Αξιολόγηση Αυτοματοποιημένων Εργαλείων Στατικής Ανάλυσης C/C++ Κώδικα Με Εστίαση Σε Ευπάθειες Υπερχείλισης

Η Διπλωματική εργασία κατατέθηκε στο τμήμα
Μηχανικών Πληροφοριακών & Επικοινωνιακών Συστημάτων
της Πολυτεχνικής Σχολής του Πανεπιστημίου Αιγαίου

UNIVERSITY OF THE AEGEAN

Νικόλας Βαμβακάκης

Επιτροπή

Επιβλέπων: Καθηγητής Γεώργιος Καμπουράκης
Μόνιμος Επίκουρος Καθηγητής Αλέξης Καπόρης
Επίκουρος Καθηγητής Δημήτριος Σκούτας

Ιανουάριος 2023

Assessment of Static Analysis Tools for C/C++ Code Focused on Overflow Vulnerabilities

A thesis statement
submitted to the Department of Information & Communication Systems Engineering
School of Engineering of the University Of The Aegean

UNIVERSITY OF THE AEGEAN

Nikolas Vamvakakis

Committee

Supervisor: Professor Georgios Kambourakis
Permanent Assistant Professor Alexis Kaporis
Assistant Professor Dimitrios Skoutas

January 2023

Δήλωση Αυθεντικότητας

Βεβαιώνω ότι είμαι συγγραφέας αυτής της διπλωματικής μεταπτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης έχω αναφέρει τις πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Τέλος, βεβαιώνω ότι αυτή η διπλωματική εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του μεταπτυχιακού προγράμματος σπουδών του Τμήματος Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων του Πανεπιστημίου Αιγαίου στη Σάμο.

Καρλόβασι, 5 Ιανουαρίου 2023

Νικόλας Βαμβακάκης

(Υπογραφή)

Περίληψη

Η παρούσα διπλώματική εργασία μελετάει και αξιολογεί την απόδοση ανοιχτού κώδικα εργαλείων στατικής ανάλυσης εφαρμοσμένα σε δημοφιλή, ανοιχτού κώδικα πραγματικά πακέτα λογισμικού υλοποιημένα σε C/C++. Η βιβλιογραφία περιέχει αρκετές μελέτες τέτοιων εργαλείων, αλλά η πλειοψηφία τους εφαρμόζεται σε σουίτες ελέγχου ειδικά διαμορφωμένες ώστε εσκεμμένα να περιλαμβάνουν πληθώρα προβλημάτων ασφάλειας. Συγκεκριμένα, παρά την αυξημένη δυσκολία λόγω της απαίτησης για χειροκίνητη επιβεβαίωση των ευρημάτων, σε αντίθεση με την προαναφερόμενη προσέγγιση, θεωρήθηκε σημαντικό τα εργαλεία ανάλυσης κώδικα να εκτελεστούν σε πραγματικά πακέτα λογισμικού, συγκεκριμένα στα Kodi Home Theater και Telegram. Αυτό θα επιτρέψει την εξαγωγή συμπερασμάτων σχετικά με την απόδοση των εργαλείων σε πραγματικές συνθήκες. Τα πακέτα λογισμικού που επιλέχθηκαν βρίσκονται σε ώριμο επίπεδο ανάπτυξης, άρα το ενδιαφέρον επικεντώνεται σε τυχόν παραμένοντα προβλήματα ασφάλειας. Η μελέτη επικεντρώθηκε σε προβλήματα υπερχείλισης και πιο συγκεκριμένα σε Buffer Overflows (CWE-120) και Integer Overflows (CWE-190). Ως εργαλεία στατικού ελέγχου επιλέχτηκαν τα Flawfinder και CppCheck (cli|gui). Συνολικά, αφού παρουσιαστεί το θεωρητικό υπόβαθρο, η διπλωματική εστιάζει στο πειραματικό μέρος, δηλαδή την εκτέλεση των εργαλείων ανάλυσης κώδικα στα δύο επιλεγμένα πακέτα λογισμικού και τον έλεγχο της ορθότητας των ευρημάτων. Συμπερασματικά, παρόλο που και τα δύο εργαλεία ανίχνευσαν κενά ασφαλείας στο κώδικα και των δύο πακέτων λογισμικού, τα προβλήματα αυτά δεν είναι άμεσα (πρακτικά) εκμεταλλεύσιμα.

Νικόλας Βαμβακάκης
Τμήμα Μηχανικών Πληροφοριακών & Επικοινωνιακών Συστημάτων
Πανεπιστήμιο Αιγαίου

Abstract

This Master thesis explores and assesses the performance of open source static analysis tools, applied to popular open source real-life projects implemented in C/C++. The bibliography presents several studies of such tools, however their majority have been applied to specially designed software test suites which include a great number of deliberate security flaws. Even though the difficulty of such a process is increased due to the need for manual verification of the findings, in contrast to the aforementioned approaches, it was considered important that such tools should be applied to real-life software projects, specifically Kodi Home Theater and Telegram. This will enable testers to draw conclusions about the performance of such tools against real-life mature software and provide comparisons with results stemming from software test suites. Specifically, the chosen software projects are in a mature development stage, assuring that any verification actions by the development team are already complete. Therefore, it is interesting to check if the analysis tools can identify any remaining, latent security flaws. The study is focused on overflow vulnerabilities, specifically on Buffer Overflows (CWE-120) and Integer Overflows (CWE-190). Flawfinder and CppCheck (cli|gui) were chosen as the static analysis tools to be tested. Initially, a theoretical analysis of all components that contributed to this thesis is presented, which is followed by the tools' execution targeting the aforementioned software projects. We present the findings after verifying them; this provides an initial, empirical assessment of the tool's performance as well. It is interesting to see that although both the tools were able to detect existing security flaws in various parts of the software projects, these flaws were structured in such a way that they are not directly exploitable.

Nikolas Vamvakakis
Department of Information and Communication Systems Engineering
University of the Aegean

Περιεχόμενα

Λίστα Σχημάτων

Λίστα Πινάκων

Κεφάλαιο 1

Εισαγωγή

Καθώς η τεχνολογία εξελίσσεται όλο και ταχύτερα, όλο και περισσότερες καθημερινές και μη διαδικασίες πλέον υλοποιούνται ψηφιακά και ως αποτέλεσμα οι απαιτήσεις για ασφαλή και υψηλής ποιότητας λογισμικό αυξάνονται διαρκώς. Ταυτόχρονα, η αύξηση του μεγέθους και της πολυπλοκότητας αυτών των συστημάτων, των αναγκών για διαρκή διασυνδεσιμότητα μεταξύ τους αλλά και το εύρος της εφαρμογής τους αποτελούν παράγοντες οι οποίοι συμβάλουν καθοριστικά στην αξία προστασίας τους. Εξαλλού, οι προαναφερθέντες παράγοντες αυξάνουν τη δυσκολία για την επίτευξη των απαιτήσεων ασφάλειας των εμπλεκομένων πληροφοριακών και επικοινωνιακών συστημάτων. Ταυτόχρονα, σε επίπεδο οργανισμών, η εξασφάλιση της ασφάλειας των προϊόντων που αφορούν το λογισμικό αποτελεί πλέον μία από τις σημαντικότερες και κρισιμότερες διαδικασίες που έχουν να διαχειριστούν σε καθημερινή βάση [1].

1.1 Επισκόπηση του πρόβλματος

Εξαιτίας της τεχνολογικής έκρηξης έχουν αυξηθεί δραματικά οι κίνδυνοι αλλά και οι απειλές απέναντι σε ψηφιακά συστήματα. Συγκεκριμένα, χρόνο με το χρόνο, παρατηρείται μία συνεχόμενη αύξηση του αριθμού των ευπαθειών οι οποίες ανακαλύπτονται σε μεγάλο εύρος προϊόντων λογισμικού. Οι ευπάθειες αυτές, μόνες ή συνδυαστικά, δύναται να δημιουργήσουν απειλές για τους χρήστες οι οποίες συνήθως μεταφράζονται σε περιστατικά παρακολούθησης, κλοπής ταυτότητας, μη εξουσιοδοτημένης πρόσβασης σε πληροφορίες, διακοπής σημαντικών υπηρεσιών εξαιτίας επιθέσεων άρνησης υπηρεσίας (DoS) και πολλά ακόμα [2]. Όλα αυτά οφείλονται συχνά στη χαμηλή ποιότητα κώδικα, ο οποίος μπορεί μεν να είναι πλήρως λειτουργικός, αλλά να περιλαμβάνει σημαντικά κενά ασφαλείας που οδηγούν σε άμεσα ή έμμεσα εκμεταλεύσιμες ευπάθειες. Είναι γενικά παραδεκτό ότι περίπου 90% των ανιχνευμένων επιθέσεων οφείλονται σε προβλήματα του κώδικα που εκτελείται στα αντίστοιχα συστήματα [3].

1.2 Υπάρχουσες Προσεγγίσεις

Με βάση τα παραπάνω, καθίσταται σαφής η σημασία της ανάπτυξης ασφαλούς λογισμικού. Αυτό περιλαμβάνει και διαδικασίες ελέγχου του ήδη υπάρχοντος λογισμικού για λανθάνοντα κενά ασφαλείας. Ακόμα και σήμερα αποτελεί συνήθη πρακτική διαδικασίες που στοχεύουν την ασφάλεια του λογισμικού να λαμβάνουν χώρα μετά την ανάπτυξή του, όμως σε αυτό το στάδιο η πολυπλοκότητα και ο χρόνος που απαιτείται είναι ήδη αρκετά αυξημένα [4]. Αντίθετα, σύμφωνα με τη βιβλιογραφία τα ζητήματα ελέγχου πρέπει να λαμβάνουν χώρα καθ΄ όλη τη διάρκεια του κύκλου ανάπτυξης του λογισμικού και μάλιστα όσο συντομότερα συμβεί κάτι τέτοιο, τόσο ευκολότερη και οικονομικότερη είναι η διαδικασία επιδιόρθωσης των κενών ασφάλειας [1, 3, 4]. Μάλιστα, ο κύκλος ζωής ανάπτυξης ασφαλούς λογισμικού θα πρέπει να ενσωματώνει καλές πρακτικές ασφάλειας σε κάθε στάδιο της διαδικασίας τόσο του σχεδιασμού, όσο και της ανάπτυξης [1].

Για την επίτευξη του παραπάνω στόχου υπάρχουν πολλές τεχνικές οι οποίες μπορούν να εφαρμοστούν, όπως για παράδειγμα η χρήση σταθερών μοντέλων όσον αφορά στην ανάλυση του προς υλοποίηση συστήματος, η επιμόρφωση των προγραμματιστών σε θέματα συγγραφής ασφαλούς κώδικα, η διασφάλιση ασφαλούς περιβάλλοντος κατά την ανάπτυξη του λογισμικού, η ανάλυση πηγαίου κώδικα, και πολλές άλλες [3]. Η παρούσα μεταπτυχιακή διατριβή επικεντρώνεται στο τελευταίο ζήτημα, δηλαδή την ανάλυση πηγαίου κώδικα και συγκεκριμένα τη στατική ανάλυση κώδικα.

1.3 Σκοπός και Στόχοι Εργασίας

Εξαιτίας της αξίας της στατικής ανάλυσης στην ανάπτυξη ασφαλούς κώδικα, έχουν εμφανιστεί πολλά σχετικά εργαλεία, τόσο ανοικτού όσο και κλειστού κώδικα. Όπως είναι φυσικό, έχουν ήδη γίνει αρκετές προσπαθειες σύγκρισης της απόδοσης τέτοιων εργαλείων. Παρόλα αυτά, η ανομοιογένεια που εμφανίζουν ως προς το εύρος των σφαλμάτων ασφαλείας τα οποία δύναται να αναγνωρίσουν, τις τεχνικές αναγνώρισης τις οποίες εφαρμόζουν, την ευκολία χρήσης τους, αλλά και τον τρόπο με τον οποίο παρουσιάζουν τα ευρήματα τους δημιουργούν εμπόδια σε αυτή τη σύγκριση.

Δυσκολία παρουσιάζεται επιπλέον και στη διαδικασία επιβεβαίωσης των ευρημάτων που παρουσιάζουν τα εργαλεία, καθώς τις περισσότερες φορές απαιτείται η χειροκίνητη εξέταση τους, γεγονός το οποίο οδηγεί συνήθως στην εφαρμογή τους σε ειδικά διαμορφωμένες σουίτες ελέγχου, οι οποίες επιλύουν ακριβώς αυτό το πρόβλημα. Παρόλα αυτά, τέτοιες σουίτες δεν μπορούν να προσομοιώσουν με ακρίβεια ένα πραγματικό πακέτο λογισμικού, και για αυτόν το λόγο παρατηρείται έλλειψη μελετών στη βιβλιογραφία οι οποίες αξιολογούν την απόδοση τέτοιων εργαλείων σε πραγματικές συνθήκες.

Παρακινούμενοι από αυτό το κενό στη βιβλιογραφία, σκοπός της παρούσας με-

ταπτυχιακής διατριβής είναι η σύγκριση εργαλείων ανοιχτού κώδικα, τα οποία όμως εφαρμόζονται σε (επίσης ανοιχτού κώδικα) πραγματικά πακέτα λογισμικού. Λόγω της πολυπλοκότητας που παρουσιάζει μία τέτοια ανάλυση, αποφασίστηκε επίσης η μελέτη να επικεντρωθεί μόνο σε σφάλματα υπερχείλισης (καταχωρητή/ακεραίου), όπως θα παρουσιαστούν στην ενότητα 2.4, καθώς αποτελούν κρίσιμα και παράλληλα συνήθη ζητήματα κατά τη συγγραφή κώδικα. Επιπλέον, λόγω του προαναφερόμενου κενού στη βιβλιογραφία, αποφασίστηκε τόσο τα εργαλεία όσο και τα πακέτα λογισμικού που θα εξεταστούν να είναι δημοφιλή στους χρήστες, ώστε τα αποτελέσματα να μπορούν να χρησιμοποιηθούν ως βάση για μελλοντικές μελέτες στην ίδια περιοχή.

1.4 Διάρθρωση Εργασίας

Το κεφάλαιο 2 προσφέρει μια σύντομη βιβλιογραφική επισκόπηση, αναλύοντας τα βασικά θέματα τα οποία απαιτείται να γνωρίζει ο αναγνώστης για την καλύτερη κατανόηση της διπλωματικής. Στο κεφάλαιο 3, περγράφονται και συγκρίνονται μεταξύ τους τα εργαλεία ανάλυσης κώδικα που χρησιμοποιήθηκαν, ενώ στο κεφάλαιο 4 παρουσιάζονται και αναλύονται τα αποτελέσματα. Η μελέτη ολοκληρώνεται στο κεφάλαιο 5 παρουσιάζοντας τα συμπεράσματα και κάποιες προτάσεις για μελλοντική εργασία. Επιπλέον, στο παράρτημα Α περιγράφονται συνοπτικά οι έλεγχοι που είναι ικανό να πραγματοποιήσει το εργαλείο CppCheck ενώ το παράτημα Β περιλαμβάνει πίνακες με το σύνολο των ευρημάτων των εργαλείων.

Κεφάλαιο 2

Βιβλιογραφική Επισκόπηση

Πριν από το πειραματικό μέρος που περιλαμβάνει την εκτέλεση των εργαλείων στατικής ανάλυσης και παρουσιάζεται στην ενότητα 3, κρίνεται απαραίτητο να περιγραφούν ορισμένα προαπαιτούμενα στα οποία βασίζεται η παρούσα διπλωματική εργασία. Συγκεκριμένα, παρουσιάζοται γενικές έννοιες σχετικά με την ανάλυση πηγαίου κώδικα, τη στατική ανάλυση (στην οποία επικεντρώνεται η διπλωματική), τα ανοικτού κώδικα εργαλεία αυτοματοποιημένου ελέγχου και τις ευπάθιες που μελετούνται. Επίσης, παρουσιάζονται τα δύο πακέτα λογισμικού ανοιχτού κώδικα που επιλέχτηκαν για να αναλυθούν από τα εργαλεία στατικής ανάλυσης.

2.1 Ανάλυση Πηγαίου Κώδικα και Εργαλεία Στατικής Ανάλυσης

2.1.1 Ανάλυση Πηγαίου Κώδικα

Η ανάλυση πηγαίου κώδικα ορίζεται ως ο έλεγχος του πηγαίου κώδικα με σκοπό την εύρεση και επιδιόρθωση σφαλμάτων, ευπαθειών ή ακόμα και κακών πρακτικών σύμφωνα με το techtarget [5], τα οποία πολλές φορές οδηγούν σε κενά ασφαλείας. Η ανάλυση του πηγαίου κώδικα πραγματοποιείται είτε δυναμικά, κατά την εκτέλεση δηλαδή του προγράμματος, είτε στατικά με την εξέταση του κώδικα για σφάλματα, χωρίς όμως αυτός να εκτελείται [2].

2.1.2 Στατική Ανάλυση Κώδικα

Η στατική ανάλυση κώδικα αποτελεί ουσιαστικά έλεγχο του κώδικα για σφάλματα στην υλοποίηση ή χρήση μη προτεινόμενων συναρτήσεων, τα οποία αν δεν διορθωθούν δύναται να οδηγήσουν σε άμεσα ή έμμεσα εκμεταλλεύσιμες ευπάθειες [2]. Αυτό μπορεί να πραγματοποιηθεί είτε χειροκίνητα είτε αυτοματοποιημένα με χρήση κατάλληλων εργαλείων. Παρόλα αυτά, όπως ειπώθηκε προηγουμένως, το μέγεθος των πακέτων

λογισμικού μεταφραζόμενο σε σειρές κώδικα αυξάνεται διαρκώς, καθιστώντας τον χειροκίνητο έλεγχο ασύμφορο, αργό και σε αρκετές περιπτώσεις ανέφικτο [4, 6]. Για το λόγο αυτό αναπτύσσονται διαρκώς νέα αυτοματοποιημένα εργαλεία στατικού ελέγχου, τόσο ανοιχτού κώδικα για ελεύθερη χρήση, όσο και εμπορικά [3].

2.1.3 Βασικά Χαρακτηριστικά Εργαλείων

Τα αυτοματοποιημένα εργαλεία ελέγχου κώδικα αναπτύχθηκαν και συνεχίζουν να εξελίσσονται έχοντας ως στόχο την ελαχιστοποίηση του χρόνου και της ενέργειας που απαιτείται για να γίνει έλεγχος του κώδικα. Παράλληλα, αποτελούν σημαντικό βοήθημα για τους προγραμματιστές ώστε να παράγουν όσο το δυνατόν ασφαλέστερο, σταθερότερο, αποδοτικότερο και πιο αξιόπιστο κώδικα [3]. Όπως ειπώθηκε ήδη, τα εργαλεία στατικής ανάλυσης αυτά δεν απαιτούν την εκτέλεση του κώδικα του εξεταζόμενου λογισμικού, γεγονός που συμβάλει στη συντομότερη και ευκολότερη αναγνώριση και επιδιόρθωση τυχόν σφαλμάτων κατά τη διάρκεια ανάπτυξης του λογισμικού [2]. Σημαντικό είναι να αναφερθεί ότι όλα τα σχετικά εργαλεία παρουσιάζουν ψευδώς αρνητικά αποτελέσματα, δηλαδή δεν είναι σε θέση να αναγνωρίσουν όλα τα σφάλματα που υπάρχουν στον κώδικα, αλλά επίσης και ψευδώς θετικά, δηλαδή κάθε εύρημα του εργαλείου δεν αποτελεί με σιγουριά σφάλμα στον κώδικα [2]. Όσον αφορά τον βαθμό στον οποίο παρουσιάζονται ψευδώς θετικά ή αρνητικά αποτελέσματα, αυτό σχετίζεται συνήθως και με τη μεθοδολογία που υλοποιεί το εκάστοτε εργαλείο. Για παράδειγμα, υπάρχουν εργαλεία τα οποία επιλέγουν να μεγιστοποιήσουν τα ψευδώς θετικά αποτελέσματα, με σκοπό την κατά το δυνατόν ελαχιστοποίηση ψευδώς αρνητικών αποτελεσμάτων. Για αυτά τα εργαλεία προτείνεται προσεκτική χειροκίνητη επανεξέταση των αποτελεσμάτων για τον καθορισμό των αληθώς και ψευδώς θετικών ευρημάτων. Από την άλλη, το ποσοστό εντοπισμού πιθανών (άρα και υπαρκτών) σφαλμάτων είναι μεγαλύτερος. Αντίθετα, υπάρχουν εργαλεία τα οποία εμφανίζουν χαμηλό αριθμό ψευδώς θετικών αποτελεσμάτων, καθώς πρωτίστως εστιάζουν στην ορθότητα των ευρημάτων τους. Παράλληλα, τα αποτελέσματα των αληθώς θετικών αποτελεσμάτων είναι επίσης λιγότερα. Τέτοια εργαλεία επικεντρώνονται συνήθως στην εύρεση των κρισιμότερων σφαλμάτων μόνο, ενώ λόγω του συνήθως μικρότερου πλήθους συνολικών ευρημάτων η διαχείριση τους είναι ταχύτερη.

Εκτός των παραπάνω, όπως συναντάται συχνά στη βιβλιογραφία [7, 9], τα εργαλεία αυτά έχουν συγκεκριμένο εύρος εφαρμογής όσον αφορά τις κατηγορίες ευπαθειών τις οποίες μπορούν να αναγνωρίσουν. Το ίδιο ισχύει και για τις διαφορετικές υποπεριπτώσεις σφαλμάτων που αναγνωρίζουν σε κάθε κατηγορία [3, 7]. Για αυτόν το λόγο, παρότι η σημασία τέτοιων εργαλείων ελέγχου είναι πλέον αδιαμφισβήτητη, συνήθως προτείνεται η επιλογή τους να γίνεται με βάση τις ανάγκες των ελέγχων τους οποίους θα υποστηρίξουν. Προτείνεται επίσης η χρήση πολλαπλών εργαλείων τα οπο-

ία θα λειτουργούν συμπληρωματικά [7]. Επιπλέον, αυτού του τύπου τα εργαλεία δεν έχουν σκοπό να αντικαταστήσουν τις τυπικές διαδικασίες ελέγχου του κώδικα, αλλά να συμβάλουν στη μεγιστοποίηση της αποδοτικότητάς της [6]. Σε αυτό το σημείο είναι σημαντικό να αναφερθεί ότι τα εργαλεία αυτά, ανεξάρτητα της χρησιμότητας τους, εξακολουθούν να είναι 'εργαλεία' [7]. Αυτό σημαίνει ότι η ανθρώπινη κρίση είναι αναντικατάστατη, καθώς επίσης και ότι ο χειριστής ενός τέτοιου εργαλείου χρειάζεται να έχει επαρκή εμπειρία σε θέματα συγγραφής ασφαλούς κώδικα για να το αξιοποιήσει, εννοώντας να επιβεβαιώσει και ακολούθως να διορθώσει την πηγή των σφαλμάτων στον κώδικα [2].

Επιπλέον, τα εν λόγω εργαλεία δεν ακολουθούν κάποιο πρότυπο παραγωγής αναφορών, γεγονός το οποίο σε συνδυασμό με τη διαφορετικότητα που παρουσιάζουν σχετικά με τα σφάλματα που αναγνωρίζουν δημιουργεί δυσκολίες τόσο στη μεταξύ τους σύγκριση όσο και στην παρουσίαση των τελικών ευρημάτων όταν χρησιμοποιούνται παράλληλα. Αυτό σημαίνει ότι την ευθύνη διαχείρισης του εκάστοτε εργαλείο έχει ο χειριστής του.

2.1.4 Γλώσσες προγραμματισμού C/C++

Οι γλώσσες προγραμματισμού C, C++ έχουν σχεδιαστεί εστιάζοντας στην αποδοτικότητα και στη φορητότητα. Αυτό έχει ως αποτέλεσμα να παρέχουν μικρή υποστήριξη στην αποφυγή και διαχείριση σφαλμάτων κατά την εκτέλεση. Για παράδειγμα, σε αυτές τις γλώσσες, δε γίνεται έλεγχος για το αν η ανάγνωση ή η εγγραφή που πραγματοποιείται σε κάποιον πίνακα είναι εντός ορίων, με αποτέλεσμα τέτοιοι έλεγχοι να πρέπει να γίνουν απευθείας από τον προγραμματιστή ή, ανάλογα με την περίπτωση, να εξασφαλίζεται ότι δεν απαιτούνται [6]. Αυτό έχει ιδιαίτερη αξία στο πλαίσιο της παρούσας διπλωματικής καθώς για τις ανάγκες της θα γίνει έλεγχος σε λογισμικό που έχει υλοποιηθεί σε C & C++.
Παρακάτω ακολουθεί σύντομη επισκόπηση των εργαλείων στατικής ανάλυσης κώδικα Flawfinder και CppCheck τα οποία αξιοποιήθηκαν στο πλαίσιο της διπλωματικής.

