1. cvičení – řešení úkolů

a)

Jak silné heslo potřebujete? Následující tabulka udává požadovanou entropii vytvořeného hesla v závislosti na počtu iterací při jeho hašování a v závislosti na předpokládaném útočníkovi.

Schopnosti útočníka (počet hašovacích operací)	**Uložení hesla**			
	5000 iterací	1000 iterací	1 iterace	5 mil. iterací
2^70 (NSA za rok)	58,5	61	71	49
2^58 (firma za půl roku)	46,5	49	59	37
2^52 (jednotlivec s GPU za měsíc)	41,5	43	53	31
2^42 (jednotlivec s CPU za týden)	31,5	33	43	21

Jak jsem spočítal tuto tabulku? První sloupec udává typ útočníka a odhad počtu operací kryptografického hašovacího algoritmu[1], které je schopen provést za uvedenou dobu.

Ve studii „Towards Reliable Storage of 56-bit Secrets in Human Memory“[2] z roku 2014 autoři odhadují, že NSA je schopna za 1 milion USD sestavit zařízení, které spočítá 2^{70} hašů za rok (předpokládají se operace SHA, SHA-2 či SHA-3)[3]. Při vynaložení miliardy USD se výkon zvýší na 2^{80} hašů za rok.

Ne každý je cílem NSA. Firma[4] či zločinecká skupina je schopna si sestavit či zakoupit speciální server pro lámání. Firma Sagita HPC prodává za necelých 20 tis USD server Brutallis, který je schopen za vteřinu spočítat 11 231 miliónů hašů algoritmu SHA-256. Za půl roku je to 2^58 hašů.

Pokud by Vaše heslo lámal jednotlivec pomocí grafické karty „AMD Radeon R9 290 Series“, tak je schopen za měsíc spočítat 2^{52} hašů SHA-256.

Na procesoru Core i7-2600K se za týden spočítá 2^{42} hašů SHA-256.

Počet vyzkoušených kombinací závisí i na způsobu uložení hesla – na počtu iterací hašovací funkce. Např. 1000 iterací snižuje požadovanou entropii o 10, 5000 iterací o 11,5. Aby pravděpodobnost uhodnutí byla menší než 50% za uvedenou dobu, tak přičteme k entropii hodnotu 1.

Svoje hesla ukládám do správce hesel Keepass. Databáze hesel se šifruje symetrickou šifrou, klíč pro šifru se odvozuje z hesla uživatele. Pro odvození klíče se používá stejný postup jako při ukládání hesel. Lze zvolit počet iterací – mám nastaveno 5 000 000 iterací.

---

[1] Obecné kryptografické hašovací funkce jako MD5, SHA1, SHA-2, SHA-3 mají podobnou rychlost. Rychlost specializovaných algoritmů pro ukládání hesel (bcrypt, scrypt, argon2, …) se výrazně liší, ale tyto algoritmy nejsou národními standardy, takže mohou bránit certifikaci aplikace.

[2] BONNEAU, Joseph; SCHECHTER, Stuart. Towards reliable storage of 56-bit secrets in human memory. In: *23rd USENIX Security Symposium (USENIX Security 14)*. 2014. p. 607-623.

[3] Je to odhad z roku 2014. Moorův zákon tvrdí, že každých 18 měsíců se výkon zdvojnásobí. Na základě toho by bylo možné odhadnout, že v roce 2020 za stejnou částku bude schopnost NSA okolo 2^74.

[4] Některé firmy preventivně prolamují hesla svých zaměstnanců. Pokud se to podaří, tak si zaměstnanec musí změnit heslo.

