

/\*\*\*\*\*

## Command Line Interface for the ISC Credential Management Utility (CMU)

Version: 2.0.00

Date: 13 June 2008

Authors: Michael Markowitz, Jonathan Schulze-Hewett

Copyright© 2004-2008 Information Security Corp. All rights reserved.

\*\*\*\*\*/

### USER COMMANDS

CMU(1)

#### NAME

cmu - ISC credential management utility

#### SYNOPSIS

```
cmu -h

cmu b

cmu c [-i akid] [-l logfile] [-q] [-t cfg_file] [-v] [-z zones]

cmu d [-l logfile] [[-u postfile] | [-w postdata]] url file

cmu e [-N] [-c upc-folder] [-d directory_list] [-f] [-k dbfpwd]
      [-l logfile] [-p p12pwd] [-q] [-t cfg_file] [-v] [-z zones] [-L]
      [-P constraints] [-T] [file...]

cmu i [-N] [-c upc-folder] [-d directory_list] [-f] [-k dbfpwd]
      [-l logfile] [-p p12pwd] [-q] [-t cfg_file] [-v] [-z zones] [-L]
      [-P constraints] [-T] [file...]

cmu l [-c upc-folder] [-p p12pwd] [-l logfile] [-t cfg_file] [-v]

cmu m [-N] [-c upc-folder] [-d directory_list] [-f] [-k dbfpwd]
      [-l logfile] [-p p12pwd] [-q] [-t cfg_file] [-v] [-z zones] [-L]
      [-P constraints] [-T] [file...]

cmu p [-c upc-folder] [-l logfile] [-p password] [-s] [-t cfg_file] [-v] [p12file]

cmu r [-a] [-c upc-folder] [-d directory_list] [-f] [-k dbfpwd] [-l logfile]
      [-t cfg_file] [-P constraints] [-v]

cmu s [-N] [-c upc-folder] [-d directory_list] [-f] [-k dbfpwd]
      [-l logfile] [-p p12pwd] [-q] [-t cfg_file] [-v] [-z zones] [-L]
      [-P constraints] [-T] [file...]

cmu u [file...]
```

#### DESCRIPTION

The cmu command helps simplify the PKI experience for end-users by off-loading the 'know-how' of credential management to system administrators. A cmu-based script written by an administrator can be 'pushed down' (along with the cmu application, if necessary) to each end-user system where it can be run largely without manual intervention. cmu functions support assisted PKI enrollment, the installation of end-user credentials, and the synchronization of credentials between several supported Windows applications and, for backup and recovery purposes, a user's personal directory cache (the "UPC"). Some useful CAPI, MAPI, S/MIME, and Outlook configuration functions are also provided.

cmu's primary action is determined by its key argument, the first argument placed after the program name on the command line. The key is a single character from the following list of *function letters*: b, c, d, e, i, l, m, p, r, s, u.

The default UPC directory is 'u:\private\certificates\' , but the -c *upc-folder* argument can be used to specify an alternate directory for the user's credentials. To facilitate identification and avoid duplicates, credentials are maintained in the UPC folder as PKCS#12 files with filenames that are formed by appending the extension '.p12' to the SHA-1 message digest of the end-user certificate contained in the file. (At this time we do not anticipate the need to store PKCS#7 files in the UPC.) Thus the filename coincides with the certificate's "Thumbprint" as it might be displayed by the MSIE GUI when viewing the certificate's properties using the following sequence of menu/tab/button selections: Tools | Internet Options | Content | Certificates | <select certificate> | View | Details.

The -d *directory\_list* argument can be used to specify one or more Windows directories (or subtrees) containing the user's NSS database files ('cert7.db' or 'cert8.db' and 'key3.db'). If this argument is supplied, cmu searches (recursively through each of the directories in *directory\_list*, a comma-delimited list of directories, and all their subdirectories) for certificate and key database files, in addition to performing its default searches through 'u:\\_sys\netscape' and 'u:\\_sys\mozilla'.

Normally, cmu aborts execution of operations that require read and/or write access to one or more Netscape certificate databases if it detects any version of Netscape, Mozilla or FireFox running on the user's system. (This is done to prevent certificate database access conflicts.) The -f option forces cmu to skip runtime checks for Netscape processes.

The -l *logfile* argument can be used to specify a pathname for the log file, thereby overriding the hard-wired default value 'u:\private\certificates\logfile.txt'.

The -p *p12pwd* argument supplies your PKCS#12 password and is required for all Export, Synchronize and Publish to GAL operations. If not explicitly provided on the command line, the user will be interactively prompted for it at runtime.

The -k *dbfpwd* argument supplies your NSS database password and is required for the Reinitialize operation and for all Export and Synchronize operations if your NSS database is password protected. If there are Netscape databases in the directory search paths but this argument is not explicitly provided on the command line, the user will be interactively prompted for it at runtime.

