Wydział Podstawowych Problemów Techniki
Politechnika Wrocławska

Memory-hard Functions

Konrad Świerczyński
Nr indeksu: 229818

Promotor
dr Filip Zagórski

Politechnika
Wrocławska
Wrocław 2019

Spis treści

Wstęp

Moderately hard functions. Funkcje umiarkowanie ciężkie do obliczenia mają wiele zastosowań takich jak dowody pracy (ang. *proofs of work*), funkcje wyprowadzenia klucza oraz password hashing. Przy przechowywaniu haseł ważne jest, aby zminimalizować skutki wycieknięcia pliku z hasłami. Zamiast przechowywać krotki ($login$, $password$) tekstem jawnym, dodaje się losową sól i przechowuje w postaci ($login$, $f(password, salt)$, $salt$), gdzie f jest moderately hard function. Oznacza to, że funkcja ta musi być obliczana podczas każdego uwierzytelniania w celu sprawdzenia poprawności hasła. Nie może być ona zatem zbyt ciężka do obliczenia dla aplikacji uwierzytelniającej. Z drugiej strony, gdy krotka ($login$, y, $salt$) wycieknie, adwersarz może przeprowadzać atak słownikowy obliczając funkcję f przy każdej próbie, co powinno być kosztowne. W tym celu zaczęto stosować funkcje, które obliczają wiele razy kryptograficzną funkcję skrótu. Popularnym przykładem takiej funkcji jest PBKDF2 (ang. *Pssowrd-Based Key Derivation Function 2*), dla której zalecanym parametrem bezpieczeństwa w 2000 roku było 1024 iteracji, a już w 2005 zaczęto zalecać 4096 iteracji, z powodu wzrostu wydajności CPU. Niestety takie podejście nie gwarantuje zabezpieczenia przed adwersarzem używającym sepcjalizowany układ scalony (ang. *ASIC - Application-Specyfic Integrated Circut*). Układy takie są znacznie bardziej wydajne poz względem szybkości obliczania funkcji skrótu takich jak SHA256 czy MD5 niż tradycyjne architektury.

– Porównianie Antminer - GPU - CPU –

Zauważono jednak, że na różnych architekturach koszt dostępu do pamięci jest dużo bardziej zrównoważony niż koszt obliczeń. [Percival [16]] Zaproponowano więc memory-hard functions (MHF), które wywołują podaczas obliczania wiele kosztownych czasowo odwołań do pamięci.

scrypt - pierwsza taka funkcja

O MHF można myśleć jako o pewnej kolejności dostępu do komórek pamięci. Odwołania następują do już wcześniej obliczonych wartości w komórkach. Zatem kolejność tą można opisać jako acykliczny graf skierowany (DAG).

.....

Analiza problemu

Definicja 2.1 *(Parallel/Sequential Graph Pebbling). Niech $G = (V, E)$ będzie grafem skierowanym grafem acyklicznym i niech $T \subset V$ będzie zbiorem wierzchołków do oetykietowania. T będzie nazywane celem. Stanem etykietowania G jest zbiór $P_i \subset V$. Poprawnym etykietowaniem równoległym jest ciąg $P = (P_0, \ldots, P_t)$ stanów etykietowania G, gdzie $P_0 = \emptyset$ oraz gdzie spełnione są warunki 1 oraz 2 poniżej. Etykietowanie sekwencyjne musi dodatkowo spełniać warunek 3.*

1. *Każdy wierzchołek z celu jest w pewnej konfiguracji oetykietowany (nie koniecznie wszystkie jednocześnie).*
$$\forall x \in T \exists x \leqslant t : x \in P_x$$
2. *Oetykietować wierzchołek można tylko wtedy, gdy wszyscy jego rodzice są oetykietowani w poprzednim kroku.*
$$\forall i \in [t] : x \in (P_i \setminus P_{i-1}) \Rightarrow parents(x) \subset P_{i-1}$$
3. *W każdym kroku można oetykietować co najwyżej jeden wierzchołek.*
$$\forall i \in [t] : |P_i \setminus P_{i-1}| \leqslant 1$$

Zbiory poprawnych etykietowań sekwencyjnych i równoległych grafu G z celem T oznaczamy odpowiednio jako $\mathcal{P}_{G,T}$ oraz $\mathcal{P}^{\parallel}_{G,T}$. Etykietowania najbardziej interesujących przypadków, gdy $T = sinks(G)$, oznaczamy $\mathcal{P}_G$ oraz $\mathcal{P}^{\parallel}_G$.