2.2 Flawfinder

Το Flawfinder αποτελεί ένα γνωστό και επιτηδευμένα απλό εργαλείο στατικής ανάλυσης κώδικα από τον David A. Wheeler [7]. Το συγκεκριμένο εργαλείο εστιάζει στις γλώσσες προγραμματισμού C/C++ [8]. Χρησιμοποιεί απλοϊκές ευρετικές τεχνικές, γεγονός το οποίο μειώνει την ακρίβεια του εργαλείου, αλλά η φιλοσοφία του βασίζεται στο να λανθάνει από τη σκοπιά της επιφυλακτικότητας, δηλαδή προτιμάει να παρουσιάσει παραπάνω ψευδώς θετικά αποτελέσματα, τα οποία θα εξεταστούν και θα απορριφθούν, παρά ψευδώς αρνητικά, δηλαδή να αποτύχει να εντοπίσει υπαρκτά προβλήματα

[8]. Το Flawfinder προσφέρεται με την άδεια GNU GPL 2ης ή μεταγενέστερης έκδοσης (GPLv2+), ενώ για τους σκοπούς αυτής της διπλωματικής χρησιμοποιείται η έκδοση 2.0.19 [8]. Επίσης, το εν λόγω εργαλείο είναι συμβατό με το πρότυπο Common Weakness Enumeration (CWE), ενώ έχει λάβει την πιστοποίηση CII Best Practices "passing" [7]. Είναι γραμμένο στη γλώσσα Python για λόγους απλοποίησης της συγγραφής και επέκτασης του, το οποίο το κάνει πιο αργό από ότι αν είχε γραφεί σε C [7]. Τέλος, το Flawfinder συνεργάζεται καλά με διάφορα εργαλεία συγγραφής κειμένου και ενσωματωμένα περιβάλλοντα ανάπτυξης λογισμικού [8].

2.2.1 Βασική Λειτουργικότητα

Το Flawfinder δεν αποτελεί κάποιο σύνθετο εργαλείο στατικού ελέγχου λογισμικού, παρόλα αυτά η απλότητα χρήσης και η αποτελεσματικότητα του το έχουν καταστήσει ιδιαίτερα δημοφιλές εδώ και αρκετά χρόνια. Το εργαλείο δεν ασχολείται με αναλύσεις ροής δεδομένων ή ελέγχους ροής, αλλά κυρίως με αντιστοίχηση μοτίβων κειμένου, αγνοώντας τα σχόλια [7]. Χρησιμοποιεί μία ενσωματωμένη βάση δεδομένων από συναρτήσεις της C/C++ με γνωστά προβλήματα, όπως buffer overflows (παραδείγματος χάριν, οι strcpy(), scanf() και άλλες), η οποία καλείται 'σύνολο κανόνων' (ruleset) [8]. Εκτός των συναρτήσεων οι οποίες θεωρούνται γενικά 'επικίνδυνες', το Flawfinder έχει τη δυνατότητα να αναγνωρίσει και συναρτήσεις οι οποίες είναι προβληματικές για συγκεκριμένα συστήματα, όπως Windows και Linux. Ο έλεγχος γίνεται αντιστοιχίζοντας αυτές τις συναρτήσεις με τον κώδικα, αγνοώντας τα σχόλια και τις γραμματοσειρές, με εξαίρεση εντολές σχετικές με το Flawfinder [7]. Εκτός των παραπάνω, το εργαλείο έχει επίγνωση ορισμένων εξαιρέσεων, για παράδειγμα όταν const strings δίδονται ως όρισμα στη συνάρτηση gettext() δεν αναφέρεται κάποιο σφάλμα αφού αυτή είναι η ενδεδειγμένη χρήση, μειώνοντας με αυτόν τον τρόπο τον πλήθος των ψευδώς θετικών αποτελεσμάτων.

Αφού εκτελεστεί, το Flawfinder παράγει μία λίστα ευρημάτων από υποψήφια σφάλματα ασφαλείας, ταξινομημένα από τα κρισιμότερα στα λιγότερο κρίσιμα. Το επίπεδο κρισιμότητας, εκτός από την υπό εξέταση συνάρτηση, εξαρτάται και από τις τιμές των ορισμάτων της, καθώς για παράδειγμα ένα const string είναι ασφαλέστερο από ένα μεταβλητό. Επίσης, η απλή μέθοδος που εφαρμόζει το εργαλείο του επιτρέπει να μην μπερδεύεται από macro definitions και άλλες παραμέτρους, οι οποίες δυσκολεύουν συνθετότερα παρόμοια εργαλεία [7].

2.2.2 Χρήση

Η χρήση του Flawfinder είναι ιδιαίτερα απλή, αφού αρκεί να του δοθεί μία λίστα φακέλων ή αρχείων. Από αυτά θα εξετάσει όλα όσα έχουν επέκταση η οποία προσδιορίζει αρχεία τύπου C/C++, όπως για παράδειγμα *.cpp, *.hpp, κλπ [8]. Έτσι, για παράδειγ-

μα, αρκεί κάποιος να εκτελέσει το εργαλείο από το βασικό φάκελο του λογισμικού το οποίο θέλει να εξετάσει και αυτό θα αναγνωρίσει και θα αναλύσει όλα τα σχετικά αρχεία. Για τη βελτιστοποίηση της απόδοσης του, το εργαλείο προσφέρει δυνατότητες ανάλυσης μόνο των αλλαγών που έχουν υπάρξει σε κάποιον πηγαίο κώδικα μέσω της επιλογής "–patch/-P" και εκμετάλλευσης patch αρχείων, παραγόμενα από κάποιο αποθετήριο λογισμικού τύπου git ή svn [8]. Γενικότερα, το Flawfinder περιλαμβάνει πλήθος από επιλογές: έλεγχος εισόδου δεδομένων, επιλογή της μορφής εξαγωγής των αποτελεσμάτων, επιλογή του είδους των ευρημάτων τα οποία θα παρουσιαστούν, κ.ά.

2.2.3 Εξαγωγή Αποτελεσμάτων

Κάθε εύρημα της ανάλυσης αποτελείται από [8]:

- Το όνομα του αρχείου μαζί με την πλήρη διαδρομή σε αυτό.
- Τον αριθμό γραμμής στην οποία αναγνωρίστηκε το πρόβλημα.
- Το επίπεδο κρισιμότητας σε αγκύλες ([0]-[5]).
- Την κατηγορία του σφάλματος.
- Το όνομα της συνάρτησης.
- Μία σύντομη περιγραφή αναφορικά με τον λόγο που θεωρεί το εργαλείο ότι πρόκειται για ευπάθεια καθώς και κάποια πρόταση αντιμετώπισης της.

Παράδειγμα αποτελέσματος:

```
./lib/libUPnP/Platinum/Source/Tests/FileMediaServer/FileMediaServerTe
st.cpp:151: [5] (buffer) gets:
 Does not check for buffer overflows (CWE-120, CWE-20). Use fgets() instead.
  while (gets(buf)) {
```

Τέλος, το Flawfinder επιτρέπει στο χρήστη να αγνοήσει προβλήματα που έχουν αναγνωριστεί από το εργαλείο, αλλά ο ίδιος θεωρεί ότι δεν είναι ορθά, με χρήση του παρακάτω σχολίου:

```
/* Flawfinder: ignore */
```

2.2.4 Κάλυψη Ευπαθειών

Το Flawfinder έχει τη δυνατότητα να αναγνωρίσει τις παρακάτω ευπάθειες με βάση το πρότυπο Common Weaknesses Enumeration (CWE) [8]:

- CWE-20: Λανθασμένη επικύρωση εισόδου (Improper Input Validation)
- CWE-22: Εσφαλμένος περιορισμός ονόματος διαδρομής σε περιορισμένο από δικαιώματα κατάλογο (Improper Limitation of a Pathname to a Restricted Directory ("Path Traversal"))
- **CWE-78**: Ακατάλληλη εξουδετέρωση ειδικών στοιχείων που χρησιμοποιούνται σε εντολή λειτουργικού συστήματος (Improper Neutralization of Special Elements used in an OS Command ("OS Command Injection"))
- CWE-119: Ακατάλληλος περιορισμός λειτουργιών εντός των ορίων μιας ενδιάμεσης μνήμης (buffer) (Improper Restriction of Operations within the Bounds of a Memory Buffer)
- **CWE-120**: Αντιγραφή buffer χωρίς ελέγχο του μεγέθους εισόδου (Buffer Copy without Checking Size of Input ("Classic Buffer Overflow"))
- CWE-126: Υπερανάγνωση buffer (Buffer Over-read)
- **CWE-134**: String μη ελεγχόμενης μορφής (Uncontrolled Format String)
- **CWE-190**: Υπερχείλιση ακεραίου (Integer Overflow)
- CWE-250: Εκτέλεση με περιττά προνόμια (Execution with Unnecessary Privileges)
- **CWE-327**: Χρήση ξεπερασμένου ή επισφαλή κρυπτογραφικού αλγορίθμου (Use of a Broken or Risky Cryptographic Algorithm)
- CWE-362: Ταυτόχρονη εκτέλεση χρήσης κοινόχρηστου πόρου με ακατάλληλο συγχρονισμό (Concurrent Execution using Shared Resource with Improper Synchronization ("Race Condition"))
- CWE-377: Ανασφαλές προσωρινό αρχείο (Insecure Temporary File)
- **CWE-676**: Χρήση δυνητικά επικίνδυνης συνάρτησης (Use of Potentially Dangerous Function)
- **CWE-732**: Εσφαλμένη εκχώρηση άδειας σε κρίσιμο πόρο (Incorrect Permission Assignment for Critical Resource)

- CWE-785: Χρήση συνάρτησης χειρισμού διαδρομής χωρίς buffer μέγιστου μεγέθους (Use of Path Manipulation Function without Maximum-sized Buffer)
- **CWE-807**: Εξάρτηση από αναξιόπιστες εισόδους σε απόφαση ασφάλειας (Reliance on Untrusted Inputs in a Security Decision)
- **CWE-829**: Συμπερίληψη λειτουργικότητας από αναξιόπιστο πλαίσιο ελέγχου (Inclusion of Functionality from Untrusted Control Sphere)

Τα CWEs σε έντονο χρώμα εμπεριέχονται στη λίστα των 25 κορυφαίων CWE/SANS. Επίσης, εκτελώντας την εντολή "flawfinder –listrules" ο χρήστης μπορεί να δεί τους κανόνες με βάση τους οποίους γίνεται η αναγνώριση σφαλμάτων από το Flawfinder.

2.3 CppCheck

Το CppCheck είναι κι αυτό εργαλείο στατικής ανάλυσης κώδικα γραμμένο σε C ή C++. Εφαρμόζει μοναδικά χαρακτηριστικά στην ανάλυση του για τον εντοπισμό σφαλμάτων, ενώ επικεντρώνεται σε απροσδιόριστες συμπεριφορές και επικίνδυνες δομές κώδικα [9]. Αντίθετα με το Flawfinder, το CppCheck στοχεύει στο να εντοπίσει μόνο πραγματικά προβλήματα και ως εκ τούτου να παράξει όσο το δυνατό λιγότερα ψευδώς θετικά αποτελέσματα. Επίσης, είναι ικανό να αναλύσει κώδικα που περιέχει μη προτυποποιημένο συντακτικό, όπως συνηθίζεται για παράδειγμα στα ενσωματωμένα συστήματα [10]. Το CppCheck διανέμεται ως ανοιχτού κώδικα, αλλά παρέχεται και σε "Premium" έκδοση με επιπλεόν λειτουργικότητα και υποστήριξη [9]. Για τις ανάγκες της παρούσας διπωματικής, χρησιμοποιείται η ανοιχτού κώδικα έκδοση 2.9.

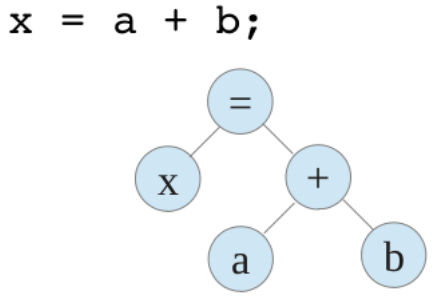
2.3.1 Βασική Λειτουργικότητα

Το CppCheck είναι συμβατό με οποιονδήποτε μεταγλωττιστή (compiler) υποστηρίζει C++11 ή μεταγενέστερη. Είναι cross-platform με εφαρμογή σε διάφορα περιβάλλοντα και προσφέρει δυνατότητες ανάλυσης μη προτυποποιημένου κώδικα [10]. Το CppCheck δρομολογεί unsound [11] flow sensitive [12] ανάλυση, σε αντίθεση με πολλά άλλα εργαλεία τα οποία χρησιμοποιούν path sensitive [13] ανάλυση, βασισμένη σε αυθαίρετες ερμηνείες [9]. Παρόλα αυτά, εξ ορισμού, η ανάλυση path sensitive είναι αποδοτικότερη από την unsound flow sensitive, αν και η χρήση της τελευταίας επιφέρει αποτελέσματα διαφορετικά από την πρώτη, καθιστώντας και τις δύο τεχνικές πολύτιμες.

2.3.2 Βασικά Στοιχεία Επεξεργασίας Δεδομένων

Η παρούσα ενότητα παρουσιάζει τις συνιστώσες οι οποίες προετοιμάζουν τα δεδομένα για ανάλυση από τους checkers του CppCheck [14].

Προεπεξεργαστής: Οι checkers δεν έχουν πρόσβαση σε ακατέργαστο ή προεπεξεργασμένο κώδικα. Έτσι, όλα τα πηγαία αρχεία προεπεξεργάζονται και στη συνέχεια συμβολοποιούνται ώστε να γίνονται κατανοητά από τους checkers του CppCheck.
Σύμβολα: Μετά την αρχική προεπεξεργασία, ο κώδικας αναλύεται σε μία λίστα συμβόλων. Για παράδειγμα, η εντολή "a = b – c;" αναλύεται στα σύμβολα [a, =, b, -, c, ;], το σύνολο των οποίων αποτελεί τη λίστα συμβόλων. Αυτά στη συνέχεια αναλύονται από τους checkers του εργαλείου.
Συντακτικό Δένδρο: Κάθε εντολή αναπαρίσταται από ένα συντακτικό δένδρο, όπου τα σύμβολα χρησιμοποιούνται ως κόμβοι. Για παράδειγμα, το συντακτικό δένδρο της εντολής "x = a + b;" αναπαρίσταται όπως φαίνεται στο σχήμα 2.1:



Σχήμα 2.1: Συντακτικό δένδρο εντολής "x = a + b;" [14]

Βάση δεδομένων συμβόλων: Αποτελεί μία βάση δεδομένων η οποία περιέχει πληροφορίες σχετικά με τύπους, συναρτήσεις, και άλλα χρήσιμα στοιχεία του πηγαίου κώδικα.
Ροή τιμών: Πραγματοποιείται, ευαίσθητη στο περιεχόμενο, ανάλυση ροής τιμών των μεταβλητών, όπου όλες οι πιθανές τιμές εντοπίζονται και αποθηκεύονται στο σχετικό σύμβολο. Έτσι, κάθε σύμβολο συνοδεύεται από μία λίστα με τις πιθανές τιμές που δύναται να λάβει.
Βιβλιοθήκη: Περιέχει πληροφορίες σχετικά με τις συναρτήσεις που χρησιμοποιούνται, όπως για παράδειγμα εάν μία συνάρτηση επιστρέφει κάποια τιμή ή όχι, δεσμεύει μνήμη, κ.ά..

Παρακάτω παρουσιάζεται ένα παράδειγμα εντοπισμού buffer overflow για καλύτερη κατανόηση της λειτουργίας του εργαλείου στην πράξη.

2.3.3 Εντοπισμός Buffer Overflow

Κατά την ανάλυση για υπερχείλιση το CppCheck δεν δρομολογεί ανάλυση ελέγχου ροής. Έτσι, δεν θα προσπαθήσει να καθορίσει το πώς και εάν μία εντολή μπορεί να εκτελεστεί, αλλά υποθέτει πως όλες οι περιπτώσεις είναι εφικτές [15]. Συνεπώς, στο παρακάτω παράδειγμα δε θα ασχοληθεί με το εάν η συνθήκη "x == y" πραγματοποιείται, αλλά θα αναφέρει το πρόβλημα σε κάθε περίπτωση:

```
void overflowExample()
{
  char array[4];
  if (x == y)
  {
    array[4] = 0;
  }
}
```

Ο δημιουργός του εργαλείου CppCheck υποστηρίζει ότι τέτοια σφάλματα πρέπει να εντοπίζονται καθώς η εντολή υπάρχει, όπως και το πρόβλημα σε αυτή [15]. Παρόλα αυτά, υπάρχουν περιπτώσεις κλήσης συναρτήσεων όπου πρέπει να πραγματοποιηθεί ανάλυση ροής ελέγχου για να αποφευχθούν ψευδώς θετικά αποτελέσματα, τις οποίες το CppCheck διαχειρίζεται κατάλληλα.

2.3.4 Κρισιμότητα κινδύνων

Το CppCheck κατηγοριοποιεί τα ευρήματα στις εξής κατηγορίες με βάση την κρισιμότητά τους [10]:

- **Σφάλμα (error):** Όταν κατά την εκτέλεση υπάρχει απροσδιόριστη συμπεριφορά ή άλλου είδους σφάλμα, όπως για παράδειγμα διαρροή μνήμης ή πόρων.
- **Προειδοποίηση (warning):** Όταν εκτελείται κώδικας που πιθανόν οδηγεί σε απροσδιόριστη συμπεριφορά.
- **Στυλ (style):** Στυλιστικά προβλήματα όπως συναρτήσεις οι οποίες δεν καλούνται, περιττός κώδικας, χρήση σταθερών δομών, προτεραιότητες πράξεων, κ.ά..
- **Απόδοση (performance):** Βασικές προτάσεις βελτίωσης του χρόνου εκτέλεσης, οι οποίες όμως το εργαλείο δεν υπόσχεται ότι θα οδηγήσουν σε πραγματική διαφορά στην ταχύτητα εκτέλεσης του κώδικα.
- **Φορητότητα (portability):** Ειδοποιήσεις σχετικές με προβλήματα μεταφερσιμότητας, όπως συμπεριφορές που εξαρτούνται από το υλικό του συστήματος και άλλα.
- **Πληροφορία (information):** Προτάσεις σε αλλαγές της παραμετροποίησης του CppCheck με βάση τον κώδικα.

2.3.5 Εξαγωγή Αποτελεσμάτων

Κατά την παρουσίαση των αποτελεσμάτων κάθε σφάλμα αναφέρεται με μια εγγραφή λάθους <(error)>. Αυτή αποτελείται από τα παρακάτω πεδία [10]:

- **Αναγνωριστικό (id):** Αναγνωριστικό σχετικά με τη κατηγορία του σφάλματος.
- **Κρισιμότητα (severity):** Τύπος της επικινδυνότητας του ευρήματος. Αναλύεται στις κατηγορίες που περιγράφονται στην ενότητα 'Κρισιμότητα Κινδύνων'.
- **Μήνυμα (msg):** Σύντομο μήνυμα για το σφάλμα.
- **Αναλυτικό Μήνυμα (verbose):** Αναλυτικό μήνυμα σφάλματος.
- **Ασαφές (inconclusive):** Γνώρισμα το οποίο χρησιμοποιείται μόνο όταν το μήνυμα σφάλματος είναι ασαφές.
- **CWE (cwe):**Σεχτικό αναγνωριστικό όταν το σφάλμα ανήκει σε κάποια κατηγορία CWE.

Χρησιμοποιείται επίσης το στοιχείο <(location)>, το οποίο εμφανίζει πληροφορίες σχετικά με όλες τις τοποθεσίες οι οποίες έχουν σχέση με το σφάλμα, ενώ προτεραιότητα έχουν οι κύριες τοποθεσίες.
Αποτελείται από τα παρακάτω γνωρίσματα [10]:

- **Αρχείο (file):** Το όνομα μαζί με τη διαδρομή του αρχείου όπου εμφανίζεται το σφάλμα.
- **Αρχείο0 (file0):** Όνομα του πηγαίου αρχείου.
- **Γραμμή (line):** Άριθμός γραμμής όπου εμφανίζεται το σφάλμα.
- **Πληροφορίες (info):** Σύντομες πληροφορίες σχετικά με την κάθε τοποθεσία.

Παράδειγμα αποτελέσματος:

<error id="integerOverflow" severity="error" msg="Signed integer overflow for expression '0xff<<(j*8)'." verbose="Signed integer overflow for expression '0xff<<(j*8)'." cwe="190" file0="lib/libUPnP/Neptune/ThirdParty/axTLS/crypto/bigint.c">
<location file="lib/libUPnP/Neptune/ThirdParty/axTLS/crypto/bigint.c" line="721" column="30" info="Integer overflow"/>
<location file="lib/libUPnP/Neptune/ThirdParty/axTLS/crypto/bigint.c" line="719" column="23" info="Assuming that condition 'j<4' is not redundant"/>
</error>

2.3.6 Κάλυψη Ευπαθειών

Παρακάτω παρουσιάζονται οι κατηγορίες ευπαθειών τις οποίες δύναται να αναγνωρίσει το CppCheck [9]:

- Νεκροί δείκτες (Dead pointers).
- Διαίρεση με μηδέν (Division by zero).
- Υπερχείλιση ακεραίου (Integer overflows).
- Μη έγκυροι τελεστές μετατόπισης bit (Invalid bit shift operands)
- Μη έγκυρες μετατροπές (Invalid conversions).
- Μη έγκυρη χρήση της βιβλιοθήκης STL (Invalid usage of STL).
- Διαχείριση μνήμης (Memory management).
- Αναφορά σε μεταβλητή μηδενικού δείκτη (Null pointer dereferences).
- Έλεγχοι εκτός ορίων (Out of bounds checking).
- Μη αρχικοποιημένες μεταβλητές (Uninitialized variables).
- Εγγραφή σε σταθερές (Writing const data).

Ενδιαφέρον έχει επίσης ότι τα παρακάτω CWEs προστέθηκαν χάρη στο CppCheck [9]:

CVE-2017-1000249: Buffer overflow βασισμένο σε στοίβα. Προστέθηκε από τον Thomas Jarosch και οφείλεται στην ύπαρξη λάθους υλοποίησης σε μία κατάσταση (condition, π.χ. a <b).
CVE-2013-6462: Stack overflow στο X.org, η οποία υπήρχε για πάνω από δύο δεκαετίες.
CVE-2012-1147: Το αρχείο readfilemap.c του εργαλείου expat [16] πριν την έκδοση 2.1.0 επιτρέπει σε επιτιθέμενους να προκαλέσουν άρνηση παροχής υπηρεσιών μέσω ενός μεγάλου αριθμού από αρχεία xml.
Στο παράρτημα Β παρουσιάζονται όλοι οι έλεγχοι τους οποίους δύναται να πραγματοποιήσει το CppCheck [17].

2.3.7 CppCheck-GUI

Το CppCheck διανέμεται επίσης και σε έκδοση με γραφικό περιβάλλον (GUI). Σε αυτήν, υπάρχουν επιλογές οι οποίες δεν είναι διαθέσιμες από τη γραμμή εντολών [10] και για αυτόν το λόγο αποφασίστηκε να χρησιμοποιηθεί και να συγκριθούν τα ευρήματα της με την κλασική έκδοση. Για την εκτέλεση του αρκεί κάποιος να δημιουργήσει ένα νέο έργο και να ορίσει το φάκελο ή τους φακέλους τους οποίους θέλει να αναλύσει. Το γραφικό περιβάλλον είναι εύχρηστο, με επιλογές παραμετροποίησης του εργαλείου, επιλογή πλατφόρμας για την ανάλυση, διάφορες επιλογές φιλτραρίσματος των ευρημάτων, ταξινόμησης, κ.ά..

2.4 Ευπάθειες υπό Εξέταση

Παρακάτω παρουσιάζονται οι ευπάθειες τύπου CWE-120, δηλαδή ευπάθειες Buffer Overflow και τύπου CWE-190, δηλαδή υπερχείλισης ακεραίου (Integer Overflow).

2.4.1 Buffer Overflow Χωρίς Έλεγχο Μεγέθους Εισόδου - CWE-120

Το buffer overflow χωρίς έλεγχο μεγέθους εισόδου αφορά την αντιγραφή ενός buffer εισόδου σε buffer εξόδου, χωρίς να υπάρχει επιβεβαίωση ότι το μέγεθος του πρώτου είναι μικρότερο του τελευταίου, οδηγώντας στην υπερχείλιση του [18]. Αυτή η κατάσταση δημιουργείται όταν γίνεται προσπάθεια εισαγωγής περισσότερων δεδομένων σε έναν buffer από αυτά που μπορεί να αποθηκεύσει ή εισαγωγής σε περιοχές της μνήμης έξω από τα όρια του buffer [18]. Αποτελεί τη συνηθέστερη αιτία υπερχείλισης, γι' αυτό αποκαλείται και 'Κλασική Υπερχείλιση' (Classic Buffer Overflow) [18]. Η ύπαρξη τέτοιων υπερχειλίσεων υποδεικνύει συνήθως ότι ο συγγραφέας του κώδικα δεν έχει λάβει υπόψη βασικά μέτρα ασφάλειας.

Η πιθανότητα εκμετάλλευσης τέτοιων ευπαθειών είναι υψηλή, ενώ οι συνηθέστερες επιπτώσεις αφορούν υπερχειλίσεις οι οποίες επιτρέπουν την εκτέλεση αυθαίρετου κώδικα, γεγονός το οποίο παραβιάζει την πολιτικής ασφαλείας του λογισμικού και μπορεί να οδηγήσει στην ανατροπή οποιασδήποτε άλλης υπηρεσίας ασφαλείας. Τούτο πλήττει τόσο την ακεραιότητα όσο και την εμπιστευτικότητα και διαθεσιμότητα του συστήματος [18]. Επιπτώσεις στη διαθεσιμότητα προκύπτουν και από διαφόρων τύπων υπερχειλίσεις, οι οποίες οδηγούν σε κατάρρευση του προγράμματος και μπορούν να συμβούν τυχαία. Τέλος, ο συνολικός αριθμός καταγεγραμμένων ευπαθειών αυτής της κατηγορίας είναι 1347 [19].

