:CreateTime ntpstamp="0xbc723b45.0xef449129">
           2000-03-09T10:01:25.93464-05:00
           </idmef:CreateTime>
           <idmef:Source ident="sourceExample1">
              <idmef:Node ident="maliciousNode">
                 <idmef:name>badguy.example.net</idmef:name>
                 <idmef:Address ident="a1b2c3d4-002"
                 category=" ipv4-addr-hex ">
                     <idmef:address>0xde796f70</idmef:address>
                </idmef:Address>
               </idmef:Node>
           </idmef:Source>
           <idmef:Target ident="targetExample1">
              <idmef:Node ident="victimNode" category="dns">
                 <idmef:Address ident="a1b2c3d4-002"
                 category="ipv4-net-mask">
                    <idmef:address>192.168.1.254</idmef:address>
                    <idmef:netmask>255.255.255.255</idmef:netmask>
                 </idmef:Address>
              </idmef:Node>
           </idmef:Target>
           <idmef:Classification text="teardrop_attack">
              <idmef:Reference origin="bugtraqid">
                 <idmef:name>124</idmef:name>
                 <idmef:url>http://www.securityfocus.com/bid/124
                </idmef:url>
              </idmef:Reference>
           </idmef:Classification>
  </idmef:Alert>
```



Σχήμα 40: Η διαμόρφωση της οντολογίας μετά την αποθήκευση της παραπάνω IDMEF ειδοποίησης.

5 Συμπεράσματα και Μελλοντικές Κατευθύνσεις

5.1 Τελικά Συμπεράσματα

Στην παρούσα εργασία παρουσιάστηκε μία προσέγγιση για την αποθήκευση σε μία οντολογία συμβάντων ασφάλειας που ακολουθούν το μορφότυπο που ορίζει το IDMEF. Να σημειωθεί ότι οντολογίες για την αναπαράσταση συμβάντων ασφαλείας έχουν χρησιμοποιηθεί και στο παρελθόν με μεγάλη επιτυχία. Τα συμβάντα ασφαλείας είναι πλούσια σε πληροφορίες με μεγάλη σημασιολογική αξία. Σε αυτό το πλαίσιο, η προτεινόμενη προσέγγιση στοχεύει στην εξαγωγή πρόσθετης γνώσης και συμπερασμάτων σχετικά με πιθανές νέες επιθέσεις και απειλές. Η χρήση των IDMEF περιστατικών ασφάλειας ως στοιχείο εισόδου κρίνεται ιδανική, καθώς τα δεδομένα ακολουθούν την προκαθορισμένη μορφή που ορίζει το XML Schema του IDMEF, διευκολύνοντας έτσι το διάβασμα των δεδομένων.

Η αποθήκευση των ειδοποιήσεων στην οντολογία μπορεί να δώσει λύση σε προβλήματα που παρουσιάζει το IDMEF. Ειδικότερα, πολύ συχνά παρατηρείται ότι ενώ δύο ειδοποιήσεις έχουν προκύψει ως αποτέλεσμα της ίδιας επίθεσης, λόγω της ανομοιογένειας των αναλυτών οι ειδοποιήσεις που παράγονται περιέχουν διαφορετική πληροφορία. Μέσω της αποθήκευσής τους στην οντολογία, οι ειδοποιήσεις μπορούν να συσχετιστούν και να δηλωθεί ότι αντιστοιχούν στην ίδια επίθεση.

Κατά την εκπόνηση της πτυχιακής εργασίας, πραγματοποιήθηκε εκτενής μελέτη των δύο προτύπων που έχει ορίσει η IETF για την αναπαράσταση συμβάντων ασφαλείας, δηλαδή των IDMEF και IODEF. Η παρούσα εργασία παρείχε τη δυνατότητα εξοικείωσης με τον Σημασιολογικό Ιστό και ιδιαίτερα με τις γλώσσες XML, RDF, RDF-S, OWL. Τέλος η παρούσα εργασία βοήθησε στην εξοικείωση με εργαλεία σχετικά με την επεξεργασία οντολογιών όπως, ο editor Protégé, αλλά και τη Java βιβλιοθήκη Jena.