Można zauważyć, że $\mathcal{P}_{G,T} \subset \mathcal{P}^{\parallel}_{G,T}$.

Definicja 2.2 *Złożoność czasową, pamięciową, pamięciowo-czasową oraz łączna etykietowania $P = (P_0, \ldots, P_t) \in \mathcal{P}^{\parallel}_G$ są zdefiniowane jako*

$$\Pi_t(P) = t, \Pi_s(P) = \max_{y \in [t]} |P_i|, \Pi_{st}(P) = \Pi_t(P) * \Pi_s(P), \Pi_{cc}(P) = \sum_{i \in [t]} |P_i|$$

Dla $\alpha \in s, t, st, cc$ oraz celu $T \subset V$, złożoności sekwencyjnego oraz równoległego etykietowania grafu G definiujemy jako

$$\Pi_\alpha(G, T) = \min_{P \in \mathcal{P}_{G,T}} \Pi_\alpha(P)$$

$$\Pi^{\parallel}_\alpha(G, T) = \min_{P \in \mathcal{P}^{\parallel}_{G,T}} \Pi_\alpha(P)$$

Kiedy $T = sinks(G)$, piszemy $\Pi^{\parallel}_\alpha(G)$ oraz $\Pi_\alpha(G)$.

Definicja 2.3 *(N-Superconcentrator) Skierowany graf acykliczny $G = (V, E)$ o ustalonym stopniu, N wejściach i N wyjściach nazywany jest N-Superkoncentratorem gdy dla każdego $k \in [N]$ oraz dla każdej pary podzbiorów $V_1 \subset V$ k wejść i $V_2 \subset V$ k wyjść istnieje k wierzchołkowo-rozłącznych ścieżek łączących wierzchołki ze zbioru V_1 z wierzchołkami w V_2.*

Definicja 2.4 *((N, λ)-Superconcentrator) Niech $G_i, i = 0, \ldots, \lambda - 1$ będą N-Superkoncentratorami. Niech graf G będzie połączeniem wyjść G_i do odpowiadających wejść w G_{i+1} dla $i = 0, \ldots, \lambda - 2$. Graf G jest nazywany (N, λ)-Superkoncentratorem.*

Twierdzenie 2.1 *(Ograniczenie dolne dla (N, λ)-Superkoncentratora)) Pebbling a (N, λ)-Superconcentrator using $S \leqslant N/20$ pebbles requires T placements such that*

$$T \geqslant N \left(\frac{\lambda N}{64S}\right)^{\lambda}.$$

Definicja 2.5 *(Depth-Robustness) Dla $n \in \mathbb{N}$ oraz $e, d \in [n]$ acykliczny graf skierowany $G = (V, E)$ jest (e, d)-depth-robust jeżeli*

$$\forall S \subset V |S| \leqslant e \Rightarrow depth(G - S) \geqslant d$$

Definicja 2.6 *(Dependencies) Niech $G = (V, E)$ będzie acyklicznym grafem skierowanym. Niech $L \subseteq V$. Mówimy, że L ma (z, g)-dependency jeżeli istnieją wierzchołkowo rozłączne ścieżki $p_1, \ldots, p_z$ kończące się w L, gdzie każda jest długości co najmniej g.*

Definicja 2.7 *(Dispresed Graph) Niech $g, k \in \mathbb{N}$ i $g \geqslant k$. DAG G jest nazywany (g, k)-dispresed jeżeli istnieje uporządkowanie jego wierzchołków takie, że następujące warunki są spełnione. Niech $[k]$ oznacza ostatnie k wierzchołków o uporządkowaniu G i niech $L_j = [jg, (j+1)g - 1]$ będzie j-tym podprzedziałem. Wtedy $\forall j \in [\lfloor k/g \rfloor]$ przedział L_j ma (g, g)-dependency. W ogólności, jeżeli dla $\epsilon \in (0, 1]$ każdy przedział L_j ma tylko $(\epsilon g, g)$-dependency, graf G nazywany jest (ϵ, g, g)-dispresed.*

