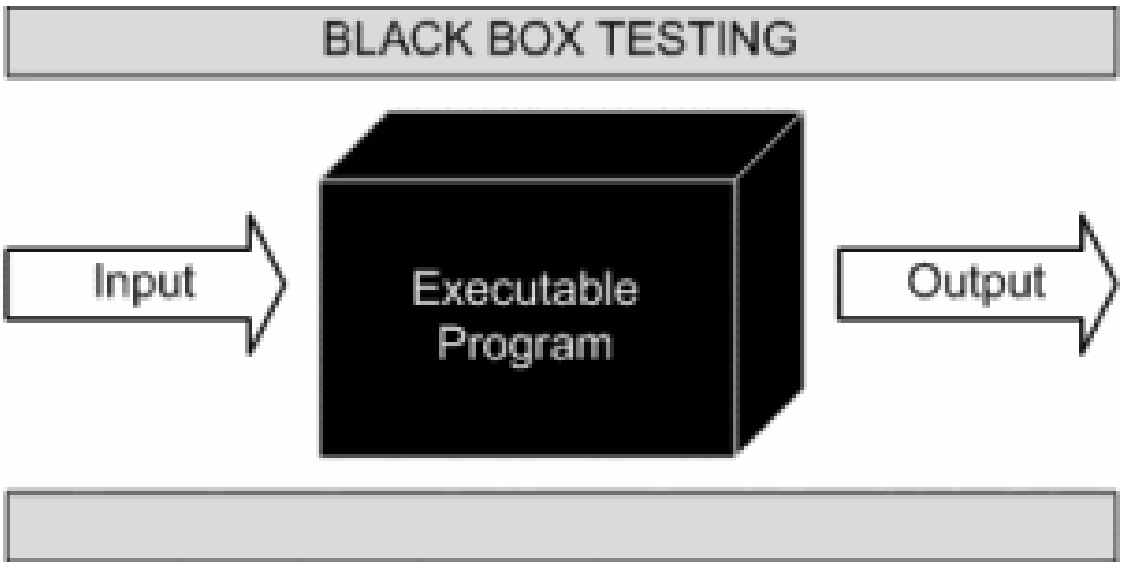
Random Testing – Black Box Testing



Εργασία : Επαλήθευση, Επικύρωση και Συντήρηση Λογισμικού

Εισαγωγή:

Το θέμα με το οποίο ασχοληθήκαμε σε αυτή την εργασία είναι το black box testing. Αυτός ο τύπος ελέγχου και γενικά όλοι οι τύποι ελέγχου δημιουργήθηκαν για να κάνουν τα λογισμικά πιο ασφαλή έτσι ώστε και οι χρήστες ναι είναι ασφαλείς και το πρόγραμμα να λειτουργεί σωστά και να βγάζει έγκυρα αποτελέσματα. Πιο συγκεκριμένα στην εργασία μας θα διαβάσετε και θα μάθετε τα πάντα σχετικά με το black box ,όπως το τι είναι και πως λειτουργεί, μια μικρή αναφορά στο white box και γιατί προτιμούμε το black box έναντι του white box κάθως καθώς και τους τρόπους και τις τεχνικές του black box που χρησιμοποιούνται για τον έλεγχο των προγραμμάτων.

Περιεχόμενα

Ιστορία του ελέγχου διείσδυσης

Στα μέσα της δεκαετίας του 1960 όπου ξεκίνησε η μεγάλη αύξηση της χρήσεις των συστημάτων υπολογιστών δημιουργήθηκαν προβλήματα ασφαλείας. Έτσι λοιπόν το 1960 σηματοδότησε την πραγματική αρχή της εποχής της ασφάλειας των υπολογιστών .Το 1965 κορυφαίοι εμπειρογνώμονες στον τομέα της ασφάλειας πραγματοποίησαν ένα από τα πρώτα μεγάλα συνέδρια για την ασφάλεια τον υπολογιστικών συστημάτων. Έπειτα από μελέτες που διεξήχθησαν σε τομείς όπως η παραβίαση της προστασίας ασφάλειας στα συστήματα ξεκίνησαν τα πρώτα αιτήματα για χρήση της διείσδυσης υπολογιστή ως εργαλείο για την μελέτη ασφάλειας των συστημάτων.

Για να κατανοήσουν καλύτερα τις αδυναμίες των συστημάτων η κυβέρνηση άρχισε σύντομα την πρόσληψη ομάδων προγραμματιστών για να χρησιμοποιούν τη διείσδυση υπολογιστών για να ελέγξουν την ασφάλεια του συστήματος. Οι ομάδες αυτές γνωστές και ως tiger team προσπάθησαν να διαλύσουν τις άμυνες των συστημάτων υπολογιστών σε μια προσπάθεια να αποκαλυφθούν, και τελικά να διορθωθούν, τρύπες ασφαλείας.

Στις αρχές του 1971, η Πολεμική Αεροπορία των ΗΠΑ ανέθεσε με σύμβαση στην ιδιωτική εταιρεία του Άντερσον να μελετήσει την ασφάλεια του συστήματος στο Πεντάγωνο. Στη μελέτη του, ο Άντερσον περιέγραψε μια σειρά από σημαντικούς παράγοντες που εμπλέκονται στη διείσδυση των υπολογιστών. Ο Άντερσον περιέγραψε μια γενική ακολουθία επίθεσης σε βήματα:

1. **<u>Συλλογή Πληροφοριών.</u>**

- Η πράξη της συλλογής σημαντικών πληροφοριών για ένα σύστημα που στόχοποιειται είναι οι πληροφορίες που μπορούν να χρησιμοποιηθούν για την επιλογή εργαλείων κατά την επίθεση του στόχου. Για παράδειγμα

  • Κοινωνική Μηχανική (Social engineering).

  • Phishing/Spear-Phishing.

  • Εύρεση σημείων εισόδου.

  • Χαρτογράφηση Δικτύου και υπολογιστών.

2. **<u>Ανάλυση πληροφοριών και Σχεδιασμός</u>**

- Χρησιμοποιεί τεχνικά εργαλεία για την περαιτέρω γνώση του συστήματος του εισβολέα. Για παράδειγμα:

  - <u>Nmap</u> μπορεί να χρησιμοποιηθεί για σάρωση για ανοιχτές θύρες.
  - <u>Nessus</u> μας ενημερώνει για το ποια αποτα network services που έχουν εντοπιστεί, έχουν γνωστές αδυναμίες και πώς μπορούμε να τις εκμεταλλευτούμε
  - Το Metasploit Framework είναι ένα πρόγραμμα ανοιχτού κώδικα που επιτρέπει στους διαχειριστές να ανιχνεύουν και να εκμεταλλεύονται κενά ασφαλείας σε πληροφοριακά συστήματα και δίκτυα.

3. **<u>Ανίχνευση Ευπαθειών.</u>**

- Χρησιμοποιώντας τα δεδομένα που συγκεντρώθηκαν στις φάσεις συλλογής πληροφοριών και ανάλυση και σχεδιασμό , ο εισβολέας ελέγχει για όλες τις γνώστες και πιθανές ευπάθειες οι οποίες χωρίζονται σε δυο φάσεις:

  • Ανίχνευση ευπαθειών ως εξωτερικός παρατηρητής

  • Ανίχνευση ευπαθειών ως χρήστης ή διαχειριστής του συστήματος

4. **Ελεγχόμενη Επίθεση / Κλιμάκωση πρόσβασης.**

- Ο εισβολέας εκμεταλλεύομενος των ελαττωματικό σχεδιασμό ή τις ευπάθιες στο σύστημα ή στο λογισισμικό προσπαθεί να αποκτήσει πρόσβαση σε υπολογιστικούς πόρους που κανονικα προστατεύνται και η χρήση τους επιτρέπεται μόνο από εξουσιοδοτημένα πρόσωπα και αναλυονται σε :

  • Κάθετη κλιμάκωση οπου χρήστες με χαμηλότερα δικαιώματα αποκτούν πρόσβαση σε ευαίσθητες λειτουργίες ή περιεχόμενο

  • Οριζόντια κλιμάκωση όπου ένας απλός χρήστης αποκτά πρόσβαση σε λειτουργίες ή περιεχόμενο που προορίζεται για άλλους απλούς χρήστες

5. **Καθαρισμός και εξαγωγή.**

- Ο εισβολέας πρέπει να διαγράψει οποιοδήποτε ίχνος που αποκαλύπτει την ταυτότητα του στο σύστημα που επιτίθεται, οποιονδήποτε τύπο δεδομένων που συλλέγεται, καταγραφή συμβάντων, προκειμένου να παραμείνει ανώνυμος.

Με την παραπάνω περιγραφή των γενικών βημάτων που έδωσε ο Άντερσον βοήθησε πολλούς ειδικούς την ασφάλειας να αξιολογήσουν τα ηλεκτρονικά τους συστήματα. Στα επόμενα χρόνια η διείσδυση των υπολογιστών εξελίχθηκε και έπειτα από μια έκθεση που υποστηρίχθηκε από τον Willis Ware έδειξε πως έμπειροι προγραμματιστές μπορούσαν να διεισδύσουν ενεργά σε υπολογιστές, να κλέψουν ή να αντιγράψουν ηλεκτρονικά αρχεία και να υπονομεύσουν τις συσκευές που συνήθως φυλάσσουν πληροφορίες απόρρητων πληροφοριών.

Τι είναι το black box:

Το black-box testing ή αλλιώς functional testing είναι η διαδικασία ελέγχου ενός ολοκληρωμένου λογισμικού στην οποία τα test cases που θα χρησιμοποιηθούν για τον έλεγχο σχεδιάζονται με βάση τις απαιτήσεις του συστήματος και δεν υπάρχει εικόνα του κώδικα του συστήματος το οποίο ελέγχεται. Είναι δηλαδή μια μέθοδος οπού ο ελεγκτής του λογισμικού δεν γνωρίζει καθόλου τον κώδικα που έχουν γράψει οι προγραμματιστές αλλά ούτε έχει και πρόσβαση σε αυτόν. Εκτελώντας λοιπόν ελέγχους με τη μέθοδο του black box testing δεν μας απασχολούν οι μηχανισμοί του συστήματος αλλά μόνο τα αποτελέσματα που παράγονται όταν βάλουμε συγκεκριμένα δεδομένα μέσα στο πρόγραμμά μας. Τον κώδικα λοιπόν μπορούμε να τον παρομοιάσουμε με ένα μεγάλο

μαύρο κουτί(black box) οπού ο άνθρωπος που επιτελεί τον έλεγχο δεν μπορεί να δει τι περιέχει μέσα αλλά η μόνη πληροφορία που μπορεί να γνωρίζει ο ελεγκτής του συστήματος είναι του τι μπορεί να εισχωρήσει ως δεδομένα μέσα στο πρόγραμμα και τι θα του εμφανίσει αυτό μετά την εκτέλεση του.

Το black box testing χρησιμοποιείται για να επαληθευτεί η λειτουργικότητα του συστήματος. Κατά την εκτέλεση του μπορούν να ανιχνευτούν προδιαγραφές του συστήματος που προκαλούν προβλήματα για να ξεκινήσει η διαδικασία της διόρθωσης τους. Το black box testing γίνεται από την πλευρά του τελικού χρήστη προκειμένου να διαπιστωθεί αν το σύστημα εκτελεί τις λειτουργίες που του έχουν οριστεί χωρίς λάθη. Αξίζει να σημειωθεί ότι κατά τη διαδικασία αυτή χρησιμοποιούνται σωστά και λανθασμένα παραδείγματα για να ελεγχθεί η λειτουργικότητα. Ο ελεγκτής του προγράμματος έπειτα από την εκτέλεσή του συγκρίνει τα αποτελέσματα και τα ελέγχει προκειμένου να διαπιστώσει ότι είναι πράγματι τα αποτελέσματα που περίμενε.

