World Academy of Science, Engineering and Technology
International Journal of Humanities and Social Sciences
Vol:9, No:12, 2015

Password Cracking on Graphics Processing Unit Based Systems

N. Gopalakrishna Kini, Ranjana Paleppady, Akshata K. Naik

***Abstract*—**Password authentication is one of the widely used methods to achieve authentication for legal users of computers and defense against attackers. There are many different ways to authenticate users of a system and there are many password cracking methods also developed. This paper proposes how best password cracking can be performed on a CPU-GPGPU based system. The main objective of this work is to project how quickly a password can be cracked with some knowledge about the computer security and password cracking if sufficient security is not incorporated to the system.

***Keywords*—**GPGPU, password cracking, secret key, user authentication.

I. Introduction

PASSWORDS are designed to provide a system authentication. There are many different ways to authenticate users of a system such as log in to the system using a user name and password pair, a user can present a physical object like a key card, prove identity using biometric like a fingerprint, face recognition etc. [1].

System authentication based on password, work by comparing user supplied passwords with stored secrets. When the system administrator or attacker gets the equivalent privileges then he/she can access these secrets. Usually the passwords are not stored in plaintext. Most of the time, it is in encrypted form.

Password cracking is the process of recovering passwords from data that have been stored in or transmitted by a computer system. It can also be defined as the process of getting the normal text passwords which collides with the used hash function. A few years back, all-to-all major password breach happened for a website wherein the attacker leaked the complete list of passwords as the passwords were stored in clear text in the database with the possible security by that website [2], [4].

Brute-force cracking is one of the techniques in which a computer tries every possible password until it succeeds. Brute-force attack keeps guessing repeatedly for every possible password and checks against the available encrypted password. Here, as the password length increases, Brute-force can take huge amount of time; i.e., each additional character in a password exponentially increases the Brute-force cracking time [3], [4]. The existing hardware architectures were dedicated to password cracking, such as Field-programmable gate array (FPGAs) that are devices containing fully programmable logic and CELL processor [6].

Password cracking is required for various reasons. Positively speaking, it might be required to recover a forgotten password or to identify the passwords that can be cracked easily so that the System Administrator can initiate necessary preventive measure. To speak negatively, an attacker gaining unauthorized access to the system. If it is encrypted password then the attacker need to verify whether a guessed password is successfully decoded from encrypted password. The cryptographic function used by the system for generating the password affects the rate, at which the password is guessed [5], [6].

An attacker can easily guess the password that is easy to remember for the authenticated user. There are various ways with which the security of the system reduces if the passwords are difficult to remember. Users may have to preserve the password by writing down on a paper or using an insecure method to store the password such as in a file.

The more stringent requirements for strengthening the passwords are to have a mix of lowercase and uppercase letters along with numbers and punctuation characters [1]. Passwords framed based on picking the first character of each word of a phrase are easily memorable as a selected password, and as hard to crack as randomly generated passwords. Another good way is to have a password is by combining two unrelated words.

The effort it has taken to crack the password decides the strength of a password. One can increase the length of the password for a cracking attempt to take more time. This results in more difficulty in recovering the password.

It is going to be great time consuming process if you rely wholly on the general purpose CPU based system to recover the password. Here we propose how best password cracking can be performed on a CPU-GPGPU based system [7].

The computational power of General-Purpose computation on Graphics Processing Units (GPGPUs) is exploited in our work. Computationally intensive tasks are solved using GPGPU where parallel computation is needed. Computational work is allocated in parallel operations among large number of cores within the GPGPU. In other words, data-parallel computations are highly supported by GPGPUs. Each program can be executed on many data elements in parallel. Here sophisticated flow control is not required among the separate threads.

