

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

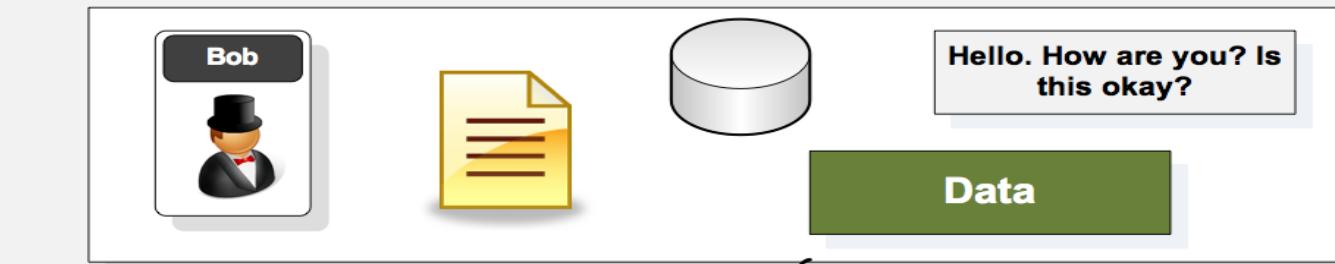
Secret Shares.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>

How do we get a finger-print for data?



With a
fingerprint we
can hopefully tell
if Eve has
modified any of
the data

**Solved by Prof Ron Rivest
with the MD5 hash
signature.**



Author: Prof Bill Buchanan



**Hashing
Algorithm (MD5)
- 128 bit signature**



hello

XUFAKrxLKna5cz2REBffkg

Hello

ixqZU8RhEpaoJ6v4xHgE1w

Hello. How are you?

CysDE5j+zoubCYztTdsFiw

Napier

j4NXH5Mkrk4j13N1MFxHtg

Base-64

hello

5D41402ABC4B2A76B9719D911017C592

Hello

8B1A9953C4611296A827ABF8C47804D7

Hello. How are you?

CC708153987BF9AD833BEBF90239BF0F

Napier

8F83571F9324AE4E23D773753055C7B6

Hex



**Hashing
Algorithm (SHA-1)**
- 160 bit signature



hello

qvTGHdzF6KLavt4P00gs2a6pQ00=

Hello

9/+ei3uy4Jtwk1pdeF4MxdnQq/A=

Hello. How are you?

Puh2Am76bhjqE51btTwtsqbdFC8=

Napier

v4GxNavod2b09GR2Tqw4yopOuro=

Base-64

hello

AAF4C61DDCC5E8A2DABEDE0F3B482CD9AEA9434D

Hello

F7FF9E8B7BB2E09B70935A5D785E0CC5D9D0ABF0

Hello. How are you?

3EE876026EFA6E18EA13995B4D6B70B2A6DD142F

Napier

BF81B135A5687766F4F464764EAC38CA8A4EBABA

Hex



**Hashing
Algorithm (MD5)
- 128 bit signature**



Security and mobility are two of the most important issues on the Internet, as they will allow users to secure their data transmissions, and also break their link with physical connections.

F94FBED3DAE05D223E6B963B9076C4EC

+U++09rgXSI+a5Y7kHbE7A==

Base-64

Security and mobility are two of the most important issues on the Internet, as they will allow users to secure their data transmissions, and also break their link with physical connections.

8A8BDC3FF80A01917D0432800201CFBF

iovcP/gKAZF9BDKAAgHPVW==

Hex

OpenSSL

```
root@kali:~# echo -n "hello" | openssl md5  
(stdin)= 5d41402abc4b2a76b9719d911017c592
```

```
root@kali:~# echo -n "hello" | md5sum  
5d41402abc4b2a76b9719d911017c592 -
```

```
root@kali:~# openssl md5 pw  
MD5(pw)= 859b6a9be3b45262c4414bd1696ba91b
```

```
root@kali:~# md5sum pw  
859b6a9be3b45262c4414bd1696ba91b pw
```

Hash methods supported:

```
md2          md4          md5          rmd160        sha  
sha1
```



Authentication

Message Hash

[Path] / filename

[C:\windows\System32\
12520437.cpx
12520850.cpx
8point1.wav
aaclient.dll
AC3ACM.acm
Ac3audio.ax
ac3filter.cpl
accessibilitycpl.dll
ACCTRES.dll
acledit.dll
.
.
ZSHP1020.CHM
ZSHP1020.EXE
ZSHP1020.HLP
ZSPOOL.DLL
ZTAG.DLL
ZTAG32.DLL

MD 5 sum

0a0feb9eb28bde8cd835716343 b03b14
d 69ae057cd82d04ee7d311809 abefb2a
beab 165fa58ec5253185 f32e124685d5
ad 45dedfdcf69a28cbaf6a2ca84b5f1e
59683d1e4cd0b1ad6ae32e1d627ae25f
4b87d889edf278e5fa223734 a9bbe79a
10b27174d46094984 e7a05f3c36acd2a
ac 4cecc86eeb8e1cc2e9fe022cff3ac1
58f57f2f2133a2a77607 c8ccc9a30f73
0bcee3f36752213d1b09d18e69383898

c 671ed [Path] / filename

96e45a
[C:\windows\system32\
12520437.cpx
12520850.cpx
8point1.wav
aaclient.dll
AC3ACM.acm
Ac3audio.ax