5.2 Μελλοντικές Κατευθύνσεις

Σε ό,τι αφορά μελλοντικές κατευθύνσεις για περαιτέρω εξέλιξη του συστήματος λογισμικού που αναπτύχθηκε στην εργασία ιδιαίτερο ενδιαφέρον παρουσιάζει: η δημιουργία ενός reasoner με υψηλή συλλογιστική δύναμη, ο οποίος με επεξεργασία των αποθηκευμένων συμβάντων ασφάλειας που υπάρχουν στην οντολογία, θα είναι σε θέση:

- Να συσχετίζει αποδοτικά συμβάντα ασφαλείας μεταξύ τους τα οποία εκ πρώτης όψεως φαίνονται ασύνδετα
- Να κάνει εκτιμήσεις για το σε ποιους κόμβους μπορεί να έχει εγκατασταθεί κακόβουλο λογισμικό με αποτέλεσμα αυτοί να έχουν γίνει «σκλάβοι» (bots-zombies) και να χρησιμοποιούνται για επιθέσεις εν αγνοία τους

- Να κάνει εκτιμήσεις για το σε ποιο botnet ανήκουν οι κόμβοι οι οποίοι έχουν διαπιστωθεί ότι είναι σκλάβοι
- Να ανακαλυφθούν νέα είδη επιθέσεων που χρησιμοποιούνται
- Μία πιθανή βελτίωση θα μπορούσε να αποτελέσει η ανάπτυξη μίας οντολογίας ειδικά για κατηγοριοποίηση επιθέσεων, απειλών και κενών ασφάλειας, η οποία θα συνενωνόταν με την υπάρχουσα οντολογία και θα αντικαθιστούσε την οντολογική κλάση `Classification`. Ακόμα, θα μπορούσε να δημιουργεί ένας πράκτορας λογισμικού για την περιοδική αυτόματη ενημέρωση της οντολογίας με τις νέες απειλές και επιθέσεις οι οποίες έχουν δημοσιευτεί
- Τέλος, κρίνεται σκόπιμη η δημιουργία μίας οντολογίας αναφορικά με ενέργειες που θα εκτελούνται όταν διαπιστωθεί ένα είδος επίθεσης (countermeasures). Μέσω της συνένωσης της εν λόγω οντολογίας με την υπάρχουσα και της αντιστοίχισης στιγμιοτύπων της κλάσης `Classification` με στιγμιότυπα της `CounterMeasures`, το σύστημα θα έχει τη δυνατότητα να καταπολεμήσει πολλές επιθέσεις σε πραγματικό χρόνο αυτοματοποιημένα.

ΟΡΟΛΟΓΙΑ

Αγγλικός Όρος	**Ελληνικός όρος**
abstract syntax	αφηρημένη σύνταξη
address	διεύθυνση
software agents	πράκτορας λογισμικού
analyzer	αναλυτής
assertion	δήλωση
axioms	αξίωμα
class extension	επέκταση κλάσης
cohesion	συνοχή
compatability	συμβατότητα
concepts	έννοιες
coupling	σύζευξη
database	βάση δεδομένων
datatype	τύπος δεδομένων
decidable	αποφασίσιμη
deprecated	παρωχημένος
description logic	περιγραφική λογική
disjoint	αμοιβαίως αποκλειόμενες
domain	τομέας
dynamic data	δυναμικά δεδομένα
event correlation	συσχετισμός γεγονότων
expressive power	εκφραστική δύναμη
extensible	επεκτάσιμο
facts	γεγονότα
formal meaning	τυπικό νόημα
free-form textual documents	αρχεία σε ελεύθερη μορφή
functional	συναρτησιακή
functional decomposition	λειτουργική αποσύνθεση
heading	επικεφαλίδα
heterogeneous	ετερογενής
hierarchy	ιεραρχία
increments	προσαύξηση
individual	άτομο/στιγμιότυπο
interconnectivity	διασυνδεσιμότητα
interoperability	διαλειτουργικότητα
inverse	ανάστροφος
knowledge base	γνωσιακή βάση
layers	στρώματα
literal	λεκτικό
log files	αρχείο καταγραφής
logical	λογικό

mahine understandable	κατανοητό από μηχανές
malicious software	κακόβουλο λογισμικό
manager	διευθυντής
metadata	μεταδεδομένα
metamodeling	μεταμοντελοποιήση
mutual authentication mechanism	μηχανισμός αμοιβαίας αυθεντικοποίησης
namespace	μετα-μοντελοποιήση
nodes	κόμβοι
object	αντικείμενο
