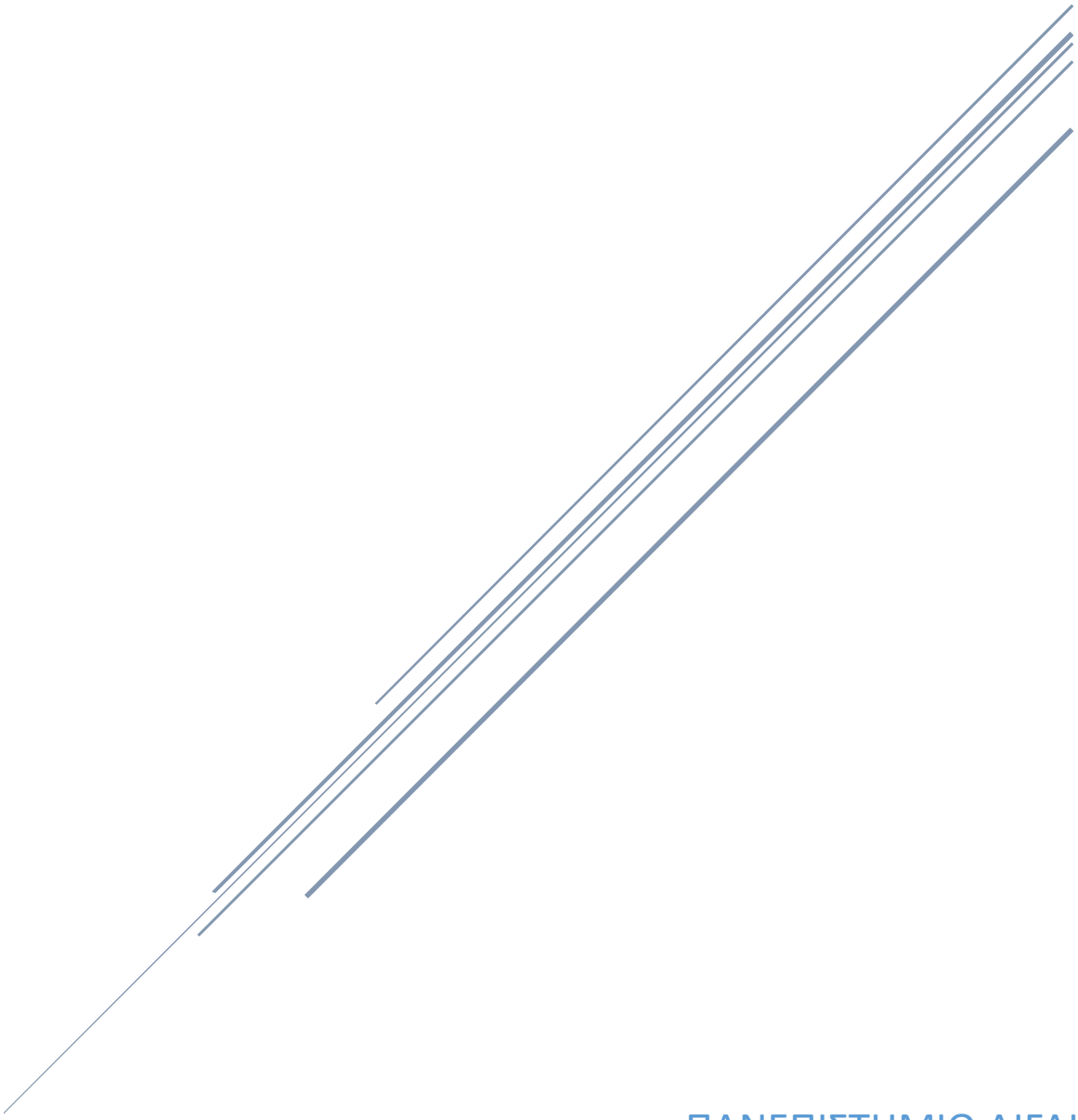
ΕΦΑΡΜΟΓΗ ΔΙΑΜΟΙΡΑΣΜΟΥ ΚΩΔΙΚΩΝ

Sharepass



ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

Εφαρμογή Διαμοιρασμού Κωδικών Sharepass

Η Διπλωματική Εργασία παρουσιάστηκε

Ενώπιων του Διδακτικού Προσωπικού του

Πανεπιστημίου Αιγαίου

Σε Μερική εκπλήρωση των Απαιτήσεων για το

Μεταπτυχιακό Δίπλωμα Ειδίκευσης (ΜΔΕ) στα

«Τηλεπικοινωνιακά και Πληροφοριακά Συστήματα»

Του

ΓΚΟΥΜΑ Μιχαήλ

(Α.Μ. 3262017002)

2019

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά την Επίκουρη Καθηγήτρια κ. Μαρία Καρύδα του Τμήματος Μηχανικών και Πληροφοριακών Συστημάτων του Πανεπιστημίου Αιγαίου γιατί χωρίς την πολύτιμη συμβολή της δεν θα μπορούσε να πραγματοποιηθεί η παρούσα εργασία.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ ΚΑΙ ΕΙΚΟΝΩΝ

ΠΕΡΙΛΗΨΗ – ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Η εργασία έχει θέμα τον ασφαλή διαμοιρασμό κωδικών μέσω της δημιουργίας προγράμματος για την παραγωγή κωδικών με την χρήση τετραψήφιων κλειδιών. Σκοπός αυτού είναι να μπορούν δύο ή περισσότερα άτομα να γνωρίζουν και να χρησιμοποιούν τον ίδιο κωδικό χωρίς ποτέ να χρειαστεί να το μοιραστούν ή να το κοινοποιήσουν μεταξύ τους με οποιονδήποτε τρόπο θέτοντας σε κίνδυνο την ακεραιότητά του. Η διαδικασία χωρίζεται σε δύο μέρη. Στο πρώτο υπάρχει το αρχείο εγκατάστασης. Αυτό δημιουργήθηκε αφ' ενός για λόγους ασφαλείας, αφ' ετέρου για λόγους χρηστικότητας. Η κάθε εγκατάσταση του προγράμματος απαιτεί την χρήση μιας φράσης ασφαλείας κάνοντας έτσι μοναδικό τον αλγόριθμο που δημιουργείται κατά την εγκατάσταση. Με αυτό τον τρόπο ασφαλίζεται το αποτέλεσμα από κακόβουλη διαρροή του αλγόριθμου, είναι δυνατή η αλλαγή του σε περιπτώσεις παραβίασης ασφαλείας αλλά και δίνεται η δυνατότητα χρήσης της εφαρμογής σε πολλές διαφορετικές ομάδες χρηστών.

Το δεύτερο μέρος είναι το κυρίως αρχείο και η παραγωγή του κωδικού. Κατά την διαδικασία αυτή θα πρέπει οι χρήστες να μοιραστούν ένα τετραψήφιο κλειδί χωρίς όμως αυτό να έχει επίδραση στην ακεραιότητα της διαδικασίας ή του κωδικού, καθώς χωρίς την σωστή εγκατάσταση του αρχείου δεν μπορεί να γίνει αντιστροφή της διαδικασίας. Μετά την εισαγωγή του κλειδιού, το πρόγραμμα επιστρέφει τον παραχθέν κωδικό καθώς και ένα τετραψήφιο κωδικό επιβεβαίωσης προκειμένου επιβεβαιωθεί η ορθότητα της διαδικασίας.

Στην παρούσα εργασία παρουσιάζονται όλα τα προγράμματα, οι τεχνικές αλλά και τα εργαλεία που χρησιμοποιήθηκαν. Επιπλέον περιλαμβάνονται και αναλύονται οι αλγόριθμοι των προγραμμάτων.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: PASSWORD, ΑΛΓΟΡΙΘΜΟΣ, ΑΣΦΑΛΕΙΑ,

ΕΦΑΡΜΟΓΗ, ΔΙΑΜΟΙΡΑΣΜΟΣ.

ΓΚΟΥΜΑΣ ΜΙΧΑΗΛ

Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

2019

ABSTRACT – KEY WORDS

This thesis is about an application to produce passwords with the use of four-digit keys. Its purpose is that two or more persons can know and use the same password without ever sharing it in any way. The production of the password is divided in two sections. The first section there is the installation file. This was created for security but also for usability reasons. The installation requires the use of a pass phrase making the algorithm installed unique every time. That way the security of the procedure is secured from malicious use, the change to the installed program is possible in case of security breach but also it is possible to use the application in different instances for different person groups.

The second part is the main file and the password production. To do so the users must share a four digit key. The sharing of this key is not of any importance as far as security is concerned because, without the proper installation of the application there is not the possibility of reversing the proses and exposing the password. After the key is inserted the application returns the password with a four digit check key. That way the validity of the procedure can be confirmed.

In this thesis every program, technique and tool used are presented. Also, the algorithms are included and explained in detail.

KEY WORDS: PASSWORD, ALGORITHM SECURITY,

APPLICATION, SHARING.

GKOUMAS MICHAIL

Department of Information and Communication Systems Engineering

UNIVERSITY OF THE AEGEAN

2019

1. ΕΙΣΑΓΩΓΗ

Όσο οι υπολογιστές «εισβάλουν» όλο και περισσότερο στην ζωή μας, η ανάγκη για ασφάλεια των λογαριασμών και των δεδομένων μας οδηγεί στη παραγωγή και χρήση όλο και περισσότερων μυστικών κωδικών ασφαλείας (passwords). Σαν κωδικός ασφαλείας εννοούμε μια μυστική λέξη η οποία χρησιμοποιείται για να προστατέψουμε την πρόσβαση του χρήστη σε κάποια πληροφορία. Από την πρόσβαση στους προσωπικούς λογαριασμούς, στο ξεκλείδωμα και την χρήση συσκευών μέχρι την προστασία αρχείων και δεδομένων οι κωδικοί ασφαλείας μας προστατεύουν από μη εξουσιοδοτημένη πρόσβαση στις πληροφορίες μας.

Υπάρχουν όμως φορές που είναι απαραίτητος ο διαμοιρασμός αυτών των κωδικών μεταξύ δύο ή και περισσοτέρων χρηστών. Παρόλο που στην πλειοψηφία των περιπτώσεων οι κωδικοί ασφαλείας είναι προσωπικοί, υπάρχουν φορές που θα πρέπει να δώσουμε πρόσβαση σε λογαριασμό ή σε προστατευμένα αρχεία, σε άλλα άτομα διατηρώντας την ασφάλεια απέναντι σε τρίτους. Ενώ λοιπόν υπάρχουν πολλά εργαλεία που μας βοηθούν να δημιουργήσουμε ή και να αποθηκεύσουμε του κωδικούς μας με σωστό τρόπο και σύμφωνα με τους κανόνες ασφαλείας, δεν υπάρχουν πολλές ενδεδειγμένες και ασφαλείς τεχνικές για να μοιραστούμε ένα κωδικό με άλλους χρήστες. Σε ένα εγγενώς ανασφαλή δίκτυο όπως το ίντερνετ, το να μοιραστείς μια τόσο ευαίσθητη πληροφορία εμπεριέχει πολλούς κινδύνους για την ακεραιότητα της.

Στόχος της εργασίας αυτής είναι να δώσει λύση στο πρόβλημα του διαμοιρασμού ενός κωδικού ασφαλείας μεταξύ δύο ή πολλών ατόμων χωρίς να τίθεται σε κίνδυνο η ακεραιότητά του. Αυτό επιτυγχάνεται με την δημιουργία μιας εφαρμογής η οποία ουσιαστικά μας επιτρέπει να παραγάγουμε τον ίδιο κωδικό κοινοποιώντας μια πληροφορία η οποία δεν θέτει σε κίνδυνο την μυστικότητα αυτού καθώς χωρίς την χρήση της εφαρμογής δεν μπορεί να γίνει συσχετισμός και παραγωγή του κωδικού ασφαλείας από μη εξουσιοδοτημένους χρήστες. Εξαλείφεται έτσι ο κίνδυνος και η ανάγκη ανεύρεσης πλήρως ασφαλούς τρόπου επικοινωνίας της πληροφορίας καθώς η κοινοποιούμενη πληροφορία δεν είναι πλέον τόσο ευαίσθητη.

Όπως θα δούμε και στην συνέχεια έχουν ληφθεί υπ' όψιν όλες οι αρχές που διέπουν την δημιουργία ενός ασφαλούς κωδικού. Από την ασφαλή εισαγωγή των ευαίσθητων πληροφοριών κατά την εγκατάσταση, μέχρι την δομή και το περιεχόμενο του ίδιο του κωδικού. Επιπλέον έχουν ληφθεί τα απαραίτητα μέτρα για την προστασία των ίδιων των αλγόριθμων της εφαρμογής και των βοηθητικών αρχείων που δημιουργούνται κατά την εγκατάσταση.

Επίσης έχει δοθεί μεγάλη σημασία στην διαφορετικότητα της κάθε εγκατάστασης. Είναι σημαντικό να μπορεί ο χρήστης να αλλάξει την εφαρμογή αλλά και να χρησιμοποιεί διαφορετικές «εκδοχές» της εφαρμογής με διαφορετικές ομάδες χρηστών. Έτσι με την παραμικρή αλλαγή κατά την εγκατάσταση της εφαρμογής γίνεται υπολογισμός νέων μεταβλητών οι οποίες διαφοροποιούν τον τελικό αλγόριθμο αρκετά ούτως ώστε να εξαλείφεται η περίπτωση ύπαρξης δύο διαφορετικών εγκαταστάσεων που να μας επιστρέφουν ίδια αποτελέσματα. Αυτό, εκτός από το χρηστικό κομμάτι, είναι σημαντικό και για λόγους ασφαλείας στις περιπτώσεις που έχουμε πιθανή ή και αποδεδειγμένη παραβίαση της ασφάλειας της εφαρμογής. Μπορούμε έτσι λοιπόν να έχουμε τον έλεγχο του ποιος θα έχει την δυνατότητα να παράξει τον ίδιο κωδικό.

Τέλος και καθώς οι αυτοματισμοί είναι σημαντικό κομμάτι της τεχνολογίας σήμερα με την άμεση επικοινωνία μεταξύ συσκευών, έχει προβλεφθεί η χρήση της εφαρμογής και από αυτοματοποιημένα συστήματα και προγράμματα με την υποστήριξη και της χρήσης μέσω γραμμής εντολών. Μέσω μιας απλής «κλήσης» της εφαρμογής μπορεί να επιστραφεί το αποτέλεσμα χωρίς το περιβάλλον χρήσης κάνοντας έτσι την διαδικασία κατάλληλη για πιο «μηχανικές» εφαρμογές. Φυσικά δίνεται η δυνατότητα χρήσης της εφαρμογής και με τρόπο φιλικό στον χρήστη. Σε αυτή την περίπτωση ο ίδιος ο χρήστης καλείται να εισάγει την πληροφορία και τα αποτελέσματα παρουσιάζονται σε αυτόν με ένα πιο φιλικό, σε αυτόν, τρόπο.

Για να μπορούμε να κατανοήσουμε καλύτερα τους όρους που θα χρησιμοποιηθούν για την ανάλυση της λειτουργίας της εφαρμογής ακολουθεί πίνακας στον οποίο περιλαμβάνονται οι βασικοί όροι που χρησιμοποιήθηκαν κατά την υλοποίηση και η έννοια αυτών:

ΠΙΝΑΚΑΣ 1 *Κατάλογος Όρων*

Μυστικός κωδικός (ή κωδικός)	Η μυστική λέξη η οποία χρησιμοποιείται για την ασφάλιση των λογαριασμών και των πληροφοριών μας. Η εφαρμογή παράγει ένα κωδικό δεκαέξι ψηφίων αποτελούμενο από συνδυασμό 91 διαφορετικών χαρακτήρων σε τυχαία σειρά
Τετραψήφιο κλειδί	Είναι ένα τετραψήφιο PIN το οποίο χρησιμοποιείται σαν εισαγωγή στην χρήση της εφαρμογής. Για κάθε διαφορετικό τετραψήφιο κλειδί και κάθε διαφορετική εγκατάσταση της εφαρμογής, επιστρέφεται ένας μοναδικός μυστικός κωδικός.
Κωδικός επιβεβαίωσης	Μαζί με τον μυστικό κωδικό, η εφαρμογή επιστρέφει και ένα τετραψήφιο κωδικό επιβεβαίωσης. Αυτό βοηθάει στην επιβεβαίωση της ορθότητας της διαδικασίας.
Μυστική φράση	Χρησιμοποιείται κατά την εγκατάσταση της εφαρμογής. Δεν έχει περιορισμό στο μέγεθος. Αποτελείται από οποιοδήποτε ειδικό χαρακτήρα (συμπεριλαμβανομένου του κενού διαστήματος) και των λατινικών χαρακτήρων.

Αφού, λοιπόν, περιγράψουμε τόσο τους κανόνες που πρέπει να πληροί ένας κωδικός για να κρίνεται ασφαλής αλλά και ορισμένες περιπτώσεις κατά τις οποίες χρειάζεται να μοιραστούμε έναν κωδικό, θα αναλύσουμε τόσο τα εργαλεία όσο και τις τεχνικές που χρησιμοποιήθηκαν για να δημιουργηθεί η εφαρμογή. Τέλος θα γίνει μια αναλυτική παρουσίαση του αλγόριθμου και των λειτουργιών αυτού.

1.1 Τα password στην ζωή μας

Είναι γεγονός ότι η χρήση των υπολογιστικών συστημάτων αυξάνεται δραστικά και μαζί με αυτή αυξάνεται και η ανάγκη να προστατέψουμε της πληροφορίες και τα δεδομένα μας. Αυτό μας οδηγεί στο να παράγουμε και να χρησιμοποιούμε όλο και περισσότερους κωδικούς ασφαλείας. Από τους τετραψήφιους κωδικούς (PIN) των καρτών στους πιο πολύπλοκους κωδικούς που προστατεύουν τους λογαριασμούς μας στις διάφορες ιστοσελίδες μέχρι και στις φράσεις ασφαλείας (passphrase) που έχουν κάνει την εμφάνισή τους σαν πιο ασφαλής τρόπος προστασίας, οι κωδικοί που πρέπει να θυμάται ο μέσος χρήστης αυξάνονται και σε πλήθος αλλά και σε πολυπλοκότητα. Αυτή ακριβώς η εξέλιξη της τεχνολογίας είναι που κάνει τα υπολογιστικά συστήματα πιο γρήγορα και πιο ικανά και αυτό εκμεταλλεύονται όσοι θέλουν να παραβιάσουν την ιδιωτικότητά μας χρησιμοποιώντας την αυξημένη υπολογιστική ισχύ για να παραβιάσουν τους κωδικούς. Και αυτό μας οδηγεί στην χρήση όλο και πιο πολύπλοκων κωδικών.

Επιπλέον το πλήθος των κωδικών που καλείται ο μέσος χρήστης να θυμάται έχει περάσει κάθε προηγούμενο και συνεχώς αυξάνεται. Σύμφωνα με την SOPHOS (Sophos, 17-10-2014) ο μέσος χρήστης χρησιμοποιεί 19 διαφορετικά passwords. Κάθε τι που κάνουμε στον ψηφιακό κόσμο πρέπει να προστατεύεται από κωδικό. Ενδεικτικά οι κωδικοί που πρέπει να θυμάται ο μέσος χρήστης πρέπει να είναι:

- PIN καρτών τραπεζών
- PIN κινητών τηλεφώνων και συσκευών
- Κωδικοί λογαριασμών email
- Κωδικοί λογαριασμών Social media
- Κωδικοί εισόδου σε ηλεκτρονικές τράπεζες (e-banking)
- Κωδικοί λογαριασμών ηλεκτρονικών αγορών
- Κωδικοί εισόδου στα οικιακά και επαγγελματικά δίκτια
- Κωδικοί κλειδώματος ευαίσθητων αρχείων και μηχανημάτων

Γίνεται λοιπόν εύκολα κατανοητό ότι ο αριθμός αυτός αλλά και η πολυπλοκότητα των κωδικών θα συνεχίσει να αυξάνεται κάνοντας την χρήση τους όλο και πιο δύσκολη για τον μέσο χρήστη. Αυτός ίσως είναι και ο βασικός λόγος που, σύμφωνα με την ίδια πηγή, ένας στους τρείς χρήστες χρησιμοποιεί ακόμη εύκολους και ανασφαλείς κωδικούς. Εν έτη 2018 η λίστα με τα χειρότερα password που χρησιμοποιούνται, σύμφωνα με την εταιρία TeamsID (JOHN HALL, n.d.) περιλαμβάνει κωδικούς που δεν πληρούν κανένα από τους κανόνες ασφαλείας.

ΠΙΝΑΚΑΣ 2 *τα χειρότερα password του 2018*

1 123456	14 666666
2 password	15 abc123
3 123456789	16 football
4 12345678	17 123123
5 12345	18 monkey
6 111111	19 654321
7 1234567	20 !@#$%^&*
8 sunshine	21 charlie
9 qwerty	22 aa123456
10 iloveyou	23 donald
11 princess	24 password1
12 admin	25 qwerty123
13 welcome	

Το πιο ανησυχητικό είναι ότι το 10% των χρηστών έχει χρησιμοποιήσει τουλάχιστον ένα από τους παραπάνω κωδικούς και το 3% έχουν χρησιμοποιήσει τον πρώτο στην λίστα «123456». Παρ’ όλα τα προγράμματα που υπάρχουν για την διατήρηση και την διαχείριση των κωδικών η χρήση τους για τον μέσο χρήστη , αποδεδειγμένα, είναι και γίνεται όλο και πιο δύσκολη.

2. ΑΡΧΕΣ ΑΣΦΑΛΕΙΑΣ ΤΩΝ ΚΩΔΙΚΩΝ

2.1 Χαρακτήρες κωδικών

Όπως περιγράψαμε στο προηγούμενο κεφάλαιο, οι κωδικοί είναι απαραίτητο να πληρούν ορισμένους κανόνες ασφαλείας. Μπορεί να είναι αυτονόητο, είναι όμως και σημαντικό να θυμόμαστε ότι οι κωδικοί αποτελούνται από χαρακτήρες που μπορεί να εισάγει ο χρήστης στο σύστημα. Δεν είναι λοιπόν τίποτε περισσότερο από συνδυασμό ενός πεπερασμένου αριθμού στοιχείων. Επειδή πρέπει ο χρήστης να έχει την δυνατότητα να εισάγει τον κωδικό του σε οποιοδήποτε σύστημα, ανεξαρτήτως γεωγραφικής θέσης ή ρυθμίσεων του συστήματος, έχουν επιλεγεί σαν κατάλληλοι χαρακτήρες τα παρακάτω:

- Οι λατινικοί χαρακτήρες κεφαλαία και μικρά (52 χαρακτήρες)
- Οι αριθμοί (10 χαρακτήρες)
- Οι ειδικοί χαρακτήρες του qwerty πληκτρολογίου (33 χαρακτήρες)

ΠΙΝΑΚΑΣ 3 *Ειδικοί Χαρακτήρες*

Space	+	@
!	,	[
"	-	\
#	.	]
$	/	^
%	:	_
&	;	`
'	<	{
(	=	|
)	>	}
*	?	~

2.2 Μήκος και πολυπλοκότητα κωδικού

Βλέπουμε λοιπόν ότι ένας κωδικός είναι ουσιαστικά μια διάταξη με επανάληψη ενός συνόλου n χαρακτήρων ανά k όπου n το πλήθος των διαφορετικών χαρακτήρων που μπορούμε να χρησιμοποιήσουμε και k το συνολικό μήκος του κωδικού μας. Σύμφωνα λοιπόν με την βασική αρχή απαρίθμησης είναι δυνατή η παραγωγή n^k διαφορετικών κωδικών. Έχοντας λοιπόν στην διάθεσή μας 95 διαφορετικούς χαρακτήρες, για ένα κωδικό μήκους 4 χαρακτήρων μπορούν να παραχθούν 95^4=81.450.625 διαφορετικοί κωδικοί. Ο αριθμός αυτός αυξάνεται εκθετικά όσο μεγαλώνουμε το μήκος του κωδικού μας.

Μπορεί οι αριθμοί αυτοί να ακούγονται τεράστιοι αλλά η αυξημένη υπολογιστική ισχύς των σύγχρονων υπολογιστικών συστημάτων και η πτώση του κόστους απόκτησης των απαραίτητων μηχανημάτων έχει κάνει πιο εύκολο και γρήγορο τον υπολογισμό και την προσπάθεια παραβίασης των κωδικών.

Σύμφωνα λοιπόν και με την εταιρία Norton (How to choose a secure password, n.d.) λαμβάνοντας υπ' όψιν τις δυνατότητες των σημερινών συστημάτων, για να έχουμε ένα ασφαλές κωδικό θα πρέπει αφ' ενός να χρησιμοποιούμε συνδυασμό όλων των χαρακτήρων που μπορούμε και να φτιάξουμε ένα κωδικό μήκους τουλάχιστον 12 χαρακτήρων. Αυτό μας δίνει την δυνατότητα να παράξουμε 95^{12}=540.360.087.662.636.962.890.625 διαφορετικούς κωδικούς. Αριθμός που μπορεί να μας δώσει μια ασφάλεια όσον αφορά τις απόπειρες δοκιμής πολλών διαφορετικών κωδικών μέχρι να «μαντέψει» κάποιος τον σωστό (επιθέσεις εξαντλητικής αναζήτησης). Αυτό ακριβώς θεωρείται και το σημείο διαχωρισμού ενός μυστικού κωδικού (password) και μιας μυστικής φράσης (passphrase). Είναι πολύ σημαντικό λοιπόν τα παράγωγα της εφαρμογής μας να έχουν μήκος μεγαλύτερο από δώδεκα χαρακτήρες για να μπορούμε να πούμε ότι έχουμε ένα ασφαλές για χρήση αποτέλεσμα.

2.3 Περιεχόμενο κωδικών

Δυστυχώς όμως αυτό δεν είναι από μόνο του αρκετό για να μας δώσει ένα ασφαλή κωδικό κυρίως όταν αυτός παράγεται από τον χρήστη και όχι από κάποιο πρόγραμμα. Και αυτό γιατί ο χρήστης έχει την ανάγκη να θυμάται τον κωδικό ακόμη και μετά από ένα μεγάλο χρονικό διάστημα μη χρήσης του. Έτσι παρά τον τεράστιο διαθέσιμο αριθμό διαφορετικών κωδικών αυτή η ανάγκη του ανθρώπου τον οδηγεί να χρησιμοποιεί λέξεις και λήμματα που μπορεί να αναγνωρίσει.

Αυτό όπως γίνεται εύκολα κατανοητό περιορίζει κατά πολύ τον συνολικό αριθμό διαφορετικών κωδικών που παράγουμε ως χρήστες. Η Ελληνική γλώσσα, η οποία θεωρείται μια από τις πιο πλούσιες, σύμφωνα με τον Καθηγητή και Πρύτανη του Πανεπιστημίου Αθηνών κ. Γεώργιο Μπαμπινιώτη έχει περίπου 100.000 διαφορετικές λέξεις και 300.000 σημασίες (Kalinda, 11-11-2016). Επίσης σύμφωνα με τον Καθηγητή της Γλωσσολογίας στη Φιλοσοφική Σχολή του Πανεπιστημίου Αθηνών κ. Χριστόφορου Χαραλαμπάκη '*Στο Μέγα Λεξικόν όλης της ελληνικής γλώσσης του Δ. Δημητράκου υπάρχουν γύρω στα 200.000 λήμματα. Η νέα ελληνική με τις διαλέκτους και τα ιδιώματά της διαθέτει σήμερα περισσότερες από μισό εκατομμύριο λέξεις, όσες περίπου και η Αγγλική, παγκόσμια γλώσσα με τεράστιο κύρος. Στον αριθμό αυτό συμπεριλαμβάνονται οι επιστημονικοί όροι που βρίσκονται σε μεγάλη διάδοση*' (Χριστόφορος Χαραλαμπάκης, 02-05-2007).

Καταλαβαίνουμε λοιπόν ότι όποιος και αν είναι ο πραγματικός συνολικός λημμάτων και όσες παραλλαγές αυτών και αν χρησιμοποιήσουμε δεν μπορούμε να φτάσουμε το τεράστιο αριθμό διαφορετικών κωδικών που μπορεί να μας δώσει μια πιο «ψυχρή» και μηχανική προσέγγιση. Επιπλέον ακόμη και αν χρησιμοποιήσουμε την τεχνική της αντικατάστασης ορισμένων χαρακτήρων με αντίστοιχους ειδικούς χαρακτήρες, παρά το αυξημένο πλήθος πιθανών κωδικών, η παραβίασή τους γίνεται εύκολη καθώς υπάρχει ένα πρότυπο που μπορεί να ακολουθηθεί από κάποιον που θέλει να παραβιάσει τον κωδικό (επιθέσεις λεξικού). Έτσι δύο ακόμη «κανόνες» μπορούμε να πούμε ότι είναι να μην χρησιμοποιούμε λέξεις που μπορούν να βρεθούν σε λεξικό και να μην χρησιμοποιούμε εμφανείς αντικαταστάσεις χαρακτήρων.

