HASHCAT

TIPOS DE HASHES

V. MEXICANH

NOVIEMBRE 2013

COMPILED BY MEXICANH TEAM

Dev/null/ antes que el deshonor

@MexicanH

AGRADECIMIENTOS

Principalmente queremos a agradecer al creador de la familia hashcat Atom.

Gracias a el equipo MexicanH que sin ellos nada de esto sería posible.

AVISO LEGAL

Este manual está hecho para fines educativos únicamente, los autores no se hacen responsables del mal uso que se le pueda llegar a dar.

ACERCA

Cuando estamos auditando contraseñas, a menudo nos encontramos con la necesidad de desencriptar contraseñas de hashes tales como MD5, SHA-1, DES (UNIX), etc. Pero como podemos identificar que tipo de algoritmo tenemos con solo mirar a simple vista? En este manual cubriremos los algoritmos más comunes que encontramos en el mundo de pentesting. Cubriremos los algoritmos soportados por diferentes herramientas tales como la familia hashcat y mysql collider.

La familia hashcat está en constante desarrollo así que en un futuro más algoritmos de los aquí presentados podrán ser añadidos. La herramienta mysql collider se encuentra en la versión beta 5.

Piénselo bien, que es un hash? “los hashes son contraseñas encriptadas”, cualquier lector podría no entenderlo del todo en la primera vez. La multitud de hashes es un mundo entero, un hash es primero que nada el guardián de un secreto. Este mundo misterioso está lleno de secretos, enigmas, manchas oscuras y oportunidades por descubrir.

Si, el caótico juego de caracteres es de hecho una contraseña. Pero una contraseña es la llave de una puerta a un secreto, mientras que ese secreto puede ser de cualquier tipo. Un hash puede esconder un aburrido mensaje en un foro. Puede ser la contraseña que te dará acceso a la base de datos con información privilegiada.

Suponemos que el lector ya tiene conocimientos de cómo usar hashcat, de otro modo le sugerimos buscar nuestro manual en la red.

Ahora después de tanta teoría comencemos.

TIPOS DE HASHES

Antes de comenzar cabe algo importante que destacar, los algoritmos de hashes son funciones no reversibles, eso quiere decir que "unhash" una contraseña no es posible y aplicar esa expresión es totalmente erróneo.

Algo que también sucede en el mundo de los hashes, son las colisiones, estas ocurren debido a que el rango de todos los posibles hashes es finito y es definido por el tamaño del hash(por ejemplo, para el algoritmo MD5 el numero de todos los probables hashes es 2^{128}), mientras que el numero de valores de entrada es infinito. Obviamente, hay cadenas de texto que tendrán hashes iguales. A estas fuentes se les llaman colisiones, que veremos con mysql collider.

Salt($salt) o sal en español son bits aleatorios que se añaden a nuestra contraseña para hacer aun mas difícil y lenta su recuperación, con esto se logra que las tablas de almacenamiento sean inútiles ya que para una misma palabra tal como "qwert123" el hash resultante será distinto.

Cabe destacar también que en el caso del algoritmo MD5, SHA-1, etc. aunque puede ser usado en todos los algoritmos, existen muchas variantes, tales como encriptar dos o tres veces una contraseña, hay hashes que lucen iguales tales como el MD5 Y el NTLM, el sha-1 y el mysql5. La única manera de saber que algoritmo tenemos es saber la fuente del hash y si es posible revisar el código con el que la contraseña fue encriptada. Ya que si tenemos un aparente hash MD5, en el que antes las contraseña fue encriptada 2 veces, MD5(M5($pass)), e intentamos recuperarla como un simple MD5, podremos intentar infinitamente y nunca podremos recuperarla, por eso es importante saber la fuente, revisar el código con que él fue encriptada, y si no se tiene la certeza entonces será necesario probar con diferentes algoritmos.

- MD5(Message-Digest algorithm)

Es el más comunmente usado

Ejemplo: d41d8cd98f00b204e9800998ecf8427e

Tienes 32 caracteres de largo

En hashcatcli usar el modo –m 0

En cudahascat y oclhashcat usar el modo –m 0

En hashcatlite usar el modo –m 0

- MD5($pass.$salt)

Ejemplo: d41d8cd98f00b204e9800998ecf8427e:12345

Utilizados en WB News.

Es usado también en joomla, pero estos son almacenados en de una manera particular que mostraremos mas adelante.

Tiene 32 caracteres de largo, la longitud del salt es variada.

En hashcatcli usar el modo –m 10

En cudahascat y oclhashcat usar el modo –m 10

En hashcatlite usar el modo –m 10

- JOOMLA(MD5($pass.$salt))

Ejemplo:
5f34c774ac406dfd1f56098eb20bfd62:ePTJxKIKY0oNS9XhCJhnfI3ZO1CCdQc6

Utilizados en las web joomla.

Puede usarse también el modo –m 10 pero preferentemente usar el modo –m 11

En hashcatcli usar el modo –m 11

En cudahascat y oclhashcat usar el modo –m 11

En hashcatlite usar el modo –m 11

- MD5($salt.$pass)

Utilizado en osCommerce, AEF, Galeria y otros CMS.

Ejemplo: d41d8cd98f00b204e9800998ecf8427e:12

En hashcatcli usar el modo –m 20

En cudahascat y oclhashcat usar el modo –m 20

En hashcatlite usar el modo –m 20

- osCommerce, xt:Commerce MD5($salt.$pass)

Ejemplo: d41d8cd98f00b204e9800998ecf8427e:12

El modo de encriptación es igual que el anterior, solo que en hashcat esta optimizado.

En hashcatcli usar el modo –m 21

En cudahascat y oclhashcat usar el modo –m 21

En hashcatlite usar el modo –m 21

- MD5(unicode($pass).$salt)

Con esto se cambiara la codificación de nuestra contraseña a Unicode antes de ser encriptada

Ejemplo: d41d8cd98f00b204e9800998ecf8427e:12345

En cudahascat y oclhashcat usar el modo –m 30

- MD5($salt.unicode($pass))

Con esto se cambiara la codificación de nuestra contraseña a Unicode

Ejemplo: d41d8cd98f00b204e9800998ecf8427e:12345

En cudahascat y oclhashcat usar el modo –m 40

- HMAC-MD5 (key = $pass) , HMAC-MD5 (key = $salt)

El objeto HMACMD5 es un tipo de algoritmo hash con clave que se crea desde la función hash MD5 y se utiliza como HMAC (Código de autenticación de mensajes basado en hash). El proceso HMAC combina una clave secreta con los datos de mensaje, aplica el algoritmo hash al resultado con la función hash, combina de nuevo ese valor hash con la clave secreta y, a continuación, aplica la función hash por segunda vez. El hash de salida tiene 128 bits de longitud.