plaintext format	μορφή απλού κειμένου
predicate	κατηγόρημα
processes	διεργασίες
property	ιδιότητα
range	σύνολο τιμών
reasoning power	συλλογιστική δύναμη
receiver	παραλήπτης
reflexive	αυτόματος
relations	σχέσεις
requirements elicitation	εκμαίευση απαιτήσεων
requirements refinement	εκλέπτυνση απαιτήσεων
resource	πόρος
reusability	επαναχρησιμοποίηση
schema	σχήμα
semantics	σημασιολογία
sender	αποστολέας
sensor	αισθητήρας
sentence	πρόταση
sequence diagram	διάγραμμα ακολουθίας
services	υπηρεσίες
software engineering	τεχνολογία λογισμικού
standardised	τυποποιημένη
subclass	υποκλάση
subject	υποκείμενο
sub-property	υπο-ιδιότητα
superclass	υπερκλάση
super-property	υπερ-ιδιότητα
symmetric	συμμετρική
system administrator	διαχειριστής συστήματος
system design	σχεδιασμός συστήματος
tag	ετικέτα
taxonomy	ταξονομία
testing	έλεγχος
transitive	μεταβατικός
triple	τριάδα

Trojan horse	Δούρειος Ίππος
undecidable	μη-αποφασίσιμη
users	χρήστες
valid	έγκυρος
vulnerabilities	αδυναμίες
well formed	καλά ορισμένο μορφολογικά
workflow	ροή εργασιών

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

API	Application Programming Interface
CSIRT	Computer Security Incident Response Team
CVE	Common Vulnerabilities and Exposures
DTD	Document Type Definition
HTML	HyperText Markup Language
IDMEF	Intrusion Detection Message Exchange Format
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IODEF	Incident Object Description Exchange Format
OSVDB	Open Source Vulnerability Databse
OWG	Ontology Working Group
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RFC	Request For Comments
RSS	Really Simple Syndication
SMIL	Synchronized Multimedia Integration Language
URIs	Uniform Resource Identifier
URLs	Uniform Resource Locator
W3C	World Wide Web Consortium
WAP	Wireless Application Protocol
WSDL	Web Service Definition Language
WWW	World Wide Web
XHTML	Extensible HyperText Markup Language
XML	EXtensible Markup Language
XSD	XML Schema

Βιβλιογραφία

[1] Debar, H., Curry, D., Feinstein, B.: The Intrusion Detection Message Exchange Format (IDMEF). IETF Request for Comments 4765 (March 2007).

[2] M. Wood, M. Erlinger : Intrusion Detection Message Exchange Requirements. IETF Request for Comments 4766 (March 2007).

[3] R. Danyliw, J. Meijer, Y. Demchenko : The Incident Object Description Exchange Format : Request for Comments 4765 (December 2007).

[4] López de Vergara, J.E., Vázquez, E., Martin, A., Dubus, S., Lepareux, M.N.: Use of ontologies for the definition of alerts and policies in a network security platform. Journal of Networks 4(8), 720–733 (2009).

[5] Anna P. Antonakopoulou, Fotios I. Gogoulos, Georgios V. Lioudakis, Aziz S. Mousas, Dimitra I. Kaklamani and Iakovos S. Venieris : An Ontology for Privacy-Aware Access Control in Network Monitoring Environments.

[6] Nora Cuppens-Boulahia, Frederic Cuppens, Jorge E. Lopez de Vergara, Enrique Vasquez, Javier Guerra, Herve Debar : An ontology-based approach to react to network attacks. Third International Conference on Risks and Security of Internet and Systems: CRiSIS'2008.