2.4.2 Υπερχείλιση Ακεραίου - CWE-190

Σε αυτήν την περίπτωση πραγματοποιείται κάποιος υπολογισμός, όπου η τιμή του αποτελέσματος υποτίθεται ότι είναι μεγαλύτερη της αρχικής, ο οποίος μπορεί να υπερχειλίσει το μέγιστο μέγεθος ενός ακεραίου ή να πραγματοποιήσει περιστροφή της τιμής του. Επίσης, μπορεί ενεργοποιήσει άλλες ευπάθειες όταν ο υπολογισμός αφορά τη διαχείριση πόρων ή τον έλεγχο εκτέλεσης [20].

Υπερχείλιση ακεραίου συμβαίνει όταν η τιμή ενός ακεραίου αυξάνεται σε τιμή μεγαλύτερη από ότι μπορεί να αποθηκεύσει με βάση τον ορισμό του τύπου του. Σε περίπτωση που συμβεί κάτι τέτοιο η τιμή μπορεί να περιστραφεί και να αποκτήσει μία πολύ μικρή ή ακόμα και αρνητική ποσότητα. Η υπερχείλιση θεωρείται ιδιαίτερα κρίσιμη σε περιπτώσεις που μπορεί να πυροδοτηθεί από την είσοδο του χρήστη. Τέτοιες περιπτώσεις αφορούν εκμετάλλευση για έλεγχο των βρόγχων, λήψη αποφάσεων ασφαλείας, καθορισμό μεγεθών τα οποία αφορούν κατανομή μνήμης, αντιγραφή τιμών, κ.ά. [20].

Η πιθανότητα εκμετάλλευσης σε αυτήν την περίπτωση είναι μέτριας κρισιμότητας, ενώ υπάρχει ποικιλία επιπτώσεων εάν κάτι τέτοιο συμβεί. Μία τέτοια ευπάθεια λοιπόν μπορεί να οδηγήσει σε απροσδιόριστη συμπεριφορά, με αποτέλεσμα να καταρρεύσει η εκτελούμενη εφαρμογή, απειλώντας τη διαθεσιμότητα του συστήματος. Στην περίπτωση που περιλαμβάνονται μεταβλητές ευρετηρίου βρόγχου, αυξάνεται επίσης η πιθανότητα ατέρμονων βρόγχων [20]. Επιπτώσεις στην ακεραιότητα είναι επίσης δυνατές, εάν οι αντίστοιχες τιμές υπό περιλαμβάνουν σημαντικά δεδομένα. Εάν η περιστροφή οδηγήσει σε άλλα προβλήματα, όπως buffer overflow, μπορεί να προκληθεί περαιτέρω ζημιά στη μνήμη. Τέλος, αυτή η ευπάθεια μπορεί να παρέχει περιθώρια για πραγματοποίηση buffer overflow, με τη βοήθεια του οποίου μπορεί να εκτελεστεί αυθαίρετος κώδικας, γεγονός το οποίο είναι υψηλής κρισιμότητας όσον αφορά θέματα εμπιστευτικότητας, διαθεσιμότητας αλλά και ελέγχου πρόσβασης. Πρέπει να αναφερθεί ότι μέχρι στιγμής ο συνολικός αριθμός καταγεγραμμένων ευπαθειών αυτής της κατηγορίας είναι 1713 [19].

2.4.3 Επιλογή Ευπαθειών

Η ανάλυση των δύο επιλεγέντων εργαλείων αποφασίστηκε να επικεντρωθεί στις παραπάνω δύο κατηγορίες λαθών καθώς συναντώνται συχνά και αποτελούν πηγή πολλών ευπαθειών. Παράλληλα, η εκμετάλλευσή τους ενδέχεται να επιφέρει σημαντική ζημία στο σύστημα. Λόγω της δημοτικότητας τους αναφέρονται συχνά στη βιβλιογραφία, όπως και σε μελέτες σύγκρισης της απόδοσης στατικών εργαλείων ανάλυσης πηγαίου κώδικα, με χρήση κάποιας πλατφόρμας ειδικά διαμορφωμένης για τέτοιες δοκιμές. Συγκεκριμένα, τέτοιες μελέτες λαμβάνουν υπόψη τους και τα εργαλεία που μελετούνται στην παρούσα εργασία. Έτσι, αναφορικά με την CWE-120, το Flawfinder έχει χρησιμοποιηθεί για την ανίχνευση ευπαθειών σε τέσσερις δημοσιεύσεις [1, 3, 4, 22], ενώ το CppCheck σε τρεις [1, 3, 22]. Για την CWE-190, το Flawfinder έχει αξιοποιηθεί με επιτυχία σε δύο δημοσιεύσεις [1, 3], ενώ έχει δοκιμαστεί ανεπιτυχώς στην εργασία [22]. Παράλληλα, το CppCheck έχει χρησιμοποιηθεί με επιτυχία για τον εντοπισμό ευπαθειών στην εργασία [22] (στην ίδια εργασία το Flawfinder είχε χρησιμοποιηθεί χωρίς επιτυχία). Με αναφορά στα παραπάνω, ιδιαίτερο ενδιαφέρον παρουσιάζει η εξέταση της αποτελεσματικότητας των εργαλείων ως συμπληρωματικά το ένα του άλλου με στόχο τη μεγιστοποίηση των ευρημάτων.

Τέλος, πρέπει να σημειωθεί ότι στην παρούσα εργασία δε χρησιμοποιείται κάποιο testing benchmark όπως στην προαναφερθείσα βιβλιογραφία, αλλά αναλύονται δημοφιλή, ώριμα πακέτα λογισμικού ανοιχτού κώδικα, καθώς η έρευνα επικεντρώνεται στην απόδοση των εργαλείων σε πραγματικές συνθήκες.

2.5 Λογισμικά προς ανάλυση

Στα πλαίσια της εργασίας αποφασίστηκε να χρησιμοποιηθούν δύο λογισμικά ανοιχτού κώδικα, τα οποία θα αναλυθούν για σφάλματα υλοποίησης από τα προαναφερθέντα εργαλεία. Τα πακέτα λογισμικού που επιλέχτηκαν είναι το “Kodi Home Theater” [23] και το “Telegram Desktop” [26], τα οποία περιγράφονται συνοπτικά στη συνέχεια.

2.5.1 Kodi Home Theater Software

Το Kodi Home Theater αποτελεί εφαρμογή αναπαραγωγής πολυμέσων και κόμβο ψυχαγωγίας για ψηφιακά μέσα. Είναι γραμμένο σε C++ σε ποσοστό 87% και διανέμεται μέσω github [23]. Επιλέχθηκε καθώς αποτελεί ένα από τα δέκα δημοφιλέστερα έργα λογισμικού ανοιχτού κώδικα, γραμμένα σε C++, τα οποία επιζητούν συνεισφορά στην περαιτέρω ανάπτυξη τους σύμφωνα με το Codacy [24], αλλά και ένα από τα 23 κορυφαία έργα λογισμικού ανοιχτού κώδικα σε C++ σύμφωνα με το LibHunt [25]. Επιπλέον, το Kodi Home Theater αποτελεί ένα ώριμο λογισμικό, το οποίο αναπτύσσεται από το 2003 [23], γεγονός το οποίο σε περίπτωση αναγνώρισης σφαλμάτων δείχνει ότι οποιοδήποτε

λογισμικό παρουσιάζει ανάγκες βελτίωσης της ασφάλειάς του ανεξαρτήτως της φάσης του κύκλου ζωής του. Εκτός της δημοτικότητάς του, το Kodi Home Theater παρουσιάζει επίσης μεγάλο εύρος εφαρμογής, καθώς διανέμεται για τα λειτουργικά συστήματα Android, Linux, BSD, macOS, iOS, tvOS και Windows [23]. Τέλος, αποτελεί μία εύχρηστη εφαρμογή με γραφικό περιβάλλον (GUI), στην οποία μπορεί να γίνει εύκολα έλεγχος για επιβεβαίωση ευρημάτων και εκμεταλλεύσιμων ευπαθειών.

2.5.2 Telegram Desktop

Όπως αναφέρεται και στον επίσημο ιστότοπο του [26], το Telegram Desktop αποτελεί μία εφαρμογή ανταλλαγής μηνυμάτων, η οποία επικεντρώνεται στην ταχύτητα και ασφάλεια, ενώ παράλληλα είναι απλή και διατίθεται δωρεάν. Μπορεί να χρησιμοποιηθεί σε διαφορετικές συσκευές (πλατφόρμες), ενώ έως σήμερα μετράει πάνω από 700 εκατομμύρια ενεργούς χρήστες και βρίσκεται στην κορυφαία δεκάδα εφαρμογών με τις περισσότερες λήψεις στον κόσμο. Τα παραπάνω χαρακτηριστικά μαζί με το γεγονός ότι αποτελεί ένα από τα κορυφαία έργα λογισμικού σε C++ όσον αφορά τις τάσεις του github για τον τελευταίο μήνα (κατά την περίοδο συγγραφής της διπλωματικής) [27] αποτελούν τους κύριους λόγους επιλογής του. Επιπλέον, είναι γραμμένο σε C++ σε ποσοστό 97% και ο κώδικας του μπορεί να βρεθεί στο github [27]. Αποτελεί επίσης ώριμο έργο λογισμικού, καθώς κυκλοφόρησε για πρώτη φορά τον Αύγουστο του 2013 [26].

Όπως παρατηρούμε από παραπάνω, τα δύο επιλεγέντα πακέτα λογισμικού παρουσιάζουν αρκετά κοινά ποιοτικά χαρακτηριστικά, τα οποίο είναι επιθυμητά για τις ανάγκες της μελέτης της παρούσας διπλωματικής εργασίας.

Κεφάλαιο 3

Εκτέλεση Εργαλείων Στατικού Ελέγχου

Αφού παρουσιάστηκε η απαραίτητη προαπαιτούμενη γνώση ώστε να γίνουν κατανοητοί οι λόγοι που οδήγησαν στην παρούσα μελέτη, αλλά και τα μέσα τα οποία επέτρεψαν την πραγματοποίηση της, ακολουθεί η μεθοδολογία που επιλέχτηκε κατά την εκτέλεση των πειραμάτων και την εξαγωγή δεδομένων από αυτά.

3.1 Εκτέλεση εργαλείων ανάλυσης κώδικα

Όσον αφορά το Kodi Home Theater, τα εργαλεία ανάλυσης κώδικα που περιγράφηκαν στις ενότητες 2.2, 2.3 εφαρμόστηκαν τόσο στο φάκελο αρχείων του kodi [23], όσο και στο φάκελο αρχείων που παράχθηκε κατά το 'χτίσιμο' της εφαρμογής (kodi-build) και είχαν ως σημεία εισόδου τους ίδιους φάκελους. Η ίδια διαδικασία ακολουθήθηκε για το Telegram, όπου τα αρχεία που παράχθηκαν κατά το 'χτίσιμο' του λογισμικού βρίσκονται στον φάκελο out. Σε αντίθεση με το Kodi τα αποτελέσματα για το Telegram παρουσιάζονται συμψηφισμένα, εφόσον στο φάκελο out υπήρχαν μόνο τρία ευρήματα (για το Flawfinder), από τα οποία μάλιστα το ένα είναι εκτός του θέματος της διπλωματικής και τα υπόλοιπα αποτελούν ψευδώς θετικά ευρήματα.

Επιπλέον, και τα δύο εργαλεία προσφέρουν επιλογές για πιο συγκεκριμένους ελέγχους ή φιλτράρισμα των αποτελεσμάτων με βάση επιλεγμένα CWEs, οι οποίες όμως δεν αξιοποιήθηκαν στην παρούσα διπλωματική.

3.1.1 Flawfinder

Το Flawfinder εκτελέστηκε μέσω της εντολής:

flawfinder –allowlink –followdotdir –neverignore –context .

> /results/flawfinder/resultsFlawfinder.txt

όπου:

–allowlink: Επιτρέπει τη χρήση symbolic links, οι οποίοι κανονικά παρακάμπτονται

–followdotdir: Είσοδος του εργαλείου σε κρυφούς για το χρήστη φακέλους για ανάλυση

–neverignore: Δεν αγνοείται κανένα ευρήμα ασφαλείας

–context: Εμφάνιση μεγαλύτερου περιεχομένου όσον αφορά το σφάλμα, χρήσιμο για ευκολότερη ανάγνωση από τον μηχανικό ασφαλείας

Τελεία (.): Υποδεικνύει ότι το εργαλείο εκτελείται στον παρόντα φάκελο

>: Εγγράφει την έξοδο των αποτελεσμάτων στο αρχείο /results/flawfinder/results-Flawfinder.txt

Αυτή η παραμετροποίηση έχει σκοπό μία κατά το δυνατόν ευρεία και σε βάθος χρήση του εργαλείου, παρότι η ανάλυση επικεντρώνεται στις ευπάθειες CWE-120 και CWE-190. Αυτό γιατί το Flawfinder έχει μικρούς χρόνους εκτέλεσης.

3.1.2 CppCheck

Με παρόμοιο τρόπο, το CppCheck εκτελέστηκε όπως φαίνεται παρακάτω:

cppcheck –enable=all –force –inconclusive –max-ctu-depth=10 –xml

–output-file= /results/cppcheck/resultCppCheck.xml -v .

όπου:

–enable=all: Ενεργοποιεί όλους τους ελέγχους

–force: Εκτελεί υποχρεωτικά έλεγχο όλων των παραμετροποιήσεων στα αρχεία

–inconclusive: Επιτρέπει στο εργαλείο την παρουσίαση σφαλμάτων ακόμα και όταν η ανάλυση είναι ασαφής. Ως αποτέλεσμα υπάρχει αύξηση στα ψευδώς θετικά ευρήματα, αλλά αυξάνονται οι πιθανότητες εύρεσης παραπάνω αληθώς θετικών

–max-ctu-depth=10: Αφορά το μέγιστο βάθος στο οποίο θα πραγματοποιηθεί η ανάλυση. Η προεπιλεγμένη τιμή είναι 2 και η ανάλυση επιβραδύνεται με αυτόν τον τρόπο, αυξάνοντας όμως την πιθανότητα αναγνώρισης παραπάνω σφαλμάτων

–xml: Εγγραφή των αποτελεσμάτων σε μορφή xml για βελτίωση της αναγνωσιμότητας

-output-file: Η διαδρομή στο αρχείο όπου θα αποθηκευτούν τα αποτελέσματα

-v: Παρουσίαση περισσότερων λεπτομερειών όσον αφορά τα ευρήματα

Τελεία (.): Υποδεικνύει ότι το εργαλείο εκτελείται στον παρόν φάκελο

Ομοίως με το Flawfinder, η παραμετροποίηση στοχεύει στην κατά το δυνατόν μεγιστοποίηση των ευρημάτων του εργαλείου.

3.1.3 CppCheck-GUI

Το CppCheck-GUI παραμετροποιήθηκε όπως παρουσιάζεται στη συνέχεια:

- Κατά τη φάση της ανάλυσης του λογισμικού, στην καρτέλα “Paths and Defines” ορίζεται ο φάκελος ή το σύνολο των φακέλων από όπου θα ξεκινήσει η ανάλυση.
- Στην καρτέλα “Analysis” και την ομώνυμη κατηγορία, επιλέγεται ο έλεγχος κάθε κλάσης για ασφαλή δημόσια διεπαφή (Check that each class has a safe public interface).

- Στην ίδια καρτέλα, αλλά στην κατηγορία “Limit analysis” (όρια ανάλυσης) επιλέγεται η ανάλυση του κώδικα στις κεφαλίδες (Check code in headers) και σε αχρησιμοποίητα πρότυπα (Check code in unused templates). Επίσης, ορίζεται το μέγιστο βάθος ανάλυσης (Max CTU depth) στην τιμή 10, όπως ορίστηκε και στο CppCheck. Στην ίδια καρτέλα, ο μέγιστος αριθμός αναδρομής στην προτυποποίηση των στιγμιότυπων (Max recursion in template instantiation) ορίστηκε στην τιμή 150.
- Τέλος, στην καρτέλα “Addons” και την ομώνυμη κατηγορία επιλέγεται η προέκταση που αφορά την ασφάλεια των νημάτων της διεργασίας (Thread safety).

3.2 Αποτελέσματα

Παρακάτω παρουσιάζονται τα αποτελέσματα των δύο εργαλείων όσον αφορά τις υπό εξέταση ευπάθειες CWE-120 και CWE-190, ενώ το παράρτημα Β περιέχει πίνακες με λεπτομέριες των ευρημάτων οι οποίες αφορούν τα αντίστοιχα αρχεία και τη γραμμή του κώδικα που αυτά βρέθηκαν. Μία πρώτη παρατήρηση είναι ότι το Flawfinder αναγνώρισε με επιτυχία ευπάθειες του τύπου CWE-120 (Buffer Overflow), ενώ δεν κατάφερε να αναγνωρίσει ευπάθειες του τύπου CWE-190 (Integer Overflow) για το ορισμένο επίπεδο επικινδυνότητας, οπότε τα αποτελέσματα επικεντρώνονται εκεί. Αντίθετα, τα CppCheck και CppCheck-GUI αναγνώρισαν επιτυχώς ευπάθειες του τύπου CWE-190, αλλά όχι CWE-120 και αντιμετωπίζονται αντίστοιχα.

3.2.1 Flawfinder

Όπως ειπώθηκε ήδη στην ενότητα 2.2 για το Flawfinder λαμβάνονται υπ΄ όψην τα επίπεδα επικινδυνότητας [3]-[5], με το [5] να εκφράζει το κρισιμότερο. Ακολουθούν οι πίνακες 3.1, 3.2, 3.3 οι οποίοι συγκεντρώνουν τα αποτελέσματα του εργαλείου τόσο για το Kodi Home Theater, αλλά και για το Telegram σε σχέση με την ευπάθεια CWE-120.

Kodi Home Theater

Τα αποτελέσματα που αφορούν το Kodi παρουσιάζονται στον πίνακα 3.1.

Kodi - CWE-120			
Συνάρτηση	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
gets()	[5]	9	Επιδιόρθωση προβλήματος με χρήση της συνάρτησης fgets()
sprintf()	[4]	17	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων sprintf_s() ή vsnprintf()
vsprintf()	[4]	1	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων sprintf_s(), sprintf() ή vsnprintf()
strcat()	[4]	9	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων strcat_s(), strncat(), strlcat() ή snprintf()
strcpy()	[4]	28	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων snprintf(), strcpy_s() ή strlcpy()
lstrcpyW()	[4]	1	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων snprintf(), strcpy_s() ή strlcpy()
sscanf()	[4]	2	Επιδιόρθωση με τον ορισμό ορίου για τη γραμματοσειρά που εισάγεται στο %s ή χρήση άλλης συνάρτησης εισόδου
realpath()	[3]	4	Ο buffer προορισμού πρέπει να έχει μέγεθος τουλάχιστον όσο το MAXPATHLEN και το όρισμα εισόδου πρέπει να ελεγχθεί ότι έχει μέγεθος το πολύ MAXPATHLEN

Πίνακας 3.1: Πίνακας αποτελεσμάτων του Flawfinder για το Kodi

Ακολουθούν τα αποτελέσματα για το Kodi-build στον πίνακα 3.2.

Kodi-build - CWE-120			
Συνάρτηση	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
strncat()	[5]	1	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων strcat_s(), strlcat() ή snprintf() ή αυτόματης αλλαγής μεγέθους του string
sprintf()	[4]	16	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων sprintf_s(), ή vsnprintf()
wcscpy()	[4]	6	Επιδιόρθωση με χρήση έκδοσης της συνάρτησης η οποία σταματάει να αντιγράφει στο τέλος του buffer
strcat()	[4]	5	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων strcat_s(), strncat(), strlcat() ή snprintf()
strcpy()	[4]	31	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων snprintf(), strcpy_s() ή strlcpy()
sscanf()	[4]	26	Επιδιόρθωση με τον ορισμό ορίου για τη γραμματοσειρά που εισάγεται στο %s ή χρήση άλλης συνάρτησης εισόδου
getopt()	[3]	27	Επιδιόρθωση με χρήση ασφαλούς έκδοσης
realpath()	[3]	1	Ο buffer προορισμού πρέπει να έχει μέγεθος τουλάχιστον όσο το MAXPATHLEN και το όρισμα εισόδου πρέπει να ελεγχθεί ότι έχει μέγεθος το πολύ MAXPATHLEN

Πίνακας 3.2: Πίνακας αποτελεσμάτων του Flawfinder για το Kodi-build

Telegram

Τέλος, παρουσιάζονται τα αποτελέσματα για το Telegram στον πίνακα 3.3. Σε αγκύλες αναγράφεται ο αριθμός των ευρημάτων τα οποία αφορούν το φάκελο 'χτισίματος' του λογισμικού (out).

Telegram - CWE-120			
Συνάρτηση	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
MultiByteToWideChar()	[5]	1	Απαιτεί το μέγιστο μήκος σε χαρακτήρες, ενώ δίνεται σε byte
sprintf()	[4]	7 [1]	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων sprintf_s(), ή vsnprintf()
strcat()	[4]	52	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων strcat_s(), strncat(), strlcat() ή snprintf()
strcpy()	[4]	102 [2]	Επιδιόρθωση προβλήματος με χρήση των συναρτήσεων snprintf(), strcpy_s() ή strlcpy()
sscanf()	[4]	8	Επιδιόρθωση με τον ορισμό ορίου για τη γραμματοσειρά που εισάγεται στο %s ή χρήση άλλης συνάρτησης εισόδου
scanf()	[4]	1	Επιδιόρθωση με τον ορισμό ορίου για τη γραμματοσειρά που εισάγεται στο %s ή χρήση άλλης συνάρτησης εισόδου
getopt()	[3]	1	Επιδιόρθωση με χρήση ασφαλούς έκδοσης
realpath()	[3]	1	Ο buffer προορισμού πρέπει να έχει μέγεθος τουλάχιστον όσο το MAXPATHLEN και το όρισμα εισόδου πρέπει να ελεγχθεί ότι έχει μέγεθος το πολύ MAXPATHLEN

Πίνακας 3.3: Πίνακας αποτελεσμάτων του Flawfinder για το Telegram

3.2.2 CppCheck

Για το CppCheck λαμβάνονται υπόψη τα επίπεδα επικινδυνότητας warning και error. Ακολουθούν οι πίνακες 3.4, 3.5, 3.6 παρόμοιας λογικής με τα αποτελέσματα της ανάλυσης του εργαλείου Flawfinder. Αυτή τη φορά, τα ευρήματα εστιάζουν στην ευπάθεια CWE-190, καθώς όπως έχει αναφερθεί στην ενότητα 3.2 δεν υπήρχαν ευρήματα για την CWE-120. Εδώ η ανάλυση δεν επικεντρώνεται σε συγκεκριμένες συναρτήσεις και έτσι το πεδίο "Συνάρτηση" αντικαθίσταται με το πεδίο "Υποκατηγορία Ευπάθειας".

Kodi Home Theater

Ακολουθούν τα αποτελέσματα για το Kodi στον πίνακα 3.4.

Kodi - CWE-190			
Υποκατηγορία Ευπάθειας	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
integerOverflow	error	3	Υπερχείλιση προσημασμένου ακεραίου σε έκφραση
integerOverflow Cond	warning	2	Χρήση περιττής κατάστασης ή υπερχείλιση προσημασμένου ακεραίου σε έκφραση

Πίνακας 3.4: Πίνακας αποτελεσμάτων του CppCheck για το Kodi

Στον πίνακα 3.5 εμφανίζονται τα αποτελέσματα για το Kodi-build.

Kodi-build - CWE-190			
Υποκατηγορία Ευπάθειας	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
integerOverflow	error	19	Υπερχείλιση προσημασμένου ακεραίου σε έκφραση
integerOverflow Cond	warning	10	Χρήση περιττής κατάστασης ή υπερχείλιση προσημασμένου ακεραίου σε έκφραση

Πίνακας 3.5: Πίνακας αποτελεσμάτων του CppCheck για το Kodi-build

Telegram

Τα αποτελέσματα που αφορούν το Telegram ακόλουθούν στον πίνακα 3.6.

Telegram - CWE-190			
Υποκατηγορία Ευπάθειας	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
integerOverflow	error	3	Υπερχείλιση προσημασμένου ακεραίου σε έκφραση

Πίνακας 3.6: Πίνακας αποτελεσμάτων του CppCheck για το Telegram

3.2.3 CppCheck-GUI

Όπως για το CppCheck, έτσι και εδώ λαμβάνονται υπ' όψη τα επίπεδα επικινδυνότητας warning και error. Ακολουθούν οι πίνακες 3.7, 3.8, 3.9 παρόμοια με το CppCheck.

Kodi Home Theater

Αρχικά, παρουσιάζονται τα αποτελέσματα σχετικά με το Kodi στον πίνακα 3.7.

Kodi - CWE-190			
Υποκατηγορία Ευπάθειας	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
integerOverflow	error	2	Υπερχείλιση προσημασμένου ακεραίου σε έκφραση
floatConversion Overflow	error	12	Ασαφής συμπεριφορά: Υπερχείλιση μετατροπής από δεκαδικό σε ακέραιο
safeInteger Overflow	error	54	Έλεγχοι ασφάλειας: Υπερχείλιση προσημασμένου ακεραίου σε έκφραση
IntegerOverflow Cond	warning	2	Χρήση περιττής κατάστασης ή υπερχείλιση προσημασμένου ακεραίου σε έκφραση

Πίνακας 3.7: Πίνακας αποτελεσμάτων του CppCheck-GUI για το Kodi

Ακολουθούν τα αποτελέσματα για το Kodi-build στον πίνακα 3.8.

