Grid Implemented John the Ripper aka GI John

Bucsay Balázs
<balazs.bucsay [at] rycon [dot] hu>
BB$HT

Magamról

- 2004-től foglalkozom IT-Securityval
- 2008 PR-Audit alkalmazottja (http://www.praudit.hu)

- 2009-től okleveles Programtervező Informatikus BSc (ME-GÉIK)
- 2009-től Matematikus MSc (DE-TTK)

GI John

- Jelszótörő
- Platformfüggetlen
- Elosztott
- Skálázható
- Könnyen kezelhető

John the Ripper 1/2

- Solar Designer (Alexander Peslyak)
- Rengeteg saját fejlesztésű technika, publikáció
- 2009. Életműdíj (Black Hat Security Conference)
- John the Ripper, 1996-máig

John the Ripper 2/2

- nyíltforrású
- platform független
- szabadon módosítható
- modularizált, rengeteg ismert kripto hash algo.
- rengeteg opció
- effektív
- karbantartott (máig is)

Igény a gyorsaságra

- Szükséges hogy jelszavak gyorsan meglegyenek
- Egy gép már kevés ehhez, egy erősebb algoritmusnál
- Elosztottság a megoldás
- De mégis mit használjon az ember?

Megvalósítások 1/3

- ElcomSoft: Distributed Password Recovery
- Hátrányai:
  - Csak Windows alapú
  - Zárt forrású
  - Fizetős
  - Kevés kriptográfiai hash algoritmus

Megvalósítások 2/3

- DJohn, egy John the Ripper kiegészítő:
- Hátrányai:
  - Nem megfelelően tesztelt
  - Rengeteg implementációs és programozási hiba
  - Nincs optimalizálva
  - Nem követett adatközlés, adatvesztés
  - Nehézkes kezelés
  - Emlékezetmentes, nincs vagy kezdetleges archívum a hasheknek

Megvalósítások 3/3

- CPUShare/MPI megoldások
- Hátrányok:
  - Nehéz kezelhetőség
  - Keretrendszer szükséges

Felsorakozatott igények

- Gyors implementáció (ne kelljen sokat a fejlesztéssel foglalkozni)
- Optimalizált kód
- Egyszerű használhatóság
- Extra libraryk nem szükségesek (libcurl, libxml)
- Lehető legtöbb opció
- Bővíthető, minél több algoritmus ismerete

John Torrent

- 2005, Hacktivity bejelentés
- Páncél Zoltán, Buherátor
- Teljesen sajátkezű, web management
- Félig nyíltforrás
- Sosem készül el teljesen

GI John

- ~3év várakozás, 2008 nyara a project kezdete
- 2008-as Hacktivity-n próba a bemutatására
  - közel használható, de még használhatatlan
  - nincs webes frontend és köztes állapotban a kommunikáció
- 2009. február - 1. hivatalos, használható verzió

Igények VS. GI John

- Gyors implementáció (ne kelljen sokat a fejlesztéssel foglalkozni)
- Optimalizált kód
- Egyszerű használhatóság (cli, web)
- Extra libraryk nem szükségesek (libcurl, libxml)
- Lehető legtöbb opció, lehetőség
- Bővíthető, lehető legtöbb algoritmus ismerése

Kliens rész

- C-ben írt patch a John the Ripper kódjához
- Ebből fakadóan nyíltforrású
- Szabadon módosítható (GPLv2)
- Nincs extra library, ahol a John fordul ott a GI John patch is!

Server rész

- FAMP (FreeBSD+Apache+MySQL+PHP) jelenleg
- +c-ben írt crypto hash extension
- Könnyen kezelhető
- PHP rész nem nyíltforrású
- Angol felület, FAQ
- http://gijohn.info / https://gijohn.info

Szótár

- hashcsoport
- salted/sózott hash
- kezdő ill. véghossz
- karakterkészlet
- kulcstér
- részkulcstér

