Honeyword Generation Approach Using ASCII Values for Security Enhancement

Ms. KomalNaik
M.E. Student, Department of Computer,
SLRTCE-Mumbai University,
Mumbai, Maharashtra, India
komalnaik20@gmail.com

Prof. VarshaBhosale
Associate Professor, Department of Information Technology,
VIT-Mumbai University,
Mumbai, Maharashtra, India
varsha.bhosale@vit.edu.in

Prof. Vinayak D.Shinde
Assistant professor, Department of Computer,
SLRTCE-Mumbai University
Mumbai, Maharashtra, India
vdshinde@gmail.com

Abstract **-** The innovation in the field of data has provided people satisfaction and comfort, but there are many issues related to security of data. To overcome these issues Juels and Rivest have recently introduced the honeyword concept. Many existing methods for honeyword generation are available but there are limitations related to storage cost, multiple system vulnerability, DoS attack resistance, etc. In this paper we have proposed a honeyword generation approach using ASCII values which provides better security against brute force attack. In this algorithm randomly generated numbers are used to generate the honeywords from the original password. The overall analysis shows that ASCII method achieves a high detection rate.

Index Terms **-** *ASCII, Authentication, Detection, Honeyword, Intruder.*

*****

I. INTRODUCTION

A password is a secret word, which a user must input during a login, only after that it is possible to get access. Security of password is an important aspect. Password protection helps us to protect information from unauthorized users. An intruder who steals a file of hashed passwords mostly uses brute-force search to find the password. Many companies and software industries store their data in databases. So, all the details required to get into the system (i.e. username and password) are stored in encrypted form in database. It is easy to capture the plaintext passwords after the password file is stolen. So for avoiding it, passwords must be protected and secured by using the appropriate algorithm and a secure system should detect the entry of unauthorized user in the system.

Password protection was handled using hashing algorithms. The password is converted into hash using the existing hashing algorithms. So, the username and hash of password is stored into the database. Using Inversion attack model the attacker can successfully invert the hashes from the compromised password file. Later, study focusses on detecting the entry of unauthorized user using the concept of fake passwords known as honeywords.

A. Honeywords

Honeywords are false (decoy) passwords. For each user account, the legitimate password is stored with several honeywords in order to sense impersonation[1]. The file of hashed passwords include the legitimate as well as decoy passwords. After inverting the hashes if the honeywords are selected properly, an attacker who steals a file is not sure if it is the real password or a honeyword for any account. If he enters the honeyword for login, it will trigger an alarm notifying the administrator about a password file breach. Various existing honeyword generation algorithms are used to generate honeywords. Thus, this approach puts the attacker at risk of being detected with every attempted login. Using brute force attack also it is sure that the attacker will be detected. Hence, honeywords can provide a very useful layer of defense.

II. HONEYWORD GENERATION METHODS AND DISCUSSION

Honeyword generation algorithms can be categorized into two groups.

A. *Legacy-UI* procedures: The password-change UI is unchanged i.e. it takes the same password entered by the user for honeyword generation[1].

1) Chaffing-by-tweaking

In this method, the user password gives the generator algorithm for which changing selected character positions of the real password to produce the honeywords. Each character of a user password in already determined positions is replaced by a randomly chosen character of the same type. Number of positions to be changed are denoted as t.

For example the password is tom1991 .Suppose the following honeywords are generated with t = 4 and k = 9 for this password.The digits in the honeywords seem not relevant, but the correct password tom1991 makes sense for an adversary.

tom6333 tom9048 tom1991
tom2070 tom0946 tom2785
tom5469 tom8147 tom9105

Limitation: Many users have the tendency to choose the numbers included in passwords related to a special date. Most of the hacked password hints are related to a date. Apart from the use of a date in passwords, many users prefer to write consecutive numbers to their password heads, like ’456’, ’4567’, due to the tendency of users to choose rememberable number patterns. This model leads an adversary to make a correct selection.

2) Chaffing-with-a-password-model

The generator algorithm takes the password from the user and relying on a probabilistic model of real passwords it produces the honeywords. This method is named as the modeling syntax. In this model, the password is splitted into character sets. For example, gold4hair is decomposed as 4-letters + 1-digit + 4-letters = L4+D1+L4 and replaced with the same composition like urvi5ring.