[7] Jorge E. López de Vergara, Enrique Vázquez, Antony Martin, Samuel Dubus, Marie-Noëlle Lepareux : Use of Ontologies for the Definition of Alerts and Policies in a Network Security Platform. JOURNAL OF NETWORKS, VOL. 4, NO. 8, OCTOBER 2009.

[8] Jorge E. López de Vergara, Víctor A. Villagrá, Pilar Holgado,Elena de Frutos, and Iván Sanz : A Semantic Web Approach to Share Alerts among Security Information Management Systems.

[9] Matthew Horrdige : A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools Edition 1.3. March 24, 2011.

[10] Description Logic : A Formal Foundation for Ontology Languages and Tools.

[11] Jena – A Semantic Web Framework for Java, http://jena.sourceforge.net/.

[12] An Introduction to RFD and the Jena RDF API.

[13] Natalya F. Noy, Deborah L. McGuinness : Ontology Development 101: A Guide to Creating Your First Ontology.

[14] Introduction to RDF : www.w3cschools.com/webservices/ws_rdf_intro.asp.

[15] Introduction to OWL : www.w3schools.com/webservices/ws_rdf_owl.asp.

[16] Web Ontology Language (OWL), available at http://www.w3.org/2004/OWL.

[17] Naming and Addressing: URIs, URLs…, available at http://www.w3.org/Addressing.

[18] Extensible Markup Language (XML), available at www.w3.org/XML/

[19] OWL API, available at http://owlapi.sourceforge.net/index.html.

[20] Thomas Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing", August, 1993.

[21] RDF Schema, available at http://www.w3.org/TR/rdf-schema/.

[22] SPARQL, available at http://www.w3.org/TR/rdf-sparql-query/.

[23] RIF, available at http://www.w3.org/2005/rules/wiki/RIF_Working_Group.

[24] Obrst, L., Wray, R.E. and Liu, H., Ontological Engineering for B2B E-Commerce. in International Conference on Formal Ontology in Information Systems (FOIS'01), (Ogunquit, Maine, USA, 2001), ACM Press, 117-126.

[25] K. Bontcheva and H. Cunningham. 2003. The Semantic Web: A New Opportunity and Challenge for HLT. In Proceedings of the Workshop HLT for the Semantic Web and Web Services at ISWC 2003.

[26] Γιώργος Στοϊλος. Γλώσσες Αναπαράστασης Γνώσης στο Σημασιολογικό Ιστό.

[27] Beckett, D. (2003) RDF/XML Syntax Specification (Revised). World Wide Web Consortium, 10 October 2003.

[28] Bechhofer, S., van Harmelen F., Hendler, J., Horrocks, I., McGuinness., D.L., Patel-Schneider P.F., Stein, A.L., eds. (2004) OWL Web Ontology Language Reference.

[29] Boris, M., (2005) On the Properties of Metamodeling in OWL. Proceedings of the 4th International Semantic Web Conference (ISWC 05), pp. 548-562 Galway, Ireland.

[30] Brickey, D., Guha, R.V. (2000) RDF Vocabulary Description Language 1.0: RDF Schema, W3C.

[31] Hayes, P. (2003) RDF Semantics. World Wide Web Consortium, 10 October 2003.

[32] Horrocks, I., Patel-Schneider, P.F. (2003) Reducing OWL Entailment to Description Logic Satisfiability. Proceedings of the 2nd International Semantic Web Conference (ISWC '03).

[33] Klyne, G., Carroll, J. (2003) Resource Description Framework (RDF): Concepts and Abstract Syntax. World Wide Web Consortium.

[34] Lassila, O., Swick, R. (1999) W3C Resource Description Framework model and syntax specification.

[35] Software Development Life Cycle Models: available at www.codebetter.com/raymondlewallen/2005/07/13/software-development-life-cycle-models/.

[36] Ontology Visualization Protégé Tools- A Review : R. Sivakumar, P.V. Arivoli August 2011.