Ejemplo: 80070713463e7749b90c2dc24911e275:key

En ocasiones el hash puede venir con un "0x" al inicio, hay que omitirlo.

En cudahascat y oclhashcat usar el modo –m 50 para HMAC-MD5 (key = $pass)

En cudahascat y oclhashcat usar el modo –m 60 para HMAC-MD5 (key = $salt)

- SHA-1

Ahora hay una tendencia en usarlo mucho más que el MD5 debido a la poca seguridad que el MD5 ofrece, el SHA-1 se usa en muchos foros, webs y CMS.

Usa una cadena de 40 caracteres

Ejemplo: da39a3ee5e6b4b0d3255bfef95601890afd80709

En hashcatcli usar el modo –m 100

En cudahascat y oclhashcat usar el modo –m 100

En hashcatlite usar el modo –m 100

- nsldap, SHA-1(Base64), Netscape LDAP SHA

Usado en OpenLDAP Software, puede ser usado como valor “userPassword” y/o valor “rootpw”.

Usa una cadena de 28 caracteres, para ser usado con hashcat-plus hay que añadir {SHA} al inicio del hash.

Ejemplo: {SHA}fDYHuOYbzxlE6ehQOmYPIfS28/E=

En cudahascat y oclhashcat usar el modo –m 101

En hashcatlite usar el modo –m 101

- SHA1($pass.$salt)

Al igual que los otros algoritmos como SHA1 y MD5 este es usado en muchas webs para almacenar las contraseñas de los usuarios, en foros y demás bases de datos. El salt es usado hacer aun más lenta y difícil su recuperación.la longitud del salt puede variar.

Ejemplo: da39a3ee5e6b4b0d3255bfef95601890afd80709:12345

En hashcatcli usar el modo –m 110

En cudahascat y oclhashcat usar el modo –m 110

En hashcatlite usar el modo –m 110

- nsldaps, SSHA-1(Base64), Netscape LDAP SSHA

Usado en OpenLDAP Software, puede ser usado como valor “userPassword” y/o valor “rootpw”. También es usado en el software Netscape LDAP para almacenar los valores de contraseñas. Los hashes tienen que tener el marcador {SSHA} al inicio.

Ejemplo Netscape LDAP SSHA:

{SSHA}WTT3B9Jjr8gOt0Q7WMs9/XvukyhTQj0Ns0jMKQ== (40 caracteres)

Ejemplo OpenLDAP SSHA:

{SSHA}hHSEPW3qeiOo5Pl2MpHQCXh0vgfyVR/X (32 caracteres)

En hashcatcli usar el modo –m 111

En cudahascat y oclhashcat usar el modo –m 111

En hashcatlite usar el modo –m 111

- Oracle 11g

Usado en las bases de datos Oracle 11g para almacenar los valores de contraseñas.

Ejemplo:

5FDAB69F543563582BA57894FE1C1361FB8ED57B903603F2C52ED1B4D642 (60 caracteres)

En cudahascat y oclhashcat usar el modo –m 112

En hashcatlite usar el modo –m 112

- SHA1($salt.$pass)

Al igual que los otros algoritmos como SHA1 y MD5 este es usado en muchas webs para almacenar las contraseñas de los usuarios, en foros y demás bases de datos. El salt es usado hacer aun más lenta y difícil su recuperación. La longitud del salt puede variar.

Ejemplo:

6728f4ab7fcb042d956b4bbb998aec9094054354:RAGABASH

En hashcatcli usar el modo –m 120

En cudahascat y oclhashcat usar el modo –m 120

- SMF > v1.1 (sha1(strtolower($username).$pass))o SHA1($salt.$pass)

Se utiliza en los foros Simple Machines Forum (SMF), el nombre de usuario es usado como salt, en dado caso no tener los nombres de usuario se puede cargar una lista usando el operador –e en hashcat, oclhashcat-plus no soporta esta función.

Ejemplo:

4f810efbff1ac5595939820450bf14e9e4d93daf:peacebreaker

En hashcatcli usar el modo –m 121

En cudahascat y oclhashcat usar el modo –m 121

- OSX v10.4, v10.5, v10.6

Usado en el sistema operativo MAC para almacenar los valores de contraseñas. Tienen una longitud de 48 caracteres.

Ejemplo:

2e8d6a1c229338bcf341804764faa02a53328cba7ee75466 (pass sheila1443q)

En hashcatcli usar el modo –m 122

En cudahascat y oclhashcat usar el modo –m 122

- SHA1($salt.unicode($pass))

Con esto se cambiara la codificación de nuestra contraseña a Unicode

Ejemplo: 6728f4ab7fcb042d956b4bbb998aec9094054354:RAGABASH

En cudahascat y oclhashcat usar el modo –m 130

- MSSQL(2000)

Utilizado en las bases de datos Microsoft 2000 para almacenar los valores de contraseñas. Tiene una longitud de 94 caracteres.

Ejemplo: 0x0100663134380000000000000000000000000000000000000000949a551630560094a998e377e260c9b33aaa8b04

En hashcatcli usar el modo –m 131

En cudahascat y oclhashcat usar el modo –m 131

En hashcatlite usar el modo –m 131

- MSSQL(2005)

Utilizado en las bases de datos Microsoft 2005 para almacenar los valores de contraseñas. Tiene una longitud de 54 caracteres.

Ejemplo: 0x01007770468559209df860bb3266603db96cdaf0c0703a80870d

En cudahascat y oclhashcat usar el modo –m 132

En hashcatlite usar el modo –m 132

- SHA1($salt.unicode($pass))

Con esto se cambiara la codificación de nuestra contraseña a Unicode

Ejemplo: 6728f4ab7fcb042d956b4bbb998aec9094054354:RAGABASH

En cudahascat y oclhashcat usar el modo –m 140

- EPiServer 6.x < v4

Utilizado en el software EPiServer 6.x

Ejemplo:
$episerver$*0*fGJ2wn/5WlzqQoDeCA2kXA==*zycIUapZz/v84FF93rAWDlCA3x8=

$episerver$ <- signature

* separador

0 <- versión (solo se permite el valor 0. 0 = sha1)

* separador

fGJ2wn/5WlzqQoDeCA2kXA== <- salt, base64 encoded

* separador

zycIUapZz/v84FF93rAWDlCA3x8= <- sha1 hash, base64 encoded

En cudahascat y oclhashcat usar el modo –m 141

En hashcatlite usar el modo –m 141

- HMAC-SHA1 (key = $pass) , HMAC-SHA1 (key = $salt)

HMACSHA1 es un tipo de algoritmo hash en clave que se crea desde la función hash SHA1 y se utiliza como HMAC o como código de autenticación de mensajes basado en hash. El proceso HMAC combina una clave secreta con los datos de mensaje, aplica el algoritmo hash a los resultados con la función hash, combina de nuevo ese valor hash con la clave secreta y, a continuación, aplica la función hash por segunda vez. El valor hash de salida tiene una longitud de 160 bits.