Kodi-build - CWE-190			
Υποκατηγορία Ευπάθειας	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
integerOverflow	error	24	Υπερχείλιση προσημασμένου ακεραίου σε έκφραση
safeInteger Overflow	error	1	Έλεγχοι ασφάλειας: Υπερχείλιση προσημασμένου ακεραίου σε έκφραση
IntegerOverflow Cond	warning	10	Χρήση περιττής κατάστασης ή υπερχείλιση προσημασμένου ακεραίου σε έκφραση

Πίνακας 3.8: Πίνακας αποτελεσμάτων του CppCheck-GUI για το Kodi-build

Telegram

Τέλος, παρουσιάζονται τα αποτελέσματα για το Telegram στον πίνακα 3.9.

Telegram - CWE-190			
Υποκατηγορία Ευπάθειας	**Κρισιμότητα**	**Στιγμιότυπα**	**Σχόλιο**
integerOverflow	error	3	Υπερχείλιση προσημασμένου ακεραίου σε έκφραση
safeInteger Overflow	error	7	Έλεγχοι ασφάλειας: Υπερχείλιση προσημασμένου ακεραίου σε έκφραση

Πίνακας 3.9: Πίνακας αποτελεσμάτων του CppCheck-GUI για το Telegram

3.3 Σύγκριση Εργαλείων

Μετά από μία αρχική, υψηλού επιπέδου ανάλυση των αποτελεσμάτων των εργαλείων μπορούν να εξαχθούν ορισμένα βασικά συμπεράσματα. Αρχικά, όπως αναφέρθηκε ήδη στην ενότητα 3.2 το Flawfinder ήταν ικανό να αναγνωρίσει προβλήματα που σχετίζονται μόνο με την ευπάθεια buffer overflow (CWE-120), ενώ τα εργαλεία CppCheck αναγνωρίσαν μόνο ευπάθειες του τύπου υπερχείλισης ακεραίου (CWE-190). Επίσης, το Flawfinder χρησιμοποιεί πενταβάθμια κλίμακα κρισιμότητας των ευπαθειών, όπως ειπώθηκε στην ενότητα 2.2, ενώ τα εργαλεία

CppCheck κατηγοριοποιούν τις ευπάθειες ανάλογα με τη φύση τους, όπως έχει περιγραφεί στην ενότητα 2.3.

Επιπλέον, όπως παρατηρούμε από τα αποτελέσματα των πινάκων 3.1, 3.2, 3.3 το Flawfinder επικεντρώνεται στην αναγνώριση επίφοβων συναρτήσεων και την παρότρυνση χρήσης παρόμοιων, ασφλέστερων παραλλαγών τους, ενώ από τους πίνακες 3.4, 3.5, 3.6, 3.7, 3.8, 3.9 τα CppCheck αναγνωρίζουν κινδύνους βασιζόμενα στη δομή του κώδικα και συνοδεύονται από ένα σύντομο σχολιασμό σχετικά με τον κίνδυνο ο οποίος ενδεχομένως υπάρχει. Από το σύνολο των πινάκων παρατηρούμε επιπλέον ότι το σύνολο των ευρημάτων του Flawfinder (kodi:71, kodi-build:113, telegram:173) είναι πολύ μεγαλύτερο από αυτό του CppCheck (kodi:5, kodi-build:29, telegram:3), αλλά και του CppCheck-GUI (kodi:70, kodi-build:35, telegram:10). Αυτό είναι αναμενόμενο καθώς το Flawfinder στοχεύει στη μεγιστοποίηση των ευρημάτων, ενώ τα CppCheck επικεντρώνονται στον περιορισμό των ψευδώς θετικών.

Μια πρόσθετη πρατήρηση είναι ότι το Flawfinder εμφανίζει επαναλήψεις ευρημάτων, αλλά αυτές αναφέρονται σε όμοια αρχεία τα οποία αναλύθηκαν σε διαφορετικές διαδρομές καταλόγου, όπως παραδείγματος χάριν τα αρχεία στις θέσεις:

./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/pktdumper.c

./build/ffmpeg/src/ffmpeg/tools/pktdumper.c

Επαναλήψεις παρουσιάζουν και τα CppCheck, αλλά αυτές αναφέρονται στα ίδια αρχεία αυτήν τη φορά και οφείλονται στην αναγνώριση κάποιου σφάλματος με χρήση περισσότερων από μιας μεθόδου.

Συγκρίνοντας τις παραλλαγές του CppCheck παρατηρείται ότι για το Kodi Home Theater μέσω της γραμμής εντολών βρέθηκαν δύο επιπλέον ευρήματα κρισιμότητας σφάλματος (error) και κατηγορίας integerOverflow, ένα στο kodi και ένα στο kodi-build. Τα τελευταία δεν εντοπίστηκαν στην έκδοση που χρησιμοποιεί γραφικό περιβάλλον (GUI). Η έκδοση του γραφικού περιβάλλοντος διαφοροποιήθηκε από την έκδοση γραμμής εντολών, εντοπίζοντας 73 επιπλέον συνολικά σφάλματα. Από αυτά, τα 12 (6 μοναδικά) ήταν κατηγορίας floatConversionOverflow και τα 54 (53 μοναδικά) κατηγορίας safeIntegerOverflow. Τα τελευαία ανιχνεύτηκαν στον kodi. Τα υπόλοιπα 7 ανιχνεύτηκαν στον kodi-build. Τα έξι από αυτά ανήκουν στην κατηγορία integerOverflow και το άλλο στην κατηγορία safeIntegerOverflow.

Για το Telegram, η έκδοση που διαθέτει γραφικό περιβάλλον (GUI) παρουσίασε όλα τα σφάλματα τα οποία παρουσίασε κι η κλασική έκδοση και επιπλέον επτά κατηγορίας safeIntegerOverflow.

Κεφάλαιο 4

Ανάλυση Ευρημάτων Στατικού Ελέγχου

Το παρόν κεφάλαιο παρουσιάζει τα αναλυτικά αποτελέσματα όσον αφορά αληθώς και ψευδώς θετικά ευρήματα των δύο εργαλείων ανάλυσης λογισμικού. Συγκεριμένα, για το Flawfinder τα αποτελέσματα συνοψίζονται στους πίνακες 4.1, 4.2, 4.3, ενώ για το CppCheck στους πίνακες 4.4, 4.5, 4.6 και για το CppCheck-GUI στους πίνακες 4.7, 4.8, 4.11. Τα αποτελέσματα που παρουσιάζονται στους πίνακες έχουν αναλυθεί και επιβεβαιωθεί ωε προς την εγκυρότητά τους με χειροκίνητο τρόπο. Επίσης, προσδιορίζονται τα ευρήματα εντός θέματος της διπλωματικής ως όλα εκτός αυτών που αναφέρονται σε test ή tutorial, τα οποία δεν αναλύονται. Τέλος, παρουσιάζεται το σύνολο των ευρημάτων, των αληθώς/ψευδώς θετικών, των εκτός σκοπιάς, των κοινών ανάμεσα στα CppCheck, καθώς και ποσοστιαίες τιμές με στόχο την ευκολότερη εξαγωγή συμπερασμάτων από τα αποτελέσματα.

Χρειάζεται να ειπωθεί ότι ευρήματα τα οποία αφορούν ασφαλή κώδικα, ο οποίος όμως δυνητικά ενδέχεται να δημιουργήσει κάποιο πρόβλημα ασφάλειας αξιολογούνται ως αληθώς θετικά, καθώς μελλοντική χρήση τους σε άλλα μέρη του λογισμικού εξακολουθεί να κρύβει κινδύνους. Για παράδειγμα, τέτοιες περιπτώσεις αφορούν συναρτήσεις οι οποίες δεν περιλαμβάνουν επαρκείς ελέγχους σχετικά με τα ορίσματα που δέχονται. Αυτές μπορεί στις υπάρχουσες περιπτώσεις να χρησιμοποιούνται ασφαλώς από το λογισμικό, αλλά η αδυναμία υπάρχει και σε μελλοντική εξέλιξη του λογισμικού δύναται να χρησιμοποιηθούν με ανασφαλή τρόπο. Επίσης, η συντριπτική πλειοψηφία των αληθώς θετικών αποτελεσμάτων αφορά τέτοιες περιπτώσεις. Δηλαδή, ανασφαλείς δομές κώδικα οι οποίες χρησιμοποιούνται με ασφάλεια από το λογισμικό και άρα πρακτικά δεν δύναται να εκμεταλλευθούν.

4.1 Ανάλυση Αποτελεσμάτων Flawfinder

Όπως έχει ήδη ειπωθεί στην ενότητα 2.2 το Flawfinder αναγνωρίζει επικίνδυνες συναρτήσεις και για αυτό οι παρακάτω πίνακες προσδιορίζονται με βάση αυτή τη δυνατότητα. Επίσης, στους πίνακες 4.2, 4.3 με αγκύλες '[#]' παρουσιάζονται τα ευρήματα που έχουν ανιχνευτεί ήδη σε ίδιο αρχείο το οποίο βρίσκεται σε άλλη τοποθεσία, και για αυτόν το λόγο δε λαμβάνονται υπόψη. Με κόκκινο σε αυτές τις περιπτώσεις αναγράφονται τα ευρήματα, χωρίς την ύπαρξη διπλότυπων, τα οποία θεωρούνται και τα τελικά ευρήματα που θα ληφθούν υπόψη.

4.1.1 Kodi

Kodi - CWE-120				
Συνάρτηση	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**
gets()	4	5	4	0
sprintf()	10	7	4	6
vsprintf()	0	1	0	0
strcat()	9	0	0	9
strcpy()	28	0	5	23
lstrcpyW()	1	0	0	1
sscanf()	2	0	2	0
realpath()	4	0	0	4
Σύνολο	**58**	**13**	**15 | 25,86%**	**43 | 74,14%**

Πίνακας 4.1: Ανάλυση ευρημάτων του Flawfinder για το Kodi

Από τον πίνακα 4.1 αρχικά παρατηρείται ότι έχουν αναγνωριστεί 8 διαφορετικές επικίνδυνες συναρτήσεις, από τις οποίες βέβαια η vsprintf συναντάται μόνο σε εκτός σκοπιάς

κώδικα και άρα δεν λαμβάνεται υπόψη. Επίσης, το εύρος με βάση τον αριθμό των ευρημάτων που αναφέρονται σε συγκεκριμένη συνάρτηση κυμαίνεται από μία ή δύο και φτάνει μέχρι 28 για την strcpy, ενώ ο συνολικός αριθμός ευρημάτων είναι 58. Αληθώς θετικά ευρήματα βρέθηκαν σε 4 από τις 7 συναρτήσεις, με συνολικό αριθμό 15 το οποίο αποτελεί το 25,86% του συνόλου των ευρημάτων. Από την άλλη, 43 ψευδώς θετικά παρουσιάζονται σε 5 από τις 7 συναρτήσεις, το οποίο αποτελεί το 74,14% του συνόλου. Η αναλογία 25,86/74,14% έρχεται σε συμφωνία με τη φιλοσοφία του εργαλείου το οποίο εστιάζει σε μεγιστοποίηση των ευρημάτων με κόστος το υψηλό ποσοστό ψευδώς θετικών αποτελεσμάτων. Σημειώνεται ότι εκτός σκοπιάς αποτελέσματα βρέθηκαν σε 3 από τις συνολικά 8 συναρτήσεις με συνολικό αριθμό 13.

4.1.2 Kodi-Build

Kodi-Build - CWE-120				
Συνάρτηση	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**
strncat()	1	0	0	1
sprintf()	5	11	4	1
wcscpy()	6 [3] | 3*	0	1	2
strcat()	5 [1] | 4*	0	0	4
strcpy()	28 [11] | 17*	3	0	17
sscanf()	26 [13] | 13*	0	11	2
getopt()	17 [8] | 9*	10 [4] | 6*	0	9
realpath()	1	0	0	1
Σύνολο	**53**	**20**	**16 | 30,19%**	**37 | 69,81%**

*(σύνολο [διπλότυπα] | μοναδικά)

Πίνακας 4.2: Ανάλυση ευρημάτων του Flawfinder για το Kodi-build

Στον πίνακα 4.2 το σύνολο των συναρτήσεων είναι 8 (διαφορετικές με τον πίνακα 4.1), με όλες να περιλαμβάνουν ευρήματα εντός σκοπιάς αυτή τη φορά. Τα ευρήματα σε 5 από τις 8 συναρτήσεις περιλαμβάνουν διπλότυπα αρχεία, με το συνολικό πλήθος επαναλαμβανόμενων ευρημάτων να είναι 36. Το εύρος σε αυτήν την περίπτωση περιλαμβάνει από 1 εύρημα μέχρι και 17, με τον συνολικό πλήθος τους να είναι 53. Αληθώς θετικά ευρήματα παρουσιάζονται σε 3 συναρτήσεις με συνολικό πλήθος 16, τα οποία αποτελούν το 30,19% των συνολικών ευρημάτων. Από την άλλη, ψευδώς θετικά παρατηρήθηκαν σε μικρότερο ή μεγαλύτερο πλήθος σε όλες τις συναρτήσεις οι οποίες ανιχνεύτηκαν, με τον συνολικό αριθμό τους να είναι 37, δηλαδή το 69,81% όλων των ευρημάτων. Και εδώ, παρότι μικρότερη, η αναλογία 30,19/69,81% συμβαδίζει με τη λογική των πολλών ψευδώς θετικών αποτελεσμάτων. Τα εκτός σκοπιάς ευρήματα για το συγκεκριμένο πακέτο λογισμικού ήταν 20 τα οποία εντοπίστηκαν σε 3 από τις 8 συναρτήσεις.

4.1.3 Telegram

Telegram - CWE-120				
Συνάρτηση	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**
MultiByteTo WideChar()	1	0	0	1
sprintf()	5	2 [1] | 1*	1	4
strcat()	52	0	20	32
strcpy()	102	0	22	80
sscanf()	8	0	1	7
scanf()	1	0	1	0
getopt()	1	0	0	1
realpath()	1	0	0	1
Σύνολο	**171**	**1**	**45 | 26,32%**	**126 | 73,68%**

*(σύνολο [διπλότυπα] | μοναδικά)

Πίνακας 4.3: Ανάλυση ευρημάτων του Flawfinder για το Telegram

Όπως και στις προηγούμενες περιπτώσεις, στον πίνακα 4.3 που αναφέρεται στο Telegram, το σύνολο των συναρτήσεων οι οποίες ανιχνεύτηκαν είναι 8. Τα στιγμιότυπα κάποιας συνάρτησης μέσα στο έργο λογισμικού κυμαίνονται από 1, η οποία αφορά 4 συναρτήσεις, μέχρι και 102 οι οποίες παρατηρήθηκαν για τη συνάρτηση strcpy. Επίσης, 6 από τις συναρτήσεις παρουσιάζουν μέχρι και 5 στιγμιότυπα, με μόνες διαφοροποιήσεις την προαναφερόμενη strcpy και την strcat η οποία παρουσιάζεται σε 52 ευρήματα.Όπως φαίνεται στον πίνακα 4.3 ο συνολικός αριθμός ευρημάτων είναι 171, με τα αληθώς θετικά να παρατηρούνται σε 5 συναρτήσεις με συνολικό πλήθος 45, το οποίο αποτελεί το 26,32% των ευρημάτων. Ψευδώς θετικά ευρήματα εμφανίστηκαν σε όλες τις συναρτήσεις, εκτός της scanf, με συνολικό πλήθος 126, δηλαδή το 73,68% των ευρημάτων. Η αναλογία αυτή, συγκεκριμένα 26,32/73,68%, για τον λόγο που αναπτύχθηκε στην ενότητα 2.2 ήταν αναμενόμενη.

Να σημειωθεί ότι κατά την ανάλυση του λογισμικού προέκυψε ότι όλα τα αληθώς θετικά ευρήματα θα μπορούσαν να διορθωθούν με τουλάχιστον μία από τις προτεινόμενες λύσεις που προτείνει το Flawfinder. Η πλειοψηφία αυτών των λύσεων αφορά την αντικατάσταση των συναρτήσεων με παρόμοιες, ασφαλέστερες εκδόσεις.

4.2 Ανάλυση Αποτελεσμάτων CppCheck

Το CppCheck κατηγοριοποιεί τα ευρήματα με βάση υποκατηγορίες βασικών ευπαθειών, όπως αυτές οι οποίες αναλύονται στην παρούσα διπλωματική. Έτσι, τα ευρήματα σε αυτήν την περίπτωση ταξινομούνται με βάση αυτές τις υποκατηγορίες. Συγκεκριμένα, τα ευρήματα σε αγκύλες '[#]' αφορούν ευπάθειες οι οποίες έχουν ήδη ανιχνευτεί κάνοντας χρήση τιμής αντίθετου προσήμου από το CppCheck. Για παράδειγμα, μπορεί να υπάρχει ευπάθεια όταν ένα όρισμα πάρει την τιμή 2147483647, αλλά και όταν πάρει την τιμή -2147483647. Επειδή αυτά τα ευρήματα αφορούν το ίδιο κενό ασφαλείας, λαμβάνονται υπόψη σαν ένα εύρημα. Αντίστοιχα με πριν, ο τελικός αριθμός των ευρημάτων τα οποία λαμβάνονται υπόψη επισημαίνεται με κόκκινο χρώμα. Τέλος, λόγω προφανών ομοιοτήτων, αναφέρεται και ο αριθμός των κοινών ευρημάτων κάθε υποκατηγορίας με το CppCheck-GUI.

4.2.1 Kodi

Kodi - CWE-190					
Υποκατηγορία	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**	**Κοινά Με GUI**
IntegerOverflow	3	0	0	3	2
IntegerOverflow Cond	2	0	0	2	2
Σύνολο	**5**	**0**	**0 | 0%**	**5 | 100%**	**4 | 80%**

Πίνακας 4.4: Ανάλυση ευρημάτων του CppCheck για το Kodi

Από τα αποτελέσματα του πίνακα 4.4 βλέπουμε ότι ευπάθειες βρέθηκαν μόνογια δύο υποκατηγορίες. Συγκεκριμένα, ο συνολικός αριθμός των ευρημάτων είναι 5, ενώ χρειάζεται να σημειωθεί ότι το σύνολο τους αφορά ψευδώς θετικά. Τέλος, τα 4 από αυτά (80%) βρέθηκαν επίσης από το CppCheck-GUI.

4.2.2 Kodi-Build

Kodi-Build - CWE-190					
Υποκατηγορία	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**	**Κοινά Με GUI**
IntegerOverflow	19 [4] | 15*	0	3	12	18 [4] | 14*
IntegerOverflow Cond	10	0	0	10	10
Σύνολο	**25**	0	**3 | 12%**	**22 | 88%**	**24 | 96%**

*(σύνολο [διπλότυπα] | μοναδικά)

Πίνακας 4.5: Ανάλυση ευρημάτων του CppCheck για το Kodi-build

Από τον πίνακα 4.5 είναι φανερό ότι και στον κώδικα ο οποίος βρίσκεται στο kodi-build οι υποκατηγορίες που ανιχνεύτηκαν είναι οι ίδιες με προηγουμένως, αλλά εδώ το σύνολο των ευρημάτων είναι 25, με τέσσερα από αυτά να είναι διπλότυπα. Τα αληθώς θετικά, με πλήθος 3 ή το 12% των συνολικών ευρημάτων, αφορούν μόνο στην υποκατηγορία IntegerOverflow. Αντιθέτα, τα ψευδώς θετικά είναι 22, δηλαδή το 88% των συνολικών ευρημάτων. Παρατηρείται επιπλέον ότι τα κοινά ευρήματα με το CppCheck-GUI είναι 24, αποτελώντας το 96% των συνολικών ευρημάτων.
Αξίζει να σημειωθεί ότι από το σύνολο των ευρημάτων (30), τόσο σε kodi αλλά και kodi-build, μόνο 2 δεν εντοπίστηκαν και από το CppCheck-GUI.

4.2.3 Telegram

Telegram - CWE-190					
Υποκατηγορία	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**	**Κοινά Με GUI**
IntegerOverflow	3	0	0	3	3
Σύνολο	**3**	**0**	**0 | 0%**	**3 | 100%**	**3 | 100%**

Πίνακας 4.6: Ανάλυση ευρημάτων του CppCheck για το Telegram

Με αναφορά στον πίνακα 4.6, το CppCheck σε αυτήν την περίπτωση ανίχνευσε μόνο 3 υποψήφιες ευπάθειες, όλες κατηγορίας IntegerOverflow. Επίσης, όπως και στην

περίπτωση του Kodi όλες οι εν λόγω ευπάθειες αποτελούν ψευδώς θετικά ευρήματα. Τέλος, όλα τα προαναφερόμενα ευρήματα αναγνωρίστηκαν και από το CppCheck-GUI.

4.3 Ανάλυση Αποτελεσμάτων Cppcheck-GUI

Οι πίνακες αποτελεσμάτων 4.7, 4.8 και 4.11 οι οποίοι αφορούν το CppCheck-GUI ακολουθούν παρόμοια λογική με αυτή που περιγράφηκε για το CppCheck.

4.3.1 Kodi

Kodi - CWE-190					
Υποκατηγορία	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**	**Κοινά Με CLI**
IntegerOverflow	2	0	0	2	2
FloatConversion Overflow	12 [6] | 6*	0	6	0	0
SafeInteger Overflow	50	4	48	2	0
IntegerOverflow Cond	2	0	0	2	2
Σύνολο	**60**	**4**	**54 | 90%**	**6 | 10%**	**4 | 6.67%**

*(σύνολο [διπλότυπα] | μοναδικά)

Πίνακας 4.7: Ανάλυση ευρημάτων του CppCheck-GUI για το Kodi

Συγκεκριμένα, ο πίνακας 4.7 δείχνει ότι το CppCheck-GUI αναγνώρισε 4 υποκατηγορίες ευπαθειών, οι οποίες περιλαμβάνουν συνολικά 60 ευρήματα. Αυτό έρχεται σε αντίθεση με το CppCheck, το οποίο εντόπισε συνολικά μόνο 5 ευπάθειες. Στα 60 αυτά ευρήματα εμπεριέχονται και 6 τα οποία έχουν εντοπιστεί ήδη και άρα δεν αναλύονται ξανά στην παρούσα ενότητα. Οι υποκατηγορίες IntegerOverflow και IntegerOverflowCond, εμπεριέχουν μόνο 4 ευρήματα, κοινά με το CppCheck, τα οποία όπως έχει αναφερθεί στην ενότητα 4.2 είναι ψευδώς θετικά. Εκτός αυτών, δύο ακόμα ευρήματα της κατηγορίας safeIntegerOverflow αποτελούν ψευδώς θετικά ευρήματα, αυξάνοντας των συνολικό αριθμό τους σε 6. Όλα τα υπόλοιπα ταξινομούνται ως αληθώς θετικά με συνολικό πλήθος 54. Έτσι, τα αληθώς θετικά αποτελούν το 90% των συνολικών ευρημάτων, ενώ τα ψευδώς το υπολειπόμενο 10%. Το ποσοστό των κοινών αποτελεσμάτων με το CppCheck είναι 6,67%, ενώ 4 από τα ευρήματα αφορούν κώδικα εκτός σκοπιάς

και παραλείπονται. Σε αντίθεση με τα απότελέσματα του CppCheck στον πίνακα 4.4 η αναλογία αληθώς/ψευδώς θετικών είναι 90/10%. Αυτό έρχεται σε συμφωνία με τη στρατηγική που ακολουθεί το συγκεκριμένο εργαλείο για ελαχιστοποίηση των ψευδώς θετικών αποτελεσμάτων.

4.3.2 Kodi-Build

Kodi-Build - CWE-190					
Υποκατηγορία	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**	**Κοινά Με CLI**
IntegerOverflow	24 [4] | 20*	0	9	11	18 [4] | 14*
SafeInteger Overflow	0	1	0	0	0
IntegerOverflow Cond	10	0	0	10	10
Σύνολο	**30**	**1**	**9 | 30%**	**21 | 70%**	**24 | 80%**

*(σύνολο [διπλότυπα] | μοναδικά)

Πίνακας 4.8: Ανάλυση ευρημάτων του CppCheck-GUI για το Kodi-build

Αναφορικά με το kodi-build, τα αποτελέσματα του οποίου παρουσιάζονται στον πίνακα 4.8, το CppCheck-GUI ανίχνευσε 3 υποκατηγορίες ευπαθειών εκ των οποίων η safeIntegerOverflow δεν λαμβάνεται υπόψη καθώς το μοναδικό της εύρημα αφορά κώδικα εκτός σκοπιάς. Τα υπόλοιπα ευρήματα είναι 30 συνολικά, με 4 από αυτά να αναφέρονται σε διπλότυπα. Τα 9 αληθώς θετικά ευρήματα παρουσιάζονται μόνο στην κατηγορία IntegerOverflow, πλήθος που αποτελεί το 30% όλων των ευρημάτων. Τα υπόλοιπα 21 ευρήματα είναι ψευδώς θετικά, αποτελώντας το 70% του συνόλου. Επίσης, 24 από τα ευρήματα ανιχνεύθηκαν και από το CppCheck, το οποίο αποτελεί το 80% του συνόλου, ενώ εντύπωση προκαλεί η παρατήρηση ότι το σύνολο των ψευδώς θετικών αποτελεσμάτων (100%) είναι κοινά με το CppCheck.

