BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2024
http://excel.fit.vutbr.cz

Lámání hesel pomocí Rainbow Tables na GPU

Bc. David Jahoda*

Abstrakt

Ve světe kde je stále dominantní formou přístupu k systému autentizace heslem je stále nezbytné zabývat se metodami pro jejich prolamování a zkoumat jakou sílu mají. Jednou z těchto metod je time-memory tradeoff [1] ve formě využití rainbow tables [2] a ty zkoumá i tato práce. Práce zkoumá účinnost jak původně navržených způsobů realizace, tak i rozšířených způsobů. To vše se snahou o co největší akceleraci skrze GPU. Tato práce si klade za cíl prozkoumat účinnost rainbow table po 20 letech od jejich prvního publikování. Jak si tato technika vede při použití možností aktuálního hardwaru a softwaru (masivní paralelismus, akcelerace na GPU, ...) a to proti v současné době zavedeným opatřením a algoritmům (salting hesel, iterativní hashování, ...).

*xjahod05@stud.fit.vutbr.cz, *Fakulta informačních technologií, Vysoké učení technické v Brně*

1. Úvod

Hesla jsou stále nejběžnějším způsobem přihlášení k uživatelským účtům ať již se jedná o multimediální předplatitelské služby, hry nebo i emailové služby. A pokud daný systém neobsahuje další formy autentizace nebo jich uživatel nevyužívá (např. dvoufázové ověření) jedná se jedinou překážku od potencionálního kompromitování účtu.

Nicméně i když pomineme slabé hesla, phishing a jiné praktiky sociálního inženýrství, tak zde stále přetrvává jeden možný vektor útoku. Konkrétně samotný poskytoval dané služby a jeho úložiště hesel. Pokud se útočníkovi povede prolomit bezpečnostní opatření poskytovatele a zcizit databázi hesel, tak mohou být v ten moment kompromitovány všechny uživatelské účty. I z tohoto důvodu tedy nejsou hesla ukládána v otevřeném textu, ale v podobě hashe. Avšak i na něj je možné útočit a jedním z těchto útoků je i ten založený na duhových tabulkách (rainbow tables).

2. Útok pomocí duhových tabulek

Útok pomocí duhových tabulek (rainbow tables) byl poprvé zveřejněn v článku[2] Philippa Oechslin roku 2003. Spočívalo v předvypočítání tabulky tzv. duhových řetězů, kdy každý řetěz pokrýval část stavového prostoru hesel. Dosaženo je to alterací dvou operací a to zahashováním otevřeného textu hesla a následnou aplikací redukční funkce z prostoru hashů zpět do prostoru hesel. Její aplikací na vzniklý hash se vytvoří opět otevřený text hesla. Zřetězení těchto dvou operací pak vzniká řetěz o délce N. Ten je ve výsledné tabulce reprezentován pouze startovním a koncovým bodem řetězu (původní heslo a koncový hash). Pro prolomení vybraného hashe hesla se pak prohledává tato tabulka, hledají se kandidátní řetězy a ty se následně znovu rozgenerovávají. Pokud se v některém řetězu vyskytne lámaný hash, tak před ním nacházející se otevřený text v řetězu je pak jeho otevřenou podobou. Detailnější popis viz původní článek[2]. To inspirovalo řadu implementací, jmenovitě např. nástroj Rainbow Crack[1], jenž je obecně považován za referenční implementaci. Zásadním rozšířením pak byla práce R.E. Gravese[3] z roku 2008, který celý koncept duhových tabulek převedl na GPU.

Aktuálně dostupná řešení však trápí několik neduhů. Jedním z nich je zpoplatnění jednotlivých částí nástroje, kdy GPU akcelerace byly možna pouze u prodávaných tabulek a nikoliv uživatelem generovaných (případ Rainbow Cracku). Další problémem je, že většina těchto řešení je z dob uveřejnění těchto prací a nebyly dále rozšiřovány. To dnes ústí problémy s kompatibilitou na aktuálních systémech (případ Cryptohaze[2]). Např. akcelerace na NVIDIA GPU u kterých nemá jejich kód garantován dopřednou kompatibilitu. Jelikož

[1] http://project-rainbowcrack.com/
[2] https://www.cryptohaze.com/

jsou typicky neudržovaná, tak ani nevyužívají pokroků na poli knihoven, možností programovacích jazyků a schopností aktuálního hardwaru. Jedná se tedy o potenciálně ztracený výkon.

Toto je v konečném důsledkem cílem této práce. Vypracovat moderní implementaci nástroje pro lámání hesel využívající duhových tabulek, postaveným na aktuálních technologiích jak na poli softwaru tak i hardwaru. Následně provést výkonností testování a porovnání s existujícími řešeními a provést zhodnocení relevantnosti duhových tabulek ve světle zavedených protiopatření vůči nim.

3. Návrh řešení

Návrh řešení se sestává z terminálové aplikace. Ta se bude opírat o možnosti knihoven OpenMP a NVIDIA CUDA, pro CPU, respektive GPU akceleraci. Samotná aplikace se bude skládat ze tří funkčních modulů: generátoru duhových tabulek, postprocesoru tabulek a modulu pro prohledávání tabulek neboli lámání hashů. U každého z těchto modulů budou k dispozici různé míry akcelerace, od sekvenční verze, vícevláknovou CPU až po GPU a multi-GPU akcelerovanou. Samotná GPU akcelerace se bude opírat o dvě konkrétní techniky. Překrytí paměťových transférů užitečnou prací a využitím heterogenního zpracování.

