Recovering domain cached passwords

Passcape Software

1 Intro

As you probably know already, Windows NT-based operating systems store or, to be more accurate, cache passwords of the domain account on a local machine. That is done in order to allow to log users into their accounts even when the login server is not available for some reason. In this case, the user will have access to any network resources that do not require user authentication.

2 Password caching in Windows NT

Windows NT operating systems use two general types of password caching:

- General caching
- Domain-level caching

The domain-level caching provides at least two functionalities:

1. Provide access to computer resources even if a domain controller is not available. For example, on a laptop user logs on to his domain account. Then the user moves the laptop to a place where the domain is not available. In such a case, Windows would use cached data for logging on to the system locally and ensuring access to the local resources of the computer.
2. Single Sign-On (SSO) - the functionality carries out a one-time network authentication using data obtained during the first interactive sign-on and excluding the reentering of it.

If the domain caching is disabled, an attempt to logon to the server should cause the following message:

The system can not log you on now because the domain DOMAIN_NAME is not available

If the domain controller is not available, the system uses data saved in a cache previously. So the prompt message will look a bit differently:

A domain controller for your domain could not be contacted. You have been logged on using cached account information. Changes to your profile since you last logged on may not be available.

3 Setting up caching options

Number of cached credentials stored on a client side
By editing the Windows registry, you can manually set the required number of logon attempts to be cached by operating system. That value may vary between 0 and 50. If the value is set to 0, caching will be disabled. By default, Windows stores 10 successful logon attempts. Caching is controlled with the following registry key:

Key name: **HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\Current Version\Winlogon**
Value name: **CachedLogonsCount**
Data Type: **REG_SZ**
Values: **0 - 50**

Notification on using domain cached password
By default, if you attempt to connect to a domain (using a Windows-based workstation) while the domain controller is not available, you will not see any warning messages and, therefore, scarcely notice that have signed on using cached credentials. If you want to configure Windows the way that every time you sign on using cached password it would show the corresponding warning, set up these two registry keys:

Key name: **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft \Windows NT\CurrentVersion\Winlogon**
ValueName: **ReportControllerMissing**
Data Type: **REG_SZ**
Values: **TRUE**

And then for each user of the system:

Key name:
HKEY_CURRENT_USER\Software\Microsoft\WindowsNT\CurrentVersion\Winlogon
ValueName: **ReportDC**
Data Type: **REG_DWORD**
Values: **1**

4 Security of the domain cached credentials

The term 'domain cached credentials' doesn't fully reflect the way Windows caches and stores private information. Beginning with Windows 2000, user passwords are not stored as a plaintext. On the contrary, the system stores a password hash, slightly modified with salt (i.e. salted hash), where the salt is the name of the user in the Unicode format.

Microsoft claims that the following conclusions can be drawn from this encryption algorithm:
- Precomputed tables are not applicable since a salted hash is used
- DCC hash cannot be used for signing on to other systems

Approaching this issue from the other point may reveal that:
- Precomputed tables can be created for aliases, i.e. for known user names; e.g., for the built-in Administrator or Guest account. On the other hand, this may turn out to be a waste of time, since a built-in administrator or guest account isn't that frequently enabled, to put it mildly.
- To completely eliminate the possibility of signing on to other systems, the domain name should also be salted along with the user name.

5 Potential attacks on domain cached passwords

A potential malefactor, to gain access to the computer, can easily add a new or overwrite an existing hash with his own value. The procedure assumes physical access to the computer.

However, the mere rewriting or replacement of the cached password doesn't provide the potential malefactor access to the private user data, encrypted files, data protected with DPAPI (Internet Explorer, Outlook, Windows Mail, WPA passwords, and other data).

6 Domain cached credentials - a look from within

Domain cached data is stored encrypted in the Windows registry at the following location: **HKEY_LOCAL_MACHINE\SECURITY\Cache**. The binary value **NL$x** indicates a single cache record, where 'x' stands for the record number. Access to this registry key is granted to the system only.

Each record contains:

- Extended information on the user. Namely: user's short and full name, time of the last access, identifier in the system, domain name, domain DNS name, sign-on domain, script, path to the user profile, home directory, home directory drive, membership in groups, etc.
- Private information, including password hash, additional secrets, and Pbkdf2 iteration counter (used for encryption in Windows Vista and later OSes).