Υπόθεση Αποτελεσμάτων CppCheck

Παρατηρώντας τα αποτελέσματα των εργαλείων CppCheck και GUI στους πίνακες 4.4, 4.5, 4.7 και 4.8 μπορεί κανείς εύκολα να διακρίνει ότι το CppCheck εμφάνισε εξαιρετικά υψηλό ποσοστό ψευδώς θετικών αποτελεσμάτων, ξεπερνώντας ακόμα και το Flawfinder, για το οποίο τα αποτελέσματα ήταν αναμενόμενα. Κάτι άλλο που παρατηρείται είναι το μικρό πλήθος των συνολικών ευρημάτων, όχι μόνο σε σχέση με το

Flawfinder, αλλά και με το CppCheck-GUI (αλλά και το ιδιαίτερα υψηλό ποσοστό κοινών ευρημάτων με το τελευταίο). Όσον αφορά το gui, αυτό παρουσιάζει πολύ καλό ποσοστό αληθώς θετικών ευρημάτων και παράλληλα χαμηλή αναλογία κοινών ευρημάτων με το CppCheck για τον κώδικα που υπάρχει στο kodi. Αντίθετα, το ποσοστό αληθώς θετικών για το kodi-build είναι αμφιλεγόμενο με βάση τη λειτουργικότητα του εργαλείου, αλλά παράλληλα παρατηρείται ότι το ποσοστό κοινών ευρημάτων με το CppCheck σε αυτήν την περίπτωση είναι ιδιαίτερα υψηλό.

Οι παραπάνω παρατηρήσεις θα μπορούσαν να εξηγηθούν σε συνδυασμό και με τη δημοτικότητα του εργαλείου, εάν γινόταν η υπόθεση ότι έχει γίνει ήδη χρήση του CppCheck για αντιμετώπιση προβλημάτων ασφαλείας στο έργο λογισμικού. Καταρχάς, αυτό θα εξηγούσε τόσο το μικρό πλήθος των ανιχνευμένων ευρημάτων, αλλά και το υψηλότατο ποσοστό ψευδώς θετικών, αφού σε περίπτωση ελέγχου θα ήταν αναμενόμενο να διορθωθούν τα αληθώς θετικά ευρήματα, και τα υπόλοιπα να παραμείνουν ως έχει αν κριθούν ασφαλή. Επιπλέον, σε μια τέτοια περίπτωση, έρχεται να προστεθεί το γεγονός ότι όπως ειπώθηκε ήδη στην ενότητα 4.3 το CppCheck-GUI στο kodi παρουσιάζει χαμηλό ποσοστό κοινών ευρημάτων, ενώ στο kodi-build ισχύει το αντίθετο, στοιχείο το οποίο φαίνεται να ανταναικλάται στις αναλογίες αληθώς/ψευδώς ευρημάτων. Παράλληλα, η μεγάλη πλειοψηφία των κοινών ευρημάτων απαρτίζεται από ψευδώς θετικά.

Οι παραπάνω παρατηρήσεις αποτελούν κίνητρο να μελετηθεί περαιτέρω η παραπάνω υπόθεση και να διερευνηθεί το πώς θα μεταβληθούν τα αποτελέσματα του CppCheck-GUI εάν αποκλειστούν αυτά τα οποία είναι κοινά με το CppCheck. Η λογική πίσω από αυτήν την απόφαση είναι απλή και εξακολουθεί να βασίζεται στην αρχική υπόθεση: «Εάν έχουν ήδη ανιχνευτεί και επιδιορθωθεί τα αληθώς θετικά ευρήματα, τότε δεν θα περιλαμβάνονται στα αποτελέσματα, οπότε υπό το ίδιο πρίσμα ενδιαφέρον θα είχε να μελετήσουμε τα αποτελέσματα χωρίς να λάβουμε υπόψη ούτε τα ψευδώς θετικά, τα οποία πιθανώς απλά αφέθηκαν ως είχαν».

Ακολουθούν οι συγκριτικοί πίνακες 4.9 και 4.10 τόσο για το kodi, όσο και για το kodi-build, σχετικά με τα αποτελέσματα, συμπεριλαμβανομένων των κοινών με το cli, αλλά και χωρίς.

Σύγκριση Kodi Με/Χωρίς Κοινά Ευρήματα			
Κοινά	**Εντός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**
Με Κοινά	60	54 | 90%	6 | 10%
Χωρίς Κοινά	56	54 | 96,43%	2 | 3,57%

Πίνακας 4.9: Σύγκριση Αληθώς/Ψευδώς Θετικών συμπεριλαμβανομένων και μη κοινών ευρημάτων για το Kodi

Σύγκριση Kodi-Build Με/Χωρίς Κοινά Ευρήματα			
Κοινά	**Εντός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**
Με Κοινά	30	9 | 30%	21 | 70%
Χωρίς Κοινά	6	6 | 100%	0 | 0%

Πίνακας 4.10: Σύγκριση Αληθώς/Ψευδώς Θετικών συμπεριλαμβανομένων και μη κοινών ευρημάτων για το Kodi-Build

Όπως φαίνεται από τους πίνακες 4.9 και 4.10 παρότι τα ποσοστά του kodi αυξάνονται, το πιο αξιοσημείωτο είναι αυτό που παρατηρείται στο kodi-build. Από τη μία τα ευρήματά του μειώνονται σημαντικά σε πλήθος, αλλά παράλληλα από αυτά που αφαιρέθηκαν τα 21 είναι το σύνολο των ψευδώς θετικών, ενώ μόνο 3 χαρακτηρίζονται ως αληθώς θετικά, καταλήγοντας σε τέλειο ποσοστό. Λόγω της υπόθεσης στην οποία βασίζεται η παρούσα ανάλυση δεν είναι ορθό παρόλα αυτά να θεωρηθεί ότι η απόδοση του CppCheck-GUI είναι βέλτιστη, καθώς το ζητούμενο ήταν διαφορετικό και το baseline περιγράφηκε με βάση αυτό. Επιπλέον, η παραπάνω ανάλυση δεν αποδεικνύει την υπόθεση, απλά ενισχύει το επιχείρημα υπέρ της, καθώς αυτό θα εξηγούσε όλες τις ιδιαιτερότητες που παρουσιάζονται τόσο στα αποτελέσματα του CppCheck (λίγα ευρήματα, πολύ υψηλά ποσοστά ψευδώς θετικών), όσο και του CppCheck-GUI (περισσότερα κοινά με CppCheck → αυξημένα ψευδώς θετικά, χαμηλά ποσοστά αληθώς θετικών στο kodi-build).

4.3.3 Telegram

Telegram - CWE-190					
Υποκατηγορία	**Εντός Σκοπιάς**	**Εκτός Σκοπιάς**	**Αληθώς Θετικά**	**Ψευδώς Θετικά**	**Κοινά Με GUI**
IntegerOverflow	3	0	0	3	3
SafeInteger Overflow	7	0	6	1	0
Σύνολο	**10**	**0**	**6 | 60%**	**4 | 40%**	**3 | 30%**

Πίνακας 4.11: Ανάλυση ευρημάτων του CppCheck-GUI για το Telegram

Τέλος, όπως παρατηρούμε στον πίνακα 4.11, εκτός της IntegerOverflow, το εργαλείο αναγνώρισε επιπλέον 7 υποψήφιες ευπάθειες του τύπου SafeIntegerOverflow, εκ των οποίων οι 6 εάν χρησιμοποιηθούν εσφαλμένα δύναται να δημιουργήσουν υπερχείλιση. Τα ευρήματα κατηγορίας IntegerOverflow, είναι όμοια με αυτά που ανίχευσε το CppCheck, αφήνοντας ανοιχτό το ενδεχόμενο η προηγούμενα αναφερόμενη υπόθεση που αναπτύχθηκε στην ενότητα 4.3 να εφαρμόζεται και εδώ. Έτσι, η αναλογία αληθώς/ψευδώς θετικών ευρημάτων είναι 60/40, ενώ αυξάνεται σε 85,71/14,29% εάν δεν ληφθούν υπόψη τα κοινά με το CppCheck.

Κεφάλαιο 5

Συμπεράσματα και Προτάσεις για Μελλοντική Εργασία

5.1 Συμπεράσματα και μελλοντική εργασία

Συνοψίζοντας, η παρούσα μελέτη είχε ως στόχο την αυτοματοποιημένη στατική ανάλυση κώδικα δύο δημοφιλών και ώριμων πακέτων λογισμικού ανοιχτού κώδικα, υλοποιημένων σε C++. Αυτό έγινε με χρήση δημοφιλών, ανοιχτού κώδικα εργαλείων στατικής ανάλυσης. Συγκεκριμένα, τα πακέτα λογισμικού που επιλέχτηκαν είναι το Kodi Home Theater και το Telegram, ενώ η μελέτη επικεντρώθηκε στην αναγνώριση και ανάλυση ευρημάτων τα οποία αφορούν τις κλασικές ευπάθειες Buffer Overflow (CWE-120) και Integer Overflow (CWE-190). Τέτοιου είδους ευπάθειες συναντιούνται συχνά και η επιδιόρθωση τους θεωρείται κρίσιμη, καθώς η αδυναμία διαχείρισης τους μπορεί να έχει σοβαρές συνέπειες για το λογισμικό. Ως εργαλεία ανάλυσης επιλέχτηκαν τα Flawfinder, CppCheck και CppCheck-GUI, αφού με βάση τη βιβλιογραφία παρουσίασαν τις καλύτερες επιδόσεις σε σχέση με τις συγκεκριμένες κατηγορίες ευπαθειών, σε ειδικά διαμορφωμένες σουίτες λογισμικού με στόχο τον έλεγχο των επιδόσεων τους.

5.1.1 Συμπεράσματα

Μία παρατήρηση που διατυπώθηκε από νωρίς ήταν ότι ενώ και τα δύο εργαλεία είναι σε θέση να αναγνωρίσουν ευπάθειες και από τις δύο προαναφερόμενες κατηγορίες ευπαθειών, τόσο το Flawfinder όσο και τα CppCheck μπόρεσαν να αναγνωρίσουν κρίσιμες ευπάθειες μόνο από μία κατηγορία. Έτσι, το Flawfinder είχε ευρήματα όσον αφορά την κατηγορία CWE-120, ενώ τα CppCheck αναγνώρισαν ευπάθειες του τύπου CWE-190. Παράλληλα, το Flawfinder είχε ευρήματα της κατηγορίας CWE-190 και τα CppCheck της κατηγορίας CWE-120, αλλά χαμηλότερης κρισιμότητας, τα οποία δεν αναλύθηκαν περαιτέρω σύμφωνα με τη στοχοθεσία της διπλωματικής. Το συμπέρασμα που προκύπτει είναι ότι ενώ θεωρητικά τα εργαλεία ανάλυσης λογισμικού έχουν τη δυνατότητα

να αναγνωρίσουν ευπάθειες και των δύο κατηγοριών, πρακτικά η αποτελεσματικότητα τους εξαρτάται από την κατηγορία της ευπάθειας την οποία ανιχνεύουν.

Επιπλέον, η χειροκίνητη ανάλυση των ευρημάτων επιβεβαίωσε σε μικρότερο ή μεγαλύτερο βαθμό την τακτική που ακολουθεί το κάθε εργαλείο ανάλυσης λογισμικού. Έτσι, το Flawfinder παρουσίασε μεγάλο αριθμό συνολικών ευρημάτων, με υψηλό ποσοστό ψευδώς θετικών, ενώ το CppCheck παρότι είχε αρκετά μικρότερο αριθμό συνολικών ευρημάτων, είχε παράλληλα πολύ χαμηλότερο ποσοστό ψευδώς θετικών.

Όσον αφορά τα αληθώς θετικά ευρήματα, αυτά αποτελούν υπάρχοντα μη εκμεταλλεύσιμα όμως κενά υπερχείλησης, είτε λόγω χρήσης μη ασφαλών συναρτήσεων της C++, είτε μη ύπαρξης ελέγχων οι οποίοι θα προστάτευαν από την πραγματοποίηση υπερχείλισης. Τα συγκεκριμένα κενά ασφάλειας καθίστανται πρακτικά μη εκμεταλλεύσιμα, καθώς οι δομές λογισμικού που τα περιλαμβάνουν χρησιμοποιούνται με ασφάλεια και στα δύο αναλυθέντα πακέτα λογισμικού. Παρόλα αυτά, μελλοντική λανθασμένη χρήση του επιρρεπή σε σφάλματα υπερχείλησης κώδικα ενδέχεται να ενεργοποιήσει κάποια ευπάθεια, η οποία μπορεί να τύχει εκμετάλλευσης από επιτιθέμενους. Σε κάθε περίπτωση, κάθε αληθώς ανιχνευθείσα αδυναμία του λογισμικού θα πρέπει να διορθώνεται, ειδικά σε περιπτώσεις όπου η διόρθωσή της έχει μικρό κόστος. Έτσι, τα ευρήματα του Flawfinder μπορούν να εξαλειφθούν εύκολα με χρήση ασφαλέστερων παραλλαγών των ανιχνευμένων επικίνδυνων συναρτήσεων, οι οποίες μάλιστα συνήθως προτείνονται από το ίδιο το εργαλείο. Επιπλέον, στην πλειοψηφία τους, τα αληθώς θετικά ευρήματα του CppCheck αφορούν ενδεχόμενες υπερχειλίσεις οι οποίες θα μπορούσαν να αποτραπούν με απλούς ελέγχους του μεγέθους των ορισμάτων που δέχονται οι διάφορες συναρτήσεις του κώδικα.

Επιπλέον, το γεγονός ότι δεν παρατηρήθηκε κάποια εκμεταλλεύσιμη ευπάθεια, η οποία θα μπορούσε να επηρεάσει αρνητικά την ασφάλεια του λογισμικού στο σύνολό του, προσδίδεται στο γεγονός ότι και τα δύο πακέτα λογισμικού που εξετάστηκαν είναι σε ώριμο σημείο ανάπτυξης. Επιπρόσθετα, η δημοτικότητα και των δύο εξετασθέντων πακέτων λογισμικού αναντίρρητα επηρεάζει θετικά την ποιότητα του κώδικα και την προσοχή στη λεπτομέρεια.

Το CppCheck-GUI φαίνεται να παρέχει ελεφρά καλύτερο επίπεδο ασφάλειας συγκριτικά με την κλασική έκδοση που βασίζεται σε γραμμή εντολών. Με βάση τα αποτελέσματα, αυτό συμβαίνει καθώς συνήθως περιλαμβάνει τη μαζική πλειοψηφία των ευρημάτων του τελευταίου, και επιπλέον ευρήματα τα οποία δεν ήταν σε θέση να αναγνωρίσει. Αυτό επιδέχεται περαιτέρω μελέτης, καθώς υπάρχει το ενδεχόμενο οι συγκεκριμένες εκδόσεις του ίδιου λογισμικού να έχουν ελεγχθεί μόνο με την έκδοση γραμμής εντολών λόγω της δημοτικότητας της.

Παρότι κατά την εξαγωγή των αποτελεσμάτων δεν ακολουθήθηκε κάποια μεθοδολογία για την αντικειμενική εξαγωγή μετρικών και συμπερασμάτων σε σχέση με την ταχύτητα των εργαλείων στην ανάλυση κώδικα, εμπειρικά, το συμπέρασμα είναι ότι το

Flawfinder ήταν σημαντικά ταχύτερο των εκδόσεων του CppCheck. Συγκεκριμένα, το πρώτο εργαλείο ολοκλήρωσε την ανάλυση σε λίγα λεπτά και για τα δύο αναλυθέντα πακέτα λογισμικού, ενώ ο αντίστοιχος χρόνος για τις εκδόσεις του CppCheck ήταν τάξης μεγέθους ωρών. Αυτό σαν αποτέλεσμα επιτρέπει την εύκολη εισαγωγή τους σε διαδικασίες συνεχούς ανάπτυξης και ενσωμάτωσης, χωρίς να δημιουργούν καθυστερήσεις στη συνολική απόδοση τέτοιων συστημάτων.

Ένα τελευταίο συμπέρασμα είναι ότι τα δύο εξετασθέντα εργαλεία ανάλυσης λογισμικού φαίνεται να λειτουργούν περισσότερο συμπληρωματικά, παρά ανταγωνιστικά, γεγονός το οποίο μεταφράζεται στο ότι παράλληλη χρήση τους επιτρέπει την αναγνώριση πολύ περισσότερων ευπαθειών, χωρίς να επιβαρύνεται η συνολική απόδοση.

5.1.2 Προτάσεις για μελλοντική εργασία

Από τη μελέτη της βιβλιογραφίας παρατηρήθηκε ότι το σύνολο σχεδόν των μελετών που υπάρχουν μέχρι στιγμής διαθέσιμες σχετικά με το συγκεκριμένο τομέα εστιάζουν στη σύγκριση εργαλείων ανάλυσης λογισμικού επάνω σε ειδικά διαμορφωμένες σουίτες ελέγχου απόδοσης. Παρότι η χρήση τέτοιων σουιτών λογισμικού είναι ιδιαίτερα χρήσιμη, δεν υπάρχουν αρκετές μελέτες οι οποίες έχουν εκτελεστεί σε πραγματικές συνθήκες, δηλαδή σε πραγματικά, λιγότερο ή περισσότερο ώριμα πακέτα λογισμικού. Όπως φάνηκε από την παρούσα μελέτη, κάτι τέτοιο μπορεί να είναι πιο χρονοβόρο και να μην παρουσιάζει τις ίδιες ευκολίες εξαγωγής συμπερασμάτων από τα αποτελέσματα, δεν παύει όμως να είναι εξαιρετικής σημασίας, αφού τελικά αυτό που ενδιαφέρει τον μελετητή/αναλυτή είναι η απόδοση του εργαλείου σε πραγματικά προβλήματα και όχι σε εργαστηριακές συνθήκες.

Εξαιτίας της σημασίας της ανάλυσης λογισμικού με στατικές μεθόδους, νέα εργαλεία αναπτύσσονται διαρκώς, τόσο ανοιχτού όσο και κλειστού κώδικα. Ως αποτέλεσμα, κρίνεται σημαντικό να πραγματοποιηθούν επιπλέον μελέτες (στα πλαίσια πραγματικών συνθηκών), με χρήση διαφορετικών εργαλείων ανοιχτού κώδικα, αλλά ακόμα και να συγκριθούν οι αποδόσεις που παρουσιάζουν τα εργαλεία αυτά, σε σχέση με εμπορικά εργαλεία για να αποτυπωθούν ομοιότητες και διαφορές μεταξύ των δύο κατηγοριών.

Σε αυτό το πλαίσιο, θα μπορούσαν επίσης να μελετηθούν κι άλλες κατηγορίες ευπαθειών, είτε κλασικές, είτε περισότερρο σπάνιες, ενώ το αντικείμενο μελέτης θα μπορούσε νε διευρυνθεί με διαφοροποίηση τόσο στα χαρακτηριστικά των πακέτων λογισμικού που εξετάζονται όσο και στη μελέτη λιγότερο ή περισσότερο ώριμων ή/και δημοφιλών πακέτων.

Επιπλέον των παραπάνω, στα ήδη αναλυμένα πακέτα λογισμικού θα μπορούσαν να εφαρμοστούν διαφορετικά εργαλεία και μέθοδοι ανάλυσης με την προτεραιότητα να δίνεται σε εργαλεία δυναμικής ανάλυσης, αφού αποτελεί την άλλη όψη του νομίσματος όσον αφορά στην ανάλυση πηγαίου κώδικα. Κάτι τέτοιο θα επέτρεπε να συγκρίνουμε

όχι μόνο διαφορετικά εργαλεία μεταξύ τους, αλλά και διαφορετικές μεθόδους ώστε να καταλήξουμε στις αποδοτικότερες.

Βιβλιογραφία

[1] Arvinder Kaur and Ruchika Nayyar. "A Comparative Study of Static Code Analysis tools for Vulnerability Detection in C/C++ and JAVA Source Code". In: *Procedia Computer Science* 171 (2020), pp. 2023–2029.

[2] Gabriel Díaz and Juan Ramón Bermejo. "Static analysis of source code security: Assessment of tools against SAMATE tests". In: *Inf. Softw. Technol.* 55 (2013), pp. 1462–1476.

[3] Hanmeet Kaur Brar and Puneet Jai Kaur. "Comparing Detection Ratio of Three Static Analysis Tools". In: *International Journal of Computer Applications* 124 (2015), pp. 35–40.

[4] Rahma Mahmood and Qusay H. Mahmoud. "Evaluation of Static Analysis Tools for Finding Vulnerabilities in Java and C/C++ Source Code". In: *ArXiv* abs/1805.09040 (2018).

[5] *Definition: source code analysis*. techtarget. Library Catalog: www.techtarget.com. URL: `https://www.techtarget.com/searchsoftwarequality/definition/source-code-analysis` (visited on 10/25/2022).

[6] Pär Emanuelsson and Ulf Nilsson. "A Comparative Study of Industrial Static Analysis Tools". In: *Electron. Notes Theor. Comput. Sci.* 217 (2008), pp. 5–21.

[7] *Flawfinder*. dwheeler. Library Catalog: https://dwheeler.com. URL: `https://dwheeler.com/flawfinder/` (visited on 10/25/2022).

[8] *flawfinder - Man Page*. mankier. Library Catalog: https://www.mankier.com. URL: `https://www.mankier.com/1/flawfinder` (visited on 10/25/2022).

[9] *Cppcheck: A tool for static C/C++ code analysis*. cppcheck.sourceforge. Library Catalog: https://cppcheck.sourceforge.io. URL: `https://cppcheck.sourceforge.io/` (visited on 10/25/2022).

[10] *Cppcheck manual Version 2.9*. cppcheck.sourceforge. Library Catalog: https://cppcheck.sourceforge.io. URL: `https://cppcheck.sourceforge.io/manual.pdf` (visited on 10/25/2022).

[11] *What's the Difference Between Sound and Unsound Static Analysis?* electronicdesign. Library Catalog: https://www.electronicdesign.com. URL: https://www.electronicdesign.com/technologies/embedded-revolution/article/21806987/adacore-whats-the-difference-between-sound-and-unsound-static-analysis.

[12] Alejandro Russo and Andrei Sabelfeld. "Dynamic vs. Static Flow-Sensitive Security Analysis". In: *2010 23rd IEEE Computer Security Foundations Symposium*. Sept. 2010, pp. 186–199. DOI: 10.1109/CSF.2010.20.

[13] Kirsten Winter et al. "Path-Sensitive Data Flow Analysis Simplified". In: *ICFEM*. 2013.

[14] *Cppcheck Design, Author: Daniel Marjamäki, Date: 2014-01-21*. sourceforge. Library Catalog: https://jztkft.dl.sourceforge.net. URL: https://jztkft.dl.sourceforge.net/project/cppcheck/Articles/cppcheck-design.pdf (visited on 10/25/2022).

[15] *Cppcheck Design, Daniel Marjamäki, Cppcheck 2010*. sourceforge. Library Catalog: https://jztkft.dl.sourceforge.net. URL: https://netix.dl.sourceforge.net/project/cppcheck/Articles/cppcheck-design-2010.pdf (visited on 10/25/2022).

[16] *Welcome to Expat!* libexpat. Library Catalog: https://libexpat.github.io. URL: https://libexpat.github.io (visited on 01/21/2023).

[17] *ListOfChecks*. sourceforge. Library Catalog: https://sourceforge.net. URL: https://sourceforge.net/p/cppcheck/wiki/ListOfChecks/ (visited on 10/25/2022).

[18] *CWE-120: Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')*. mitre. Library Catalog: https://cwe.mitre.org. URL: https://cwe.mitre.org/data/definitions/120.html (visited on 10/25/2022).

[19] *CWE - 120 : Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')*. cvedetails. Library Catalog: https://www.cvedetails.com. URL: https://www.cvedetails.com/cwe-details/120/ (visited on 10/25/2022).

[20] *CWE-190: Integer Overflow or Wraparound*. mitre. Library Catalog: https://cwe.mitre.org. URL: https://cwe.mitre.org/data/definitions/190.html (visited on 10/25/2022).

[21] *CWE - 190 : Integer Overflow or Wraparound*. cvedetails. Library Catalog: www.cvedetails.com. URL: https://www.cvedetails.com/cwe-details/190/Integer-Overflow-or-Wraparound.html (visited on 10/25/2022).

[22] Andrei Arusoaie et al. “A Comparison of Open-Source Static Analysis Tools for Vulnerability Detection in C/C++ Code”. In: *2017 19th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)* (2017), pp. 161–168.

[23] *Welcome to Kodi Home Theater Software!* github. Library Catalog: https://github.com. URL: `https://github.com/xbmc/xbmc` (visited on 10/25/2022).

[24] *10 C++ open source projects welcoming contributions*. codacy. Library Catalog: https://blog.codacy.com. URL: `https://blog.codacy.com/10-cpp-open-source-projects/` (visited on 10/25/2022).

[25] *Top 23 C++ Open-Source Projects*. libhunt. Library Catalog: https://www.libhunt.com. URL: `https://www.libhunt.com/topic/c-plus-plus` (visited on 10/25/2022).

[26] *Telegram - a new era of messaging*. telegram. Library Catalog: https://telegram.org/. URL: `https://telegram.org/` (visited on 12/12/2022).