Úkol: dopočítejte minimální délky hesla pro jednotlivé entropie:

požadovaná entropie H v bitech	číslice 0-9	písmena [a-z] bez rozlišení velikosti	číslice, malá a velká písmena, [a-z, A-Z, 0-9]	tisknutelné znaky ASCII tabulky	diceware
	10	***26***	***62***	***95***	***7776***
43					
53					
59					
71					

Entropie se používá pro vyjádření variability. V informatice se obvykle vyjadřuje pomocí logaritmu o základu 2 (log_2). Používá se též pro vyjádření síly (obtížnosti prolamování) hesel. PIN o čtyřech číslicích lze sestavit 10 000 způsoby – na každém místě je 10 možností, vzájemně jsou nezávislé, takže možnosti vynásobíme mezi sebou: 10 * 10 * 10 * 10. tj. 10^4. Entropie se vypočítá jako log_2 počtu možností, tj.

$$log_2\, 10^4 = 13.28771$$

V Excelu vzorec vypadá takto:

=LOGZ(10^4;2)

Pokud je náhodný výběr a jednotlivé znaky v hesle jsou rozmístěny rovnoměrně[5], tak entropie (e) závisí na počtu prvků v množině (N) a na počtu prvků, které je mají vybrat (L):

$$S = log_2\, N^L = L\, log_2\, N$$

Z toho se již snadno odvodí vzorec na požadovaný počet znaků v hesle:

$$L = \frac{S}{log_2\, N}$$

V Excelu je to včetně zaokrouhlení nahoru (buňky v závislosti na umístění tabulky):

=ZAOKR.NAHORU($A3/LOGZ(G$2;2);1)

požadovaná entropie H v bitech	číslice 0-9	písmena [a-z] bez rozlišení velikosti	číslice, malá a velká písmena, [a-z, A-Z, 0-9]	tisknutelné znaky ASCII tabulky	diceware
	10	***26***	***62***	***95***	***7776***
43	***13***	***10***	***8***	***7***	***4***
53	***16***	***12***	***9***	***9***	***5***
59	***18***	***13***	***10***	***9***	***5***
71	***22***	***16***	***12***	***11***	***6***

[5] Porušením podmínky pravidla typu minimálně jedno písmeno a minimálně jedna číslice)

b)

InSIS generoval počáteční hesla dle schématu nnxxxnxxX (dvě číslice, tři malá písmena, jedna číslice, dvě malá písmena, velké písmeno). Jaká je entropie těchto hesel?

Schéma nnxxxnxxX – na prvním místě může být jedna z 10 číslice, na druhém opět jedna z deseti číslic, na třetím místě jedno z 26 malých písmen, atd. Počet kombinací je následující:

$$10 * 10 * 26 * 26 * 26 * 10 * 26 * 26 * 26 = 10^3 * 26^6$$

Hesla se generují náhodně, tj. všechny varianty jsou stejně pravděpodobné, proto lze pro výpočet entropie použít opět dvojkový logaritmus:

$$e = log_2\,(10^3 * 26^6) \approx 38.2$$

Kolikrát se zkrátí čas prolamování pro 9 znaková hesla proti variantě testování všech variant z 62 znaků ([a-zA-Z0-9])?

Entropie náhodně generovaného 9 znakového hesla z množiny 62 znaků je:

$$e = log_2\,(62^9) \approx 53.6$$

Rozdíl je 15 bitů, tj. čas se zkrátí přibližně $2^{15} \approx 32\,000$. Tj. pokud útočník otestuje všechny varianty hesla dle uvedeného schématu za jednu hodinu, tak náhodná 9 znaková hesla otestuje přibližně za 4 roky.

c)

Máte 8znaková náhodně generovaná hesla složená z malých/velkých písmen a číslic (minimálně jedno malé písmeno, jedno velké písmeno a jedna číslice). Chcete zvětšit odolnost vůči prolamování hrubou silou. Více variant (větší entropii) bude mít zvětšení minimálního počtu znaků na 9 či rozšíření abecedy o tisknutelné znaky ASCII tabulky (tj. na 95 znaků)?