The encryption of the domain cached data involves the LSA secret **NL$KM**, which holds the 64-byte Master Key bound to the local systemr. NL$KM is located in the SECURITY registry file and, in its turn, is also encrypted with the **SYSKEY**.

Thus, the overall domain record decryption scheme looks this way:

- SYSKEY + LSA Master Key + NL$KM = DCC Master Key
- DCC Master Key + NL$x entry = Decrypted DCC entry

Once a cached domain entry is decrypted, we gain access to the salted hash of the user account. The salted hash can be used then to guess the plaintext logon password:

For Windows 2000-2003: hash = MD4 (MD4(user password) + lowercase(user name))

Beginning with Windows Vista, the encryption algorithm has changed a bit. Now it's enforced with additional iterations of the SHA-1 hash:

hash = PBKDF2_SHA(MD4 (MD4(user password)+lowercase(user name)), iterations)

By default, the iterations counter is equal to 10240.

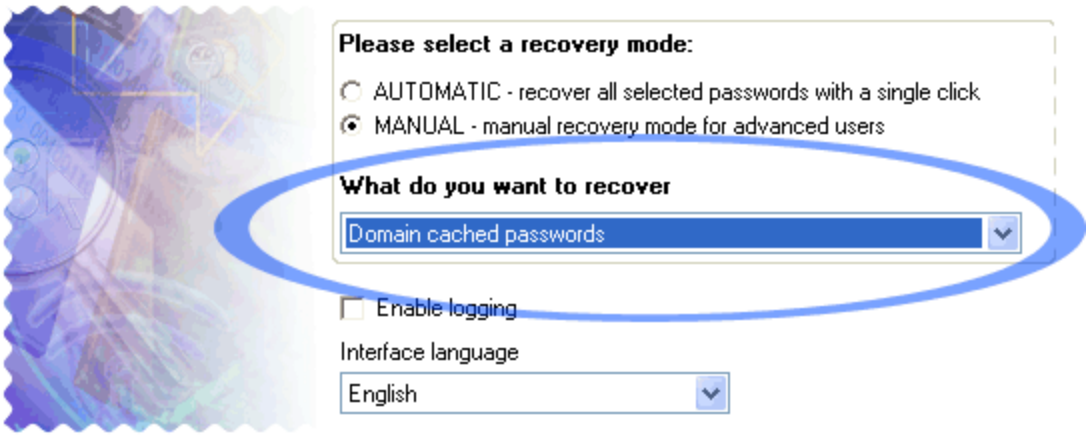
7 Practical guide on recovering domain cached credentials

The only tool we need to go through the entire recovery nightmare is Network Password Recovery Wizard. Let's split the entire workflow into three pieces:

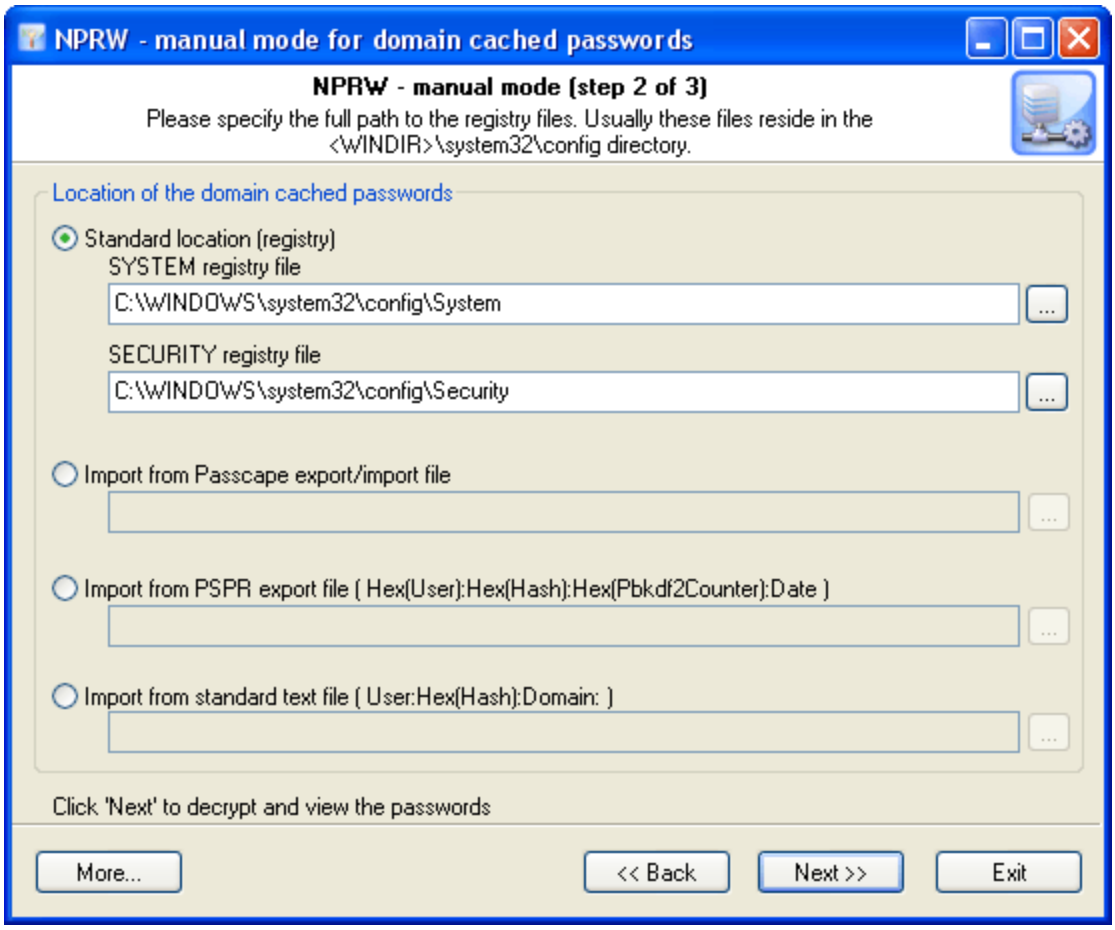
1. Decryption of the domain cached records
2. Analysis of the decrypted records
3. Password recovery of the selected entry

So, let's get going.

1. Decryption of the domain cached records
Launch the application and from the drop-down menu select what we want to recover: *Domain Cached Passwords*. If the cached records are on the local computer, with confidence fire up the automatic mode and move ahead.



The manual operating mode is much more flexible and allows users manually selecting data source, whether that's the standard location (registry files), Passcape's native import/export file or data imported from another program.