2.4 Κανόνες ασφαλείας

Από τα παραπάνω μπορούμε να καταλήξουμε σε μια λίστα κανόνων που θα πρέπει να πληρούν οι κωδικοί για να μπορούν να θεωρηθούν ασφαλείς για χρήση:

i. Να χρησιμοποιούμε συνδυασμό όλων των διαθέσιμων χαρακτήρων (κεφαλαία και μικρά λατινικά, αριθμούς και ειδικούς χαρακτήρες).
ii. Ο κωδικός μας να είναι κατ' ελάχιστο μήκους 12 χαρακτήρων.
iii. Να μην χρησιμοποιούμε ευρέως γνωστούς «εύκολους» κωδικούς
iv. Να μην χρησιμοποιούμε κοινές λέξεις που μπορούν να βρεθούν σε λεξικό.
v. Να μην κάνουμε «εμφανείς» αντικαταστάσεις χαρακτήρων (π.χ. α με @, Α με 4 κ.λ.π.)
vi. Να μην χρησιμοποιούμε προσωπικά στοιχεία όπως η ημερομηνία γέννησης μας
vii. Να μην χρησιμοποιούμε τον ίδιο κωδικό για πρόσβαση σε πολλές διαφορετικές εφαρμογές
viii. Να μην γράφουμε ή αποθηκεύουμε τους κωδικούς μας κυρίως όταν δεν μπορούμε να διασφαλίσουμε την ιδιωτικότητα αυτών των αποθηκευτικών μέσων (π.χ. σε χαρτάκια που αφήνουμε εκτεθειμένα στο γραφείο μας).
ix. Να προσέχουμε το περιβάλλον κατά την χρήση των κωδικών για να μην επιτρέψουμε κάποιον να δει την εισαγωγή.
x. Να αλλάζουμε τους κωδικούς μας συχνά.
xi. Και τέλος να μην μοιραζόμαστε ή στέλνουμε τους κωδικούς μας με τρίτους και κυρίως μέσω ανασφαλών μέσων όπως το τηλέφωνο ή το ίντερνετ.

Αυτές οι αρχές πρέπει να διέπουν όλους τους κωδικούς που χρησιμοποιούμε καθημερινά για να μπορούμε να πούμε ότι λάβαμε τα απαραίτητα μέτρα για να προστατευθούμε και εμείς και οι πληροφορίες. Στην πορεία θα δείξουμε πως μπορούμε να κάνουμε ακριβώς αυτό που απαγορεύει ο τελευταίος κανόνας της παραπάνω λίστας, όταν και όπου χρειάζεται, σεβόμενοι τους υπόλοιπους κανόνες και με γνώμονα την ασφάλεια και την ιδιωτικότητα.

3. ΚΩΔΙΚΟΙ ΚΑΙ ΔΙΑΜΟΙΡΑΣΜΟΣ

Όπως είδαμε στο προηγούμενο κεφάλαιο οι κωδικοί ασφαλείας πρέπει να είναι προσωπικοί και μυστικοί. Υπάρχουν όμως περιπτώσεις κατά τις οποίες πρέπει να μοιραστεί ένας κωδικός μεταξύ δύο ή περισσοτέρων χρηστών. Αυτό μπορεί να οφείλεται είτε στην φύση του συστήματος στο οποίο θα χρησιμοποιηθεί ο κωδικός είτε ακόμη και στην φύση του ίδιου του κωδικού.

Όπως θα δούμε στην επόμενη ενότητα αναλυτικά υπάρχουν περιπτώσεις όπου θέλουμε να μοιραστούμε ένα κωδικό μεταξύ πολλών χρηστών. Οι κυριότεροι από αυτούς είναι

- Κλείδωμα και κοινή χρήση αρχείων
- Εισαγωγή χρηστών σε συστήματα που δεν υποστηρίζουν ή είναι δύσκολη η αλλαγή του κωδικού από τον τελικό χρήστη
- Κλείδωμα δοκιμαστικών εφαρμογών και ξεκλείδωμα όταν εκπληρωθούν κάποιες προϋποθέσεις

Είτε θέλουμε λοιπόν να στείλουμε ένα κλειδωμένο αρχείο σε κάποιον συνεργάτη μας, είτε θέλουμε να δώσουμε πρόσβαση σε ένα σύστημα που δεν επιτρέπει στον χρήστη να αλλάξει τον κωδικό του, η μη ύπαρξη ενός ασφαλούς τρόπου αποστολής μιας τόσο ευαίσθητης πληροφορίας. Το κυρίως πρόβλημα ξεκινάει στο ότι δεν υπάρχει κάποιο μέσο το οποίο να μπορούμε να πούμε ότι πληροί τις προϋποθέσεις ασφαλείας.

3.1 Μέσο διαμοιρασμού

Στην ψηφιακή εποχή η χρήση του ιντερνέτ και των τηλεπικοινωνιακών συστημάτων όπως τα κινητά μας τηλέφωνα, είναι τα μέσα που χρησιμοποιούμε για να επικοινωνήσουμε. Αυτά από την μία μας δίνουν την δυνατότητα να αλληλοεπιδρούμε με ανθρώπους σε μεγάλη απόσταση από εμάς, δεν μπορούν να μας εξασφαλίσουν όμως ασφάλεια στις επικοινωνίες μας κυρίως όταν η πληροφορία την οποία θέλουμε να μεταδώσουμε είναι τόσο κρίσιμη.

3.1.1 Ιντερνετ

Το ίντερνετ είναι ένα δίκτυο υπολογιστών που συνδέονται μεταξύ τους και ανταλλάσσουν πληροφορίες. Η αρχική του δημιουργία έγινε από τις ΗΠΑ κατά την διάρκεια του ψυχρού πολέμου προκειμένου να βοηθήσει τις στρατιωτικές δυνάμεις των ΗΠΑ να αναπτυχθούν τεχνολογικά και να δημιουργηθεί ένα δίκτυο επικοινωνίας το οποίο θα μπορούσε να επιβιώσει σε μια ενδεχόμενη πυρηνική επίθεση (Wikipedia, n.d.). Η ασφάλεια αυτών των πληροφοριών όμως δεν έπαιξε πολύ σημαντικό ρόλο στην δημιουργία του καθώς αυτό το δίκτυο υπολογιστών θα ήταν κλειστό και απροσπέλαστο από εξωτερικούς χρήστες. Με την χρήση αυτής της τεχνολογίας ως πλατφόρμα για το ίντερνετ όπως το ξέρουμε σήμερα, υπάρχουν πολλά κενά ασφαλείας τα οποία βασίζονται σε αυτή ακριβώς την εγγενώς ανασφαλή φύση του. Οι χρήστες του ίντερνετ είναι ευπαθείς σε επιθέσεις παραπλάνησης και οι τεχνικές για να γίνει αυτό αυξάνονται και σε πλήθος αλλά και σε πολυπλοκότητα.

Για να μοιράσουμε μια πληροφορία μέσω του ίντερνετ υπάρχουν συγκεκριμένοι τρόποι οι οποίοι είναι γνωστοί σε όλους μας. Ο πιο διαδεδομένος είναι τα ηλεκτρονικά μηνύματα (emails). Δεν θα πρέπει όμως σε καμία περίπτωση να στέλνουμε τόσο ευαίσθητες πληροφορίες μέσω email καθώς κρίνονται από τους πιο ανασφαλείς τρόπους επικοινωνίας. Οι ποιο κοινές ευπάθειες ασφαλείας για τα email (Lawson Kurtz, Former Senior Developer at Viget, 26-04-2016) είναι:

- Επιβεβαίωση του αποστολέα. Η διεύθυνση του αποστολέα δεν προσφέρει καμία ασφάλεια όσον αφορά την ταυτότητα αυτού καθώς οποιοσδήποτε πολύ εύκολα μπορεί να προσποιηθεί ένα χρήστη. Η πλήρης επιβεβαίωση του αποστολέα δεν είναι εφικτή παρά τα μέτρα που μπορεί να λάβει κάποιος για να διασφαλίσει ότι το μήνυμα έχει σταλεί από σωστό άτομο.
- Διασφάλιση του περιεχομένου. Από προεπιλογή, τα μηνύματα ηλεκτρονικού ταχυδρομείου αποστέλλονται με καθαρό κείμενο μέσω του διαδικτύου. Το SMTP δεν περιλαμβάνει κανένα μηχανισμό για την απόκρυψη ή την προστασία του περιεχομένου των μηνυμάτων σας από άλλους που έχουν πρόσβαση στους δρομολογητές και στα δίκτυα μέσω των οποίων ταξιδεύει το ηλεκτρονικό ταχυδρομείο. Αυτό σημαίνει ότι είναι δυνατό για τρίτους που δεν έχουν

εμπιστοσύνη, όχι μόνο να διαβάζουν το ηλεκτρονικό σας ταχυδρομείο, αλλά και να το αλλάζουν.

- Η αποθήκευση. Ακόμα κι αν λάβουμε τα απαραίτητα μέτρα για την σωστή προστασία του μηνύματος ηλεκτρονικού ταχυδρομείου κατά τη μεταφορά, η ασφάλεια μετάβασης από μόνη της δεν μπορεί να διασφαλίσει ότι τα μηνύματά δεν θα διαβαστούν ή θα παραβιαστούν πριν να σταλούν ή μετά την παράδοσή τους. Σε αυτές τις περιπτώσεις, οποιοσδήποτε με επαρκή πρόσβαση στο διακομιστή ηλεκτρονικού ταχυδρομείου που αποθηκεύει τα μηνύματα θα μπορούσε να θέσει σε κίνδυνο τα μηνύματα.

Όπως καταλαβαίνουμε λοιπόν το να στείλουμε έναν κωδικό μέσω email εμπεριέχει τόσους πολλούς κινδύνους που μπορεί να θεωρηθεί παραβίαση ασφαλείας και μόνο η καταγραφή του σε ένα μήνυμα πριν αυτό καν σταλεί στον αποδέκτη του.

Μια λύση στα παραπάνω έρχεται να δώσει το openPGP πρωτόκολλο (OpenPGP, n.d.). Αυτό βασίζεται στον διαμοιρασμό κλειδιών για την κρυπτογράφηση της πληροφορίας. Έχει εφαρμογή τόσο στην κρυπτογράφηση αρχείων όσο και στην αποστολή μηνυμάτων ηλεκτρονικού ταχυδρομείου. Ουσιαστικά είναι η ίδια τεχνική κρυπτογράφησης που χρησιμοποιούν και οι ιστοσελίδες μέσω του SSL/TLS πρωτοκόλλου χωρίς όμως την παρουσία CA καθώς ο κάθε χρήστης παράγει και αποθηκεύει τα προσωπικά κλειδιά (J. Callas, L. Donnerhacke, H. Finney, D. Shaw, R. Thayer, 11-2007). Θεωρείται από τους πιο ασφαλείς τρόπους ηλεκτρονικής επικοινωνίας σήμερα και προσφέρεται δωρεάν στους προγραμματιστές για να δημιουργήσουν εφαρμογές. Ακόμη και σε αυτό όμως έχουν βρεθεί ευπάθειες που μπορεί να οδηγήσουν στην αποκάλυψη του περιεχόμενου του μηνύματος και όταν αυτό περιέχει τον ίδιο τον κωδικό καταλαβαίνουμε ότι οι επιπτώσεις θα είναι καταστροφικές.

Όπως έχει δημοσιευθεί από μια ομάδα ερευνητών τον Μάιο του 2018 (Efail, n.d.) υπάρχουν δύο ευπάθειες σε αυτή την τεχνολογία τις οποίες μπορεί να εκμεταλλευθεί ένας επιτιθέμενος για να δει το περιεχόμενο του μηνύματος. Η επίθεση με «Άμεση διήθηση» εκμεταλλεύεται κενά ασφαλείας στις ίδιες τις εφαρμογές ηλεκτρονικού ταχυδρομείου. Ο εισβολέας δημιουργεί ένα νέο e-mail πολλαπλών τμημάτων με τρία μέρη του σώματος. Το πρώτο είναι ένα μέρος του σώματος HTML που περιέχει ουσιαστικά μια ετικέτα εικόνας αλλά το χαρακτηριστικό src αυτής της ετικέτας εικόνας ανοίγει με εισαγωγικά αλλά δεν

είναι κλειστό. Το δεύτερο μέρος του σώματος περιέχει το κρυπτογράφημα και το τρίτο είναι ξανά ένα μέρος του σώματος HTML που κλείνει το χαρακτηριστικό src του πρώτου μέρους του σώματος. Ο εισβολέας στέλνει τώρα αυτό το μήνυμα ηλεκτρονικού ταχυδρομείου στο θύμα. Ο πελάτης του θύματος αποκρυπτογραφεί το κρυπτογραφημένο δεύτερο τμήμα του σώματος και συρράπτει τα τρία μέρη του σώματος μαζί σε ένα email HTML. Το χαρακτηριστικό src της ετικέτας εικόνας καλύπτει πλέον όλες τις γραμμές του μηνύματος. Στη συνέχεια, το πρόγραμμα ηλεκτρονικού ταχυδρομείου κωδικοποιεί όλους τους μη εκτυπώσιμους χαρακτήρες (π.χ. το% 20 είναι κενό διάστημα) και ζητά μια εικόνα από τη συγκεκριμένη διεύθυνση URL. Δεδομένου ότι η διαδρομή της διεύθυνσης URL περιέχει το απλό κείμενο του κρυπτογραφημένου μηνύματος ηλεκτρονικού ταχυδρομείου, ο πελάτης ηλεκτρονικού ταχυδρομείου του θύματος στέλνει το απλό κείμενο στον εισβολέα.

Αν προσθέσουμε στα ανωτέρω ότι η χρήση τέτοιων τεχνολογιών απαιτεί, εκτός από κόστος στις περισσότερες των περιπτώσεων, αλλά και γνώσεις πέραν του μέσου όρου για εγκατάσταση, χρήση και διασφάλιση της λειτουργίας, καταλαβαίνουμε ότι δεν λύνουν επαρκώς το πρόβλημα του μέσου χρήστη για καθημερινή ασφαλή επικοινωνία.

3.1.2 Κινητή τηλεφωνία

Άλλος ένας τόπος επικοινωνίας, θα μπορούσε να πει κάποιος, είναι οι λύσεις που δίνει η κινητή τηλεφωνία. Η καθημερινότητά μας είναι πλέον συνυφασμένη με την χρήση κινητών τηλεφώνων και των εφαρμογών και επιλογών που μας δίνουν αυτά. Βέβαια ως επί το πλείστον αυτή η τεχνολογία αφορά την επικοινωνία μεταξύ δύο χρηστών και ο διαμοιρασμός πληροφορίας όπως ένας κωδικός μεταξύ πολλών θα μπορούσε να γίνει μια πολύ χρονοβόρα διαδικασία. Αρκεί να σκεφθούμε πόσο χρόνο θα χρειαζόταν κάποιος να τηλεφωνήσει σε μια ομάδα 10-20 χρηστών για να τους πει τον νέο κωδικό που πρέπει να χρησιμοποιήσουν. Επιπλέον και επειδή ο κωδικός αυτός είναι πολύπλοκος, η υπαγόρευσή του και μάλιστα τόσες πολλές φορές μπορεί να οδηγήσει σε ανθρώπινα λάθη και σε περεταίρω επιβράδυνση της διαδικασίας. Τέλος μια ακόμη προϋπόθεση είναι οι χρήστες που επικοινωνούν να είναι διαθέσιμοι ταυτόχρονα. Όταν θέλουμε να μοιραστούμε πληροφορία μεταξύ ατόμων που μπορεί να διαμένουν σε διαφορετικές ζώνες ώρας αυτό

μπορεί να δυσκολέψει πολύ την άμεση επικοινωνία και όταν το πλήθος των χρηστών αυξάνεται η πιθανότητα αδυναμίας επικοινωνίας ή λάθους πρέπει να θεωρηθεί δεδομένη. Βέβαια η χρήση των «έξυπνων» κινητών μας δίνει λύσεις σε όλα τα παραπάνω. Υπάρχουν πλέον πάρα πολλές εφαρμογές μαζικής επικοινωνίας και αποστολής μηνυμάτων και στις περισσότερες από αυτές έχουν ληφθεί μέτρα κρυπτογραφίας ούτως ώστε να διασφαλίζουν την μυστικότητα της συνομιλίας μας. Καμία όμως από αυτές δεν μπορεί να θεωρηθεί αρκετά ασφαλής για την αποστολή και λήψη τόσο ευαίσθητων πληροφοριών όπως οι μυστικοί κωδικοί.

Ένας τρόπος επικοινωνίας είναι τα SMS μηνύματα. Είναι από τα πρώτα και πιο βασικά μέσα αποστολής κειμένων. Δυστυχώς στις μέρες μας, όπως και με τις προφορικές συνομιλίες που πραγματοποιούνται από δίκτυα κινητής τηλεφωνίας, είναι αρκετά εύκολο για κάποιον να παραβιάσει την ιδιωτικότητα αυτών των συνομιλιών. Υπάρχουν τόσο εφαρμογές, που πρέπει να εγκατασταθούν στις συσκευές, αλλά και τεχνικές μίμησης συσκευών του δικτύου κινητής τηλεφωνίας που μπορούν με πολύ χαμηλό κόστος να επιτρέψουν σε κάποιον να έχει μη εξουσιοδοτημένη πρόσβαση στις επικοινωνίες μας. Αν αναλογιστούμε επίσης το κόστος που έχει αυτού του τύπου η επικοινωνία, καταλαβαίνουμε εύκολα ότι δεν αποτελεί ενδεδειγμένο τρόπο συχνού και πολλαπλού διαμοιρασμού κωδικών.

Στα «έξυπνα» κινητά τηλέφωνα όμως σήμερα υπάρχουν πολλές άλλες εφαρμογές που αποτελούν επιλογές επικοινωνίας. Όλες αυτές όμως χρησιμοποιούν την τεχνολογία του ίντερνετ για να μεταδώσουν την πληροφορία κληρονομώντας όλα τα προβλήματα που είδαμε στην προηγούμενη παράγραφο. Επιπλέον αποτελούν από τους πιο διαδεδομένους στόχους επιθέσεων και παραβιάσεων. Σύμφωνα με τον καθηγητή επιστήμης υπολογιστών του πανεπηστημίου Brigham Young University κ. Daniel Zappala για να έχουμε πραγματικά ασφαλή επικοινωνία με τις εφαρμογές όπως το Viber, το WhatsApp και το Facebook Messenger πρέπει να ακολουθήσουμε μια διαδικασία αυθεντικοποίησης χωρίς την οποία δεν μπορούμε να τις θεωρήσουμε πραγματικά ασφαλή τρόπο επικοινωνίας. (SAGE LAZZARO, 11-09-2017). Για τον μέσο χρήστη όμως, πολλώ δε μάλλον για χρήστες που δεν μπορούν να πιστοποιήσουν την ταυτότητα του συνομιλητή τους λόγο απόστασης μεταξύ τους ή ελλιπών στοιχείων, δεν μπορεί να θεωρηθεί, αυτός ο τρόπος επικοινωνίας, αρκετά ασφαλής.

Σε όλα αυτά έρχονται να προστεθούν και σκάνδαλα διαρροής πληροφοριών και μάλιστα από εταιρίες κολοσσούς, όπως το σκάνδαλο Facebook – Cambridge Analytica το 2018 (Wikipedia, n.d.). Φυσικά σε αυτή τη διαρροή δεν υπήρξε πρόβλημα με τους κωδικούς των χρηστών αλλά το πρόβλημα μπορεί να αγγίξει οποιαδήποτε πληροφορία έχει διαμοιραστεί ο χρήστης με τέτοιες πλατφόρμες. Μην ξεχνάμε ότι η Facebook έχει εξαγοράσει το 2014 μια από τις πιο δημοφιλείς εφαρμογές συνομιλιών για κινητές συσκευές, την WhatsApp, όπου ο ίδιος ο συνιδρυτής της κ. Brian Acton σε ομιλία του στο Stanford University τον Μάρτιο του 2019 προέτρεψε τους χρήστες να σταματήσουν να χρησιμοποιούν τις εφαρμογές μεγάλων εταιριών όπως η Google, η Apple και η Facebook καθώς δεν όπως είπε χαρακτηριστικά *«Θεωρώ ότι είναι αδύνατο να ελέγξεις το περιεχόμενο. Για να είμαι απόλυτα ειλικρινής, οι "ανοικτές πλατφόρμες" δεν μπορούν να αποφασίσουν τι είναι περιεχόμενο μίσους και τι όχι. Η Apple παλεύει να αποφασίσει αν μία εφαρμογή είναι καλή ή κακή. Η Google παλεύει να αποφασίσει αν μία ιστοσελίδα είναι καλή ή κακή. Αυτές οι εταιρείες δεν έχουν τους μηχανισμούς για να πάρουν αυτές τις αποφάσεις. Παρόλα αυτά τους δίνουμε δύναμη και αυτό είναι το χειρότερο. Αγοράζουμε τα προϊόντα τους, εγγραφόμαστε στις ιστοσελίδες τους. Διαγράψτε το Facebook!»* (Chris Elpidis, techgear, 31-03-2019)

3.2 Αποθήκευση της πληροφορίας

Εκτός από το πρόβλημα του μέσου επικοινωνίας της πληροφορίας υπάρχει και το θέμα της αποθήκευσης αυτής. Ακριβώς επειδή είναι είτε πολύ δύσκολο είναι πολύ κοστοβόρο να βρεθεί τρόπος αποστολής της πληροφορίας, έχουν αναπτυχθεί αρκετές εφαρμογές οι οποίες βοηθούν τους χρήστες να αποθηκεύουν τις ευαίσθητες πληροφορίες τους και να δίνουν πρόσβαση σε αυτές κατά το δοκούν. Εφαρμογές όπως το Pleasant Password Server (Pleasantsolutions, n.d.) και το 1Password Families (1password, n.d.) δίνουν την δυνατότητα να αποθηκεύουμε τους κωδικούς μας και να παρέχουμε πρόσβαση σε αυτούς στα άτομα που θέλουμε. Φυσικά όλα αυτά έρχονται με ένα οικονομικό κόστος. Το πρόβλημα σε αυτό είναι ότι εξαρτόμαστε από ένα σύστημα μιας τρίτης εταιρίας, χωρίς να έχουμε τα μέσα να ελέγξουμε τις δικλείδες ασφαλείας τους και κυρίως χωρίς να είμαστε σίγουροι ότι θα ενημερωθούμε έγκαιρα σε περίπτωση παραβίασης. Πολλοί χρήστες δεν θα

αισθανόταν άνετα με το να αποθηκεύουν τους επαγγελματικούς κωδικούς τους σε εγκαταστάσεις τρίτων κυρίως όσο οι περιπτώσεις διαρροής πληροφοριών αυξάνονται ακόμη και για παγκοσμίου φήμης εταιρίες.

Επιπλέον με αυτό τον τρόπο κανένας δεν ελέγχει τους ίδιους τους κωδικούς για το περιεχόμενό τους. Και όπως είδαμε στην παράγραφο 2.3 όταν οι κωδικοί παράγονται από τους ίδιους τους χρήστες, υπόκεινται σε λάθη και κενά ασφαλείας λόγο της χρήσης γνώριμων σε εμάς λέξεων. Μια τέτοια λύση λοιπόν δεν θα μπορούσε να ελέγξει τους κωδικούς με βάση το περιεχόμενό τους.

3.3 Η λύση της εφαρμογής Sharepass

Η εφαρμογή Sharepass, λαμβάνοντας υπ' όψιν όλα τα παραπάνω, έρχεται να δώσει μια λύση σε όλα αυτά τα συσσωρευμένα προβλήματα του διαμοιρασμού μιας τόσο ευαίσθητης πληροφορίας. Έχει σχεδιαστεί έτσι ώστε να αντιμετωπίζει όλα τα κενά που αφήνουν οι άλλες διαθέσιμες εφαρμογές αλλά κυρίως το πρόβλημα τόσο της κυκλοφορίας της πληροφορίας όσο και της παραγωγής και αποθήκευσης αυτής σε ένα ανασφαλές περιβάλλον. Έτσι τα χαρακτηριστικά της εφαρμογής Sharepass είναι:

- Δεν χρειάζεται να καταγραφεί και να αποσταλεί η ευαίσθητη πληροφορία με κανένα μέσο γιατί παράγεται από τον τελικό χρήστη.
- Δεν αποθηκεύονται οι κωδικοί σε κανένα μέσο τοπικό ή απομακρυσμένο αλλά παράγονται την στιγμή που χρειάζονται
- Οι κωδικοί δημιουργούνται από την εφαρμογή με πραγματικά τυχαίο και μηχανικό τρόπο κάνοντας τους λιγότερο ευάλωτους σε «επιθέσεις λεξικών»
- Υπάρχει μεγάλη ευελιξία στην χρήση της εφαρμογής με δυνατότητα γραμμής εντολών και πολλαπλών εγκαταστάσεων για διαφορετικές ομάδες χρηστών
- Είναι εύκολη η αλλαγή της εγκατάστασης της εφαρμογής σε περιπτώσεις παραβίασης ασφαλείας

Τέλος ο τρόπος παρουσίασης των αποτελεσμάτων βοηθάει να αποφύγουμε άλλο ένα τύπο επίθεσης που είναι πολύ κοινός στα πληροφοριακά συστήματα. Πολλές φορές υπάρχουν κακόβουλα λογισμικά που εγκαθίστανται το σύστημά μας και καταγράφουν την εισαγωγή χαρακτήρων (keylogger). Με την χρήση της εφαρμογής Sharepass, παρουσιάζεται στον χρήστη ο μυστικός κωδικός και του δίνεται αρκετός χρόνος έτσι ώστε να μπορέσει να τον αντιγράψει αντί να τον πληκτρολογήσει. Με αυτό τον τρόπο, πιθανά προγράμματα keyloggers το μόνο που θα μπορέσουν να καταγράψουν είναι το τετραψήφιο κλειδί που θα εισάγει ο χρήστης, προστατεύοντας έτσι τον μυστικό κωδικό καθώς δεν είναι δυνατή η παραγωγή του χωρίς την χρήση της εφαρμογής.

Στο επόμενο κεφάλαιο θα δούμε κάποιες πιθανές χρήσεις της εφαρμογής και πως αυτή μπορεί να βοηθήσει και να δώσει λύσεις στα προβλήματα και στις ανάγκες των χρηστών.

4. ΧΡΗΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Η εφαρμογή Sharepass σχεδιάστηκε για να δώσει λύσεις σε προβλήματα παραγωγής και διαμοιρασμού κωδικών. Έχει δοθεί έμφαση τόσο στην εύχρηστη λειτουργεία της από τους τελικούς χρήστες όσο και σε ένα πιο αυτόματο τρόπο για συμβατότητα με άλλες εφαρμογές και προγράμματα. Σε αυτό το κεφάλαιο θα περιγράψουμε κάποιες πιθανές χρήσεις οι οποίες όμως δεν πρέπει να θεωρηθούν και οι μόνες. Οι παραλλαγές τόσο στην λειτουργία της όσο και στον τρόπο παραγωγής των, απαραίτητων, τετραψήφιων κλειδιών εξαρτώνται μόνο από την φαντασία των προγραμματιστών και τις ανάγκες τις οποίες καλείται να καλύψει σε κάθε δεδομένη εφαρμογή της.

4.1 Κλείδωμα αρχείων

Μία πιθανή χρήση κωδικού που απαιτεί τον διαμοιρασμό του με άλλους χρήστες είναι το κλείδωμα αρχείων και φακέλων για την ασφαλή αποστολή και αποθήκευσή τους. Αφού δημιουργήσουμε λοιπόν τα αρχεία μας, επιλέγουμε μεταξύ των προγραμμάτων που μας δίνουν την δυνατότητα να τα κλειδώσουμε με την χρήση ενός κωδικού όπως το WinRAR (Rarlab, n.d.), το WinZip ή οποιοδήποτε πρόγραμμα της αρεσκείας μας. Έτσι εξασφαλίζουμε ότι τα αρχεία μας δεν θα είναι προσπελάσιμα από μη εξουσιοδοτημένους χρήστες.

Τι γίνεται όμως στην περίπτωση που θέλουμε να στείλουμε αυτά τα αρχεία σε κάποιον τρίτο και να του επιτρέψουμε την πρόσβαση σε αυτά; Το να «στείλουμε» τα αρχεία όσο είναι κλειδωμένα είναι μια ασφαλής διαδικασία με οποιονδήποτε πρόσφορο τρόπο και αν επιλέξουμε (email, κοινόχρηστοι φάκελοι κ.λ.π.). Ο κωδικός όμως για να ξεκλειδώσουμε την πρόσβαση δεν πρέπει και δεν μπορεί να σταλεί ασφαλώς με τους παραπάνω τρόπους. Καταλαβαίνουμε ότι αν οι τρόποι αποστολής μπορούσαν να κριθούν ασφαλείς δεν θα υπήρχε η ανάγκη εξ αρχής να κλειδώσουμε τα αρχεία. Έτσι χρειαζόμαστε ένα τρόπο να γνωρίσουμε στους άλλους χρήστες τον κωδικό χωρίς όμως να διακυβεύσουμε την μυστικότητά του. Η εφαρμογή Sharepass μπορεί να δώσει λύση στην συγκεκριμένη

περίπτωση. Μπορούμε να έχουμε μια ίδια εγκατάσταση όλοι οι χρήστες και να στέλνουμε μαζί με το κλειδωμένο αρχείο και το τετραψήφιο κλειδί έτσι ώστε να παράξουν όλοι τον κωδικό ξεκλειδώματος. Έτσι η πληροφορία που θα μοιράζεται δεν είναι κρίσιμη για την εξασφάλιση της μυστικότητας του κωδικού.