Twierdzenie 2.2 *Niech $\lambda, n \in \mathbb{N}^+$ takie, że $n = \overline{n}(2\lambda c + 1)$, gdzie $c \in \mathbb{N}$ i $\overline{n} = 2^c$. Wtedy dla $g = \lfloor \sqrt{\overline{n}} \rfloor$ $RSG_{\lambda}^{\overline{n}} \in \mathbb{D}_{1,g}^{\lambda,\overline{n}}$ oraz $\Pi_{cc}^{\parallel}(RSG_{\lambda}^{\overline{n}}) = \Omega\left(\frac{n^{1.5}}{c\sqrt{c\lambda}}\right)$.*

Dowód. Niech $G = RSG_{\lambda}^{\overline{n}}$, niech $G_1, G_2, \ldots, G_{\lambda}$ będą podgrafami G opisanymi w DEF[...]. Pokażemy, że każdy G_i jest (g, $\overline{n}$)-dispresed dla $g = \lfloor \sqrt{\overline{n}} \rfloor$.

Wybierzmy $i \in [\lambda]$ niech L_1 będzie ostatnimi $\overline{n}$ wierzchołkami w porządku topologicznym grafu G_i. Oznaczamy wierzchołki zbioru L_1 poprzez $1 \times [\overline{n}]$, gdzie druga pozycja odpowiada kolejności wierzchołka w porządku topologicznym. Niech $\overline{g} = \lfloor \overline{n}/g \rfloor$, dla każdego $j \in [\overline{g}]$ $L_{1,j} = \{< 1, jg + x >: x \in [0, g - 1]\}$. Pokażemy, że wszystkie $L_{1,j}$ mają (g, g)-dependency.

Niech L_0 będzie $\overline{n}$ pierwszymi wierzchołkami G_i, które oznaczamy $0 \times [\overline{n}]$ (ponownie druga pozycja odpowiada porządkowi topograficznemu). Zauważmy, że dla $n>1$ i $g = \lfloor \sqrt{\overline{n}} \rfloor$ prawdą jest, że $g(g-2c+1) \leqslant n$. Zatem zbiór $S = \{< 0, i(g - 2c + 1_{)} >: i \in [g]\}$ jest całkowicie zawarty w L_0.

Z własności *RSG* [Superconcentrator- ....] wynika, że skoro zbiory S oraz $L_{1,j}$ mają po g wierzchołków, to istnieje g wierzchołkowo-rozłącznych ścieżek o długości $2c$ między wierzchołkami tych zbiorów. Zatem $L_{1,j}$ ma (g, 2c)-dependency.

Rozszerzmy to do (g,g)-dependency. Niech ścieżka p zaczynająca się w wierzchołku $< 0, v > \in S$ będzie ścieżką w (g, 2c)-dependency $L_{1,j}$. Zauważmy, że istnieje ścieżka przechodząca przez wszystkie wierzchołki L_0 oraz, że wierzchołki zbiory S są oddzielone między sobą o $g - 2c$ wierzchołków. Możemy dodać na początek ścieżki p ścieżkę $(< 0, v - (g - 2c - 1) >, < 0, v - (g - 2c - 2) >, \ldots, < 0, v >)$. Otrzymujemy w ten sposób ścieżkę p_+ o długości $2c + g - 2c = g$. Ponieważ każda para ścieżek $p \neq q$ w (g, 2c)-dependency $L_{1,j}$ jest wierzchołkowo-rozłączna, to w szczególności zaczynać się muszą w różnych wierzchołkach $< 0, v_p > \neq < 0, v_q >$. Ponieważ wierzchołki w S są od siebie oddalone o $g - 2c$ wierzchołków, zatem ścieżki p^+ i q_+ nadal pozostają rozłączne. Rozszerzając w ten sposób wszystkie ścieżki z (g, 2c)-dependency otrzymujemy ścieżki wierzchołkowo-rozłączne długości g. Z tego wynika, że $L_{1,j}$ ma (g, g)-dependency, co dowodzi, że $RSG_{\lambda}^{\overline{n}} \in \mathbb{D}_{1,g}^{\lambda,\overline{n}}$. Pozostaje obliczyć górne ograniczenie używając [Theorem 6 ABP2017].

$$\Pi_{cc}^{\parallel}(RSG_{\lambda}^{\overline{n}}) = \lambda g\left(\frac{\overline{n}}{2} - g\right) \geqslant \lambda\lfloor\sqrt{\overline{n}}\rfloor\left(\frac{\overline{n}}{2} - \lfloor\sqrt{\overline{n}}\rfloor\right) = \lambda\sqrt{\overline{n}}\left(\frac{\overline{n}}{2} - \sqrt{\overline{n}}\right) - O(\overline{n}) = \Omega\left(\lambda\overline{n}\right) = \Omega\left(\frac{n^{1.5}}{c\sqrt{c\lambda}}\right) \quad \square$$