Hrubý výsledek se získá z porovnání hodnot 62^9 (rozšíření na 9 znaků hesla) a 95^8 (všechny tisknutelné znaky).

Přesnější je ale počítat v první variantě, že musí být aspoň jedna číslice, aspoň jedno malé písmeno a aspoň jedno velké písmeno. O zbývajících 6 znacích nic nevíme, tj. může tam být libovolný z 62 znaků z množiny.

$$10 * 26 * 26 * 62^5$$

Podobně je potřeba upřesnit výpočet pro druhou variantu:

$$10 * 26 * 26 * 95^4$$

Pokud si ale heslo vybírá sám uživatel, tak se obvykle množina 33 nepísmenných a nečíselných znaků zmenší na 6 až 10 speciálních znaků (znaky tečka, plus, mínus, …). Zvlášť v českém prostředí, kdy uživatelé musí přepínat mezi klávesnicemi či si pamatovat Alt a číselnou kombinaci (např. pro znak @).

d)

Následující tabulka obsahuje rychlost prolamování některých typů hesel pomocí CPU i pomocí GPU:

	John the Ripper, Intel i7 2600	**oclhashcat, AMD Radeon HD 7970**
DEScrypt	18 284K/s	65 594K/s
MD5 crypt	66,9K/s	3 592K/s
Bcrypt,	4,8K/s	4,1K/s
LM hash	88 834K/s	2 384M/s

Za jak dlouho by program oclhashcat otestoval všechny možné varianty počátečních hesel ze zadání b) na grafické kartě AMD Radeon HD 7970, pokud jsou hesla uložena pomocí LM hash?

Toto vede k velmi jednoduchému výpočtu – počet variant se vydělí rychlostí, tj.

$$\frac{(10^3 * 26^6)}{2384000000}$$

Výsledek je přibližně 130 sekund [6].

Jak dlouho by to trvalo, pokud by hesla byla uložena pomocí algoritmu Bcrypt?

Opět je výpočet velmi jednoduchý:

$$\frac{(10^3 * 26^6)}{4100}$$

Výsledek je přibližně 872 dnů, tj za 2 roky a téměř 5 měsíců získá všechny varianty. Ale jen pro jednu sůl. Pokud by útočník získal haše 600 nových uživatelů, tak v případě LM hash získá za 2 až 3 minuty hesla všech uživatelů (LM hash nemá sůl). Pokud budou hesla uložena pomocí Bcrypt, tak jedno heslo získá v průměru za 436 dní. Všech 600 hesel by získal za téměř 1000 roků.

e)

Spočtěte průměrnou dobu pro prolomení jednoho náhodně vygenerovaného hesla uloženého pomocí MD5crypt, pokud útoční použije program oclhashcat a AMD Radeon HD 7970:

entropie hesla	doba louskání ve vteřinách	doba louskání ve dnech
16		
32		
40		
48		

Budou se lišit výsledky v případě, že heslo je uloženo se solí o délce 48 bitů?

To jsou opět jednoduché výpočty. Entropie je mocnina o základu 2, takže např. všechny varianty hesel s entropii 16 se otestují za

$$\frac{2^{16}}{3592000}$$

tj. asi 0,018 vteřiny. V zadání se ale mluví o průměrné době pro prolomení – výsledek je potřeba vydělit dvěma. Některé heslo se najde hned a některé až téměř po prozkoumání téměř všech kombinací.

Sůl na tyto výpočty nemá vliv – v zadání se mluví o prolomení jednoho náhodně vygenerovaného hesla.

Situace se změní, pokud by se mělo prolamovat např. 100 hesel. Pokud se nepoužije sůl, tak všechna hesla se najdou za 0,018 sekundy. Pokud by pro každé heslo byla jiná sůl, tak na prolomení každého hesla potřebujete v průměru 0,009 sekundy a všechny budete mít za 0,9 sekundy.