Limitation: Leaked password databases have shown us that some passwords have a well-known pattern. If there is a correlation between the username and the password, then the password can be easily distinguished from the honeywords. For example, the password smittom123 with a username smittom can be easily distinguished from the corresponding honeywords.

3) Chaffing with Tough Nuts

Some honeywords are much harder to crack than the average so much that they would probably never be cracked by an adversary. There may possibly be some un-crackable hashes For example, what should the adversary do with the following list?
gm79, tom!!yahoo, ? , g*7vk45, rauid/49frogs!, ?
Having some "tough nuts"among the honeywords might give the adversary additional reason to pause before searching in and trying to log in with one of the cracked ones. "Tough nuts" represent potentially correct passwords whose plausibility the adversary cannot evaluate. Thus, in the presence of "tough nuts," the adversary cannot fully evaluate the likelihood of being detected during login, even if the adversary has cracked all other sweetwords.

4) *Hybrid Method*

Another method is combining the strength of different honeyword generation methods, e.g. chaffing-by-tweaking digits and chaffing-with-a-password-model. By using this technique, random password model will be generated.
For example let the correct password be litchi1903. Then the honeywords demon2562 and sad9137 should be produced as seeds to chaffing-by-tweaking digits. For t = 3 and k = 4 for each seed, the sweetwords table given below may be attained:

sad9679 litchi1422 demon2656
sad9757 litchi1903 demon2036
sad9743 litchi1172 demon2849
sad9392 litchi1792 demon2562

Remark: This method reduces the chance of an adversary in guessing the correct password from the sweetwords.

B. Modified-UI procedures: The password-change UI is modified to allow better honey-word generation [1].

1) Take-a-tail method

The user's actual password is modified at the end with a randomly chosen value to form a new user password. For example, xyz213 is user's password then system generates '@15' as a tail. So now user's new password becomes xyz@15. This method has good security standard but is very poor in usability. It is very difficult for user to remember the system generated information for his different accounts.

III. LITERATURE SURVEY

A. Achieving Flatness: Selecting the Honeywords from Existing User Passwords.

The system proposed in this paper works on the issue to overcome the security problems. A new honeyword generation algorithm in which honeywords are generated from the existing user passwords is proposed. It provides realistic honeywords. It shows better results with respect to flatness, DOS resistance and storage. New honeyword generation method reduces storage cost of the honeyword scheme[1].

B.The Dangers of Weak Hashes

K. Brown in 2012 proposed that the damaged caused by password leaks can be reduced by implementing few good practices.In June 2012 many companies likeLinkedIn, Yahoo etc... Were affected by password leaks which were publicly spread. Secure system should not have any loop holes that will allow intruders to get access to password files and should make sure that if the password hashes are been hacked it should not be easy to generate passwords from the hashes. Due to weak hashing mechanism these companies were highly affected is proved. In this paper the author has discussed about basics of password hashing and best practices that should be followed while password storage[2].

C.Guess again (and again and again): Measuring password strength by simulating password-cracking.

In this study in 2012 authors understood the effects of password composition methods on guessing ability of passwords. In-spite of rapid advancement in password cracking capability of hacker, text based password is yet a leading authentication method. Considering this threat, password composition methods are becoming complex day-by-day. But, there is less research for calculating the password strength. In this paper a new, efficient technique for calculating the password strength of password-guessing algorithms is introduced[4].

D.Honeywords: Making Password-Cracking Detectable

Juels and Rivest in 2013 proposed a method in field of security for improving the security of hashed passwords. To improve the security of the hashed password, decoy passwords needs to be generated for each user account. An attacker who steals a file of hashed passwords and inverts the hash function cannot tell if he has found the password or a honeyword. If the attacker tries to login with the honeyword the server will set off an alarm. Brute-force attack helps to detect the adversary when honeyword model is used [5].

E. Examination of a New Defense Mechanism: Honeywords.

It has become much easier to crack a password hash with the advancements in the graphical processing unit (GPU)

technology. They propose an approach for user authentication, in which some honeywords are added into a password file, in order to detect impersonation.An interesting defense mechanism where an adversary steals the file of password hashes and inverts most or many of the hashes is proposed. The honeyword system is powerful defense mechanism here. It is sure that the adversary will be detected [7].