Σε αυτή την περίπτωση η εγκατάσταση του αρχείου μπορεί να γίνει εφ' άπαξ από ένα χρήστη και να μοιραστεί στους υπόλοιπους ή να βρεθεί ένας ασφαλής τρόπος να γνωρίζουμε την μυστική φράση που θα χρησιμοποιήσουμε για την εγκατάσταση. Όποιος τρόπος και να επιλεγεί η εφαρμογή Sharepass μας δίνει μια επιπλέον δυνατότητα που μπορεί να φανεί πολύ χρήσιμη. Στην πορεία της χρήσης της εφαρμογής υπάρχει η περίπτωση ένας οι περισσότεροι χρήστες χρειαστεί να πάψει να έχει πρόσβαση στα αρχεία. Αυτό μπορεί να συμβεί για παράδειγμα αν κάνουμε χρήση της εφαρμογής για διαμοιρασμό επαγγελματικών εγγράφων και πληροφοριών και ένας από τους υπαλλήλους πάψει να εργάζεται για την εταιρία μας. Σε αυτή την περίπτωση και για να διασφαλίσουμε ότι δεν υπάρχει μη εξουσιοδοτημένο αντίγραφο της εφαρμογής, μπορούμε να αλλάξουμε την εγκατάσταση οι υπόλοιποι χρήστες και να φτιάξουμε μια νέα «έκδοση» της εφαρμογής η οποία θα μας δίνει τελείως διαφορετικά αποτελέσματα.

4.2 Πρόσβαση σε απομακρυσμένα συστήματα.

Τα περισσότερα συστήματα που υποστηρίζουν πρόσβαση χρηστών (user access) δίνουν την επιλογή στους χρήστες να αλλάξουν τον κωδικό πρόσβασης. Έτσι οι διαχειριστές του συστήματος δημιουργούν τους χρήστες και βάζουν ένα προσωρινό κωδικό ο οποίος μετά την χρήση του πρέπει να αλλαχθεί από τον τελικό χρήστη. Υπάρχουν όμως συστήματα που δεν δίνουν αυτή την δυνατότητα. Ένα τέτοιο σύστημα είναι για παράδειγμα η απομακρυσμένη πρόσβαση, μέσω VPN, των firewall της εταιρίας Cyberoam (Cyberoam, n.d.). Σε αυτά τα συστήματα η αλλαγή του κωδικού πρόσβασης από τον χρήστη δεν είναι τόσο εύκολη καθώς για να γίνει αυτό θα πρέπει να γνωστοποιηθεί στον χρήστη ευαίσθητη πληροφορία του συστήματος όπως η ηλεκτρονική διεύθυνση της κονσόλας διαχείρισης του firewall. Σε πολλές περιπτώσεις αυτό δεν είναι επιθυμητό, παρά τους περιορισμούς στην πρόσβαση που μπορούν να ρυθμιστούν στο σύστημα και η

διαδικασία απαιτεί γνώσεις πέραν του μέσου χρήστη. Επιπλέον όταν ο αριθμός των χρηστών μεγαλώνει, αυξάνεται και ο κίνδυνος διαρροής ευαίσθητης πληροφορίας του συστήματος ή κάποιου κωδικού που σε λάθος χέρια μπορεί να επιφέρει ανεπιθύμητες συνέπειες για την ασφάλεια του συστήματος.

Σε τέτοιες περιπτώσεις η εφαρμογή Sharepass δίνει την δυνατότητα στους διαχειριστές να δημιουργούν χρήστες και κωδικούς και να μοιράζονται την πληροφορία με ασφάλεια. Η παραγωγή των κωδικών μπορεί να γίνει από τους διαχειριστές και με μια απλή κοινοποίηση του τετραψήφιου κωδικού, να μπορεί ο χρήστης να εισέλθει στο σύστημα χωρίς να του γνωστοποιηθούν ευαίσθητες πληροφορίες και κυρίως χωρίς να διακυβεύεται η ακεραιότητα του ίδιου του κωδικού. Επιπλέον βοηθάει στην τακτική αλλαγή των κωδικών, που όπως είδαμε στο κεφάλαιο 2 είναι από τις ποιο κρίσιμες αρχές ασφαλείας ενός συστήματος. Με την χρήση της ίδιας εγκατάστασης της εφαρμογής λοιπόν, το μόνο που χρειάζεται να κοινοποιηθεί στους χρήστες είναι με ποιο κλειδί, ο καθένας από αυτούς, θα μπορεί να παράξει τον κωδικό του.

4.3 Αυτόματο κλείδωμα εφαρμογών

Έχει δοθεί ιδιαίτερη σημασία στην χρήση της εφαρμογής με αυτόματο τρόπο. Αυτό έγινε γιατί θέλαμε να μπορεί να «κληθεί» από άλλες εφαρμογές και διαδικασίες, ούτως ώστε να γίνεται αυτόματο κλείδωμα αρχείων και εφαρμογών και μόνο ο διαχειριστής να μπορεί να τις ξεκλειδώσει.

Όπως θα δούμε και στο κεφάλαιο 6.2 αρκεί μια κλήση της εφαρμογής από την γραμμή εντολών του λειτουργικού μας συστήματος για να μας επιστραφεί ο μυστικός κωδικός και ο τετραψήφιος κωδικός επιβεβαίωσης. Έτσι μπορεί να αναπτυχθεί μια εφαρμογή η οποία θα καλεί την Sharepass, θα αποθηκεύει τον κωδικό και θα τον χρησιμοποιεί για να «κλειδώσει» κάποια αρχεία ή εφαρμογές αυτόματα.

Αυτό μπορεί για παράδειγμα να έχει χρήση στις περιπτώσεις που θέλουμε να δώσουμε μια δοκιμαστική «έκδοση» μιας εφαρμογής η οποία θα κλειδώσει μετά από κάποιο χρονικό διάστημα. Ο κώδικάς μας το μόνο που χρειάζεται να κάνει είναι να παράξει ένα, τυχαίο ή μη, τετραψήφιο αριθμό και να χρησιμοποιήσει το Sharepass για να δημιουργήσει τον

κωδικό και να κλειδώσει με ασφάλεια. Σε περίπτωση που ο δημιουργός της εφαρμογής θέλει να την «ξεκλειδώσει» το μόνο που χρειάζεται είναι να έχει τον τετραψήφιο αριθμό που χρησιμοποιήθηκε και της ίδια εγκατάσταση της εφαρμογής Sharepass. Επιπλέον η όλη διαδικασία μπορεί να επιβεβαιωθεί ως προς την ορθότητά της με τον κωδικό επιβεβαίωσης.

Για την δημιουργία του τετραψήφιου κλειδιού, εκτός από την τελείως τυχαία επιλογή του με χρήση των αντίστοιχων εντολών, θα μπορούσε για παράδειγμα να γίνει με χρήση της ημερομηνίας κατά την οποία έγινε το κλείδωμα. Έτσι ο δημιουργός της εφαρμογής χρειάζεται να πληροφορηθεί μόνο την ημέρα που κλείδωσε η εφαρμογή του και να παράξει τον μυστικό κωδικό επιβεβαιώνοντας την ορθότητά του με την χρήση του κωδικού επιβεβαίωσης. Φυσικά η διαδικασία παραγωγής του κλειδιού δεν περιέχει κάποια δέσμευση και είναι στην διακριτική ευχέρεια του εκάστοτε προγραμματιστή.

Όλα τα παραπάνω μπορούν να φανούν χρήσιμα καθώς ο δημιουργός της εκάστοτε εφαρμογής δεν χρειάζεται να θυμάται ή να αποθηκεύσει τον κωδικό που κλειδώνει την εφαρμογή του. Επιπλέον το κάθε αντίγραφο της δοκιμαστικής λειτουργείας που θα διαθέτει, θα κλειδώνεται με διαφορετικό κωδικό. Με αυτό τον τρόπο δεν υπάρχει ο κίνδυνος να διαρρεύσει, μεταξύ των ατόμων που έλαβαν τις δοκιμαστικές εκδώσεις, ο κωδικός και να μπορέσει κάποιος από αυτούς να ξεκλειδώσει την εφαρμογή χωρίς να πληροί τα κριτήρια για κάτι τέτοιο.

4.4 Διαχείριση κωδικών

Μια τελευταία πιθανή εφαρμογή της Sharepass είναι σαν διαχειριστής κωδικών. Όπως είπαμε και στη παράγραφο 1.1 ο όγκος των κωδικών που πρέπει να αποστηθίσουμε γίνεται όλο και πιο μεγάλος. Επιπλέον είδαμε στην παράγραφο 2.3 όταν οι κωδικοί δημιουργούνται από τους χρήστες απευθείας, περιλαμβάνουν λέξεις και αντικαταστάσεις οι οποίες κάνουν τους κωδικούς πιο εύκολους στο να τους θυμόμαστε αλλά ταυτόχρονα και πιο ευπαθείς σε επιθέσεις.

Με την εφαρμογή Sharepass μπορούμε να δημιουργήσουμε κωδικούς οι οποίοι δεν περιέχουν λέξεις που υπάρχουν σε λεξικά και ταυτόχρονα δεν χρειάζεται να τους

θυμόμαστε. Το μόνο που πρέπει να θυμόμαστε είναι ένα, πολύ πιο εύκολο στην αποστήθιση, τετραψήφιο κλειδί για κάθε κωδικό που θέλουμε να χρησιμοποιήσουμε.

Υπάρχουν πολλές εφαρμογές διαχείρισης κωδικών που, με την χρήση κρυπτογραφίας, αποθηκεύουν τους κωδικούς μας και τους κρατάνε προστατευμένους. Η διαφορά της Sharepass είναι ότι οι κωδικοί δεν αποθηκεύονται πουθενά αλλά κάθε φορά που θέλουμε να τους χρησιμοποιήσουμε, παράγονται από την αρχή. Έτσι γίνεται πολύ πιο δύσκολο για κάποιον να μπορέσει να «σπάσει» το κλείδωμα και να έχει πρόσβαση στους κωδικούς μας. Τέλος έτσι δεν υπάρχει η εξάρτηση από ένα συγκεκριμένο αποθηκευμένο αρχείο. Ακόμη και να χάσουμε ή να σβήσουμε την εφαρμογή, αρκεί μια νέα εγκατάσταση με την ίδια μυστική φράση για να έχουμε και πάλι πρόσβαση στην ίδια πληροφορία. Επιπλέον προστατευόμαστε ακόμη περισσότερο από κακόβουλα αρχεία τύπου keylogger. Όλες οι εφαρμογές που χειρίζονται τους κωδικούς μας, είναι προστατευμένες και οι ίδιες από ένα κωδικό. Αν υπάρχει διαρροή αυτού του κωδικού, όλοι οι υπόλοιποι πρέπει να θεωρηθούν ανασφαλείς. Με την εφαρμογή Sharepass δεν εισάγεται και ως εκ τούτου δεν μπορεί να καταγραφεί κανένας κωδικός ασφαλείας. Ακόμη και ο παραχθέν κωδικός μπορεί να χρησιμοποιηθεί με αντιγραφή κάνοντας το σύστημά μας πραγματικά ασφαλισμένο έναντι απειλών καταγραφής του πληκτρολογίου.

Όπως γίνεται εύκολα κατανοητό οι περιπτώσεις χρήσης της εφαρμογής Sharepass δεν περιορίζονται στα παραπάνω. Οτιδήποτε απαιτεί την παραγωγή ενός ασφαλούς κωδικού μπορεί να δημιουργήσει μια νέα πιθανή χρήση της εφαρμογής. Οι τρόποι με τους οποίους μπορούμε να χρησιμοποιήσουμε την εφαρμογή και η απλότητα στην χρήση την κάνουν κατάλληλη για κάθε πιθανή ανάγκη παραγωγής και διαμοιρασμού με ασφάλεια ενός ή περισσοτέρων κωδικών κλειδώματος.

5. ΠΛΑΤΦΟΡΜΑ ΥΛΟΠΟΙΗΣΗΣ ΚΑΙ ΕΡΓΑΛΕΙΑ

Για την υλοποίηση της λύσης έχουν χρησιμοποιηθεί ορισμένα εργαλεία αλλά και τεχνικές προγραμματισμού σε περιβάλλον windows. Σε αυτή την ενότητα θα αναφερθούμε γενικά στα εργαλεία που χρησιμοποιήθηκαν

5.1 Batch file

Όλοι οι κώδικες έχουν γραφτεί σε batch files. Το batch file ένα script που χρησιμοποιεί τις εντολές του DOS περιβάλλοντος. Είναι ουσιαστικά ένα κείμενο που περιλαμβάνει μια σειρά εντολών που μπορεί ο μεταγλωττιστής των windows (COMMAND.COM) να εκτελέσει (Wikipedia, n.d.).

Τα batch files προέρχονται από την εποχή που τα windows αποτελούσαν μια προθήκη γραφικού περιβάλλοντος του DOS. Για να εκτελεστεί ένα batch file αρκούσε από το DOS περιβάλλον να πληκτρολογήσουμε το όνομα του αρχείου.

Στις πρώτες εκδώσεις των windows έπρεπε να χρησιμοποιήσει κάποιος το MS-DOS των windows για να μπορέσει να εκτελέσει ένα batch file. Από τα windows 3.1x και μετά προστέθηκε ο μεταγλωττιστής COMMAND.COM για να μπορούν να εκτελούνται τέτοια αρχεία και από το γραφικό περιβάλλον (GUI). Από τα windows NT και μετά, όταν και έπαψαν να εξαρτώνται από το DOS περιβάλλον, έχει προστεθεί ο μεταγλωττιστής CMD.EXE (βλ. 5.2 Cmd) ακριβώς για να εκτελούνται αρχεία τέτοιου τύπου.

Η δημιουργία αυτών των αρχείων είναι μια σχετικά απλή διαδικασία. Το μόνο που χρειάζεται είναι οποιοσδήποτε κειμενογράφος και φυσικά η γνώση τόσο των εντολών όσο και του τρόπου σύνταξης αυτών καθώς υπάρχουν διαφορές, κυρίως στην χρήση των ειδικών χαρακτήρων, στον τρόπο σύνταξης μιας εντολής σε batch file από την χρήση αυτής απευθείας σε γραμμή εντολών (π.χ. ο ειδικός χαρακτήρας % που καλεί μια μεταβλητή πρέπει να γραφτεί δύο φορές %% σε ένα batch file). Τέλος αρκεί να αποθηκευτεί το

κείμενο με επέκταση .BAT ή . CMD και έχει δημιουργηθεί ένα εκτελέσιμο αρχείο για το περιβάλλον των windows.

5.2 Cmd

Command Prompt ή cmd (από το όνομα του εκτελέσιμου αρχείου του cmd.exe) είναι ο μεταγλωττιστής του λειτουργικού συστήματος των windows NT και μεταγενέστερα. Είναι ουσιαστικά η γραμμή εντολών του DOS αλλά σε γραφικό περιβάλλον. Υπάρχουν όμως ορισμένες διαφορές από το MS-DOS Prompt (Wikipedia, n.d.).

- Δίνει πιο εμπεριστατωμένα μηνύματα σφαλμάτων από το απλό «Bad command or file name" του DOS.
- Υποστηρίζει την χρήση των πλήκτρων κατεύθυνσης για την περιήγηση στο ιστορικό των εντολών
- Υποστηρίζει την συμπλήρωση των εντολών με την χρήση του Tab
- Χρησιμοποιεί τον χαρακτήρα ^ σαν παράκαμψη ούτως ώστε να μπορεί να γίνει χρήση των ειδικών χαρακτήρων με την κυριολεκτική τους έννοια και όχι σαν εντολή του συστήματος (π.χ. οι χαρακτήρες "<", ">" και "|" έχουν συγκεκριμένες έννοιες για την γραμμή εντολών και αν θέλουμε να τους χρησιμοποιήσουμε σαν απλούς χαρακτήρες θα πρέπει να βάλουμε πριν από αυτούς στον κώδικά μας το "^")
- Υποστηρίζει το Delayed Variable Expansion το οποίο επιτρέπει στον κώδικα να χρησιμοποιήσει τις μεταβλητές με λιγότερο αυστηρό τρόπο. Επιτρέπει δηλαδή στον κώδικα να απεμπλακεί από τον περιοριστικό επιτακτικό προγραμματισμό.

Για όλα όσα θα δούμε παρακάτω, όταν και θα αναλύσουμε τους κώδικες σε βάθος, χρησιμοποιήθηκαν μόλις 12 διαφορετικές εντολές του cmd. Ακολουθεί πίνακας με τις εντολές που έχουν χρησιμοποιηθεί καθώς και περιγραφή αυτών:

ΠΙΝΑΚΑΣ 4 *Λίστα Εντολών*

Attrib	Αλλάζει τα χαρακτηριστικά ενός ή περισσοτέρων αρχείων.
Cls	Καθαρίζει την οθόνη από οποιαδήποτε μηνύματα είναι εμφανισμένα.
Del	Σβήνει ένα ή περισσότερα αρχεία.
Echo	Εμφανίζει στην οθόνη κάποιο μήνυμα. Εναλλακτικά με την χρήση των > και >> προσθέτει το μήνυμα σε ένα αρχείο κειμένου.
For	Είναι η εντολή επανάληψης. Μπορεί επίσης να διαβάσει ένα αρχείο κειμένου ανά γραμμή και να επαναλάβει ένα κομμάτι αλγόριθμου για κάθε γραμμή ξεχωριστά.
Goto	Ανακατευθύνει την ροή του κώδικα σε κάποιο προκαθορισμένο σημείο.
If	Είναι η εντολή ελέγχου. Εκτελεί ένα αλγόριθμο αν η συνθήκη ισχύει.
Rd	Σβήνει ένα φάκελο και όλα τα αρχεία που περιέχει.
Ren	Μετονομάζει αρχεία ή φακέλους.
Set	Είναι η εντολή η οποία δίνει τιμές στις μεταβλητές του κώδικα.
Setlocal	Αλλάζει τα χαρακτηριστικά του περιβάλλοντος στο οποίο εκτελείται ο αλγόριθμός.
Timeout	Σταματάει την εκτέλεση του κώδικα για συγκεκριμένο χρονικό διάστημα ή μέχρι να πατηθεί κάποιο πλήκτρο.

5.3 Notepad++

Όπως εξηγήσαμε και παραπάνω, υπήρξε η ανάγκη χρήσης ενός κειμενογράφου για την αποτύπωση των εντολών και την αποθήκευσή τους στα batch αρχεία. Γι’ αυτό το σκοπό επιλέχθηκε το πρόγραμμα Notepad++ (Don Ho, n.d.).

Το notepad++ είναι ένας δωρεάν κειμενογράφος ο οποίος είναι πολύ φιλικός στην δημιουργία κώδικα. Είναι βασισμένος στο εργαλείο τροποποίησης κώδικα Scintilla (A free source code editing component, 07-03-2019) και διανέμεται με την GPL άδεια του προγράμματος GNU (Gnu Operating System, n.d.). Έχει σχεδιαστεί για να βοηθάει τους προγραμματιστές στην δημιουργία κώδικα αποτελούμενο από πολλές γραμμές και κάποια από τα προτερήματα του είναι:

- Η αυτόματη τροποποίηση της εμφάνισης των λέξεων όταν αυτές αναγνωριστούν σαν εντολές
- Αυτόματη συμπλήρωση των λέξεων
- Υποστήριξη καρτελών για ταυτόχρονη εργασία σε πολλά αρχεία κώδικα
- Δεν εμποδίζει την αλλαγή των αρχείων από άλλα προγράμματα όταν αυτά είναι ανοιγμένα στο notepad++
- Η πλήρης υποστήριξη των εντολών του cmd και των batch files.
- Η υποστήριξη regular expression για την αναζήτηση και αντικατάσταση μέρους των κειμένων

5.4 FCIV

Κατά την υλοποίηση του προγράμματος, υπήρξε η ανάγκη να χρησιμοποιηθεί μια hash function. Οι hash functions είναι διεργασίες (προγράμματα) οι οποίες παίρνουν οποιαδήποτε εισαγωγή δεδομένων και παράγουν ένα συγκεκριμένου μεγέθους αποτέλεσμα διαφορετικό για κάθε δεδομένο που εισάγεται. Το σημαντικό είναι ότι η διαδικασία αυτή είναι μη αναστρέψιμη και ότι παράγει κάθε φορά διαφορετικό αποτέλεσμα για διαφορετικές εισαγωγές (Wikipedia, n.d.). Μπορεί κάποιος, ξέροντας ποια hash function έχει χρησιμοποιηθεί, να βρει το αποτέλεσμα για κάποιο δεδομένο αλλά δεν μπορεί κανένας να βρει το δεδομένο ξέροντας το αποτέλεσμα της hash function. Σε περίπτωση που βρεθεί τρόπος να αναστραφεί η διαδικασία, η hash function θεωρείται ότι έχει «σπάσει». Γι' αυτό το λόγο χρησιμοποιούνται στην κρυπτογραφία.

Για τον υπολογισμό λοιπόν της hash function χρησιμοποιήθηκε το πρόγραμμα της Microsoft FCIV (Microsoft Corp, 2018). Το FCIV είναι ένα δωρεάν πρόγραμμα με δυνατότητα γραμμής εντολών, το οποίο μπορεί να υπολογίσει το αποτέλεσμα δυο, εκ των πιο γνωστών, hash function για ένα αρχείο, της md5 και της sha1. Φυσικά ανάμεσα στις δύο αυτές επιλέχθηκε η sha1 καθώς, παρόλο που η πρώτη είναι πολύ πιο γρήγορη, χαρακτηρίζεται πολύ πιο ασφαλής και δεν έχουν καταφέρει να αναστρέψουν την λειτουργία της (δεν έχει «σπάσει» όπως η md5). Έχοντας λοιπόν περισσότερο ενδιαφέρον

στην ασφάλεια από ότι στην ταχύτητα, επιλέχθηκε η hash function sha1 για τον υπολογισμό των μεταβλητών μας.

Η sha1 (Secure Hash Algorithm) είναι ένας αλγόριθμός ο οποίος παίρνει οποιουδήποτε μεγέθους δεδομένα και επιστρέφει ένα 160-bit (20-byte) αποτέλεσμα. Το αποτέλεσμα είναι σε δεκαεξαδική μορφή, αποτελείται δηλαδή από νούμερα και γράμματα από το a έως το f (Wikipedia, n.d.). Αν και σε καμία περίπτωση δεν μπορούμε να πούμε ότι είναι η πιο ασφαλής hash function, αποτελεί μία πολύ καλή λύση και χρησιμοποιείται στην κρυπτογραφία μέχρι και σήμερα παρότι το 2017 η CWI Amstradam με την Google ανακοίνωσαν ότι βρέθηκαν δύο διαφορετικά PDF αρχεία που μας δίνουν το ίδιο αποτέλεσμα. Δεν έχει γίνει γνωστό όμως μέχρι σήμερα να έχει καταφέρει κάποιος να αντιστρέψει την διαδικασία της sha1. Η παραγωγή ίδιων αποτελεσμάτων για διαφορετικά δεδομένα λοιπόν είναι το μεγαλύτερο πρόβλημα που έχει η sha1 και αυτό ήταν κάτι το οποίο έπρεπε να αντιμετωπίσουμε στο πρόγραμμά μας για να έχουμε το δυνατών μεγαλύτερη ασφάλεια.

5.5 Bat_To_Exe_Converter και Maskedinput

Όπως γίνεται εύκολα κατανοητό, ένα πρόγραμμα που αφορά θέματα ασφάλειας δεν θα μπορούσε να μην περιλαμβάνει τις βασικές λειτουργίες ασφαλείας και το ίδιο. Έτσι θα πρέπει κατά την εισαγωγή της μυστικής φράσης, για την εγκατάσταση της εφαρμογής, τα ψηφία που εισάγονται θα πρέπει να καλύπτονται για να εξασφαλιστεί η ακεραιότητά τους. Με το σκεπτικό λοιπόν ότι μια ενδεχόμενη διαρροή του κλειδιού παραγωγής του εκάστοτε κωδικού δεν θα μπορεί να θεωρηθεί παράβαση ασφαλείας καθώς δεν θα μπορεί να οδηγήσει στην παραγωγή του σωστού κωδικού, η μυστική φράση είναι και το μοναδικό σημείο αποτυχίας των μέτρων ασφαλείας.

Τέλος υπήρξε η ανάγκη μετατροπής των batch αρχείων σε εκτελέσιμα (.exe). Παρόλο που το πρόγραμμα σχεδιάστηκε να μπορεί να διατεθεί δωρεάν, η προστασία των αλγόριθμων έπρεπε να γίνει για να «προστατευθεί» ο κώδικας από κάποιον που θα ήθελε να αντιστρέψει τις διαδικασίες, «σπάζοντας» τις δικλείδες ασφαλείας και καθιστώντας τους αλγόριθμους ανασφαλείς. Η ευκολία και η αμεσότητα των batch files έρχεται με ένα

τίμημα. Όλοι οι κειμενογράφοι μπορούν να ανοίξουν το αρχείο καθιστώντας των πηγαίο κώδικα προσβάσιμο σε οποιονδήποτε. Είναι λοιπόν πολύ απλό για κάποιο που ξέρει να αναγνωρίζει τον κώδικα, να ανοίξει το αρχείο και να προσπαθήσει να κατανοήσει τις διεργασίες που γίνονται σε αυτόν. Μετά θα μπορεί, όχι μόνο να αναπαράξει τους κώδικες αλλά και να αντιστρέψει τις διαδικασίες με αποτέλεσμα να μην μπορούμε να θεωρήσουμε τους αλγόριθμους αυτούς ασφαλείς. Γι' αυτούς τους λόγους το πρόγραμμα μπορεί να είναι δωρεάν (freeware) αλλά όχι και ανοιχτού κώδικα (open source).

Για να γίνει επιτευχθούν τα ανωτέρω επιλέχθηκαν δυο δωρεάν προγράμματα. Το Bat_To_Exe_Converter και το Maskedinput. Όπως μαρτυρούν τα ίδια τα ονόματά τους, είναι προγράμματα που δημιουργήθηκαν από τον Fatih Kodak (f2k0 Software, n.d.) και έχουν ως στόχο την μετατροπή των batch αρχείων σε exe και την κάλυψη (mask) των ψηφίων κατά την εισαγωγή. Το πρώτο αρχείο επιλέχθηκε καθώς έχει πολλές δυνατότητες όπως:

- την εισαγωγή βοηθητικών φακέλων στο .exe αρχείο,
- την δημιουργία x64 εκτελέσιμων
- την εισαγωγή εικονιδίου
- την επιλογή δημιουργίας αρχείου χωρίς περιβάλλον παραθύρου (εκτέλεση στο background)

αλλά η δυνατότητα για την οποία επιλέχθηκαν και τα δύο έναντι κάποιων αντίστοιχων είναι γιατί είναι εκτελέσιμα μέσω γραμμής εντολών. Έτσι μπορούμε να τα χρησιμοποιήσουμε σε ένα batch αρχείο.

Όλα τα αρχεία και προγράμματα που χρησιμοποιήθηκαν διατίθενται δωρεάν για χρήση και είναι όλα σχεδιασμένα να εκτελούνται σε περιβάλλον του λειτουργικού συστήματος windows. Και τα δύο αυτά χαρακτηριστικά ήταν πολύ σημαντικά κατά την επιλογή των εργαλείων καθώς σκοπός αυτής της προσπάθειας ήταν αφ' ενός ο χρήστης να μπορεί να αποκτήσει το πρόγραμμα χωρίς να κληθεί να καλύψει κάποιο κόστος (freeware) και αφ' ετέρου να είναι συμβατό και να εκτελείται στο λειτουργικό σύστημα το οποίο είναι ποιο διαδεδομένο αυτή τη στιγμή κυρίως στους μη εξειδικευμένους χρήστες.

6. Η ΥΛΟΠΟΙΗΣΗ

Η υλοποίηση της ιδέας περιλαμβάνει δύο μέρη. Στο πρώτο μέρος υπάρχει το αρχείο εγκατάστασης το οποίο δέχεται μια μυστική φράση (pass phrase), υπολογίζει το αποτέλεσμα της hash function αλλά και του μήκους της και από εκεί δημιουργεί ένα τροποποιημένο και μοναδικό, κυρίως, αρχείο μετατρέποντας το σε .exe. Αυτό δίνει την δυνατότητα να έχουμε διαφορετικά «στιγμιότυπα» του ίδιου αρχείου, τόσο για λόγους ασφαλείας (αλλαγή σε περίπτωση ή υποψία παραβίασης ασφαλείας, προστασία από κακόβουλη χρήση, μη επιθυμητή διαρροή του κυρίως αρχείου κ.λ.π.), όσο και για λόγους χρηστικότητας (διαφορετικά στιγμιότυπα για διαφορετικές ομάδες ανθρώπων, δυνατότητα χρήσης από πολλούς διαφορετικούς χρήστες χωρίς περιπτώσεις overlapping κ.λ.π.).