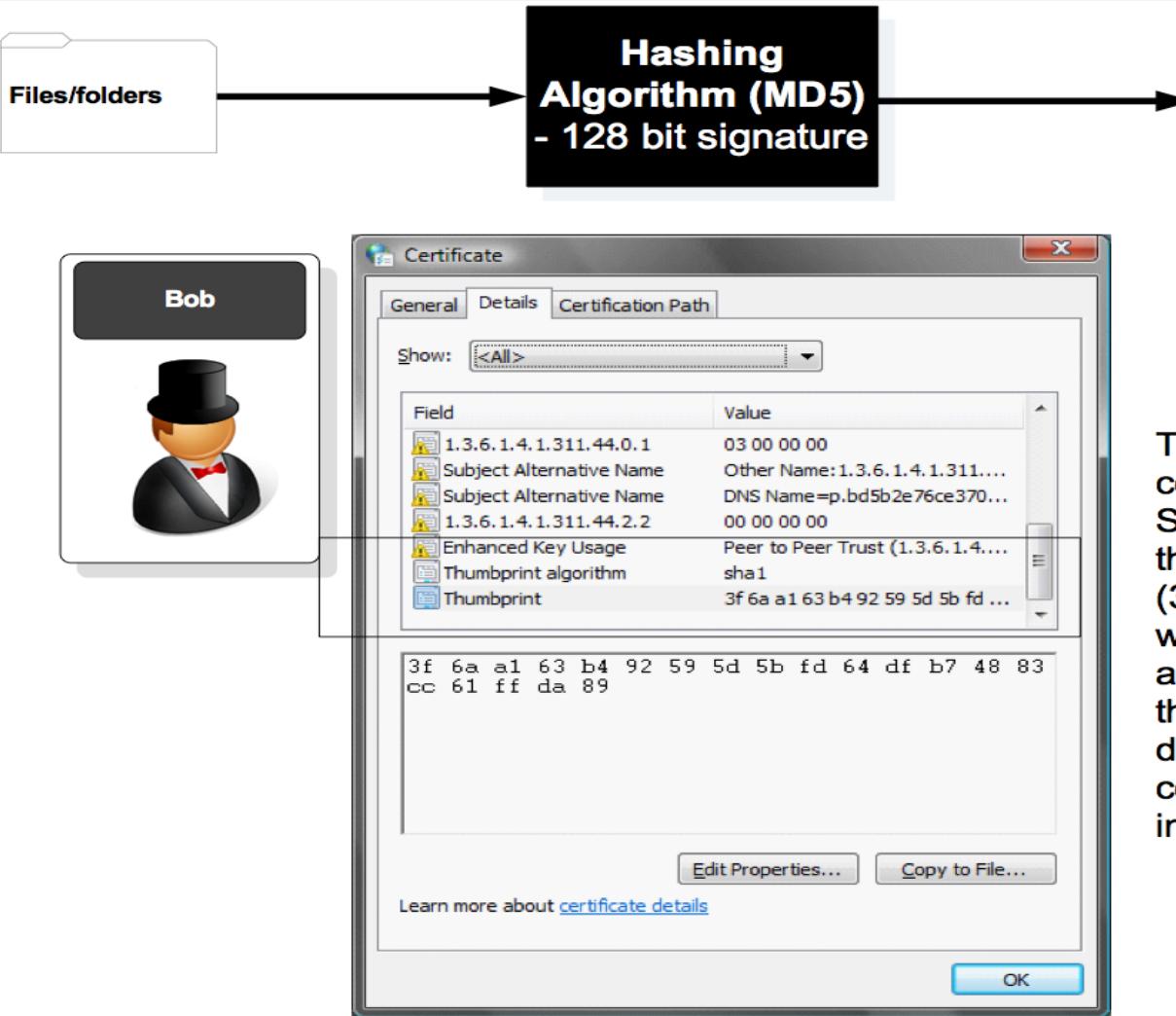
MD 5 sum

Cg /rnrKL3ozYNXFjQ7A7FA==
1prgV82C0E7n0xGAmr77Kg==
vqswX 6w0xSUxhfMuEkaF 1Q==
rUXe 39z2mijLr2osqEtFhg==
wg 9HkzQsa1q4y4dYnriXw==
S 4fYie3ye0X6Ijc0qbvnmg==

Hashing
Algorithm (MD5)
- 128 bit signature

Hash signature

- Hash signatures are used to gain a signature for files, so that they can be checked if they have been changed.



Hash signature

- Hash signatures are used to identify that a file/certificate has not been changed.

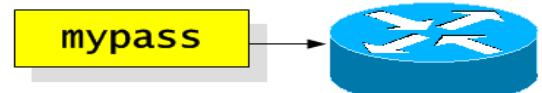
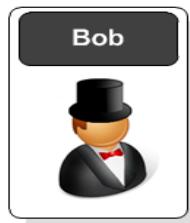
The digital certificate has an SHA-1 hash thumbprint (3f6a...89) which will be checked, and if the thumbprint is different, the certificate will be invalid.

Windows login/ authentication



NT-password
hash for Windows
NT, XP and Vista

Cisco password storage (MD5)



MD5 encoded
password

```
# config t
(config)# enable secret test

Current configuration : 542 bytes
!
version 12.1
no service single-slot-reload-enable
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname Router
!
enable secret 5 $1$/Nwk$knSEQYXZvengjwogj/TGk0
```

One-way hash

- Hashes are used for digital fingerprints (see the next unit) and for secure password storage.
- Typical methods are NT hash, MD4, MD5, and SHA-1.