## Function Letters

The function key command line argument must be one of the following letters:

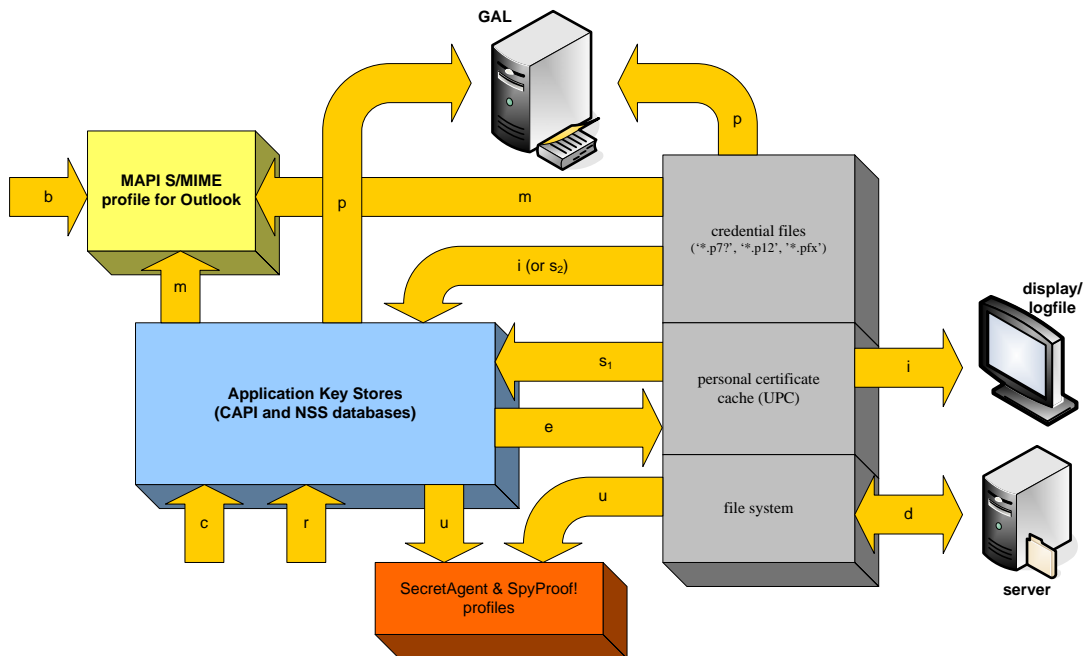
- b Add S/MIME encrypt and sign buttons to Outlook's message composition toolbar. Starts Outlook if it is not already running, but will not terminate it. (Works with Outlook 2000/2002/2003. A special build of the cmu executable is required for Outlook 97/98.)
- c Configure CAPI to use the freshest signing certificate in the user's personal certificate store as the default certificate for client authentication. (The "Client Authentication" setting under "Certificate purposes" is turned off for all certificates in this store other than the freshest signing certificate.) If the '-i' option is used, only certificates issued by the CA with the specified authority key identifier will be considered. If a '-z' option is used, cmu enables the security setting "Don't prompt for client certificate selection when no certificates or only one certificate exists" for all specified internet zones. (For details, see "Configuring Client Authentication in CAPI" in the NOTES section below.)
- d Download specified *url* as *file* via HTTPS. (Assumes 'cmu c' has already been run so that user's freshest signing certificate can be used automatically for client authentication.) If the '-u' or '-w' switches are used, the specified string literal or file contents are POST'ed to the specified *url* operand; otherwise a simple GET is employed. In either case, the results returned by the server are stored in the designated *file* operand.

NOTE: If both '-u' and '-w' are specified, a warning message is written to the logfile and only the contents of the file (the '-u' argument) are POSTed.

- e      Export. Attempts to detect all supported applications installed on the user's system and extracts all of their key pairs as PKCS#12 files in the UPC. The name of each PKCS#12 output file is constructed by appending the extension '.p12' onto a string formed by concatenating together the user's friendly name, a word indicating the keyUsage attribute of that key pair, the notAfter date formatted mmddyyyy and the first 20 characters of the "certificate thumbprint" (i.e., the first half of the SHA-1 message digest of the user's certificate). All existing PKCS#12 files in the UPC ending with the same half certificate thumbprint are deleted (or overwritten) without warning.
  
- i      Import. Causes the contents of all specified PKCS#7 and/or PKCS#12 file(s) to be imported into all supported applications on the user's system. Used by itself, the 'i' function does not alter the contents of the UPC. (PKCS#12 input files must have a '.p12' or '.pfx' extension; PKCS#7 input files must have an extension matching '.p7?'. First all p7 files on the command line are processed, then p12/pfx files).
  
- l      List. Recursively scans the directory tree rooted at the active UPC and displays a list of all PKCS#12 ('\*.p12' or '\*.pfx') files found there together with their existing friendly names (when available), or with the friendly names that they would be assigned during a 'synchronize' operation. If the '-v' option is specified, the output will also display keyUsage and notAfter attribute values.
  
- m      Set the freshest signing and/or encrypting certificate(s) found in CAPI as the S/MIME certificates in the user's default MAPI security profile for use with Outlook. (Creates a new MAPI security profile if one does not already exists.) (See the 'MAPI Security Profile Update' section under NOTES for details.)
  
- p      Publish (to GAL). Extracts the user's private key and certificate from the PKCS#12 file specified on the command line\*, and uses them to create a signed 0-length S/MIME message that is "published" to the user's entry in the global address list (GAL) using MAPI to identify the appropriate Exchange Server and current user account. (For details, see "Publish to GAL" in the NOTES section below.)
  - \* If no filename is specified on the command line the "freshest" PKCS#12 in reply to the CAPI "My" store is used.
  
- r      Reinitialize. Netscape database files associated with the user's default profile are copied into the active UPC and renamed (by prefixing their existing names with the current date and time). Then the original database files are deleted and new database files are created with the specified password. (To locate the appropriate default profile, cmu searches 'u:\\_sys\netscape', 'u:\\_sys\mozilla', and all directory trees specified with '-d', in an attempt to find a Netscape database folder containing "default" in its pathname. Only the first such folder containing a cert7.db or cert8.db file is processed.)
  