Computation on GPGPU is based on the principle of data parallelism. Individual thread is processing each data element

N. Gopalakrishna Kini, Professor, Ranjana Paleppady and Akshata K. Naik PG student are Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal University, Manipal-576104, India (phone: +91820-2924527; e-mail: ng.kini@ manipal.edu, ranjana2588@ gmail.com, akshata.999@ gmail.com).

of a task. To speed up computation with GPGPUs, data-parallel programming model can be used that needs to process a large data set. This approach is used for generating such huge combinations of possible passwords. As a result, performance gain can be achieved by using different Work Item and Work Group distributions on GPGPU computation. Parallel processing suits very much for password cracking since the input data can be distributed uniformly over the group of concurrent threads. These threads can perform the same operations on different set of data until a matching password is identified, concurrently and independently.

In this paper, we present how easily the password can be cracked if enough security is not given to the system. Paper also talks on how an attacker can crack the password when he/she have the ability to attempt to log in to the system using a user name and password pair. Here we have assumed that an attacker has access to the password file, whichever format they are stored on the system for password crack work.

II. Methodology

Normally the authenticated users will have the password with a mix of uppercase and lowercase letters along with numbers and punctuation characters. Brute-force cracking procedure has been implemented on a machine having the specification as Intel i3 dual core CPU with operating frequency 1.70 GHz, 4 GB RAM, 64-bit Windows 8.1 and 1TB HDD. The system is supporting NVIDIA GeForce 820M GPU, 96 cores; 2GB dedicated RAM with OpenCL support.

Open Computing Language (OpenCL) is used to ease the parallel programming methodology when developing applications for heterogeneous systems. OpenCL also addresses the current trend to increase the number of cores on a given architecture.

OpenCL framework supports execution on multi-core CPU and GPGPUs. There are many more heterogeneous devices supported by OpenCL framework which is seldom of interest to our work. The heterogeneous architecture has already supported to cover a wide range of approaches to extract parallelism and efficiency from memory systems and instruction streams. OpenCL has standard abstractions and interfaces that allow programmers to develop the application within which execution can occur on a rich set of heterogeneous devices [8].

Here a parallel code developed [9]-[12] in OpenCL can generate a certain length of password based upon all permutations and combinations of uppercase and lowercase letters along with numbers and punctuation characters.

The code is written such that it can generate all combination of two character length passwords, three character length passwords and so on up to *n* character length passwords containing uppercase and lowercase letters along with numbers and punctuation characters.

In this paper, we have considered the generation of password with maximum length 4 and this password can contain various combinational mix of uppercase and lowercase letters along with numbers and punctuation characters from maximum of 52 character set.

Even though the generation of passwords of different lengths is sequential, the generation of passwords with all permutations and combinations for a certain length happens in parallel. In our proposed algorithm, this is how we have reduced Brute-force cracking time greatly in contrast with each additional character in a password that exponentially increasing the cracking time on CPU based system [2], [5].

This Brute-force method further could be enhanced to crack the encrypted password if the secret key is known. Attacker now can easily crack an authenticated user's password by searching for the matching combination in authenticated encrypted password file of the system.

Absolutely in no time attacker can decode an authenticated password of a user if sufficient security is not incorporated to the system.

III. Result

Our experiment tests the power of parallel computing on GPGPU against the computations needed by Brute-force technique to generate the two, three and four character length passwords containing uppercase and lowercase letters. This also explores the performance gain which can be achieved by using different Work Item and Work Group distributions on the GPGPU.

Figs. 1 (a)-(c) show partial snapshots of two, three and four character length passwords generated. Power of parallel computing analysis also has been done for different Work Item and Work Group creation.