Στο δεύτερο μέρος υπάρχει το κυρίως αρχείο. Εκεί και με την βοήθεια ορισμένων μεταβλητών που υπολογίστηκαν στο πρώτο μέρος, γίνεται

- Η εισαγωγή του κλειδιού,
- η καταχώρηση των χαρακτήρων που θα συνθέσουν το password σε ένα array,
- Ο υπολογισμός του κωδικού επιβεβαίωσης,
- Ο υπολογισμός των σημείων που array όπου βρίσκονται οι χαρακτήρες του password,
- Η σύνθεση του password και τέλος
- Η παρουσίαση του password και του κωδικού επιβεβαίωσης στον χρήστη.

Το κυρίως αρχείο είναι τελείως αυτόνομο και ανεξάρτητο από οποιοδήποτε άλλο βοηθητικό αρχείο από αυτά που περιγράψαμε στο 5ο κεφάλαιο του παρόντος. Δεν απαιτεί την ύπαρξη κάποιου άλλου αρχείου, ούτε καν του αρχείου εγκατάστασης. Αυτό πρακτικά μας δίνει την δυνατότητα να δημιουργήσουμε ένα στιγμιότυπο με την χρήση του αρχείου εγκατάστασης και μιας μυστικής φράσης και κατόπιν να διανείμουμε, με ασφαλή τρόπο, μόνο το κυρίως αρχείο στους χρήστες μας. Έτσι διασφαλίζουμε την ορθότητα της εγκατάστασης, την ακεραιότητα της μυστικής φράσης καθώς και ότι οι χρήστες μας θα

μπορούν να χρησιμοποιήσουν το αρχείο για παραγωγή passwords αλλά δεν θα έχουν την δυνατότητα παραγωγής νέων «στιγμιότυπων» του προγράμματος για άλλη χρήση.

6.1 Το αρχείο εγκατάστασης

Σε αυτό το σημείο θα παρουσιάσουμε και αναλύσουμε τον κώδικα του αρχείου εγκατάστασης, επεξηγώντας όσο το δυνατόν ποιο αναλυτικά τόσο το περιεχόμενο του κώδικα όσο και τους λόγους για τους οποίους γίνονται οι διεργασίες. Πλήρη και συγκεντρωτικό αντίγραφο του κώδικα, με σχόλια, περιλαμβάνεται στο ΠΑΡΑΡΤΗΜΑ Α του παρόντος.

6.1.1 Εισαγωγικά

```
@echo off

color b

@%~d0
@cd "%~p0"

setlocal EnableDelayedExpansion
```

Αποτελούν την επικεφαλίδα του κώδικα. Ουσιαστικά τροποποιούν το «περιβάλλον» στο οποίο θα εκτελεστεί ο κώδικας. Αρχικά λοιπόν κλείνουμε την εκτύπωση των εντολών για το παράθυρο εκτέλεσης. Έτσι η εκτέλεση του αρχείου δεν θα «δείχνει» στον χρήστη τις εντολές που εκτελούνται. Αυτό πέραν του καλύτερου αισθητικού αποτελέσματος στην εμπειρία του χρήστη, συμβάλει και στην ασφάλεια του προγράμματος καθώς έτσι διασφαλίζεται η μη έκθεση του κώδικα.

Στη συνέχεια αλλάζουμε το χρώμα της γραμματοσειράς για μια πιο ευχάριστη εμπειρία για τον χρήστη και επιβεβαιώνουμε ότι το πρόγραμμα θα εκτελεστεί στον φάκελο

που το έχουμε αποθηκευμένο. Τα "" διασφαλίζουν ότι ακόμη και αν υπάρχουν κενά στην διαδρομή του φακέλου, θα εκτελεστεί στον σωστό φάκελο.

Τέλος ενεργοποιούμε το Delayed Expansion. Αυτό, όπως εξηγήσαμε και στο 5.2 Cmd επιτρέπει στον αλγόριθμο να απεμπλακεί από τον περιοριστικό επιτακτικό προγραμματισμό. Επιπλέον με την χρήση των ! ! για να καλέσουμε τις μεταβλητές μας μπορούμε να παρακάμψουμε το πρόβλημα που προκύπτει με τους ειδικούς χαρακτήρες που χρησιμοποιούμε για την δημιουργία του μυστικού κωδικού οι οποίοι σε διαφορετική περίπτωση «ερμηνεύονται» από τον κώδικα σαν εντολές.

6.1.2 Βοηθητικοί φάκελοι και αρχεία

```
ren help utl

attrib +h +s "utl" /s /d

if exist utl\hfps.mik del utl\hfps.mik
if exist utl\hfps1.mik del utl\hfps1.mik
if exist utl\instps.mik del utl\instps.mik
if exist utl\alg.mik del utl\alg.mik
```

Στην συνέχεια του κώδικα γίνονται οι απαραίτητοι έλεγχοι και ενέργειες που αφορούν τους βοηθητικούς φακέλους και τα αρχεία.

Το πρόγραμμα εγκατάστασης περιλαμβάνει ένα φάκελο με την ονομασία help. Εκεί είναι αποθηκευμένα στην αρχή όλα τα βοηθητικά προγράμματα που είδαμε στο 5ο Κεφάλαιο. Κατά την εγκατάσταση εκεί επίσης αποθηκεύονται οποιαδήποτε βοηθητικά αρχεία χρειάζεται να δημιουργηθούν.

Στην αρχή λοιπόν για λόγους ασφαλείας μετονομάζουμε τον βοηθητικό φάκελο από “help” σε “utl”. Αυτό γίνεται γιατί ακόμη και στην περίπτωση που κάποιος καταφέρει να δει το αρχικό όνομα του φακέλου, αυτό να αλλάξει και να μην μπορεί να έχει πρόσβαση. Στην συνέχεια αλλάζουμε τις ιδιότητες του φακέλου κάνοντας τον κρυφό (+h) και φάκελο συστήματος (+s). Με αυτόν τον τρόπο «κρύβουμε» τον φάκελο τελείως. Δεν εμφανίζεται

στο παράθυρο ακόμη και αν έχουμε επιλέξει να μας δείχνει τα κρυφά αρχεία και δεν εμφανίζεται στις λίστες της εντολής dir. Ο μόνος τρόπος για πρόσβαση είναι αν γνωρίζει κάποιος το όνομα του φακέλου.

Τέλος ελέγχουμε στον βοηθητικό φάκελο αν υπάρχουν αρχεία με τα ονόματα των βοηθητικών αρχείων που θα δημιουργήσουμε στην συνέχεια και αν υπάρχουν τα διαγράφουμε καθώς δεν θέλουμε να διακινδυνεύσουμε τις διπλοεγγραφές που μπορεί να επιφέρουν προβλήματα στην λειτουργία του αρχείου.

6.1.3 Εισαγωγή μυστικής φράσης

```
utl\maskedinput.exe "Give the install passphrase:" > utl\instps.mik
```

Κατόπιν περνάμε στην εισαγωγή της μυστικής φράσης. Ζητάμε λοιπόν από τον χρήστη να εισάγει την μυστική φράση. Αυτό γίνεται με την χρήση του προγράμματος maskedinput.exe για να υπερκαλύπτονται οι χαρακτήρες που εισάγουμε από αστερίσκους. Καθώς η μυστική φράση είναι το πιο σημαντικό δεδομένο που εισάγουμε στο πρόγραμμα καθ' όλη την χρήση της εφαρμογής, κρίθηκε απαραίτητο να προστατευθεί από αδιάκριτα βλέμματα κατά την εισαγωγή.

Σαν μυστική φράση μπορούμε να χρησιμοποιήσουμε οτιδήποτε ανεξαιρέτως μεγέθους. Είναι η μόνη πληροφορία που πρέπει να διαμοιραστεί με ασφαλή τρόπο μεταξύ των χρηστών αλλά η χρήση της γίνεται μόνο μια φορά, κατά την εγκατάσταση. Στην πορεία δεν χρειάζεται να την αποθηκεύσουμε ή να την θυμόμαστε για κάποιο λόγο. Έτσι προτείνονται τα παρακάτω:

- Η μυστική φράση να είναι όσο πιο πολύπλοκη γίνεται. Μια καλή επιλογή μπορεί να είναι απόσπασμα από κάποιο βιβλίο ή συγκεκριμένο κομμάτι κειμένου ενός άρθρου. Σε αυτή την περίπτωση μπορεί να γνωστοποιηθεί στους χρήστες σε κωδικοποιημένη μορφή όπως για παράδειγμα [όνομα βιβλίου] – [αριθμός σελίδας] – [γραμμή από – έως].
- Ο διαμοιρασμός της μπορεί να γίνει μαζί με την παραλαβή του αρχείου εγκατάστασης σε μορφή σειριακού αριθμού.

- Η εγκατάσταση να γίνει μόνο από ένα χρήστη. Στην συνέχεια θα διαμοιραστούν αντίγραφα του εκτελέσιμου αρχείου που θα δημιουργηθεί στους υπόλοιπους. Έτσι μόνο ένας ξέρει την μυστική φράση και εξαλείφεται η πιθανότητα λάθους κατά την εγκατάσταση.

Όπως καταλαβαίνουμε οι επιλογές είναι πολλές και γι' αυτό ακριβώς τον σκοπό δεν έχει περιοριστεί με κάποιο τρόπο η μυστική φράση. Αρκεί να κατανοήσουμε πόσο σημαντική είναι η ακεραιότητά του για την εφαρμογή και να χρησιμοποιήσουμε όποιο τρόπο μας διευκολύνει έτσι ώστε να μοιραστούμε εφ' άπαξ την πληροφορία με τους υπόλοιπους χρήστες.

6.1.4 SHA1

```
for %%? in (utl\instps.mik) do ( set /a len=%%~z? -2 )

for /f "tokens=* USEBACKQ" %%F in (`utl\fciv.exe -sha1 utl\instps.mik`)
do echo %%F> utl\hfps.mik

del utl\instps.mik

for /f "tokens=* USEBACKQ" %%F in (`utl\fciv.exe -sha1 utl\hfps.mik`) do
echo %%F> utl\hfps1.mik

del utl\hfps.mik

for /f "tokens=1,*" %%A in (utl\hfps1.mik) do (
    set hf=%%A
)

del utl\hfps1.mik

for /L %%A in (0,1,39) do (
    set ca[%%A]=!hf:~%%A,1!
)
```

Στην συνέχεια αποθηκεύουμε την μυστική φράση σε ένα βοηθητικό αρχείο utl\instps.mik. Η προέκταση .mik έχει επιλεγεί για όλα τα βοηθητικά αρχεία που δημιουργούνται κατά την εγκατάσταση, καθώς δεν αποτελεί κάποια γνωστή προέκταση αρχείου και ακόμη και αν κάποιος καταφέρει να έχει πρόσβαση σε αυτό δεν είναι εμφανές εξ αρχής και δεν γνωρίζει το λειτουργικό σύστημα με ποιο πρόγραμμα θα μπορούσε να «ανοίξει» αυτό το αρχείο.

Στο επόμενο βήμα υπολογίζουμε το μήκος της μυστικής φράσης. Το αποθηκεύουμε σε μια μεταβλητή LEN καθώς θα είναι ένα από τα στοιχεία που θα χρησιμοποιηθούν στον υπολογισμό των μεταβλητών παρακάτω. Αυτό γίνεται για να μοναδικοποιήσουμε όσο το δυνατών περισσότερο τις τιμές των τελικών μεταβλητών μας. Ακόμη λοιπόν και στην περίπτωση που η sha1 μας δώσει ίδιο αποτέλεσμα για δύο διαφορετικές μυστικές φράσεις, σύμφωνα με το πρόβλημα που έχει παρατηρηθεί (βλ. 5.4 FCIV παρ. 3) οι τιμές των μεταβλητών μας θα διαφοροποιούνται καθώς το αρχικό μήκος των μυστικών φράσεων θα είναι διαφορετικό.

Έπειτα χρησιμοποιούμε το εργαλείο FCIV για να υπολογίσουμε το αποτέλεσμα της sha1 για το αρχείο instps.mik. Το αποτέλεσμά της το αποθηκεύουμε στο utl\hfps.mik και σβήνουμε το πρώτο αρχείο μιας και δεν μας είναι πλέον απαραίτητο.

Η όσο το δυνατόν συντομότερη διαγραφή των βοηθητικών αρχείων κρίνεται σημαντική καθώς ελαχιστοποιεί τον κίνδυνο αθέμητης αποκάλυψης των στοιχείων που είναι αποθηκευμένα σε αυτά. Το ίδιο ισχύει και στην εν γένει χρήση τους η οποία έχει περιοριστεί στο ελάχιστο. Συγκεκριμένα και για τους ίδιους λόγους θα δούμε ότι στο κυρίως πρόγραμμα δεν χρησιμοποιούνται βοηθητικά αρχεία και καμία πληροφορία δεν αποθηκεύεται σε κάποιο αποθηκευτικό μέσο.

Συνεχίζοντας επαναλαμβάνουμε την διαδικασία της sha1 άλλη μια φορά για το hfps.mik κάνοντας έτσι την αναστροφή της διαδικασίας ακόμη πιο πολύπλοκη. Επειδή το πρόγραμμα FCIV σαν έξοδο αποθηκεύει και άλλα στοιχεία πέραν του αποτελέσματος της sha1, εξαγάγουμε από το νέο αρχείο που δημιουργήσαμε, την τιμή του αποτελέσματος στην μεταβλητή HF. Το αποτέλεσμα είναι μεγέθους 40 χαρακτήρων δεξαεξαδικού συστήματος.

Τέλος αποθηκεύομε ένα ένα τους χαρακτήρες του αποτελέσματος σε ένα array με όνομα ca[0-39]. Αυτό γίνεται καθώς στην συνέχεια δεν θα χρησιμοποιήσουμε το αποτέλεσμα της hash function αυτούσιο αλλά από αυτό θα υπολογίσουμε κάποιες μεταβλητές.

6.1.5 Υπολογισμός μεταβλητών

```
set /a count=0
set /a cthn=0
set /a cthg=0
for /L %%A in (0,1,39) do (
	set /a ts=!ca[%%A]!
	if !ts! neq 0 (
		set /a count+=!ca[%%A]!*%%A
		set /a cthn+=%%A
	) else (
		set /a cthg+=%%A
	)
)

set /a count=!count!-!len!
set /a cthn=!cthn!+!len!
set /a cthg=!cthg!+!len!

if !cthn! gtr !cthg! (
	set /a cthd=!cthn!-!cthg!
) else (
	set /a cthd=!cthg!-!cthn!
)

set /a arst=!count!/!len!
set /a hlp=!cthg!/!cthd!
set /a arst=!arst!+!hlp!
set /a hlp=!cthn!/!cthd!
set /a arst=!arst!+!hlp!
```

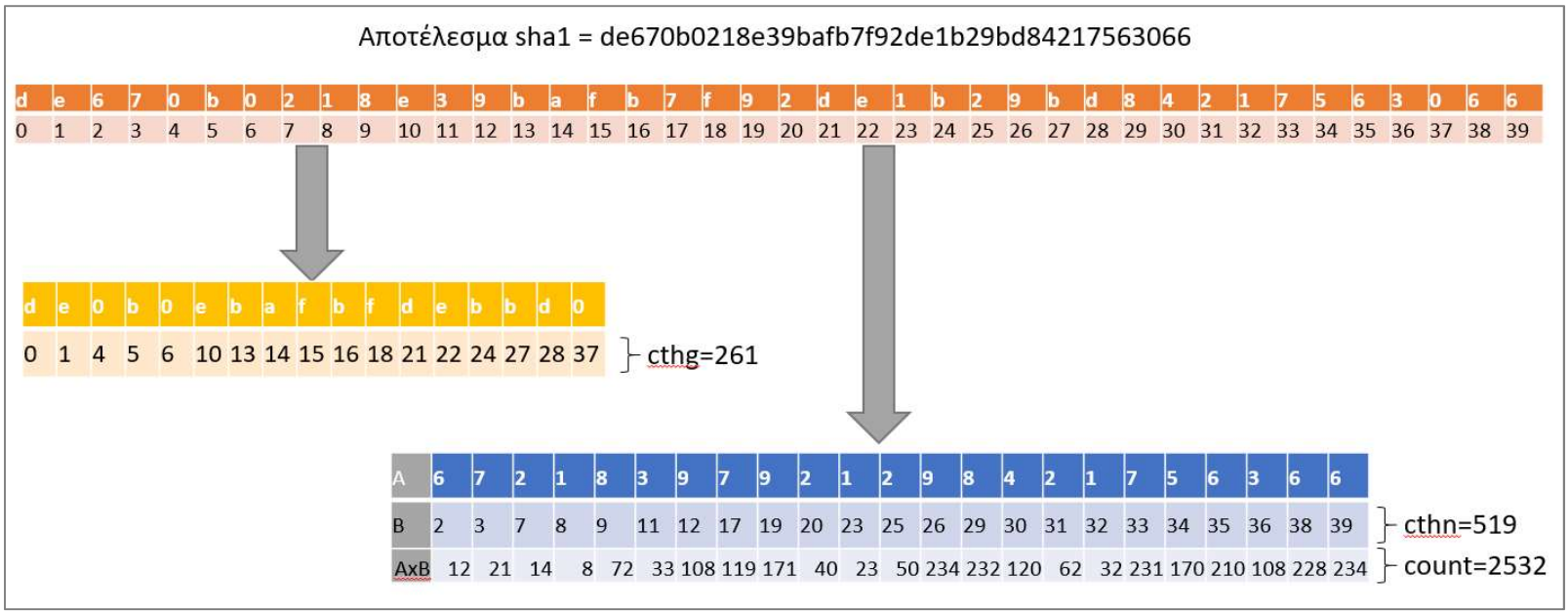
```
:recheckarst
if !arst! geq 100 (
	set /a arst=!arst!/2
	goto recheckarst
)
```

Επόμενο βήμα είναι να χρησιμοποιήσουμε το αποτέλεσμα της sha1 και να δημιουργήσουμε τις μεταβλητές μας οι οποίες θα χρησιμοποιηθούν για να διαφοροποιήσουν και να μοναδικοποιήσουν το τελικό αρχείο. Οι υπολογισμοί που γίνονται σε αυτό το σημείο είναι αρκετά πολύπλοκοι με στόχο όχι μόνο να μοναδικοποιήσουν το δυνατόν περισσότερο τα αποτελέσματα αλλά και για να καταστήσουν την αντιστροφή της διαδικασίας δύσκολη. Ακόμη λοιπόν και αν κάποιος γνώριζε την διαδικασία που ακολουθούμε, θα ήταν πολύ δύσκολο σε μια πιθανή διαρροή των αποτελεσμάτων της κύριας εφαρμογής, να μπορέσει να αντιστρέψει όλες τις πράξεις και να φτάσει στις αρχικές τιμές και στον ακριβή υπολογισμό των τιμών που θα του έδινε την δυνατότητα να «σπάσει» τον κώδικα και να παράξει τα ίδια αποτελέσματα.

Ξεκινάμε λοιπόν με τον μηδενισμό των μεταβλητών. Στην συνέχεια ελέγχουμε με την σειρά όλα τα ψηφία του αποτελέσματος της sha1 αν είναι γράμματα ή αριθμοί. Η μεταβλητή cthg υπολογίζει το άθροισμα των θέσεων στην συστοιχία (array) οι οποίες περιέχουν γράμματα ή την τιμή μηδέν ενώ η μεταβλητή cthn το άθροισμα των θέσεων που περιέχουν αριθμούς εκτός του μηδενός. Τέλος η μεταβλητή count αθροίζει το γινόμενο όλων των αριθμών εκτός του μηδενός, που περιλαμβάνονται στο αποτέλεσμα της sha1 επί της θέσης στην οποία βρίσκονται στην συστοιχία.

Για να καταλάβουμε καλύτερα τους υπολογισμούς που γίνονται αρκεί να δούμε την εικόνα που ακολουθεί:

ΕΙΚΟΝΑ 1 *Υπολογισμός μεταβλητών*



Επιπλέον οι τιμές των παραπάνω μεταβλητών αλλάζουν βάση του συνολικού μήκους της μυστικής φράσης. Έτσι στις μεταβλητές cthg και cthn προστίθεται το μήκος της μυστικής φράσης και από την μεταβλητή count αφαιρείται. Αυτή η διαδικασία μοναδικοποιεί ακόμη περισσότερο τις τιμές με βάση την μυστική φράση που έχουμε εισάγει.

Στην συνέχεια υπολογίζουμε την μεταβλητή arst. Αυτή η μεταβλητή είναι που θα χρησιμοποιηθεί στο κυρίως αρχείο και υποδηλώνει την «αρχική θέση» από την οποία θα ξεκινήσει να γίνεται η καταχώρηση, στο array, των χαρακτήρων που θα χρησιμοποιηθούν για την παραγωγή του κωδικού μας. Περισσότερα θα αναλύσουμε στην παράγραφο 6.2.4. Υπολογίζουμε και αποθηκεύουμε στην μεταβλητή cthd, την διαφορά των μεταβλητών cthn και cthg (ελέγχοντας ποια είναι η μικρότερη έτσι ώστε να αφαιρεθεί από την μεγαλύτερη για να έχουμε πάντα θετικό αριθμό) και στην συνέχεια αρχίζουμε τον υπολογισμό της arts η οποία είναι:

$$Arts=(count/len)+(cthg/cthd)+(cthn/cthd)$$

δηλαδή το άθροισμα του πηλίκου της count με το μήκος της μυστικής φράσης και των cthg και cthn με την διαφορά τους. Στο τέλος γίνεται ένας τελευταίος έλεγχος για να δούμε αν

το αποτέλεσμα της arst είναι μεγαλύτερο από το 100 και σε θετική περίπτωση το διαιρούμε στην μέση. Αυτό είναι απαραίτητο γιατί όπως θα δούμε στην παράγραφο 6.2.4 το χρησιμοποιούμε για να γεμίσουμε ένα array μήκους 100 χαρακτήρων (0-99) οπότε το σημείο αρχής αυτού του array δεν μπορεί να είναι μεγαλύτερο από το συνολικό μήκος του.

6.1.6 Δημιουργία κυρίως αρχείου

Επόμενο βήμα είναι η δημιουργία του κώδικα για το κυρίως αρχείο της λεπτομέρειες για το οποίο θα αναλύσουμε στην Παράγραφο 6.2. Ο κώδικας αποθηκεύεται προσωρινά σε ένα βοηθητικό αρχείο alg.mik.

```
utl\Bat_To_Exe.exe /bat utl\alg.mik /exe sharepass.exe /icon utl/key.ico
>nul

del utl\alg.mik
```

Επόμενο βήμα είναι η μετατροπή του βοηθητικού αρχείου σε εκτελέσιμο για να προστατευθεί ο κώδικάς του. Αυτό γίνεται με την εφαρμογή Bat_To_Exe που περιγράψαμε στην Παράγραφο 5.5 και αμέσως μετά σβήνουμε το βοηθητικό αρχείο.

6.1.7 Κλείσιμο του κώδικα.

```
set /a inst_pass=0
set /a len=0
set /a hf=0
set /a ts=0
set /a hlp=0
set /a arst=0
for /L %%A in (0,1,39) do (
      set ca[%%A]=0
)
set /a count=0
set /a cthn=0
```

```
set /a cthg=0

if exist utl\*.mik del /Q /F utl\*.mik
timeout /T 3 /NOBREAK >nul
del /Q /F utl\Bat_To_Exe.exe
rd /S /Q utl\

cls
echo Installation completed successfully
echo Please run SHAREPASS.EXE to generate your passwords
echo.
echo For more information run "SHAREPASS.EXE help"

timeout /T 3 /NOBREAK >nul
```

Σαν τελευταίο βήμα έχουμε τον μηδενισμό όλων των μεταβλητών που χρησιμοποιήθηκαν και στο σβήσιμο όλων των βοηθητικών αρχείων και φακέλων. Αυτό εξασφαλίζει ότι καμία από τις τιμές των μεταβλητών μας δεν θα παραμείνουν αποθηκευμένες στην προσωρινή μνήμη του συστήματος μας το οποίο θα αποτελούσε ένα ακόμη πιθανό σημείο αποτυχίας εξασφάλισης της ακεραιότητας της εφαρμογής.

Κλείνοντας το πρόγραμμα επιστρέφει το μήνυμα της επιτυχούς ολοκλήρωσης της εγκατάστασης και ενημερώνει τον χρήστη για τον τρόπο λειτουργείας της εφαρμογής.

```
set /a cmp=0
set br==
cls
echo Instalation Prosses Completed %br% %cmp%%%
```

Τέλος υπάρχει ένα κομμάτι αλγόριθμου το οποίο επαναλαμβάνεται σε τακτά χρονικά διαστήματα. Αυτό γίνεται με γωόμονα την εμπειρία του χρήστη κατά την λειτουργεία της εφαρμογής καθώς τον ενημερώνει οπτικά για την πορεία της εγκατάστασης.

6.2 Το κυρίως αρχείο

Σε αυτό το σημείο θα παρουσιάσουμε και αναλύσουμε τον κώδικα του κυρίως αρχείου της εφαρμογής, επεξηγώντας όσο το δυνατόν ποιο αναλυτικά τόσο το περιεχόμενο του κώδικα όσο και τις τεχνικές που υιοθετήθηκαν. Πλήρη και συγκεντρωτικό αντίγραφο του κώδικα, με σχόλια, περιλαμβάνεται στο ΠΑΡΑΡΤΗΜΑ Β του παρόντος.

6.2.1 Εισαγωγικά

```
@echo off
color b
setlocal EnableDelayedExpansion

set /a check=0

set /a n=%ARST%
```

Ο κώδικας ξεκινάει, όπως είδαμε και στην εισαγωγή του κώδικα εγκατάστασης, με το κλείσιμο της εκτύπωσης, την αλλαγή του χρώματος της γραμματοσειράς που εμφανίζεται στην οθόνη και την ενεργοποίηση του Delayed Expansion προκειμένου να απεμπλακεί από τον περιοριστικό επιτακτικό προγραμματισμό. Στην συνέχεια μηδενίζουμε την τιμή της μεταβλητής check την οποία θα χρησιμοποιήσουμε στην πορεία.

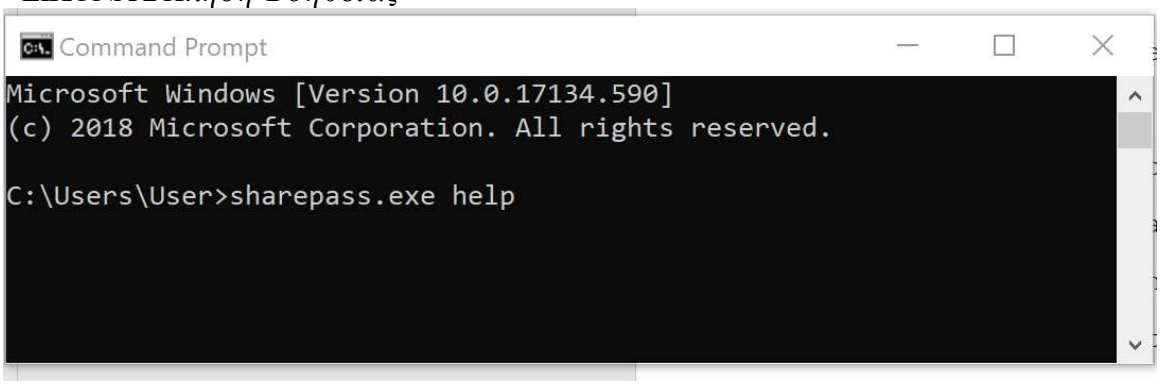
Τέλος δίνουμε την τιμή της μεταβλητής arst που αναλύσαμε πως υπολογίζεται στην παράγραφο 6.1.5 του προηγούμενου κεφαλαίου. Σημειώνεται εδώ ότι όταν δημιουργηθεί ο κώδικας από το αρχείο εγκατάστασης, αντί για το όνομα της μεταβλητής υπάρχει η τιμή που έχει πάρει αυτή. Έτσι μεταφέρεται ο υπολογισμός της από το ένα αρχείο στο άλλο αλλάζοντας και κάνοντας μοναδικό το κυρίως αρχείο.

