Lehren für die Passwort-Sicherheit aus dem LinkedIn-Leak

Die gestohlenen oder „durchgesickerten" Hashwerte von Millionen LinkedIn-Nutzerpasswörtern haben wieder einmal ausführliche Analysen des „Real-World"-Verhaltens von Anwendern und der damit verbundenen Bedrohungslage ermöglicht.

Von Francois Pesce, Redwood Shores (US/CA)

Im Juni 2012 haben Hacker fast 6,5 Millionen SHA-1-Hashwerte für Passwörter von LinkedIn-Accounts veröffentlicht. Anschließend begann der übliche „Run", um möglichst schnell möglichst viele verwendete Passwörter aufzudecken – teils haben Community-Analysten ihre Ergebnisse in Echtzeit gebloggt (vgl. z. B. [2]). Wie zu erwarten, waren auch bei diesem „professionellen" sozialen Netz viele schwache Passwörter schnell zu finden. Hier soll es jedoch nicht vorrangig darum gehen, statistisch nachzuweisen, wie viele schwache oder zu kurze Passwörter (wieder einmal) Verwendung fanden, sondern vielmehr anhand von – eher einfachen – Analysestrategien zu zeigen, was auch an Komplizierterem alles zu finden ist.

Bei etwa 3,5 Millionen der LinkedIn-Hashes waren die ersten fünf Stellen mit Nullen überschrieben – möglicherweise handelt es sich um Werte, für die der oder die ursprünglichen Hacker bereits vor der Preisgabe an die Community die zugehörigen Passwörter identifiziert hatten, wurde spekuliert. So fänden sich beispielsweise keine vollständigen Hashes für „password" und einige andere Trivialpasswörter in der Liste, entsprechend verkürzte Hashes seien jedoch präsent. Andere Quellen widersprechen dieser Theorie und führen an, dass ein beträchtlicher Teil der „genullten" Hashes vermutlich Duplikate von weiterhin präsenten Werten seien. Für gängige Analysetools waren jedenfalls in Bälde entsprechende Patches verfügbar, um auch die verkürzten Werte bearbeiten zu können.

Nachdem die OpenWall-Community einen entsprechenden Patch für „John the Ripper" (JTR) bereitgestellt hatte, hat sich auch der Autor dieses Beitrags – nach einiger Zeit der „Abstinenz" von JTR – näher mit den Hashwerten und der aktuellen „Community-enhanced Jumbo Version" des Open-Source-Tools befasst (www.openwall.com/john/), die zur Leistungssteigerung übrigens auch Grafikprozessoren nutzen kann.

Diese Software geht generell einen klassischen Weg: Passwort raten, SHA-1-Hash davon bilden und anschließend prüfen, ob der errechnete Wert in der Referenzdatei erscheint. Dabei kann JTR auf Wörterbücher (Dictionary-Attack) zurückgreifen und enthaltene Wörter regelbasiert modifizieren, um seine Versuche zu erweitern – das vollständige Durchprobieren (Brute Force) eines vorgegebenen Suchraums ist auch möglich, aber naturgemäß für größere Zeichenvorräte und Passwortlängen sehr rechenzeitintensiv.

Ripped by John

Im konkreten Fall hat der Autor einen älteren Computer ohne Grafikprozessor (GPU) oder vorberechnete Werte (Rainbowtables) benutzt und schlicht die „gute alte" Wörterbuch- und Regeln-Methode verwendet. Am Rande sei bemerkt, dass hierbei das fehlende kryptografische „Salz" (Salting) in den Hashwerten die Berechnungen deutlich erleichtert hat. Im ersten Anlauf lieferten selbst die einfachen Default-Regeln von JTR (26 Zeilen ohne Kommentare) auf der Basis eines kleinen Default-Dictionaries (unter 4000 Wörter) sowie ein anschließender Brute-Force-Anlauf nach rund 4 Std. ungefähr 900 000

Passwörter. Die eingesetzten Default-Regeln waren dabei so simpel wie etwa „ergänze eine 1 am Anfang/am Ende jedes Wortes“. Beim späteren „Incremental Mode“ kam auch ein umfassenderer Satz von rund 550 Regeln zum Einsatz, der in den späten Neunzigerjahren zum Knacken von DES-Passwörtern entwickelt wurde und versucht, eine optimierte Reihenfolge von Brute-Force-Attacken zu verwenden, sodass wahrscheinlichere Passwörter zuerst getestet werden.

Eine weitere Analyse mit einer Serie älterer, beim Autor vorhandener Wörterbücher – von einer 16-KByte-Liste häufiger Passwörter bis zu einem 40-MByte-Dictionary mit Wörtern aus allen existierenden Sprachen – erwies sich als sehr effizient und deckte in weniger als einer Stunde weitere rund 500 000 Passwörter auf. Auffällig war dabei, dass die bereits zehn Jahre alten Wörterbücher, in denen neue Begriffe, wie etwa „linkedin“, überhaupt nicht vorkamen, aufgrund der Regeln dennoch gute Fortschritte ermöglichten.