- s      Synchronize. Recursively scans the directory tree rooted at the active UPC for PKCS#12 ('\*.p12' or '\*.pfx') files and imports them, together with all PKCS#7 ('\*.p7b') and PKCS#12 input files specified on the command line, into all supported applications installed on the user's system.
  
- u      Update. Install freshest certificates into SecretAgent and SpyProof! profiles.

Note that with release 2.0, multiple functions can no longer be combined in a single invocation of the program; if cmu multiple functions are required, multiple invocations of the executable must be made.

The following diagram attempts to illustrate the actions of the various functions provided by cmu:



## OPTIONS

The following options are supported:

- L Perform CAPI-related operations using the local machine store (CERT\_SYSTEM\_STORE\_LOCAL\_MACHINE) rather than the default current user store (CERT\_SYSTEM\_STORE\_CURRENT\_USER). This allows processes with administrative privileges to silently install trusted root certificates on the user's system.
- N Perform requested operation on (or using) Netscape databases; ignore CAPI stores.
- P *constraints*  
Specify the set of requirements that user-created passwords must meet. *Constraints* is a string of the following form #ansp where:

Item	Description
#	A number specifying the minimum password length
a	The password must include alphabetic characters
n	The password must include numeric characters
s	The password must include special characters ~@#\$\$%^&*()-_+={}[]\ <>/\
p	The password must include punctuation `.,?!"';:!

For example `-P 17ap` requires any user created password to be at least 17 characters long and to contain at least one letter and at least one punctuation mark.

- T Import end-entity certificates in found PKCS#7 files into the "Trusted Publishers" CAPI store rather than the "Other People" store.
- a Process all databases in the NSS database search path when reinitializing NSS databases with the 'r' operation. If this option is omitted, cmu will process only the *first* database found in 'u:\\_sys\netscape', 'u:\\_sys\mozilla' (or their D\_NETSCAPE or D\_MOZILLA replacements), or in a "default" subdirectory of a '-d' argument.
- c *upc-folder*  
Use the specified *upc-folder* rather than 'u:\private\certificates' as the UPC. (The specified folder is tested for existence, and created if necessary, only when used with the 'e' and 's' functions.)

**-d *directory\_list***  
 Recursively scan each element of the comma-delimited *directory\_list* and their child directories for NSS database files. For example, one might wish to use a command line argument of the form "-d u:\\_sys\Phoenix,g:\apps\Mozilla" or "-d c:\Documents and Settings\John Doe\Application Data\Phoenix\Profiles" to search for NSS databases in Firefox profiles. (Note that the specified *directory\_list* subtrees are scanned *in addition* to the two hard-wired default searches rooted in 'u:\\_sys\netscape' and 'u:\\_sys\mozilla', or their D\_NETSCAPE/D\_MOZILLA replacements.)

NOTE: If some element of the specified *directory\_list* argument is too close to the root of a large directory tree, the recursive search through its child directories could be quite expensive in terms of wasted execution time!

**-f** Force execution regardless of whether the user is currently running a Netscape, Mozilla, or FireFox application.

**-h** Display brief usage summary and exit.

**-i *akid*** Configure CAPI to use for TLS client authentication ("-c") the freshest signing certificate issued by the CA with authority key identifier *akid*. (The *akid* is normally specified as 40 uppercase hex digits - representing the SHA-1 hash of the issuer's public key -- with no embedded spaces.)

**-l *logfile***  
 Use the specified log file rather than 'u:\private\logfile.txt' as the log file in which to record all functions performed by the current invocation of the cmu command. If the specified file does not exist and cannot be created, the file '\logfile.txt' is used instead.

**-k *dbfpwd***  
 NSS database password. Must be provided for 'i', 'e' and 's' operations if your key database is password protected. If there are Netscape databases in the directory search paths but this argument is not explicitly provided on the command line, the user will be interactively prompted for it at runtime.

**WARNING:** Do not group any other simple command line switches before the '-k' switch. Doing so may cause its *dbfpwd* argument to be leaked to the log file.

**-p *p12pwd***  
 Use the specified password *whenever possible* when importing private keys into (or extracting exportable private keys out of) one of the supported applications or, with the p function, from the supplied *p12file*. (Use of this option with the 'e' function may or may not suppress an application's native password dialog.) This argument is required if any one of the letters 'e', 'l' or 's' is included in the function key; if not explicitly provided on the command line, the user will be interactively prompted for it at runtime.

**WARNING:** Do not group any other simple command line switches before the '-p' switch. Doing so may cause its *p12pwd* argument to be leaked to the log file.

**-q** Quiet operation. When importing PKCS#12 files into the CAPI store, suppresses the "Importing a new private exchange key" dialog that normally allows a use to specify a security level (and optionally to set a password) for the private key being imported.

**-s** Do not modify the userSMIMECertificate attribute when publishing to GAL; only userCertificate is updated.

**-t *cfg\_file***  
 Use text strings in *cfg\_file* to override various hardwired default values, such as user prompts and NSS database directories. *cfg\_file* must be an ASCII text file containing lines of the form:

*tag=string\_value*

where *tag* is one of strings defined in the following table.