Το Black Box Testing πρόκειται λοιπόν για μια μέθοδο δοκιμής λογισμικού που αναλύει τη λειτουργικότητα ενός λογισμικού / εφαρμογής χωρίς να γνωρίζει πολλά για την εσωτερική δομή / σχεδιασμό του αντικειμένου που δοκιμάζεται και συγκρίνει την τιμή εισόδου με την τιμή εξόδου. Ο κύριος στόχος στο Black Box Testing είναι στη λειτουργικότητα του συστήματος στο σύνολό του.

Ο όρος **«Behavioral Testing»** χρησιμοποιείται επίσης για Black Box Testing. Ο σχεδιασμός δοκιμής συμπεριφοράς είναι ελαφρώς διαφορετικός από τον σχεδιασμό δοκιμών μαύρου κουτιού, επειδή η χρήση εσωτερικών γνώσεων δεν απαγορεύεται αυστηρά, αλλά εξακολουθεί να αποθαρρύνεται.

Κάθε μέθοδος δοκιμής έχει τα δικά της πλεονεκτήματα και μειονεκτήματα. Υπάρχουν κάποια σφάλματα που δεν μπορούν να βρεθούν χρησιμοποιώντας με τη τεχνική του Black Box Testing η του White Box Testing.Η πλειονότητα των εφαρμογών δοκιμάζεται με τη μέθοδο Black Box.

Με αυτήν τη μέθοδο δοκιμής είναι δυνατό να βρεθούν λανθασμένες λειτουργίες, καθώς και αρχικοποίηση και τερματισμός, συμπεριφορά ή απόδοση, δομή δεδομένων ή πρόσβαση σε εξωτερική βάση δεδομένων και σφάλματα διασύνδεσης. Οι δοκιμασμένες είσοδοι περιλαμβάνουν δεδομένα, κλικ, κύλιση, άγγιγμα, φωνή, ροές ή οποιοδήποτε άλλο μέσο που μπορεί να χρησιμοποιήσει ένας χρήστης με το λογισμικό.

Η έλλειψη γνώσης τους για την εσωτερική δομή δεν σημαίνει ότι οι δοκιμαστές μαύρου κουτιού δοκιμάζουν εντελώς τυφλούς. Ο πελάτης θα παρέχει συνήθως μια σύντομη περιγραφή του τι πρέπει να κάνει το λογισμικό (όπως περιγράφεται παρακάτω). Στην περίπτωση λειτουργικών δοκιμών και περιπτώσεων δοκιμής, το έντυπο θα καλύπτει ποιες συγκεκριμένες λειτουργίες ή χρήσεις πρέπει να δοκιμάσουν οι δοκιμαστές.

Η δοκιμή σε αυτήν τη μέθοδο μπορεί να είναι λειτουργική ή μη λειτουργική, αν και αυτό που είναι συχνότερο είναι ο λειτουργικός έλεγχος με τον σχεδιαστή να επιλέγει τις έγκυρες και μη έγκυρες εισόδους καθώς και τη σωστή έξοδο. Ο έλεγχος όλων των πιθανών συνδυασμών τιμών εισόδου για το υπό δοκιμή λογισμικό (SUT) είναι εξαντλητικός και

ανέφικτος τις περισσότερες φορές. Έτσι, η δοκιμή μαύρου κουτιού αναλύει συνήθως την είσοδο / έξοδο, παρατηρει την συμπεριφορά και τις ευρετικές μεθόδους μέσω πολλών συμπληρωματικών τεχνικών. Υπάρχουν 2 τύποι δοκιμών:

1. **Λειτουργική δοκιμή**

- Αυτή η δοκιμή ασχολείται με τις λειτουργικές απαιτήσεις ή προδιαγραφές μιας εφαρμογής. Εδώ, δοκιμάζονται διαφορετικές ενέργειες ή λειτουργίες του συστήματος παρέχοντας την είσοδο και συγκρίνοντας την πραγματική έξοδο με την αναμενόμενη έξοδο.

- Πρέπει να υπάρχει κάτι που να καθορίζει τι είναι αποδεκτή συμπεριφορά και τι όχι. Αυτό καθορίζεται σε μια λειτουργική ή προδιαγραφή απαίτησης. Είναι ένα έγγραφο που περιγράφει τι επιτρέπεται σε έναν χρήστη να το κάνει, ώστε να μπορεί να καθορίσει τη συμμόρφωση της εφαρμογής ή του συστήματος προς αυτήν. Επιπλέον, μερικές φορές αυτό θα μπορούσε επίσης να συνεπάγεται την επικύρωση των πραγματικών επιχειρηματικών σεναρίων.

- Επομένως, ο έλεγχος λειτουργικότητας μπορεί να πραγματοποιηθεί μέσω **δύο δημοφιλών τεχνικών** :

  • **Δοκιμή βάσει απαιτήσεων:** Περιέχει όλες τις λειτουργικές προδιαγραφές που αποτελούν τη βάση για όλες τις δοκιμές που πρέπει να διεξαχθούν.

  • **Δοκιμή βάσει επιχειρηματικών σεναρίων:** Περιέχει τις πληροφορίες σχετικά με τον τρόπο με τον οποίο το σύστημα θα γίνει αντιληπτό από μια προοπτική επιχειρηματικής διαδικασίας.

2. **Μη λειτουργική δοκιμή**

- Εκτός από τις λειτουργίες των απαιτήσεων, υπάρχουν και πολλές μη λειτουργικές πτυχές που πρέπει να δοκιμαστούν για τη βελτίωση της ποιότητας και της απόδοσης της εφαρμογής.

- Οι μη λειτουργικές δοκιμές γίνονται για την επαλήθευση της μη λειτουργικής απαίτησης της εφαρμογής όπως Απόδοση, Χρησιμότητα, κ.λπ.

- Επαληθεύει εάν η συμπεριφορά του συστήματος είναι σύμφωνα με την απαίτηση ή όχι. Καλύπτει όλες τις πτυχές που δεν καλύπτονται σε λειτουργικές δοκιμές . Στις καθημερινές δοκιμές, δίνεται μεγάλη προσοχή στις λειτουργικές δοκιμές και στις λειτουργικές απαιτήσεις.

Πλεονεκτήματα της δοκιμής Black Box:

Το κύριο πλεονέκτημα του black box testing είναι ότι οι ελεγκτές(testers) δεν χρειάζονται να έχουν ούτε ειδικές γνώσεις πάνω σε γλώσσες προγραμματισμού ούτε γνώσεις πάνω στην υλοποίηση. Στο black box testing και οι προγραμματιστές και ελεγκτές είναι ανεξάρτητοι ο ένας από τον άλλον. Ένα άλλο πλεονέκτημα είναι ότι ο έλεγχος γίνεται από την μεριά του

χρήστη. Αλλά το κυριότερο πλεονέκτημα του black box είναι ότι βοηθάει να αποκαλύψει οποιεσδήποτε διφορούμενες ή ασυνέπειες στις προϋποθέσεις των προδιαγραφών

Αντί να «δοκιμάσουν τον κώδικα», οι υπεύθυνοι δοκιμής μαύρου κουτιού μπορούν να ανακαλύψουν ένα εντελώς διαφορετικό σύνολο σφαλμάτων και ζητημάτων από τους αρχικούς προγραμματιστές, καθώς έρχονται στο λογισμικό χωρίς προκαταλήψεις. Στην πραγματικότητα, οι υπεύθυνοι δοκιμής μαύρου κουτιού δεν χρειάζεται καν να γνωρίζουν τη γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάπτυξη του λογισμικού, καθώς δοκιμάζουν περιπτώσεις χωρίς πρόσβαση στην εσωτερική δομή. Ο στόχος είναι να αλληλοεπιδράσουν με το λογισμικό σας όπως μπορεί ο χρήστης. Η πραγματική είσοδος χρήστη μπαίνει στο παιχνίδι αργότερα με δοκιμές beta και αποδοχής.

Η δοκιμή μαύρου κουτιού είναι επίσης πιο ανοιχτή, καθώς οι δοκιμαστές δεν γνωρίζουν τις εσωτερικές λεπτομέρειες του συστήματος ή αυτό που οι προγραμματιστές περιμένουν να κάνουν οι χρήστες. Ως αποτέλεσμα, οι δοκιμές μπορούν να ολοκληρωθούν από μη τεχνικούς συνεισφέροντες - όπως δοκιμαστές QA ή διαχειριστές προϊόντων. Αυτό σημαίνει επίσης ότι οι δοκιμές μπορούν να εκτελεστούν έξω από τον οργανισμό, είτε από ανάδοχο είτε μέσω δοκιμών crowdsourcing.

Επειδή ο στόχος της δοκιμής QA(quality assurance) είναι τελικά ο διασφαλισμός ότι η εφαρμογή πληροί το επίπεδο ποιότητας που αναμένεται από τους χρήστες, η δοκιμή μαύρου κουτιού είναι χρήσιμη επειδή μιμείται στενότερα τον τρόπο με τον οποίο οι χρήστες θα βιώσουν την εφαρμογή. Ενώ άλλες μορφές δοκιμών, συμπεριλαμβανομένων των δοκιμών μονάδας και δοκιμών API, είναι απαραίτητες για τη διαδικασία QA, η δοκιμή μαύρου κουτιού διασφαλίζει ότι όλα τα στοιχεία της εφαρμογής λειτουργούν στο επίπεδο UI.

Ένα άλλο πλεονέκτημα του black box είναι ότι τα test cases μπορούν να σχεδιαστούν μόλις ολοκληρωθούν οι προδιαγραφές, οι οποίες μπορούν να βοηθήσουν να γίνει η διαδικασία δοκιμής πιο αποτελεσματική. Αυτό παρέχει επίσης είσοδο νωρίτερα στη διαδικασία, η οποία μπορεί να μειώσει το κόστος, καθώς απαιτείται λιγότερος χρόνος για την αποκατάσταση ελαττωμάτων νωρίτερα στον κύκλο ανάπτυξης. Αυτό το καθιστά ιδιαίτερα χρήσιμο στη δοκιμή μεγάλων συστημάτων.

- Ο εισβολέας μπορεί να μην είναι προγραμματιστής.
- Χρησιμοποιείται για την επαλήθευση αντιφάσεων στο πραγματικό σύστημα και τις προδιαγραφές.
- Οι δοκιμαστικές θήκες μπορούν να σχεδιαστούν μόλις ολοκληρωθούν οι λειτουργικές προδιαγραφές
- Οι δοκιμές γίνονται από τη σκοπιά ενός χρήστη και βοηθούν να αποκαλυφθούν διαφορές στις προδιαγραφές.
- Ο ελεγκτής δεν χρειάζεται να γνωρίζει καμία γλώσσα προγραμματισμού ή να γνωρίζει πώς έχει εφαρμοστεί το λογισμικό.
- Οι δοκιμές μπορούν να πραγματοποιηθούν από ένα ανεξάρτητο σώμα των προγραμματιστών, το οποίο αποτελεί στόχο έκθεση δοκιμής αποτελέσματα. Μια

ανάπτυξη έχει το δικό της λογισμικό που μπορεί να προκαλέσει κάποια τύφλωση σε σφάλματα.