Ejemplo: de7c9b85b8b78aa6bc8a7a36f70a90701c9db4d9:key

En ocasiones el hash puede venir con un "0x" al inicio, hay que omitirlo.

En cudahascat y oclhashcat usar el modo –m 150 para HMAC-SHA1 (key = $pass)

En cudahascat y oclhashcat usar el modo –m 160 para HMAC-SHA1 (key = $salt)

- MySQL o MySQL < 4.1

Ya casi no son usados en las versiones menores 4.1 de MYSQL. Aunque ya casi son poco conocidos y muchos no saben que hacer con ellos tienen la gran desventaja que pueden ser colisionados.

Ejemplo: 606727496645bcba

En hashcatcli usar el modo –m 200

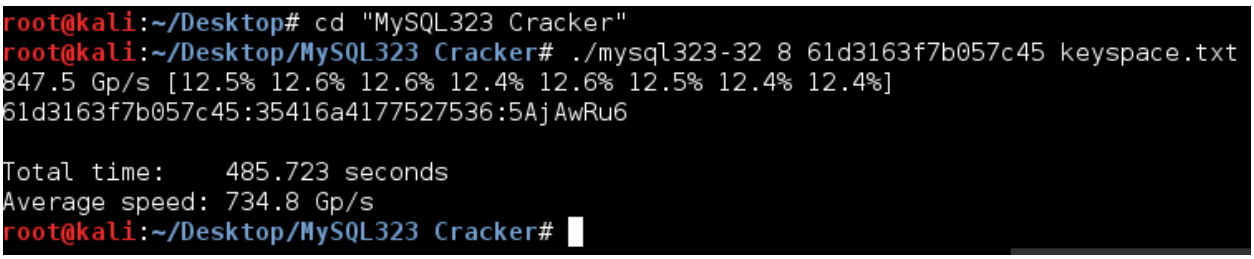
Hagamos un pequeño apartado sobre este hash.

Hashcatcli soporta este tipo de hash, pero debido a que es vulnerable, podemos usar las colisiones para romperlo, quizás nunca sepamos el valor original de la contraseña pero lo importante es saber que no la necesitaremos, ya que otra cadena de texto puede generar el mismo hash.

En el ejemplo de abajo usamos el script mysql collider, que descargamos de http://www.tobtu.com/mysql323.php

Como podemos observar rompimos el hash en tan solo un par de minutos, aprox. unos 10 min.

El comando fue ./mysql323-32 8 61d3163f7b057c45 keyspace.txt



Sin complicados métodos, diccionarios y reglas. Es por eso que este tipo de algoritmo no debe ser nunca más usado.

- MySQL o MySQL > 4.1 SHA-1(SHA-1($pass))

Usado en las nuevas versiones de Mysql para almacenar los valores de contraseñas de los usuarios.

Es fácil distinguirlo porque inicia siempre con un asterisco "*", todas las letras están en mayúsculas y tiene una longitud de 41 caracteres.

Ejemplo: *C8EB599B8E8EE7BE9F1A5691B7BC9ECCB8DE1C75

En hashcatcli usar el modo –m 300

En cudahascat y oclhashcat usar el modo –m 300

En hashcatlite usar el modo –m 300

- phpass, MD5(Wordpress), MD5(phpBB3)

Estos se dividen en dos tipos. El primero es el wordpress, usado en todos los sitios de wordpress y uno de los que mas confusiones genera a los principiantes, inicia con el marcador P seguido de el "costo" o el numero de iteraciones, normalmente es "9" o "B" y seguido el salt que es de 8 caracteres y el hash, tiene una longitud de 32 caracteres.

Ejemplo: P9QGUsR07ob2qNMbmSCRh3Moi6ehJZR1

P - Marcador

9 o B - Costo

QGUsR07o – Salt

b2qNMbmSCRh3Moi6ehJZR1 - Hash

El segundo es usado en los foros PHPBB también genera mucha confusión a los principiantes, inicia con el marcador H, seguido de el costo que normalmente es "9" seguido de 8 caracteres que es el salt y el hash. Tiene 34 caracteres de longitud.

Ejemplo: H9xAbu5SruQM5WvBldAnS46kQMEw2EQ0

En hashcatcli usar el modo –m 400

En cudahascat y oclhashcat usar el modo –m 400

- md5crypt, MD5(Unix), FreeBSD MD5, Cisco-IOS MD5

  Utilizados en Linux y otros similares OS.

  El hash comienza con el marcador 1 seguido por el salt de máximo 8 caracteres seguido por el hash.

  Ejemplo: $1$12345678$XM4P3PrKBgKNnTaqG9P0T/

1 -Marcador

12345678 - Salt

XM4P3PrKBgKNnTaqG9P0T/ -Hash

En hashcatcli usar el modo –m 500

En cudahascat y oclhashcat usar el modo –m 500

- SHA-1(Django) sha1($salt.$pass)

Utilizado en los entornos web django. Inicia con el marcador sha1, seguido por el salt que es de 8 caracteres y por último el hash.

Ejemplo: sha1$12345678$90fbbcf2b72b5973ae42cd3a19ab4ae8a1bd210b

En hashcatcli usar el modo –m 800

- MD4

Es uno de los algoritmos de primera generación, influencio diseños posteriores tales como el MD5, SHA etc. Totalmente obsoleto no debe usarse jamás. Tiene la misma longitud que el MD5 de 32 caracteres.

Ejemplo: ff577308c7b010415387be295e183e50

En hashcatcli usar el modo –m 900

En cudahascat y oclhashcat usar el modo –m 900

En hashcatlite usar el modo –m 900

- MD4 ($pass.$salt)

Es uno de los algoritmos de primera generación, influencio diseños posteriores tales como el MD5, SHA etc. Totalmente obsoleto no debe usarse jamás. Tiene la misma longitud que el MD5 de 32 caracteres. El salt es usado hacer aun más lenta y difícil su recuperación. La longitud del salt puede variar.