Tag	Description
P_P12PWD	prompt for PKCS#12 password
P_P12CONFIRM	prompt to confirm PKCS#12 password
P_PWDMATCH	prompt to reenter password after confirmation mismatch
P_PWDEEMPTY	inform user that an empty password is not acceptable
P_NSSPWD	prompt for NSS database password
P_CLOSEBROWSER	tell user that browser is running and ask that it be closed
P_NSSNEWPWD	prompt for new NSS database password
P_NSSCONFIRM	prompt to confirm new NSS password
D_NETSCAPE	default root directory for Netscape profiles
D_MOZILLA	default root directory for Mozilla profiles
D_NSS320	Default NSS 3.2.0 tools directory
D_NSS361	Default NSS 3.6.1 tools directory
D_NSS390	Default NSS 3.9.0 tools directory

If the '-t' option is omitted, or any tags are missing from the specified *cfg\_file*, string values hard-wired into the cmu executable are used. (See supplied sample 'default.cfg' file for the default values.) Unless redirected, the NSS tools directories are expected to be found immediately below the directory containing the cmu executable.

NOTE: If you provide an empty *string\_value* for either D\_NETSCAPE or D\_MOZILLA, no directories will be searched for databases belonging to that application.

**-u file**

With the 'd' function, POST contents of *file* to specified *url* operand and return result in designated target *file*.

**-v**

Verbose mode. Output to log file extended execution information, including all internally generated 'certutil' and 'pk12util' command lines. (For security reasons password arguments are always suppressed from this output.)

**-w postdata**

With the 'd' function, POST specified *postdata* to specified *url* operand and return result in designated target *file*.

**-z zones**

Internet zones for which the security setting "Don't prompt for client certificate selection when no certificates or only one certificate exists" is to be enabled. Only works in conjunction with the 'c' function. *zones* is a word composed of one or more of the following decimal digits:

Digit	Description
0	URLZONE_LOCAL_MACHINE - local machine
1	URLZONE_INTRANET - intranet
2	URLZONE_TRUSTED - trusted sites
3	URLZONE_INTERNET - internet sites
4	URLZONE_UNTRUSTED - untrusted sites

## OPERANDS

The following operands are supported:

- file*      The pathname of a PKCS#7 or PKCS#12 file containing the user's credentials or organizational certificate chain, or the name of a file in which to store the data returned from the specified *url* by the 'd' function. For the 'i' and 's' functions, an arbitrary number of p7 and/or p12/pfx files may be specified in a whitespace-delimited list on the same command line. If a filename argument is not supplied on the command line for the 'p' function, the "freshest" PKCS#12 file in the cache is published to GAL.
- url*      For the 'd' function, the (HTTPS) URL from which data is to be requested.

## ENVIRONMENT VARIABLES

If the TMP environment variable is defined and set to a valid directory name, temporary certificate and PKCS#12 files will be written into that directory using unique file names during the import of PKCS#7 certificate chains and PKCS#12 files. If the TMP environment variable is not defined, or if it is set to the name of a directory that does not exist, the active UPC directory is used for this purpose. Temporary certificate and PKCS#12 files are deleted once they are no longer required.

## EXIT STATUS

The following values are returned:

- 0 Successful completion of all requested operations.
- >0 An error occurred or a warning was issued -- not all requested operations completed successfully. See the list below for brief descriptions of all possible error codes. Consult the log file for additional details whenever an error occurs.

```
enum {
    SUCCESS = 0,

    // command line errors
    ERR_CMDLINE = 1000, // attempt to combine functions or illegal command line
    ERR_OPTION, // unsupported command line option

    // file-related errors
    ERR_FILE_NOT_FOUND, // specified file not found
    ERR_FILE_EMPTY, // specified file must be non-empty
    ERR_FILE_OPEN, // cannot open file
    ERR_FILE_READ, // failed to read as many bytes as expected
    ERR_FILE_WRITE, // cannot write file

    // password-related errors
    PWD_INVALID, // incorrect password
    PWD_REJECTED, // password fails to meet security requirements
    PWD_NOT_SPECIFIED, // password not specified

    // PDU processing errors
    ERR_P12_NOT_SPECIFIED, // missing p12 filespec
    ERR_P12_PROCESSING, // cannot open or parse p12 file
    ERR_PDU_PARSE, // failed to parse PDU
    ERR_PDU_CREATE, // cannot create appropriate PDU

    // certificate handling errors
    ERR_UPC, // cannot find (or create) personal certificate folder

    // CAPI/MAPI/GAL errors
    ERR_NO_CAPI_CERT, // no appropriate key pair found in CAPI
    ERR_MAPI_COM, // MAPI communications/Outlook configuration error
    ERR_GAL_UPDATE, // cannot update GAL entry
    ERR_P7_IMPORT, // failed to import into CAPI the contents of a p7 file
    ERR_CAPI_EXPORT, // cannot export credentials to CAPI
    ERR_NO_SIGNING_CERT, // signing certificate not found in CAPI
    ERR_ISSUER_NOT_FOUND, // cannot find certificate with specified issuer in CAPI
    ERR_CAPI_CLIENTAUTH, // failed to set client authentication properties for one
    // or more signing certificates
    ERR_CAPI_SMIME, // cannot set signing and encrypting certificates

    // HTTP request errors
    ERR_URL_NOT_SPECIFIED, // URL missing
    ERR_HTTP_PARM, // URL or capture file not specified
    ERR_HTTP_COM, // communications error

    // Netscape errors
    ERR_CLOSE_BROWSER, // database ops shouldn't be performed while browser is running
    ERR_NDB_INIT, // cannot initialize Netscape databases
    ERR_NDB_IMPORT, // cannot import certificates into Netscape databases
    ERR_NDB_EXPORT, // cannot export credentials from Netscape databases

    // Outlook errors
    ERR_OUTLOOK_SMIME, // cannot add buttons to Outlook toolbar

    // ISC application update errors
    ERR_APP_CONFIG, // requested application not installed or configuration error

    // audit trail errors
    ERR_AUDIT_TRAIL, // auditing system error
}
```