aa	ba	ca	da	ea	fa	ga	ha	ia	ja	ka	la	ma
	na	oa	pa	qa	ra	sa	ta	ua	va	wa	xa	ya
	za	Aa	Ba	Ca	Da	Ea	Fa	Ga	Ha	Ia	Ja	Ka
	La	Ma	Na	Oa	Pa	Qa	Ra	Sa	Ta	Ua	Va	Wa
	Xa	Ya	Za	ab	bb	cb	db	eb	fb	gb	hb	ib
	jb	kb	lb	mb	nb	ob	pb	qb	rb	sb	tb	ub
	vb	wb	xb	yb	zb	Ab	Bb	Cb	Db	Eb	Fb	Gb
	Hb	Ib	Jb	Kb	Lb	Mb	Nb	Ob	Pb	Qb	Rb	Sb
	Tb	Ub	Vb	Wb	Xb	Yb	Zb	ac	bc	cc	dc	ec
	fc	gc	hc	ic	jc	kc	lc	mc	nc	oc	pc	qc
	rc	sc	tc	uc	vc	wc	xc	yc	zc	Ac	Bc	Cc
	Dc	Ec	Fc	Gc	Hc	Ic	Jc	Kc	Lc	Mc	Nc	Oc
	Pc	Qc	Rc	Sc	Tc	Uc	Vc	Wc	Xc	Yc	Zc	ad
	bd	cd	dd	ed	fd	gd	hd	id	jd	kd	ld	md
	nd	od	pd	qd	rd	sd	td	ud	vd	wd	xd	yd
	zd	Ad	Bd	Cd	Dd	Ed	Fd	Gd	Hd	Id	Jd	Kd
	Ld	Md	Nd	Od	Pd	Qd	Rd	Sd	Td	Ud	Vd	Wd
	Xd	Yd	Zd	ae	be	ce	de	ee	fe	ge	he	ie
	je	ke	le	me	ne	oe	pe	qe	re	se	te	ue
	ve	we	xe	ye	ze	Ae	Be	Ce	De	Ee	Fe	Ge
	He	Ie	Je	Ke	Le	Me	Ne	Oe	Pe	Qe	Re	Se
	Te	Ue	Ve	We	Xe	Ye	Ze	af	bf	cf	df	ef
	ff	gf	hf	if	jf	kf	lf	mf	nf	of	pf	qf
	rf	sf	tf	uf	vf	wf	xf	yf	zf	Af	Bf	Cf
	Df	Ef	Ff	Gf	Hf	If	Jf	Kf	Lf	Mf	Nf	Of
	Pf	Qf	Rf	Sf	Tf	Uf	Vf	Wf	Xf	Yf	Zf	ag

Fig. 1 (a) Partial list of all combinations of two character length passwords containing uppercase and lowercase letters

aaa	baa	caa	daa	eaa	faa	gaa	haa	iaa	jaa	kaa	laa	maa
	naa	oaa	paa	qaa	raa	saa	taa	uaa	vaa	waa	xaa	yaa
	zaa	Aaa	Baa	Caa	Daa	Eaa	Faa	Gaa	Haa	Iaa	Jaa	Kaa
	Laa	Maa	Naa	Oaa	Paa	Qaa	Raa	Saa	Taa	Uaa	Vaa	Waa
	Xaa	Yaa	Zaa	aba	bba	cba	dba	eba	fba	gba	hba	iba
	jba	kba	lba	mba	nba	oba	pba	qba	rba	sba	tba	uba
	vba	wba	xba	yba	zba	Aba	Bba	Cba	Dba	Eba	Fba	Gba
	Hba	Iba	Jba	Kba	Lba	Mba	Nba	Oba	Pba	Qba	Rba	Sba
	Tba	Uba	Vba	Wba	Xba	Yba	Zba	aca	bca	cca	dca	eca
	fca	gca	hca	ica	jca	kca	lca	mca	nca	oca	pca	qca
	rca	sca	tca	uca	vca	wca	xca	yca	zca	Aca	Bca	Cca
	Dca	Eca	Fca	Gca	Hca	Ica	Jca	Kca	Lca	Mca	Nca	Oca
	Pca	Qca	Rca	Sca	Tca	Uca	Vca	Wca	Xca	Yca	Zca	ada
	bda	cda	dda	eda	fda	gda	hda	ida	jda	kda	lda	mda
	nda	oda	pda	qda	rda	sda	tda	uda	vda	wda	xda	yda
	zda	Ada	Bda	Cda	Dda	Eda	Fda	Gda	Hda	Ida	Jda	Kda
	Lda	Mda	Nda	Oda	Pda	Qda	Rda	Sda	Tda	Uda	Vda	Wda
	Xda	Yda	Zda	aea	bea	cea	dea	eea	fea	gea	hea	iea
	jea	kea	lea	mea	nea	oea	pea	qea	rea	sea	tea	uea
	vea	wea	xea	yea	zea	Aea	Bea	Cea	Dea	Eea	Fea	Gea
	Hea	Iea	Jea	Kea	Lea	Mea	Nea	Oea	Pea	Qea	Rea	Sea
	Tea	Uea	Vea	Wea	Xea	Yea	Zea	afa	bfa	cfa	dfa	efa
	ffa	gfa	hfa	ifa	jfa	kfa	lfa	mfa	nfa	ofa	pfa	qfa
	rfa	sfa	tfa	ufa	vfa	wfa	xfa	yfa	zfa	Afa	Bfa	Cfa
	Dfa	Efa	Ffa	Gfa	Hfa	Ifa	Jfa	Kfa	Lfa	Mfa	Nfa	Ofa
	Pfa	Qfa	Rfa	Sfa	Tfa	Ufa	Vfa	Wfa	Xfa	Yfa	Zfa	aga