En hashcatlite usar el modo –m 910

- NTLM Y LM

Es el protocolo de autentificación realizado por Microsoft para Windows. Este se encuentra almacenado de la siguiente manera junto con el algoritmo LM que son el mismo valor de contraseña.

Ejemplo:
CAF976DC7580FA22AAD3B435B51404EE:B80D3EE5B0AAD7141C99ABF4193FF423:::

CAF976DC7580FA22AAD3B435B51404EE – LM hash

B80D3EE5B0AAD7141C99ABF4193FF423 – NTLM hash

En hashcatcli usar el modo –m 1000 para NTLM

En cudahascat y oclhashcat usar el modo –m 1000 para NTLM

En hashcatlite usar el modo –m 1000 para NTLM

En cudahascat y oclhashcat usar el modo –m 3000 para LM

En hashcatlite usar el modo –m 3000 para LM

- Domain Cached Credentials, mscash
  MD4(MD4(Unicode($pass)).Unicode(strtolower($username)))

Utilizado para el almacenamiento en cache de las contraseñas en dominios de Windows. Es también un algoritmo poco seguro, aun usando un salt. El salt es siempre es un nombre de usuario.

Ejemplo: Admin:b474d48cdfc4974d86ef4d24904cdd91

En hashcatcli usar el modo –m 1100

En cudahascat y oclhashcat usar el modo –m 1100

En hashcatlite usar el modo –m 1100

- SHA256

Es el sucesor del algoritmo SHA1 y es un de los algoritmos más fuertes que existe. Sirve para almacenar valores de contraseña en sitios web, bases de datos y foros. Tiene una longitud de 64 caracteres.

Ejemplo:
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855

En hashcatcli usar el modo –m 1400

En cudahascat y oclhashcat usar el modo –m 1400

En hashcatlite usar el modo –m 1400

- SHA256($pass.$salt)

Al igual que lo anterior descrito, al algoritmo se le agrega el salt para hacer aun más difícil su recuperación. La longitud del salt puede variar.

Ejemplo:
852C96A928396E63A0D16656FCC785451F02EDA42CF67270C74ACE87AE20DCF2:46728322

En hashcatcli usar el modo –m 1410

En cudahascat y oclhashcat usar el modo –m 1410

En hashcatlite usar el modo –m 1410

- SHA256($salt.$pass)

Al igual que lo anterior descrito, al algoritmo se le agrega el salt para hacer aun más difícil su recuperación. La longitud del salt puede variar.

Ejemplo:
852C96A928396E63A0D16656FCC785451F02EDA42CF67270C74ACE87AE20DCF2:46728322

En hashcatcli usar el modo –m 1420

En cudahascat y oclhashcat usar el modo –m 1420

En hashcatlite usar el modo –m 1420

- sha256(unicode($pass).$salt) o sha256($salt.unicode($pass))

Al algoritmo sha256 se le agrega el salt para hacer aun más difícil su recuperación. La longitud del salt puede variar. Además antes de ser convertido, a la contraseña se le cambia el tipo de codificación a Unicode.

Ejemplo:
852C96A928396E63A0D16656FCC785451F02EDA42CF67270C74ACE87AE20DCF2:46728322

En cudahascat y oclhashcat usar el modo –m 1430 para sha256(unicode($pass).$salt)

En cudahascat y oclhashcat usar el modo –m 1440 para sha256($salt.unicode($pass)

- EPiServer 6.x >= v4

Utilizado en el software EPiServer 6.x

Ejemplo:
$episerver$*1*MDEyMzQ1Njc4OWFiY2RlZg==*lRjiU46qHA7S6ZE7RfKUcYhB85ofArj1j7TrCtu3u6Y

$episerver$ <- signature

* separador

1 <- versión (solo se permite el valor 1. 1 = sha256)

* separador

MDEyMzQ1Njc4OWFiY2RlZg== <- salt, base64 encoded

* separador

lRjiU46qHA7S6ZE7RfKUcYhB85ofArj1j7TrCtu3u6Y <- sha256 hash, base64 encoded

En cudahascat y oclhashcat usar el modo –m 1441

En hashcatlite usar el modo –m 1441

- HMAC-SHA256 (key = $pass) , HMAC-SHA256 (key = $salt)

El objeto HMACMD5 es un tipo de algoritmo hash con clave que se crea desde la función hash SHA256 y se utiliza como HMAC (Código de autenticación de mensajes basado en hash). El proceso HMAC combina una clave secreta con los datos de mensaje, aplica el algoritmo hash al resultado con la función hash, combina de nuevo ese valor hash con la clave secreta y, a continuación, aplica la función hash por segunda vez. El hash de salida tiene 128 bits de longitud.

Ejemplo:
852C96A928396E63A0D16656FCC785451F02EDA42CF67270C74ACE87AE20DCF2
:key

En ocasiones el hash puede venir con un "0x" al inicio, hay que omitirlo.

En cudahascat y oclhashcat usar el modo –m 1450 para HMAC-SHA256 (key = $pass)

En cudahascat y oclhashcat usar el modo –m 1460 para HMAC-SHA256 (key = $salt)

- descrypt, DES(Unix), Traditional DES

Usado en Linux y otros SO similares. Tiene una longitud de 13 caracteres. Los dos primeros caracteres son el salt y los restatnes es el hash.

Ejemplo: IvS7aeT4NzQPM

Iv – salt

S7aeT4NzQPM – hash

En cudahascat y oclhashcat usar el modo –m 1500

En hashcatlite usar el modo –m 1500

- md5apr1, MD5(APR), Apache MD5

Usado en Linux y otros SO similares. Inicia con el marcador $apr1$ seguido de 8 caracteres que es el salt y lo restante

después del separador $ es el hash. Tiene una longitud de 37 caracteres.

Ejemplo: $apr1$12345678$auQSX8Mvzt.tdBi4y6Xgj.

En hashcatcli usar el modo –m 1600

En cudahascat y oclhashcat usar el modo –m 1600

- SHA512

Es el sucesor del algoritmo SHA1, al igual que el SHA256 fue desarrollado por la NSA o National Security Agency de Estados Unidos. Sirve para almacenar valores de contraseña en sitios web, bases de datos, foros y aplicaciones en la telefonía movil. Tiene una longitud de 128 caracteres.

