Lab 5 Report  
Passwords

# Task 1: Password Files

1. What error did you receive when you tried to view the shadow password file without privilege?

more: cannot open /etc/shadow: Permission denied

1. As recorded in item #1 above, you received an error when you tried to view the shadow password file. Why is this error a good thing?

This prevents random users without granted permissions to access your data.

1. What is the hash algorithm that was used to store your password information in the password file?

6: SHA-512

1. What salt value was used to generate your stored hash value?

7voVz3cj

1. When was your password chosen, as reported by the chage command?

July 7th, 2018

1. In item #5 above you recorded the date/time when your password was selected. Why does the system need to keep track of this information?

The system needs to track how old are passwords are to determine if they are at higher risk for being solved and compromise the account it is linked to.

# Task 2: Dictionary Attacks

1. By examining the passwords in htpasswd-sha1, which users have the same password?

Alice and Dave have the same password, linked by the hash value starting with A9Z8J..

1. List the usernames and passwords of any accounts in htpasswd-sha1 that were cracked when using tinylist.txt as the dictionary.

Alice and Dave’s password were cracked, they share the password ‘awesome’.

1. List the usernames and passwords of any accounts in htpasswd-sha1 that were cracked when using biglist.txt as the dictionary.

5 of the 7 user’s passwords were cracked:

- carol: ‘2cute4u’

- alice & dave: ‘awesome’

- bob: ‘password1’

- eve: ‘zaq12wsx’

1. Record the displayed statistics when you performed a dictionary attack using biglist.txt as the dictionary.
   1. Number of words tried: 2198690
   2. Number of passwords found: 5
   3. Number of seconds: 5.317 seconds
2. When a dictionary attack fails to crack one or more passwords (as was the case in the above cracking attempts), what can be said about those passwords?

The smaller dictionaries may be using less complex phrases and only common words rather than intricacies with numbers and symbols.

# Task 3: Considering Execution Times

1. Record the displayed statistics when you performed a dictionary attack using biglist.txt on htpasswd-md5.
   1. Number of words tried: 2198690
   2. Number of passwords found: 5
   3. Number of seconds: 5.367 seconds
2. Item #12 recorded the time it took to perform a dictionary attack on a file of MD5 digests. Assume there is a hash algorithm called APR1 that is simply 1000 iterations of MD5. If this password file indicated that APR1 had been used instead of MD5, approximately how many seconds would it have taken? Show your work.

5.367 seconds \* 1000 iterations = 5,367 seconds for APR1 hash file

1. Item #12 recorded the time it took to perform a dictionary attack on a file of MD5 digests. If this password file also contained salt values that were used in the creation of the MD5 digests (i.e., it contained the following: username, salt, digest), roughly how many seconds would it have taken? Why?

Closer to 6 seconds, or 1000 iterations would be 6000 seconds.

Salt values contain unique values, even if two users share the same password which makes cracking them in dictionary attacks harder to do.

1. Record the output data when using biglist.txt on htpasswd-sha512.
   1. Number of words tried: 2198690
   2. Number of passwords found: 5
   3. Number of seconds: 5.691 seconds
2. Referring to the times recorded in #12 and #15, if a system was using MD5 as the hash function for storing password information, but then switched to SHA512, by what percentage would it slow down a dictionary attack (or a brute force attack)? Show your work.

5.691s / 5.367s = 1.0603 = 6.03%

1. Review the results of the spreadsheet when 10,000,000,000 passwords/sec was entered. From the point of view of a computer security officer, what conclusions or observations can be made?

Regardless of how the passwords are composed, even at the level of lower-case passwords, it still takes 137,663,307 years to crack those passwords. With all characters involved, it would take 44,492,465,575,346,500 years to solve every password.

1. Record the output data when pre-hashed passwords are used to crack htpasswd-sha1.
   1. Number of words tried: 3693
   2. Number of passwords found: 5
   3. Number of seconds: 0.008 seconds
2. Explain why crackPre.py did not try all the words in the dictionary.

crackPre.py only looked into the a9 hash value, only containing thousands of passwords in the dictionary. The entire dictionary has millions of passwords to try, if we wanted to try all of them, we would have to include all hash values.

# Task 4: Personal Experimentation

1. Record your observations and conclusions from your personal experimentation.

Only one of the three passwords I created were cracked in this experiment. Adding symbols and numbers will make the password harder to crack.

1. What did you learn from this lab exercise?

Using simple phrases like password and similar words are some of the worst you can use for a password. Adding symbols and numbers, along with randomized characters will make for a much stronger password.

1. How could this lab exercise be improved?

N/A.