- Δοκιμαστικές περιπτώσεις μπορεί να γίνει μετά την ολοκλήρωση των προδιαγραφών.
- Ο ελεγκτής δεν χρειάζεται καμία τεχνική γνώση για να ελέγξει το σύστημα. Είναι σημαντικό να κατανοήσουμε την προοπτική του χρήστη.
- Ο έλεγχος πραγματοποιείται μετά την ανάπτυξη και οι δύο δραστηριότητες είναι ανεξάρτητες μεταξύ τους.
- Λειτουργεί για μια πιο εκτεταμένη κάλυψη που συνήθως χάνεται από τους υπεύθυνους δοκιμών καθώς δεν βλέπουν τη μεγαλύτερη εικόνα του λογισμικού.
- Οι δοκιμαστικές θήκες μπορούν να δημιουργηθούν πριν από την ανάπτυξη και αμέσως μετά τις προδιαγραφές.
- Η μεθοδολογία δοκιμής μαύρου κουτιού είναι σχεδόν ευκίνητη.
- Έχει μικρότερο κόστος συγκριτικά με το white box

Μειονεκτήματα της δοκιμής Black Box:

Η δοκιμή μαύρου κουτιού μπορεί να είναι περιττή, καθώς δοκιμάζει συχνά τις ίδιες λειτουργίες που μπορούν επίσης να δοκιμαστούν κατά τη διάρκεια της δοκιμής μονάδας(unit testing). Αυτή η μορφή δοκιμών μπορεί επίσης να είναι χρονοβόρα, προσπαθώντας να δοκιμάσετε κάθε πιθανή διαδρομή χρήστη μπορεί να οδηγήσει σε μια φουσκωμένη **δοκιμαστική σουίτα**. Προκειμένου να αξιοποιήσουν στο έπακρο τις δοκιμές μαύρου κουτιού, οι ομάδες πρέπει να επικεντρωθούν στη δημιουργία δοκιμαστικών περιπτώσεων που διασφαλίζουν ότι καλύπτονται κρίσιμες ροές χρηστών.

- Μόνο ένας μικρός αριθμός πιθανών δοκιμαστικών περιπτώσεων μπορεί να δοκιμαστεί, έτσι ώστε πολλές διαδρομές προγραμμάτων να παραμείνουν μη ελεγμένες.
- Σε πολλές περιπτώσεις οι προδιαγραφές δεν είναι σαφείς, γεγονός που καθιστά δύσκολη τη σχεδίαση των δοκιμαστικών περιπτώσεων.
- Αν ο σχεδιαστής λογισμικού ή ο προγραμματιστής έχει ήδη πραγματοποιήσει μια συγκεκριμένη δοκιμαστική περίπτωση, γίνεται μερικές φορές χωρίς λόγο.
- Στο Black Box Testing δεν είμαστε ποτέ σίγουροι αν έχουμε δοκιμάσει όλες τις σχετικές γωνίες του λογισμικού.
- Οι είσοδοι δοκιμής πρέπει να είναι από μεγάλο χώρο δείγματος.

Τι είναι το white box:

Στο white box testing ή αλλιώς structural testing ή glass box testing είναι το κομμάτι του λογισμικού που ελέγχεται και 1αντιμετωπίζεται σαν ”λευκό κουτί”. Αυτό σημαίνει ότι αυτός που κάνει τον έλεγχο έχει εποπτεία για το πως υλοποιείται το σύστημα που ελέγχει, διότι συνήθως το άτομο που εκτελεί αύτη τη δοκιμή είναι και ο προγραμματιστής επομένως γνωρίζει τα πάντα για το πρόγραμμα ,από το πώς δουλεύει μέχρι το τι δεδομένα θα εισάγει και τι αποτέλεσμα θα βγάλει. Σε σχέση με το black box testing μας ενδιαφέρουν τώρα οι εσωτερικοί μηχανισμοί του συστήματος οι οποίοι είναι στοχευμένοι στον έλεγχο ροής ή στη ροή των δεδομένων του προγράμματος. Η επιλογή των test cases σε αυτή την περίπτωση γίνεται με βάση την υλοποίηση του λογισμικού. Με
το white box testing μπορούμε να ελέγξουμε συγκεκριμένα κομμάτια του κώδικα ξεχωριστά από το υπόλοιπο πρόγραμμα, όπως μεμονωμένες συναρτήσεις. Όπως προείπαμε για να είναι αποτελεσματικό πρέπει να εκτελεστεί από έναν προγραμματιστή με πολύ καλή κατανόηση της εσωτερικής δομής του συστήματος, εφόσον τα test case που θα γραφούν θα πρέπει να είναι αρκετά εύστοχα ώστε να καλύπτουν κάθε κομμάτι του κώδικα.
Το white box testing χρησιμοποιείται κυρίως για να εντοπιστούν λογικά λάθη στον κώδικα. Είναι χρήσιμο ώστε να κάνουμε debugging τον κώδικα, να βρίσκουμε τυπογραφικά λάθη και να ανακαλύπτουμε ποιες παραδοχές ήταν λάθος κατά τη συγγραφή του προγράμματος. Μπορεί να εφαρμοστεί από χαμηλό επίπεδο του λογισμικού όπως σε
μεμονωμένα units στον κώδικα καθώς και σε ανώτερα στάδια.

Το black box και το white box testing θεωρούνται οτι αλληλεξαρτούνται. Αρκετοί ερευνητές το υπογραμμίζουν αυτό καθώς πιστεύουν οτι με αυτόν τον τρόπο τα λογισμικά έχουν ελεγχθεί πιο διεξοδικά.

Πλεονεκτήματα της δοκιμής White-box :

Τα πλεονεκτήματα της δοκιμής λευκού κουτιού περιλαμβάνουν πληρότητα, αυτοματισμό, χρόνο, βελτιστοποίηση και ενδοσκοπίες.

Επιμέλεια:

Ο κύριος μισθωτής των δοκιμών λευκού κουτιού είναι η πλήρης κάλυψη κώδικα. Βασικά, η ιδέα είναι να δοκιμάσετε όσο το δυνατόν περισσότερο τον κώδικα, ο οποίος είναι πολύ πιο λεπτομερής από τις παραδοσιακές δοκιμές μαύρου κουτιού. Η λεπτομερής φύση των δοκιμών λευκού κουτιού δίνει επίσης μια σαφή δομή στις δοκιμές. Οι δοκιμές πρέπει να είναι σαφείς, να βασίζονται στη μηχανική και να έχουν σαφώς καθορισμένους κανόνες.

Δυνατότητα αυτοματοποίησης:

Η γνώση των εσωτερικών της εφαρμογής επιτρέπει τη διενέργεια δοκιμών μονάδας. Όπως υποδηλώνει το όνομα, οι δοκιμές μονάδας δοκιμάζουν μικρά κομμάτια κώδικα, ή μονάδες, για να δουν αν λειτουργούν όπως αναμενόταν. Επειδή αυτές οι δοκιμές είναι απλές, οι προγραμματιστές μπορούν να εκτελέσουν αυτές τις δοκιμές μέσω προγραμματισμού για να δουν γρήγορα εάν κάτι έχει σπάσει. Οι δοκιμές μονάδας είναι ένας καλός τρόπος δοκιμής εάν κάτι, το οποίο προηγουμένως λειτούργησε, είχε πρόσφατα σπάσει.

Χρόνος:

Υπάρχουν πάντα προθεσμίες που πρέπει να τηρούνται κατά την ανάπτυξη λογισμικού που καθιστούν το χρόνο προτεραιότητα στη διαδικασία ανάπτυξης. Η δοκιμή λευκού κουτιού μπορεί να επιταχύνει σημαντικά τη διαδικασία δοκιμής. Συχνά, ένας προγραμματιστής μπορεί να δει ένα σφάλμα και να έχει αμέσως μια γενική ιδέα για το ποιο είναι το πρόβλημα και πώς να το διορθώσει. Επιπλέον, η δοκιμή λευκού κουτιού εξαλείφει το κόστος επικοινωνίας μεταξύ προγραμματιστών και QA, καθώς οι προγραμματιστές βρίσκουν και διορθώνουν τα προβλήματα τους χωρίς να χρειάζεται να περιμένουν το QA.

Βελτιστοποίηση:

Η μετάβαση στην ενότητα κώδικα ανά ενότητα επιτρέπει στους προγραμματιστές να αφαιρέσουν περιττές ενότητες κώδικα ή να συμπυκνώσουν τον υπάρχοντα κώδικα. Επίσης, ο κώδικας μπορεί να βελτιστοποιηθεί με την αφαίρεση κρυφών σφαλμάτων που ενδέχεται να μην εμφανιστούν κατά τη διάρκεια της κανονικής δοκιμής.

Ενδοσκόπηση:

Η δοκιμή λευκού κουτιού επιτρέπει στους προγραμματιστές να εξετάζουν προσεκτικά την εφαρμογή. Οι προγραμματιστές αναγκάζονται να εξετάσουν μεμονωμένες ενότητες κώδικα και πώς συνδέονται με άλλες ενότητες. Ίσως η τρέχουσα εφαρμογή να είναι καλή, αλλά δεν θα κλιμακωθεί καλά στο μέλλον ή έχει περιττά μέρη που μπορούν να αποκοπούν. Η δοκιμή λευκού κουτιού δίνει στους προγραμματιστές την ευκαιρία να επανεκτιμήσουν τα σχέδια και πώς θα μπορούσαν να βελτιωθούν.

Μειονεκτήματα της δοκιμής White-box :

Τα μειονεκτήματα στη δοκιμή white-box περιλαμβάνουν το κόστος, τον ταχύτατα μεταβαλλόμενο κώδικα και τις χαμένες περιπτώσεις.

Ακριβός:

Επειδή η δοκιμή λευκού κουτιού είναι πιο εμπεριστατωμένη, γίνεται πολύ ακριβή σε χρόνο και κόστος η διεξαγωγή της. Παρόλο που οι δοκιμές μονάδας το ανακουφίζουν κάπως, υπάρχει μια αρχική επένδυση που πρέπει να γίνει για τη σύνταξη των δοκιμών μονάδας. Επίσης, αυτός ο τύπος δοκιμών μπορεί να κλιμακωθεί άσχημα με μεγάλες εφαρμογές. Είναι σχεδόν αδύνατο να δοκιμάσετε κάθε κλάδο κώδικα.

Σε σύγκριση με τη δοκιμή μαύρου κουτιού, η δοκιμή λευκού κουτιού απαιτεί εξειδικευμένους δοκιμαστές με γνώσεις προγραμματισμού. Αυτό αυξάνει το κόστος και μπορεί να σημαίνει ότι οι προγραμματιστές αποσύρονται από την ανάπτυξη νέων δυνατοτήτων. Αυτά τα έξοδα πρέπει να ληφθούν υπόψη κατά τη διεξαγωγή δοκιμών λευκού κουτιού.

Γρήγορη αλλαγή της βάσης κώδικα:

Οι αυτοματοποιημένες δοκιμαστικές περιπτώσεις γίνονται σπατάλη εάν η βάση κώδικα αλλάζει γρήγορα. Συχνά, οι επανασχεδιασμοί ή οι ανακατασκευές θα κάνουν τις περισσότερες γραπτές περιπτώσεις δοκιμών να είναι άχρηστες και να χρειάζονται επανεγγραφή.

Αναπάντητες περιπτώσεις:

Η δοκιμή λευκού πλαισίου επικυρώνει μόνο και δοκιμάζει λειτουργίες που υπάρχουν αυτήν τη στιγμή. Εάν μια λειτουργία εφαρμόζεται μόνο εν μέρει ή κάτι λείπει, δεν θα γίνει δοκιμή λευκού πλαισίου. Αυτό είναι όπου οι δοκιμές που βασίζονται στις απαιτήσεις είναι μαύρες.