6.2.2 Βοηθητικό μήνυμα

```
if "%~1"=="help" (
    echo ------------------------------
    echo :                            :
    echo :       sharepass.exe        :
    echo :          v 1.0             :
    echo :     2019 Mike Gkoumas      :
    echo :                            :
    echo ------------------------------
    echo.
    echo.
    echo This script produces a safe password
    echo when given a four digit number.
    echo It also returns a four digit check number.
    echo.
    echo Run it as an application and follow
    echo the instructions on the screen.
    echo.
    echo For command line and batch file usage type
    echo sharepass.exe [key]
    echo where [key] is the four digit number
    echo and the password with the check number
    echo will be printed separated by a whitespace
    echo.
    echo This application was created with the use
    echo of a passphrase during the installation.
    echo Only installations with the same passphrase will
    echo generate the same passwords and check numbers.
    echo.
    pause
    exit
)
```

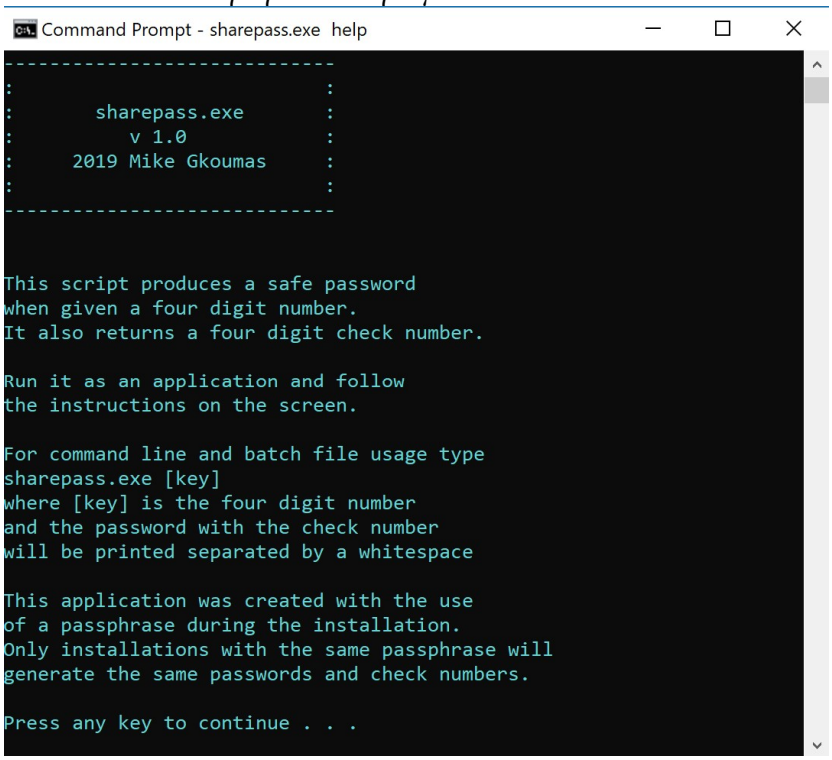
Στην συνέχεια του κώδικα δημιουργούμε το μήνυμα που παρουσιάζεται στον χρήστη σαν βοήθεια. Για να έχει πρόσβαση σε αυτό το μήνυμα ο χρήστης, αρκεί να καλέσει την εφαρμογή από την γραμμή εντολών και μετά να γράψουμε την λέξη help όπως βλέπουμε στην εικόνα που ακολουθεί.

ΕΙΚΟΝΑ 2 *Κλήση Βοήθειας*



Όταν γίνει αυτό, στον χρήστη επιστρέφεται ένα μήνυμα που του εξηγεί την λειτουργία της εφαρμογής. Ξεκινάει με ενημέρωση για την έκδοση της εφαρμογής και στην πορεία υπάρχει ένα κείμενο που εξηγεί πως μπορεί να χρησιμοποιηθεί αυτή τόσο με κλήση της από γραμμή εντολών όσο και με εκτέλεση από περιβάλλον windows καθώς και τα αποτελέσματα που επιστρέφει.

ΕΙΚΟΝΑ 3 *Βοηθητικό Μήνυμα*



6.2.3 Έλεγχος τρόπου λειτουργίας

```
if not "%~1"=="" set /a ena=%1
if "%~1"=="" set ena=notthere
```

Στην συνέχεια ελέγχουμε τον τύπο λειτουργείας της εφαρμογής. Αυτό είναι σημαντικό για να ξέρει η εφαρμογή στην πορεία ποιο τρόπο χρήσης έχει επιλέξει ο χρήστης και να επιστρέψει τα αποτελέσματα με τον σωστό τρόπο. Η τιμή της μεταβλητής %~1 είναι αυτή που παίρνει την τιμή της πρώτης λέξης που βρίσκεται μετά το όνομα του αρχείου που καλούμε στην γραμμή εντολών όπως είδαμε στην ΕΙΚΟΝΑ 2.

6.2.4 Γέμισμα του array

```
set "ar[%n%]=@"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=/"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=~"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=?"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=>"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=."
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=8"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=7"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=4"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=;"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=O"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=z"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=["
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=}"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=d"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=@"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=R"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=8"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=m"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=0"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=k"
if %n% equ 99 set /a n=-1
```

```
set /a n+=1
set "ar[%n%]=2"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=y"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=Z"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=v"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=5"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=r"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=<"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=n"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=@"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=M"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=t"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=b"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=+"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]={"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=-"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=U"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=&"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=o"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=#"
if %n% equ 99 set /a n=-1
```

```
set /a n+=1
set "ar[%n%]=L"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=^"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=("
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=3"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=x"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=1"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=C"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=:"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=G"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=$"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=K"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=f"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=H"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=E"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=W"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=="
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=Y"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=1"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=e"
if %n% equ 99 set /a n=-1
```

```
set /a n+=1
set "ar[%n%]=5"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=q"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=9"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=7"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=6"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=3"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=\"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=A"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=]"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=u"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=6"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=T"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=i"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=h"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=S"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=p"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=s"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=c"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=w"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=a"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=)"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=D"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=|"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=0"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=9"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=,"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=%%"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=*"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=4"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=_"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=2"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=X"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=j"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=N"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=P"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=Q"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=g"
```

```
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=J"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=F"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=V"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=B"
```

Επόμενο βήμα είναι να γεμίσουμε το array των χαρακτήρων που θα αποτελέσουν την φαρέτρα από όπου θα επιλεγούν στην συνέχεια οι χαρακτήρες που θα δημιουργήσουν τον κωδικό ασφαλείας μας.

Για την δημιουργία του κωδικού χρησιμοποιήθηκαν 89 διαφορετικοί χαρακτήρες. Όπως είδαμε στην παράγραφο 2.1 Χαρακτήρες Κωδικών για να έχουμε ένα ασφαλή κωδικό είναι απαραίτητο να χρησιμοποιηθούν όσο το δυνατόν περισσότεροι χαρακτήρες. Έτσι η εφαρμογή μας χρησιμοποιεί όλους τους χαρακτήρες αριθμών (0-9) αλλά υπάρχουν ορισμένοι περιορισμοί στις άλλες 2 ομάδες χαρακτήρων.

6.2.4.1 Ειδικοί χαρακτήρες

Για την δημιουργία του κωδικού έχουν χρησιμοποιηθεί 29 από τους 33 ειδικούς χαρακτήρες που είδαμε στην παράγραφο 2.1. Δεν έχουν χρησιμοποιηθεί οι παρακάτω χαρακτήρες:

- Ο χαρακτήρας του κενού (Space) καθώς όχι απλώς δημιουργεί πρόβλημα στον κώδικα αναγνωρίζοντας δύο λέξεις αντί για μια αλλά μπορεί να μπερδέψει τον χρήστη καθώς δεν είναι ένας χαρακτήρας που εκτυπώνει κάτι στην οθόνη. Κυρίως για την αυτοματοποιημένη χρήση της εφαρμογής το επιστρεφόμενο αποτέλεσμα πρέπει να είναι δύο λέξεις, ο μυστικός κωδικός και ο κωδικός επιβεβαίωσης. Αυτό κάνει εύκολο τον διαχωρισμό τους από άλλα προγράμματα τύπου cmd scripts. Αν

ο μυστικός κωδικός περιλάμβανε τον χαρακτήρα του κενού, το επιστρεφόμενο αποτέλεσμα θα ήταν 3 διαφορετικές λέξεις και οποιοδήποτε αυτόματο πρόγραμμα θα είχε προβλήματα να ξεχωρίσει που τελειώνει ο μυστικός κωδικός και που αρχίζει ο κωδικός επιβεβαίωσης.

- Ο χαρακτήρας του θαυμαστικού (!). Αυτό έγινε λόγω περιορισμού των batch files καθώς αυτός ο χαρακτήρας αναπαριστά την κλήση των μεταβλητών και δημιουργούσε πρόβλημα στην λειτουργία του κώδικα.
- Οι χαρακτήρες μονών εισαγωγικών (‘ και `) καθώς είναι πανομοιότυποι και θα ήταν εύκολο να μπερδέψουν τον χρήστη να ξεχωρίσει ποιος από τους δύο περιλαμβάνεται στον κώδικα.

6.2.4.2 Λατινικοί χαρακτήρες

Επίσης υπάρχουν δύο λατινικοί χαρακτήρες που δεν έχουν χρησιμοποιηθεί γιατί η εκτύπωσή τους είναι οπτικά πολύ όμοια και δεν θα μπορούσε εύκολα να ξεχωρίσει ο χρήστης ποιος από τους δύο χρησιμοποιείται κάθε φορά. Αυτοί είναι το μικρό «ελ» (l) και το κεφαλαίο «άι» (I).

Η εξαίρεση των χαρακτήρων που είναι οπτικά όμοιοι είναι κοινή πρακτική. Το Bitcoin, και άλλα κρυπτονομίσματα, για την δημιουργία των διευθύνσεών του χρησιμοποιεί την ομάδα χαρακτήρων BASE58 που δημιούργησε ο Satoshi Nakamoto (Wikipedia, n.d.) η οποία εξαιρεί τους χαρακτήρες που έχουν όμοια απεικόνιση προκειμένου να μην δυσκολεύει την χρήση τους από τους ιδιοκτήτες του κρυπτονομίσματος. Εκτός από τους παραπάνω χαρακτήρες ο ομάδα BASE58 εξαιρεί, για τους ίδιους λόγους, το μηδέν και το κεφαλαίο όμικρον αλλά και τους χαρακτήρες + και / γιατί δεν βοηθάνε στην γρήγορη επιλογή μιας σειράς χαρακτήρων για αντιγραφή.

Έτσι καταλήγουμε σε ένα συνολικό αριθμό από 89 διαφορετικούς χαρακτήρες. Αυτό όπως είδαμε στην παράγραφο 2.2 και γνωρίζοντας ότι το μήκος του κωδικού που θα δημιουργήσουμε είναι δεκαέξι χαρακτήρων, μας δίνει ένα συνολικό αριθμό 89^{16}= 15.496.731.425.178.936.435.099.327.730.561 αριθμός που μπορεί να θεωρηθεί άκρως ικανοποιητικός και ασφαλής απέναντι σε επιθέσεις εξαντλητικής αναζήτησης.

Καθώς όμως το array πρέπει να έχει εκατό στοιχεία, υπάρχουν 11 χαρακτήρες που επαναλαμβάνονται. Αυτό για να ενισχύσει την πολυπλοκότητα αποφασίστηκε να επαναληφθεί η «ομάδα» χαρακτήρων που περιέχει το μικρότερο πλήθος για να αυξηθούν οι πιθανότητες ο κωδικός να περιλαμβάνει κάποιον από αυτούς. Έτσι οι αριθμοί επαναλαμβάνονται μαζί με τον χαρακτήρα @.

6.2.4.3 Σειρά χαρακτήρων

Όλα τα παραπάνω μας δίνουν ένα συνολικό πλήθος 100 χαρακτήρων από τους οποίους θα επιλέξουμε τους δεκαέξι που θα συνθέσουν τον μυστικό κωδικό μας. Έπρεπε λοιπόν αυτοί οι χαρακτήρες να μπούνε σε μια σειρά έτσι ώστε να δημιουργηθεί το array αλλά ταυτόχρονα να μην ήταν μια «λογική» σειρά η οποία θα μπορούσε με απλούς υπολογισμούς να οδηγήσει σε διαπίστωση της σειράς τους και κατ' επέκταση σε κενό ασφαλείας του αλγόριθμου.

Έτσι λοιπόν στην αρχή ομαδοποιήθηκαν οι χαρακτήρες σε 4 γκρουπ:

- Μικροί λατινικοί χαρακτήρες
- Αριθμοί και οι πρώτοι 15 ειδικοί χαρακτήρες (κατά σειρά που παρουσιάζονται στο πληκτρολόγιο qwerty)
- Κεφαλαίοι λατινικοί χαρακτήρες
- Αριθμοί και οι υπόλοιποι 15 ειδικοί χαρακτήρες (με επανάληψη του @)

Στην συνέχεια ταξινομήθηκαν οι χαρακτήρες σε ένα πίνακα 10x10 με επιλογή ενός χαρακτήρα από κάθε ομάδα, γεμίζοντας τις θέσεις του διαγώνια ξεκινώντας από την κύρια διαγώνιο και συνεχίζοντας εναλλάξ επάνω και κάτω από αυτή. Αυτό μας έδωσε τον παρακάτω πίνακα:

ΠΙΝΑΚΑΣ 5 *Σειρά Χαρακτήρων*

	0	1	2	3	4	5	6	7	8	9
0	@	O	k	M	L	K	q	T	D	X
1	/	z	2	t	^	f	9	i	|	j
2	~	[	y	b	(	H	7	h	0	N
3	?	}	Z	+	3	E	6	S	9	P
4	>	d	v	{	x	W	3	p	,	Q
5	.	"	5	-	1	=	\	s	%	g
6	8	R	r	U	C	Y	A	c	*	J
7	7	8	<	&	:	1	]	w	4	F
8	4	m	n	o	G	e	u	a	_	V
9	;	0	@	#	$	5	6	)	2	B

Έτσι μπορούμε πλέον να δημιουργήσουμε το array καθώς έχουμε την αντιστοιχία θέσης με κάθε χαρακτήρα. Βλέπουμε για παράδειγμα ότι η θέση 35 αντιστοιχεί στον χαρακτήρα E, η θέση 82 στον χαρακτήρα n κ.λ.π.

Σαν ένα τελευταίο μέτρο ασφαλείας η καταχώρηση του παραπάνω πίνακα στο array δεν γίνεται από την πρώτη θέση 0 αλλά ξεκινάει από την θέση που έχει υπολογιστεί και αποθηκεύει στην μεταβλητή arst στο αρχείο εγκατάστασης. Αυτό μοναδικοποιεί την κάθε εγκατάσταση με αποτέλεσμα τα ίδια κλειδιά να παράγουν διαφορετικό κωδικό για κάθε διαφορετική «εκδοχή» της εφαρμογής. Ξεκινώντας λοιπόν από την θέση arst γεμίζουμε το array και όταν φτάσουμε στην τελευταία θέση 99 συνεχίζουμε από την αρχή του array τοποθετώντας έτσι όλους τους χαρακτήρες.

6.2.5 Εισαγωγή Κλειδιού

```
:redo
set /a chk=0
if %ena%==notthere (
    set /p give="Enter the given passkey: "
    set /a strt=!give!
)
if not %ena%==notthere (
    set /a strt=%ena%
```

```
)
if %strt% leq 999 (
    set /a chk=1
    if not %ena%==notthere set /a chk=3
)
if %strt% geq 10000 (
    set /a chk=1
    if not %ena%==notthere set /a chk=3
)
if %chk%==1 (
    cls
    echo.
    echo wrong passkey
    echo please try again
    echo.
    goto redo
)
if %chk%==3 exit
```

Επόμενο βήμα είναι η εισαγωγή του κλειδιού παραγωγής του κωδικού. Γίνεται έλεγχος του τρόπου λειτουργείας και ανάλογα η μεταβλητή strt είτε παίρνει την τιμή που δώσαμε κατά την κλήση της εφαρμογής από την γραμμή εντολών, είτε ζητείται από τον χρήστη να εισάγει τον κωδικό.

Στην συνέχεια γίνεται έλεγχος του κλειδιού αν είναι τετραψήφιο. Αν έχουμε εισάγει λάθος κλειδί, εκτυπώνεται το ανάλογο μήνυμα λάθους και ο κώδικας επιστρέφει στην εισαγωγή του. Αν η κλήση της εφαρμογής έγινε από γραμμή εργαλείων και δεν έχουμε δώσει σωστό κλειδί η εφαρμογή κλείνει την λειτουργία της χωρίς να επιστρέψει κάποιο αποτέλεσμα.

6.2.6 Κωδικός επιβεβαίωσης

```
set /a strt=%strt%+%COUNT%
if %strt% geq 10000 ( set /a strt=%strt%/10 )
set /a fst=%strt%/1000
set /a sub=%fst%*1000
set /a rem=%strt%-%sub%
set /a snd=%rem%/100
set /a sub=%snd%*100
set /a rem=%rem%-%sub%
set /a trd=%rem%/10
set /a sub=%trd%*10
set /a frt=%rem%-%sub%
set /a sm=%fst%+%snd%+%trd%+%frt%
set /a mpa=%CTHN%+%sm%
set /a mpb=%CTHG%+%sm%
set /a mpc=%CTHD%+%sm%
set /a mpd=%n%+%sm%
set /a smaa=%frt%*%mpa%
set /a smbb=%trd%*%mpb%
set /a smcc=%snd%*%mpc%
set /a smdd=%fst%*%mpd%
set /a check=%smaa%+%smbb%+%smcc%+%smdd%+%COUNT%
:recalc

if %check% geq 10000 (
    set /a check=%check%/10
    goto recalc
)
set /a fstc=%check%/1000
set /a sub=%fstc%*1000
set /a rem=%check%-%sub%
set /a sndc=%rem%/100
```

```
set /a sub=%sndc%*100
set /a rem=%rem%-%sub%
set /a trdc=%rem%/10
set /a sub=%trdc%*10
set /a frtc=%rem%-%sub%
```

Επόμενο βήμα είναι να επεξεργαστούμε το κλειδί εισαγωγής και με αυτό να φτιάξουμε τον κωδικό επιβεβαίωσης και τις θέσεις του array από όπου θα πάρουμε τους χαρακτήρες για να δημιουργήσουμε τον μυστικό κωδικό μας.

Στην αρχή προσθέτουμε την τιμή της μεταβλητής count που υπολογίστηκε κατά την εγκατάσταση, όπως είδαμε στην παράγραφο 6.1.5, σαν ένα ακόμη βήμα για να κάνουμε την κάθε εγκατάσταση μοναδική. Στην συνέχεια χωρίζουμε το κάθε ψηφίο του και το αποθηκεύουμε στις μεταβλητές fst, snd, trd και frt αντίστοιχα. Αφού υπολογίσουμε το άθροισμα των ψηφίων, το χρησιμοποιούμε για να συνθέσουμε τις μεταβλητές που θα μας βοηθήσουν να υπολογίσουμε τον κωδικό επιβεβαίωσης. Έτσι προσθέτουμε στο άθροισμα τις μεταβλητές cthn, cthg, ctgd και arst που υπολογίσαμε κατά την εγκατάσταση (βλ. παράγραφο 6.1.5). αποθηκεύουμε τα αποτελέσματα στις μεταβλητές mpa, mpb, mpc και mpd. Πολλαπλασιάζουμε αυτές τις τιμές με τα ψηφία του κλειδιού εισαγωγής και προσθέτουμε τα αποτελέσματα μαζί με την τιμή της μεταβλητής count. Αν το αποτέλεσμα είναι πενταψήφιο, παίρνουμε τα 4 πρώτα ψηφία και αυτό θα αποτελεί τον κωδικό επιβεβαίωσης. Τέλος χωρίζουμε τα ψηφία του κωδικού αυτού και τα αποθηκεύουμε στις μεταβλητές fstc, sndc, trdc και frtc.

Όπως γίνεται κατανοητό οι μεταβλητές που έχουν υπολογιστεί κατά την εγκατάσταση του προγράμματος παίζουν σημαντικό ρόλο στον υπολογισμό όλων των βασικών μεταβλητών της κύρια εφαρμογής. Αυτό εξασφαλίζει την μοναδικότητα της κάθε εγκατάστασης. Από το μήκος της μυστικής φράσης, στο αποτέλεσμα της sha1 αυτής, στην θέση των διαφορετικών ψηφίων αυτής μέχρι στην αποθήκευση των χαρακτήρων στο array και τον υπολογισμό των θέσεων και του κωδικού επιβεβαίωσης η κάθε εγκατάσταση μπορούμε να πούμε με ασφάλεια ότι είναι μοναδική. Επιπλέων οι υπολογισμοί έχουν γίνει πολύπλοκοι με μοναδικό στόχο να μην μπορεί να γίνει αντιστροφή της διαδικασίας και να έχουμε έτσι κενό ασφαλείας.

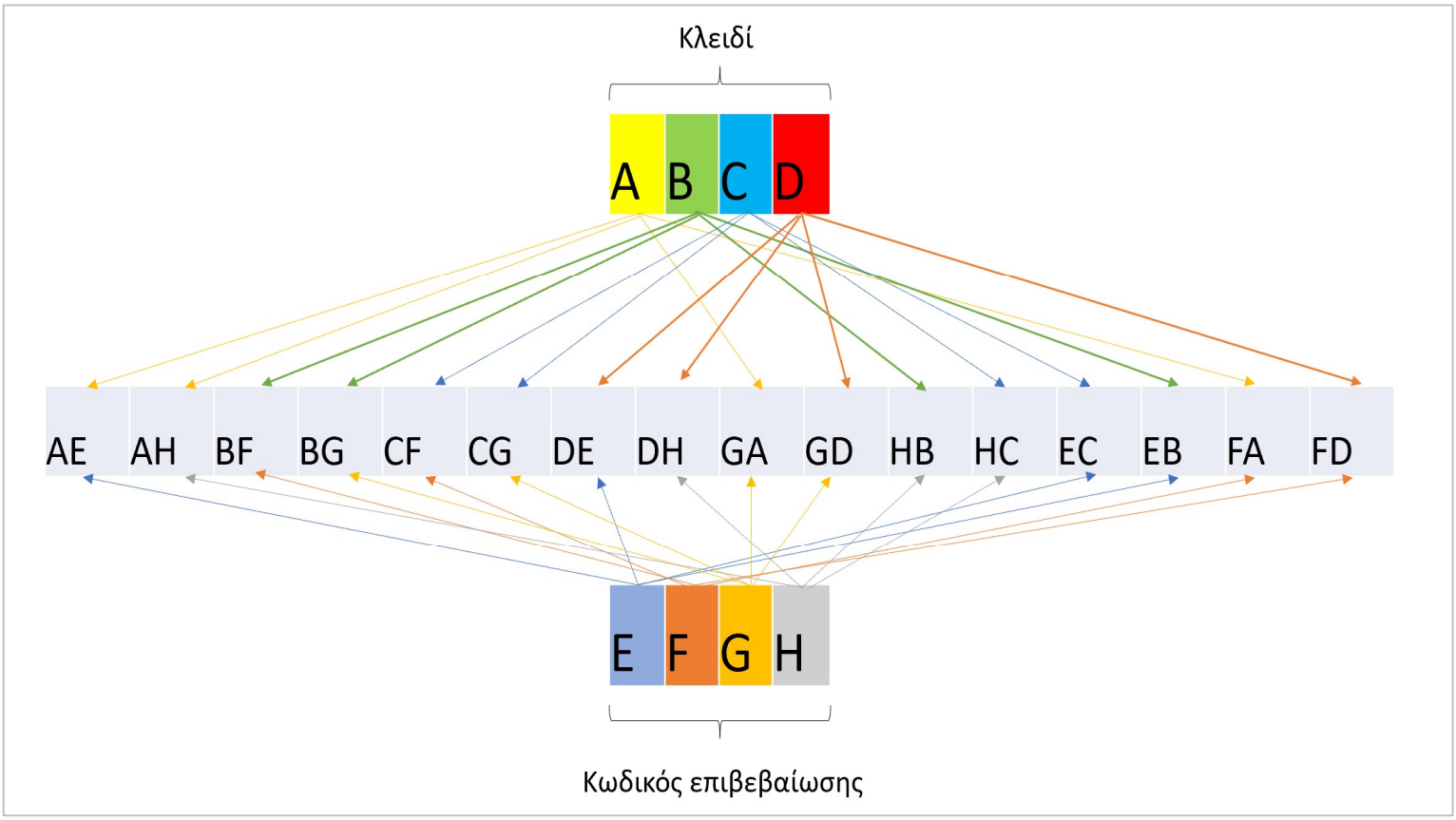
6.2.7 Σύνθεση του μυστικού κωδικού

```
set /a sub=%fst%*10
set /a a=%sub%+%fstc%
set /a e=%sub%+%frtc%
set /a sub=%snd%*10
set /a b=%sub%+%sndc%
set /a f=%sub%+%trdc%
set /a sub=%trd%*10
set /a c=%sub%+%trdc%
set /a g=%sub%+%sndc%
set /a sub=%frt%*10
set /a d=%sub%+%fstc%
set /a h=%sub%+%frtc%
set /a sub=%trdc%*10
set /a i=%sub%+%fst%
set /a p=%sub%+%frt%
set /a sub=%frtc%*10
set /a j=%sub%+%snd%
set /a o=%sub%+%trd%
set /a sub=%fstc%*10
set /a k=%sub%+%trd%
set /a n=%sub%+%snd%
set /a sub=%sndc%*10
set /a m=%sub%+%fst%
set /a l=%sub%+%frt%
set
ps=!ar[%a%]!!ar[%b%]!!ar[%c%]!!ar[%d%]!!ar[%e%]!!ar[%f%]!!ar[%g%]!!ar[%h
%]!!ar[%i%]!!ar[%j%]!!ar[%k%]!!ar[%l%]!!ar[%m%]!!ar[%n%]!!ar[%o%]!!ar[%p
%]!
```

Το μόνο που απομένει πλέον είναι να συνθέσουμε τον μυστικό κωδικό. Αυτό γίνεται με συνδυασμό των ψηφίων του κλειδιού και του κωδικού επιβεβαίωσης. Ο συνδυασμός αυτών μας δίνει δεκαέξι αριθμούς που υποδηλώνουν τις θέσεις του array που

περιέχουν τα ψηφία του μυστικού κωδικού. Ο τρόπος συνδυασμού αναλύεται και παρουσιάζεται στην παρακάτω εικόνα.

ΕΙΚΟΝΑ 4 *Μυστικός Κωδικός*



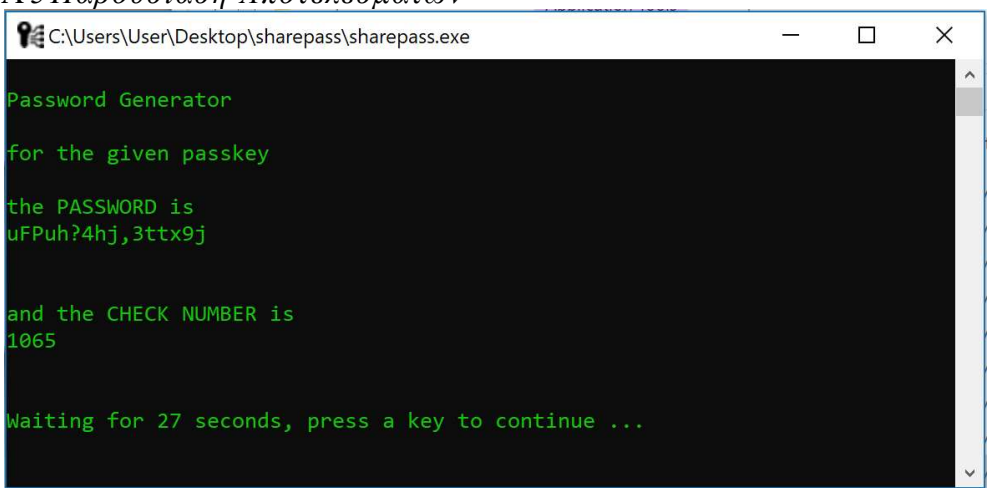
Συνδυάζοντας λοιπόν κατάλληλα τα ψηφία των δύο τετραψήφιων κωδικών δημιουργούμε τον μυστικό κωδικό από τους χαρακτήρες που είναι αποθηκευμένοι στις αντίστοιχες θέσεις του array και τον αποθηκεύουμε στην μεταβλητή ps.

6.2.8 Παρουσίαση αποτελεσμάτων

```
color a
if %ena%==notthere (
    cls
    echo Password Generator
    echo.
    echo for the given passkey
    echo.
    echo the PASSWORD is
    echo !ps!
    echo and the CHECK NUMBER is
    echo %check%
)
if not %ena%==notthere (
    echo !ps! %check%
)
```

Στην συνέχεια παρουσιάζονται τα αποτελέσματα στον χρήστη. Γίνεται έλεγχος του τρόπου λειτουργίας και είτε εμφανίζονται στην οθόνη τα αποτελέσματα σε ένα φιλικό προς τον χρήστη τρόπο περιμένοντας αρκετό χρόνο για να μπορεί ο να γίνει αντιγραφή της πληροφορίας, είτε επιστρέφεται ο μυστικός κωδικός και ο κωδικός επιβεβαίωσης χωρισμένοι με ένα κενό, για χρήση από οποιοδήποτε αυτοματοποιημένο πρόγραμμα.