## EXAMPLES

The following command:

```
cmu i -p MyPassword mykeys.pl2
```

uses 'MyPassword' to decrypt the input PKCS#12 file 'mykeys.pl2' and import its key pair into CAPI. (Use '-N' if you prefer to import the user credentials into all supported Netscape-based applications installed on the user's system rather than CAPI. In this case, if one or more cert7.db files are found in the default NSS database search paths, the user will be prompted for a database password.) An audit trail of the operations performed by this command is appended to the default log file, 'u:\private\logfile.txt'.

The command:

```
cmu e -f -c c:\tmp\credentials
```

exports the key pairs from all supported and installed applications (even if a Netscape browser is currently running) and stores them as PKCS#12 files in the folder 'c:\tmp\credentials' (which is created if it doesn't already exist). Since a PKCS#12 password is not supplied on the command line, the user will be prompted to enter it at runtime. Similarly, if one or more cert7.db files are found in the default NSS database search paths, the user will also be prompted for a database password. An audit trail of the operations performed by this command is appended to the default log file.

The command:

```
cmu s -l c:\tmp\mylog.txt -p MyPassword
```

recursively scans the directory tree rooted at the default UPC folder 'u:\private\certificates' and imports any '.pl2' and '.pfx' files found there into all supported and installed applications. If one or more cert7.db files are found in the default NSS database search paths, the user will also be prompted for a database password. An audit trail of these operations is appended to the specified log file 'c:\tmp\mylog.txt'.

The command:

```
cmu l
```

prompts the user for his/her PKCS#12 password, then recursively scans the directory tree rooted at the default UPC folder for PKCS#12 files in an attempt to print their friendly names. Friendly names listed in closed braces ([]) actually appear in the corresponding PKCS#12 file. If a PKCS#12 file doesn't contain a friendly name, this command prints in angle brackets (<>) what CAPI thinks the friendly name should be. To view the keyUsage and notAfter attribute values of each certificate, append the '-v' switch to the command line.

The command:

```
cmu r -f
```

attempts to locate the user's Netscape database folder. If it is found, the user is prompted for a new database password and asked to confirm it. The existing database files are renamed and moved into the UPC, then an empty database is created with the new password.

The command:

```
cmu d https://myserver.com/testpages/getfile.jsp c:\tmp\testfile.bin
```

downloads the data provided by the referenced URL (in this case, a JSP page) and stores it in the specified file 'c:\tmp\testfile.bin'. Use the '-u' or '-w' options to optionally POST a literal string ('-w') or the contents of a file ('-u') to the specified URL before storing the POST results in the designated output file.

The command:

```
cmu c -i F9B20F9778E6D5090F2AC47EBBC6E7AA353C76FB -z123
```

configures the user's personal CAPI store so that only the freshest signing certificate issued by the CA with the specified authority key identifier will be used for client authentication. In addition, prompting for client certificate selection is disabled for internet zones 1, 2 and 3 (intranet, trusted sites, and internet sites not otherwise categorized).

## SEE ALSO

This section lists some references on credential management, especially for the supported applications.

[How Users Manage Cryptographic Digital IDs in Outlook](#), Office 2003 Editions Resource Kit, Messaging, Administering Cryptography in Outlook 2003, Microsoft, Sept. 4, 2003.

[How To Assign an S/MIME Certificate to a MAPI Profile for Use with Outlook](#), Microsoft Knowledge Base Article 312900, Microsoft, June 29, 2004.

[How To Automate Outlook Using Visual C++/MFC](#), Microsoft Knowledge Base Article 220600, Microsoft, June 29, 2004.

[How To Modify Recipients of Exchange Global Address List](#), Microsoft Knowledge Base Article No. 197191, Microsoft, July 1, 2004. (MAPI-based approach to modifying GAL entries.)

[How To Use Microsoft Outlook Object Model From Visual C++ Through an #IMPORT Statement](#), Microsoft Knowledge Base Article 259298, Microsoft, July 13, 2004.

[HOW TO: Programmatically Install SSL Certificates for Internet Information Server \(IIS\)](#), Microsoft Knowledge Base Article 313624, Microsoft, November 4, 2003.

[Common Access Card Setup](#), DLA On-Line Documentation. (Explains how to use Outlook to 'publish to GAL' your CAC credentials.)

[NSS Security Services](#), The Mozilla Organization, Jan. 13, 2004. (Detailed information on NSS up through release 3.9.)

[Using the Certificate Database Tool](#), The Mozilla Organization, Dec. 20, 2002. (Documents the command line syntax and usage of version 2.0 of the 'certutil' tool that is used to maintain Netscape 'cert7.db' and 'key3.db' database files. An older version of this )

[P12util](#), The Mozilla Organization, May 1, 2000. (Supposed to document the 'p12util' tool that is used to import/export key pairs as PKCS#12 files into/from the Netscape databases, but neglects to document, or even mention, most of the required command line arguments. More information is available on the [USPTO webpage](#). Although it discusses an extension to 'p12util', [HP's page on 'certmig'](#) contains the best usage guide I've been able to find - most of their sample 'certmig' commands work with 'p12util' without change and they actually describe the command line arguments! The failure of 'p12util' to import PKCS12 files without friendly names or DNs is discussed in a mozilla-crypto mailing list [entry](#).)