Η δοκιμή λευκού κουτιού έχει αρκετά σαφή πλεονεκτήματα και μειονεκτήματα. Εάν αξίζει το κόστος τα πλεονεκτήματα πρέπει να εξεταστούν προσεκτικά, ειδικά επειδή η χιλιομετρική απόσταση μπορεί να διαφέρει από έργο σε έργο.

Πώς λειτουργεί η δοκιμή Black Box :

Όπως και οι περισσότερες δοκιμές έτσι και το black box μπορεί να εκτελεστεί είτε αυτόματα είτε χειροκίνητα. Επειδή ένα από τα επιθυμητά αποτελέσματα της δοκιμής μαύρου κουτιού είναι η επιβεβαίωση ότι οι τελικοί χρήστες θα μπορούν να χρησιμοποιούν το προϊόν, ένα από τα πλεονεκτήματα της δοκιμής μαύρου κουτιού που εκτελείται από τον άνθρωπο είναι η μίμηση της εμπειρίας του χρήστη μέσω των δοκιμών. Αυτή η διαδικασία βοηθά στην εμφάνιση ζητημάτων που ενδέχεται να αντιμετωπίσουν οι χρήστες. Εναλλακτικά, η αυτοματοποιημένη δοκιμή μαύρου κουτιού παρέχει μια ταχύτερη, πιθανώς πιο επεκτάσιμη εναλλακτική μέθοδο εκτέλεσης. Ωστόσο, οι αυτοματοποιημένες δοκιμές μαύρου κουτιού απαιτούν μεγαλύτερη ποσότητα συντήρησης και τεχνικές γνώσεις από τις αντίστοιχες χειροκίνητες.

Ποιος είναι ο σκοπός της δοκιμής Black Box :

Ο κύριος σκοπός του Black Box είναι να καθορίσει εάν το λογισμικό ανταποκρίνεται στις προσδοκίες του χρήστη ή όχι.

Με αυτό το τεστ προσπαθούμε να εντοπίσουμε σφάλματα στις ακόλουθες κατηγορίες:

- Σφάλματα αρχικοποίησης και τερματισμού.
- Λανθασμένες ή ελλείπουσες λειτουργίες.
- Σφάλματα διεπαφής.

- Σφάλματα στις δομές δεδομένων ή στην πρόσβαση σε εξωτερικές βάσεις δεδομένων.
- Σφάλματα συμπεριφοράς ή εκτέλεσης.
- Σε αντίθεση με τις παραδοσιακές δοκιμές λευκού κουτιού, η δοκιμή μαύρου κουτιού είναι επωφελής για τον έλεγχο της χρηστικότητας του λογισμικού.
- Η δοκιμή μαύρου κουτιού δίνει μια ευρύτερη εικόνα του λογισμικού.
- Αυτή η δοκιμαστική προσέγγιση βλέπει μια εφαρμογή από την πλευρά του χρήστη.
- Για να δοκιμάσετε το λογισμικό στο σύνολό του αντί για διαφορετικές ενότητες.

Πότε ξεκινάνε οι δοκιμές του black box testing :

Οι δοκιμές του black box testing ξεκινάνε αμέσως και εκτελούνται παράλληλα καθόλη την διάρκεια συγγραφής του λογισμικού. Όλα τα μέλη της ομάδας που εκτελούν τον έλεγχο πρέπει να είναι από αρχή της συγγραφής ώστε όλα μέλη να γνωρίζουν τα του τι έχει γίνει από την αρχή της υλοποίησης του λογισμικού. Κατά την διάρκεια του black box testing οι ελεγκτές (tester) πρέπει να είναι ξέρουν τι συμπεράσματα έχουν προκύψει από την φάση της ανάλυσης που έγινε από τους πελάτες για να έχουν επαρκή δεδομένα .Στη φάση της σχεδίασης οι έλεγχοι των δεδομένων και των σεναρίων πρέπει ναι είναι έτοιμα.

Προετοιμασία για δοκιμή Black Box :

Η δοκιμή μαύρου κουτιού έχει πολλά πλεονεκτήματα. Ωστόσο, κατά την προετοιμασία να συνεργαστεί με δοκιμαστές μαύρου κουτιού, πληρώνει για να γνωρίζει επίσης τις προκλήσεις του.

Λαμβάνοντας υπόψη ότι οι δοκιμαστές πλησιάζουν το σχεδιασμό της δοκιμαστικής θήκης χωρίς σαφείς λειτουργικές προδιαγραφές, μπορεί να είναι πιο δύσκολο να σχεδιαστούν δοκιμαστικές θήκες. Επίσης, ο χρόνος ήταν περιορισμένος στα περισσότερα περιβάλλοντα δοκιμών, ωστόσο ο υπεύθυνος δοκιμών πρέπει να αναπτύξει συγκεκριμένες εισόδους δοκιμών και να προσδιορίσει όλες τις πιθανές εισόδους χωρίς να το γνωρίζει κάποιος. Αυτό μπορεί να οδηγήσει σε δοκιμαστές να επαναλάβουν τις δοκιμές που έχουν ήδη πραγματοποιηθεί από τον προγραμματιστή σε δοκιμές λευκού κουτιού. Τέλος, ο εξεταστής μπορεί να περιπλανηθεί σε μια άγνωστη διαδρομή κατά τη διάρκεια της δοκιμής, η οποία μπορεί ή δεν μπορεί να αποδειχθεί επωφελής.
Λοιπόν, τι μπορείτε να κάνετε για να προετοιμάσετε καλύτερα τη δοκιμή μαύρου κουτιού που ανιχνεύει τον μέγιστο αριθμό ελαττωμάτων με τον ελάχιστο αριθμό δοκιμαστικών περιπτώσεων που εκτελέστηκαν;

Ένα λεπτομερές έγγραφο ή ένας σύντομος ελεγκτής που δίνει μπορεί να περιλαμβάνει μερικά ή όλα τα ακόλουθα:

Ένα συνολικό πρόγραμμα δοκιμών

Εκκαθάριση τεκμηρίωσης του σχεδίου δοκιμών
Πιθανές είσοδοι και αναμενόμενα αποτελέσματα
δοκιμαστικών περιπτώσεων Προσεκτικά επιλεγμένες περιπτώσεις δοκιμών που αποφεύγουν την επανάληψη με προηγούμενες δοκιμές προγραμματιστή και μειώνουν τον συνολικό αριθμό δοκιμών
Προτεραιότητα των δοκιμαστικών περιπτώσεων
Ευθύνη εργασίας
Οποιοσδήποτε κίνδυνος που απαιτεί σχεδιασμό έκτακτης ανάγκης

Δοκιμή μαύρου κουτιού σε δράση :

Ας εξετάσουμε μερικές από αυτές τις τεχνικές δοκιμών μαύρου κουτιού σε δράση. Φανταστείτε, αν θέλετε, ένας δοκιμαστής να αλληλεπιδρά με μια νέα εφαρμογή παιχνιδιών κηπουρικής.

Μια πρώιμη αλληλεπίδραση με την εφαρμογή ενδέχεται να δοκιμάσει τη διαδικασία εγγραφής. Για παράδειγμα, για να δοκιμάσετε ένα πλαίσιο κειμένου εισαγωγής στο οποίο απαιτείται ένας χρήστης για να εισαγάγει την ημερομηνία γέννησής του, ο ελεγκτής μπορεί να ξεκινήσει με την κατανομή ισοδυναμίας.

Αντί να δοκιμάσουν κάθε φανταστικό αριθμό, γράμμα, ειδικό χαρακτήρα και συνδυασμό αυτών των τριών, μπορούν να προσδιορίσουν ομάδες για τις οποίες θα σχεδιάσουν δοκιμαστικές θήκες. Για παράδειγμα, θα δημιουργούσαν μια δοκιμαστική θήκη για να επιβεβαιώσουν ότι η αριθμητική είσοδος είναι λειτουργική και άλλες δοκιμαστικές περιπτώσεις για να επιβεβαιώσουν ότι οι είσοδοι αλφαβήτου, άλφα αριθμητικοί και ειδικοί χαρακτήρες θεωρούνται σφάλματα.

Ο ελεγκτής μπορεί στη συνέχεια να προχωρήσει στην ανάλυση οριακής τιμής. Η ημερομηνία γέννησης πρέπει να καταχωρηθεί ως HH-MM-EEEE. Ωστόσο, ο υπεύθυνος δοκιμών θα θέλει να προσδιορίσει τι κάνει το σύστημα σε απάντηση σε πολλές άλλες τιμές όπως:

Ένα μονοψήφιο (ελάχιστο όριο)
Έξι ψηφία (εντός ορίου)
Οκτώ ψηφία (μέγιστο)
Εννέα ψηφία (ακριβώς έξω από το όριο)

Για την ίδια εφαρμογή, Οι δοκιμαστές θα μπορούσαν επίσης να χρησιμοποιήσουν δοκιμές μετάβασης κατάστασης. Ίσως το παιχνίδι επιτρέπει στους χρήστες να συλλέγουν σπόρους για ανταλλαγή νέων φυτών, εργαλείων κήπου ή βελτιωμένων αρδευτικών συστημάτων.

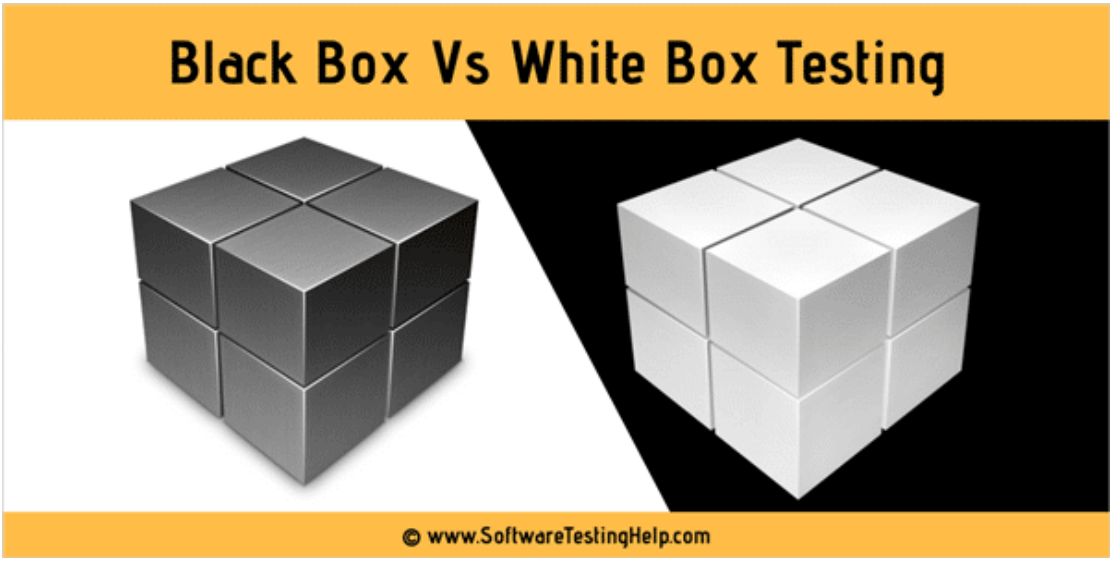
Ο υπεύθυνος δοκιμών θα πρέπει να εξετάσει τη λειτουργικότητα του λογισμικού όταν ο χρήστης ανταλλάσσει τους σπόρους του. Η κατάσταση του χρήστη μπορεί να αλλάξει από το Keeper of the Dead Plants σε Green Thumb Gardener, το οποίο θα μπορούσε να προκαλέσει νέες ενέργειες. Επιπλέον, ο χρήστης πρέπει να μπορεί να ψάξει στο απόθεμά

του (ή υπόστεγο εργαλείων) για το νέο στοιχείο. Επίσης, ο ελεγκτής μπορεί να διασφαλίσει ότι όταν ο εικονικός κηπουρός προσπαθεί να αγοράσει κάτι άλλο, η εφαρμογή γνωρίζει ότι δεν μπορεί πλέον να το αντέξει οικονομικά (έχοντας μόλις εξαντλήσει το κατάστημα των σπόρων τους).