[27] *Telegram Desktop – Official Messenger*. github. Library Catalog: https://github.com. URL: `https://github.com/telegramdesktop/tdesktop` (visited on 12/12/2022).

Παράρτημα

Παράρτημα Α

Έλεγχοι CppCheck

Εδώ παρουσιάζονται αναλυτικά οι έλεγχοι τους οποίους πραγματοποιεί το cppcheck [17]. Μπορεί κανείς να τους εποπτεύσει εκτελώντας το εργαλείο στη γραμμή εντολών με την επιλογή "–doc".

64-bit portability

Check if there is 64-bit portability issues:

- assign address to/from int/long
- casting address from/to integer when returning from function

Assert

Warn if there are side effects in assert statements (since this cause different behaviour in debug/release builds).

Auto Variables

A pointer to a variable is only valid as long as the variable is in scope. Check:

- returning a pointer to auto or temporary variable
- assigning address of an variable to an effective parameter of a function
- returning reference to local/temporary variable
- returning address of function parameter
- suspicious assignment of pointer argument
- useless assignment of function argument

Boolean

Boolean type checks

- using increment on boolean
- comparison of a boolean expression with an integer other than 0 or 1
- comparison of a function returning boolean value using relational operator
- comparison of a boolean value with boolean value using relational operator
- using bool in bitwise expression
- pointer addition in condition (either dereference is forgot or pointer overflow is required to make the condition false)

- Assigning bool value to pointer or float
- Returning an integer other than 0 or 1 from a function with boolean return value

Boost usage

Check for invalid usage of Boost:

- container modification during BOOST_FOREACH

Bounds checking

Out of bounds checking:

- Array index out of bounds
- Pointer arithmetic overflow
- Buffer overflow
- Dangerous usage of strncat()
- Using array index before checking it
- Partial string write that leads to buffer that is not zero terminated.
- Check for large enough arrays being passed to functions

Check function usage

Check function usage:

- missing 'return' in non-void function
- return value of certain functions not used
- invalid input values for functions
- Warn if a function is called whose usage is discouraged
- memset() third argument is zero
- memset() with a value out of range as the 2nd parameter
- memset() with a float as the 2nd parameter
- copy elision optimization for returning value affected by std::move

Class

Check the code for each class.

- Missing constructors and copy constructors
- Constructors which should be explicit
- Are all variables initialized by the constructors?
- Are all variables assigned by 'operator='?
- Warn if memset, memcpy etc are used on a class
- Warn if memory for classes is allocated with malloc()
- If it's a base class, check that the destructor is virtual
- Are there unused private functions?
- 'operator=' should check for assignment to self
- Constness for member functions
- Order of initializations
- Suggest usage of initialization list

- Initialization of a member with itself
- Suspicious subtraction from 'this'
- Call of pure virtual function in constructor/destructor
- Duplicated inherited data members
- Check that arbitrary usage of public interface does not result in division by zero
- Delete "self pointer" and then access 'this'
- Check that the 'override' keyword is used when overriding virtual functions
- Check that the 'one definition rule' is not violated

Condition

Match conditions with assignments and other conditions:

- Mismatching assignment and comparison =>comparison is always true/false
- Mismatching lhs and rhs in comparison =>comparison is always true/false
- Detect usage of — where & should be used
- Duplicate condition and assignment
- Detect matching 'if' and 'else if' conditions
- Mismatching bitand (a &= 0xf0; a &= 1; =>a = 0)
- Opposite inner condition is always false
- Identical condition after early exit is always false
- Condition that is always true/false
- Mutual exclusion over —— always evaluating to true
- Comparisons of modulo results that are always true/false.
- Known variable values =¿ condition is always true/false
- Invalid test for overflow. Some mainstream compilers remove such overflow tests when optimising code.
- Suspicious assignment of container/iterator in condition =¿ condition is always true.

Exception Safety

Checking exception safety - Throwing exceptions in destructors

- Throwing exception during invalid state
- Throwing a copy of a caught exception instead of rethrowing the original exception
- Exception caught by value instead of by reference
- Throwing exception in noexcept, nothrow(), attribute((nothrow)) or __declspec(nothrow) function
- Unhandled exception specification when calling function foo()
- Rethrow without currently handled exception

IO using format string

Check format string input/output operations.

- Bad usage of the function 'sprintf' (overlapping data)
- Missing or wrong width specifiers in 'scanf' format string
- Use a file that has been closed
- File input/output without positioning results in undefined behaviour
- Read to a file that has only been opened for writing (or vice versa)
- Repositioning operation on a file opened in append mode
- The same file can't be open for read and write at the same time on different streams
- Using fflush() on an input stream
- Invalid usage of output stream. For example: 'std::cout ¡¡ std::cout;'
- Wrong number of arguments given to 'printf' or 'scanf;'

Leaks (auto variables)

Detect when a auto variable is allocated but not deallocated or deallocated twice.

Memory leaks (address not taken)

Not taking the address to allocated memory

Memory leaks (class variables)

If the constructor allocate memory then the destructor must deallocate it.

Memory leaks (function variables)

Is there any allocated memory when a function goes out of scope

Memory leaks (struct members)

Don't forget to deallocate struct members

Null pointer

Null pointers

- null pointer dereferencing
- undefined null pointer arithmetic

Other

Other checks

- division with zero
- scoped object destroyed immediately after construction
- assignment in an assert statement
- free() or delete of an invalid memory location
- bitwise operation with negative right operand
- provide wrong dimensioned array to pipe() system command (–std=posix)
- cast the return values of getc(),fgetc() and getchar() to character and compare it to EOF
- race condition with non-interlocked access after InterlockedDecrement() call
- expression 'x = x++;' depends on order of evaluation of side effects
- overlapping write of union
- either division by zero or useless condition

- access of moved or forwarded variable.
- redundant data copying for const variable
- subsequent assignment or copying to a variable or buffer
- passing parameter by value
- Passing NULL pointer to function with variable number of arguments leads to UB.
- C-style pointer cast in C++ code
- casting between incompatible pointer types
- Incomplete statement
- check how signed char variables are used
- variable scope can be limited
- unusual pointer arithmetic. For example: "abc" + 'd'
- redundant assignment, increment, or bitwise operation in a switch statement
- redundant strcpy in a switch statement
- Suspicious case labels in switch()
- assignment of a variable to itself
- Comparison of values leading always to true or false
- Clarify calculation with parentheses
- suspicious comparison of '\0' with a char* variable
- duplicate break statement
- unreachable code
- testing if unsigned variable is negative/positive
- Suspicious use of ; at the end of 'if/for/while' statement.
- Array filled incompletely using memset/memcpy/memmove.
- NaN (not a number) value used in arithmetic expression.
- comma in return statement (the comma can easily be misread as a semicolon).
- prefer erfc, expm1 or log1p to avoid loss of precision.
- identical code in both branches of if/else or ternary operator.
- redundant pointer operation on pointer like &*some_ptr.
- find unused 'goto' labels.
- function declaration and definition argument names different.
- function declaration and definition argument order different.
- shadow variable.
- variable can be declared const.
- calculating modulo of one.
- known function argument, suspicious calculation.

STL usage

Check for invalid usage of STL:

- out of bounds errors

- misuse of iterators when iterating through a container
- mismatching containers in calls
- same iterators in calls
- dereferencing an erased iterator
- for vectors: using iterator/pointer after push_back has been used
- optimisation: use empty() instead of size() to guarantee fast code
- suspicious condition when using find
- unnecessary searching in associative containers
- redundant condition
- common mistakes when using string::c_str()
- useless calls of string and STL functions
- dereferencing an invalid iterator
- reading from empty STL container
- iterating over an empty STL container
- consider using an STL algorithm instead of raw loop
- incorrect locking with mutex

Sizeof

sizeof() usage checks

- sizeof for array given as function argument
- sizeof for numeric given as function argument
- using sizeof(pointer) instead of the size of pointed data
- look for 'sizeof sizeof ..'
- look for calculations inside sizeof()
- look for function calls inside sizeof()
- look for suspicious calculations with sizeof()
- using 'sizeof(void)' which is undefined

String

Detect misusage of C-style strings:

- overlapping buffers passed to sprintf as source and destination
- incorrect length arguments for 'substr' and 'strncmp'
- suspicious condition (runtime comparison of string literals)
- suspicious condition (string/char literals as boolean)
- suspicious comparison of a string literal with a char* variable
- suspicious comparison of '\0' with a char* variable
- overlapping strcmp() expression

Type

Type checks

- bitwise shift by too many bits (only enabled when –platform is used)
- signed integer overflow (only enabled when –platform is used)

- dangerous sign conversion, when signed value can be negative
- possible loss of information when assigning int result to long variable
- possible loss of information when returning int result as long return value
- float conversion overflow

Uninitialized variables

Uninitialized variables

- using uninitialized local variables
- using allocated data before it has been initialized

Unused functions

Check for functions that are never called

UnusedVar

UnusedVar checks

- unused variable
- allocated but unused variable
- unread variable
- unassigned variable
- unused struct member

Using postfix operators

Warn if using postfix operators ++ or – rather than prefix operator

Vaarg

Check for misusage of variable argument lists:

- Wrong parameter passed to va_start()
- Reference passed to va_start()
- Missing va_end()
- Using va_list before it is opened
- Subsequent calls to va_start/va_copy()

Παράρτημα Β

Πίνακες Ευρημάτων

Παρακάτω παρουσιάζονται πίνακες με λεπτομέρειες των ευρημάτων που ανιχνεύθηκαν στα Kodi, Kodi-build και Telegram από τα εργαλεία Flawfinder, CppCheck και CppCheck-GUI σε σχέση με τα αρχεία και τις γραμμές όπου εμφανίζονται.

B.1 Flawfinder

Kodi

gets()		
Αρχείο	**Γραμμή**	**Κώδικας**
./lib/libUPnP/Platinum/Source/Apps/FrameStreamer/main.cpp	238	while (gets(buf))
./lib/libUPnP/Platinum/Source/Apps/MediaConnect/main.cpp	121	while (gets(buf))
./lib/libUPnP/Platinum/Source/Apps/MediaCrawler/main.cpp	74	while (gets(buf)) {
./lib/libUPnP/Platinum/Source/Tests/FileMediaServer/FileMediaServerTest.cpp	151	while (gets(buf)) {
./lib/libUPnP/Platinum/Source/Tests/LightSample/LightSampleTest.cpp	68	while (gets(buf)) {
./lib/libUPnP/Platinum/Source/Tests/MediaRenderer/MediaRendererTest.cpp	105	while (gets(buf)) {
./lib/libUPnP/Platinum/Source/Tests/Simple/SimpleTest.cpp	55	while (gets(buf)) {
./lib/libUPnP/Platinum/Source/Tests/Ssdp/SsdpTest1.cpp	58	while (gets(buf)) {
./lib/libUPnP/Platinum/Source/Tools/SsdpProxy/SsdpProxy.cpp	399	while (gets(buf)) {

sprintf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./tools/EventClients/Clients/ WiiRemote/ CWIID_WiiRemote.cpp	193	sprintf(m_JoyMap, "JS0:%s", JoyMap);
./tools/depends/target/fmt/ native/test/module-test.cc	479	TEST(module_test, sprintf) {
./tools/depends/target/fmt/ native/test/printf-test.cc	57	EXPECT_EQ(L"test", fmt::sprintf(L"test"));
./tools/depends/target/fmt/ native/test/printf-test.cc	66	EXPECT_EQ(L"%", fmt::sprintf(L"%%"));
./tools/depends/target/fmt/ native/test/printf-test.cc	67	EXPECT_EQ(L"before %", fmt::sprintf(L"before %%"));
./tools/depends/target/fmt/ native/test/printf-test.cc	68	EXPECT_EQ(L"% after", fmt::sprintf(L"%% after"));
./tools/depends/target/fmt/ native/test/printf-test.cc	69	EXPECT_EQ(L"before % after", fmt::sprintf(L"before %% after"));
./tools/depends/target/fmt/ native/test/printf-test.cc	70	EXPECT_EQ(L"%s", fmt::sprintf(L"%%s"));
./xbmc/cores/DllLoader/coff.cpp	383	sprintf(szBuf, "Load code Sections %s Memory %p,Length %x\n", namebuf,
./xbmc/dbwrappers/ sqlitedataset.cpp	458	sprintf(sqlcmd, "DROP INDEX '%s'", res.records[i]->at(0).get_asString().c_str());
./xbmc/dbwrappers/ sqlitedataset.cpp	471	sprintf(sqlcmd, "DROP VIEW '%s'", res.records[i]->at(0).get_asString().c_str());
./xbmc/dbwrappers/ sqlitedataset.cpp	484	sprintf(sqlcmd, "DROP TRIGGER '%s'", res.records[i]->at(0).get_asString().c_str());
./xbmc/network/cddb.cpp	714	sprintf(cddb_command, "cddb read %s %s", genre, discid);
./xbmc/platform/android/ network/NetworkAndroid.cpp	297	sprintf(cmd_line, "ping -c 1 -w %d %s", timeout_ms / 1000 + (timeout_ms % 1000) != 0, inet_ntoa(host_ip));

./xbmc/platform/darwin/osx/network/NetworkOsx.cpp	214	sprintf(cmd_line, "ping -c 1 -t %d %s", timeout_ms / 1000 + (timeout_ms % 1000) != 0,
./xbmc/platform/freebsd/network/NetworkFreebsd.cpp	231	sprintf(cmd_line, "ping -c 1 -t %d %s", timeout_ms / 1000 + (timeout_ms % 1000) != 0,
./xbmc/platform/linux/network/NetworkLinux.cpp	207	sprintf(cmd_line, "ping -c 1 -w %d %s", timeout_ms / 1000 + (timeout_ms % 1000) != 0,

vsprintf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./tools/depends/target/fmt/native/test/module-test.cc	507	TEST(module_test, vsprintf) {

strcat()		
Αρχείο	**Γραμμή**	**Κώδικας**
./tools/depends/native/TexturePacker/src/Win32/dirent.c	55	strcat(strcpy(dir->name, name), all);
./xbmc/Util.cpp	223	strcat(given_path, CCompileInfo::GetAppName());
./xbmc/cores/VideoPlayer/DVDSubtitles/DVDSubtitlesLibass.cpp	676	strcat(appendedText, text);
./xbmc/network/cddb.cpp	928	strcat(query_buffer, tmp_buffer);
./xbmc/network/cddb.cpp	933	strcat(query_buffer, tmp_buffer);
./xbmc/network/cddb.cpp	939	strcat(query_buffer, tmp_buffer);
./xbmc/network/cddb.cpp	944	strcat(query_buffer, tmp_buffer);
./xbmc/pictures/ExifParse.cpp	861	strcat(latLongString, latLong);
./xbmc/pictures/ExifParse.cpp	942	strcat(m_ExifInfo->GpsAlt, temp);

strcpy()		
Αρχείο	**Γραμμή**	**Κώδικας**
./tools/depends/native/TexturePacker/src/Win32/dirent.c	55	strcat(strcpy(dir->name, name), all);
./tools/depends/native/TexturePacker/src/cmdlineargs.h	43	strcpy (m_cmdline, cmdline);
./tools/depends/native/TexturePacker/src/cmdlineargs.h	76	strcpy(m_cmdline, cmdline.c_str());
./xbmc/addons/kodi-dev-kit/include/kodi/addon-instance/peripheral/PeripheralUtils.h	337	std::strcpy(info.name, m_strName.c_str());
./xbmc/addons/kodi-dev-kit/include/kodi/addon-instance/peripheral/PeripheralUtils.h	736	std::strcpy(info.provider, m_provider.c_str());
./xbmc/addons/kodi-dev-kit/include/kodi/addon-instance/peripheral/PeripheralUtils.h	1259	std::strcpy(feature.name, m_name.c_str());
./xbmc/cores/DllLoader/DllLoader.cpp	556	strcpy(const_cast<char*>(entry->exp.name), sFunctionName);
./xbmc/cores/DllLoader/DllLoader.cpp	573	strcpy(const_cast<char*>(entry->exp.name), sFunctionName);
./xbmc/cores/DllLoader/dll.cpp	78	strcpy(libpath, DEFAULT_DLLPATH);
./xbmc/cores/DllLoader/dll.cpp	234	strcpy(strModuleName, lpModuleName);
./xbmc/cores/VideoPlayer/DVDDemuxers/DVDDemuxVobsub.cpp	117	strcpy((char*)m_Streams[i]->ExtraData, state.extra.c_str());
./xbmc/cores/VideoPlayer/DVDSubtitles/DVDSubtitlesLibass.cpp	675	strcpy(appendedText, assEvent->Text);
./xbmc/games/addons/GameClientProperties.cpp	124	std::strcpy(resourceDir, resourcePath.c_str());
./xbmc/games/addons/GameClientProperties.cpp	149	std::strcpy(addonProfileDir, addonProfile.c_str());
./xbmc/games/addons/GameClientProperties.cpp	155	std::strcpy(addonPathDir, addonPath.c_str());

./xbmc/games/addons/ GameClientProperties.cpp	186	std::strcpy(ext, extension.c_str());
./xbmc/games/addons/ GameClientProperties.cpp	276	std::strcpy(libPath, strLibPath.c_str());
./xbmc/guilib/ GUISpinControl.cpp	99	std::strcpy(m_szTyped, control.m_szTyped);
./xbmc/network/cddb.cpp	128	strcpy(tmp_buffer.get(), (const char*)buffer);
./xbmc/platform/linux/input/ LIRC.cpp	190	strcpy(addr_un.sun_path, socket_path);
./xbmc/platform/linux/network/ NetworkLinux.cpp	134	strcpy(ifr.ifr_name, interfaceName.c_str());
./xbmc/platform/posix/network/ NetworkPosix.cpp	41	strcpy(ifr.ifr_name, m_interfaceName.c_str());
./xbmc/platform/posix/network/ NetworkPosix.cpp	53	strcpy(ifr.ifr_name, m_interfaceName.c_str());
./xbmc/platform/posix/network/ NetworkPosix.cpp	72	strcpy(ifr.ifr_name, m_interfaceName.c_str());
./xbmc/platform/posix/network/ NetworkPosix.cpp	87	strcpy(ifr.ifr_name, m_interfaceName.c_str());
./xbmc/platform/win10/ Win10App.cpp	114	std::strcpy(val, str.c_str());
./xbmc/windowing/X11/ WinEventsX11.cpp	181	strcpy(old_locale, p);
./xbmc/windowing/X11/ WinEventsX11.cpp	187	strcpy(old_modifiers, p);
lstrcpyW()		
Αρχείο	**Γραμμή**	**Κώδικας**
./xbmc/windowing/windows/ WinSystemWin32.cpp	818	lstrcpyW(ddMon.DeviceString, L"Dummy Monitor"); // safe: large static array

sscanf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./xbmc/cores/VideoPlayer/ DVDDemuxers/ DVDDemuxVobsub.cpp	249	if (sscanf(line.c_str(), "%d:%d:%d:%d, filepos:%" PRIx64, &h, &m, &s, &ms, &timestamp.pos) != 5)
./xbmc/platform/linux/input/ LIRC.cpp	128	sscanf(buf, "%s %s %s %s", &scanCode[0], &repeatStr[0], &buttonName[0], &deviceName[0]);

realpath()		
Αρχείο	**Γραμμή**	**Κώδικας**
./tools/depends/target/ flatbuffers/native/src/util.cpp	203	char *abs_path_temp = realpath(filepath.c_str(), nullptr);
./xbmc/Util.cpp	249	if (realpath(given_path, real_path) != NULL)
./xbmc/utils/FileUtils.cpp	302	char *fullpath = realpath(decodePath.c_str(), nullptr);
./xbmc/utils/FileUtils.cpp	313	char *realtemp = realpath(w.c_str(), nullptr);

Kodi-build

strncat()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/libdvdread/src/ libdvdread/src/dvd_udf.c	837	strncat(tokenline, filename, MAX_UDF_FILE_NAME_LEN - 1);

sprintf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/libdvdcss/src/libdvdcss/src/libdvdcss.c	458	i += sprintf(dvdcss->psz_cachefile + i, "/%s-%s-%s",
./build/libdvdread/src/libdvdread/src/dvd_reader.c	737	sprintf(filename, "%s%s%s", path, ((path[strlen(path) - 1] == '/') ? "" : "/"), file);
./build/libdvdread/src/libdvdread/src/dvd_reader.c	750	sprintf(filename, "%s%s%s", path,
./build/libdvdread/src/libdvdread/src/dvd_reader.c	779	sprintf(video_path, "%s/VIDEO_TS/", dvd->rd->path_root);
./build/libdvdread/src/libdvdread/src/dvd_reader.c	783	sprintf(video_path, "%s/video_ts/", dvd->rd->path_root);
./build/rapidjson/src/rapidjson/example/tutorial/tutorial.cpp	125	int len = sprintf(buffer2, "%s %s", "Milo", "Yip"); // synthetic example of dynamically created string.
./build/rapidjson/src/rapidjson/test/perftest/perftest.h	137	sprintf(filename, "%s/%s", typespaths[i], typesfilenames[j]);
./build/rapidjson/src/rapidjson/test/perftest/schematest.cpp	26	sprintf(buffer, "%s%s", paths[i], filename);
./build/rapidjson/src/rapidjson/test/perftest/schematest.cpp	88	sprintf(filename, "jsonschema/tests/draft4/%s", filenames[i]);
./build/rapidjson/src/rapidjson/test/unittest/documenttest.cpp	157	sprintf(buffer, "%s/%s", paths[i], filename);
./build/rapidjson/src/rapidjson/test/unittest/encodedstreamtest.cpp	54	sprintf(buffer, "%s/%s", paths[i], filename);
./build/rapidjson/src/rapidjson/test/unittest/istreamwrappertest.cpp	116	sprintf(buffer, "%s/%s", paths[i], filename);
./build/rapidjson/src/rapidjson/test/unittest/jsoncheckertest.cpp	32	sprintf(buffer, "%s/%s", paths[i], filename);
./build/rapidjson/src/rapidjson/test/unittest/readertest.cpp	428	sprintf(buffer, "%s", str);
./build/rapidjson/src/rapidjson/test/unittest/schematest.cpp	988	sprintf(buffer, "%s%s", paths[i], filename);
./build/rapidjson/src/rapidjson/test/unittest/schematest.cpp	1177	sprintf(filename, "jsonschema/tests/draft4/%s", filenames[i]);

wcscpy()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/compat/w32dlfcn.h	52	wcscpy(path + pathlen + 1, name_w);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/compat/w32dlfcn.h	60	wcscpy(path + pathlen + 1, name_w);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavdevice/dshow_filter.c	138	wcscpy(this->info.achName, name);
./build/ffmpeg/src/ffmpeg/compat/w32dlfcn.h	52	wcscpy(path + pathlen + 1, name_w);
./build/ffmpeg/src/ffmpeg/compat/w32dlfcn.h	60	wcscpy(path + pathlen + 1, name_w);
./build/ffmpeg/src/ffmpeg/libavdevice/dshow_filter.c	138	wcscpy(this->info.achName, name);
strcat()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/pktdumper.c	89	strcat(fntemplate, PKTFILE-SUFF);
./build/ffmpeg/src/ffmpeg/tools/pktdumper.c	89	strcat(fntemplate, PKTFILE-SUFF);
./build/libdvdread/src/libdvdread/msvc/contrib/dirent/dirent.c	52	strcat(strcpy(dir->name, name), all);
./build/libdvdread/src/libdvdread/src/dvd_reader.c	314	strcat(new_path, strstr(path, ”/dsk/”) + strlen(”/dsk/”));
./build/libdvdread/src/libdvdread/src/dvd_reader.c	344	strcat(new_path, path + strlen(”/dev/”)

strcpy()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/avio.c	105	strcpy(uc->filename, filename);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/hls.c	346	strcpy(var->audio_group, info-¿audio);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/hls.c	347	strcpy(var->video_group, info->video);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/hls.c	348	strcpy(var->subtitles_group, info->subtitles);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/hls.c	513	strcpy(rend->group_id, info->group_id);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/hls.c	514	strcpy(rend->language, info->language);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/hls.c	515	strcpy(rend->name, info->name);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/rtsp.c	1940	strcpy(real_challenge, reply->real_challenge);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/srtdec.c	179	strcpy(line_cache, line);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/wtvdec.c	491	strcpy(buf, avio_rl32(pb) ? "true" : "false");
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavutil/log.c	385	strcpy(prev, line);
./build/ffmpeg/src/ffmpeg/ libavformat/avio.c	105	strcpy(uc->filename, filename);

./build/ffmpeg/src/ffmpeg/libavformat/hls.c	346	strcpy(var->audio_group, info->audio);
./build/ffmpeg/src/ffmpeg/libavformat/hls.c	347	strcpy(var->video_group, info->video);
./build/ffmpeg/src/ffmpeg/libavformat/hls.c	348	strcpy(var->subtitles_group, info->subtitles);
./build/ffmpeg/src/ffmpeg/libavformat/hls.c	513	strcpy(rend->group_id, info->group_id);
./build/ffmpeg/src/ffmpeg/libavformat/hls.c	514	strcpy(rend->language, info->language);
./build/ffmpeg/src/ffmpeg/libavformat/hls.c	515	strcpy(rend->name, info->name);
./build/ffmpeg/src/ffmpeg/libavformat/rtsp.c	1940	strcpy(real_challenge, reply->real_challenge);
./build/ffmpeg/src/ffmpeg/libavformat/srtdec.c	179	strcpy(line_cache, line);
./build/ffmpeg/src/ffmpeg/libavformat/wtvdec.c	491	strcpy(buf, avio_rl32(pb) ? "true" : "false");
./build/ffmpeg/src/ffmpeg/libavutil/log.c	385	strcpy(prev, line);
./build/include/kodi/addon-instance/peripheral/PeripheralUtils.h	337	std::strcpy(info.name, m_strName.c_str());
./build/include/kodi/addon-instance/peripheral/PeripheralUtils.h	736	std::strcpy(info.provider, m_provider.c_str());
./build/include/kodi/addon-instance/peripheral/PeripheralUtils.h	1259	std::strcpy(feature.name, m_name.c_str());
./build/libdvdcss/src/libdvdcss/src/libdvdcss.c	284	strcpy(dvdcss->psz_cachefile, psz_unixroot);
./build/libdvdcss/src/libdvdcss/test/dvd_region.c	243	strcpy(device_name, DEFAULT_DEVICE);