Twierdzenie 2.3 *Niech $\lambda, g \in \mathbb{N}^+$, $N = 2^g$, $n = N(2\lambda g + 1)$, wtedy*

$$\Pi_{cc}^{\parallel}(RSG_{\lambda}^{\overline{n}}) = O\left(n^{1.\overline{6}}\right)$$

Dowód. Kożystając, z [...], RSG_λ^N jest λ-Stacked Sandwich Graph. Z [Lemma 4.2 AB16] wynika więc, że RSG_λ^N jest $(n/t,\ \lambda + t - \lambda - 1)$-reducible dla dowolnego $t \geqslant 1$. Z twierdzenia 10 [ABP17] wynika, że $\Pi_{cc}^{\|}(RSG_\lambda^{\overline{n}}) = O\left(n\left(\sqrt{(\lambda t + t - \lambda - 1)n\delta} + \frac{n}{t}\right)\right)$. Aby dostać najdokładniejsze (najmniejsze) ograniczenie górne trzeba zminimalizować $n\left(\sqrt{(\lambda t + t - \lambda - 1)n\delta} + \frac{n}{t}\right)$. Zanim jednak przejdziemy do minimalizowania, uprośćmy nieco to wyrażenie

$$n\left(\sqrt{(\lambda t + t - \lambda - 1)n\delta} + \frac{n}{t}\right) \leqslant n\left(\sqrt{2\lambda t n\delta} + \frac{n}{t}\right).$$

Teraz możemy znaleźć minimum względem naszego parametru t.

$$\frac{\partial}{\partial t} n\left(\sqrt{(2\lambda t)n\delta} + \frac{n}{t}\right) = \frac{\sqrt{2\delta\lambda n^3}}{2t} - \frac{n^2}{t^2}$$

Minimum znajduje się w punkcje, gdzie pochodna ma wartość zero.

$$\frac{\sqrt{2\delta\lambda n^3}}{2t} - \frac{n^2}{t^2} = 0$$

$$t = \frac{2n^2}{\sqrt{2\delta\lambda n^3}} = O\left(n^{\frac{1}{3}}\right)$$

Zatem podstawiając t minimalizujące ograniczenie górne do wzoru z twierdzenie 10 [ABP17] otrzymujemy

$$\Pi_{cc}^{\|}(RSG_\lambda^{\overline{n}}) = O\left(n\left(\sqrt{(\lambda n^{\frac{1}{3}} + n^{\frac{1}{3}} - \lambda - 1)n\delta} + \frac{n}{n^{\frac{1}{3}}}\right)\right) = O\left(n^{1.\overline{6}}\right)$$

□

Riffle Scrambler

Implementacja

Instalacja i wdrożenie

W tym rozdziale należy omówić zawartość pakietu instalacyjnego oraz założenia co do środowiska, w którym realizowany system będzie instalowany. Należy przedstawić procedurę instalacji i wdrożenia systemu. Czynności instalacyjne powinny być szczegółowo rozpisane na kroki. Procedura wdrożenia powinna obejmować konfigurację platformy sprzętowej, OS (np. konfiguracje niezbędnych sterowników) oraz konfigurację wdrażanego systemu, m.in. tworzenia niezbędnych kont użytkowników. Procedura instalacji powinna prowadzić od stanu, w którym nie są zainstalowane żadne składniki systemu, do stanu w którym system jest gotowy do pracy i oczekuje na akcje typowego użytkownika.

Podsumowanie

W podsumowanie należy określić stan zakończonych prac projektowych i implementacyjnych. Zaznaczyć, które z zakładanych funkcjonalności systemu udało się zrealizować. Omówić aspekty pielęgnacji systemu w środowisku wdrożeniowym. Wskazać dalsze możliwe kierunki rozwoju systemu, np. dodawanie nowych komponentów realizujących nowe funkcje.

W podsumowaniu należy podkreślić nowatorskie rozwiązania zastosowane w projekcie i implementacji (niebanalne algorytmy, nowe technologie, itp.).

Bibliografia

Zawartość płyty CD

W tym rozdziale należy krótko omówić zawartość dołączonej płyty CD.