Bemutató

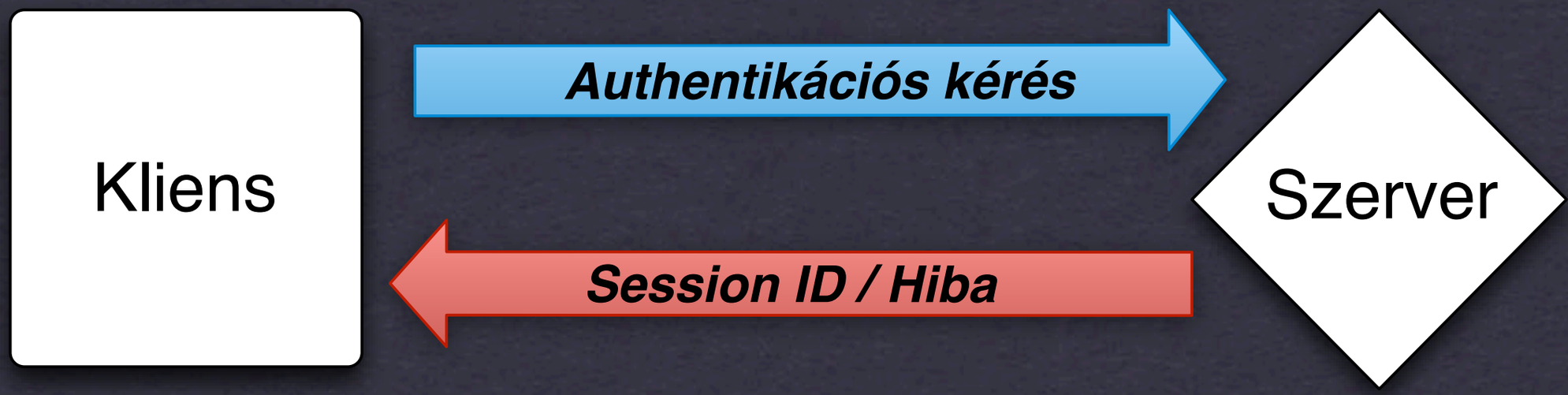
Pontozási rendszer 1/2

- Igazságosnak készült
- Teljesítmény és nem idő függő
- Etalon:
  - AMD Athlon X2 4200+ DualCore (1mag)
  - Raw-MD5 algoritmus

Pontozási rendszer 2/2

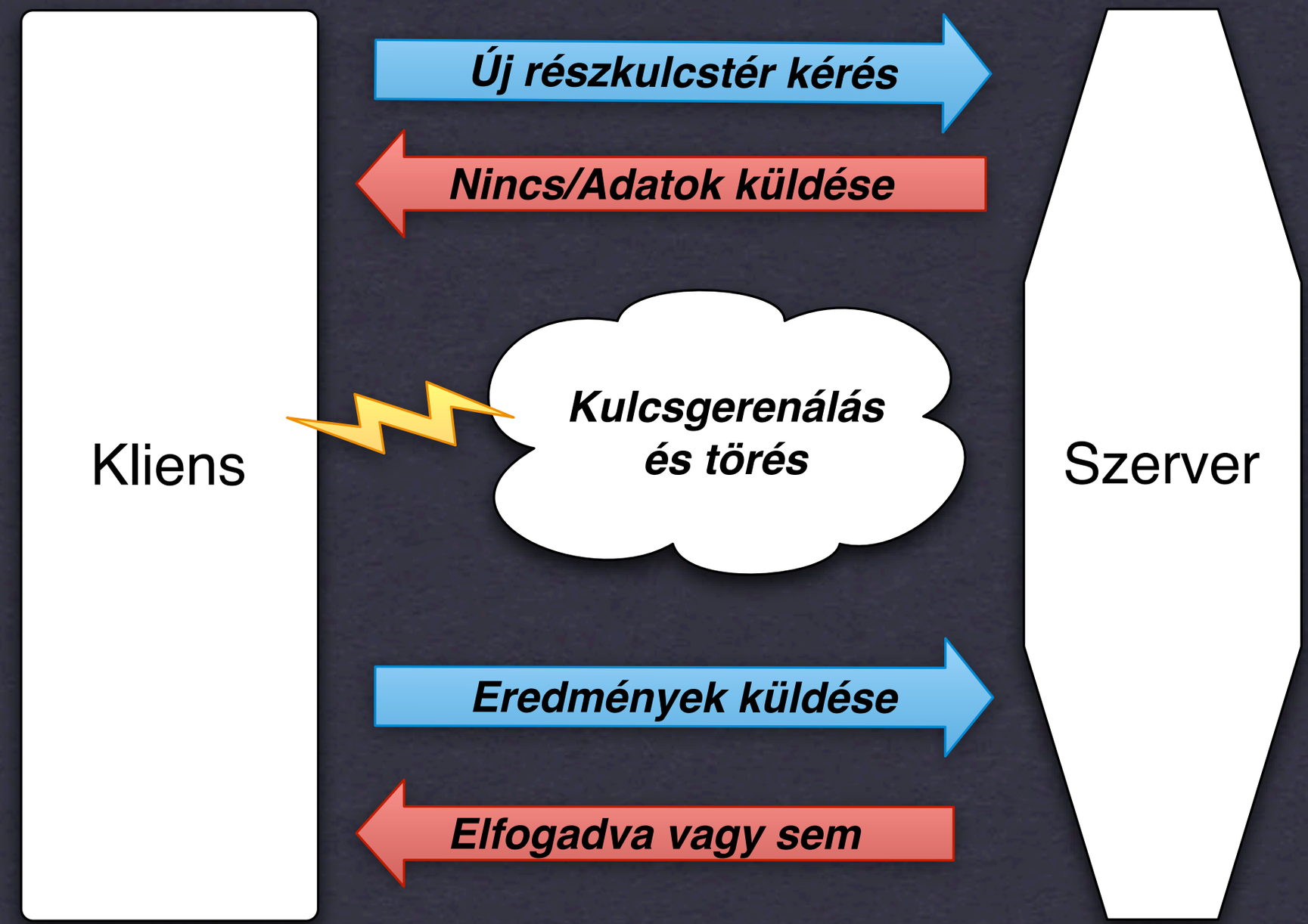
- 1óra munka: +1pont
- 1talált hash: algoritmusnak megfelelő súly érték (raw-md5 relatív)
- prioritás ára: -1000pont
- hashcsoport feltöltésének ára függ:
  - algoritmus gépigényétől
  - sózott illetve nem sózott
  - implementálás módja (lineáris keresés, hash táblák)