Fig. 1 (b) Partial list of all combinations of three character length passwords containing uppercase and lowercase letters

aaaa	baaa	caaa	daaa	eaaa	faaa	gaaa	haaa	iaaa	jaaa	kaaa	laaa	maaa
	naaa	oaaa	paaa	qaaa	raaa	saaa	taaa	uaaa	vaaa	waaa	xaaa	yaaa
	zaaa	Aaaa	Baaa	Caaa	Daaa	Eaaa	Faaa	Gaaa	Haaa	Iaaa	Jaaa	Kaaa
	Laaa	Maaa	Naaa	Oaaa	Paaa	Qaaa	Raaa	Saaa	Taaa	Uaaa	Vaaa	Waaa
	Xaaa	Yaaa	Zaaa	abaa	bbaa	cbaa	dbaa	ebaa	fbaa	gbaa	hbaa	ibaa
	jbaa	kbaa	lbaa	mbaa	nbaa	obaa	pbaa	qbaa	rbaa	sbaa	tbaa	ubaa
	vbaa	wbaa	xbaa	ybaa	zbaa	Abaa	Bbaa	Cbaa	Dbaa	Ebaa	Fbaa	Gbaa
	Hbaa	Ibaa	Jbaa	Kbaa	Lbaa	Mbaa	Nbaa	Obaa	Pbaa	Qbaa	Rbaa	Sbaa
	Tbaa	Ubaa	Vbaa	Wbaa	Xbaa	Ybaa	Zbaa	acaa	bcaa	ccaa	dcaa	ecaa
	fcaa	gcaa	hcaa	icaa	jcaa	kcaa	lcaa	mcaa	ncaa	ocaa	pcaa	qcaa
	rcaa	scaa	tcaa	ucaa	vcaa	wcaa	xcaa	ycaa	zcaa	Acaa	Bcaa	Ccaa
	Dcaa	Ecaa	Fcaa	Gcaa	Hcaa	Icaa	Jcaa	Kcaa	Lcaa	Mcaa	Ncaa	Ocaa
	Pcaa	Qcaa	Rcaa	Scaa	Tcaa	Ucaa	Vcaa	Wcaa	Xcaa	Ycaa	Zcaa	adaa
	bdaa	cdaa	ddaa	edaa	fdaa	gdaa	hdaa	idaa	jdaa	kdaa	ldaa	mdaa
	ndaa	odaa	pdaa	qdaa	rdaa	sdaa	tdaa	udaa	vdaa	wdaa	xdaa	ydaa
	zdaa	Adaa	Bdaa	Cdaa	Ddaa	Edaa	Fdaa	Gdaa	Hdaa	Idaa	Jdaa	Kdaa
	Ldaa	Mdaa	Ndaa	Odaa	Pdaa	Qdaa	Rdaa	Sdaa	Tdaa	Udaa	Vdaa	Wdaa
	Xdaa	Ydaa	Zdaa	aeaa	beaa	ceaa	deaa	eeaa	feaa	geaa	heaa	ieaa
	jeaa	keaa	leaa	meaa	neaa	oeaa	peaa	qeaa	reaa	seaa	teaa	ueaa
	veaa	weaa	xeaa	yeaa	zeaa	Aeaa	Beaa	Ceaa	Deaa	Eeaa	Feaa	Geaa
	Heaa	Ieaa	Jeaa	Keaa	Leaa	Meaa	Neaa	Oeaa	Peaa	Qeaa	Reaa	Seaa
	Teaa	Ueaa	Veaa	Weaa	Xeaa	Yeaa	Zeaa	afaa	bfaa	cfaa	dfaa	efaa
	ffaa	gfaa	hfaa	ifaa	jfaa	kfaa	lfaa	mfaa	nfaa	ofaa	pfaa	qfaa
	rfaa	sfaa	tfaa	ufaa	vfaa	wfaa	xfaa	yfaa	zfaa	Afaa	Bfaa	Cfaa
	Dfaa	Efaa	Ffaa	Gfaa	Hfaa	Ifaa	Jfaa	Kfaa	Lfaa	Mfaa	Nfaa	Ofaa