./build/libdvdread/src/ libdvdread/msvc/ contrib/dirent/dirent.c	52	strcat(strcpy(dir->name, name), all);
./build/libdvdread/src/ libdvdread/src/dvd_reader.c	311	strcpy(new_path, path);
./build/rapidjson/src/rapidjson/ test/unittest/simdtest.cpp	123	strcpy(backup, json); // insitu parsing will overwrite buffer, so need to backup first
./build/rapidjson/src/rapidjson/ test/unittest/simdtest.cpp	147	strcpy(backup, json); // insitu parsing will overwrite buffer, so need to backup first

sscanf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavdevice/iec61883.c	283	if (sscanf(dv->device_guid, "%"SCNu64, &guid) != 1) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/aadec.c	124	ret = sscanf(val, "%"SCNu32"%"SCNu32" %"SCNu32"%"SCNu32,
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/aqtitledec.c	73	if (sscanf(line, "–>>%"SCNd64, &frame) == 1) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/ffmetadec.c	110	ret = sscanf(line, "START=%"SCNd64, &start);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/ffmetadec.c	118	ret = sscanf(line, "END=%"SCNd64, &end);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/lrcdec.c	185	sscanf(comma_offset + 1, "%"SCNd64, &lrc->ts_offset) != 1) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/mpeg.c	839	if (sscanf(p, "%02d:%02d:%02d:%03d, filepos: %"SCNx64,
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/nistspheredec.c	50	sscanf(buffer, "%"SCNd32, &header_size);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/nistspheredec.c	111	sscanf(buffer, "%*s %*s %"SCNd64, &st->duration);
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/pjsdec.c	55	if (sscanf(*line, "%"SCNd64",%"SCNd64, &start, &end) == 2) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/rdt.c	451	if (sscanf(p, "%*1[Aa]verage%*1[Bb]andwidth=%"SCNd64, &st->codecpar->bit_rate) == 1)

./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/rpl.c	288	if (3 != sscanf(line, "%"SCNd64" , %"SCNd64" ; %"SCNd64,
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/srtdec.c	80	if (sscanf(line, "%d:%d:%d%*1[,.]%d –>%d:%d:%d%*1[,.]%d"
./build/ffmpeg/src/ffmpeg/libavdevice/iec61883.c	283	if (sscanf(dv->device_guid, "%"SCNu64, &guid) != 1) {
./build/ffmpeg/src/ffmpeg/libavformat/aadec.c	124	ret = sscanf(val, "%"SCNu32"%"SCNu32" %"SCNu32"%"SCNu32,
./build/ffmpeg/src/ffmpeg/libavformat/aqtitledec.c	73	if (sscanf(line, "–>>%"SCNd64, &frame) == 1) {
./build/ffmpeg/src/ffmpeg/libavformat/ffmetadec.c	110	ret = sscanf(line, "START=%"SCNd64, &start);
./build/ffmpeg/src/ffmpeg/libavformat/ffmetadec.c	118	ret = sscanf(line, "END=%"SCNd64, &end);
./build/ffmpeg/src/ffmpeg/libavformat/lrcdec.c	185	sscanf(comma_offset + 1, "%"SCNd64, &lrc->ts_offset) != 1) {
./build/ffmpeg/src/ffmpeg/libavformat/mpeg.c	839	if (sscanf(p, "%02d:%02d:%02d:%03d, filepos: %"SCNx64,
./build/ffmpeg/src/ffmpeg/libavformat/nistspheredec.c	50	sscanf(buffer, "%"SCNd32, &header_size);
./build/ffmpeg/src/ffmpeg/libavformat/nistspheredec.c	111	sscanf(buffer, "%*s %*s %"SCNd64, &st->duration);
./build/ffmpeg/src/ffmpeg/libavformat/pjsdec.c	55	if (sscanf(*line, "%"SCNd64",%"SCNd64, &start, &end) == 2) {
./build/ffmpeg/src/ffmpeg/libavformat/rdt.c	451	if (sscanf(p, "%*1[Aa]verage%*1[Bb]andwidth= %"SCNd64, &st->codecpar->bit_rate) == 1)
./build/ffmpeg/src/ffmpeg/libavformat/rpl.c	288	if (3 != sscanf(line, "%"SCNd64" , %"SCNd64" ; %"SCNd64,
./build/ffmpeg/src/ffmpeg/libavformat/srtdec.c	80	if (3 != sscanf(line, "%"SCNd64" , %"SCNd64" ; %"SCNd64,

getopt()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/compat/getopt.c	41	static int getopt(int argc, char *argv[], char *opts)
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavcodec/tests/dct.c	479	c = getopt(argc, argv, "ih4t");
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavcodec/tests/fft.c	405	int c = getopt(argc, argv, "hsimrdn:f:c:");
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavformat/tests/movenc.c	365	c = getopt(argc, argv, "wh");
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/libavutil/tests/cpu.c	117	int c = getopt(argc, argv, "c:t:");
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/crypto_bench.c	673	while ((opt = getopt(argc, argv, "hl:a:r:")) != -1) {
/build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/ffescape.c	69	while ((c = getopt(argc, argv, "ef:hi:l:o:m:p:s:")) != -1) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/ffeval.c	71	while ((c = getopt(argc, argv, "ehi:o:p:")) != -1) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/fourcc2pixfmt.c	71	while ((c = getopt(argc, argv, "hp:lL")) != -1) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/graph2dot.c	121	while ((c = getopt(argc, argv, "hi:o:")) != -1) {
./build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/seek_print.c	52	while ((opt = getopt(argc, argv, "h")) != -1) {

/build/ffmpeg/src/ffmpeg-build/ffmpeg-prefix/src/ffmpeg-build/src/tools/zmqsend.c	64	while ((c = getopt(argc, argv, "b:hi:")) != -1) {
./build/ffmpeg/src/ffmpeg/compat/getopt.c	41	static int getopt(int argc, char *argv[], char *opts)
./build/ffmpeg/src/ffmpeg/libavcodec/tests/dct.c	479	c = getopt(argc, argv, "ih4t");
./build/ffmpeg/src/ffmpeg/libavcodec/tests/fft.c	405	int c = getopt(argc, argv, "hsimrdn:f:c:");
./build/ffmpeg/src/ffmpeg/libavformat/tests/movenc.c	365	c = getopt(argc, argv, "wh");
./build/ffmpeg/src/ffmpeg/libavutil/tests/cpu.c	117	int c = getopt(argc, argv, "c:t:");
/build/ffmpeg/src/ffmpeg/tools/crypto_bench.c	673	while ((opt = getopt(argc, argv, "hl:a:r:")) != -1) {
./build/ffmpeg/src/ffmpeg/tools/ffescape.c	69	while ((c = getopt(argc, argv, "ef:hi:l:o:m:p:s:")) != -1) {
./build/ffmpeg/src/ffmpeg/tools/ffeval.c	71	while ((c = getopt(argc, argv, "ehi:o:p:")) != -1) {
./build/ffmpeg/src/ffmpeg/tools/fourcc2pixfmt.c	71	while ((c = getopt(argc, argv, "hp:lL")) != -1) {
./build/ffmpeg/src/ffmpeg/tools/graph2dot.c	121	while ((c = getopt(argc, argv, "hi:o:")) != -1) {
./build/ffmpeg/src/ffmpeg/tools/seek_print.c	52	while ((opt = getopt(argc, argv, "h")) != -1) {
./build/ffmpeg/src/ffmpeg/tools/zmqsend.c	64	while ((c = getopt(argc, argv, "b:hi:")) != -1) {
./build/libdvdcss/src/libdvdcss/test/dvd_region.c	245	while((c = getopt(argc, argv, "d:sr:h?")) != EOF) {
./build/libdvdnav/src/libdvdnav/msvc/contrib/getopt.c	926	getopt(argc, argv, optstring)
./build/libdvdnav/src/libdvdnav/msvc/contrib/getopt.c	956	c = getopt(argc, argv, "abc:d:0123456789");

realpath()		
Αρχείο	**Γραμμή**	**Κώδικας**
./build/libdvdread/src/libdvdread/src/dvd_reader.c	471	new_path = realpath(path_copy, NULL);

Telegram

MultiByteToWideChar()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/ThirdParty/libtgvoip/os/windows/CXWrapper.cpp	157	MultiByteToWideChar(CP_UTF8, 0, v, -1, buf, sizeof(buf));
sprintf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/SourceFiles/_other/updater_linux.cpp	65	sprintf(logName, "%sDebugLogs/%04d%02d%02d_%02d%02d%02d_upd.txt", workDir.c_str(),
./Telegram/ThirdParty/hime/src/hime-conf.c	192	sprintf (fname, "%s/%s", TableDir, name);
./Telegram/ThirdParty/hime/src/modules/chewing-conf.c	50	sprintf (pszChewingConfig, "%s%s", pszHome, HIME_CHEWING_CONFIG);
./Telegram/ThirdParty/hime/src/modules/chewing-conf.c	156	sprintf (pszHimeKBConfig, "%s%s", pszHome, HIME_KB_CONFIG);
./Telegram/ThirdParty/hime/src/modules/chewing.c	169	sprintf (pszChewingHashDir, "%s/.chewing", pszHome);
./Telegram/ThirdParty/jemalloc/test/unit/stats.c	229	sprintf(cmd, "stats.arenas.%u.bins.0.%s", arena_ind, name);
./out/cmake/external/jemalloc/jemalloc-prefix/src/jemalloc/test/unit/stats.c	229	sprintf(cmd, "stats.arenas.%u.bins.0.%s", arena_ind, name);

strcat()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/ThirdParty/QR/c/ qrcodegen-demo.c	131	strcat(concat, silver0);
./Telegram/ThirdParty/QR/c/ qrcodegen-demo.c	132	strcat(concat, silver1);
./Telegram/ThirdParty/QR/c/ qrcodegen-demo.c	163	strcat(concat, golden0);
./Telegram/ThirdParty/QR/c/ qrcodegen-demo.c	164	strcat(concat, golden1);
./Telegram/ThirdParty/QR/c/ qrcodegen-demo.c	165	strcat(concat, golden2);
./Telegram/ThirdParty/hime/ src/eve.c	185	strcat (*buf, text);
./Telegram/ThirdParty/hime/ src/gtab-buf.c	74	strcat (out, gbuf[i + start].ch);
./Telegram/ThirdParty/hime/ src/gtab-buf.c	179	strcat (addspc, gbuf[i].ch);
./Telegram/ThirdParty/hime/ src/gtab-buf.c	331	strcat (t, gbuf[j].ch);
./Telegram/ThirdParty/hime/ src/gtab-buf.c	792	strcat (tt, s);
./Telegram/ThirdParty/hime/ src/gtab-buf.c	994	strcat (tt, ts);
./Telegram/ThirdParty/hime/ src/gtab-init.c	125	strcat (out_file, fname);
./Telegram/ThirdParty/hime/ src/gtab-init.c	305	strcat (tt, th.selkey2);
./Telegram/ThirdParty/hime/ src/gtab.c	204	strcat (out, tbuf[i]);
./Telegram/ThirdParty/hime/ src/gtab.c	257	return strcat (s, t);
./Telegram/ThirdParty/hime/ src/gtab.c	265	return strcat (s, t);
./Telegram/ThirdParty/hime/ src/gtab.c	643	strcat (tt, htmlspecialchars (seltab[0], uu));
./Telegram/ThirdParty/hime/ src/gtab.c	670	strcat (tt, www);

./Telegram/ThirdParty/hime/src/gtab.c	677	strcat (strcat (tt, selback), " ");
./Telegram/ThirdParty/hime/src/gtab.c	684	strcat (tt, uu);
./Telegram/ThirdParty/hime/src/gtab.c	698	strcat (strcat (tt, uu), " ");
./Telegram/ThirdParty/hime/src/gtab.c	720	strcat (tt, tstr2);
./Telegram/ThirdParty/hime/src/hime-cin2gtab.c	632	strcat(strcpy(bzip2, "bzip2 -f -k "), fname_tab);
./Telegram/ThirdParty/hime/src/hime-conf.c	100	strcat (strcat (fname, "/"), name);
./Telegram/ThirdParty/hime/src/hime-conf.c	106	strcat (strcat (fname, "/config/"), name);
./Telegram/ThirdParty/hime/src/hime-juyin-learn.c	75	strcat (tt, phostr);
./Telegram/ThirdParty/hime/src/hime-ts-edit.c	241	strcat (line, t = g_markup_escape_text (tt, -1));
./Telegram/ThirdParty/hime/src/hime-ts-edit.c	247	strcat (line, tt);
./Telegram/ThirdParty/hime/src/hime-tsin2gtab-phrase.c	313	strcat (kstr, tkey);
./Telegram/ThirdParty/hime/src/modules/anthy.c	712	strcat (out, gtk_label_get_text (GTK_LABEL (seg[i].label)));
./Telegram/ThirdParty/hime/src/modules/anthy.c	726	strcat (out, idx_hira_kata (jp[i], always_hira));
./Telegram/ThirdParty/hime/src/modules/anthy.c	1382	strcat (str, s);
./Telegram/ThirdParty/hime/src/modules/anthy.c	1398	strcat (str, keys);
./Telegram/ThirdParty/hime/src/modules/anthy.c	1406	strcat (str, s);
./Telegram/ThirdParty/hime/src/modules/anthy.c	1412	strcat (str, keys);
./Telegram/ThirdParty/hime/src/modules/chewing.c	563	strcat (pszStr, pszTmpStr);
./Telegram/ThirdParty/hime/src/modules/chewing.c	568	strcat (pszStr, pszZuinStr);

./Telegram/ThirdParty/hime/ src/pho-util.c	270	strcat (out, pstr);
./Telegram/ThirdParty/hime/ src/pho2pinyin.c	59	strcat (tt, tone);
./Telegram/ThirdParty/hime/ src/tray-appindicator.c	115	strcat (s, iconfile);
./Telegram/ThirdParty/hime/ src/tray-appindicator.c	133	strcat (st_str, st_full);
./Telegram/ThirdParty/hime/ src/tray-appindicator.c	135	strcat (st_str, st_half);
./Telegram/ThirdParty/hime/ src/tray-appindicator.c	137	strcat (st_str, st_gb);
./Telegram/ThirdParty/hime/ src/tray-double.c	333	strcat (tt, iconame);
./Telegram/ThirdParty/hime/ src/tsin.c	149	strcat (ch + ofs, str);
./Telegram/ThirdParty/hime/ src/win-kbm.c	450	strcat (strcpy (tt, t), str);
./Telegram/ThirdParty/hime/ src/win-save-phrase.c	164	strcat (tt, phokey_to_str (wsp[i].key));
./Telegram/ThirdParty/hime/ src/win-sym.c:	57	strcat (strcat (strcpy (fname, TableDir), "/"), filename);
./Telegram/ThirdParty/hime/ src/win-sym.c:	318	strcat (outstr, tt);
./Telegram/ThirdParty/lz4/ programs/lz4cli.c	693	strcat(dynNameSpace, LZ4_EXTENSION);
./Telegram/ThirdParty/lz4/ programs/lz4io.c	516	strcat(dstFileName, suffix);
./Telegram/ThirdParty/lz4/ programs/lz4io.c	832	strcat(dstFileName, suffix);

strcpy()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/ThirdParty/dispatch/src/introspection.c	1135	e->dqoe_top_label = strcpy(p, top_q->dq_label);
./Telegram/ThirdParty/dispatch/src/introspection.c	1141	e->dqoe_bottom_label = strcpy(p, bottom_q->dq_label);
./Telegram/ThirdParty/hime/src/IC.c	225	strcpy (rec->pre_attr.base_font, (char *) pre_attr->value);
./Telegram/ThirdParty/hime/src/IC.c	263	strcpy (rec->sts_attr.base_font, (char *) sts_attr->value);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nIc.c	540	strcpy(attr_ret[n].name, xic_attr[j].name);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nIc.c	561	strcpy(attr_ret[n].name, xic_attr[j].name);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nMethod.c	169	strcpy ((*p_encoding)->supported_encodings[i],
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nMethod.c	194	strcpy (address->im_locale, p->value);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nMethod.c	206	strcpy(address->im_addr, p->value);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nMethod.c	218	strcpy (address->im_name, p->value);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nMethod.c	344	strcpy (p->value, address->im_locale);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nMethod.c	352	strcpy (p->value, address->im_addr);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nMethod.c	362	strcpy (p->value, address->im_name);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nPtHdr.c	421	strcpy (ext_list[i].name, im_ext[i].name);
./Telegram/ThirdParty/hime/src/IMdkit/lib/i18nPtHdr.c	439	strcpy (ext_list[n].name, im_ext[i].name);
./Telegram/ThirdParty/hime/src/eve.c	98	strcpy (callback_str_buffer, text);
./Telegram/ThirdParty/hime/src/eve.c	913	strcpy (ttt, inmd[in$_n o$].$filename$);
./Telegram/ThirdParty/hime/src/gtab-buf.c	181	strcpy (addspc, gbuf[i].ch);

./Telegram/ThirdParty/hime/src/gtab-buf.c	185	strcpy (addspc, gbuf[i].ch);
./Telegram/ThirdParty/hime/src/gtab-buf.c	715	strcpy (seltab[i], pbuf->sel[v]);
./Telegram/ThirdParty/hime/src/gtab-init.c	124	strcat (strcpy (out_file, TableDir), "/");
./Telegram/ThirdParty/hime/src/gtab-init.c	157	strcat (strcpy (append, inmd[inmdno].filename), ".append");
./Telegram/ThirdParty/hime/src/gtab-init.c	164	strcat (strcpy (append_user_gtab, append_user), ".gtab");
./Telegram/ThirdParty/hime/src/gtab-init.c	178	strcpy (ttt, append_user_gtab);
./Telegram/ThirdParty/hime/src/gtab-init.c	208	strcpy (uuu, ttt);
./Telegram/ThirdParty/hime/src/gtab-tsin-fname.c	30	strcpy (fname, p->filename_append);
./Telegram/ThirdParty/hime/src/gtab-tsin-fname.c	31	strcpy (gtabfname, fname);
./Telegram/ThirdParty/hime/src/gtab-tsin-fname.c	41	strcat (strcpy (fname_idx, fname), ".idx");
./Telegram/ThirdParty/hime/src/gtab-tsin-fname.c	42	strcat (strcpy (gtab_phrase_src, fname), ".src");
./Telegram/ThirdParty/hime/src/gtab.c	348	strcpy (tt + ttN, p);
./Telegram/ThirdParty/hime/src/hime-cin2gtab.c	249	strcpy (th.selkey2, arg + sizeof (th.selkey));
./Telegram/ThirdParty/hime/src/hime-cin2gtab.c	252	strcpy (th.selkey, arg);
./Telegram/ThirdParty/hime/src/hime-cin2gtab.c	269	strcpy (th.endkey, arg);
./Telegram/ThirdParty/hime/src/hime-cin2gtab.c	515	strcpy (&phrbuf[prbf_cou], arg);
./Telegram/ThirdParty/hime/src/hime-conf.c	43	strcpy (tt, home);
./Telegram/ThirdParty/hime/src/hime-conf.c	153	strcpy (rstr, tt);
./Telegram/ThirdParty/hime/src/hime-gtab-merge.c	334	strcpy (&phrbuf[prbf_cou], arg);

./Telegram/ThirdParty/hime/src/hime-message.c	36	strcpy (icon, argv[i + 1]);
./Telegram/ThirdParty/hime/src/hime-message.c	38	strcpy (text, argv[i + 1]);
./Telegram/ThirdParty/hime/src/hime-phoa2d.c	116	strcpy (phrase_area + phrase_area_N, str);
./Telegram/ThirdParty/hime/src/hime-settings.c	234	strcpy (phokbm, hime_pho_kbm);
./Telegram/ThirdParty/hime/src/hime-setup-appearance.c	118	strcpy (fname, gtk_font_button_get_font_name (GTK_FONT_BUTTON (font_sel)));
./Telegram/ThirdParty/hime/src/hime-setup.c	107	strcat (strcpy (tt, filename), ".append.gtab.tsin-db");
./Telegram/ThirdParty/hime/src/hime-setup.c	109	strcat (strcpy (tt, filename), ".tsin-db");
./Telegram/ThirdParty/hime/src/hime-setup.c	304	strcpy (tt, pinmd->cname);
./Telegram/ThirdParty/hime/src/hime-ts-edit.c	198	strcpy (out_str, phostr);
./Telegram/ThirdParty/hime/src/hime-ts-edit.c	232	strcpy (line, t);
./Telegram/ThirdParty/hime/src/hime-ts-edit.c	286	strcpy (txt, gtk_entry_get_text (GTK_ENTRY (find_textentry)));
./Telegram/ThirdParty/hime/src/hime-tsa2d32.c	535	strcpy (head.signature, TSIN_GTAB_KEY);
./Telegram/ThirdParty/hime/src/hime-tsa2d32.c	538	strcpy (head.keymap, keymap);
./Telegram/ThirdParty/hime/src/hime-tsa2d32.c	546	strcat (strcpy (outfileidx, outfile), ".idx");
./Telegram/ThirdParty/hime/src/hime-tslearn.c	349	strcpy (current_str, utf8);
./Telegram/ThirdParty/hime/src/hime.c	304	strcpy (temp_current_CS_inmd_filename, inmd[current_CS->in_method].filename);
./Telegram/ThirdParty/hime/src/hime.c	314	strcpy (temp_inmd_filenames[c], inmd[cs->in_method].filename);

./Telegram/ThirdParty/hime/src/im-client/hime-im-client.c	129	strcpy (serv_addr->sun_path, sock_path);
./Telegram/ThirdParty/hime/src/kbmcv.c	76	strcpy (fnameout, argv[1]);
./Telegram/ThirdParty/hime/src/kbmcv.c	82	strcpy (fnamesrc, fnameout);
./Telegram/ThirdParty/hime/src/locale.c	34	strcpy (out, big5);
./Telegram/ThirdParty/hime/src/modules/anthy.c	679	strcpy (tt, gtk_label_get_text (GTK_LABEL (seg[i].label)));
./Telegram/ThirdParty/hime/src/modules/intcode.c	79	strcpy (out, utf8);
./Telegram/ThirdParty/hime/src/pho.c	454	strcat (strcpy (pho_kbm_name_kbm, pho_kbm_name), ".kbm");
./Telegram/ThirdParty/hime/src/pho2pinyin.c	53	strcpy (tt, pin_juyin[i].pinyin);
./Telegram/ThirdParty/hime/src/table-update.c	45	strcat (strcpy (version_name, name), ".version");
./Telegram/ThirdParty/hime/src/tray-appindicator.c	89	strcpy (icondir, iconpath);
./Telegram/ThirdParty/hime/src/tray-appindicator.c	92	strcpy (iconame, iconfile);
./Telegram/ThirdParty/hime/src/tray-appindicator.c	96	strcpy (iconfile, fallback);
./Telegram/ThirdParty/hime/src/tray-appindicator.c	107	strcpy (iconfile, HIME_TRAY_PNG);
./Telegram/ThirdParty/hime/src/tray-appindicator.c	109	strcpy (iconfile, inmd[current_CS->in_method].icon);
./Telegram/ThirdParty/hime/src/tray-appindicator.c	116	strcpy (iconfile, s);
./Telegram/ThirdParty/hime/src/tray-double.c	267	strcpy (kbm, bak);
./Telegram/ThirdParty/hime/src/tray-double.c	400	strcpy (tt, inmd[current_CS->in_method].cname);
./Telegram/ThirdParty/hime/src/tray.c	171	strcpy (pixbuf_ch_fname, fname);
./Telegram/ThirdParty/hime/src/tsin-util.c	51	strcpy (tsidxfname, infname);