Recursive, see Recursive

Im nächsten Schritt begann dann eine interessante Rekursion: Durch Abwandlung bereits gefundener Passwörter sollten sich doch weitere, komplexer „verunstaltete“ Passwörter finden lassen?! Und so war es auch: Mit den bereits gefundenen 1,4 Mio. Passwörtern als Dictionary-Basis lieferte JTR weitere 554 404 – in der nächsten Iteration noch einmal 22 688 Passwörter, bis zur zehnten Iteration kamen noch einmal über Fünftausend hinzu.

Ein beträchtlicher Anteil der komplizierteren Passwörter in der dritten und den weiteren Iterationen erwies sich dabei als Modifikation von „linkedin“ – etwa durch Einsetzen von Einsen oder Ausrufezeichen für die „i“s (l!nked1n) oder rückwärts schreiben. Als zwei Beispiele seien hier genannt:

——— „m0c.nideknil“, das im siebten Durchlauf entdeckt wurde, und
——— „lsw4linkedin“, das einzige Ergebnis der zehnten Iteration.

Um es noch einmal zu betonen: „linkedin“ war nicht einmal Teil des ursprünglichen Wörterbuchs. Zu dem letztgenannten Passwort wurschtelte sich JTR in den verschiedenen Iterationen durch Regelanwendung ausgehend vom Wörterbucheintrag „pwlink“ durch: über „pwdlink“, „pwd4link“, „pwd4linked“, „pw4linked“, „pw4linkedin“, „mpw4linkedin“, „mw4linkedin“, „smw4linkedin“ und „sw4linkedin“ bis hin zum „lsw4linkedin“. Hier wurde also jeweils von gefundenem Passwort zu gefundenem Passwort nur ein einzelner Buchstabe oder eine Endsilbe ergänzt oder weggelassen.

Wikipedia attacks

Eine ebenfalls hervorragende – und weithin unterschätzte – Quelle für aktuelle Wörterbuchangriffe ist im Übrigen die Artikelliste der Wikipedia, die sich über http://dumps.wikimedia.org/*XX*wiki/latest/*XX*wiki-latest-all-titles-in-ns0.gz (mit dem gewünschten zweibuchstabigen Länderkürzel statt *XX*) abrufen lässt. Beispiele für Passwörter, die im LinkedIn-Fall via Wikipedia-Listen gefunden wurden, sind: „StrangerInAStrangeLand“, „xenathewarriorprincess“, „from genesis to revelations“, „thelightshinesinthedarkness“, „in the beginning was the word“, „Jantje zag eens pruimen hangen“ und „savethecheerleadersavetheworld“.

Die leicht erreichbare Auflistung von Namen, vermeintlich „exotischen“ Begriffen sowie Satz-

fragmenten, die unter anderem in Liedtexten, Film- oder Buchtiteln erscheinen, macht all diese gut merkbaren Zeichenfolgen (und ihre Abwandlungen) unbrauchbar für den Einsatz in Passwörtern: Auch wenn sie lang sind, kann ein Tool sie mithilfe der Wiki-Liste „erraten". Gerade die deutschsprachige Wikipedia mit ihrer großen aktiven Nutzerbasis und Seitenzahl liefert hier reichlich Material, das immer „up to date" ist. Doch auch andere große Onlinequellen helfen dabei, auf lange „Passphrases" zu schließen: Qualys hat beispielsweise für den „Crack Me If You Can"-Contest zur Defcon 2012 mithilfe populärer Bücher aus dem Gutenberg-Projekt (www.gutenberg.org) ein Wörterbuch der Wendungen und Satzteile zusammengestellt.

Numerologie

Obwohl ein rein numerisches Passwort sich eigentlich schon durch den geringen Suchaufwand bei Brute-Force-Angriffen verbieten sollte, scheint das einen beträchtlichen Teil der Nutzer dennoch nicht davon abzuhalten, solche Ziffernfolgen zu verwenden. Im Zuge der Analyse wurden vom Autor rund 200 000 Hashes von rein numerischen Passwörtern identifiziert – knapp die Hälfte davon enthielt sechs Ziffern, ein gutes Viertel acht Ziffern [1].

Bei einer näheren Betrachtung zeigte sich zudem, dass diese Ziffern beileibe nicht zufällig gewählt wurden: Zum einen gab es etliche Wiederholungen von kürzeren Abfolgen zu beobachten (also bspw. 313131, 424242 oder 36713671). Man darf dabei getrost vermuten, dass ähnliche „Verlängerungsmethoden" auch bei alphanumerischen Zeichenfolgen zum Einsatz kommen – für die Verdreifachung von zwei Kleinbuchstaben wurden zum Beispiel in der aktuellen LinkedIn-Sammlung 474 von 676 möglichen Kombinationen tatsächlich gefunden (wie z. B. ababab).