Για ποιο λόγο χρησιμοποιούμε το black box έναντι του white box :

Το black box testing έχει ένα πολύ σημαντικό ρόλο στις δοκιμές του λογισμικού ,γιατί βοηθάει στο να επαληθεύσουμε αν το λογισμικό μας λειτουργεί ορθά και δεν έχει κενά ασφαλείας και δεν εμφανίζει απρόβλεπτα αποτελέσματα. Το black box testing γίνεται σύμφωνα με τις απαιτήσεις του πελάτη επομένως οποιεσδήποτε ατέλειωτες ή απρόβλεπτες απαίτησης μπορούν εύκολα να αναγνωριστούν και να διορθωθούν αργότερα. Το black box testing γίνεται σύμφωνα με την προοπτική του χρήστη, δηλαδή η κύρια σημασία αυτού το ελέγχου είναι ότι μπορεί να διαχειριστεί και τα σωστά και τα λάθος δεδομένα από την μεριά του πελάτη που μπορεί να εκχωρήσει.

Διαφορές Black-Box και White-Box Testing :



Το Black-Box Testing:

- Είναι μια μέθοδος δοκιμής χωρίς να γνωρίζουμε τον πραγματικό κώδικα ή την εσωτερική δομή της εφαρμογής

- Πρόκειται για δοκιμές υψηλότερου επιπέδου, όπως λειτουργικές δοκιμές .
- Επικεντρώνεται στη λειτουργικότητα του υπό δοκιμή συστήματος
- Η δοκιμή Black-Box απαιτεί προδιαγραφή απαιτήσεων για δοκιμή

To White-Box Testing:

- Είναι μια μέθοδος δοκιμής που έχεις γνώσεις σχετικά με τον πραγματικό κώδικα και την εσωτερική δομή της εφαρμογής
- Αυτός ο τύπος δοκιμών πραγματοποιείται σε χαμηλότερο επίπεδο δοκιμών, όπως Δοκιμή μονάδας, Δοκιμή ολοκλήρωσης
- Επικεντρώνεται στον πραγματικό κώδικα - πρόγραμμα και τη σύνταξη του
- Η δοκιμή White Box απαιτεί έγγραφα σχεδίασης με διαγράμματα ροής δεδομένων, διαγράμματα ροής κ.λπ.
- Η δοκιμή White Box γίνεται από προγραμματιστές με γνώσεις προγραμματισμού.

Τρόποι και τεχνικές black box:

Τεχνικές black box:

1. Equivalence Class Partitioning (έλεγχος τάξης ισοδυναμίας)

Χρησιμοποιείται ευρέως για τη σύνταξη δοκιμαστικών περιπτώσεων. Μπορεί να είναι χρήσιμη στη μείωση ενός ευρέος συνόλου πιθανών εισόδων σε μικρότερες. Πραγματοποιείται μέσω του διαχωρισμού των εισόδων ως κλάσεων και σε κάθε τάξη δίνεται μια τιμή. Εφαρμόζεται όταν προκύπτει ανάγκη για εξαντλητικές δοκιμές και για αντίσταση στον πλεονασμό των εισροών.

2. Boundary Value Analysis (ανάλυση οριακής τιμής)

Η ανάλυση της οριακής τιμής ελέγχει τα όρια των τύπων που είναι αποδεκτά από το πρόγραμμα. Αυτή η δοκιμή θα εξετάσει το μέγιστο, το ελάχιστο, ακριβώς μέσα και έξω από τα όρια, καθώς και τις τυπικές και τις τιμές σφάλματος.

3. Fuzz testing

Η εκτέλεση σπάνιων σεναρίων καθώς και τυπικών συμβάλλει στην αύξηση της πιθανότητας πρόκλησης βλάβης στο σύστημα. Οι δοκιμαστές σχεδιάζουν επίσης δοκιμαστικές θήκες για να συμπεριφέρονται με ακραίο τρόπο προκειμένου να εντοπίζουν σφάλματα. Αυτό μπορεί να περιλαμβάνει προσομοίωση μιας επίθεσης άρνησης υπηρεσίας, τεράστιου όγκου πληροφοριών που αποστέλλονται το συντομότερο δυνατό, ή τεράστιο αριθμό χρηστών Ιστού σε μια δεδομένη εφαρμογή ταυτόχρονα. Τελικά, αυτός ο τύπος δοκιμών στοχεύει να προβλέψει όλα τα τρελά πράγματα που μπορεί να κάνει ένας χρήστης για να σπάσει το σύστημα.

4. State Transition Testing(or) State Transition Diagrams(διάγραμμα κατάστασης μετάβασης)

Ένα σύστημα σπάνια βρίσκεται σε πεπερασμένη κατάσταση, επομένως αυτή η ανάλυση επιτρέπει στον ελεγκτή να δει τι συμβαίνει όταν οι εισόδους ενεργοποιούν αλλαγές ή όταν το λογισμικό μεταβαίνει μεταξύ καταστάσεων.

5. Decision Tables (πίνακας αποφάσεων)

Χρησιμοποιείται σε σύνθετα σενάρια για την εξέταση συνδυασμών εισόδων που έχουν ως αποτέλεσμα διαφορετικά αποτελέσματα, οι πίνακες αποφάσεων μπορούν να προσφέρουν στους δοκιμαστές μια συστηματική εικόνα των διαφόρων δυνατοτήτων. Αυτή η προσέγγιση για τη δοκιμή δηλώνει συχνά συνθήκες και ενέργειες στη δομή Boolean (αληθής ή ψευδής) για τη δημιουργία συνδυασμών που διαφορετικά δεν θα είχαν προβλεφθεί. Οι συνθήκες αντιπροσωπεύουν την είσοδο ενώ οι ενέργειες είναι τα συμβάντα που πρέπει να ενεργοποιήσουν οι ενέργειες.

6. Error Guessing(τεχνική εκτίμησης σφαλμάτων)

Οι έμπειροι δοκιμαστές μπορούν να χρησιμοποιήσουν τη διαίσθησή τους σε δοκιμές μαύρου κουτιού για να «μαντέψουν» πού μπορεί να υπάρχουν σφάλματα. Γνωρίζοντας πιθανά σφάλματα ή καταστάσεις που είναι επιρρεπείς σε σφάλματα, ο υπεύθυνος δοκιμών μπορεί να στοχεύει στον εντοπισμό τμημάτων κωδικού υψηλού κινδύνου ή περιοχών που θα μπορούσαν να υποβληθούν σε πιο λεπτομερή δοκιμή. Το μειονέκτημα, και γιατί αυτή η προσέγγιση δεν εφαρμόζεται από μόνη της, είναι ότι ο εξεταστής πρέπει να είναι πολύ έμπειρος (ή διορατικός) για να γράφει αποτελεσματικές περιπτώσεις δοκιμών.

7. All Pair Testing

Τα test cases έχουν σχεδιαστεί για να εκτελούν όλες τις δυνατότητες των διακριτών συνδυασμών κάθε ζεύγους παραμέτρων εισόδου. Στις δοκιμές ανά ζευγάρι πρέπει να εκτελέσουμε όλους του συνδυασμούς με όλα τα ζεύγη τιμών κατά τη διάρκεια της δοκιμής.

Περεταίρω ανάλυση των τεχνικών black box

1.Equivalence Class Partitioning

Equivalence Class Testing, γνωστό και
ως **Equivalence Class Partitioning (ECP)** και **Equivalence Partitioning** είναι μια πολύ σημαντική τεχνική ελέγχου η οποία χρησιμοποιείται από τους ελεγκτές για να ομαδοποιήσουν και για να διαχωρίσουν σε τμήματα τα εισερχόμενα δεδομένα ,τα οποία μετά από αυτές τις ενέργειες χρησιμοποιούνται για τον έλεγχο του λογισμικού σε διαφορετικές κατηγορίες.

Αυτές οι διαφορετικές κατηγορίες μοιάζουν με τις καθορισμένες απαιτήσεις και την κοινή συμπεριφορά ή τα χαρακτηριστικά των συγκεντρωτικών εισχωρήσεων. Μετά οι δοκιμαστικές υποθέσεις σχεδιάζονται και δημιουργούνται με βάση κάθε χαρακτηριστικό κλάσης και ένα στοιχείο ή είσοδος χρησιμοποιείται από κάθε τάξη για την εκτέλεση της δοκιμής για την επικύρωση της λειτουργίας του λογισμικού και ταυτόχρονα επικυρώνει την παρόμοια λειτουργία του προϊόντος λογισμικού για όλα τα άλλες εισόδους που υπάρχουν στις αντίστοιχες τάξεις τους.

Πλεονεκτήματα και μειονεκτήματα της Equivalence Class Partitioning:

Η **Equivalence Class Partitioning** παίζει ισχυρό ρόλο στη μείωση του πλεονασμού στις δοκιμές και στην ευέλικτη και ισχυρή διαδικασία. Είναι μεταξύ αυτών των τεχνικών δοκιμών που προσφέρουν πολλά οφέλη στην ομάδα και διασφαλίζει τη συμμόρφωση του προϊόντος με τις απαιτήσεις των πελατών. Ωστόσο, υπάρχουν λίγα μειονεκτήματα που σχετίζονται με αυτόν τον τύπο δοκιμών, τα οποία παρατίθενται παρακάτω μαζί με τα διάφορα πλεονεκτήματά του.

- Πλεονεκτήματα:
- Η δοκιμή κλάσης ισοδυναμίας συμβάλλει στη μείωση του αριθμού των δοκιμαστικών περιπτώσεων, χωρίς να διακυβεύεται η κάλυψη των δοκιμών.
- Μειώνει τον συνολικό χρόνο εκτέλεσης της δοκιμής καθώς ελαχιστοποιεί το σύνολο των δεδομένων δοκιμής .
- Μπορεί να εφαρμοστεί σε όλα τα επίπεδα της δοκιμής, όπως δοκιμές μονάδα , δοκιμές ολοκλήρωσης , ελέγχου του συστήματος , κλπ
- Επιτρέπει στους υπεύθυνους δοκιμών να επικεντρωθούν σε μικρότερα σύνολα δεδομένων, γεγονός που αυξάνει την πιθανότητα αποκάλυψης περισσότερων ελαττωμάτων στο προϊόν λογισμικού.
- Χρησιμοποιείται σε περιπτώσεις όπου η εκτέλεση εξαντλητικών δοκιμών είναι δύσκολη, αλλά ταυτόχρονα απαιτείται διατήρηση καλής κάλυψης.

- Μειονεκτήματα:
- Δεν λαμβάνει υπόψη τους όρους για την οριακή τιμή.
- Ο προσδιορισμός των κατηγοριών ισοδυναμίας βασίζεται σε μεγάλο βαθμό στην εμπειρία των υπευθύνων δοκιμών.
- Οι δοκιμαστές μπορεί να υποθέσουν ότι η έξοδος για όλα τα δεδομένα εισόδου δεδομένων είναι σωστή, κάτι που μπορεί να αποτελέσει μεγάλο εμπόδιο στη δοκιμή .