	Pfaa	Qfaa	Rfaa	Sfaa	Tfaa	Ufaa	Vfaa	Wfaa	Xfaa	Yfaa	Zfaa	agaa
	bgaa	cgaa	dgaa	egaa	fgaa	ggaa	hgaa	igaa	jgaa	kgaa	lgaa	mgaa
	ngaa	ogaa	pgaa	qgaa	rgaa	sgaa	tgaa	ugaa	vgaa	wgaa	xgaa	ygaa
	zgaa	Agaa	Bgaa	Cgaa	Dgaa	Egaa	Fgaa	Ggaa	Hgaa	Igaa	Jgaa	Kgaa
	Lgaa	Mgaa	Ngaa	Ogaa	Pgaa	Qgaa	Rgaa	Sgaa	Tgaa	Ugaa	Vgaa	Wgaa
	Xgaa	Ygaa	Zgaa	ahaa	bhaa	chaa	dhaa	ehaa	fhaa	ghaa	hhaa	ihaa
	jhaa	khaa	lhaa	mhaa	nhaa	ohaa	phaa	qhaa	rhaa	shaa	thaa	uhaa
	vhaa	whaa	xhaa	yhaa	zhaa	Ahaa	Bhaa	Chaa	Dhaa	Ehaa	Fhaa	Ghaa
	Hhaa	Ihaa	Jhaa	Khaa	Lhaa	Mhaa	Nhaa	Ohaa	Phaa	Qhaa	Rhaa	Shaa
	Thaa	Uhaa	Vhaa	Whaa	Xhaa	Yhaa	Zhaa	aiaa	biaa	ciaa	diaa	eiaa
	fiaa	giaa	hiaa	iiaa	jiaa	kiaa	liaa	miaa	niaa	oiaa	piaa	qiaa
	riaa	siaa	tiaa	uiaa	viaa	wiaa	xiaa	yiaa	ziaa	Aiaa	Biaa	Ciaa

Fig. 1 (c) Partial list of all combinations of four character length passwords containing uppercase and lowercase letters

Fig. 2 (a) gives the details on the amount of computational time taken by the Kernel and also by the whole program. This computational time is to generate all possible combinations of two, three and four character length passwords with Work Item size = 1 and Work Group size = 52.

In Fig. 2 (b), the computational time by the Kernel code and the whole program is considered with Work Item size = 13 and Work Group size = 4. Fig. 2 (c) enunciates the computational time by the Kernel and the whole program with Work Item size = 26 and Work Group size = 2.

It has been observed that the amount of time taken to run the whole program is almost identical in all the cases. However, it is the rate at which the generation of all possible combination of passwords matters.