[6] Prolamování by bylo ještě rychlejší – v algoritmu LM hash je maximální délka hesla 14 znaků, pokud je heslo kratší, tak se doplní binárními nulami. Dále se všechna písmena hesla převedou na velká – to nám v našem příkladu nepomůže, neboť víme dopředu velikost písmen. Následně se heslo rozdělí na dvě poloviny po 7 znacích. Každá polovina má délku 56 bitů, která se použije jako klíč pro šifrovací algoritmus DES pro zašifrování řetězce „KGS!@#$%”. Výsledky se spojí do jednoho výsledného řetězce. Pokud se toto vezme do úvahy, tak celková doba prolamování se bude pohybovat okolo vteřiny.

f)

Entropie hesel vytvářených uživateli se měří poměrně špatně. Knihovna zxcvbn odhaduje entropii na základě existence částí hesla ve slovnících nejčastějších hesel plus zohledňuje „běžné“ operace obsažené v nástrojích pro lámání hesel. Přidává entropii za použití velkých písmen či substituci znaků (např. o se nahradí číslicí 0). Odebírá za posloupnosti či za opakování stejných znaků či skupin znaků.

Heslu poté přiřadí skóre dle následující tabulky:

Skóre	Skóre text	Entropie od	Entropie do
0	velmi slabé	0	< 10
1	slabé	10	< 20
2	průměrné	20	< 27
3	silné	27	< 33
4	velmi silné	33	

Ve vytvářené aplikaci hesla před uložením do databáze hašujete pomocí PBKDF2 s funkcí HMAC-SHA256 (v každé iteraci se provádějí dvě operace SHA-256). V pravidlech pro hesla požadujete minimální skóre 3. Předpokládáte, že v systému bude uloženo přibližně 20 000 účtů. Vašim úkolem je spočítat počet iterací pro PBKDF2 tak, aby se výrazně snížilo nebezpečí odhalení hesel při ukradení celé databáze. Předpokládáte, že útočník pro prolamování použije počítač s grafickou kartou AMD HD 7970, na které je schopen spočítat 1 032 milionů SHA-256 haší za sekundu (**1 032 MH/s**). Za jeden den by měl útočník získat v průměru jedno heslo.

Útočník za den provede

$$\left(\frac{1\,032\,000\,000}{2}\right) * (60 * 60 * 24) \approx 4,4 * 10^{13}$$

operací HMAC-SHA-256 (jedna operace HMAC obsahuje 2 operace SHA-256). Zadávaná hesla mají minimální entropii 27, tj. jako by uživatelé vybírali heslo z množiny 2^{27} variant. Potřebný počet iterací se spočte již jednoduše:

$$\frac{4,4 * 10^{13}}{2^{27}} \approx 332\,165$$

Tj. nastavit 332 165 iterací. Předpokládáme, že každé heslo má svoji vlastní dostatečně dlouhou sůl.

Všechna hesla nebudou mít entropii 27, průměr odhaduji okolo 33. Bylo by možné použít toto číslo ve výpočtu (potřebný počet iterací se sníží na přibližně 10 tisíc), ale dostávali bychom „falešné“ výsledky. Stále by se našlo v souboru dostatek hesel s entropii 27 a tudíž první dny/týdny by útočník získal velké množství hesel. Pokud by v souboru byly 5% hesel s entropii 27 a útočník se rozhodne hádat heslo někoho z těchto uživatelů, tak ho získá v průměru za 23 minut.

Pokud administrátoři musí mít velmi silná hesla (skóre 4), tak při 332 165 iteracích získá útočník jedno takové heslo přibližně za n[7] dní.

Pokud bychom chtěli, aby útočník mohl získat jednoho heslo za 90 dní, tak je potřeba počet iterací vynásobit 90. Tj. přibližně 30 milionů iterací.