Ejemplo:
cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce
47d0d13c5d85f2b0ff8318d2877eec2f63b931bd47417a81a538327af927da3e

En hashcatcli usar el modo –m 1700

En cudahascat y oclhashcat usar el modo –m 1700

En hashcatlite usar el modo –m 1700

- SHA512($pass.$salt)

Al igual que lo anterior descrito, al algoritmo se le agrega el salt para hacer aun más difícil su recuperación. La longitud del salt puede variar.

Ejemplo:
cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce
47d0d13c5d85f2b0ff8318d2877eec2f63b931bd47417a81a538327af927da3e
:12345678

En hashcatcli usar el modo –m 1710

En cudahascat y oclhashcat usar el modo –m 1710

En hashcatlite usar el modo –m 1710

- SSHA-512(Base64), LDAP {SSHA512}

Básicamente este formato es el mismo que el –m 1710 con la diferencia de que el formato de salida está en base64 y con un salt de 16 dígitos que se añade al hash antes de convertirlo a base64, aunque puede que el valor de este salt pueda variar. Se utiliza en los sistemas OpenLDAP.

Ejemplo:

{SSHA512}ALtwKGBdRgD+U0fPAy31C28RyKYx7+a8kmfksccsOeLknLHv2DBXYI7TDnTolQMBuPkWDISgZr2cHfnNPFjGZTEyNDU4OTkw

En cudahascat y oclhashcat usar el modo –m 1711

- SHA512($salt.$pass)

Al igual que lo anterior descrito, al algoritmo se le agrega el salt para hacer aun más difícil su recuperación. La longitud del salt puede variar.

Ejemplo:
cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce47d0d13c5d85f2b0ff8318d2877eec2f63b931bd47417a81a538327af927da3e:12345678

En hashcatcli usar el modo –m 1720

En cudahascat y oclhashcat usar el modo –m 1720

En hashcatlite usar el modo –m 1720

- OS X v10.7

Utilizado en el Mac OS X Lion. Se almacena como un SHA512 con un salt de 7 caracteres.

Ejemplo:
cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce47d0d13c5d85f2b0ff8318d2877eec2f63b931bd47417a81a538327af927da3e:1234567

En hashcatcli usar el modo –m 1722

En cudahascat y oclhashcat usar el modo –m 1722

- sha512(unicode($pass).$salt)

Ejemplo:

13070359002b6fbb3d28e50fba55efcf3d7cc115fe6e3f6c98bf0e3210f1c692
3427a1e1a3b214c1de92c467683f6466727ba3a51684022be5cc2ffcb78457d2
:341351589

Al algoritmo se le agrega el salt para hacer aun más difícil su recuperación, además antes de ser convertido a la contraseña se le cambia el tipo de codificación a Unicode. La longitud del salt puede variar.

En cudahascat y oclhashcat usar el modo –m 1730

- MSSQL 2012

Usado en Microsoft SQL server 2012. Tiene una longitud de 142 caracteres todas en mayúsculas, inicia con "0x", no debe ser omitido.

Ejemplo:
0x02006BF4AB05873FF0C8A4AFD1DC5912CBFDEF62E0520A3353B04E1184F05C
873C9C76BBADDEAAC1E9948C7B6ABFFD62BFEFD7139F17F6AFE10BE0FEE7A178
644623067C2423

En hashcatcli usar el modo –m 1731

- sha512($salt.unicode($pass))

Ejemplo:

13070359002b6fbb3d28e50fba55efcf3d7cc115fe6e3f6c98bf0e3210f1c692
3427a1e1a3b214c1de92c467683f6466727ba3a51684022be5cc2ffcb78457d2
:341351589

Al algoritmo se le agrega el salt para hacer aun más difícil su recuperación, además antes de ser convertido a la contraseña se le cambia el tipo de codificación a Unicode. La longitud del salt puede variar.

En cudahascat y oclhashcat usar el modo –m 1740

- HMAC-SHA512 (key = $pass)

El objeto HMAC-SHA512 es un tipo de algoritmo hash con clave que se crea desde la función hash SHA512 y se utiliza como HMAC (Código de autenticación de mensajes basado en hash). El proceso HMAC combina una clave secreta con los datos de mensaje, aplica el algoritmo hash al resultado con la función hash, combina de nuevo ese valor hash con la clave secreta y, a continuación, aplica la función hash por segunda vez. El hash de salida tiene 512 bits de longitud.

Ejemplo:
94cb9e31137913665dbea7b058e10be5f050cc356062a2c9679ed0ad6119648e7be620e9d4e1199220cd02b9efb2b1c78234fa1000c728f82bf9f14ed82c1976:1234

En ocasiones el hash puede venir con un "0x" al inicio, hay que omitirlo.

En cudahascat y oclhashcat usar el modo –m 1750 para HMAC-SHA512 (key = $pass)

En cudahascat y oclhashcat usar el modo –m 1760 para HMAC-SHA512 (key = $salt)

- sha512crypt, SHA512(Unix)

Este tipo de algoritmo es usado en los sistemas basados en Linux, normalmente almacenan las contraseñas de los usuarios. El hash conmienza con la firma 6 de ahí sigue un salt de 8 caracteres aleatorios, después el símbolo $ seguido por el hash. Es un algoritmo altamente iterado lo que lo hace muy difícil de recuperar, la cantidad de iteraciones son 5000.

Ejemplo:

$6$12345678$U6Yv5E1lWn6mEESzKen42o6rbEmFNLlq6Ik9X3reMXY3doKEuxrcDohKUx0Oxf44aeTIxGEjssvtT1aKyZHjs

En cudahascat y oclhashcat usar el modo –m 1800

En hashcatcli usar el modo –m 1800

- Domain Cached Credentials2, mscash2

Este tipo de hashes se almacenan en las computadoras con Windows en el registro. Se hace por si en algún momento esta máquina tiene una cuenta en un dominio y no se puede acceder al debido a una caída del servidor. Por defecto se almacenan 10 hashes.

Ejemplo:

fe778f07253e423f2090105b08a6c69b:61010842200

En cudahascat y oclhashcat usar el modo –m 2100

- Cisco-PIX MD5

Este algoritmo es comúnmente encontrado en los firewall Cisco PIX, para mas detalles visitar http://pythonhosted.org/passlib/lib/passlib.hash.cisco_pix.html

Ejemplo:

A5XOy94YKDPXCo7U

En cudahascat y oclhashcat usar el modo –m 2400

En hashcatcli usar el modo –m 2400

En hashcatlite usar el modo –m 2400

- WPA/WPA2

Este tipo de encriptación se utiliza en las redes wifi, antes se usaba WEP pero este tipo es fácilmente roto sin usar ataques de diccionario. A pesar de que WPA/WPA2 sustituyo a la encriptación WEP este es vulnerable a los ataque por fuerza bruta. Para auditar este algoritmo es necesario contar con el archivo de extensión .hccap que proviene de el archivo de captura .cap. para convertirlo se puede usar aircrack-ng usando el operador –J.