Technikai részletek



KOMMUNIKÁCIÓ 1/2

AUTHENTIKÁLÁS HTTP/XML



KOMMUNIKÁCIÓ 2/2

AUTHENTIKÁLT KOMMUNIKÁCIÓ HTTP/XML

Részkulcstér adatai

- A lényegi kérés válaszának tartalma:
  - kezdő és vég kulcs
  - karakterkészlet
  - formátum azonosítója
  - flag/flagek
  - törendő hashek (hozzáadás)
  - megtört hashek (törlés)

Kulcsgenerálás

- Kliens megkapja a kezdő és vég kulcsot
- Átváltja a karakterkészletnek megfelelő számrendszerre
- Kiszámolja a távolságot
- Ennek megfelelő iterációban legenerálja a kulcsokat
- Közel a leggyorsabb generálási algoritmus (GI John)
- DJohn?

Hashek tárolása

- A hashek SQL backendben tárolódnak
- Felhasználónév feltöltéskor törlődnek
- Formális ellenőrzés (regexp)
- Feltöltéskor kikeresi a hash-t:
  - Találatkor rögtön írja a plaintextet
  - Ha nincs találat, akkor eltárolja
  - Duplikációkat nem tárol
- Archiválás, a tulajdonos bármikor hozzáfér

Törési sorrend

- 0. kör: kliensek ismereti köre (algoritmusok)
- 1. kör: 0. körnek megfelelően a lejárt illetve befejezetlen részek kerülnek kiosztásra
- 2. kör: legmagasabb prioritású hashcsoport kerül kiosztásra
- 3. kör: feltöltési sorrend szerint kerül sorra

Nyomonkövetés

- Minden kiosztott részkulcsteret tárol a rendszer
- A részek 12 órás lejárati időt kapnak
- Ha egy kliens kilép, vagy lejár a 12 óra akkor újra szabad lesz a rész, kiosztásra kerül
- Veszteségmentes!

Csalás/Spoof 1/2

- Minden részkulcstérben generálódik egy fake-hash
- Ezt a fakehasht meg kell törni és visszajuttatni
- Ez a hash az intervallumban pszeudó véletlen szerűen generálódik
- Visszakapott, helyes hash után kap csak pontot
- Nem 100%-os, de a kliensek törnek!