[7] *n si spočítejte.*

g)

Pokračování předchozího zadání. Kolik uživatelů se bude moci za sekundu přihlásit na Vašem serveru se čtyřmi jádry Intel Xeon E3-1230? Tento procesor zvládá 36 miliónů SHA256 operací za vteřinu na jednom jádru.

Pokud v PBKDF2 nastavíte 664 330 iterací, tak jedno jádro za sekundu ověří heslo 54 uživatelům.

$$\frac{(36\,000\,000\ /\ 2)}{332\,165} \approx 54$$

Výsledek nelze automaticky rozšířit na 4 jádra – procesory musí zajistit i další operace, nějaký čas zabere režie. Odhaduji, že server by byl schopen ověřit 100 hesel za vteřinu.

Pokud by bylo nastaveno 30 mil. iterací, tak jedno jádro ověří uživatele přibližně za téměř 2 vteřiny[8]. při zapojení dvou jader jednoho uživatele za vteřinu. To může být v některých případech málo. Např. by server sloužil pro přijímací testy, na test dorazí 350 studentů. Tj. jen přihlášení by studentům trvalo 6 minut – systém by byl označen za nepoužitelný pro toto užití.

Jaké by byly možnosti delší odolnosti uložených hesel:

- ***vyžadovat velmi silná hesla,***
- ***změnit způsob uložení hesel. Problém vzniká z velkého rozdílu rychlosti hašování v CPU a na grafické kartě či specializovaných obvodech. Některé algoritmy tímto problémem netrpí – bcrypt (viz tabulka u bodu d), scrypt či argon2. Argon2 je vítěz soutěže hledání nejvhodnější hašovací funkce pro uložení hesel, tudíž v nových systémech se doporučuje použití této funkce.***

Zadání na entropii

i)

Při vytváření virtuálních serverů ve VMware se náhodně generuje ethernetová adresa (6B). První tři bajty jsou dány (00:50:56), další tři se náhodně generují. Jaká je pravděpodobnost kolize MAC adres v podsíti s maskou /22 (1022 počítačů).

Následující popis převzat z české Wikipedie:

V teorii pravděpodobnosti je narozeninovým problémem (či narozeninovým paradoxem) myšleno, že u skupiny náhodně vybraných 23 lidí je více než 50% pravděpodobnost, že nějací dva členové budou mít narozeniny ve stejný den. Pro 57 a více lidí je ona pravděpodobnost více než 99%, postupně roste až k 100% pro 366 lidí (za předpokladu že pracujeme s rokem o 365 dnech).

Výpočet. Je jednodušší nejprve spočítat pravděpodobnost, že všech „n" narozenin je rozdílných. Pro „n" > 365 je tato pravděpodobnost rovna nule. Pro „n" ≤ 365 je dána následujícím vzorcem. Ten vychází z toho, že druhá osoba nemůže mít stejné narozeniny jako první (pravděpodobnost $^{364}/_{365}$, kterou lze vyjádřit i jako $1 - {}^{1}/_{365}$), třetí nemůže mít stejné narozeniny jako první dvě (pravděpodobnost 363/365), atd. Vzorec:

$$\bar{p} = 1 * \left(1 - \frac{1}{365}\right) * \left(1 - \frac{2}{365}\right) * \ldots * \left(1 - \frac{n}{365}\right)$$

Skutečnost, že nejméně dva z „n" lidí mají stejné narozeniny je komplementární jevu, že všechny data narozenin jsou různé. A její pravděpodobnost p je

$$p = 1 - \bar{p}$$

---

[8] *Spočtěte sami.*

Proměnná p překračuje hodnotu 0,5 pro $n = 23$. Pro 23 osob je pravděpodobnost kolize (tj. že dva lidi mají narozeniny ve stejný den) $p \approx 50,7\%$.[9] Pokud ze sady o velikosti H vybereme n prvků, tak pravděpodobnost kolize lze spočítat dle následujícího přibližného vzorce:

$$p \approx 1 - e^{-n^2/2H}$$

V našem příkladu za H doplníme 2^{24} (poslední tři byty MAC adresy mají 24 bitů), za n doplníme 1022. Pravděpodobnost kolize je 3,06 %.

j 2015)

Předpokládejte, že 20 000 uživatelů na škole si vygeneruje PGP klíč. Běžně se tyto klíče identifikují pomocí ID o délce 32 bitů (např. 0xB9F8D6D9), které má náhodné rozdělení. Jaká je pravděpodobnost, že dva uživatelé na škole budou mít u svého klíče stejné ID?