One-way hash

Types

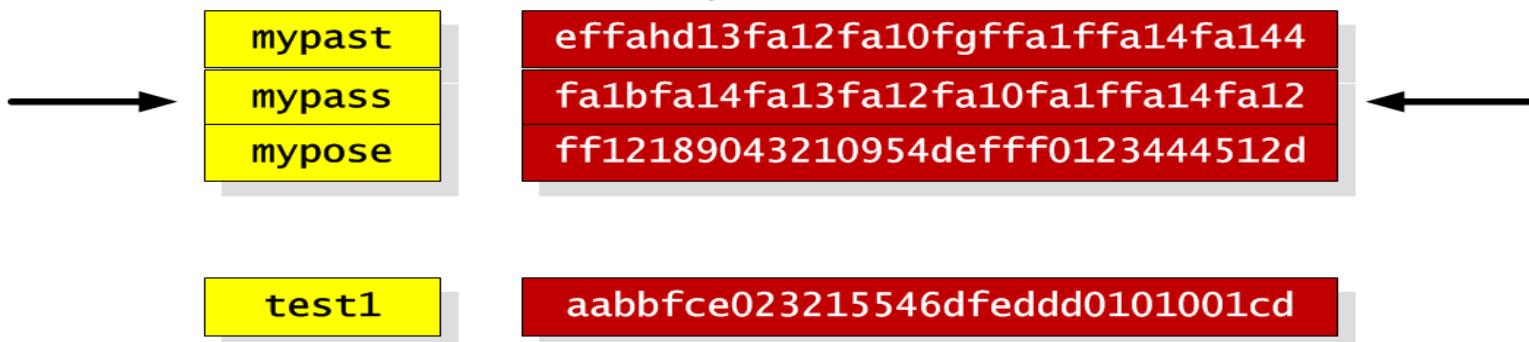
One-way hash

Windows login/ authentication



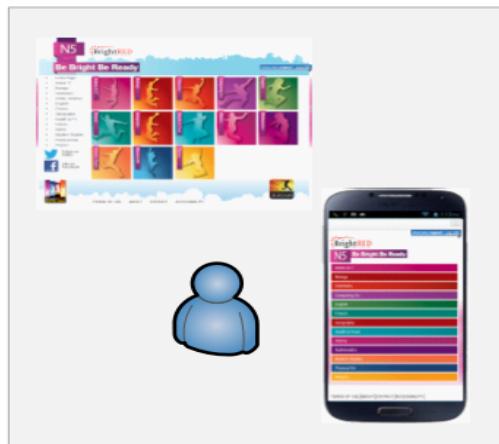
Hashing suffers from **dictionary attacks** where the signatures of well known words are stored in a table, and the intruders does a lookup on this

NT-password
hash for Windows
NT, XP and Vista





Risk 4: One Password Fits All



TJ-maxx
Marshalls.

47 million accounts



1 million accounts – in plain text. 77 million compromised

LinkedIn

6.5 million accounts
(June 2013)



150 million accounts compromised

#	Count	Ciphertext	Plaintext
1.	1911938	EQ7fIpT7i/Q=	123456
2.	446162	j9p+HwtWWT86aMjgZFLzYg==	123456789
3.	345834	L8qbAD3j13jioxG6CatHBw==	password
4.	211659	BB4e6x+b2xLioxG6CatHBw==	adobe123
5.	201580	j9p+HwtWWT/ioxG6CatHBw==	12345678
6.	130832	5djv7ZCI2ws=	qwerty
7.	124253	dQi0asWPYvQ=	1234567
8.	113884	7LqYZKVeq8I=	111111
9.	83411	PMDTbP0LZxu03SwrFUVYGA==	photoshop
10.	82694	e6MPXQ5G6a8=	123123



One account hack ... leads to others



Dropbox
compromised 2013

citigroup

200,000 client accounts

Brute Force - How many hash codes?

- 7 digit password with [a-z] ... how many?
 - Ans:
 - Time to crack - 100 billion per second:
- 7 digit with [a-zA-Z] ... how many?
 - Ans:
 - Time to crack – 100 billion per second:
- 7 digit with [a-zA-Z!@#\$%^&*()] ... how many?
 - Ans:
 - Time to crack – 100 billion per second:

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Secret Shares.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>

LM Hash

LM Hash. LM Hash is used in many versions of Windows to store user passwords that are fewer than 15 characters long.

SHA-3

SHA-3. SHA-3 was known as Keccak and is a hash function designed by Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. MD5 and SHA-0 have been shown to be susceptible to attacks, along with theoretical attacks on SHA-1. NIST thus defined there was a need for a new hashing method which did not use the existing methods for hashing, and setup a competition for competing algorithms. In October 2012, Keccak won the NIST hash function competition, and is proposed as the SHA-3 standard.

Tiger

Bcrypt

Bcrypt. This creates a hash value which has salt.

RIPEMD

RIPEMD (RACE Integrity Primitives Evaluation Message Digest) and GOST. RIPEMD160. RIPEMD is a 128-bit, 160-bit, 256-bit or 320-bit cryptographic hash function, and was created by Hans Dobbertin, Antoon Bosselaers and Bart Preneel. It is used on TrueCrypt, and is open source. The 160-bit version is seen as an alternative to SHA-1, and is part of ISO/IEC 10118

Tiger. Tiger is a 192-bit hash function, and was designed by Ross Anderson and Eli Biham in 1995. It is often used by clients within Gnutella file sharing networks, and does not suffer from known attacks on MD5 and SHA-0/SHA-1. Tiger2 is an addition, in which the message is padded with a byte of 0x80 (in a similar way to MD4, MD5 and SHA), whereas in Tiger it is 0x01. Otherwise the two methods are the same in their operation.

