Linux Server Security Review Scenario
By Gerald Blackmon @ BlueJay Labs

The following command line was observed when analyzing the logs inside a Linux Server.

```
[root@prod01 ~]# cat /etc/shadow
root:$1$5f4dcc3b5aa765d61d8327deb882cf99:15651:0:99999:7:::
```

Part I - Initial assessments

The example listed above displays the contents of a shadow file stored within the Linux systems etc directory.

The Command above can lead us to a number of cascading conclusions which start with a “root” login. This level of login on a system is seen as an insecure method of operation. Also, the result of the cat /etc/shadow command shows only one result with its own set of problems as follows:

1) There is only 1 username available and it is “root”. This is an access control problem due to the concept of anonymity for anyone that logs into this machine. There will be no way to track the movement and commands of any individual person if all users use the one and only username.
2) Root username is provided with the system as a default and the associated password is likely a default password easily breached without any effort.
3) Root level access is without restrictions and leaves all of the system resources vulnerable to attack by anyone who logs in. This level of access can even lockout previously authorized access to the original account owner.

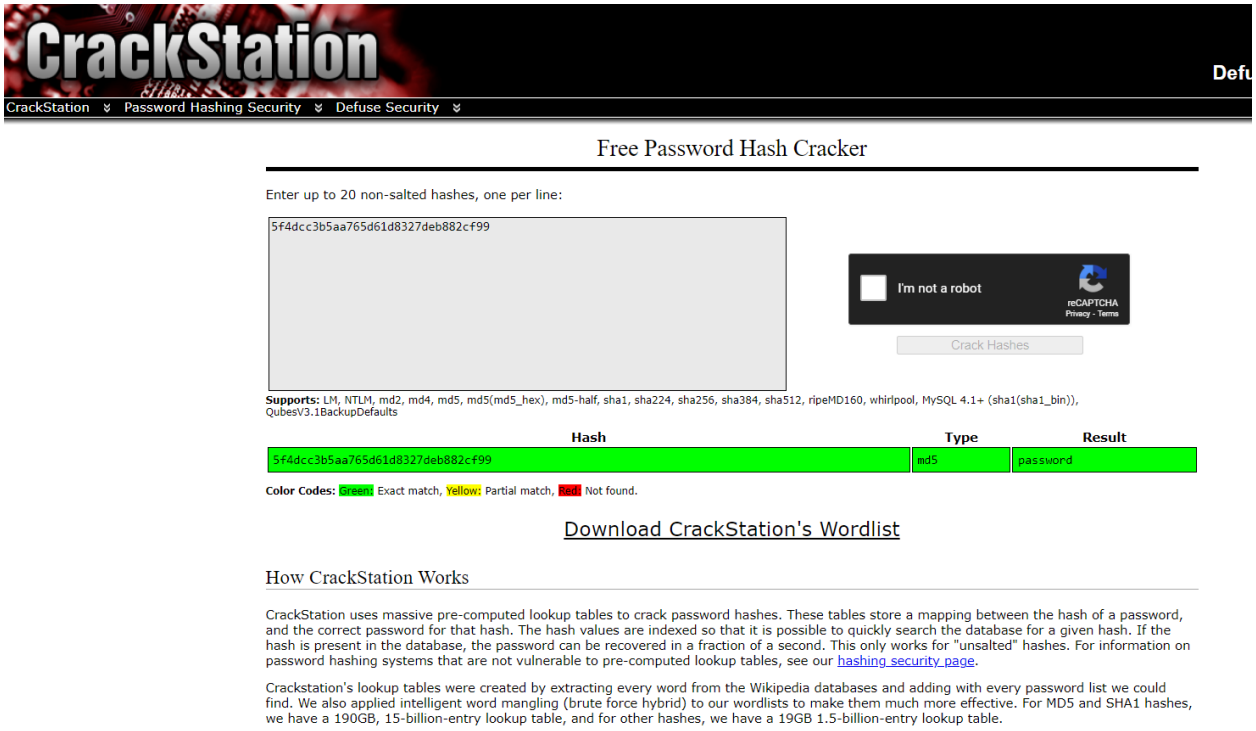
Additional information can be obtained from the above command line according to a breakdown sample case listed below. Basically, the /etc/shadow file stores secure user

account information. All fields are separated by a colon (`:`) symbol. It contains one entry per line for each user listed in /etc/passwd file. Generally, shadow file entry looks as follows:

1. **Username** : It is your login name.
2. **Password** : It is your encrypted password. The password should be minimum 8-12 characters long including special characters, digits, lower case alphabetic and more. Usually password format is set to `$id$salt$hashed`, The `$id` is the algorithm used On GNU/Linux as follows:
    1. **`$1$`** is MD5
    2. **`$2a$`** is Blowfish
    3. **`$2y$`** is Blowfish
    4. **`$5$`** is SHA-256
    5. **`$6$`** is SHA-512
3. **Last password change (lastchanged)** : Days since Jan 1, 1970 that password was last changed
4. **Minimum** : The minimum number of days required between password changes i.e. the number of days left before the user is allowed to change his/her password
5. **Maximum** : The maximum number of days the password is valid (after that user is forced to change his/her password)
6. **Warn** : The number of days before password is to expire that user is warned that his/her password must be changed
7. **Inactive** : The number of days after password expires that account is disabled
8. **Expire** : days since Jan 1, 1970 that account is disabled i.e. an absolute date specifying when the login may no longer be used.

Part II - Deeper analysis

By applying this information we can also see that the level of encryption used for the hash of the password is MD5. This level of encryption can easily be “cracked” with a number of easily available online tools. In this case we will use a site known as crackstation.net to acquire the actual password to the account.



From the example shown above we can see that the cracked hash is MD5 encryption and the result is “password”. This is a security concern which leads us to the conclusion that proper security has not ever been implemented on this machine. Username “root” with password set to “password” is typical of default out of the box settings on some linux operating systems. Default password hacking is nearly always a starting point for malicious actors when attempting to gain access to a system since it is well known.

We can also see that according to the hash this password is set to never require change as the date is set to 99999. This will inevitably lead to a compromise in security

given enough time. In addition, if the password to a User-level account has password time limits in place, a hacker who has gained access will then be locked out.

We can also see the “1” which indicates MD5 encryption type. MD5 hashes are **no longer considered cryptographically secure methods** and should not be used for cryptographic authentication, according to the Internet Engineering Task Force. The IETF publishes Requests for Comments (RFC’s) authored by network operators, engineers, and computer scientists to document methods, behaviors, research, or innovations applicable to the Internet.

The portion with number “15651” indicates this password has not been changed since Wednesday November 12th 2012. This allows anyone with a hacked password persistent access without making any changes to the system that might otherwise be detected.

Part III - Best Practices and Remediation

NIST 800-53 provides a list of controls that support the development of secure and resilient information systems. These controls are the operational, technical, and management standards and guidelines used by information systems to maintain confidentiality, integrity, and availability. NIST password policy best practices offer the following guidelines:

Password length: Minimum password length (for user-selected passwords) is 10 characters with up to 64 (or more) allowed. Reasons for this prevent Brute force software from making a match in time and and multiple attempts.

Password complexity (e.g. requiring at least one upper- and lowercase, numeric, and special character): NIST recommends password complexity not be imposed but encouraged.

Character sets: The recommendation is all printing ASCII and UNICODE characters be allowed.

Password “hints”/authentication questions (e.g. what was your first car?): Password hints/authentication questions shouldn’t be used.

Check for “known bad” passwords: New and changed passwords are to be checked against a list of common or previously compromised passwords from password dictionary files, previous breaches, keyboard patterns, and contextual words (such as users name) One such example of an online resource to check compromised passwords is haveibeenpwned.com

Throttling: Implement throttling to limit failed authentication attempts. This measure will prevent hacking attempts by locking out an account if failed attempts exceed a predefined amount.

Password expiration: Organizations should require users to change their password at defined intervals (e.g. 45, 60, or 90 days)

Stronger encryption: Alternatives to MD5 encryption can be SHA256 or SHA512 in order to strengthen security.

Proper access control: A set of user names and associated user access group levels is required to maintain best security practices. Often a set of privileges can be associated with each level of access. Additionally, individual user names will remove the anonymity from logged in users of all levels. One secure example of new access control groups would include:

1. Administration level access group - This group would maintain full access to make system changes but unlike root level access the user would not be anonymous and changes made by them would now be recorded and attributed to the proper person. This level would have “sudo” ability.
2. User level access group - This group would perform common read, write, and executable functions without access to protected system resources. One such example would be that directories such as the /etc would be inaccessible. Also, “sudo” functions disabled.
3. Client level access group - This group could be limited to read-only access in order to simply view file contents without access to make changes.

The measures listed within this report encompass a summary of industry-standard accepted best practices according to multiple security publications. However, the level

of security can impose a certain degree of inconvenience to Users. Administrators can implement additional stronger measures if the tradeoff is deemed acceptable and necessary.