Henson, Stephen N., [Netscape Certificate Database Information](#) and [Netscape Communicator Key Database Format](#)

Luvisetto, M., [A Brief Guide to Certificate Management](#), INFN-Bologna, Feb. 20, 2004. (Browser-centric introduction, with some details on the use of openssl.)

## DIAGNOSTICS

Diagnostic messages are output to the console only for fatal errors. For details on other warnings and errors encountered during execution, the user should consult the active log file. (Use the '-v' switch to include verbose trace information in the diagnostic output.)

## BUGS AND LIMITATIONS

The only platforms that we anticipate supporting at this time are: Windows NT/2000/XP. The only supported applications are:

- SecretAgent 5.7.x
- Microsoft Outlook 2000, 2002, and 2003
- Microsoft Internet Explorer 5.0 through 6.0
- Netscape 4.75 and above, and
- Mozilla 1.1, 1.6, and above (Firefox can be supported by using the '-d' option to specify the root directory of its user profile tree.)

SecretAgent and the Microsoft products are all supported through the standard CAPI certificate stores and Microsoft-supplied Windows runtime libraries, while the Netscape and Mozilla products are supported through their respective proprietary certificate databases (cert7/8.db, key3/4.db) and Network Security Services runtime libraries (NSS 3.2.0, 3.6.1, and 3.9.0 using only the 'frozen' [NSS 3.4 public functions](#)). (Slightly modified versions of the Netscape 'certutil' and 'pk12util' tools are bundled into the cmu distribution for this purpose.)

## NOTES

### Import

Existing certificates in a Netscape database are never overwritten, so using cmu to import a PKCS#7 or PKCS#12 file will not update the friendly name of an existing certificate. Trust attributes of existing CA certificates in the database are never modified, but the trust attribute on a existing end-user certificate will be modified to 'u,pu,u' during import of a PKCS#12 file that contains it.

All new CA certificates found in PKCS#7 or PKCS#12 input file are imported into Netscape certificate databases using the trust attribute 'TC,C,C', which corresponds to manually setting the three trust checkboxes in the Netscape certificate management GUI. New end-user certificates imported from a PKCS#12 file are given a trust attribute of 'u,pu,u'.

The 3.9.0 version of the Netscape 'pl2util' tool will successfully import PKCS#12 files without friendly names, but it does so by generating very unfriendly substitute names (md5 hash values?). The 3.2.0 and 3.6.1 versions simply fail to import PKCS#12 files that do not already have a friendly name. In an attempt to avoid these problems, cmu "invents" a friendly name for each PKCS#12 input file that lacks one using a CAPI call of the form CertGetNameString(...,CERT\_NAME\_SIMPLE\_DISPLAY\_TYPE,...). When necessary, it creates a temporary file containing the invented friendly name and the key information from the original file, imports the temporary PKCS#12 file, and then deletes it.)

### Export

The certificate stores of all supported applications are searched for end-user certificate key pairs so that they may be exported as PKCS#12 files into the active UPC folder ('u:\private\certificates' if not overridden using the -c switch).

The search order is:

- the user's "MY" CAPI store (assumed to always exist)
- all subdirectories under 'u:\\_sys\Netscape'
- all subdirectories under 'u:\\_sys\Mozilla'
- all subdirectories under the optional -d command line argument, if supplied

We regard as end-user certificates all those certificates in a Netscape store with a trust attribute matching "\*",\*,\*u".

It is recommended that the user run the cmu export function ('cmu e') upon initial installation of the cmu software package, and whenever a new browser certificate has been obtained. If certificates are not obtained via a browser-based enrollment process, cmu's import (and possibly the synchronize) function may be all that is required to maintain the user's credentials after an initial discovery run.

### Reinitialize

The 'reinitialize database(s)' operation functions as follows:

1. 'u:\\_sys\Netscape' (or its D\_NETSCAPE replacement) is inspected for a 'cert[78].db' file
2. 'u:\\_sys\Mozilla' (or its D\_MOZILLA replacement) is inspected for a 'cert[78].db' file
3. all '-d' arguments are recursively searched for 'cert[78].db' files with 'default' appearing in their full pathnames
4. for the first match, or for all matches if '-a' is specified, cmu
  - moves into the UPC all existing '\*.db' files in the directory and renames them with a date and time prefix
  - recreates the database files in their original location with a new password provided via by the '-k' argument or, if '-k' is omitted, via an interactive prompt (with confirmation)