Murmur

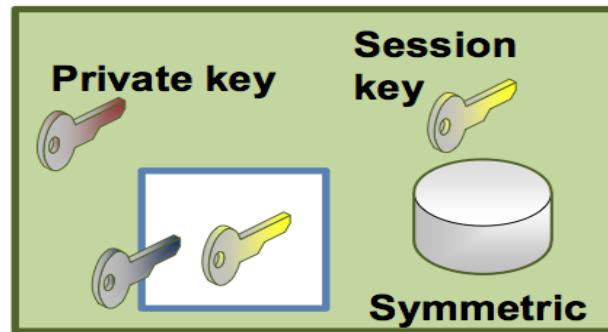
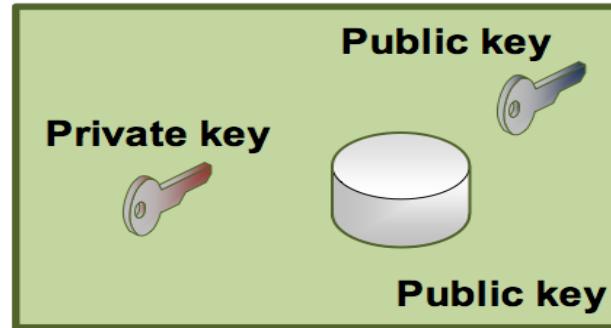
While hashing methods such as MD5 and SHA-1 use crypto methods, the Murmur and FNV hashes uses a non-cryptographic hash function. The Murmur hash, designed by Austin Appleby, uses a non-cryptographic hash function. This can be used for general hash-based lookups. It has a good performance compared with other hashing methods, and generally provide a good balance between performance and CPU utilization. Also it performs well in terms of hash collisions.

FNV

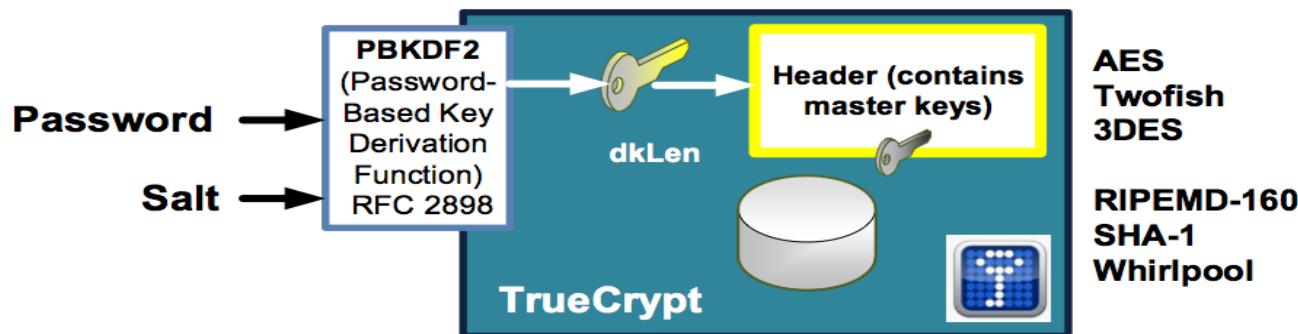
FNV (Fowler–Noll–Vo) is a 64-bit non-cryptographic hash function developed by Glenn Fowler, Landon Curt Noll, and Phong Vo. There are two main versions, of which 1a is the most up-to-date version.



Bob



Alice (Web)



DK = PBKDF2(PRF, Password, Salt, c, dkLen)

DK = PBKDF2(HMAC-SHA1, passphrase, ssid, 4096, 256)

Encrypting disks

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Secret Shares.



Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>

Adding salt

- Salt increases the range of the possible signatures



NT-password
hash for Windows
NT, XP and Vista

Salt increase the range of the signatures



password

\$1\$fred\$bATAk8UUH/IDAp9sd6IUv/

1

fred



bATAk8UUH/IDAp9sd6IUv/

password

bATAk8UUH/IDAp9sd6IUv/

fred

```
C:\openssl>openssl passwd -1 -salt fred password  
$1$fred$bATAk8UUH/IDAp9sd6IUv/
```



```
# cat /etc/shadow
root:$1$Etg2ExUZ$F9NTP7omafhK1lqaBMqng1:15651:0:99999:7:::
# openssl passwd -1 -salt Etg2ExUZ redhat
$1$Etg2ExUZ$F9NTP7omafhK1lqaBMqng1
```

```
$ openssl version
OpenSSL 1.0.1f 6 Jan 2014
```

```
$ openssl dgst -md5 file
MD5(file)= b1946ac92492d2347c6235b4d2611184
```

```
$ openssl genrsa -out mykey.pem 1024
Generating RSA private key, 1024 bit long modulus
.
.
.
e is 65537 (0x10001)
```

```
$ openssl rsa -in mykey.pem -pubout > mykey.pub
writing RSA key
```

```
$ cat mykey.pub
-----BEGIN PUBLIC KEY-----
MIIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDXv9HSFkpM+ZoOQcpdHBZiuwX8
EzIKm0nsgjc5ZTYVaF9CMLtmKoTzep7aQX9o9nKepFt1kq73Ta9v0Pd6Cx61/cgY
xy2tShw0imrtFaVDFjX+7kLmc0uwbFFCoZMtJxIaXaa9SV2kARxOCTJ2u0jRTCCe
XU09IJGHnIhSNJeIJQIDAQAB
-----END PUBLIC KEY-----
```

```
$ cat /etc/shadow
root:$1$Etg2ExUZ$F9NTP7omafhK1lqaBMqng1:15651:0:99999:7:::
```

```
$ openssl passwd -1 -salt Etg2ExUZ redhat
$1$Etg2ExUZ$F9NTP7omafhK1lqaBMqng1
```