Do přibližného vzorce z předchozího příkladu za H doplníme 2^{32}, za n doplníme 20 000. Výsledek je 4,55%.

j 2016)

Budete ukládat 20 000 hesel, pro každé uložené heslo vygenerujete náhodnou sůl. Kolik bitů má mít sůl, aby pravděpodobnost stejné soli (tj. kolize) byla menší než 1%?

Pro řešení úlohy jsem v excelu vytvořil následující tabulku, ve kterém používám přibližný vzorec pro výpočet pravděpodobnosti kolize: **=1-EXP(-(20000^2)/(2*počet_variant))**

bitů	variant	pravděpodobnost kolize 2 prvků
15	32768	100,00000%
16	65536	100,00000%
17	131072	100,00000%
18	262144	100,00000%
19	524288	100,00000%
20	1048576	100,00000%
21	2097152	100,00000%
22	4194304	100,00000%
23	8388608	100,00000%
24	16777216	99,99934%
25	33554432	99,74213%
26	67108864	94,92190%
27	134217728	77,46535%
28	268435456	52,52933%
29	536870912	31,10103%
30	1073741824	16,99460%
31	2147483648	8,89270%
32	4294967296	4,54986%
33	8589934592	2,30141%
34	17179869184	1,15740%
35	34359738368	0,58039%

[9] *Jaká je pravděpodobnost, pokud mám na cvičení 25 studentů?*

36	68719476736	0,29062%
37	1,37439E+11	0,14541%
38	2,74878E+11	0,07273%
39	5,49756E+11	0,03637%
40	1,09951E+12	0,01819%
41	2,19902E+12	0,00909%
42	4,39805E+12	0,00455%
43	8,79609E+12	0,00227%
44	1,75922E+13	0,00114%
45	3,51844E+13	0,00057%
46	7,03687E+13	0,00028%
47	1,40737E+14	0,00014%
48	2,81475E+14	0,00007%

Tj. u soli s minimálně 35 bity je pravděpodobnosti existence kolize menší než 1%. V praxi se doporučuje sůl o délce 64 bitů. Algoritmus bcrypt používá sůl o délce 128 bitů.

V tabulce začínám u hodnoty 15 bitů – 2^{15} je 32768 a to je první číslo větší než počet uložených hesel.

Nejmenší množství n vybraných prvků k dosažení p pravděpodobnosti kolize spočítáme dle následujícího přibližného vzorce:

$$n \approx \sqrt{2H * ln\frac{1}{1-p}}$$

Pro přístupový systém uživatelům generujete náhodný PIN (číselná hesla) o délce 4 číslic. Kolik PINů musíte vygenerovat, aby pravděpodobnost kolize byla 95% (tj. že dva uživatelé mají stejný PIN).

Do uvedeného vzorce doplníme H=10000 a p=0,95. Výsledek je 245 PINů. Pro 50% pravděpodobnost stačí 118 PINů[10].

[10] V tomto semestru je na předmětu 126 studentů – pokud by se všichni shromáždili na přednášce a každý navrhl náhodný PIN, tak pravděpodobnost, že se najdou dva návrhy se stejným PINem je téměř 55%. Lidi nejsou vhodné generátory náhodných čísel a též se vzájemně ovlivňují – znalci behaviorální ekonomie mohou odhadnout, zda se zvýší pravděpodobnost kolize či sníží.