ΕΙΚΟΝΑ 5 *Παρουσίαση Αποτελεσμάτων*



6.2.9 Κλείσιμο κώδικα

```
set ps=0
set /a check=0
set a=0
set b=0
set c=0
set d=0
set e=0
set f=0
set g=0
set h=0
set i=0
set g=0
set k=0
set l=0
set m=0
set n=0
set o=0
set p=0
set /a sub=0
set /a rem=0
set /a mpa=0
set /a mpb=0
set /a mpc=0
set /a mpd=0
set /a smaa=0
set /a smab=0
set /a smac=0
set /a smad=0
set /a fst=0
set /a snd=0
set /a trd=0
```

```
set /a frt=0
set /a fstc=0
set /a sndc=0
set /a trdc=0
set /a frtc=0
set /a strt=0
set /a chk=0
set give=0
for /L %%A in (0,1,39) do (
     set ar[%%A]=0
)
if %ena%==notthere (
    set /a ena=0
    timeout /T 30
)
set /a ena=0
```

Τέλος γίνεται ο μηδενισμός των μεταβλητών που χρησιμοποιήθηκαν κατά την λειτουργεία του κυρίους κώδικα, όπως είδαμε και στην παράγραφο 6.1.7 και δίνεται αρκετός χρόνος στον χρήστη πριν κλείσει η εφαρμογή.

7. ΑΝΑΛΥΣΗ ΕΥΠΑΘΕΙΩΝ

Η ίδια η χρήση της εφαρμογής επιτάσσει να έχουν ληφθεί όλα τα απαραίτητα μέτρα για την δημιουργία μιας ασφαλούς εφαρμογής η οποία θα μπορεί να δώσει σιγουριά και εμπιστοσύνη κατά την χρήση της. Από την αντιμετώπιση πιθανών ευπαθειών της sha1 έως στον πολύπλοκο υπολογισμό των μεταβλητών, έχει ληφθεί μέριμνα ούτως ώστε να μην υπάρχουν κενά ασφαλείας έτσι ώστε ο τελικός χρήστης να ξέρει ότι χειρίζεται μια εφαρμογή που περιλαμβάνει κάθε δυνατό μέτρο ασφαλείας.

7.1 Κλειδιά παραγωγής και κωδικός επιβεβαίωσης

Τα κλειδιά παραγωγής των κωδικών είναι η πληροφορία η οποία πρέπει να διαμοιραστεί. Καθώς δεν μπορούμε να εξασφαλίσουμε την ιδιωτικότητα της πληροφορίας που, με οποιονδήποτε τρόπο, κοινοποιείται, έχει ληφθεί μέριμνα ούτως ώστε, χωρίς την εγκατεστημένη εφαρμογή, να μην είναι δυνατή η αντιστροφή της διαδικασίας και η αποκάλυψη των μυστικών κωδικών. Ακόμη και στην περίπτωση της διαρροής του κωδικού επιβεβαίωσης, τον οποίο σε συγκεκριμένες περιπτώσεις χρήσης είναι απαραίτητο να μοιραστούμε με τους υπόλοιπους χρήστες επίσης, ο επιτιθέμενος δεν θα μπορεί να κάνει τον συσχετισμό. Αυτό εξασφαλίζεται από τους πολύπλοκους υπολογισμούς που γίνονται για την δημιουργία του κωδικού επιβεβαίωσης και το πλήθος των μεταβλητών που λαμβάνουν μέρος στην διαδικασία. Όπως είδαμε και στις παραγράφους 6.1.5 και 6.2.6 υπάρχουν πέντε διαφορετικές μεταβλητές οι οποίες διαμορφώνουν το τελικό αποτέλεσμα και οι μαθηματικές πράξεις που γίνονται δεν αρκούνται στην πρόσθεσή τους αλλά υπολογίζονται γινόμενα και πηλίκα αυτών κάνοντας έτσι την διαδικασία αναστροφής της λειτουργίας (reverse engineering) πολύ πιο δύσκολη για τον κάθε επιτιθέμενο.

7.2 Τα EXE αρχεία

Τόσο το αρχείο εγκατάστασης όσο και το κυρίως αρχείο που δημιουργείται, έχουν μετατραπεί σε εκτελέσιμα (EXE) αρχεία με χρήση του Bat_To_Exe_Converter (Fatih Kodak, f2k0 Software, n.d.). Αυτό προστατεύει τον πηγαίο κώδικα από οποιαδήποτε ανεπιθύμητη πρόσβαση αποκλείοντας την πιθανότητα να καταφέρει κάποιος να διαπιστώσει την διαδικασία υπολογισμού των μεταβλητών και των υπόλοιπων ευαίσθητων πληροφοριών και έτσι να μπορέσει να επαναλάβει την διαδικασία και να αποκαλύψει τους κωδικούς.

Έχει γίνει ένας έλεγχος ασφαλείας σε αυτά τα αρχεία προκειμένου να διαπιστωθεί πόσο εύκολη είναι η αποκάλυψη του πηγαίου κώδικα. Στον έλεγχο χρησιμοποιήθηκε το πρόγραμμα Resource hacker (Angus Johnson, 03-01-2019) το οποίο έχει σχεδιαστεί να αποκρυπτογραφεί εκτελέσιμα αρχεία. Τόσο στην 32 όσο και στην 64 bit έκδοση των αρχείων της εφαρμογής, δεν κατέστη δυνατή η «αποκάλυψη» του πηγαίου κώδικα.

7.3 Εγκατάσταση

Η εγκατάσταση μπορούμε να πούμε ότι είναι η στιγμή με τα περισσότερα σημεία πιθανής ύπαρξης ευπαθειών. Το πρώτο και σημαντικότερο είναι ο φάκελος με τα βοηθητικά αρχεία και εργαλεία. Όπως είδαμε στην παράγραφο 6.1.2 αυτός ο φάκελος όχι απλά μετονομάζεται αλλά γίνεται κρυφός και φάκελος συστήματος με αποτέλεσμα να μην εμφανίζεται τόσο σε γραφικό περιβάλλον όσο και σε περιβάλλον γραμμής εντολών (dir command). Ο μόνος τρόπος πρόσβασης σε αυτόν λοιπόν είναι να γνωρίζει ο επιτιθέμενος, όχι την αρχική ονομασία του, αλλά το όνομα που δίνεται στον φάκελο από το πρόγραμμα εγκατάστασης. Και αυτό χωρίς την πρόσβαση στον πηγαίο κώδικα δεν μπορεί να γίνει.

Κατά την εγκατάσταση μας ζητείται η εισαγωγή της μυστικής φράσης. Καθώς η εφαρμογή έχει σχεδιαστεί για ελεύθερη χρήση με αποτέλεσμα αντίγραφο του αρχείου εγκατάστασης να είναι διαθέσιμο σε όλους, η διαρροή αυτής της πληροφορίας μπορεί να επιφέρει ολέθριες συνέπειες στην ασφάλεια της εφαρμογής. Έτσι, παρόλο που συνίσταται η χρήση σε ασφαλές περιβάλλον, έχει προστατευθεί η εισαγωγή της μυστικής φράσης με το πρόγραμμα Maskedinput (Fatih Kodak, f2k0 Software, n.d.).

Τέλος γίνεται χρήση της sha1 για τον υπολογισμό των μεταβλητών. Η sha1 όπως και οι περισσότερες hash functions έχουν ή μπορεί να αποκτήσουν οποιαδήποτε στιγμή στο μέλλον, κενά ασφαλείας. Έχοντας υπ' όψιν αυτό δημιουργήθηκε μια διαδικασία η οποία υπολογίζει δύο φορές το αποτέλεσμα της sha1, με την δεύτερη να περιλαμβάνει και πρόσθετη πληροφορία για το βοηθητικό αρχείο που χρησιμοποιείται σε μία μορφής salted sha1, και δεν ξαναγίνεται χρήση της κατά την λειτουργεία του κυρίως αρχείου.

7.4 Το Εκτελέσιμο αρχείο

Σε αντίθεση με το αρχείο εγκατάστασης, που έχει σχεδιαστεί για λιγότερο συχνή χρήση, το κάθε εκτελέσιμο αρχείο που δημιουργείται είναι μοναδικό και ο κώδικάς του προστατεύεται καθώς περιλαμβάνει τις τελικές τιμές των μεταβλητών σε μια προσπάθεια να μην γίνει χρήση βοηθητικών αρχείων που θα μπορούσαν να αποτελέσουν κίνδυνο για την ασφάλεια της εφαρμογής. Όσες φορές και αν το εκτελέσουμε λοιπόν δεν δημιουργούνται βοηθητικά αρχεία και δεν χρησιμοποιούνται εφαρμογές τρίτων. Η διαδικασία έχει γίνει αρκετά πολύπλοκη για αντιστροφή καθώς υπάρχουν πολλά σημεία που διαφοροποιούν την κάθε «έκδοσή» της. Από την αρχική θέση και την σειρά του array, μέχρι την σύνθεση του κωδικού επιβεβαίωσης στην τελική δημιουργία του μυστικού κωδικού, χωρίς πρόσβαση στον πηγαίο κώδικα η απόπειρα αντιστροφής της διαδικασίας θα ήθελε πολλαπλούς, πολύπλοκους και χρονοβόρους υπολογισμούς.

Από τα ανωτέρω γίνεται κατανοητό ότι μπορούμε με μια σχετική ευκολία να μιλήσουμε για μια ασφαλή στην χρήση της εφαρμογή. Η μόνη περίπτωση ευπάθειας είναι μια πιθανή καταγραφή των εντολών που εκτελούνται κατά την χρήση του αρχείου από την γραμμή εντολών. Σε κάθε περίπτωση η χρήση της εφαρμογής πρέπει να γίνεται σε ένα ασφαλές υπολογιστικό σύστημα. Αλλά ακόμη και στην πιθανότητα ή υποψία έκθεσης της εφαρμογής σε κακόβουλη ή μη αποδεκτή χρήση έχει δοθεί στον χρήστη η λύση της αλλαγής της εγκατάστασης της εφαρμογής με ένα πολύ απλό τρόπο και με αποτέλεσμα την παραγωγή μιας τελείως ξεχωριστής και μοναδικής «εκδοχής» η οποία θα απαιτούσε τον επανυπολογισμό όλων των μεταβλητών από τον οποιοδήποτε επιτιθέμενο.

8. ΣΥΜΠΕΡΑΣΜΑΤΑ

Όπως είδαμε λοιπόν, το πλήθος και η πολυπλοκότητα των κωδικών που χρησιμοποιούμε καθημερινά αυξάνεται συνεχώς κάνοντας την αποστήθιση αυτών, κυρίως από τον μέσο χρήστη, πολύ δύσκολη. Όσο περισσότερα πληροφοριακά και τηλεπικοινωνιακά συστήματα χρησιμοποιούμε, τόσο περισσότερο γίνεται απαραίτητη η χρήση κωδικών.

Η ανάπτυξη όμως της επιστήμης των υπολογιστών αυξάνει τις δυνατότητες των επιθέσεων κατά των κωδικών αυτών κάνοντας την τεχνολογία πιο προσιτή οικονομικά και με μεγαλύτερες δυνατότητες. Αυτός ακριβώς είναι και ο λόγος που επιτάσσεται οι κωδικοί μας να είναι αρκετά πολύπλοκοι και μεγάλοι δυσκολεύοντας ακόμη περισσότερο την χρήση τους.

Υπάρχουν πολλές επιλογές για να παράξουμε και να αποθηκεύσουμε με ασφάλεια ένα κωδικό. Πολλές εφαρμογές υπόσχονται ασφαλή αποθήκευση και διασφάλιση της ιδιωτικότητά του. Δεν υπάρχουν όμως πολλές επιλογές που να δίνουν λύση στην ανάγκη που δημιουργείται, κάποιες φορές, στο να μοιραστούμε ένα κωδικό με ασφάλεια. Η κρυπτογραφία έχει δώσει ορισμένες εφαρμογές οι οποίες προσπαθούν να λύσουν αυτό το πρόβλημα αλλά σε ένα περιβάλλον εγγενώς ανασφαλές, όπως το ίντερνετ, οι πληροφορίες δεν μπορούν ποτέ να θεωρηθούν τελείως ασφαλείς. Επιπλέον δεν συνίσταται η χρήση της κινητής τηλεφωνίας για να μοιραστούμε μια τόσο ευαίσθητη πληροφορία γιατί όχι μόνο έχει σχεδιαστεί για άμεση επικοινωνία μεταξύ λίγων ατόμων, με τις όποιες δυσκολίες και περιορισμούς μας δίνει αυτό, αλλά δεν είναι ένας τρόπος επικοινωνίας σχεδιασμένος να περιέχει μεγάλα επίπεδα ασφαλείας.

Τέλος υπάρχει και το θέμα της αποθήκευσης των κωδικών μας στην προσπάθειά μας να τους κοινοποιήσουμε. Όλες οι απόπειρες αποθήκευσης σε ηλεκτρονικά ή μη μέσα δημιουργεί άλλο ένα κενό ασφαλείας για τους κωδικούς μας και τους κάνει ευάλωτους σε επιθέσεις.

Όλα τα παραπάνω οδήγησαν στην δημιουργία μιας εφαρμογής η οποία μπορεί να χρησιμοποιηθεί με διαφορετικούς τρόπους ανάλογα με την ανάγκη την οποία καλείται να

αντιμετωπίσει. Από το κλείδωμα των αρχείων μας στην πρόσβαση σε συστήματα μέχρι το αυτόματο κλείδωμα εφαρμογών και την διαχείριση των κωδικών μας. Μπορούμε να παράξουμε τους κωδικούς μας με ασφάλεια, χωρίς να αποθηκεύονται πουθενά, αλλά και να τους γνωρίσουμε σε τρίτους απλά κοινοποιώντας μια πληροφορία που χωρίς την χρήση της εφαρμογής δεν μπορεί να οδηγήσει κάποιο επιτιθέμενο στην αποκάλυψη του κωδικού.

Για την χρήση της εφαρμογής χρειάζεται πρώτα μια εγκατάσταση με χρήση μιας μυστικής φράσης. Το μέγεθος της φράσης αυτής δεν έχει κανένα περιορισμό και προτείνεται η χρήση κάτι αρκετά πολύπλοκου όπως απόσπασμα από κάποιο βιβλίο ή κάτι παρόμοιο που θα μπορεί εύκολα ο χρήστης να ανακαλέσει στην μνήμη του αλλά ταυτόχρονα να είναι δύσκολο να γίνει αυτό από κάποια επίθεση. Η εγκατάσταση της εφαρμογής εξασφαλίζει διαφορετικότητα σε κάθε «έκδοσης» αυτής για χρήση με διαφορετικές ομάδες χρηστών ή και για λόγους ασφαλείας όταν κριθεί ότι χρειάζεται να γίνει κάποια αλλαγή.

Η εγκατάσταση μας δίνει ένα εκτελέσιμο αρχείο που μπορούμε να το χρησιμοποιήσουμε με δύο τρόπους. Μπορούμε να το καλέσουμε από γραμμή εντολών για μια πιο αυτοματοποιημένη χρήση ή να το εκτελέσουμε απλά και να εκμεταλλευτούμε το πιο φιλικό περιβάλλον προς τον χρήστη. Για την παραγωγή του κωδικού αρκεί η εισαγωγή ενός τετραψήφιου κλειδιού.

Τα αποτελέσματα που μας επιστρέφει σε κάθε περίπτωση είναι ο μυστικός κωδικός και ένας τετραψήφιος κωδικός επιβεβαίωσης για να μπορούμε να διαπιστώσουμε την σωστή και κοινή λειτουργία της εφαρμογής. Οι κωδικοί που παράγονται είναι σύμφωνοι με όλους τους κανόνες ασφαλείας καθώς

- Έχουν μήκος δεκαέξι χαρακτήρων κάνοντας τον συνολικό πιθανό αριθμό πολύ μεγάλο για να μπορεί μια επίθεση εξαντλητικής αναζήτησης να δημιουργήσει ευπάθεια
- Περιλαμβάνει ψηφία από ένα σύνολο ογδόντα εννέα διαφορετικών χαρακτήρων που μπορούν να κατηγοριοποιηθούν σε τέσσερις μεγάλες ομάδες, εξασφαλίζοντας έτσι μεγάλη πολυπλοκότητα.
- Η επιλογή των ψηφίων γίνεται με ένα τελείως τυχαίο τρόπο κάνοντας τους κωδικούς αδύνατο να πληγούν από μια πιθανή επίθεση λεξικού.

- Οι κωδικοί δεν αποθηκεύονται πουθενά αλλά παράγονται κάθε φορά που τους χρειαζόμαστε. Έτσι δεν υπάρχει πιθανότητα αποκάλυψης τους σε περίπτωση συμβάντος ασφαλείας στο αποθηκευτικό μέσο.
- Μπορεί να γίνει απλή αντιγραφή του κωδικού και επικόλληση όπου χρειάζεται να εισαχθεί, αποφεύγοντας έτσι τις κακόβουλες επιθέσεις καταγραφής της εισαγωγής χαρακτήρων (keylogger).

Η δημιουργία της εφαρμογής έγινε με γνώμονα τον τελικό χρήστη. Από τον εύκολο τρόπο λειτουργίας στην καθαρή και απλή παρουσίαση των αποτελεσμάτων μέχρι και στην επιλογή των χαρακτήρων που συνθέτουν τον κωδικό, ο χρήστης δεν χρειάζεται να έχει ιδιαίτερες γνώσεις για να μπορέσει να χρησιμοποιήσει την εφαρμογή. Επίσης όλα τα εργαλεία που χρησιμοποιήθηκαν διαθέτοντας δωρεάν για να μπορεί να δημιουργηθεί μια εφαρμογή η οποία δεν θα έχει κάποιο κόστος για τον τελικό χρήστη. Τέλος ακόμη και να μην υπάρχει η ανάγκη διαμοιρασμού κάποιου κωδικού, η εφαρμογή μπορεί να βοηθήσει τον χρήστη να θυμάται και να χρησιμοποιεί πολύπλοκους κωδικούς με το να συγκρατεί στην μνήμη του απλούς τετραψήφια κλειδιά.

Οι πολύπλοκες μαθηματικές πράξεις στον υπολογισμό των μεταβλητών και η χρήση της sha1 hash function μας εξασφαλίζουν ότι δεν μπορεί να γίνει αντιστροφή της διαδικασίας και αποκάλυψης των κωδικών μας.

Όπως βλέπουμε οι χρήσεις της εφαρμογής είναι πολλές. Ο κάθε χρήστης βάση των αναγκών του αλλά και του επιπέδου του μπορεί να βρει τον δικό του τρόπο να χρησιμοποιήσει την εφαρμογή και να παράξει ένα κωδικό που είναι σύμφωνος με όλους τους κανόνες ασφαλείας κάνοντας την Sharepass μια πολύ χρήσιμη εφαρμογή.