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Secret Shares.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>

A major factor with hash signatures is:

- **Collision.** This is where another match is found, no matter the similarity of the original message. This can be defined as a **Collision attack**.
- **Similar context.** This is where part of the message has some significance to the original, and generates the same hash signature. This can be defined as a Pre-image attack.
- **Full context.** This is where an alternative message is created with the same hash signature, and has a direct relation to the original message. This is an extension to a Pre-image attack.

In 2006 it was shown that MD5 can produce collision within less than a minute.

A 50% probability of a collision is:

$$\sqrt{N(\text{signatures})} = \sqrt{2^n} = 2^{\frac{n}{2}}$$



where n is the number of bits in the signature. For example, for MD5 (128-bit) the number of operations that would be required for a better-than-50% chance of a collision is:

$$2^{64}$$

Note, in 2006, for SHA-1 the best time has been 18 hours

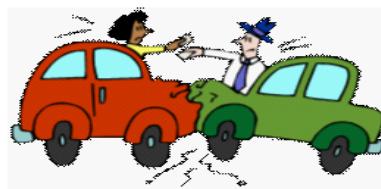
```
d131dd02c5e6eec4693d9a0698aff95c  
2fcab58712467eab4004583eb8fb7f89  
55ad340609f4b30283e488832571415a  
085125e8f7cdc99fd91dbdf280373c5b  
d8823e3156348f5bae6dacd436c919c6  
dd53e2b487da03fd02396306d248cda0  
e99f33420f577ee8ce54b67080a80d1e  
c69821bcb6a8839396f9652b6ff72a70
```

```
d131dd02c5e6eec4693d9a0698aff95c  
2fcab50712467eab4004583eb8fb7f89  
55ad340609f4b30283e4888325f1415a  
085125e8f7cdc99fd91dbd7280373c5b  
d8823e3156348f5bae6dacd436c919c6  
dd53e23487da03fd02396306d248cda0  
e99f33420f577ee8ce54b67080280d1e  
c69821bcb6a8839396f965ab6ff72a70
```

The MD5 signature
gives the same
result



79054025255FB1A26E4BC422AEF54EB4





Nat McHugh

- 10 hours of computing on the Amazon GPU Cloud.
- Cost: 60 cents
- Used: Hashcat (on CUDA)
- Birthday attack: A group size of only 70 people results in a 99.9% chance of two people sharing the same birthday.
- M-bit output there are 2^m messages, and the same hash value would only require $2^{(m/2)}$ random messages.
 $18,446,744,073,709,551,616$.

```
C:\openssl>openssl md5 hash01.jpg  
MD5(hash01.jpg)= e06723d4961a0a3f950e7786f3766338
```

```
C:\openssl>openssl md5 hash02.jpg  
MD5(hash02.jpg)= e06723d4961a0a3f950e7786f3766338
```

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Secret Shares.