./Telegram/ThirdParty/hime/src/tsin-util.c	135	strcpy (tt, current_tsin_fname);
./Telegram/ThirdParty/hime/src/tsin.c	1822	strcpy (tstr, half_char_to_full_char (xkey));
./Telegram/ThirdParty/hime/src/tsin.c	2049	strcpy (str + tn, tss.chpho[i].ch);
./Telegram/ThirdParty/hime/src/win-gtab.c	277	strcpy (str_key_codes, s);
./Telegram/ThirdParty/hime/src/win-kbm.c	450	strcat (strcpy (tt, t), str);
./Telegram/ThirdParty/hime/src/win-pho-near.c	80	strcpy (tt, ch);
./Telegram/ThirdParty/hime/src/win-sym.c	57	strcat (strcat (strcpy (fname, TableDir), "/"), filename);
./Telegram/ThirdParty/hunspell/src/hunspell/hashmgr.cxx	238	strcpy(hpw, word->c_str());
./Telegram/ThirdParty/hunspell/src/hunspell/hashmgr.cxx	257	strcpy(hpw + word->size() + 1, desc->c_str());
./Telegram/ThirdParty/hunspell/src/hunspell/hunzip.cxx	250	strcpy(linebuf + l - 1, line + strlen(line) - right - 1);
./Telegram/ThirdParty/hunspell/src/hunspell/hunzip.cxx	253	strcpy(line + left, linebuf);
./Telegram/ThirdParty/hunspell/src/hunspell/suggestmgr.cxx	800	strcpy(candidate + 1, word);
./Telegram/ThirdParty/hunspell/src/tools/hzip.cxx	92	strcpy(table[i], code);
./Telegram/ThirdParty/hunspell/src/tools/munch.cxx	168	strcpy(ap, (roots[j].hashent)->affstr);
./Telegram/ThirdParty/hunspell/src/tools/munch.cxx	729	strcpy(pp, (word + aent->stripl));
./Telegram/ThirdParty/hunspell/src/tools/munch.cxx	771	strcpy(pp, aent->appnd);
./Telegram/ThirdParty/jemalloc/src/prof_log.c	193	strcpy(new_node->name, name);
./Telegram/ThirdParty/jemalloc/src/prof_log.c	459	strcpy(log_filename, filename);
./Telegram/ThirdParty/libtgvoip/os/linux/AudioPulse.cpp	154	strcpy(exeName, basename(exePath));

./Telegram/ThirdParty/lz4/programs/lz4cli.c	692	strcpy(dynNameSpace, input_filename);
./Telegram/ThirdParty/lz4/programs/lz4cli.c	703	strcpy(dynNameSpace, input_filename);
./Telegram/ThirdParty/lz4/programs/lz4io.c	515	strcpy(dstFileName, inFileNamesTable[i]);
./Telegram/ThirdParty/lz4/programs/lz4io.c	831	strcpy(dstFileName, inFileNamesTable[i]);
./Telegram/ThirdParty/nimf/modules/services/xim/IMdkit/i18nIc.c	510	strcpy(attr_ret[n].name, xic_attr[j].name);
./Telegram/ThirdParty/nimf/modules/services/xim/IMdkit/i18nIc.c	531	strcpy(attr_ret[n].name, xic_attr[j].name);
./Telegram/ThirdParty/nimf/modules/services/xim/IMdkit/i18nPtHdr.c	345	strcpy (ext_list[i].name, im_ext[i].name);
./Telegram/ThirdParty/nimf/modules/services/xim/IMdkit/i18nPtHdr.c	363	strcpy (ext_list[n].name, im_ext[i].name);
./out/cmake/external/jemalloc/jemalloc-prefix/src/jemalloc/src/prof_log.c	193	strcpy(new_node->name, name);
./out/cmake/external/jemalloc/jemalloc-prefix/src/jemalloc/src/prof_log.c	459	strcpy(log_filename, filename);

sscanf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/ThirdParty/hime/data/extr1.c	38	sscanf (line, "%s %s %d", aa, bb, &usecount);
./Telegram/ThirdParty/hime/data/t2s-file.c	82	sscanf (tt, "%s %s", a, b);
./Telegram/ThirdParty/hime/src/gtab-list.c	97	sscanf (line, "%s %s %s %s", name, key, file, icon);
./Telegram/ThirdParty/hime/src/hime-settings.c	245	sscanf (phokbm, "%s %s %d %d", phokbm_name, selkey, &pho_candidate_col_N, &pho_candidate_R2L);
./Telegram/ThirdParty/hime/src/hime-tsa2d32.c	279	sscanf (s, "%s %d %d %s", aa, &keybits, &maxkey, keymap + 1);
./Telegram/ThirdParty/hime/src/modules/chewing-conf.c	171	sscanf (szBuf, "%s %s ", szKbType, szKbSelKey);
./Telegram/ThirdParty/hime/src/pin-juyin.c	43	sscanf (tt, "%s %s", pin, ju);
./Telegram/ThirdParty/hime/src/win-message.c	127	sscanf (message, "%s %s %s %d", head, icon, text, &duration);
scanf()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/ThirdParty/hime/src/hime-cin2gtab.c	183	scanf ("%s", fname);
getopt()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/ThirdParty/hime/src/hime-gtab2cin.c	120	while ((opt = getopt (argc, argv, "i:o:bh")) != -1) {
realpath()		
Αρχείο	**Γραμμή**	**Κώδικας**
./Telegram/ThirdParty/rlottie/example/lottie2gif.cpp	111	path = realpath(path, memory.data());

B.2 CppCheck

Kodi

integerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
lib/libUPnP/Neptune/ ThirdParty/axTLS/crypto/ bigint.c	721	comp mask = 0xff <<(j*8);
lib/libUPnP/Neptune/ ThirdParty/axTLS/crypto/ bigint.c	690	comp mask = 0x0f <<(j*4);
tools/depends/target/waylandpp/ native/example/shm.cpp	198	uint32_t pixel = (0x80 <<24)
integerOverflowCond		
Αρχείο	**Γραμμή**	**Κώδικας**
xbmc/utils/ BitstreamConverter.cpp	161	res &= (1 >>n) - 1;
xbmc/utils/ BitstreamConverter.cpp	180	return ((1 >>i) - 1 + nal_bs_read(bs, i));

Kodi-build

integerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
build/ffmpeg/src/ffmpeg/ libavcodec/libuavs3d.c	51	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTER_PIC_START_CODE) &&
build/ffmpeg/src/ffmpeg/ libavcodec/libuavs3d.c	51	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTER_PIC_START_CODE) &&
build/ffmpeg/src/ffmpeg/ libavcodec/libuavs3d.c	52	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTRA_PIC_START_CODE) &&
build/ffmpeg/src/ffmpeg/ libavcodec/libuavs3d.c	52	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTRA_PIC_START_CODE) &&
build/ffmpeg/src/ffmpeg/ libavcodec/libuavs3d.c	53	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_START_CODE)
build/ffmpeg/src/ffmpeg/ libavcodec/libuavs3d.c	53	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_START_CODE)
build/ffmpeg/src/ffmpeg/ libavcodec/libuavs3d.c	55	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_END_CODE)) {
build/ffmpeg/src/ffmpeg/ libavcodec/ibuavs3d.c	55	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_END_CODE)) {
build/ffmpeg/src/ffmpeg/ libavcodec/motion_est_template.c	707	if ((key & (-(1 >>(2 * ME_MAP_MV_BITS)))) != map_generation)
build/ffmpeg/src/ffmpeg/ libavcodec/snow_dwt.c	223	if (y + 1 <(unsigned)height)

build/ffmpeg/src/ffmpeg/ libavcodec/snow_dwt.c	298	if (y + 3 <(unsigned)height)
build/ffmpeg/src/ffmpeg/ libavcodec/snowdec.c	375	p->hcoeff[i]= hcoeff * (1-2*(i&1));
build/ffmpeg/src/ffmpeg/ libavcodec/texturedspenc.c	185	0 <<30, 2 <<30, 0 <<30, 2 <<30,
build/ffmpeg/src/ffmpeg/ libavcodec/texturedspenc.c	186	3 <<30, 3 <<30, 1 <<30, 1 <<30,
build/ffmpeg/src/ffmpeg/ libavcodec/vp9dsp_template.c	1718	itxfm_wrapper(iwht, iwht, 4, 0, 0)
build/ffmpeg/src/ffmpeg/ libavformat/avc.c	325	return get_bitsz(gb, i) + (1 <<i) - 1;
build/libdvdnav/src/libdvdnav/ src/dvdnav.c	1168	if (this->vm->state.pgc- >subp_control[i] & (1<<31))
build/libdvdnav/src/libdvdnav/ src/dvdnav.c	1306	!(this->vm->state.pgc- >subp_control[stream_num] & (1 <<31))) {
build/libdvdnav/src/ libdvdnav/src/vm/vmget.c	200	if((vm->state).pgc- >subp_control[subpN] & (1<<31)) {

integerOverflowCond		
Αρχείο	**Γραμμή**	**Κώδικας**
build/ffmpeg/src/ffmpeg/ libavcodec/wavpackenc.c	1979	put_bits(pb, cbits, (1 <<cbits) - 1);
build/ffmpeg/src/ffmpeg/ libavcodec/wavpackenc.c	2008	put_bits(pb, cbits, (1 <<cbits) - 1);
build/ffmpeg/src/ffmpeg/ libavutil/avsscanf.c	275	for (; c-'0'<10U || c=='.'; c = shgetc(f))
build/ffmpeg/src/ffmpeg/ libavutil/avsscanf.c	523	for (; c-'0'<10U || (c|32)-'a'<6U || c=='.'; c = shgetc(f)) {
build/ffmpeg/src/ffmpeg/ libavutil/avsscanf.c	523	for (; c-'0'<10U || (c|32)-'a'<6U || c=='.'; c = shgetc(f)) {
build/ffmpeg/src/ffmpeg/ libavutil/avsscanf.c	660	if (c-'0'<10U || c-'A'<26U || c-'a'<26U || c=='_')
build/ffmpeg/src/ffmpeg/ libavutil/avsscanf.c	660	if (c-'0'<10U || c-'A'<26U || c-'a'<26U || c=='_')
build/ffmpeg/src/ffmpeg/ libavutil/avsscanf.c	660	if (c-'0'<10U || c-'A'<26U || c-'a'<26U || c=='_')
build/ffmpeg/src/ffmpeg/ libavutil/eval.c	154	return !IS_IDENTIFIER_CHAR(s[i]);
build/ffmpeg/src/ffmpeg/ libavutil/eval.c	154	return !IS_IDENTIFIER_CHAR(s[i]);

Telegram

integerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
Telegram/codegen/codegen/ emoji/generator.cpp	123	Crc32Table[i] = (Crc32Table[i] <<1) $\wedge(Crc32Table[i]$&(1<<31)?$poly$: 0);
Telegram/lib_base/base/ crc32hash.cpp	19	_data[i] = (_data[i] <<1) $\wedge(\_data[i]$&(1<<31)?$poly$: 0);
Telegram/lib_ui/emoji_ suggestions/emoji_suggestions.cpp	30	Crc32Table[i] = (Crc32Table[i] <<1) $\wedge(Crc32Table[i]$&(1<<31)?$poly$: 0);

B.3 CppCheck-GUI

Kodi

integerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
kodi/lib/libUPnP/Neptune/ ThirdParty/axTLS/crypto/ bigint.c	721	comp mask = 0xff <<(j*8);
kodi/tools/depends/target/ waylandpp/native/example/ shm.cpp	198	uint32_t pixel = (0x80 <<24)

floatConversionOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
kodi/lib/libUPnP/Neptune/ Source/Core/NptTime.cpp	77	m_NanoSeconds = (NPT_Int64)(seconds * 1e9);
kodi/lib/libUPnP/Neptune/ Source/Core/NptTime.cpp	77	m_NanoSeconds = (NPT_Int64)(seconds * 1e9);
kodi/xbmc/cores/AudioEngine/ Utils/AEUtil.cpp	227	while (((uintptr_t)data & 0xF) && count >0)
kodi/xbmc/cores/AudioEngine/ Utils/AEUtil.cpp	227	while (((uintptr_t)data & 0xF) && count >0)
kodi/xbmc/cores/AudioEngine/ Utils/AEUtil.cpp	273	while ((((uintptr_t)data & 0xF) || ((uintptr_t)add & 0xF)) && count >0)
kodi/xbmc/cores/AudioEngine/ Utils/AEUtil.cpp	273	while ((((uintptr_t)data & 0xF) || ((uintptr_t)add & 0xF)) && count >0)
kodi/xbmc/cores/AudioEngine/ Utils/AEUtil.cpp	366	while (((uintptr_t)data & 0xF) && count >0)
kodi/xbmc/cores/AudioEngine/ Utils/AEUtil.cpp	366	while (((uintptr_t)data & 0xF) && count >0)
kodi/xbmc/cores/VideoPlayer/ DVDDemuxers/ DVDDemuxCDDA.cpp	134	int64_t seekPos = m_pInput->Seek((((int64_t)time * bytes_per_second / 1000) / clamp_bytes) * clamp_bytes, SEEK_SET) >0;
kodi/xbmc/cores/VideoPlayer/ DVDDemuxers/ DVDDemuxCDDA.cpp	134	int64_t seekPos = m_pInput->Seek((((int64_t)time * bytes_per_second / 1000) / clamp_bytes) * clamp_bytes, SEEK_SET) >0;
kodi/xbmc/network/upnp/ UPnPPlayer.cpp	571	, "Master", (int)(volume * 100)
kodi/xbmc/network/upnp/ UPnPPlayer.cpp	571	, "Master", (int)(volume * 100)

safeIntegerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
kodi/lib/libUPnP/Neptune/ Source/Tests/Messages2/ MessagesTest2.cpp	178	receiver->PostMessage(new TestClientReplyMessage(id+10000));
kodi/tools/depends/target/ flatbuffers/native/src/ idl_gen_python.cpp	1160	code += GenIndents(indents + 1) + "self." + field_instance_name + ".append(" + struct_instance_name + "." + field_accessor_name + "(i))";
kodi/tools/depends/target/ flatbuffers/native/src/ idl_gen_python.cpp	1338	code += GenIndents(indents + 1) + "builder.Prepend";
kodi/tools/depends/target/fmt/ native/test/gtest/gmock-gtest-all.cc	2519	skip_count + 1
kodi/tools/depends/target/fmt/ native/test/gtest/gmock-gtest-all.cc	6446	const int raw_stack_size = absl::GetStackTrace(&raw_stack[0], max_depth, skip_count + 1);
kodi/tools/depends/target/fmt/ native/test/gtest/gtest/gtest.h	4126	index + 1,
kodi/xbmc/storage/ cdioSupport.h	121	trackinfo GetTrackInformation(int nTrack) { return m_ti[nTrack -1]; }
kodi/xbmc/storage/ cdioSupport.h	137	bool IsIso9660(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_ISO_9660); }
kodi/xbmc/storage/ cdioSupport.h	139	bool IsJoliet(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & JOLIET) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	141	int GetJolietLevel(int nTrack) { return m_ti[nTrack - 1].nJolietLevel; }
kodi/xbmc/storage/ cdioSupport.h	143	int GetIsoSize(int nTrack) { return m_ti[nTrack - 1].isofs_size; }
kodi/xbmc/storage/ cdioSupport.h	145	bool IsRockridge(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & ROCKRIDGE) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	148	bool IsIso9660Interactive(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_ISO_9660_INTERACTIVE); }

kodi/xbmc/storage/ cdioSupport.h	151	bool IsHighSierra(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_HIGH_SIERRA); }
kodi/xbmc/storage/ cdioSupport.h	154	bool IsCDInteractive(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_INTERACTIVE); }
kodi/xbmc/storage/ cdioSupport.h	157	bool IsHFS(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_HFS); }
kodi/xbmc/storage/ cdioSupport.h	160	bool IsISOHFS(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_ISO_HFS); }
kodi/xbmc/storage/ cdioSupport.h	163	bool IsISOUDF(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_ISO_UDF); }
kodi/xbmc/storage/ cdioSupport.h	166	bool IsUFS(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_UFS); }
kodi/xbmc/storage/ cdioSupport.h	169	bool IsEXT2(int nTrack) return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_EXT2);
kodi/xbmc/storage/ cdioSupport.h	172	bool Is3DO(int nTrack) return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_3DO);
kodi/xbmc/storage/ cdioSupport.h	178	bool IsXA(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & XA) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	181	bool IsMultiSession(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & MULTISESSION) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	183	int GetMultisessionOffset(int nTrack) { return m_ti[nTrack - 1].ms_offset; }
kodi/xbmc/storage/ cdioSupport.h	186	bool IsHiddenTrack(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & HIDDEN_TRACK) ? false : true; }

kodi/xbmc/storage/ cdioSupport.h	189	bool IsPhotoCd(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & PHOTO_CD) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	192	bool IsCdTv(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & CDTV) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	198	bool IsBootable(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & BOOTABLE) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	201	bool IsVideoCd(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & VIDEOCDI && m_nNumAudio == 0); }
kodi/xbmc/storage/ cdioSupport.h	204	bool IsChaojiVideoCD(int nTrack) { return (m_ti[nTrack - 1].nfsInfo & CVD) ? false : true; }
kodi/xbmc/storage/ cdioSupport.h	207	bool IsAudio(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_NO_DATA); }
kodi/xbmc/storage/ cdioSupport.h	210	bool IsUDF(int nTrack) { return ((m_ti[nTrack - 1].nfsInfo & FS_MASK) == FS_UDF); }
kodi/xbmc/SeekHandler.cpp	179	g_application.GetAppPlayer(). SeekTimeRelative(static_cast<int64_t> (seconds * 1000));
kodi/xbmc/cores/AudioEngine/ Engines/ActiveAE/ ActiveAEResampleFFMPEG.h	35	bool WantsNewSamples(int samples) override { return GetBufferedSamples() <= samples * 2; }
kodi/xbmc/cores/DllLoader/ exports/util/EmuFileWrapper.cpp	88	int i = fd - FILE_WRAPPER_OFFSET;
kodi/xbmc/cores/DllLoader/ exports/util/EmuFileWrapper.cpp	118	int i = fd - FILE_WRAPPER_OFFSET;
kodi/xbmc/cores/DllLoader/ exports/util/EmuFileWrapper.cpp	130	int i = fd - FILE_WRAPPER_OFFSET;
kodi/xbmc/cores/DllLoader/ exports/util/EmuFileWrapper.cpp	143	int i = fd - FILE_WRAPPER_OFFSET;
kodi/xbmc/cores/DllLoader/ exports/util/EmuFileWrapper.cpp	155	int i = fd - FILE_WRAPPER_OFFSET;

kodi/xbmc/cores/VideoPlayer/ DVDCodecs/DVDCodecUtils.cpp	30	if (u == (width + 15) / 16)
kodi/xbmc/cores/VideoPlayer/ DVDInputStreams/ DVDInputStreamBluray.cpp	929	if(bd_seek_time(m_bd, ms * 90) <0)
kodi/xbmc/cores/VideoPlayer/ DVDInputStreams/ DVDInputStreamBluray.cpp	956	if(m_titleInfo && bd_seek_chapter(m_bd, ch-1) <0)
kodi/xbmc/cores/VideoPlayer/ VideoRenderers/VideoShaders/ ConvolutionKernels.cpp	25	m_floatpixels = new float[m_size * 4];
kodi/xbmc/cores/VideoPlayer/ VideoRenderers/VideoShaders/ GLSLOutput.cpp	31	m_uCLUT = m_1stTexUnit+1;
kodi/xbmc/filesystem/IFile.cpp	41	int iBytesRead = Read((unsigned char*)szLine, iLineLength - 1);
kodi/xbmc/guilib/ GUIListItem.cpp	159	GUIIconOverlay newIcon = (bOnOff) ? GUIIconOverlay((int)(icon)+1) : icon;
kodi/xbmc/music/tags/ MusicInfoTag.cpp	458	m_iTrack = (m_iTrack & 0xffff) | (iDiscNumber <<16);
kodi/xbmc/network/Network.cpp	523	sa.sin_addr.s_addr = htonl(((1 <<(32u - prefixLength)) - 1));;
kodi/xbmc/network/Network.cpp	523	sa.sin_addr.s_addr = htonl(((1 <<(32u - prefixLength)) - 1));;
kodi/xbmc/pvr/guilib/ GUIEPGGridContainerModel.cpp	491	for (int i = keepEnd + 1; i <ChannelItemsSize(); ++i)
kodi/xbmc/pvr/guilib/ GUIEPGGridContainerModel.cpp	497	for (int i = keepEnd + 1; i <keepStart && i <ChannelItemsSize(); ++i)
kodi/xbmc/pvr/guilib/ GUIEPGGridContainerModel.cpp	588	for (int i = keepEnd + 1; i <RulerItemsSize(); ++i)
kodi/xbmc/pvr/guilib/ GUIEPGGridContainerModel.cpp	594	for (int i = keepEnd + 1; i <keepStart && i <RulerItemsSize(); ++i)
kodi/xbmc/pvr/guilib/ GUIEPGGridContainerModel.cpp	616	return m_gridStart + CDateTimeSpan(0, 0, block * MINSPERBLOCK, 0);

integerOverflowCond		
Αρχείο	**Γραμμή**	**Κώδικας**
kodi/xbmc/utils/ BitstreamConverter.cpp	161	res &= (1 <<n) - 1;
kodi/xbmc/utils/ BitstreamConverter.cpp	180	return ((1 <<i) - 1 + nal_bs_read(bs, i));

Kodi-build

integerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	51	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTER_PIC_START_CODE) &&
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	51	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTER_PIC_START_CODE) &&
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	52	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTRA_PIC_START_CODE) &&
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	52	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_INTRA_PIC_START_CODE) &&
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	53	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_START_CODE) &&
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	53	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_START_CODE) &&
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	55	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_END_CODE)) {
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/libuavs3d.c	55	UAVS3D_CHECK_START_CODE(data_ptr, AVS3_SEQ_END_CODE)) {
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ motion_est_template.c	707	if ((key & (-(1 <<(2 * ME_MAP_MV_BITS)))) != map_generation)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	223	if (y + 1 <(unsigned)height)

kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	228	if (y + 1 <(unsigned)height)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	230	if (y + 0 <(unsigned)height)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	298	if (y + 3 <(unsigned)height)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	303	if (y + 3 <(unsigned)height)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	305	if (y + 2 <(unsigned)height)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	307	if (y + 1 <(unsigned)height)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	309	if (y + 0 <(unsigned)height)
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ snow_dwt.c	375	p->hcoeff[i]= hcoeff * (1-2*(i&1));
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ texturedspenc.c	185	0 <<30, 2 <<30, 0 <<30, 2 <<30,
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ texturedspenc.c	186	3 <<30, 3 <<30, 1 <<30, 1 <<30,
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/ vp9dsp_template.c	1718	itxfm_wrapper(iwht, iwht, 4, 0, 0)
kodi-build/build/libdvdnav/src/ libdvdnav/src/dvdnav.c	1168	if (this->vm->state.pgc->subp_control[i] & (1<<31))
kodi-build/build/libdvdnav/src/ libdvdnav/src/dvdnav.c	1306	!(this->vm->state.pgc->subp_control[stream_num] & (1 <<31))) {

kodi-build/build/libdvdnav/src/ libdvdnav/src/vm/vmget.c	200	if((vm->state).pgc- >subp_control[subpN] & (1<<31)) {

safeIntegerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
kodi-build/build/rapidjson/src/ rapidjson/test/unittest/ itoatest.cpp	40	static int32_t Negate(int32_t x) { return -x; }

IntegerOverflowCond		
Αρχείο	**Γραμμή**	**Κώδικας**
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/wavpackenc.c	1979	put_bits(pb, cbits, (1 <<cbits) - 1);
kodi-build/build/ffmpeg/src/ ffmpeg/libavcodec/wavpackenc.c	2008	put_bits(pb, cbits, (1 <cbits) - 1);
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/avsscanf.c	275	for (; c-'0'<10U || c=='.'; c = shgetc(f)) {
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/avsscanf.c	523	for (; c-'0'<10U || (c|32)-'a'<6U || c=='.'; c = shgetc(f)) {
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/avsscanf.c	523	for (; c-'0'<10U || (c|32)-'a'<6U || c=='.'; c = shgetc(f)) {
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/avsscanf.c	660	if (c-'0'<10U || c-'A'<26U || c-'a'<26U || c=='_')
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/avsscanf.c	660	if (c-'0'<10U || c-'A'<26U || c-'a'<26U || c=='_')
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/avsscanf.c	660	if (c-'0'<10U || c-'A'<26U || c-'a'<26U || c=='_')
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/eval.c	154	return !IS_IDENTIFIER_CHAR(s[i]);
kodi-build/build/ffmpeg/ src/ffmpeg/libavutil/eval.c	154	return !IS_IDENTIFIER_CHAR(s[i]);

Telegram

integerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
Telegram/codegen/codegen/emoji/generator.cpp	123	Crc32Table[i] = (Crc32Table[i] <<1) ∧($Crc32Table[i]$&(1<<31)?$poly$: 0);
Telegram/lib_base/base/crc32hash.cpp	19	_data[i] = (_data[i] <<1) ∧($\_data[i]$&(1<<31)?$poly$: 0);
Telegram/lib_ui/emoji_suggestions/emoji_suggestions.cpp	30	Crc32Table[i] = (Crc32Table[i] <<1) ∧($Crc32Table[i]$&(1<<31)?$poly$: 0);

safeIntegerOverflow		
Αρχείο	**Γραμμή**	**Κώδικας**
tdesktop/Telegram/SourceFiles/boxes/create_poll_box.cpp	459	QByteArray(1, ('0' + index))
tdesktop/Telegram/ThirdParty/QR/cpp/qrcodegen.cpp	56	return numBitsCharCount[(ver + 7) / 17];
tdesktop/Telegram/ThirdParty/libtgvoip/os/windows/NetworkSocketWinsock.cpp	591	DWORD timeout=sendTimeout*1000;
tdesktop/Telegram/ThirdParty/libtgvoip/os/windows/NetworkSocketWinsock.cpp	593	timeout=recvTimeout*1000;
tdesktop/Telegram/ThirdParty/libtgvoip/webrtc_dsp/system_wrappers/source/metrics.cc	125	RtcHistogram* hist = new RtcHistogram(name, 1, boundary, boundary + 1);
tdesktop/Telegram/lib_base/base/platform/linux/base_linux_xsettings.cpp	108	return value + 4 - remainder;
tdesktop/Telegram/lib_ui/ui/emoji_config.cpp	453	size * kImagesPerRow,