Ejemplo:

Red.hccap

En cudahascat y oclhashcat usar el modo –m 2500

En hashcatcli usar el modo –m 2500

- md5(md5($pass))

Este tipo de algoritmo no tiene mucho especial es solo que la contraseña es encriptada dos veces, para explicarlo más claro, después de haber obtenido el hash de la contraseña ese mismo hash es de nuevo encriptado usando MD5, por eso se le llama doble MD5. No hay una regla especifica donde pueda ser usado y tampoco es posible saberlo a simple vista.

Ejemplo:

a936af92b0ae20b1ff6c3347a72e5fbe

En cudahascat y oclhashcat usar el modo –m 2600

En hashcatcli usar el modo –m 2600

En hashcatlite usar el modo –m 2600

- vBulletin < v3.8.5

Este es usado en los foros Vbulletin pero este no es más que una función MD5 del tipo Md5(Md5($Pass).$Salt).

Ejemplo:

16780ba78d2d5f02f3202901c1b6d975:568

En cudahascat y oclhashcat usar el modo –m 2611

En hashcatcli usar el modo –m 2611

En hashcatlite usar el modo –m 2611

- vBulletin > v3.8.5

Este es usado en los foros Vbulletin pero este no es más que una función MD5 del tipo Md5(Md5($Pass).$Salt). La diferencia con la versión anterior es el largo del salt.

Ejemplo:

4cab87462a85a956796335e8245d9af4:0#wnP?aV|?Vb}&L~W@jFX?of9T>2p+

En cudahascat y oclhashcat usar el modo –m 2711

En hashcatcli usar el modo –m 2711

En hashcatlite usar el modo –m 2711

- IPB2+, MyBB1.2+

Usado en los programas y foros MyBB, es una variante de la función MD5 del tipo md5(md5($salt).md5($pass)), lo que dificulta su recuperación.

Ejemplo:

464e875d047d5b43df8ff0a67a9911b5:1Sns8XQe

En cudahascat y oclhashcat usar el modo –m 2811

En hashcatcli usar el modo –m 2811

En hashcatlite usar el modo –m 2811

- Oracle 7-10g, DES(Oracle)

Usado en las bases de datos Oracle versiones desde la 7 a la 10g.

Ejemplo:

7A963A529D2E3229:3682427524

En cudahascat y oclhashcat usar el modo –m 3100

En hashcatlite usar el modo –m 3100

- bcrypt, Blowfish(OpenBSD)

En php la función crypt tiene el algoritmo MD5 por defecto, en versiones antiguas era DES, ahora en PHP 5.3.7 viene ya implementado, pero porque usar blowfish? Esto es sencillo, ya que con el tiempo se construyen computadoras más modernas y con GPU (tarjetas graficas) más poderosos con lo que la función MD5 resulta ser insuficiente. Este algoritmo tiene la particularidad como otros de usar el parámetro “costo” que es usado para aumentar el poder de cómputo necesario para romper la contraseña. El costo determina cuantas iteración se harán antes de producir el resultado, este usa un logaritmo base dos, es decir 2^4 a 2^{31} (16 a 2147483648) iteraciones. Hay que tener en cuenta lo siguiente, si usamos el costo 10 y todo el proceso requiere 0.1 y tenemos 100 usuarios corriendo el mismo proceso esto tomara 10 segundos, lo que es inaceptable.

Ejemplo:

$2a$05$LhayLxezLhK1LhWvKxCyLOj0j1u.Kj0jZ0pEmm134uzrQlFvQJLF6

$2a$ - valor que índice que el algoritmo es BLOWFISH

05$ - costo

LhayLxezLhK1LhWvKxCyLOj0j1u.Kj0jZ0pEmm134uzrQlFvQJLF6 – salt

En cudahascat y oclhashcat usar el modo –m 3200

En hashcatcli usar el modo –m 3200

- MD5(Sun)

Este algoritmo fue desarrollado por Alec Muffett para Solaris, como un remplazo para la antigua función des_crypt. Fue introducido en Solaris 9u2. Mientras que está basado en la función MD5, tiene un poco en común con la función md5_crypt.

Ejemplo:

$md5$rounds904iPPKEBnEkp3JV8uX$0L6m7rOFTVFn.SGqo2M9W1

$md5 – es el prefijo usado para identificar el hash

$rounds – es el numero decimal de iteraciones a usar 5000 por ejemplo.

904$ - es el salt de 0-8 caracteres alfanuméricos, por defecto es 8.

En hashcatcli usar el modo –m 3300

- Variantes

Las siguientes variantes de encriptación son variantes de algoritmos antes mencionados, un existe una regla que diga donde usarlos y cuando, nosotros podemos encriptar una contraseña 4 veces con la función MD5, o SHA1, incluso primero pasarla por SHA1 y luego MD5 etc. Los siguientes algoritmos mencionados abajo son exclusivos de hashtacli.

3500 = md5(md5(md5($pass)))

3610 = md5(md5($salt).$pass)

3710 = md5($salt.md5($pass))

3720 = md5($pass.md5($salt))

3810 = md5($salt.$pass.$salt)

3910 = md5(md5($pass).md5($salt))

4010 = md5($salt.md5($salt.$pass))

4110 = md5($salt.md5($pass.$salt))

4210 = md5($username.0.$pass)

4300 = md5(strtoupper(md5($pass)))

4400 = md5(sha1($pass))

4500 = sha1(sha1($pass))

4600 = sha1(sha1(sha1($pass)))

4700 = sha1(md5($pass))

- WebEdition CMS

Usado en el software WebEdition CMS es básicamente de la forma md5($pass.md5($salt)).

Ejemplo :

fa01af9f0de5f377ae8befb03865178e:5678

En hashcatcli usar el modo –m 3721

- MD5(Chap)

Es usado en los protocolos de autentificación en los modernos sistemas de Windows, por ejemplo en las VPN, en donde la contrasenha es almacenada por el servidor. El hash de las contrasenha se computa de la siguiente manera, md5(id + password + challenge) donde el id es aumentado después de cada intento de autentificación. El "challenge" es generador por el servidor y enviado al cliente.