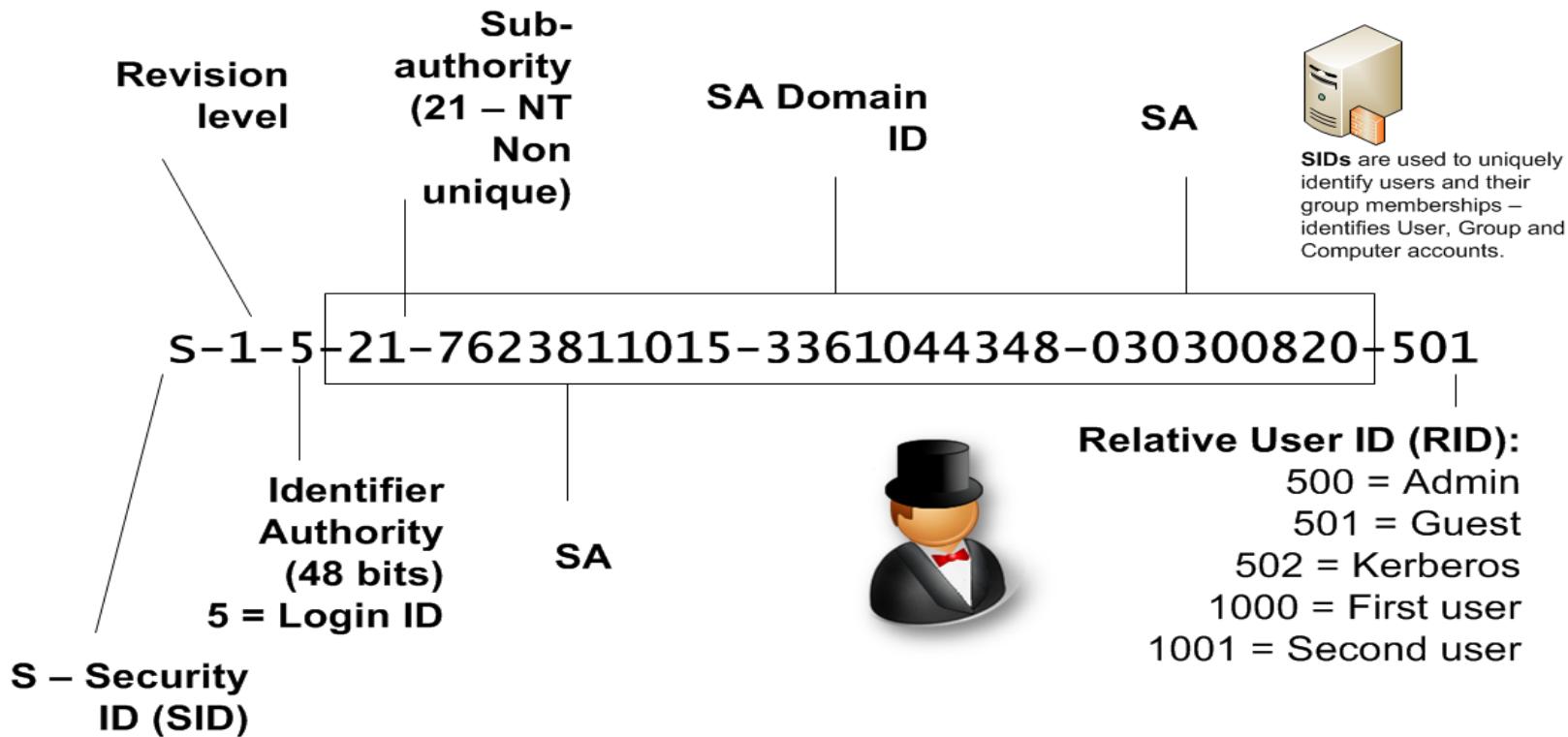
Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>

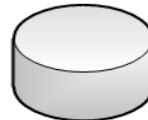
```
C:> user2sid \pluto guest  
S-1-5-21-7623811015-3361044348-030300820-501  
C:> sid2user 5 21 7623811015 3361044348 030300820 500  
Name is Fred  
Domain is PLUTO
```

Windows
CEH

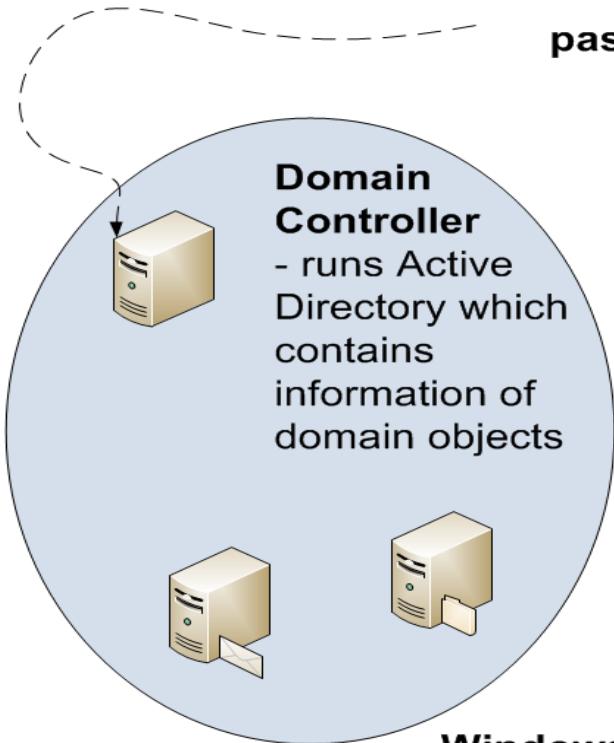
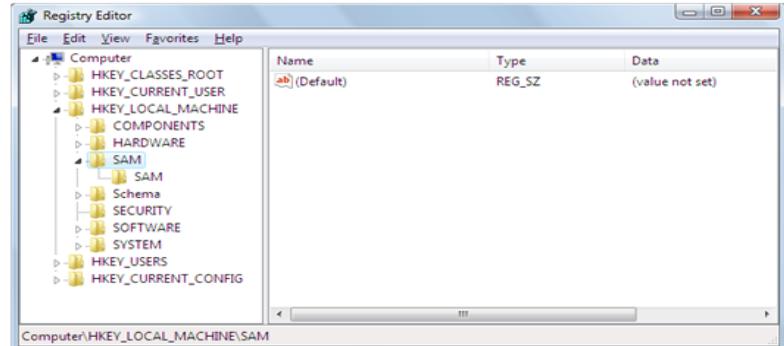




HKLM\SAM



SAM Database
(stores
usernames
and
passwords)



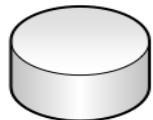
Subsystem (Lsass) – Windows Security mechanism – Attached by Sasser Worm which exploited a buffer overflow



Local Authority

- Responsible for local security policy**
- Controls access.
 - Managing password policies.
 - User authentication.
 - Audit messages.