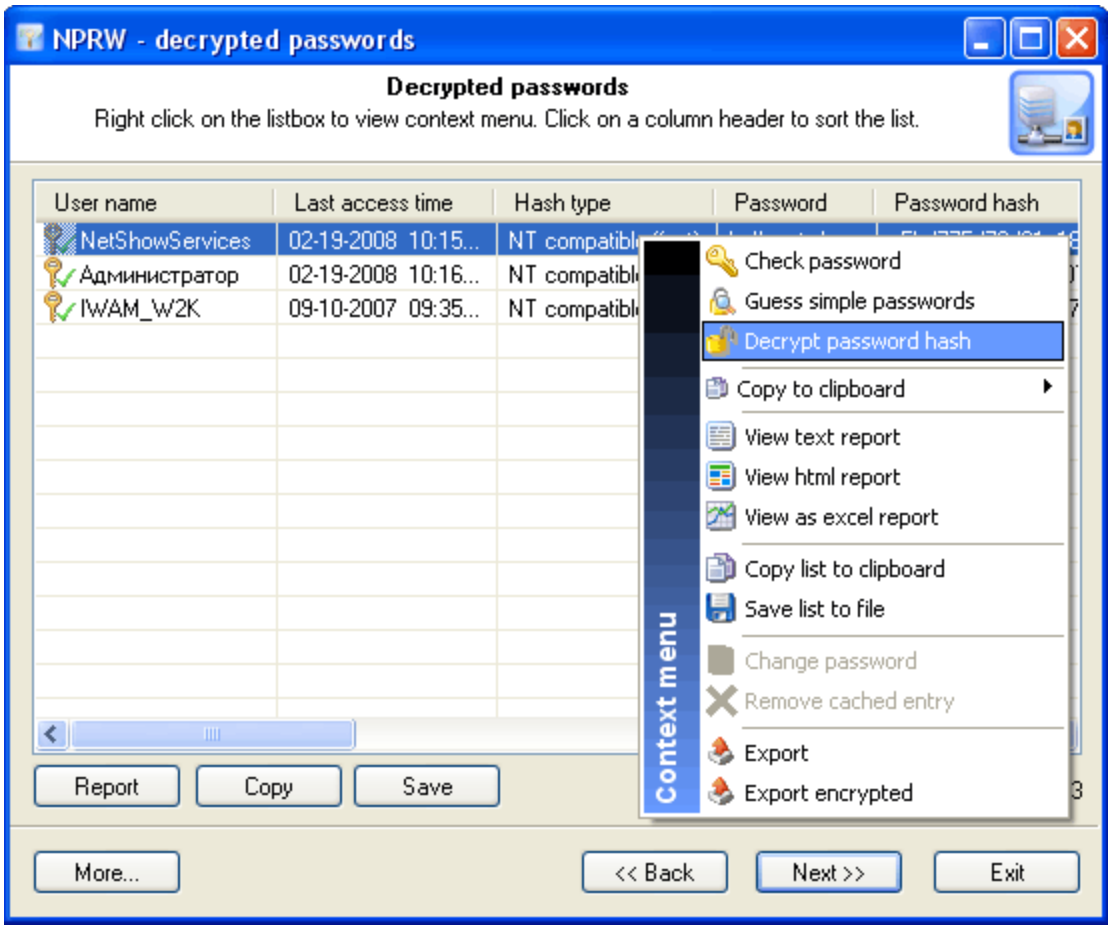


Click the 'Next' and wait until the program finishes processing the input data.

2. Analysis of the decrypted records

Upon a successful decryption and importing credentials, the user should get a list of decrypted records. Each record has several data fields; for instance, user name, server, membership in groups, etc.

Opening up the context menu by right-clicking allows engaging the program's extended capabilities. For the entire list: copying, saving, viewing the report, or exporting the entire list. For an individual record: changing the password, deleting a record, validating the password, recovering the password by searching simple and frequently used combinations, or launching a full-scale attack.



Before launching an attack for the selected password, take a note of these two fields: *'Password'* and *'Hash type'*. Sometimes a cached password can be easily guessed without using a fully scalable attack but rather launching a simple one. In such a case, the plaintext password will be displayed in the 'Password' field, and the record will be marked with the corresponding icon.

The *'Hash type'* field contains a password hash type, which can be one of 3: *'NT compatible instant'* - instant recovery, *'Win2K compatible fast'* - quick recovery at the speed of several millions of passwords per second or *'Vista, slow'* - the recovery speed is only several hundreds of passwords per second on a modern computer.

3. Password recovery of the selected entry

So, to start the password recovery of the selected record, right-click on it and then choose *'Decrypt password hash*' on the context menu. This will open up the decryption method dialog. There is no sense in describing all types of attacks. The detailed information on that can be found in the program's manual.

An unsophisticated user might ask a completely reasonable and coherent question: "There is no 100% guarantee of the recovery, although there are many methods. Which attack should be launched to raise the probability of its successful completion?"

For choosing the type and the sequence of the attacks, we advise to follow this algorithm, which is applicable in the majority of cases to wide range of password type:

- First, enable the preliminary attack option, if it is available. It helps to recover simple and frequently used combinations.
- Second, if you are aware of any specifics of the password you are looking for, it would be nice to try a mask attack or a base-word attack first. Specifically, if you know a part of the password then using a mask attack would be more effective. If you know the basic component of the password or, for example, know the password but don't remember the sequence of caps and lowercase characters in it, a base-word attack would do the job better.

If you have no information on the password you are looking for, be guided by the following sequence of steps:

1. Run the AI attack with mutation and indexing options set to min.
2. If the password was not found, just try again and set the mutation option to the 'normal' level and indexing to the 'deep'.
3. Launch the dictionary attack with the mutation option disabled.
4. Launch the dictionary attack with the mutation option enabled; the depth of mutation depends on the amount of available time and the attack speed. When searching for passwords typed in the national keyboard layout, the depth of mutation should be set to strong.
5. Select and download online dictionaries and repeat steps 3 - 4.
6. Launch the pass-phrase attack with the mutation option disabled.
7. Launch the pass-phrase attack with the mutation option enabled and set to the maximum productivity. This will allow guessing passwords typed in the national keyboard layout.
8. Select and download online pass-phrase dictionaries and repeat steps 6 - 7.
9. Launch the combined dictionary attack with phrase generation rules.
10. Select and download online dictionaries for combined attack and repeat step 9.
11. Select a charset for brute-force attack, launch the attack.
12. If necessary, select a new or complete the old character set and repeat the brute-force attack; i.e. step 11.