Ejemplo:

afd09efdd6f8ca9f18ec77c5869788c3:01020304050607080910111213141516:01

En hashcatcli usar el modo –m 4800

- SHA-3(Keccak)

Este tipo de función es relativamente nueva, su creador es Guido Bertoni, quien gano un concurso el 2 de Octubre del 2012. Este tipo de función no pretende sustituir a la función SHA-2. Tiene un largo de 64 caracteres.

Ejemplo:

203f88777f18bb4ee1226627b547808f38d90d3e106262b5de9ca943b57137b6

En cudahascat y oclhashcat usar el modo –m 5000

En hashcatcli usar el modo –m 5000

En hashcatlite usar el modo –m 5000

- Half MD5

Aunque su uso no está muy claro, este tipo de algoritmo se encuentra muy rara vez, ya que es vulnerable a las colisiones.

Ejemplo:

8743b52063cd8409

En cudahascat y oclhashcat usar el modo –m 5100

En hashcatcli usar el modo –m 5100

En hashcatlite usar el modo –m 5100

- Password Safe SHA-256

Este formato que no es precisamente un hash, sino un archivo con extensión .psafe3 el cual permite a los usuarios almacenar todas sus contrasenhas en un solo archivo. Este usado por ejemplo en Dropbox.

Ejemplo:

http://hashcat.net/misc/example_hashes/hashcat.psafe3

En cudahascat y oclhashcat usar el modo –m 5200

En hashcatcli usar el modo –m 5200

- IKE-PSK MD5

Usado en las VPN, para más información dirigirse a los links siguientes:

https://www.ernw.de/download/pskattack.pdf

https://hashcat.net/trac/ticket/5#no1

http://ikecrack.sourceforge.net/

Ejemplo:

http://hashcat.net/misc/example_hashes/hashcat.ikemd5

En cudahascat y oclhashcat usar el modo –m 5300

En hashcatcli usar el modo –m 5300

- IKE-PSK SHA1

Usado en las VPN, para más información dirigirse a los links siguientes:

https://www.ernw.de/download/pskattack.pdf

https://hashcat.net/trac/ticket/5#no1

http://ikecrack.sourceforge.net/

Ejemplo:

http://hashcat.net/misc/example_hashes/hashcat.ikesha1

En cudahascat y oclhashcat usar el modo –m 5400

En hashcatcli usar el modo –m 5400

- NetNTLMv1-VANILLA / NetNTLMv1+ESS

Este algoritmo en ocasiones es confundido con "NTLM2 SESSION". Difiere un poco del algoritmo NETNTLMV1 con la adición de un "challenge" de 8 bits. Se cree que ofrece una resistencia similar a las tablas arcoíris.

Ejemplo:

u4-
netntlm::kNS:338d08f8e26de93300000000000000000000000000000000:95
26fb8c23a90751cdd619b6cea564742e1e4bf33006ba41:cb8086049ec4736c

En cudahascat y oclhashcat usar el modo –m 5500

En hashcatcli usar el modo –m 5500

En hashcatlite usar el modo –m 5500

- NetNTLMv2

Es también usado en los sistemas basado en Windows. Este requiere un nombre de usuario, un objetivo de autentificación,el cual es sensible a mayúsculas o minisculas.

Ejemplo:

admin::N46iSNekpT:08ca45b7d7ea58ee:88dcbe4446168966a153a0064958d
ac6:5c7830315c7830310000000000000b45c67103d07d7b95acd12ffa11230e
0000000052920b85f78d013c31cdb3b92f5d765c783030

En cudahascat y oclhashcat usar el modo –m 5600

En hashcatcli usar el modo –m 5600

En hashcatlite usar el modo –m 5600

- Cisco-IOS SHA256

Usado en los sisemas operativos Cisco IOS, esta basado en la función SHA256.

Ejemplo:

2btjjy78REtmYkkW0csHUbJZOstRXoWdX1mGrmmfeHI

En cudahascat y oclhashcat usar el modo –m 5700

En hashcatcli usar el modo –m 5700

En hashcatlite usar el modo –m 5700

- Samsung Android Password/PIN

Se obtiene de los teléfonos Samsung con OS andriod. Para mas detalles:

http://hashcat.net/forum/thread-2202.html

Ejemplo:

0223b799d526b596fe4ba5628b9e65068227e68e:f6d45822728ddb2c

En cudahascat y oclhashcat usar el modo –m 5800

En hashcatcli usar el modo –m 5800

- RipeMD160

Es una versión mejorada de RIPEMD, que estaba basado sobre los principios del diseño del algoritmo MD4, y es similar en seguridad y funcionamiento al más popular SHA-1. Por otra parte, RIPEMD-160 es un diseño menos popular y correspondientemente está peor estudiado.

Ejemplo:

012cb9b334ec1aeb71a9c8ce85586082467f7eb6

En cudahascat y oclhashcat usar el modo –m 6000

- Whirlpool

Es una función de hash diseñada por Vincent Rijmen y Paulo S. L. M. Barreto. El hash ha sido recomendado por el proyecto NESSIE y ha sido adoptado por la Organización Internacional de Estandarización (ISO) y la Comisión Electrotécnica Internacional (IEC) como parte del estándar internacional ISO/IEC 10118-3.

Ejemplo:

7ca8eaaaa15eaa4c038b4c47b9313e92da827c06940e69947f85bc0fbef3eb8f
d254da220ad9e208b6b28f6bb9be31dd760f1fdb26112d83f87d96b416a4d258

En cudahascat y oclhashcat usar el modo –m 6100

- Truecrypt

621Y = TrueCrypt 5.0+ PBKDF2-HMAC-RipeMD160

622Y = TrueCrypt 5.0+ PBKDF2-HMAC-SHA512

623Y = TrueCrypt 5.0+ PBKDF2-HMAC-Whirlpool

624Y = TrueCrypt 5.0+ PBKDF2-HMAC-RipeMD160 boot-mode

62XY = TrueCrypt 5.0+

X = 1 = PBKDF2-HMAC-RipeMD160

X = 2 = PBKDF2-HMAC-SHA512

X = 3 = PBKDF2-HMAC-Whirlpool

X = 4 = PBKDF2-HMAC-RipeMD160 boot-mode

Y = 1 = XTS AES

Usados en el software de encriptación Truecrypt 5. No es un hashes sino un archivo con extensión .tc

http://hashcat.net/misc/example_hashes/hashcat_ripemd160.tc

http://hashcat.net/misc/example_hashes/hashcat_sha512.tc

http://hashcat.net/misc/example_hashes/hashcat_whirlpool.tc

http://hashcat.net/misc/example_hashes/hashcat_ripemd160_boot.tc

- AIX

6300 = AIX {smd5}

6400 = AIX {ssha256}

6500 = AIX {ssha512}

6700 = AIX {ssha1}

Estos son usados en los sistemas operativos IBM AIX basado en Unix.

