Modern Network Security Practices: Using Rainbow Tables to Solve an Organizational Issue

Christopher McMahon and Xiaowen Zhang, Ph.D.

Department of Computer Science, College of Staten Island, City University of New York

The Abstract

The purpose of this case study analysis is to examine a non-traditional method of identifying weak passwords within a large hospital organization. The process of using rainbow tables to crack passwords/ensure password compliance is discussed and specific examples are provided within this paper. This process emphasizes the notion that network security-related problems tend to be organization-specific and require creative approaches. The goal is to establish a practical use for rainbow tables within an organization as a means of enhancing network security.

The Problem

A recent problem of a large North American hospital:

- Their network security team is unable to mandate regular password changes because of the large, diverse population of close to 12,000 users.
- Many users in patient care never directly log in to a computer, only logging in to their applications , rarely checking their e-mail accounts, and would not know if their password had expired..
- Password complexity was not a requirement added until 2010. This meant that passwords set before 2010 had no restrictions on length or character types.
- Since passwords are stored as hashes with a constant length in Active Directory, it is impossible to easily determine whether a password meets the current complexity requirements.

This resulted in an unknown amount, possibly thousands, of passwords that were not compliant to current complexity requirements. To address this issue, the idea was proposed to use Rainbow Tables to identify which passwords were noncompliant.

What are Rainbow Tables?

A rainbow table is a type of hash lookup table utilizing TMTO (Time-Memory Trade-Off attack) generated to reverse cryptographic hash functions as a means to crack password hashes. It is compromised of rainbow chains, which starts with a plaintext password and uses alternating hash and reduction functions. Everything is then thrown away except for the first input and the last hash.

Rainbow Chain Example

When cracking a password, these chains are then regenerated until the hash is found. This greatly improves storage efficiency over a regular password and hash table but requires more time to perform the hash lookup.

The Method

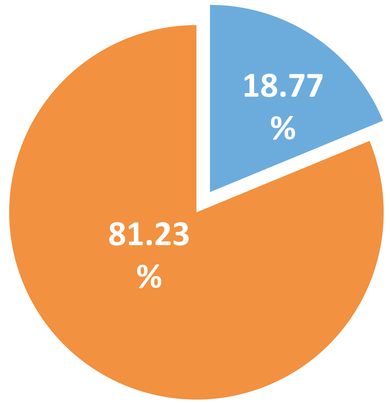
The product used is called RainbowCrack (http://project-rainbowcrack.com). It is a full suite of tools for generation, sorting, and merging of rainbow tables, as well as a lookup tool for passwords inside of the tables. The process for this project was as follows:

- Generating the tables with rtgen. For this project, we generated 50 tables of all passwords 7 characters or less.
- Sorting the tables using rtsort. This sorts each table by end point of each rainbow chain to make binary search possible.
- Extract all user accounts and password hashes from Active Directory. This was accomplished by using a PowerShell script utilizing DSInternals PowerShell module. (https://github.com/MichaelGrafnetter/DSInternals)
- Run a comparison of extracted password hashes against the generated rainbow tables using rcrack.

The Results

Total number of password hashes: 11980

- Number of passwords cracked: 2249
- Number of passwords uncracked: 9731



Password length of cracked hashes

Length	Count
7 characters	1961
6 characters	136
5 characters	131
4 characters	20
3 characters	1

The results of the project led to the following changes:

- The help desk was notified to reset non-compliant passwords and contact offending users.
- Kerberos became the required authentication protocol.
- LM (An outdated hashing function) password hashes are no longer stored in Active Directory.
- Fine-grain password policies were put in place to ensure stronger password complexity for Administrator accounts.