SAM



Registry: HKEY_LOCAL_MACHINE\SAM



- LM Hash (Windows XP, 2003)
- NTLMv2 (Windows 7, 8, etc) – connect to Active Directory
- NTLM (Windows 7, 8, etc) – No salt

```
C:\Windows\System32\config>dir  
Volume in drive C has no label.  
Volume Serial Number is A2B3-7C7A
```

```
Directory of C:\Windows\System32\config  
05-Oct-14 05:52 PM      262,144 SAM  
05-Oct-14 05:56 PM      262,144 SECURITY  
05-Oct-14 08:39 PM    149,946,368 SOFTWARE  
05-Oct-14 08:40 PM    15,728,640 SYSTEM
```

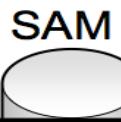
- bkhive - dumps the syskey bootkey from a Windows system hive.
- samdump2 - dumps Windows 2k/NT/XP/Vista password hashes.

hashme gives: FA-91-C4-FD-28-A2-D2-57-AA-D3-B4-35-B5-14-04-EE
FF2A43841C84518A18795AB6E3C8A62E (NTLM)

napier gives: 12-B9-C5-4F-6F-E0-EC-80-AA-D3-B4-35-B5-14-04-EE
307E40814E7D4E103F6A69B04EA78F3D (NTLM)

<user>:<id>:<LM hash>:<NTLM hash>:<comment>:<homedir>:

```
Root@kali:~# cat pw  
myuser:500:12B9C54F6FE0EC80AAD3B435B51404EE:307E40814E7D4E103F6A69B04EA78F3D:::  
Root@kali:~# john pw  
Loaded 1 password hash (LM DES [128/128 BS SSE2])  
NAPIER          (napier)  
guesses: 1  time: 0:00:00:00 100% (1)  c/s: 4850  trying: NAPIER - N4PI3R  
Use the "--show" option to display all of the cracked passwords reliably
```



Registry: HKEY_LOCAL_MACHINE\SAM

```
Root@kali:~# cat pw
myuser:500:12B9C54F6FE0EC80AAD3B435B51404EE:307E40814E7D4E103F6A69B04EA78F3D:::
Root@kali:~# john pw
Loaded 1 password hash (LM DES [128/128 BS SSE2])
NAPIER          (napier)
guesses: 1  time: 0:00:00:00 100% (1)  c/s: 4850  trying: NAPIER - N4PI3R
Use the "--show" option to display all of the cracked passwords reliably

<user>:<id>:<LM hash>:<NTLM hash>:<comment>:<homedir>:
password:500:E52CAC67419A9A224A3B108F3FA6CB6D:8846F7EAEE8
FB117AD06BDD830B7586C:$
myuser:500:12B9C54F6FE0EC80AAD3B435B51404FF·307E40814F7d4
E103F6A69B04EA78F3D:::
```

The ophcrack interface shows the following data:

User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
password	E52CAC67...	8846F7EA...	PASSWOR	D	password
myuser	12B9C54F6...	307E40814...	NAPIER	empty	napier

Progress: 34% in RAM

Preload: done Brute force: done Pwd found: 2/2 Time elapsed: 0h 0m 17s



Hash Crackers/Bit Coin Miners



25 GPU Hash Cracker

- An eight character NTLM password cracked in 5.5 hours. 14 character LM hash cracked in six minutes. 350 billion hashes per second.

Fast Hash One

- 1.536TH/s – Cost 3-5,000 dollars.



Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>

Benchmark

Ultra fast:	
Murmur:	545,716 hashes per second
Fast:	
SHA-1:	134,412
SHA-256:	126,323
MD5:	125,741
SHA-512:	76,005
SHA-3 (224-bit):	72,089
Medium speed:	
LDAP (SHA1):	13,718
MS DCC:	9,582
NT Hash:	7,782
MySQL:	7,724
Postgres (MD5):	7,284
Slow:	
PBKDF2 (SHA-256):	5,026
Cisco PIX:	4,402
MS SQL 2000:	4,225
LDAP (MD5):	4,180
Cisco Type 7:	3,775
PBKDF2 (SHA1):	2,348
Ultra-slow:	
LM Hash:	733
APR1:	234
Bcrypt:	103
DES:	88
Oracle 10:	48

Hashes "The quick brown fox jumps over the lazy dog:

SHA-1:	2fd4e1c67a2d28fcfd849ee1bb76e7391b93eb12
SHA-256:	d7a8fbb307d7809469ca9abcb0082e4f8d5651e46d3cdb762d02d0bf37c9e592
SHA-512:	07e547d9586f6a73f73fbac0435ed76951218fb7d0c8d788a309d7 85436bbb642e93a252a954f23912547d1e8a3b5ed6e1bfd7097821233fa0538f3db854fee6
MD-5:	9e107d9d372bb6826bd81d3542a419d6
DES:	ZDeS94Lcq/6zg
Bcrypt:	\$2a\$05\$2czCv5GYgkx3aobmEyewB.ejV2hePMdbvTdCyNaSzWtIGPPjB2xx6
APR1:	\$apr1\$ZDzPE45C\$3PvRanPycmNc6c2G9wT9b/
PBKDF2 (SHA1):	\$pbkdf2\$5\$WkR6UEU0NUM\$0RB2bimWrMY.EPYibpaBT2q3HFg
PBKDF2 (SHA-256):	\$pbkdf2-sha256\$5\$WkR6UEU0NUM\$yrJz2oJix7uBJZwZ/50vWUgdE I/i0ffqeU4obqC0pk4
LM Hash:	a7b07f9948d8cc7f97c4b0b30cae500f
NT Hash:	4e6a076ae1b04a815fa6332f69e2e231
MS DCC:	efa9778bbc94a7360f664eb7d7144725
LDAP (MD5):	{MD5}9e107d9d372bb6826bd81d3542a419d6
LDAP (SHA1):	{SHA}2fd4e1c67a2d28fcfd849ee1bb76e7391b93eb12
MS SQL 2000:	0x0100BF77CE595DCD1FC87A37B3DEBC27A8C97355CB96B8BAB 63E602662BA5D5D33B913E422499BE72FF3D9BB65DE
MySQL:	*A4E4D26FD0C6455E23E2187C3AABE844332AA1B3
Oracle 10:	4CDA2299FCAD0499
Postgres (MD5):	md5d44c15daa11770f25c5350f7e5408dd1
Cisco PIX:	kGyKN5CqdFQ1qJUs
Cisco Type 7:	15260309443B3E2D2B3875200108010D41505640135E1B0E080 519574156401540035E460B594D1D53020B5C