Steps 1-2 - Scan your local system for plaintext passwords.
Steps 3-5 - Extract single-word passwords.
Steps 6-10 - Generate multi-word passwords.
Steps 11-12 - Exhaustive search.

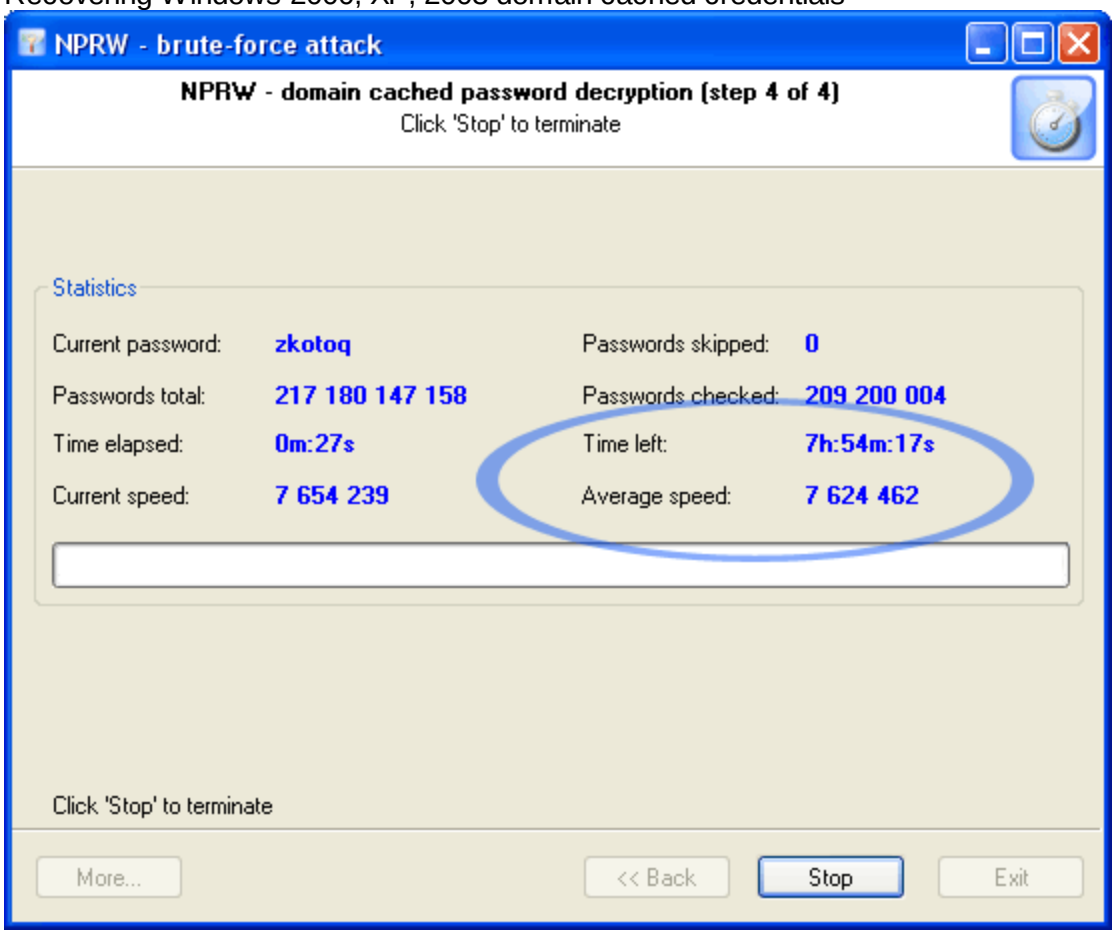
8 Outro

You can take advantage of all kind of attacks in full when decrypting Windows 2000 domain cached credentials. However, when it comes to recovering Windows Vista-compatible domain cache, it's difficult to count seriously on anything but a dictionary attack.