Csalás/Spoof 2/2

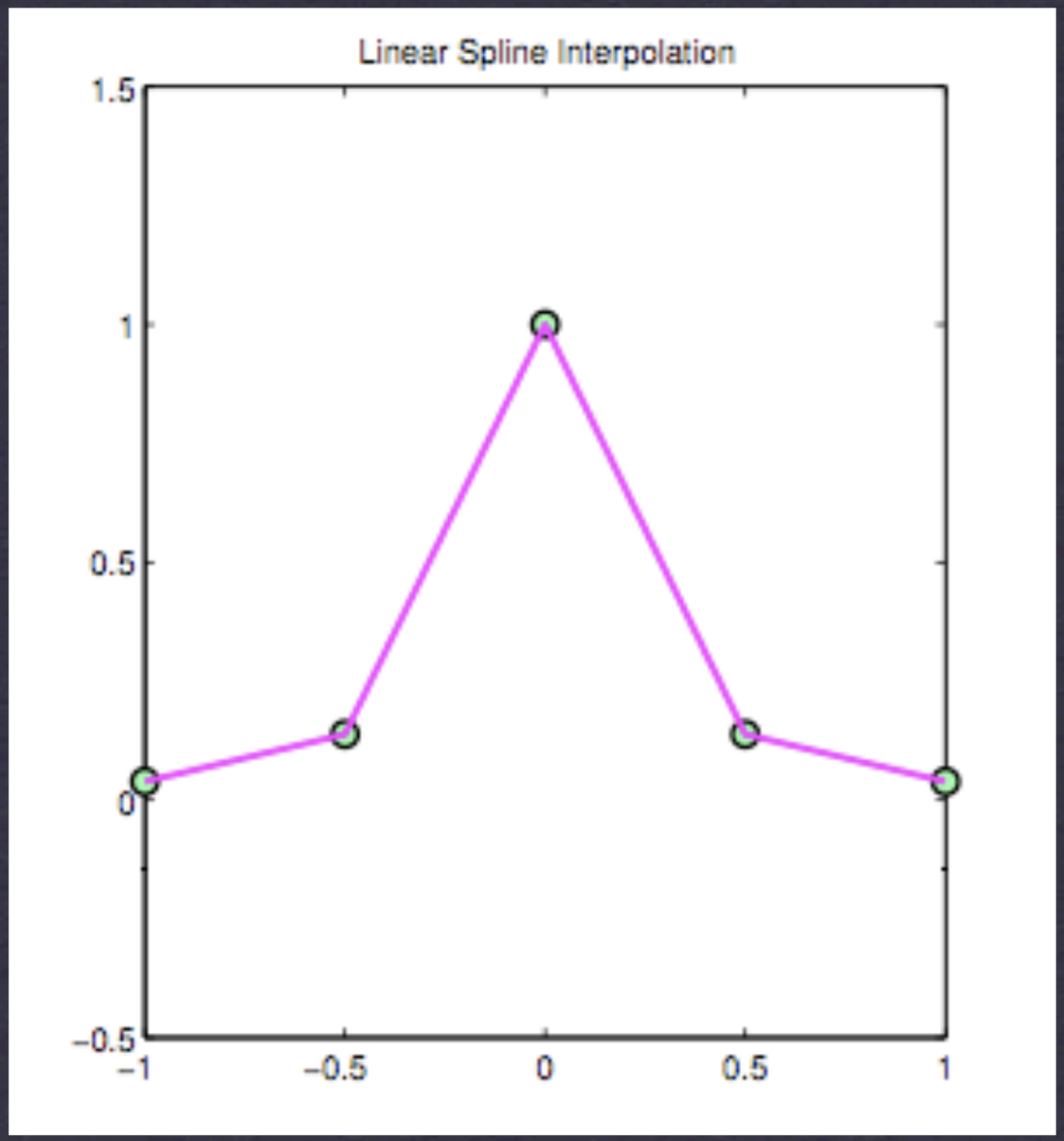
- Feltöltött nem fake hashek több ellenőrzésen mennek keresztül:
  - Érték ellenőrzés, validitás
  - Formátum ellenőrzés, formalizmus
  - Létezés, egzisztencia
- Helyesség esetén kapható csak pont

Skálázhatóság

- Minden érték dinamikus benne
- Adatbázisból dolgozik
- A kliensek teljesítmény növekedésével könnyen fel lehet venni az ütemet
- Hash algoritmusok optimalizálásával ugyanígy

Kiterjeszthet**ő**ség

- Dehát nem támogat GPU-t?!
- Könnyedén írható framework
- Támogatható vele bármi:
  - A rendszer univerzális
  - Nyílt forráskód



$$S(x) = \begin{cases} S_0(x), & x_0 \leq x \leq x_1 \\ S_1(x), & x_1 \leq x \leq x_2 \\ \vdots & \vdots \\ S_{n-1}(x), & x_{n-1} \leq x \leq x_n \end{cases}$$

$$m_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

$$S_i(x) = y_i + m_i(x - x_i)$$

O(N) VS O(N/K)

LINEÁRIS SPLINE INTERPOLÁCIÓ

Példa (FreeBSD MD5)

24-es karakterkészleten, 1-től 7-ig hosszon:
4 785 883 224 kulcs
478 588 mp
7 976 perc
132 óra
5 nap

54-es karakterkészleten, 1-től 7-ig hosszon:
1 364 187 949 794 kulcs
136 418 794 mp
2 273 646 perc
37 894 óra
1578 nap

100géppel/maggal/hyperthread akkor csak ~16nap!

Köszönet nyílvánítás

Kérdések

- Mennyire is biztonságos egy kriptográfiai hash algoritmus?
- Szükséges-e félni attól, hogy holnap feltörik a jelszavam?

Vége

Közönség kérdései?

GI John

http://gijohn.info | https://gijohn.info

<balazs.bucsay [at] rycon [dot] hu>