2.Boundary Value Analysis

Η ανάλυση οριακής τιμής είναι η πιο συχνά χρησιμοποιούμενη μέθοδος σχεδιασμού υπόθεσης για δοκιμές μαύρου κουτιού . Όπως γνωρίζουμε, το μεγαλύτερο μέρος των σφαλμάτων συμβαίνει στο όριο των τιμών εισαγωγής. Αυτή είναι μια από τις τεχνικές που χρησιμοποιούνται για την εύρεση του σφάλματος στα όρια των τιμών εισόδου και όχι στο κέντρο του εύρους τιμών εισόδου.

Ας πάρουμε ένα παράδειγμα για να το εξηγήσουμε:

Ας υποθέσουμε ότι έχουμε εφαρμογή λογισμικού που δέχεται το πλαίσιο κειμένου της τιμής εισαγωγής που κυμαίνεται από 1 έως 1000, στην περίπτωση αυτή έχουμε μη έγκυρες και έγκυρες εισόδους:

Μη έγκυρη είσοδος	Εγκυρη είσοδος	Μη έγκυρη είσοδος
0 - λιγότερο	1 - 1000	1001 - παραπάνω

Ακολουθούν οι δοκιμαστικές περιπτώσεις για την εισαγωγή αριθμών στο πλαίσιο εισαγωγής χρησιμοποιώντας ανάλυση οριακής τιμής:

Ελάχιστη τιμή - 1	0
Ελάχιστη τιμή	1
Ελάχιστη τιμή + 1	2
Κανονική τιμή	1 - 1000
Μέγιστη τιμή - 1	999
Μέγιστη τιμή	1000
Μέγιστη τιμή +1	1001

Αυτό είναι ότι οι τεχνικές δοκιμών δεν ισχύουν μόνο εάν το εύρος τιμών εισόδου δεν είναι σταθερό, δηλαδή το όριο εισόδου δεν είναι σταθερό.

- Πλεονεκτήματα και μειονεκτήματα της **Boundary Value Analysis:**

- Τα πλεονεκτήματα:
- Βοηθά στη μείωση του αριθμού των δοκιμαστικών περιπτώσεων.
- Αυτή η ανάλυση είναι πολύ βολική για τη δοκιμή λογισμικού εντατικού υπολογισμού χρησιμοποιώντας αριθμητικές οντότητες.
- Η ανάλυση οριακής τιμής διευκολύνει τη σαφήνεια στον προσδιορισμό των περιπτώσεων δοκιμής.
- Τα μειονεκτήματα:
- Δεδομένου ότι η κύρια μονάδα δοκιμάστηκε τελευταία, τα βασικά ελαττώματα στις διεπαφές που σχετίζονται με την κύρια μονάδα δοκιμάζονται στο τέλος.
- Τα προγράμματα οδήγησης δοκιμής πρέπει να δημιουργηθούν σε όλα τα επίπεδα.
- Το προϊόν δεν είναι πλήρες ή έτοιμο για κυκλοφορία μέχρι την τελευταία προσθήκη μιας μονάδας.
- Τα ίδια τα προγράμματα οδήγησης δοκιμής πρέπει να δοκιμαστούν πριν από την ανάπτυξή τους για προσέγγιση από κάτω προς τα πάνω. Αυτό αυξάνει τη συνολική προσπάθεια δοκιμών.

3.Fuzz Testing

Η Fuzz testing δοκιμή Fuzz είναι ένας τύπος δοκιμών ασφαλείας που χρησιμοποιείται από τους υπεύθυνους δοκιμών λογισμικού για την εύρεση σφαλμάτων και ελαττωμάτων στο προϊόν, για την επικύρωση της αξιοπιστίας και της ασφάλειάς του. Πιο απλά
το fuzz testing ή το fuzzing είναι μια τεχνική δοκιμών λογισμικού που βοηθά την ομάδα των δοκιμαστών να βρει ευπάθειες ασφαλείας στο λογισμικό.

Η Fuzz testing είναι μια μορφή δοκιμών ασφάλειας & black box , όπου ένας δοκιμαστής προσπαθεί να εισέλθει στο σύστημα ή στον διακομιστή ιστού με τη βοήθεια τυχαίων τιμών δεδομένων, π.χ. fuzz. Σε αυτήν τη μεθοδολογία, γενικά τα σφάλματα κωδικοποίησης και οι ευπάθειες ασφαλείας διερευνώνται τροφοδοτώντας μη έγκυρες ή τυχαίες εισόδους στο σύστημα ή την εφαρμογή λογισμικού. Μπορεί να θεωρηθεί ως αυτοματοποιημένη ή ημι-αυτοματοποιημένη διαδικασία, όπου αποκαλύπτονται σημαντικά ελαττώματα, κυρίως κενά ασφαλείας και σφάλματα, πιθανές διαρροές μνήμης κ.λπ., για να τα διορθώσουν.

- Γιατί να κάνετε δοκιμή Fuzz;

1. Συνήθως, η δοκιμή Fuzzy βρίσκει το πιο σοβαρό σφάλμα ή ελάττωμα ασφαλείας.
2. Η δοκιμή Fuzz δίνει πιο αποτελεσματικό αποτέλεσμα όταν χρησιμοποιείται με Black Box Testing , Beta Testing και άλλες μεθόδους εντοπισμού σφαλμάτων.
3. Η δοκιμή Fuzz χρησιμοποιείται για τον έλεγχο της ευπάθειας του λογισμικού. Είναι πολύ αποδοτικές τεχνικές δοκιμών.
4. Το Fuzzing είναι μια από τις πιο κοινές μεθόδους που χρησιμοποιούν οι χάκερ για να βρουν την ευπάθεια του συστήματος.

- Ο τρόπος του Fuzz testing:

Το fuzzing ακολουθεί μια αυτοματοποιημένη διαδικασία, όπου η δοκιμή μπορεί να αυτοματοποιηθεί σε υψηλό βαθμό και τα αποτελέσματα μπορούν να αξιολογηθούν και να συγκριθούν σε διάφορες διαδικτυακές εφαρμογές, προμηθευτές, λειτουργικά συστήματα και πολλά άλλα. Επιπλέον, η βασική προσέγγιση που χρησιμοποιείται για την εκτέλεση δοκιμών fuzz σε ένα προϊόν λογισμικού, αποτελείται από τις ακόλουθες δραστηριότητες:

- Προσδιορισμός του συστήματος στόχου.
- Προσδιορισμός των εισόδων.
- Δημιουργία δεδομένων ασαφών.
- Εκτέλεση με χρήση δεδομένων Fuzz.
- Παρατηρώντας τη συμπεριφορά του συστήματος.
- Ελαττώματα καταγραφής.
- Τεχνικές Fuzz testing:

Αυτός ο τύπος δοκιμών ασφαλείας μπορεί να πραγματοποιηθεί χρησιμοποιώντας οποιαδήποτε από τις ακόλουθες μεθόδους:

- Fuzzers με βάση τη μετάλλαξη: Αυτό περιλαμβάνει την τροποποίηση των υπαρχόντων και διαθέσιμων δεδομένων εισόδου για τη δημιουργία των νέων δεδομένων δοκιμής.
- Fuzzers βάσει γενιάς: Σε αυτήν την τεχνική, νέα δεδομένα δοκιμών σχεδιάζονται και προετοιμάζονται με βάση τις εισόδους του μοντέλου. Μερικά από τα παραδείγματα αυτού του μοντέλου είναι μοντέλα GUI, πρωτόκολλα δικτύου, μορφή αρχείου κ.λπ.
- Fuzzers με πρωτόκολλο ή μοντέλο: Πρόκειται για μια αποτελεσματική τεχνική, όπου σχεδιάζονται και προετοιμάζονται νέα δεδομένα δοκιμών με βάση τις γνώσεις της μορφής πρωτοκόλλου που πρόκειται να δοκιμαστεί. Γενικά, περιλαμβάνει τη σύνταξη της προδιαγραφής σε μορφή πίνακα στο εργαλείο και στη συνέχεια με

βάση την προδιαγραφή, προσθήκη παραμόρφωσης ή ελαττωμάτων στα δεδομένα εισόδου, μοτίβο, σειρά κ.λπ.

- Πλεονεκτήματα και μειονεκτήματα της Fuzz testing**:**

Ένα από τα πιο σημαντικά πλεονεκτήματα της fuzz testing είναι ότι αυτές οι δοκιμές είναι πολύ απλές για τις κατανοήσει κανείς και είναι απαλλαγμένο από τη σύλληψη σχετικά με τη συμπεριφορά του συστήματος. Ο κύριος περιορισμός με την ασαφή εύρεση σφάλματος προγράμματος είναι ότι γενικά βρίσκει μόνο πολύ απλά σφάλματα. Ένας άλλος περιορισμός με το fuzzing είναι ότι κάθε φορά που πραγματοποιούμε δοκιμές μαύρου κουτιού, έχουμε συνήθως ένα κλειστό σύστημα επίθεσης, το οποίο αυξάνει τη δυσκολία για την αξιολόγηση του αντίκτυπου της ευπάθειας που εντοπίστηκε, δηλαδή δεν είναι δυνατή η αποσφαλμάτωση. Πιο αναλυτικά διαβάστε παρακάτω.

- Πλεονεκτήματα της δοκιμής Fuzz:

Οι δοκιμές Fuzz θα πρέπει να περιλαμβάνονται σε οποιαδήποτε ολοκληρωμένη σουίτα δοκιμών, καθώς προσφέρει στην ομάδα ένα τεράστιο φάσμα πλεονεκτημάτων, τα οποία τους βοηθούν να επικυρώσουν την ποιότητα, την αποτελεσματικότητα, καθώς και την ασφάλεια του λογισμικού. Ως εκ τούτου, μερικά από αυτά τα πλεονεκτήματα είναι:

- Βελτιώνει τη δουλειά του ελέγχου ασφαλείας.
- Εξερευνήστε σοβαρά ελαττώματα, τα οποία αφήνονται αόρατα και δεν μπορούν να εξερευνηθούν, ακόμη και από τις δοκιμαστικές θήκες που σχεδιάστηκαν και προετοιμάστηκαν από έναν ειδικό εμπειρογνώμονα.
- Εξασφαλίστε την κάλυψη όλων των πιθανών αρνητικών σεναρίων για το προϊόν λογισμικού.
- Ανιχνεύει συνθήκες αγώνα και αδιέξοδα και ελέγχει την ακεραιότητα της ροής.
- Μειονεκτήματα της δοκιμής Fuzz

Εκτός από την προσφορά διαφόρων πλεονεκτημάτων στην ομάδα των υπευθύνων δοκιμών, των προγραμματιστών και των τελικών χρηστών, το fuzzing προσφέρει επίσης μερικά μειονεκτήματα, κάτι που χρειάζεται αναγνώριση. Αυτά τα μειονεκτήματα είναι:

- Απουσία κατάλληλου σχεδιασμού, καθώς και μη διαθεσιμότητα συγκεκριμένων κριτηρίων.
- Απαιτεί σημαντικό χρονικό διάστημα, για την αποτελεσματική του εκτέλεση.
- Μόνο, είναι ανίκανο να καλύψει όλες τις πιθανές αδυναμίες ασφαλείας και ελαττώματα που υπάρχουν στο προϊόν λογισμικού.