Překrytí paměťových transférů užitečnou prací spočívá v současné vytížení jak výpočetních (ALU) jednotek, tak i paměťových LOAD/STORE jednotek GPU. Dosáhne se toho za pomoci prefetchingu. Práce pro GPU se rozdělí do bloků určité velikosti (např. 1 blok prohledá 1 milion záznamů v tabulce) a ty budou postupně nahrávány do paměti GPU ke zpracování. A zatímco se bude na GPU zpracovávat jeden blok, tak výsledky z předchozího budou zasílány zpět do paměti hosta (CPU) a v tomtéž momentu bude i přenášen i další bok ke zpracování na GPU. Tím se dosáhne plného vytížení jak paměťových tak výpočetních jednotek současně a nebudou muset na sebe navzájem čekat.

Heterogenním zpracování je myšlena snaha dosáhnout součinného řešení práce jak na CPU tak i GPU a tím tak eliminovat čekání CPU na dokončení práce GPU. Z možných prací, kterými se CPU může zaobírat během práce GPU je např. už výše zmíněný prefetching v podobě načítání/ukládání částí duhových tabulek ze/do souborů. Kromě toho se však může podílet i na dynamickém plánování práce pro GPU, kdy dle vytíženosti, latenci zpracování/načtení jednotlivých bloků může měnit za běhu množství distribuované práce, vytvářet nové kernely pro chod na dalších dostupných GPU. V neposlední řadě se může také v mezičase věnovat další post-procesové práci jako např. řazení vygenerovaných řetězů nebo jejich perfektizace při generování tabulky.

Akcelerace na úrovni bloků se pak opírá o SIMT (Single instruction, multiple threads) zpracování jenž GPU využívá. Kód je zde vykonáván po tzv. warpech. Warp je balík 32 vláken, které sdílí programový čítač a vykonávají tudíž všechny tentýž kód současně. Jakmile warp narazí na instrukci, která vyžaduje čekání (paměťové operace) nebo obecně trvající desítky či spíše stovky taktů (dělení, goniometrické funkce), tak daný warp uspán a okamžitě se začne provádět jakýkoliv jiný, který má vše připraven a může okamžitě běžet. Díky tomu že toto přepnutí vykonávaných warpů trvá pouze jeden takt, lze tak velmi efektivně vykrývat jakákoliv čekání.

Aby však bylo SIMT zpracování efektivní musí kód splňovat určité podmínky. Jednak musí být rozpracováno dostatečné množství vláken, tak aby byl v každém momentě některý warp připraven k běhu. Dále pro maximální výkonnost se nesmí akcelerovaný kód nikterak větvit. Pokud ano, tak může docházet k tzv. divergenci vláken, kdy část vláken v rámci warpu postupuje jednou větví kódu a další část jinou. GPU však neobsahuje dostatečnou řídící logiku pro tento případ a tak dochází k serializaci vykonávání větví. Část vláken tak musí čekat a nevykonává žádné instrukce. Další nezbytnou podmínkou pro udržení výkonu je pak jednotkový rozestup při paměťových přístupech vláken. Pokud je toto porušeno, dojde k bankovým konfliktům u GPU paměti a vydávání dat je opět serializováno. Při dodržení jednotkových rozestupů je však možné vydat každému vláknu warpu data v tomtéž taktu.

4. Závěr

Tato práce pokrývá pouze základní případy použití pro běh na jediném systému. Rozšíření a možností jak pokračovat se nabízí celá řada. Rozvoj multi-GPU řešení a plánování. Distribuovaná varianta nástroje pro běh v rámci výpočetního clusteru, využívající různé uzly a řízené pomocí zasílání zpráv (MPI). Využití redukčních funkcí založených na statistických metodách a markovských řetězech s cílem modelovat více "lidská" hesla. Port kódu na jiné platformy s cílem pokrýt GPU všech rozšířených výrobců (HIP). Je zde tím pádem značný prostor pro rozvíjení tohoto typu nástroje a možné navazující práce.

Poděkování

Chtěl bych poděkovat Mgr. Kamilu Malinkovi, Ph.D za vedení práce a cenné rady při jejím vypracování. Dále doc. Ing. Jiřímu Jarošovi, Ph.D za konzultace v oblasti GPGPU. A taktéž i MetaCentrum za poskytnutou výpočetní infrastrukturu.

Literatura

[1] M. Hellman. A cryptanalytic time-memory trade-off. *IEEE transactions on information theory*, 26(4):401–406, 1980.

[2] Philippe Oechslin. Making a faster cryptanalytic time-memory trade-off. In *ADVANCES IN CRYPTOLOGY-CRYPTO 2003, PROCEEDINGS*, volume 2729 of *Lecture Notes in Computer Science*, pages 617–630. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[3] R.E. Graves. High performance password cracking by implementing rainbow tables on nvidia graphics cards (isecrack). Master's thesis, Iowa State University, 2008.