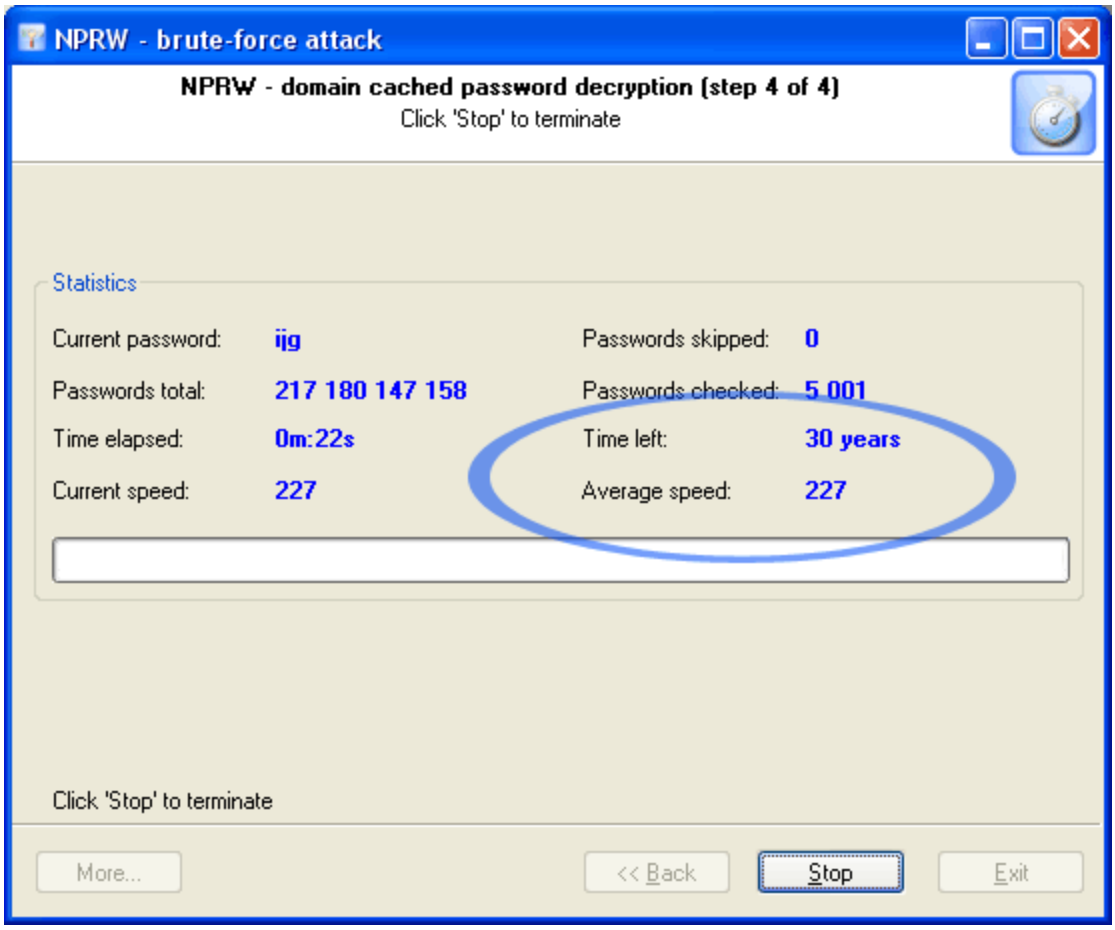
Let's figure out how long it takes to finish up an exhaustive search and recover a simple, eight-character password in both cases. To search through the entire range of a..z characters, we would need to validate not too many and not too few but exactly 217 180 147 158 password combination.

Now, let's take a look at the time.

Recovering Windows 2000, XP, 2003 domain cached credentials



Recovering Windows Vista cached credentials



9 Appendix

C++ algorithm for validating domain cached password in Windows 2000, XP, 2003:

```
BOOL CheckCachedDomainPassword(LPCTSTR cszUserName, LPCTSTR cszPassword, BYTE
pCheckHash[0x10])
{
    WCHAR wsz[256];
    BYTE pHash[0x10];
    INT iLen;

    iLen=strlen(cszPassword);
    MultiByteToWideChar(CP_ACP,MB_PRECOMPOSED,cszPassword,iLen,wsz,256);
    Md4Init();
    Md4Update((LPBYTE)wsz,iLen*2);
    Md4Final(pHash);

    iLen=strlen(cszUserName);
    MultiByteToWideChar(CP_ACP,MB_PRECOMPOSED,cszUserName,iLen,wsz,256);
    CharLowerW(wsz);
```

```
    Md4Init();
    Md4Update(pHash,0x10);
    Md4Update((LPBYTE)wsz,iLen*2);
    Md4Final(pHash);

    return ( memcmp(pCheckHash,pHash,0x10)==0 );
}
```