4.State Transition Testing(or) State Transition Diagrams

Όταν μιλάμε για την State Transition Testing μιλάμε για την εκτέλεση της δραστηριότητας δοκιμής μαύρου κουτιού στα προϊόντα λογισμικού, η οποία μπορεί να απεικονιστεί ή να αποσυντεθεί σε έναν πεπερασμένο αριθμό καταστάσεων. Αυτές οι

καταστάσεις συνδέονται με μία ή περισσότερες από μία καταστάσεις, εντός του λογισμικού, και είναι δυνατή η μετάβαση από μια κατάσταση σε άλλη κατάσταση, κατά την είσοδο εισόδου, έγκυρη ή μη έγκυρη, στο προϊόν λογισμικού. Αυτές οι μεταβάσεις καθοδηγούνται από τους κανόνες, για να συμπεριφέρονται διαφορετικά, όταν συναντάμε τους διαφορετικούς τύπους εισόδου.

State input	1	0
F1	F1	F2
F2	F2	F1

Μοντέλο μετάβασης

Ένα βασικό μοντέλο μετάβασης μπορεί να περιλαμβάνει τέσσερα βασικά συστατικά

- Καταστάσεις που ενδέχεται να καλύπτονται από το προϊόν λογισμικού (ανοιχτό / κλειστό ή χρηματοδοτούμενο / ανεπαρκές κεφάλαιο).
- Μεταβάσεις από τη μία κατάσταση στην άλλη (λαμβάνοντας υπόψη τη μη εφικτότητα όλων των μεταβάσεων).
- Δραστηριότητες που ενεργοποιούν τη μετάβαση (πραγματοποιώντας συγκεκριμένη ενέργεια ή τροφοδοσία εισόδων στο προϊόν λογισμικού).
- Ενέργειες που προκύπτουν από τη μετάβαση (όπως μήνυμα σφάλματος).

5.Decision Tables (πίνακες αποφάσεων)

Η τεχνική του πίνακα αποφάσεων είναι μια από τις ευρέως χρησιμοποιούμενες τεχνικές σχεδιασμού περιπτώσεων για τη δοκιμή μαύρου κουτιού. Πρόκειται για μια συστηματική προσέγγιση όπου διάφοροι συνδυασμοί εισόδου και η αντίστοιχη συμπεριφορά του συστήματος καταγράφονται σε μορφή πίνακα. Γι 'αυτό είναι επίσης γνωστό ως πίνακας αιτίων-αποτελεσμάτων. Αυτή η τεχνική χρησιμοποιείται για τη συλλογή των δοκιμαστικών περιπτώσεων με συστηματικό τρόπο. εξοικονομεί χρόνο δοκιμής και παρέχει καλή κάλυψη στην περιοχή δοκιμών της εφαρμογής λογισμικού. Η τεχνική του πίνακα αποφάσεων είναι κατάλληλη για τις συναρτήσεις που έχουν λογική σχέση μεταξύ δύο και περισσότερων από δύο εισόδων. Αυτή η τεχνική σχετίζεται με τον σωστό συνδυασμό εισόδων και καθορίζει το αποτέλεσμα διαφόρων συνδυασμών εισόδου. Για να σχεδιάσουμε τις δοκιμαστικές περιπτώσεις με τεχνική πίνακα αποφάσεων, πρέπει να θεωρήσουμε τις συνθήκες ως είσοδο και τις ενέργειες ως έξοδο.

- Γιατί προκύπτει η ανάγκη του πίνακα αποφάσεων;

Ο πρωταρχικός λόγος πίσω από τη χρήση του πίνακα αποφάσεων είναι να ασχοληθούμε με τους διαφορετικούς συνδυασμούς εισόδου στο μοντέλο και την εφαρμογή σύνθετων επιχειρηματικών λογικών και κανόνων. Χρησιμοποιείται για την εξαγωγή και το σχεδιασμό δοκιμαστικών περιπτώσεων για την εφαρμογή σύνθετων επιχειρησιακών και λειτουργικών απαιτήσεων.

Επιπλέον, ο αριθμός των συνδυασμών με τις διαθέσιμες εισόδους θα ήταν πολύ μεγαλύτερος, και θα ήταν σχεδόν αδύνατο να εξεταστεί κάθε συνδυασμός για τον σκοπό της δοκιμής. Ο πίνακας αποφάσεων επεξεργάζεται τη λύση για αυτό το πρόβλημα και παρέχει σημαντικά, μικρά και ακριβή υποσύνολα συνδυασμού εισόδου για τη διεξαγωγή δοκιμών με αποτελεσματικό τρόπο.

Ένας ακόμη σκοπός, για τη χρήση του πίνακα αποφάσεων, μπορεί να συναχθεί από το γεγονός ότι οι τεχνικές όπως η ανάλυση οριακής τιμής και η κατανομή ισοδυναμίας μπορούν να εφαρμοστούν και να χρησιμοποιηθούν για μια συγκεκριμένη κατάσταση ή δεδομένα εισαγωγής. Ωστόσο, αυτές οι τεχνικές δείχνουν αναποτελεσματικότητα στην αντιμετώπιση των διαφορετικών συνδυασμών εισόδου με αποτέλεσμα διαφορετικές ενέργειες ή ακολουθία γεγονότων. Η έλλειψη σε αυτές τις δύο τεχνικές για την αντιμετώπιση των περίπλοκων επιχειρηματικών απαιτήσεων ή κανόνων καλύπτεται από τον πίνακα αποφάσεων.

- Τα βήματα για τη χρήση των δοκιμών πίνακα αποφάσεων είναι τα παρακάτω:

1. Αναλύστε τις δεδομένες δοκιμαστικές εισόδους ή απαιτήσεις και αναφέρετε τις διάφορες συνθήκες στους πίνακες αποφάσεων.
2. Υπολογίστε τον αριθμό των πιθανών συνδυασμών (Κανόνες).
3. Συμπληρώστε στήλες του πίνακα αποφάσεων με όλους τους δυνατούς συνδυασμούς (Κανόνες).
4. Μάθετε περιπτώσεις όπου οι τιμές που αναλαμβάνονται από μια μεταβλητή δεν έχουν σημασία για έναν δεδομένο συνδυασμό. Συμπληρώστε το ίδιο με το σύμβολο "Μην με νοιάζει".
5. Για κάθε συνδυασμό τιμών, μάθετε τη δράση ή το αναμενόμενο αποτέλεσμα.
6. Δημιουργήστε τουλάχιστον μία υπόθεση δοκιμής για κάθε κανόνα. Εάν οι κανόνες είναι δυαδικοί, ένα μόνο τεστ για τον καθένα συνδυασμός είναι πιθανώς επαρκής. Διαφορετικά, εάν μια συνθήκη είναι μια σειρά τιμών, εξετάστε το ενδεχόμενο να δοκιμάσετε και τα δύο ,χαμηλό και υψηλό τέλος εύρους.

Παράδειγμα:

Έστω ότι θέλουμε να συνδεθούμε σε μια σελίδα με τον λογαριασμό μας ή οποία δεν λειτουργεί σύμφωνα το ISO 27000 και όταν βάζουμε λάθος το e-mail ή τον κωδικό μας εμφανίζει αντίστοιχα λάθος e-mail ή λάθος κωδικός.

Εάν και το email και ο κωδικός πρόσβασης ταιριάζουν , ο χρήστης θα αποκτήσει πρόσβαση στον λογαριασμό του στο mail. Αλλιώς , θα επιστρέψει στη σελίδα για να προσπαθήσει εκ νέου να συνδεθεί με ένα μήνυμα σφάλματος που καθορίζεται ανάλογα με το λάθος στοιχείου που έβαλε π.χ. "Λανθασμένο email" ή "Λανθασμένος κωδικός πρόσβασης"

Κάθε συνθήκη μπορεί να είναι είτε αληθής είτε ψευδής, δηλαδή θα υπάρχουν 2 στήλες για κάθε συνθήκη. Υπάρχει ένας γενικός τύπος για τον υπολογισμό του αριθμού των στηλών στον πίνακα αποφάσεων, ο οποίος μπορεί να θεωρηθεί ως εξής:

Αριθμός στήλης = 2^n όπου n = αριθμός συνθηκών άρα 4.

E-mail	TRUE	FALSE	TRUE	FALSE
Κωδικός	TRUE	TRUE	FALSE	FALSE
Αποτέλεσμα	Σύνδεση	Λάθος email	Λάθος κωδικός	Λάθος email

Στον πίνακα, υπάρχουν τέσσερις συνθήκες ή δοκιμαστικές περιπτώσεις για τη δοκιμή της λειτουργίας σύνδεσης.

Στην πρώτη κατάσταση, αν και το email και ο κωδικός πρόσβασης είναι σωστά, τότε ο χρήστης θα πρέπει να συνδεθεί.

Στην δεύτερη κατάσταση, εάν το email είναι λανθασμένο, αλλά ο κωδικός πρόσβασης είναι σωστός, τότε θα πρέπει να εμφανίζει “Λάθος email”.

Στη τρίτη κατάσταση, εάν το email είναι σωστό, αλλά ο κωδικός πρόσβασης είναι λανθασμένος, τότε η λειτουργία θα πρέπει να εμφανίζει “Λάθος κωδικός”.

Τώρα, στην τέταρτη και τελευταία κατάσταση, το email και ο κωδικός πρόσβασης είναι λανθασμένα, τότε η λειτουργία θα πρέπει να εμφανίζει "Λάθος email”.

- Πλεονεκτήματα της δοκιμής πίνακα αποφάσεων:

- Ο έλεγχος του πίνακα αποφάσεων καθιστά απλή τη μετατροπή πολλαπλών επιχειρηματικών εφαρμογών σε απλά σενάρια δοκιμαστικών περιπτώσεων.
- Η προσέγγιση πίνακα που χρησιμοποιείται εδώ είναι απλούστερη κατανοητή και εύκολη για οποιονδήποτε να δημιουργήσει δοκιμαστικές θήκες.
- Μια καλύτερη κάλυψη που επιτυγχάνεται με τη χρήση δοκιμών πίνακα αποφάσεων διασφαλίζει ότι η δοκιμή είναι αρκετά εξαντλητική.

6.Error Guessing(τεχνική εκτίμησης σφαλμάτων)

Είναι μια τεχνική βασισμένη σε υποθέσεις για την εκτέλεση της δοκιμαστικής δραστηριότητας , χρησιμοποιώντας τις δεξιότητες και την εμπειρία ενός

δοκιμαστή. Πρόκειται για μια μέθοδο δοκιμής βασισμένη στην εμπειρία , όπου η ομάδα δοκιμών μπορεί να κάνει την καλύτερη και αποτελεσματικότερη χρήση του εξειδικευμένου και έμπειρου ελεγκτή τους για να εκτελέσει το έργο της δοκιμής. Η εικασία σφάλματος είναι μια τέχνη, η οποία δίνει έμφαση στη λογική και αναλυτική ικανότητα ενός ελεγκτή, την οποία έχει αποκτήσει όλα αυτά τα χρόνια. Απελευθερώνει έναν δοκιμαστή, από την εξάρτηση από τις απαιτήσεις των επιχειρήσεων ή των χρηστών και από κάθε είδους στρατηγική ή σχέδιο τεκμηρίωσης .

- <u>Πότε κάνουμε την τεχνική εκτίμησης σφαλμάτων;</u>

Η τεχνικής εκίμησης σφαλμάτων γίνεται όταν ένας δοκιμαστής αντιμετωπίζει ορισμένες περιστάσεις, όπου πρέπει να κάνει τη βέλτιστη και καλύτερη χρήση των γνώσεων, των δεξιοτήτων και της αποκτηθείσας εμπειρίας του για να εκτελέσει τη δοκιμή. Κάποιοι από αυτούς είναι:

- Μη διαθεσιμότητα των προδιαγραφών και των απαιτήσεων.
- Ανεπαρκείς και ασαφείς προδιαγραφές και απαιτήσεις.
- Αυστηρές προθεσμίες.