ΠΑΡΑΡΤΗΜΑ Α

Κώδικας αρχείου εγκατάστασης

```
1  :: κλείνουμε το echo για να μην εκτυπώνονται στην οθόνη οι εντολές που
2  εκτελούνται.
3  @echo off
4
5  color b
6
7  :: κάνουμε το αρχείο να τρέχει στο path που το έχουμε αποθηκεύσει.
8  @%~d0
9  @cd "%~p0"
10
11 :: Το χρησιμοποιούμε για να μπορούμε να έχουμε ειδικούς χαρακτήρες στις
12 μεταβλητές μας αλλά και για να δουλεύουν οι μεταβλητές μας και στις
13 εντολές επανάληψης.
14 setlocal EnabledelayedExpansion
15
16 :: μετονομαζουμε τον φακελο help σε utl. Για λόγους ασφαλείας σε
17 περίπτωση που καταφέρει κάποιος να δεί το αρχικό όνομα του φακέλου κατά
18 την εγκατάσταση.
19 ren help utl
20
21 :: Κάνει τον φάκελο που θα αποθηκευτούν τα βοηθητικά προγράμματα αλλά
22 και τα προσωρινά αρχεία κρυφό (+h) και φάκελο συστήματος (+s) για
23 μεγαλύτερη ασφάλεια.
24 attrib +h +s "utl" /s /d
25
26 :: σβήνουμε αν υπάρχουν τα βοηθητικά αρχεία για να τα δημιουργήσουμε απο
27 την αρχή και να μην υπάρξει προβλημα με παλαιά δεδομένα.
28 if exist utl\hfps.mik del utl\hfps.mik
29 if exist utl\hfps1.mik del utl\hfps1.mik
30 if exist utl\instps.mik del utl\instps.mik
31 if exist utl\alg.mik del utl\alg.mik
32
33 :: ζητάμε την αρχική ΠΡΌΤΑΣΗ ασφαλείας.
```

```
34  utl\maskedinput.exe "Give the install passphrase:" > utl\instps.mik
35
36  :: δείχνουμε την μπάρα προόδου της εγκατάστασης.
37  set /a cmp=0
38  set br==
39  cls
40  echo Instalation Prosses Completed %br% %cmp%%%
41
42  :: Υπολογίζουμε το μήκος της πρότασης ασφαλείας.
43  for %%? in (utl\instps.mik) do ( set /a len=%%~z? -2 )
44
45  :: Υπολογίζουμε την sha1 για την πρόταση ασφαλείας και την αποθηκεύουμε
46  σε ένα προσωρινό αρχείο με επέκταση που δεν αντιστοιχεί σε κάποια γνωστή
47  κατηγορία αρχείων.
48  for /f "tokens=* USEBACKQ" %%F in (`utl\fciv.exe -sha1 utl\instps.mik`)
49  do echo %%F> utl\hfps.mik
50
51  :: διαγράφουμε το προσωρινό αρχείο της πρότασης ασφαλείας.
52  del utl\instps.mik
53
54  :: Υπολογίζουμε την sha1 για το αποτέλεσμα της hash function για
55  μεγαλύτερη ασφάλεια.
56  for /f "tokens=* USEBACKQ" %%F in (`utl\fciv.exe -sha1 utl\hfps.mik`) do
57  echo %%F> utl\hfps1.mik
58
59  :: διαγράφουμε το προσωρινό αρχείο της πρώτης sha1.
60  del utl\hfps.mik
61
62  :: παίρνουμε το αποτέλεσμα σε μια μεταβλητή (το αρχείο περιέχει και
63  άλλες πληροφορίες εκτός απο το αποτέλεσμα).
64  for /f "tokens=1,*" %%A in (utl\hfps1.mik) do (
65      set hf=%%A
66  )
67
68  set /a cmp+=9
69  set br=====
70  cls
71  echo Instalation Prosses Completed %br% %cmp%%%
72
```

```
:: διαγράφουμε το προσωρινό αρχείο της δεύτερης sha1.
del utl\hfps1.mik

:: παίρνουμε ένα ένα τα ψηφία της sha1 και τα αποθηκεύουμε σε ένα array.
for /L %%A in (0,1,39) do (
    set ca[%%A]=!hf:~%%A,1!
)

:: Προσθέτουμε το γινόμενο της τιμής με την θέση στην οποία βρίσκεται
των ψηφίων που είναι νούμερα από το αποτέλεσμα της sha1. Αυτό μας δίνει
ένα αριθμό με μέγιστο το 7020.
:: Υπολογίζουμε επίσης το άθροισμα των θέσεων που έχουν νούμερα (εκτός
του 0) και αυτών που έχουν γράμματα και 0.
set /a count=0
set /a cthn=0
set /a cthg=0
for /L %%A in (0,1,39) do (
    set /a ts=!ca[%%A]!
    if !ts! neq 0 (
        set /a count+=!ca[%%A]!*%%A
        set /a cthn+=%%A
    ) else (
        set /a cthg+=%%A
    )
)

set /a count=!count!-!len!
set /a cthn=!cthn!+!len!
set /a cthg=!cthg!+!len!

:: Υπολογίζουμε την διαφορά των δύο μετρητών θέσεων.
if !cthn! gtr !cthg! (
    set /a cthd=!cthn!-!cthg!
) else (
    set /a cthd=!cthg!-!cthn!
)

:: Υπολογίζουμε την μεταβλητή αρχής του array.
set /a arst=!count!/!len!
```

```
112  set /a hlp=!cthg!/!cthd!
113  set /a arst=!arst!+!hlp!
114  set /a hlp=!cthn!/!cthd!
115  set /a arst=!arst!+!hlp!
116
117  set /a cmp+=9
118  set br========
119  cls
120  echo Instalation Prosses Completed %br% %cmp%%%
121
122  :recheckarst
123  if !arst! geq 100 (
124      set /a arst=!arst!/2
125      goto recheckarst
126  )
127
128  :: Φτιάχνουμε τον κώδικα
129  echo @echo off >> utl\alg.mik
130  echo color b >> utl\alg.mik
131  echo setlocal EnableDelayedExpansion >> utl\alg.mik
132  echo set /a check^=0 >> utl\alg.mik
133  echo set /a n^=%arst% >> utl\alg.mik
134  echo if "%%~1"=="help" ( >> utl\alg.mik
135  echo     echo ----------------------------- >> utl\alg.mik
136  echo     echo :                           : >> utl\alg.mik
137  echo     echo :       sharepass.exe       : >> utl\alg.mik
138  echo     echo :          v 1.0            : >> utl\alg.mik
139  echo     echo :     2019 Mike Gkoumas     : >> utl\alg.mik
140  echo     echo :                           : >> utl\alg.mik
141  echo     echo ----------------------------- >> utl\alg.mik
142  echo     echo. >> utl\alg.mik
143  echo     echo. >> utl\alg.mik
144  echo     echo This script produces a safe password >> utl\alg.mik
145  echo     echo when given a four digit number. >> utl\alg.mik
146  echo     echo It also returns a four digit check number. >> utl\alg.mik
147  echo     echo. >> utl\alg.mik
148  echo     echo Run it as an application and follow >> utl\alg.mik
149  echo     echo the instructions on the screen. >> utl\alg.mik
150  echo     echo. >> utl\alg.mik
```

```
151 echo     echo For command line and batch file usage type >> utl\alg.mik
152 echo     echo sharepass.exe [key] >> utl\alg.mik
153 echo     echo where [key] is the four digit number >> utl\alg.mik
154 echo     echo and the password with the check number >> utl\alg.mik
155 echo     echo will be printed separated by a whitespace >> utl\alg.mik
156 echo     echo. >> utl\alg.mik
157 echo     echo This application was created with the use >> utl\alg.mik
158 echo     echo of a passphrase during the installation. >> utl\alg.mik
159 echo     echo Only installations with the same passphrase will >>
160 utl\alg.mik
161 echo     echo generate the same passwords and check numbers. >>
162 utl\alg.mik
163 echo     echo. >> utl\alg.mik
164 echo     pause >> utl\alg.mik
165 echo     exit >> utl\alg.mik
166 echo ) >> utl\alg.mik
167 echo if not "%%~1"=="" set /a ena=%%1 >> utl\alg.mik
168 echo if "%%~1"=="" set ena=notthere >> utl\alg.mik
169 echo set "ar[%%n%%]=@">> utl\alg.mik
170 echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
171 echo set /a n+^=1 >> utl\alg.mik
172 echo set "ar[%%n%%]=/">> utl\alg.mik
173 echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
174 echo set /a n+^=1 >> utl\alg.mik
175 echo set "ar[%%n%%]=~">> utl\alg.mik
176 echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
177 echo set /a n+^=1 >> utl\alg.mik
178 echo set "ar[%%n%%]=?">> utl\alg.mik
179 echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
180 echo set /a n+^=1 >> utl\alg.mik
181 echo set "ar[%%n%%]=^>">> utl\alg.mik
182 echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
183 echo set /a n+^=1 >> utl\alg.mik
184 echo set "ar[%%n%%]=.">> utl\alg.mik
185 echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
186 echo set /a n+^=1 >> utl\alg.mik
187 echo set "ar[%%n%%]=8">> utl\alg.mik
188 echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
189 echo set /a n+^=1 >> utl\alg.mik
```

```
190  echo set "ar[%%n%%]=7">> utl\alg.mik
191  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
192  echo set /a n+^=1 >> utl\alg.mik
193  echo set "ar[%%n%%]=4">> utl\alg.mik
194  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
195  echo set /a n+^=1 >> utl\alg.mik
196  echo set "ar[%%n%%]=;">> utl\alg.mik
197  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
198  echo set /a n+^=1 >> utl\alg.mik
199  echo set "ar[%%n%%]=O">> utl\alg.mik
200  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
201  echo set /a n+^=1 >> utl\alg.mik
202  echo set "ar[%%n%%]=z">> utl\alg.mik
203
204  set /a cmp+=9
205  set br===========
206  cls
207  echo Instalation Prosses Completed %br% %cmp%%
208
209  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
210  echo set /a n+^=1 >> utl\alg.mik
211  echo set "ar[%%n%%]=[">> utl\alg.mik
212  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
213  echo set /a n+^=1 >> utl\alg.mik
214  echo set "ar[%%n%%]=}">> utl\alg.mik
215  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
216  echo set /a n+^=1 >> utl\alg.mik
217  echo set "ar[%%n%%]=d">> utl\alg.mik
218  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
219  echo set /a n+^=1 >> utl\alg.mik
220  echo set "ar[%%n%%]=@">> utl\alg.mik
221  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
222  echo set /a n+^=1 >> utl\alg.mik
223  echo set "ar[%%n%%]=R">> utl\alg.mik
224  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
225  echo set /a n+^=1 >> utl\alg.mik
226  echo set "ar[%%n%%]=8">> utl\alg.mik
227  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
228  echo set /a n+^=1 >> utl\alg.mik
```

```
229  echo set "ar[%%n%%]=m">> utl\alg.mik
230  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
231  echo set /a n+^=1 >> utl\alg.mik
232  echo set "ar[%%n%%]=0">> utl\alg.mik
233  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
234  echo set /a n+^=1 >> utl\alg.mik
235  echo set "ar[%%n%%]=k">> utl\alg.mik
236  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
237  echo set /a n+^=1 >> utl\alg.mik
238  echo set "ar[%%n%%]=2">> utl\alg.mik
239  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
240  echo set /a n+^=1 >> utl\alg.mik
241  echo set "ar[%%n%%]=y">> utl\alg.mik
242  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
243  echo set /a n+^=1 >> utl\alg.mik
244  echo set "ar[%%n%%]=Z">> utl\alg.mik
245  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
246  echo set /a n+^=1 >> utl\alg.mik
247  echo set "ar[%%n%%]=v">> utl\alg.mik
248  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
249  echo set /a n+^=1 >> utl\alg.mik
250  echo set "ar[%%n%%]=5">> utl\alg.mik
251  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
252  echo set /a n+^=1 >> utl\alg.mik
253  echo set "ar[%%n%%]=r">> utl\alg.mik
254
255  set /a cmp+=9
256  set br==============
257  cls
258  echo Instalation Prosses Completed %br% %cmp%%
259
260  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
261  echo set /a n+^=1 >> utl\alg.mik
262  echo set "ar[%%n%%]=^<">> utl\alg.mik
263  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
264  echo set /a n+^=1 >> utl\alg.mik
265  echo set "ar[%%n%%]=n">> utl\alg.mik
266  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
267  echo set /a n+^=1 >> utl\alg.mik
```

```
268  echo set "ar[%%n%%]=@">> utl\alg.mik
269  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
270  echo set /a n+^=1 >> utl\alg.mik
271  echo set "ar[%%n%%]=M">> utl\alg.mik
272  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
273  echo set /a n+^=1 >> utl\alg.mik
274  echo set "ar[%%n%%]=t">> utl\alg.mik
275  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
276  echo set /a n+^=1 >> utl\alg.mik
277  echo set "ar[%%n%%]=b">> utl\alg.mik
278  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
279  echo set /a n+^=1 >> utl\alg.mik
280  echo set "ar[%%n%%]=+">> utl\alg.mik
281  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
282  echo set /a n+^=1 >> utl\alg.mik
283  echo set "ar[%%n%%]={">> utl\alg.mik
284  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
285  echo set /a n+^=1 >> utl\alg.mik
286  echo set "ar[%%n%%]=-">> utl\alg.mik
287  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
288  echo set /a n+^=1 >> utl\alg.mik
289  echo set "ar[%%n%%]=U">> utl\alg.mik
290  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
291  echo set /a n+^=1 >> utl\alg.mik
292  echo set "ar[%%n%%]=&">> utl\alg.mik
293  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
294  echo set /a n+^=1 >> utl\alg.mik
295  echo set "ar[%%n%%]=o">> utl\alg.mik
296  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
297  echo set /a n+^=1 >> utl\alg.mik
298  echo set "ar[%%n%%]=#">> utl\alg.mik
299  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
300  echo set /a n+^=1 >> utl\alg.mik
301  echo set "ar[%%n%%]=L">> utl\alg.mik
302
303  set /a cmp+=9
304  set br=================
305  cls
306  echo Instalation Prosses Completed %br% %cmp%%
```

```
307
308  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
309  echo set /a n+^=1 >> utl\alg.mik
310  echo set "ar[%%n%%]=^">> utl\alg.mik
311  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
312  echo set /a n+^=1 >> utl\alg.mik
313  echo set "ar[%%n%%]=(">> utl\alg.mik
314  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
315  echo set /a n+^=1 >> utl\alg.mik
316  echo set "ar[%%n%%]=3">> utl\alg.mik
317  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
318  echo set /a n+^=1 >> utl\alg.mik
319  echo set "ar[%%n%%]=x">> utl\alg.mik
320  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
321  echo set /a n+^=1 >> utl\alg.mik
322  echo set "ar[%%n%%]=1">> utl\alg.mik
323  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
324  echo set /a n+^=1 >> utl\alg.mik
325  echo set "ar[%%n%%]=C">> utl\alg.mik
326  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
327  echo set /a n+^=1 >> utl\alg.mik
328  echo set "ar[%%n%%]=:">> utl\alg.mik
329  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
330  echo set /a n+^=1 >> utl\alg.mik
331  echo set "ar[%%n%%]=f">> utl\alg.mik
332  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
333  echo set /a n+^=1 >> utl\alg.mik
334  echo set "ar[%%n%%]=H">> utl\alg.mik
335  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
336  echo set /a n+^=1 >> utl\alg.mik
337  echo set "ar[%%n%%]=E">> utl\alg.mik
338  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
339  echo set /a n+^=1 >> utl\alg.mik
340  echo set "ar[%%n%%]=W">> utl\alg.mik
341  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
342  echo set /a n+^=1 >> utl\alg.mik
343
344  set /a cmp+=9
345  set br====================
```

```
346  cls
347  echo Instalation Prosses Completed %br% %cmp%%%
348
349  echo set "ar[%%n%%]=="">> utl\alg.mik
350  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
351  echo set /a n+^=1 >> utl\alg.mik
352  echo set "ar[%%n%%]=Y">> utl\alg.mik
353  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
354  echo set /a n+^=1 >> utl\alg.mik
355  echo set "ar[%%n%%]=1">> utl\alg.mik
356  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
357  echo set /a n+^=1 >> utl\alg.mik
358  echo set "ar[%%n%%]=e">> utl\alg.mik
359  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
360  echo set /a n+^=1 >> utl\alg.mik
361  echo set "ar[%%n%%]=5">> utl\alg.mik
362  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
363  echo set /a n+^=1 >> utl\alg.mik
364  echo set "ar[%%n%%]=q">> utl\alg.mik
365  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
366  echo set /a n+^=1 >> utl\alg.mik
367  echo set "ar[%%n%%]=9">> utl\alg.mik
368  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
369  echo set /a n+^=1 >> utl\alg.mik
370  echo set "ar[%%n%%]=7">> utl\alg.mik
371  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
372  echo set /a n+^=1 >> utl\alg.mik
373  echo set "ar[%%n%%]=6">> utl\alg.mik
374  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
375  echo set /a n+^=1 >> utl\alg.mik
376  echo set "ar[%%n%%]=3">> utl\alg.mik
377  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
378  echo set /a n+^=1 >> utl\alg.mik
379  echo set "ar[%%n%%]=\">> utl\alg.mik
380  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
381  echo set /a n+^=1 >> utl\alg.mik
382  echo set "ar[%%n%%]=A">> utl\alg.mik
383  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
384  echo set /a n+^=1 >> utl\alg.mik
```

```
385  echo set "ar[%%n%%]=]">> utl\alg.mik
386  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
387  echo set /a n+^=1 >> utl\alg.mik
388  echo set "ar[%%n%%]=u">> utl\alg.mik
389  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
390  echo set /a n+^=1 >> utl\alg.mik
391  echo set "ar[%%n%%]=6">> utl\alg.mik
392  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
393  echo set /a n+^=1 >> utl\alg.mik
394  echo set "ar[%%n%%]=T">> utl\alg.mik
395
396  set /a cmp+=9
397  set br=======================
398  cls
399  echo Instalation Prosses Completed %br% %cmp%%%
400
401  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
402  echo set /a n+^=1 >> utl\alg.mik
403  echo set "ar[%%n%%]=i">> utl\alg.mik
404  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
405  echo set /a n+^=1 >> utl\alg.mik
406  echo set "ar[%%n%%]=h">> utl\alg.mik
407  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
408  echo set /a n+^=1 >> utl\alg.mik
409  echo set "ar[%%n%%]=S">> utl\alg.mik
410  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
411  echo set /a n+^=1 >> utl\alg.mik
412  echo set "ar[%%n%%]=p">> utl\alg.mik
413  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
414  echo set /a n+^=1 >> utl\alg.mik
415  echo set "ar[%%n%%]=s">> utl\alg.mik
416  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
417  echo set /a n+^=1 >> utl\alg.mik
418  echo set "ar[%%n%%]=c">> utl\alg.mik
419  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
420  echo set /a n+^=1 >> utl\alg.mik
421  echo set "ar[%%n%%]=w">> utl\alg.mik
422  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
423  echo set /a n+^=1 >> utl\alg.mik
```

```
424  echo set "ar[%%n%%]=a">> utl\alg.mik
425  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
426  echo set /a n+^=1 >> utl\alg.mik
427  echo set "ar[%%n%%]=)">> utl\alg.mik
428  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
429  echo set /a n+^=1 >> utl\alg.mik
430  echo set "ar[%%n%%]=D">> utl\alg.mik
431  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
432  echo set /a n+^=1 >> utl\alg.mik
433  echo set "ar[%%n%%]=|">> utl\alg.mik
434  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
435  echo set /a n+^=1 >> utl\alg.mik
436  echo set "ar[%%n%%]=0">> utl\alg.mik
437  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
438  echo set /a n+^=1 >> utl\alg.mik
439  echo set "ar[%%n%%]=9">> utl\alg.mik
440  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
441  echo set /a n+^=1 >> utl\alg.mik
442  echo set "ar[%%n%%]=,">> utl\alg.mik
443  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
444  echo set /a n+^=1 >> utl\alg.mik
445  echo set "ar[%%n%%]=%%%%">> utl\alg.mik
446
447  set /a cmp+=9
448  set br==============================
449  cls
450  echo Instalation Prosses Completed %br% %cmp%%%
451
452  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
453  echo set /a n+^=1 >> utl\alg.mik
454  echo set "ar[%%n%%]=*">> utl\alg.mik
455  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
456  echo set /a n+^=1 >> utl\alg.mik
457  echo set "ar[%%n%%]=4">> utl\alg.mik
458  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
459  echo set /a n+^=1 >> utl\alg.mik
460  echo set "ar[%%n%%]=_">> utl\alg.mik
461  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
462  echo set /a n+^=1 >> utl\alg.mik
```

```
463  echo set "ar[%%n%%]=2">> utl\alg.mik
464  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
465  echo set /a n+^=1 >> utl\alg.mik
466  echo set "ar[%%n%%]=X">> utl\alg.mik
467  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
468  echo set /a n+^=1 >> utl\alg.mik
469  echo set "ar[%%n%%]=j">> utl\alg.mik
470  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
471  echo set /a n+^=1 >> utl\alg.mik
472  echo set "ar[%%n%%]=N">> utl\alg.mik
473  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
474  echo set /a n+^=1 >> utl\alg.mik
475  echo set "ar[%%n%%]=P">> utl\alg.mik
476  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
477  echo set /a n+^=1 >> utl\alg.mik
478  echo set "ar[%%n%%]=Q">> utl\alg.mik
479  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
480  echo set /a n+^=1 >> utl\alg.mik
481  echo set "ar[%%n%%]=g">> utl\alg.mik
482  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
483  echo set /a n+^=1 >> utl\alg.mik
484  echo set "ar[%%n%%]=J">> utl\alg.mik
485  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
486  echo set /a n+^=1 >> utl\alg.mik
487  echo set "ar[%%n%%]=F">> utl\alg.mik
488  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
489  echo set /a n+^=1 >> utl\alg.mik
490  echo set "ar[%%n%%]=V">> utl\alg.mik
491  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
492  echo set /a n+^=1 >> utl\alg.mik
493  echo set "ar[%%n%%]=B">> utl\alg.mik
494  echo if %%n%% equ 99 set /a n^=-1 >> utl\alg.mik
495  echo set /a n+^=1 >> utl\alg.mik
496
497  set /a cmp+=9
498  set br==================================
499  cls
500  echo Instalation Prosses Completed %br% %cmp%%%
501
```

```
echo :redo >> utl\alg.mik
echo set /a chk=0 >> utl\alg.mik
echo if %%ena%%==notthere ( >> utl\alg.mik
echo  set /p give="Enter the given passkey: " >> utl\alg.mik
echo  set /a "strt=^!give^!" >> utl\alg.mik
echo ) >> utl\alg.mik
echo if not %%ena%%==notthere ( >> utl\alg.mik
echo  set /a strt=%%ena%% >> utl\alg.mik
echo ) >> utl\alg.mik
echo if %%strt%% leq 999 ( >> utl\alg.mik
echo  set /a chk=1 >> utl\alg.mik
echo  if not %%ena%%==notthere set /a chk=3 >> utl\alg.mik
echo ) >> utl\alg.mik
echo if %%strt%% geq 10000 ( >> utl\alg.mik
echo  set /a chk=1 >> utl\alg.mik
echo  if not %%ena%%==notthere set /a chk=3 >> utl\alg.mik
echo ) >> utl\alg.mik
echo if %%chk%%==1 ( >> utl\alg.mik
echo  cls >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo  echo wrong passkey >> utl\alg.mik
echo  echo please try again >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo  goto redo >> utl\alg.mik
echo ) >> utl\alg.mik
echo if %%chk%%==3 exit >> utl\alg.mik
echo set /a strt^=%%strt%%+%count% >> utl\alg.mik
echo if %%strt%% geq 10000 ^( set /a strt^=%%strt%%/10 ^) >> utl\alg.mik
echo set /a fst^=%%strt%%/1000 >> utl\alg.mik
echo set /a sub^=%%fst%%*1000 >> utl\alg.mik
echo set /a rem^=%%strt%%-%%sub%% >> utl\alg.mik
echo set /a snd^=%%rem%%/100 >> utl\alg.mik
echo set /a sub^=%%snd%%*100 >> utl\alg.mik
echo set /a rem^=%%rem%%-%%sub%% >> utl\alg.mik
echo set /a trd^=%%rem%%/10 >> utl\alg.mik
echo set /a sub^=%%trd%%*10 >> utl\alg.mik
echo set /a frt^=%%rem%%-%%sub%% >> utl\alg.mik
echo set /a sm^=%%fst%%+%%snd%%+%%trd%%+%%frt%% >> utl\alg.mik
echo set /a mpa^=%cthn%+%%sm%% >> utl\alg.mik
```

```
541  echo set /a mpb^=%cthg%+%%sm%% >> utl\alg.mik
542  echo set /a mpc^=%cthd%+%%sm%% >> utl\alg.mik
543  echo set /a mpd^=%%n%%+%%sm%% >> utl\alg.mik
544  echo set /a smaa^=%%frt%%*%%mpa%% >> utl\alg.mik
545  echo set /a smbb^=%%trd%%*%%mpb%% >> utl\alg.mik
546  echo set /a smcc^=%%snd%%*%%mpc%% >> utl\alg.mik
547  echo set /a smdd^=%%fst%%*%%mpd%% >> utl\alg.mik
548
549  set /a cmp+=9
550  set br=====================================
551  cls
552  echo Instalation Prosses Completed %br% %cmp%%%
553
554  echo set /a check^=%%smaa%%+%%smbb%%+%%smcc%%+%%smdd%%+%count% >>
555  utl\alg.mik
556  echo :recalc >> utl\alg.mik
557  echo if %%check%% geq 10000 ^( >> utl\alg.mik
558  echo  set /a check^=%%check%%/10 >> utl\alg.mik
559  echo  goto recalc >> utl\alg.mik
560  echo ^) >> utl\alg.mik
561  echo set /a fstc^=%%check%%/1000 >> utl\alg.mik
562  echo set /a sub^=%%fstc%%*1000 >> utl\alg.mik
563  echo set /a rem^=%%check%%-%%sub%% >> utl\alg.mik
564  echo set /a sndc^=%%rem%%/100 >> utl\alg.mik
565  echo set /a sub^=%%sndc%%*100 >> utl\alg.mik
566  echo set /a rem^=%%rem%%-%%sub%% >> utl\alg.mik
567  echo set /a trdc^=%%rem%%/10 >> utl\alg.mik
568  echo set /a sub^=%%trdc%%*10 >> utl\alg.mik
569  echo set /a frtc^=%%rem%%-%%sub%% >> utl\alg.mik
570  echo set /a sub^=%%fst%%*10 >> utl\alg.mik
571  echo set /a a^=%%sub%%+%%fstc%% >> utl\alg.mik
572  echo set /a e^=%%sub%%+%%frtc%% >> utl\alg.mik
573  echo set /a sub^=%%snd%%*10 >> utl\alg.mik
574  echo set /a b^=%%sub%%+%%sndc%% >> utl\alg.mik
575  echo set /a f^=%%sub%%+%%trdc%% >> utl\alg.mik
576  echo set /a sub^=%%trd%%*10 >> utl\alg.mik
577  echo set /a c^=%%sub%%+%%trdc%% >> utl\alg.mik
578  echo set /a g^=%%sub%%+%%sndc%% >> utl\alg.mik
579  echo set /a sub^=%%frt%%*10 >> utl\alg.mik
```

```
echo set /a d^=%%sub%%+%%fstc%% >> utl\alg.mik
echo set /a h^=%%sub%%+%%frtc%% >> utl\alg.mik
echo set /a sub^=%%trdc%%*10 >> utl\alg.mik
echo set /a i^=%%sub%%+%%fst%% >> utl\alg.mik
echo set /a p^=%%sub%%+%%frt%% >> utl\alg.mik
echo set /a sub^=%%frtc%%*10 >> utl\alg.mik
echo set /a j^=%%sub%%+%%snd%% >> utl\alg.mik
echo set /a o^=%%sub%%+%%trd%% >> utl\alg.mik
echo set /a sub^=%%fstc%%*10 >> utl\alg.mik
echo set /a k^=%%sub%%+%%trd%% >> utl\alg.mik
echo set /a n^=%%sub%%+%%snd%% >> utl\alg.mik
echo set /a sub^=%%sndc%%*10 >> utl\alg.mik
echo set /a m^=%%sub%%+%%fst%% >> utl\alg.mik
echo set /a l^=%%sub%%+%%frt%% >> utl\alg.mik
echo set
"ps=^!ar[%%a%%]^!^!ar[%%b%%]^!^!ar[%%c%%]^!^!ar[%%d%%]^!^!ar[%%e%%]^!^!a
r[%%f%%]^!^!ar[%%g%%]^!^!ar[%%h%%]^!^!ar[%%i%%]^!^!ar[%%j%%]^!^!ar[%%k%%
]^!^!ar[%%l%%]^!^!ar[%%m%%]^!^!ar[%%n%%]^!^!ar[%%o%%]^!^!ar[%%p%%]^!">>
utl\alg.mik
echo cls >> utl\alg.mik
echo color a >> utl\alg.mik
echo if %%ena%%==notthere ( >> utl\alg.mik
echo  cls >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo  echo Password Generator >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo  echo for the given passkey >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo  echo the PASSWORD is >> utl\alg.mik
set "th=echo ^!ps^!"
echo !th! >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo  echo and the CHECK NUMBER is >> utl\alg.mik
echo  echo %%check%% >> utl\alg.mik
echo  echo. >> utl\alg.mik
echo ) >> utl\alg.mik
echo if not %%ena%%==notthere ( >> utl\alg.mik
set "th=echo ^!ps^! %%check%%"
```

```
619  echo !th! >> utl\alg.mik
620  echo ) >> utl\alg.mik
621
622  set /a cmp+=9
623  set br=======================================
624  cls
625  echo Instalation Prosses Completed %br% %cmp%%%
626
627  echo set ps^=0 >> utl\alg.mik
628  echo set /a check^=0 >> utl\alg.mik
629  echo set a^=0 >> utl\alg.mik
630  echo set b^=0 >> utl\alg.mik
631  echo set c^=0 >> utl\alg.mik
632  echo set d^=0 >> utl\alg.mik
633  echo set e^=0 >> utl\alg.mik
634  echo set f^=0 >> utl\alg.mik
635  echo set g^=0 >> utl\alg.mik
636  echo set h^=0 >> utl\alg.mik
637  echo set i^=0 >> utl\alg.mik
638  echo set g^=0 >> utl\alg.mik
639  echo set k^=0 >> utl\alg.mik
640  echo set l^=0 >> utl\alg.mik
641  echo set m^=0 >> utl\alg.mik
642  echo set n^=0 >> utl\alg.mik
643  echo set o^=0 >> utl\alg.mik
644  echo set p^=0 >> utl\alg.mik
645  echo set /a sub^=0 >> utl\alg.mik
646  echo set /a rem^=0 >> utl\alg.mik
647  echo set /a mpa^=0 >> utl\alg.mik
648  echo set /a mpb^=0 >> utl\alg.mik
649  echo set /a mpc^=0 >> utl\alg.mik
650  echo set /a mpd^=0 >> utl\alg.mik
651  echo set /a smaa^=0 >> utl\alg.mik
652  echo set /a smab^=0 >> utl\alg.mik
653  echo set /a smac^=0 >> utl\alg.mik
654  echo set /a smad^=0 >> utl\alg.mik
655  echo set /a fst^=0 >> utl\alg.mik
656  echo set /a snd^=0 >> utl\alg.mik
657  echo set /a trd^=0 >> utl\alg.mik
```

```
658 echo set /a frt^=0 >> utl\alg.mik
659 echo set /a fstc^=0 >> utl\alg.mik
660 echo set /a sndc^=0 >> utl\alg.mik
661 echo set /a trdc^=0 >> utl\alg.mik
662 echo set /a frtc^=0 >> utl\alg.mik
663 echo set /a strt^=0 >> utl\alg.mik
664 echo set give^=0 >> utl\alg.mik
665 echo set /a chk=0 >> utl\alg.mik
666 echo set give=0 >> utl\alg.mik
667 echo for /L %%%%A in ^(0^,1^,39^) do ^( >> utl\alg.mik
668 echo   set ar[%%%%A]^=0 >> utl\alg.mik
669 echo ^) >> utl\alg.mik
670 echo if %%ena%%==notthere ( >> utl\alg.mik
671 echo  set /a ena=0 >> utl\alg.mik
672 echo  timeout /T 30 >> utl\alg.mik
673 echo ) >> utl\alg.mik
674 echo set /a ena=0 >> utl\alg.mik
675
676 :: φτιάχνουμε το εκτελέσιμο αρχείο.
677 utl\Bat_To_Exe.exe /bat utl\alg.mik /exe sharepass.exe /icon utl/key.ico
678 >nul
679 cls
680
681 set /a cmp=100
682 set br=========================================
683 cls
684 echo Instalation Prosses Completed %br% %cmp%%%
685
686 :: Σβήνουμε το αρχείο του κώδικα.
687 del utl\alg.mik
688
689 :: Σβήνουμε τις μεταβλητές
690 set /a inst_pass=0
691 set /a len=0
692 set /a hf=0
693 set /a ts=0
694 set /a hlp=0
695 set /a arst=0
696 for /L %%A in (0,1,39) do (
```

```
697         set ca[%%A]=0
698 )
699 set /a count=0
700 set /a cthn=0
701 set /a cthg=0
702
703 :: σβήνουμε τον βοηθητικό φάκελο και όλα τα αρχεία του.
704 if exist utl\*.mik del /Q /F utl\*.mik
705 timeout /T 3 /NOBREAK >nul
706 del /Q /F utl\Bat_To_Exe.exe
707 rd /S /Q utl\
708
709 cls
710 echo Installation completed successfully
711 echo Please run SHAREPASS.EXE to generate your passwords
712 echo.
713 echo For more information run "SHAREPASS.EXE help"
714
715 timeout /T 3 /NOBREAK >nul
```