Ejemplos:

{smd5}a5/yTL/u$VfvgyHx1xUlXZYBocQpQY0

{ssha256}06$aJckFGJAB30LTe10$ohUsB7LBPlgclE3hJg9x042DLJvQyxVCX.nZZLEz.g2

{ssha512}06$bJbkFGJAB30L2e23$bXiXjyH5YGIyoWWmEVwq67nCU5t7GLy9HkCzrodRCQCx3r9VvG98o7O3V0r9cVrX3LPPGuHqT5LLn0oGCuI1..

{ssha1}06$bJbkFGJAB30L2e23$dCESGOsP7jaIIAJ1QAcmaGeG.kr

- 1Password

Muy parecido a .psafe3, es un archivo con extensión .1password usado en la línea Ipod y Iphone.

Ejemplo:

http://hashcat.net/misc/example_hashes/hashcat.1password

En cudahascat y oclhashcat usar el modo –m 6600

- Lastpass

Usado para romper el software lastpass, programa para almacenar contraseñas. Mas detalles en el siguiente link:

http://hashcat.net/forum/thread-2701.html

Ejemplo:

265f92cc8dac35bf09694ff921228ef8:500:lp@trash-mail.com

En cudahascat y oclhashcat usar el modo –m 6800

- GOST R 34.11-94

Es una función criptográfica de 256 bits. Usada inicialmente por el estándar nacional ruso.

Ejemplo:

df226c2c6dcb1d995c0299a33a084b201544293c31fc3d279530121d36bbcea9

En cudahascat y oclhashcat usar el modo –m 6900

- Fortigate (FortiOS)

Usado en el sistema operativo Fortigate.

Ejemplo:

AK1AAECAwQFBgcICRARNGqgeC3is8gv2xWWRony9NJnDgEA

En hashcatcli usar el modo –m 7000

- OSX v10.8

Usado en el sistema operativo Mountain lion

Ejemplo:

ml35460$93a94bd24b5de64d79a5e49fa372827e739f4d7b6975c752c9a0ff1e5cf72e05$752351df64dd2ce9dc9c64a72ad91de6581a15c19176266b44d98919dfa81f0f96cbcb20a1ffb400718c20382030f637892f776627d34e021bad4f81b7de8222

En cudahascat y oclhashcat usar el modo –m 7100

En hashcatcli usar el modo –m 7100

- GRUB 2

GRUB2 es un gestor de arraque, Es el responsable de cargar y transferir el control al kernel de Linux, que, a su vez, inicializa el resto del sistema operativo. Este puede ser protegido por contraseña.

Ejemplo:

grub.pbkdf2.sha512.10000.7d391ef48645f626b427b1fae06a7219b5b54f4f02b2621f86b5e36e83ae492bd1db60871e45bc07925cecb46ff8ba3db31c723c0c6acbd4f06f60c5b246ecbf.26d59c52b50df90d043f070bd9cbcd92a74424

da42b3666fdeb08f1a54b8f1d2f4f56cf436f9382419c26798dc2c209a860039
82b1e5a9fcef905f4dfaa4c524

En cudahascat y oclhashcat usar el modo –m 7200

En hashcatcli usar el modo –m 7200

- IPMI2 RAKP HMAC-SHA1

No existe clara información acerca de este algoritmo.

Ejemplo:

b7c2d6f13a43dce2e44ad120a9cd8a13d0ca23f0414275c0bbe1070d2d1299b1
c04da0f1a0f1e4e2537300263a220000000000000000000014076861736863 61
74

En hashcatcli usar el modo –m 7300

- sha256crypt, SHA256(Unix)

Se utiliza en Linux y sistemas operativos parecidos.

Ejemplo:

5rounds=5000$GX7BopJZJxPc/KEK$le16UF8I2Anb.rOrn22AUPWvzUETDGef
UmAV8AZkGcD

El numero de iteraciones $rounds=5000 es opcional, seguido viene el salt.

En cudahascat y oclhashcat usar el modo –m 7400

-texto plano

Esta función sirve para poder limpiar tus diccionarios, y encontrar si una palabra se encuentra en tu archivo mas de 2 veces o saber si cierta palabra se encuentra en ese archivo sin tener que abrirlo o usar cat en Linux.

En hashcatcli usar el modo –m 9999

REFERENCIAS

http://hashcat.net/forum/

http://www.openldap.org/doc/admin24/security.html

http://www.openldap.org/faq/data/cache/347.html

http://www.question-defense.com/2012/03/11/crack-linux-openldap-sha-passwords-using-oclhashcat-plus

http://pentestmonkey.net/cheat-sheet/john-the-ripper-hash-formats

http://www.defenceindepth.net/2009/12/cracking-os-x-passwords.html

http://www.databasesecurity.com/sqlserver/cracking-sql-passwords.pdf

http://www.phillips321.co.uk/2012/04/11/oclhashcat-examples-of-lots-of-different-hash-types/

https://hashcat.net/forum

http://www.insidepro.com/doc/005e.shtml

http://forum.insidepro.com/viewtopic.php?t=8225

http://en.wikipedia.org/wiki/Hash-based_message_authentication_code

http://msdn.microsoft.com/es-es/library/system.security.cryptography.hmacmd5(v=vs.85).aspx

http://www.101hacker.com/2010/12/hashes-and-seeds-know-basics.html

http://www.tobtu.com/mysql323.php

http://pythonhosted.org/passlib/lib/passlib.hash.django_std.html

en.wikipedia.org/wiki/MD4

http://en.wikipedia.org/wiki/NTLM

http://openwall.info/wiki/john/MSCash

http://www.backtrack-linux.org/forums/showthread.php?t=37216

http://www.securitygeneration.com/security/extracting-and-cracking-mac-os-x-lion-password-hashes/

http://blog.lostpassword.com/2012/07/cracking-mac-os-x-lion-accounts-passwords/

http://www2.htw-dresden.de/~s64599/6.%20Semester/Informationssicherheit/Prakt/Prakt01/john-1.7.9-jumbo-7/src/episerver_fmt.c

https://hashcat.net/forum/thread-2506.html

https://hashcat.net/trac/ticket/188

http://passwordsafe.sourceforge.net/quickstart.shtml

http://app77.com/pwSafe/faq.html#qstart

https://en.wikipedia.org/wiki/GOST_(hash_function)

https://help.ubuntu.com/community/Grub2/Passwords