Ωστόσο, θα είναι ιδανικό και προτιμότερο να ακολουθήσετε την
τεχνική μαντεψιάς σφαλμάτων μόνο αφού περάσετε τις επίσημες τεχνικές δοκιμών. Αυτή η προσέγγιση επιτρέπει στον ελεγκτή να έχει κάποια καλή κατανόηση του προϊόντος λογισμικού μέσω επίσημων δοκιμών, οι οποίες μπορεί να αποδειχθούν επωφελείς στην τεχνική εκτίμησης σφαλμάτων για την παραγωγή ακριβών αποτελεσμάτων.

Αυτή η τεχνική μπορεί να χρησιμοποιηθεί σε οποιοδήποτε επίπεδο δοκιμών και για τη δοκιμή των κοινών λαθών όπως:

- Διαιρέστε με μηδέν
- Εισαγωγή κενών διαστημάτων στα πεδία κειμένου
- Πατώντας το κουμπί υποβολής χωρίς να εισαγάγετε τιμές
- Μεταφόρτωση αρχείων που υπερβαίνουν τα μέγιστα όρια
- Null pointer exception
- Μη έγκυρες παράμετροι

- <u>Πλεονεκτήματα της τεχνικής εκτίμησης σφαλμάτων</u>

- Αποδεικνύεται πολύ αποτελεσματικό όταν χρησιμοποιείται σε συνδυασμό με άλλες επίσημες τεχνικές δοκιμών.

- Ανακαλύπτει τα ελαττώματα που διαφορετικά δεν θα ήταν δυνατόν να εντοπιστούν, μέσω επίσημων δοκιμών. Έτσι, η εμπειρία του ελεγκτή εξοικονομεί πολύ χρόνο και προσπάθεια.
- Η εικασία σφάλματος συμπληρώνει τις τυπικές τεχνικές σχεδιασμού δοκιμών.
- Πολύ χρήσιμο να μαντέψετε προβληματικές περιοχές της εφαρμογής.

➢ Μειονεκτήματα της τεχνικής εκτίμησης σφαλμάτων

- Το κεντρικό μειονέκτημα αυτής της τεχνικής είναι ότι εξαρτάται από το άτομο και, συνεπώς, η εμπειρία του ελεγκτή ελέγχει την ποιότητα των περιπτώσεων δοκιμής.
- Επίσης, δεν μπορεί να εγγυηθεί ότι το λογισμικό έχει φτάσει στο αναμενόμενο σημείο αναφοράς ποιότητας.
- Μόνο έμπειροι δοκιμαστές μπορούν να πραγματοποιήσουν αυτήν τη δοκιμή. Δεν μπορείς να το κάνεις από φρέσκους.

7.All Pair Testing

Η τεχνική δοκιμής όλων των ζευγών είναι επίσης γνωστή ως δοκιμή κατά ζεύγη. Χρησιμοποιείται για τη δοκιμή όλων των πιθανών διακριτών συνδυασμών τιμών. Αυτή η συνδυαστική μέθοδος χρησιμοποιείται για τη δοκιμή της εφαρμογής που χρησιμοποιεί είσοδο πλαισίου ελέγχου, είσοδος κουμπιού επιλογής (το κουμπί επιλογής χρησιμοποιείται όταν πρέπει να ορίσετε μόνο μία επιλογή για παράδειγμα όταν επιλέγετε φύλο αρσενικό ή θηλυκό, μπορείτε να επιλέξετε μόνο μία επιλογή), πλαίσιο λίστας , πλαίσιο κειμένου κ.λπ.

Παράδειγμα:

Ας υποθέσουμε ότι υπάρχει μια λειτουργία με ένα πλαίσιο λίστας που περιέχει 10 στοιχεία, πλαίσιο κειμένου που μπορεί να δεχτεί 1 έως 100 χαρακτήρες, κουμπί επιλογής, πλαίσιο ελέγχου και κουμπί OK.

Οι τιμές εισόδου δίνονται παρακάτω που μπορούν να γίνουν αποδεκτές από τα πεδία της δεδομένης συνάρτησης.

1. Πλαίσιο ελέγχου - Επιλεγμένο ή Μη επιλεγμένο
2. Πλαίσιο λίστας - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
3. Κουμπί ραδιοφώνου - Ενεργοποίηση ή απενεργοποίηση
4. Πλαίσιο κειμένου - Αριθμός αλφαβήτων μεταξύ 1 και 100.
5. OK - Δεν δέχεται καμία τιμή, αλλά ανακατευθύνει μόνο στην επόμενη σελίδα.

Υπολογισμός όλων των πιθανών συνδυασμών:

- Πλαίσιο ελέγχου = 2
- Πλαίσιο λίστας = 10
- Κουμπί ραδιοφώνου = 2
- Πλαίσιο κειμένου = 100
- Συνολικός αριθμός δοκιμαστικών περιπτώσεων = 2 * 10 * 2 * 100= 4000

Δεν είναι δυνατή η μείωση των τιμών του πλαισίου ελέγχου και του κουμπιού επιλογής, επειδή ο καθένας έχει έναν συνδυασμό μόνο 2 τιμών. Η τιμή του πλαισίου κειμένου χωρίζεται σε τρεις κατηγορίες εισόδου έγκυρος ακέραιος, μη έγκυρος ακέραιος και άλφα-ειδικός χαρακτήρας.

Τώρα, έχουμε μόνο 24 περιπτώσεις δοκιμών, συμπεριλαμβανομένων αρνητικών δοκιμαστικών περιπτώσεων.

2 * 2 * 2 * 3 = 24

Τώρα, πρέπει κάνουμε συνδυασμούς για όλες τις τεχνικές ζεύγους, στις οποίες κάθε στήλη πρέπει να έχει ίσο αριθμό τιμών και η συνολική τιμή πρέπει να είναι ίση με 24.

Για να δημιουργήσετε το Text box, τοποθετήστε την πιο κοινή είσοδο στην πρώτη θέση που είναι έγκυρος ακέραιος αριθμός , στη δεύτερη θέση βάλτε τη δεύτερη πιο κοινή είσοδο που είναι μη έγκυρος ακέραιος και στην τελευταία θέση βάλτε τη λιγότερο κοινή είσοδο που είναι ένας ειδικός χαρακτήρας .

Στη συνέχεια, αρχίστε να γεμίζετε τον πίνακα, η πρώτη στήλη είναι ένα πλαίσιο κειμένου με τρεις τιμές, η επόμενη στήλη είναι ένα πλαίσιο λίστας που έχει 2 τιμές, η τρίτη στήλη είναι ένα πλαίσιο ελέγχου που έχει 2 τιμές και η τελευταία είναι ένα κουμπί επιλογής που επίσης έχει 2 τιμές.

Text box	**List Box**	**Check box**	**Κουμπί ραδιοφώνου**
Εγκυρος ακέραιος	0	Check	Ανοιχτό
Μη έγκυρος ακέραιος	Άλλο	Uncheck	Κλειστό
Εγκυρος ακέραιος	0	Check	Ανοιχτό
Μη έγκυρος ακέραιος	Άλλο	Uncheck	Κλειστό
Ειδικός χαρακτήρας	0	Check	Ανοιχτό
Ειδικός χαρακτήρας	Άλλο	Uncheck	Κλειστό

Υπάρχουν όμως πραγματικά πολλά χρώματα δοκιμών:

- Δοκιμή κιβωτίων - ελέγχει την προειδοποίηση που χρησιμοποιείται από εφαρμογές
- Δοκιμή κόκκινου κουτιού - επικυρώνει μηνύματα σφάλματος
- Έλεγχος πράσινου κουτιού - δοκιμή απελευθέρωσης για να προσδιοριστεί εάν το σύστημα είναι φιλικό προς το περιβάλλον και δεν έχει κοινωνικές επιπτώσεις
- Δοκιμή γκρι κουτιού - συνδυάζει δοκιμές λευκού και μαύρου σε ένα είδος δοκιμής συντήρησης
- Εκκαθάριση κουτιού / γυάλινο κουτί / διαφανές τεστ κουτιού - Όλος ένας άλλος τρόπος διατύπωσης δοκιμής λευκού κουτιού
- Δοκιμή κλειστού κουτιού - ένας άλλος τρόπος για να πούμε δοκιμή μαύρου κουτιού ανεξάρτητα από το τι το ονομάζετε, η δοκιμή πρέπει να είναι επαναληπτική, καθώς αποτελεί ουσιαστικό μέρος της διαδικασίας ανάπτυξης.

Εργαλεία που χρησιμοποιούνται στο black box

Εμπορικά εργαλεία

Το παρακάτω είναι ένα δείγμα εργαλείων δοκιμής μαύρου κουτιού ασφαλείας εφαρμογών που διατίθενται στο εμπόριο.

- Cenzic Hailstorm
- IBM (formerly Internet Security Systems) Internet Security Scanner
- NT Objectives NTOSpider
- IBM (formerly Watchfire and Santum) Appscan
- Security Innovation (Holodeck)
- HP (formerly SPI Dynamics) WebInspect, DevInspect

Ανοιχτός κώδικας / δωρεάν λογισμικό

Το παρακάτω είναι μια σύντομη λίστα δειγμάτων εργαλείων σάρωσης και δοκιμών ασφάλειας εφαρμογών ανοιχτού κώδικα και δωρεάν εφαρμογών.

- Nikto
- Odysseus
- OWASP WebScarab
- Paros Proxy
- SPIKE

Επιλογή εργαλείου

Η επιλογή ενός εργαλείου δοκιμής μαύρου κουτιού μπορεί να είναι μια δύσκολη εργασία εξαιτίας του μεγάλου εύρους διαθέσιμων εμπορικών προμηθευτών και έργων ανοιχτού κώδικα σε αυτόν τον τομέα. Υπάρχουν πολλά πράγματα που πρέπει να λάβει κανείς υπόψη του πρώτου επιλέξει κάποιο πρόγραμμα που θα κάνει τις δοκιμές του, μερικά είναι τα εξής:

- κάλυψη δοκιμής και πληρότητα
- ποσοστό ακρίβειας
- χωρητικότητα της βάσης δεδομένων ευπάθειας
- ικανότητα δημιουργίας προσαρμοσμένων δοκιμών
- ευκολία στη χρήση
- δυνατότητες αναφοράς
- κόστος

Πηγες:

Βιβλιογραφια:

1. https://s3.amazonaws.com/academia.edu.documents/38077013
2. https://s3.amazonaws.com/academia.edu.documents/53989321/1011ijsea04.pdf?
3. https://www.softwaretestinghelp.com/black-box-testing
4. https://www.professionalqa.com/equivalence-class-testing
5. https://www.professionalqa.com/equivalence-class-testing
6. https://www.guru99.com/equivalence-partitioning-boundary-value-analysis.html
7. https://www.professionalqa.com/boundary-value-analysis
8. https://www.guru99.com/fuzz-testing.html
9. https://www.professionalqa.com/decision-table-testing
10. https://www.javatpoint.com/decision-table-technique-in-black-box-testing
11. https://www.professionalqa.com/error-guessing
12. https://www.javatpoint.com/all-pairs-testing-technique-in-black-box-testing
13. https://www.infosec.aueb.gr/
14. https://www.javatpoint.com/decision-table-technique-in-black-box-testing
15. https://en.wikipedia.org/wiki/Black_box
16. https://test.io/black-box-testing/