Zum anderen ließ sich eine klare Vorliebe für Ziffernfolgen erkennen, die Datumsangaben darstellen – sowohl in den Formaten DDMMYY und MMDDYY als auch mit vierstelliger Jahreszahl. Und diese Vorliebe lässt sich ebenfalls auf ausgeschriebene Datumsangaben übertragen: Nicht wenige Menschen nutzen offenbar Zeichenfolgen wie „November 4, 2011", um Groß- und Kleinschreibung, Satzzeichen sowie Ziffern zu einem „starken" Passwort zu verbinden – in den LinkedIn-Hashes konnte man mehrere zehntausend Passwörter in verschiedenen Formaten verschiedener Sprachräume finden. Auch hier bieten sich somit einem Angreifer gegenüber der Brute-Force-Suche deutliche „Abkürzungen".

Fazit

Insgesamt ließen sich wohl 90–95 % der verwendeten Passwörter innerhalb der Aufmerksamkeitsspanne der Community identifizieren – neben der üblichen Masse an Trivialpasswörtern und einer großen Menge von Ableitungen, die im Zusammenhang mit dem betroffenen Dienst standen (vgl. [3]), auch etliche in einer „Qualität" oder Länge, bei der man ein schnelles Knacken bei einem flüchtigen Blick auf den Klartext wohl für eher ausgeschlossen gehalten hätte.

Das alles zeigt uns (wieder einmal), dass es Menschen schwer fällt, wirklich gute Passwörter zu erzeugen: Egal wie verzwickt oder lang eine Abwandlung von etwas „Bekanntem" oder gut Merkbarem ausfällt: Solange sie auf Wörtern/Satzfragmenten und Regeln basiert, wird sie vermutlich mit überschaubarem Aufwand zu knacken sein. Dass die Variation gefundener Passwörter zu weiteren, komplizierteren Passwörtern führt, belegt letztlich, dass Menschen nun einmal ähnlich denken und vorgehen – und eben nicht zufällig. Und für diese Vorgehensweisen lassen sich Regeln ablei-

ten, die für eine deutliche Reduktion der Angriffsdauer sorgen.

Der einzige Schutz gegen solche Angriffe besteht in echter Zufälligkeit, einem großen Suchraum (Zeichenvorrat) und einer hinreichenden Länge des Passworts. Da sich eine derartige Kombination – zumal für jeden Dienst eine andere – jedoch kein Mensch merken kann, bleibt eigentlich nur der Griff zu einem Passwort-Management-System. Dabei ist dann darauf zu achten, dass dieses eine gute Zufälligkeit an den Tag legt und zudem die gespeicherte Passwortdatenbank ordentlich sichert – denn dieser „Single Point of Attack" sollte natürlich besonders gut geschützt sein (anders als im Fall der exemplarischen Untersuchung eines Passwort-Recovery-Anbieters im März 2012, der bei 17 Passwortmanagern unter iOS und BlackBerry keinem einen ausreichenden Schutz attestierte – 10–14 Zeichen lange Masterpasswörter konnten jeweils binnen eines Tages geknackt werden [4]).

Zudem gibt es auch auf der Seite der Service-Anbieter einiges zu tun: Das beginnt beim Salting und der Auswahl des richtigen (sprich: „knackunfreundlichen") Algorithmus' zum Speichern der Passworthashes (vgl. [5]). Und darüber hinaus belegen die erfolgreichen „Leaks" (LinkedIn war ja beileibe nicht der einzige Fall in den vergangenen Monaten), dass auch in Sachen allgemeiner Sicherheitsmechanismen vielerorts noch Nachholbedarf besteht. ■

Francois Pesce ist Principal Engineer bei Qualys Inc.

Literatur

[1] Francois Pesce, Discovered Patterns in Numeric Passwords Raise New Questions, Qualys Blog, https://community.qualys.com/blogs/securitylabs/2012/07/12/discovered-patterns-in-numeric-passwords-raise-new-questions

[2] Robert David Graham, LinkedIn vs. password cracking, Errata Security Blog, http://erratasec.blogspot.de/2012/06/linkedin-vs-password-cracking.html

[3] Jeremi Gosney, The Final Word on the LinkedIn Leak, Security Nirvana Blog, http://securitynirvana.blogspot.de/2012/06/final-word-on-linkedin-leak.html

[4] Andrey Belenko, Dmitry Sklyarov, „Secure Password Managers" and „Military-Grade Encryption" on Smartphones: Oh, Really?, Whitepaper, März 2012, www.elcomsoft.com/WP/BH-EU-2012-WP.pdf

[5] „Solar Designer", Password Security: Past, Present, Future, Präsentationsfolien zu einem Vortrag auf den PHDays 2012, www.openwall.com/presentations/PHDays2012-Password-Security/

[6] Dan Goodin, Why passwords have never been weaker—and crackers have never been stronger, ars technica, http://arstechnica.com/security/2012/08/passwords-under-assault/

[7] Constance Thanner, Markus Krumm, Passwort 2010, Sichere Gestaltung und Verwaltung von Passwörtern, <kes> 2009#2, S. 6, frei zugänglich über www.kes.info/archiv/online/passwort2010.html

[8] Jürgen Pabel, Schlüsselkasten, KeePass: Open-Source-Software zum Passwortmanagement, <kes> 2008#3, S. 66