ΠΑΡΑΡΤΗΜΑ Β

Κώδικας κυρίως αρχείου

```
1  :: κλείνουμε το echo για να μην εκτυπώνονται στην οθόνη οι εντολές που
2  εκτελούνται.
3  @echo off
4  color b
5
6  :: Το χρησιμοποιούμε για να μπορούμε να έχουμε ειδικούς χαρακτήρες στις
7  μεταβλητές μας αλλά και για να δουλεύουν οι μεταβλητές μας και στις
8  εντολές επανάληψης.
9  setlocal EnableDelayedExpansion
10
11 :: Δίνουμε τις αρχικές τιμές στις μεταβλητές μας.
12 set /a check=0
13 set /a n=%ARST%
14
15 :: Βοηθητικό μήνυμα.
16 if "%~1"=="help" (
17     echo ----------------------------
18     echo :                          :
19     echo :       sharepass.exe      :
20     echo :          v 1.0           :
21     echo :     2019 Mike Gkoumas    :
22     echo :                          :
23     echo ----------------------------
24     echo.
25     echo.
26     echo This script produces a safe password
27     echo when given a four digit number.
28     echo It also returns a four digit check number.
29     echo.
```

```
30      echo Run it as an application and follow
31      echo the instructions on the screen.
32      echo.
33      echo For command line and batch file usage type
34      echo sharepass.exe [key]
35      echo where [key] is the four digit number
36      echo and the password with the check number
37      echo will be printed separated by a whitespace
38      echo.
39      echo This application was created with the use
40      echo of a passphrase during the installation.
41      echo Only installations with the same passphrase will
42      echo generate the same passwords and check numbers.
43      echo.
44      pause
45      exit
46  )
47  if not "%~1"=="" set /a ena=%1
48  if "%~1"=="" set ena=notthere
49  :: Γεμίζουμε το array των χαρακτήρων.
50  set "ar[%n%]=@"
51  if %n% equ 99 set /a n=-1
52  set /a n+=1
53  set "ar[%n%]=/"
54  if %n% equ 99 set /a n=-1
55  set /a n+=1
56  set "ar[%n%]=~"
57  if %n% equ 99 set /a n=-1
58  set /a n+=1
59  set "ar[%n%]=?"
60  if %n% equ 99 set /a n=-1
61  set /a n+=1
62  set "ar[%n%]=>"
```

```
63  if %n% equ 99 set /a n=-1
64  set /a n+=1
65  set "ar[%n%]=."
66  if %n% equ 99 set /a n=-1
67  set /a n+=1
68  set "ar[%n%]=8"
69  if %n% equ 99 set /a n=-1
70  set /a n+=1
71  set "ar[%n%]=7"
72  if %n% equ 99 set /a n=-1
73  set /a n+=1
74  set "ar[%n%]=4"
75  if %n% equ 99 set /a n=-1
76  set /a n+=1
77  set "ar[%n%]=;"
78  if %n% equ 99 set /a n=-1
79  set /a n+=1
80  set "ar[%n%]=O"
81  if %n% equ 99 set /a n=-1
82  set /a n+=1
83  set "ar[%n%]=z"
84  if %n% equ 99 set /a n=-1
85  set /a n+=1
86  set "ar[%n%]=["
87  if %n% equ 99 set /a n=-1
88  set /a n+=1
89  set "ar[%n%]=}"
90  if %n% equ 99 set /a n=-1
91  set /a n+=1
92  set "ar[%n%]=d"
93  if %n% equ 99 set /a n=-1
94  set /a n+=1
95  set "ar[%n%]=@"
```

```
96  if %n% equ 99 set /a n=-1
97  set /a n+=1
98  set "ar[%n%]=R"
99  if %n% equ 99 set /a n=-1
100 set /a n+=1
101 set "ar[%n%]=8"
102 if %n% equ 99 set /a n=-1
103 set /a n+=1
104 set "ar[%n%]=m"
105 if %n% equ 99 set /a n=-1
106 set /a n+=1
107 set "ar[%n%]=0"
108 if %n% equ 99 set /a n=-1
109 set /a n+=1
110 set "ar[%n%]=k"
111 if %n% equ 99 set /a n=-1
112 set /a n+=1
113 set "ar[%n%]=2"
114 if %n% equ 99 set /a n=-1
115 set /a n+=1
116 set "ar[%n%]=y"
117 if %n% equ 99 set /a n=-1
118 set /a n+=1
119 set "ar[%n%]=Z"
120 if %n% equ 99 set /a n=-1
121 set /a n+=1
122 set "ar[%n%]=v"
123 if %n% equ 99 set /a n=-1
124 set /a n+=1
125 set "ar[%n%]=5"
126 if %n% equ 99 set /a n=-1
127 set /a n+=1
128 set "ar[%n%]=r"
```

```
129 if %n% equ 99 set /a n=-1
130 set /a n+=1
131 set "ar[%n%]=<"
132 if %n% equ 99 set /a n=-1
133 set /a n+=1
134 set "ar[%n%]=n"
135 if %n% equ 99 set /a n=-1
136 set /a n+=1
137 set "ar[%n%]=@"
138 if %n% equ 99 set /a n=-1
139 set /a n+=1
140 set "ar[%n%]=M"
141 if %n% equ 99 set /a n=-1
142 set /a n+=1
143 set "ar[%n%]=t"
144 if %n% equ 99 set /a n=-1
145 set /a n+=1
146 set "ar[%n%]=b"
147 if %n% equ 99 set /a n=-1
148 set /a n+=1
149 set "ar[%n%]=+"
150 if %n% equ 99 set /a n=-1
151 set /a n+=1
152 set "ar[%n%]={"
153 if %n% equ 99 set /a n=-1
154 set /a n+=1
155 set "ar[%n%]=-"
156 if %n% equ 99 set /a n=-1
157 set /a n+=1
158 set "ar[%n%]=U"
159 if %n% equ 99 set /a n=-1
160 set /a n+=1
161 set "ar[%n%]=&"
```

```
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=o"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=#"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=L"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=^"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=("
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=3"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=x"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=1"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=C"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=:"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=G"
```

```
195 if %n% equ 99 set /a n=-1
196 set /a n+=1
197 set "ar[%n%]=$"
198 if %n% equ 99 set /a n=-1
199 set /a n+=1
200 set "ar[%n%]=K"
201 if %n% equ 99 set /a n=-1
202 set /a n+=1
203 set "ar[%n%]=f"
204 if %n% equ 99 set /a n=-1
205 set /a n+=1
206 set "ar[%n%]=H"
207 if %n% equ 99 set /a n=-1
208 set /a n+=1
209 set "ar[%n%]=E"
210 if %n% equ 99 set /a n=-1
211 set /a n+=1
212 set "ar[%n%]=W"
213 if %n% equ 99 set /a n=-1
214 set /a n+=1
215 set "ar[%n%]=="
216 if %n% equ 99 set /a n=-1
217 set /a n+=1
218 set "ar[%n%]=Y"
219 if %n% equ 99 set /a n=-1
220 set /a n+=1
221 set "ar[%n%]=1"
222 if %n% equ 99 set /a n=-1
223 set /a n+=1
224 set "ar[%n%]=e"
225 if %n% equ 99 set /a n=-1
226 set /a n+=1
227 set "ar[%n%]=5"
```

```
228 if %n% equ 99 set /a n=-1
229 set /a n+=1
230 set "ar[%n%]=q"
231 if %n% equ 99 set /a n=-1
232 set /a n+=1
233 set "ar[%n%]=9"
234 if %n% equ 99 set /a n=-1
235 set /a n+=1
236 set "ar[%n%]=7"
237 if %n% equ 99 set /a n=-1
238 set /a n+=1
239 set "ar[%n%]=6"
240 if %n% equ 99 set /a n=-1
241 set /a n+=1
242 set "ar[%n%]=3"
243 if %n% equ 99 set /a n=-1
244 set /a n+=1
245 set "ar[%n%]=\"
246 if %n% equ 99 set /a n=-1
247 set /a n+=1
248 set "ar[%n%]=A"
249 if %n% equ 99 set /a n=-1
250 set /a n+=1
251 set "ar[%n%]=]"
252 if %n% equ 99 set /a n=-1
253 set /a n+=1
254 set "ar[%n%]=u"
255 if %n% equ 99 set /a n=-1
256 set /a n+=1
257 set "ar[%n%]=6"
258 if %n% equ 99 set /a n=-1
259 set /a n+=1
260 set "ar[%n%]=T"
```

```
261 if %n% equ 99 set /a n=-1
262 set /a n+=1
263 set "ar[%n%]=i"
264 if %n% equ 99 set /a n=-1
265 set /a n+=1
266 set "ar[%n%]=h"
267 if %n% equ 99 set /a n=-1
268 set /a n+=1
269 set "ar[%n%]=S"
270 if %n% equ 99 set /a n=-1
271 set /a n+=1
272 set "ar[%n%]=p"
273 if %n% equ 99 set /a n=-1
274 set /a n+=1
275 set "ar[%n%]=s"
276 if %n% equ 99 set /a n=-1
277 set /a n+=1
278 set "ar[%n%]=c"
279 if %n% equ 99 set /a n=-1
280 set /a n+=1
281 set "ar[%n%]=w"
282 if %n% equ 99 set /a n=-1
283 set /a n+=1
284 set "ar[%n%]=a"
285 if %n% equ 99 set /a n=-1
286 set /a n+=1
287 set "ar[%n%]=)"
288 if %n% equ 99 set /a n=-1
289 set /a n+=1
290 set "ar[%n%]=D"
291 if %n% equ 99 set /a n=-1
292 set /a n+=1
293 set "ar[%n%]=|"
```

```
294 if %n% equ 99 set /a n=-1
295 set /a n+=1
296 set "ar[%n%]=0"
297 if %n% equ 99 set /a n=-1
298 set /a n+=1
299 set "ar[%n%]=9"
300 if %n% equ 99 set /a n=-1
301 set /a n+=1
302 set "ar[%n%]=,"
303 if %n% equ 99 set /a n=-1
304 set /a n+=1
305 set "ar[%n%]=%%"
306 if %n% equ 99 set /a n=-1
307 set /a n+=1
308 set "ar[%n%]=*"
309 if %n% equ 99 set /a n=-1
310 set /a n+=1
311 set "ar[%n%]=4"
312 if %n% equ 99 set /a n=-1
313 set /a n+=1
314 set "ar[%n%]=_"
315 if %n% equ 99 set /a n=-1
316 set /a n+=1
317 set "ar[%n%]=2"
318 if %n% equ 99 set /a n=-1
319 set /a n+=1
320 set "ar[%n%]=X"
321 if %n% equ 99 set /a n=-1
322 set /a n+=1
323 set "ar[%n%]=j"
324 if %n% equ 99 set /a n=-1
325 set /a n+=1
326 set "ar[%n%]=N"
```

```
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=P"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=Q"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=g"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=J"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=F"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=V"
if %n% equ 99 set /a n=-1
set /a n+=1
set "ar[%n%]=B"

:redo
set /a chk=0
:: Εισάγουμε το κλειδί.
if %ena%==notthere (
    set /p give="Enter the given passkey: "
    set /a strt=!give!
)
if not %ena%==notthere (
    set /a strt=%ena%
)

```

```
:: Ελέγχουμε την ορθότητα του κλειδιού.
if %strt% leq 999 (
    set /a chk=1
    if not %ena%==notthere set /a chk=3
)
if %strt% geq 10000 (
    set /a chk=1
    if not %ena%==notthere set /a chk=3
)
if %chk%==1 (
    cls
    echo.
    echo wrong passkey
    echo please try again
    echo.
    goto redo
)
if %chk%==3 exit

:: Προσθέτουμε στο κλειδί την μεταβλητή που έχει δημιουργηθεί κατα την
εγκατάσταση.
set /a strt=%strt%+%COUNT%
if %strt% geq 10000 ( set /a strt=%strt%/10 )

:: Παίρνουμε ένα ένα τα ψηφία του κλειδιού.
set /a fst=%strt%/1000
set /a sub=%fst%*1000
set /a rem=%strt%-%sub%
set /a snd=%rem%/100
set /a sub=%snd%*100
set /a rem=%rem%-%sub%
set /a trd=%rem%/10
set /a sub=%trd%*10
```

```
393 set /a frt=%rem%-%sub%
394
395 :: Υπολογίζουμε το άθροισμα των ψηφίων.
396 set /a sm=%fst%+%snd%+%trd%+%frt%
397
398 :: Δημιουργούμε τους πολλαπλασιαστές.
399 set /a mpa=%CTHN%+%sm%
400 set /a mpb=%CTHG%+%sm%
401 set /a mpc=%CTHD%+%sm%
402 set /a mpd=%n%+%sm%
403
404 :: Δημιουργούμε τα ψηφία του κωδικού επιβεβαίωσης
405 set /a smaa=%frt%*%mpa%
406 set /a smbb=%trd%*%mpb%
407 set /a smcc=%snd%*%mpc%
408 set /a smdd=%fst%*%mpd%
409
410 :: Δημιουργούμε τον κωδικό επιβεβαίωσης.
411 set /a check=%smaa%+%smbb%+%smcc%+%smdd%+%COUNT%
412
413 :: ελέγχουμε την ορθότητα του κωδικού επιβεβαίωσης.
414 :recalc
415
416 if %check% geq 10000 (
417     set /a check=%check%/10
418     goto recalc
419 )
420
421 :: Παίρνουμε ένα ένα τα ψηφία του κωδικού επιβεβαίωσης.
422 set /a fstc=%check%/1000
423 set /a sub=%fstc%*1000
424 set /a rem=%check%-%sub%
425 set /a sndc=%rem%/100
```

```
426 set /a sub=%sndc%*100
427 set /a rem=%rem%-%sub%
428 set /a trdc=%rem%/10
429 set /a sub=%trdc%*10
430 set /a frtc=%rem%-%sub%
431
432 :: Δημιουργούμε τα ζευγάρια των ψηφίων του κωδικού επιβεβαίωσης και του
433 κλειδιού για να βρούμε τις θέσεις των ψηφίων του password στο array.
434 set /a sub=%fst%*10
435 set /a a=%sub%+%fstc%
436 set /a e=%sub%+%frtc%
437 set /a sub=%snd%*10
438 set /a b=%sub%+%sndc%
439 set /a f=%sub%+%trdc%
440 set /a sub=%trd%*10
441 set /a c=%sub%+%trdc%
442 set /a g=%sub%+%sndc%
443 set /a sub=%frt%*10
444 set /a d=%sub%+%fstc%
445 set /a h=%sub%+%frtc%
446 set /a sub=%trdc%*10
447 set /a i=%sub%+%fst%
448 set /a p=%sub%+%frt%
449 set /a sub=%frtc%*10
450 set /a j=%sub%+%snd%
451 set /a o=%sub%+%trd%
452 set /a sub=%fstc%*10
453 set /a k=%sub%+%trd%
454 set /a n=%sub%+%snd%
455 set /a sub=%sndc%*10
456 set /a m=%sub%+%fst%
457 set /a l=%sub%+%frt%
458
```

```
:: Δημιουργούμε το password με βάση τις θέσεις στο array.
set
ps=!ar[%a%]!!ar[%b%]!!ar[%c%]!!ar[%d%]!!ar[%e%]!!ar[%f%]!!ar[%g%]!!ar[%h
%]!!ar[%i%]!!ar[%j%]!!ar[%k%]!!ar[%l%]!!ar[%m%]!!ar[%n%]!!ar[%o%]!!ar[%p
%]!


color a


:: Τυπώνουμε στην οθόνη τον κωδικό επιβεβαίωσης και το password.
if %ena%==notthere (
    cls
    echo.
    echo Password Generator
    echo.
    echo for the given passkey
    echo.
    echo the PASSWORD is
    echo !ps!
    echo.
    echo.
    echo and the CHECK NUMBER is
    echo %check%
    echo.
)
if not %ena%==notthere (
    echo !ps! %check%
)


:: Σβήνουμε όλες τις μεταβλητές.
set ps=0
set /a check=0
set a=0
set b=0
set c=0
```

```
493 set d=0
494 set e=0
495 set f=0
496 set g=0
497 set h=0
498 set i=0
499 set g=0
500 set k=0
501 set l=0
502 set m=0
503 set n=0
504 set o=0
505 set p=0
506 set /a sub=0
507 set /a rem=0
508 set /a mpa=0
509 set /a mpb=0
510 set /a mpc=0
511 set /a mpd=0
512 set /a smaa=0
513 set /a smab=0
514 set /a smac=0
515 set /a smad=0
516 set /a fst=0
517 set /a snd=0
518 set /a trd=0
519 set /a frt=0
520 set /a fstc=0
521 set /a sndc=0
522 set /a trdc=0
523 set /a frtc=0
524 set /a strt=0
525 set /a chk=0
```

```
526  set give=0
527  for /L %%A in (0,1,39) do (
528       set ar[%%A]=0
529  )
530
531  :: Περιμένουμε 30 δευτερόλεπτα για να κλείσει η εφαρμογή.
532  if %ena%==notthere (
533      set /a ena=0
534      timeout /T 30
535  )
536  set /a ena=0
```

ΠΑΡΑΡΤΗΜΑ Γ

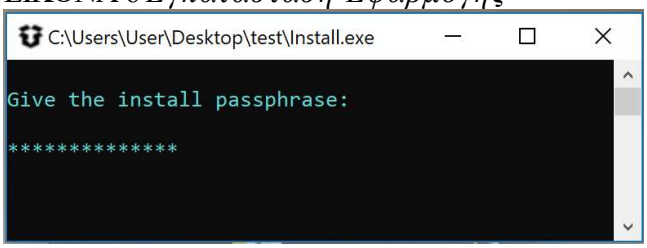
Παράδειγμα χρήσης της εφαρμογής.

Σε αυτό το κεφάλαιο θα παρουσιάσουμε μια χρήση της εφαρμογής και θα δείξουμε το περιεχόμενο όλων των βοηθητικών αρχείων και των αποτελεσμάτων που επιστρέφονται.

Η εγκατάσταση

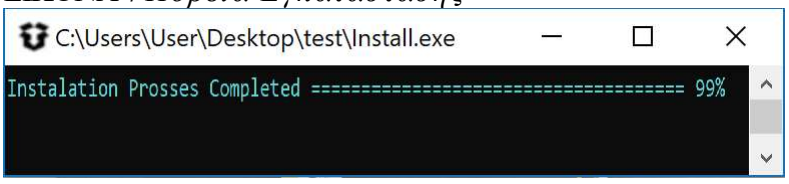
Για την εγκατάσταση του αρχείου το μόνο που χρειαζόμαστε είναι μια μυστική φράση. Στο παράδειγμά μας χρησιμοποιήσαμε την φράση «test run for check». Έτσι τρέξαμε το αρχείο εγκατάστασης (Install) και όταν μας ζητήθηκε εισάγαμε την παραπάνω μυστική φράση. Το μήκος της μυστικής μας φράσης είναι 18 χαρακτήρες οπότε η μεταβλητή len=18

ΕΙΚΟΝΑ 6 *Εγκατάσταση Εφαρμογής*



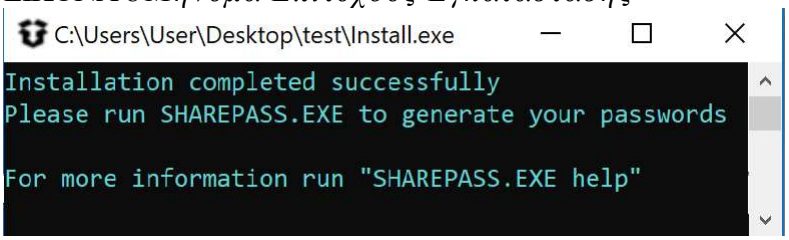
Το πρόγραμμα συνεχίζει με την εγκατάσταση και μας δείχνει την πορεία αυτής.

ΕΙΚΟΝΑ 7 *Πορεία Εγκατάστασης*



Στο τέλος μας εμφανίζει το μήνυμα της επιτυχούς εγκατάστασης πριν ολοκληρώσει η εφαρμογή και κλείσει το παράθυρο.

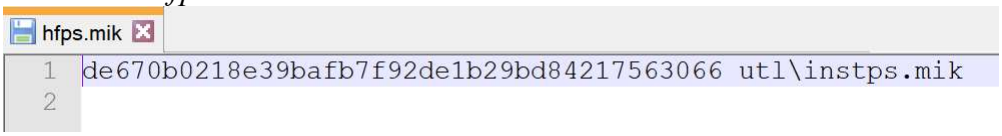
ΕΙΚΟΝΑ 8 *Μήνυμα Επιτυχούς Εγκατάστασης*



Κατά την εγκατάσταση δημιουργήθηκαν τα παρακάτω βοηθητικά αρχεία:

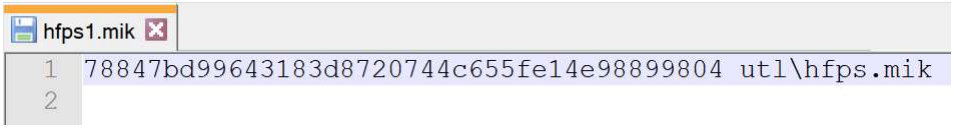
- instps.mik το οποίο περιέχει την μυστική φράση
- hfps.mik το οποίο περιέχει την sha1 του πρώτου αρχείου και το όνομα αυτού.

ΕΙΚΟΝΑ 9 *hfps.mik*



- hfps1.mik το οποίο περιέχει την sha1 του hfps.mik σαν δεύτερο βήμα ασφαλείας όπως είδαμε στην παράγραφο 6.1.4

ΕΙΚΟΝΑ 10 *hfps1.mik*



- και τέλος το αρχείο alg.mik που περιέχει τον κώδικα του κυρίως αρχείου συμπληρωμένο με τις τιμές των μεταβλητών που χρησιμοποιήθηκαν κατά την εγκατάσταση

Το αποτέλεσμα της sha1 με το οποίο υπολογίζονται οι μεταβλητές μας είναι

78847bd99643183d8720744c655fe14e98899804

Όπως είδαμε στην παράγραφο 6.1.5 οι μεταβλητές μας παίρνουν τις τιμές:

ΕΙΚΟΝΑ 11 *Μεταβλητές*

7	8	8	4	7	b	d	9	9	6	4	3	1	8	3	d	8	7	2	0	7	4	4	c	6	5	5	f	e	1	4	e	9	8	8	9	9	8	0	4
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39

7	8	8	4	7	9	9	6	4	3	1	8	3	8	7	2	7	4	4	6	5	5	1	4	9	8	8	9	9	8	4
0	1	2	3	4	7	8	9	10	11	12	13	14	16	17	18	20	21	22	24	25	26	29	30	32	33	34	35	36	37	39
0	8	16	12	28	63	72	54	40	33	12	104	42	128	119	36	140	84	88	144	125	130	29	120	288	264	272	315	324	296	156

cthn=519
count=3542

b	d	d	0	c	f	e	e	0
5	6	15	19	23	27	28	31	38

cthg=192

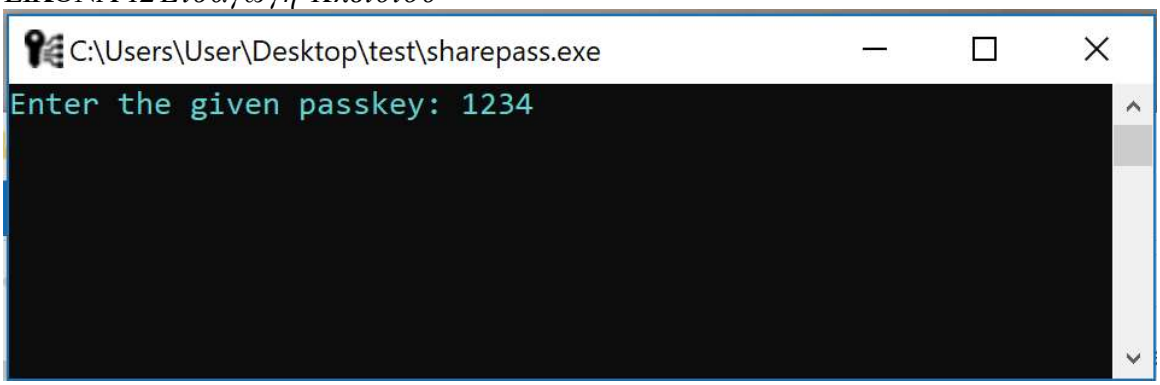
Μετά την προσθαφαίρεση του μήκους της μυστικής φράσης οι μεταβλητές έχουν τις τελικές τιμές:

- count=count-len=3542-18=3524
- cthn=cthn+len=519+18=537
- cthg=cthg+18=192+18=210
- cthd=cthn-cthg=537-210=327
- arst=(count/len)+(cthg/cthd)+(cthn/cthd)=3524/18+519/327+210/327=195+1+0=196 και επειδή είναι μεγαλύτερο του 100 το διαιρούμε με το 2 μέχρι να φτάσουμε σε ένα αριθμό από 0 έως 99, 196/2=98. Οπότε η καταχώρηση των χαρακτήρων θα ξεκινήσει από την θέση 98 του array.

Εκτέλεση του κυρίως αρχείου

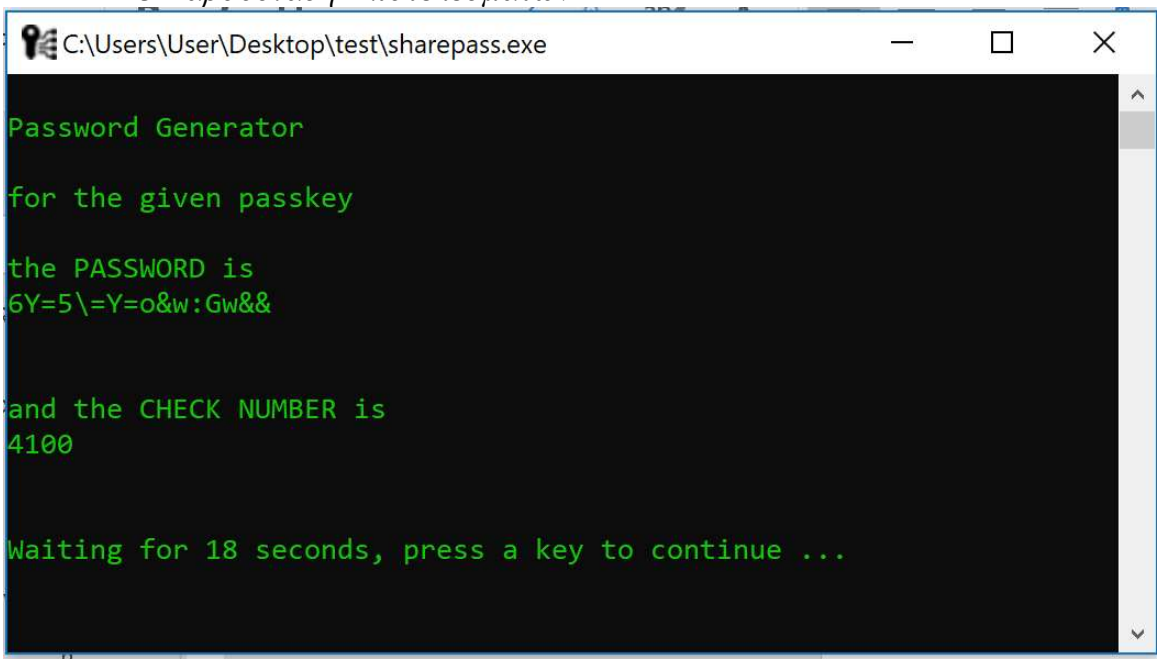
Επόμενο βήμα είναι να «τρέξουμε» το κυρίως αρχείο (sharepass)και να δούμε τα αποτελέσματα που μας επιστρέφει. Κατά την εκτέλεση λοιπόν η εφαρμογή μας ζητάει να εισάγουμε το τετραψήφιο κλειδί:

ΕΙΚΟΝΑ 12 *Εισαγωγή Κλειδιού*



Αφού εισάγουμε το κλειδί η εφαρμογή επιστρέφει τα αποτελέσματα, τον μυστικό κωδικό και το κωδικό επιβεβαίωσης.

ΕΙΚΟΝΑ 13 *Παρουσίαση Αποτελεσμάτων*



Τέλος παράγουμε δέκα διαφορετικούς κωδικούς με χρήση τυχαίων κλειδιών για να ελέγξουμε την σωστή λειτουργία της εφαρμογής.

ΠΙΝΑΚΑΣ 6 *Αποτελέσματα*

Κλειδί	Μυστικός Κωδικός	Κωδικός επιβεβαίωσης
1234	6Y=5\=Y=o&w:Gw&&	4100
7310	r=@Rr55R,:W3x:W9	1041
3984	Nr}7X+8qkQ[:KdN8	7185
2893	|Z8fs8Z153?R0?3Y	6820
5123	4$fG8f$C~CAc0ACJ	3461
4321	~ALbJTUrjG.Wf41Q	7151
9087	K$LOGK#4*A%0c*\N	5453
8465	K3|X1^9%J-2kR%#@	5860
5678	}@~?z~@/[/2jN2/z	8576
3456	jw0|20w)O)B5qB);	6274

Όπως βλέπουμε από τον πίνακα αποτελεσμάτων, όλοι οι κωδικοί που παράγονται περιέχουν χαρακτήρες από όλες τις ομάδες χαρακτήρων που περιγράψαμε στην παράγραφο 6.2.4.3 και δεν αποτελούν κάποια γνωστή λέξη αλλά εντελώς τυχαία επιλογή κάνοντας τον μυστικό κωδικό πραγματικά ασφαλή.

ΠΑΡΑΡΤΗΜΑ Δ

License Agreement

ΒΙΒΛΙΟΓΡΑΦΙΑ

1password, (n.d.) [online], Available: https://1password.com/families/ [13-03-2019]

AGE LAZZARO, (11-09-2017), Are YOUR messages secure? Study finds only 14% of people who use encrypted services such as WhatsApp properly enable the security features [online], Available: https://www.dailymail.co.uk/sciencetech/article-4782924/WhatsApp-FB-Messenger-Viber-expose-users-hacks-study.html [13-03-2019]

Angus Johnson, Resource Hacker, (03-01-2019) [online], Available: http://www.angusj.com [16-03-2019]

Chris Elpidis (31-03-2019), [online], available: https://www.techgear.gr/brian-acton-whatsapp-facebook-157956/ [01-04-2019]

Cyberoam, (n.d.), [online], Available: https://www.cyberoam.com/vpn.html [13-03-2019]

Don Ho, (n.d.) [online], Available: https://notepad-plus-plus.org [13-03-2019]

Efail, (n.d.) [online], Available: https://efail.de/ [13-03-2019]

Fatih Kodak, (n.d.), f2k0 Software, [online], Available: http://www.f2ko.de/en/index.php [13-03-2019]

Gnu Operating System, (n.d.), [online], Available: https://www.gnu.org/copyleft/gpl.html [13-03-2019]

J. Callas, L. Donnerhacke, H. Finney, D. Shaw, R. Thayer, (11-2007), OpenPGP Message Format [online], Available: https://tools.ietf.org/html/rfc4880 [13-03-2019]

JOHN HALL, (n.d.), SplashData's Top 100 Worst Passwords of 2018 [online], Available: https://www.teamsid.com/splashdatas-top-100-worst-passwords-of-2018/ [13-03-2019]

Kalinda, (11-11-2016), Μπαμπινιώτης Η ελληνική γλώσσα έχει περίπου 100.000 λέξεις και 300.000 σημασίες [online], Available: https://olympia.gr/2016/11/11/μπαμπινιώτης-η-ελληνική-γλώσσα-έχει-πε/ [13-03-2019]

Lawson KurtzViget, (26-04-2016), Viget, Email is completely insecure by default [online], Available: https://www.viget.com/articles/email-is-completely-insecure-by-default/ [13-03-2019]

Microsoft Corp (2018), File Checksum Integrity Verifier [online], Available: https://support.microsoft.com/en-us/help/841290/availability-and-description-of-the-file-checksum-integrity-verifier-u [13-03-2019]

Norton, (n.d.), How to choose a secure password [online], Available: https://us.norton.com/internetsecurity-how-to-how-to-choose-a-secure-password.html [13-03-2019]

OpenPGP, (n.d.) https://www.openpgp.org/ [13-03-2019]

Pleasantsolutions, (n.d.) [online], Available: http://www.pleasantsolutions.com/passwordserver [13-03-2019]

Rarlab (n.d.) [online], Available: https://www.rarlab.com/ [13-03-2019]

Satoshi Nakamoto, (n.d.), Wikipedia, Base58 [online], Available: https://en.wikipedia.org/wiki/Base58 [13-03-2019]

Scintilla, (07-03-2019), A free source code editing component [online], Available: https://www.scintilla.org/ [13-03-2019]

Sophos, (17-10-2014), Average person has 19 passwords – but 1 in 3 don't make them strong enough [online], Available: https://nakedsecurity.sophos.com/2014/10/17/average-person-has-19-passwords-but-1-in-3-dont-make-them-strong-enough [13-03-2019]

Wikipedia, (n.d.) Batch file [online], Available: https://en.wikipedia.org/wiki/Batch_file [13-03-2019]

Wikipedia, (n.d.) Διαδίκτυο [online], Available: https://el.wikipedia.org/wiki/Διαδίκτυο#Η_ιστορία_του_Διαδικτύου [13-03-2019]

Wikipedia, (n.d.), cmd.exe, [online], Available: https://en.wikipedia.org/wiki/Cmd.exe [13-03-2019]

Wikipedia, (n.d.), Cryptographic hash function [online], Available: https://en.wikipedia.org/wiki/Cryptographic_hash_function [13-03-2019]

Wikipedia, (n.d.) Facebook–Cambridge Analytica data scandal [online], Available: https://en.wikipedia.org/wiki/Facebook%E2%80%93Cambridge_Analytica_data_scandal [01-04-2019]

Wikipedia, (n.d.), sha-1, [online], Available: https://en.wikipedia.org/wiki/SHA-1 [13-03-2019]

Winzip (n.d.) [online], Available: https://www.winzip.com/win/en/ [13-03-2019]

Χριστόφορος Χαραλαμπάκης, (02-05-2007) Πόσες λέξεις διαθέτει τελικά η ελληνική γλώσσα; [online], Available: http://users.ntua.gr/nborbil/Article1.htm [13-03-2019]