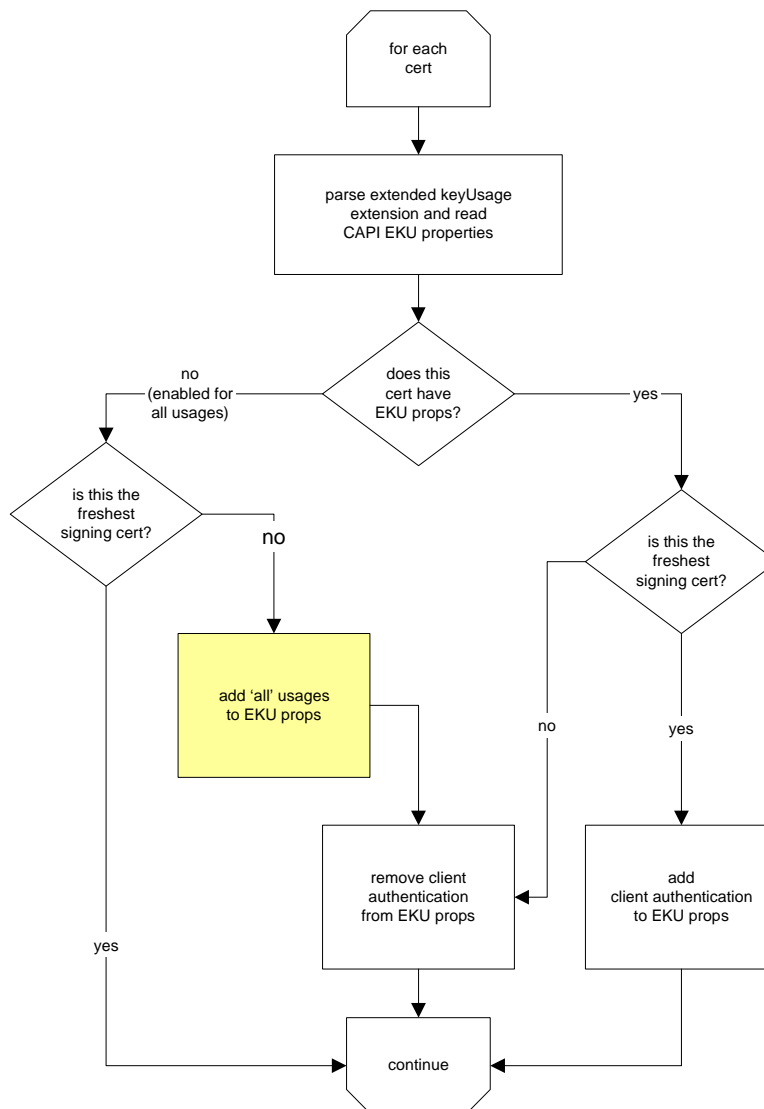
## Synchronize

All PKCS#12 files (files ending with '.p12' and '.pfx') found in the directory tree rooted at the active UPC folder ('u:\private\certificates', if not overridden using the -c switch) are imported into the certificate stores of all supported applications. (Application certificate stores are located using the same search order as is used for the Discover function described above.)

## Configure Client Authentication in CAPI

The 'c' function identifies the user's freshest signing certificate in CAPI and makes it the default certificate for client authentication. The freshest certificate is defined as the certificate having the greatest notBefore value. Once it is run, selection of the appropriate certificate will occur automatically each time the user visits a secure website requiring client authentication.

The function works by manipulating the enhanced key usage (EKU) properties associated with all certificates in the user's personal CAPI store as diagrammed in the following flowchart. Certificates containing an enhanced key usage extension (rather than just having an enhanced key usage property) which does not allow client authentication are not considered during the search process.



In order to disable usage for client authentication, the following EKU properties are added to all certificates, other than the freshest signing certificate, that initially have no EKU properties. (See yellow box in flowchart.)

```

"1.3.6.1.4.1.311.10.3.1", // CTL signing
"1.3.6.1.4.1.311.10.3.2", // time stamping
"1.3.6.1.4.1.311.10.3.4", // EFS crypto
"1.3.6.1.4.1.311.10.3.4.1", // EFS file recovery
"1.3.6.1.4.1.311.10.3.5", // WHQL driver verification
"1.3.6.1.4.1.311.10.3.6", // NT5 signed
"1.3.6.1.4.1.311.10.3.7", // WHQL OEM
"1.3.6.1.4.1.311.10.3.8", // embedded NT
"1.3.6.1.4.1.311.10.3.9", // signer of CTL containing trusted roots
"1.3.6.1.4.1.311.10.3.10", // signer of cross-cert and subordinate CA requests
"1.3.6.1.4.1.311.10.3.11", // encrypt/recover escrowed keys
"1.3.6.1.4.1.311.10.3.12", // document signer
"1.3.6.1.4.1.311.10.3.13",

"1.3.6.1.4.1.311.10.5.1", // music
"1.3.6.1.4.1.311.10.6.1", // licenses
"1.3.6.1.4.1.311.10.6.2", // license server
"1.3.6.1.4.1.311.10.12.1", // any application policy

"1.3.6.1.4.1.311.20.2.1", // enrollment agent
"1.3.6.1.4.1.311.20.2.2", // smartcard logon

"1.3.6.1.4.1.311.21.5",
"1.3.6.1.4.1.311.21.6",
"1.3.6.1.4.1.311.21.19",

"1.3.6.1.5.5.7.3.1", // TLS webservice authentication
"1.3.6.1.5.5.7.3.3", // code signing
"1.3.6.1.5.5.7.3.4", // e-mail protection
"1.3.6.1.5.5.7.3.5", // ipsec end system (deprecated by PKIX)
"1.3.6.1.5.5.7.3.6", // ipsec tunnel termination (deprecated by PKIX)
"1.3.6.1.5.5.7.3.7", // ipsec user (deprecated by PKIX)
"1.3.6.1.5.5.7.3.8", // timestamping
"1.3.6.1.5.5.7.3.9", // OCSP signing
"1.3.6.1.5.5.8.2.2", // PKIX iKEIntermediate (RFC2409)

```

Most of the Microsoft Cryptographic OIDs listed above are documented [here](#); the remaining OIDs are taken from [RFC 3280](#).