It is strongly to be noted on the amount of time taken for the kernel code running on the GPGPU of a computer. A total of 7454928 numbers of two, three and four character length passwords has been generated by the GPGPU in a very short time.

While generating the passwords, the passwords of different lengths are sequential. The generation of passwords with all permutations and combinations of characters for a certain length happens in parallel. In our proposed algorithm, this is how we have reduced Brute-force cracking time greatly in contrast with each additional character in a password that exponentially increasing the cracking time on CPU based system. This kernel time is reduced further to accelerate the number of password generation by increasing the Work item and hence reducing Work Group.

Fig. 2 (a) Time taken by the Kernel and the whole program with Work Item size = 1 and Work Group size = 52

Fig. 2 (b) Time taken by the Kernel and the whole program with Work Item size = 13 and Work Group size = 4

Fig. 2 (c) Time taken by the Kernel and the whole program with Work Item size = 26 and Work Group size = 2

The amount of time in cracking a password depends on password strength and the details of how the password is stored. In general, the total number of passwords generated by the GPGPU is calculated as:

$$L^m + L^{m+1} + \cdots + L^M$$

where L is the number of characters set, m minimum length and M is the maximum length of the password generated.

IV. Conclusion

Efficient password cracking is a trade-off between success rate and speed. In our paper, we described the one of the technique of password cracking. We have shown that how GPGPUs are efficient in a system to generate all the possible combination of passwords. It has been explained how Brute-force method can become most feasible approach towards password cracking using CPU-GPGPU based system.

In our work, we analyzed the effect of parallelism on execution of kernel on GPGPU. We proposed how best password cracking can be performed on a CPU-GPGPU based system. If sufficient security is not incorporated to the system, then it is shown how quickly a password can be cracked with some knowledge about the computer security.

The result provided here is for the password of length of 4 characters. The same can be extended to passwords of any length without compromising over the time taken for the password generation in contrast with the CPU based system.

The best method of preventing a password from being cracked is to ensure that attackers cannot get access even to the encrypted password. Encrypted passwords should be stored in the file which is accessible only to programs running with system privileges. This makes it harder for an attacker to obtain the encrypted passwords. Unfortunately, many common network protocols transmit passwords in clear text or use weak challenge/response schemes.

References

[1] Silberschatz, Galvin, Gagne, "Operating Systems Concepts," 8th Edition, Wiley India Edition, 2009.
[2] http://www.threatmetrix.com/learn-to-hack-and-crack-passwords-in-a-day-without-breaking-a-sweat-using-free-online-tools (9 October 2015).
[3] http://www.oxid.it/ca_um/topics/brute-force_password_cracker.htm (9 October 2015).
[4] Stallings W., Brown L., "Computer Security: Principles and Practice," Prentice Hall, 2nd Edition, 2011.
[5] Weir M., Aggarwal S., Collins M., Stern H., "Testing metrics for password creation policies by attacking large sets of revealed passwords," *Proceedings of the 17th ACM conference on Computer and communications security (CCS '10)*, ACM, NY, USA, 162-175, 2010.
[6] Marechal S., "Advances in password cracking," *Journal in Computer Virology*, vol. 4, no. 1, pp. 73–81, February, 2008.
[7] Aleksandar Kasabov, Jochem van Kerkwijk, Marc Smeets, and et al., "Distributed GPU Password Cracking," Universiteit Van Amsterdam, May, 2011.
[8] Benedict Gaster et al. "Heterogeneous Computing with OpenCL," Elsevier, 2012.
[9] Kai Hwang and Faye A. Briggs, "Computer Architecture and Parallel Processing," TMH Private Ltd., 2012.
[10] Peter Pachico, "An introduction to parallel programming," Elsevier, 2011.
[11] Michael J. Quinn, "Parallel computing: Theory and practice," McGraw-Hill Inc., 2008.
[12] K. Hwang, "Advanced Computer Architecture: Parallelism, Scalability, Programmability," New York, McGraw-Hill, 1993.