Benchmark

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

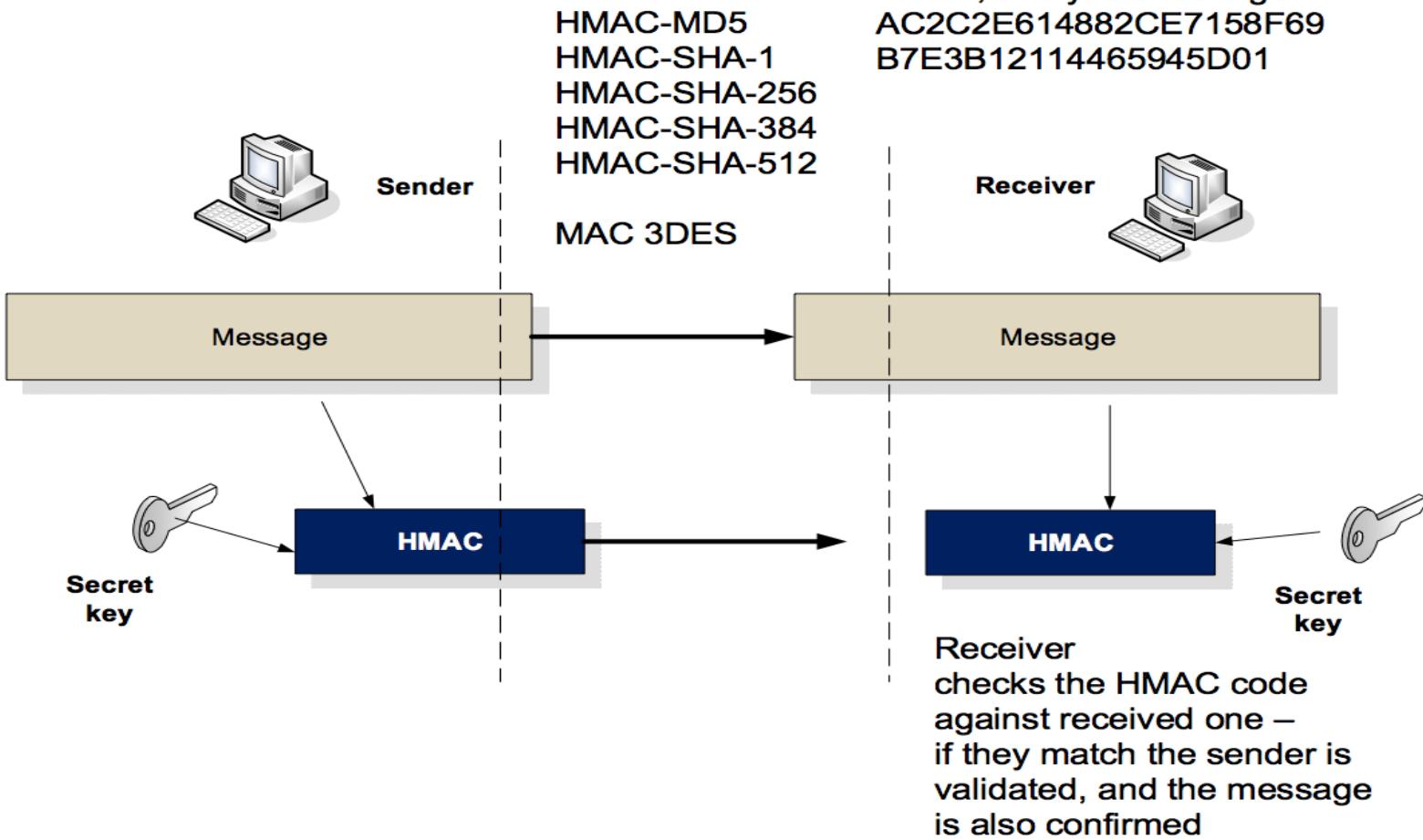
Message Authentication Codes (MACs).

OTP/HOTP.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto02>

<http://asecuritysite.com/encryption>



Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>



One-time password

$f(m)$

$f(f(m))$

$f(f(f(m)))$



System logon

One-time password (timed)

$H(t_1)$

$H(t_2)$

$H(t_3)$



System logon

One-time password (counter)

$H(c_1)$

$H(c_2)$



System logon

Chapter 3: Hashing

Hashing Types.

Hashing Methods.

Salting.

Collisions.

LM and NTLM Hashes (Windows).

Hash Benchmarks.

Message Authentication Codes (MACs).

OTP/HOTP.

Prof Bill Buchanan OBE

<http://asecuritysite.com/crypto03>

<http://asecuritysite.com/encryption>