IV. PROPOSED ALGORITHM ARCHITECTURE

The proposed ASCII algorithm works as follows

1. Firstly take the real password of the user.
2. The real password is assigned a random position.
3. Chaffing by tweaking logic is applied here where the last 'n' characters which needs to be replaced are selected.
4. Each character is assigned a randomly generated number between 33 to 126.
5. ASCII value of that randomly generated number is choosen from the standard ASCII table and the original characters are replaced with ASCII characters.
6. The above steps are repeated for generating the honeywords for a real password.

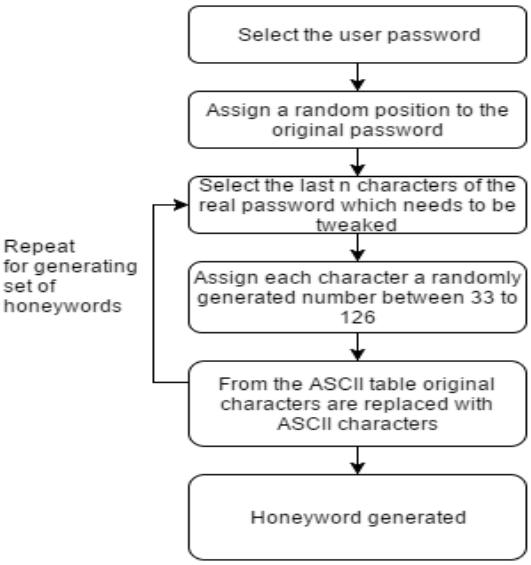

Fig. 1 Procedure of generating honeywords

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Table 1: ASCII Table

Example

Original Password	H	2	l	L	o
Random numbers generated for last 3 characters	H	2	94	78	52
ASCII values for randomly generated numbers	H	2	^	N	4
Generated honeyword	H	2	^	N	4

V. FUTURE SCOPE

Purpose of developing this algorithm is enhancing the security. There are many future scope of honeyword generation using ASCII value. Firstly the generated honeyword can be hashed or encrypted with different hashing and encryption techniques. This will help in multilevel security. Secondly as the need for data security is increasing and new technologies are getting upgraded so changes to the algorithm can be made if needed accordingly. Thus this work can be further improved in future in number of ways.

VI. CONCLUSION

The proposed methodology will give a new scope in area of honeyword generation using ASCII algorithms. This methodology is definitely an effective algorithm for honeyword generation while compared to other techniques. This algorithm is very fast, reliable, flat and secure.

VI. REFERENCES

[1] Imran Erguler, "Achieving Flatness: Selecting the Honeywords from Existing User Passwords," IEEE Transactions on Dependable and Secure Computing, vol. 13, no. 2, pp. 284 - 295, February 2015.

[2] Brown and Kelly, "The dangers of weak hashes," SANS Institute InfoSec Reading Room, November 2013

[3] Mirante, Dennis and Justin Cappos, "Understanding Password Database Compromises," Dept. of Computer Science and Engineering Polytechnic Inst. of NYU, 2013.

[4] Kelley, Patrick Gage and Michelle L. Mazurek, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," IEEE Symposium on Security and Privacy, pp. 523-537, May- 2012.

[5] J. A and L. R. R, "Honeywords: Making Password cracking Detectable," in ACM SIGSAC conference on Computer & communications security, November 2013.

[6] M. Dell'Amico, P. Michiardi and Y. Roudier, "Password Strength: An Empirical Analysis," INFOCOM'10: Proceedings of the 29th Conference on Information Communications, vol. 10, pp. 983-991, 2010.

[7] Genc, Ziya Alper and Mehmet Sabir Kiraz, "Examination of a New Defense Mechanism: Honeywords," IACR Cryptology ePrint Archive, p. 696, 2013.

[8] Bonneau and Joseph, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," IEEE Symposium on Security and Privacy, pp. 538-552, May 2012.

[9] R. Morris and K. Thompson, "Password Security: A Case History," Communications of the ACM, vol. 22, no. 11, pp. 594-597, 1979.

[10] M. Bercovitch, M. Renford and L. Hasson, "HoneyGen: An automated honeytokens generator," in IEEE International Conference on Intelligence and Security Informatics, July 2011.