## MAPI Security Profile Update

If you specify the m function the cmu will find the freshest signing and/or encrypting certificates in the user's CAPI store (using the same algorithm as the c function described above). The freshest certificate is defined as the certificate having the greatest notBefore value. It will then iterate through the user's MAPI profiles looking for their default S/MIME settings. If a default S/MIME profile is found, cmu will modify it to reference these freshest certificates; no other attribute of this profile will be changed. If no MAPI security profile exists, cmu will create one with an Outlook display name of 'default' and associate those freshest certificate(s) as the user's default certificate(s) for S/MIME (and 'all other formats'). The following algorithm OIDs are encoded into the new profile's S/MIME capabilities BLOB:

```

(1 2 840 113549 3 7) // Triple DES EDE CBC PKCS#5 Padding
(1 2 840 113549 3 2) // RC2 CBC (128-bit)
0080 (128)
(1 2 840 113549 3 2) // RC2 CBC (64-bit)
40 (64)
(1 3 14 3 2 7)// id-desCBC
(1 2 840 113549 3 2) // RC2 CBC (40-bit)
28 (40)
(1 3 14 3 2 26) // id-sha1
(1 2 840 113549 2 5) // MD5

```

The new profile is marked so that this capabilities BLOB is sent with all encrypted and/or signed messages. Certificates containing an enhanced key usage extension (rather than just having an enhanced key usage property) which does not allow secure e-mail are not considered during the search process.

## Publish to GAL

The 'publish to GAL' operation ('cmu p [-p<password>] [-s] [p12file]') performs the following tasks in a manner nearly identical to that of Outlook itself:

1. First, it uses the supplied password (from the command line or typed in by the user in response to a runtime prompt) to extract the user's certificate and private key from the supplied PKCS#12 file (or the freshest PKCS#12 file in CAPI, if a PKCS#12 file is not specified on the command line).
2. Next it reads the current userCertificate attribute in the user's Global Address Book entry, if such an attribute is present. If the certificate provided in the PKCS#12 file is not already contained in the attribute value, it is appended and user's the GAL entry is updated.
3. Finally, if '-s' is NOT specified, it produces a signed (S/MIME) CMS PDU containing the user's certificate and S/MIME capabilities (namely that the user supports TDES), and stores it into GAL as the user's userSMIMEcertificate attribute (overwriting any existing attribute value).

For convenience (and to simplify the API), steps 2 and 3 use MAPI calls rather than communicating directly with the LDAP front-end on an Exchange Server hosting the GAL.

The ASN.1 dump of a sample CMS PDU that might be produced in step 3 is displayed below. The SEQUENCE specifying the user's SMIMECapabilities, highlighted in red, begins at offset 1345.

```
0 30 1671: SEQUENCE {
4 06 9: OBJECT IDENTIFIER signedData (1 2 840 113549 1 7 2)
15 A0 1656: [0] {
19 30 1652: SEQUENCE {
23 02 1: INTEGER 1
26 31 11: SET {
28 30 9: SEQUENCE {
30 06 5: OBJECT IDENTIFIER sha1 (1 3 14 3 2 26)
37 05 0: NULL
:
:
39 30 25: SEQUENCE {
41 06 9: OBJECT IDENTIFIER data (1 2 840 113549 1 7 1)
52 A0 12: [0] {
54 04 10: OCTET STRING 'Empty Body'
:
:
66 A0 1029: [0] {
70 30 1025: SEQUENCE {
74 30 745: SEQUENCE {
78 A0 3: [0] {
80 02 1: INTEGER 2
:
:
83 02 20: INTEGER
: 14 D8 EC 7F B0 61 E3 BB 78 0A 13 AD 04 50 95 4D
: 5F 55 61 14
105 30 13: SEQUENCE {
107 06 9: OBJECT IDENTIFIER
: sha1withRSAEncryption (1 2 840 113549 1 1 5)
118 05 0: NULL
:
:
120 30 102: SEQUENCE {
122 31 11: SET {
124 30 9: SEQUENCE {
126 06 3: OBJECT IDENTIFIER countryName (2 5 4 6)
131 13 2: PrintableString 'US'
:
:
135 31 11: SET {
137 30 9: SEQUENCE {
139 06 3: OBJECT IDENTIFIER stateOrProvinceName (2 5 4 8)
144 13 2: PrintableString 'IL'
:
:
148 31 35: SET {
150 30 33: SEQUENCE {
152 06 3: OBJECT IDENTIFIER organizationName (2 5 4 10)
157 13 26: PrintableString 'Information Security Corp.'
:
:
}
```



```

:
}
750 A3 71: [3] {
752 30 69: SEQUENCE {
754 30 12: SEQUENCE {
756 06 3: OBJECT IDENTIFIER basicConstraints (2 5 29 19)
761 01 1: BOOLEAN TRUE
764 04 2: OCTET STRING
: 30 00
: }
768 30 14: SEQUENCE {
770 06 3: OBJECT IDENTIFIER keyUsage (2 5 29 15)
775 01 1: BOOLEAN TRUE
778 04 4: OCTET STRING
: 03 02 00 E8
: }
784 30 37: SEQUENCE {
786 06 3: OBJECT IDENTIFIER subjectAltName (2 5 29 17)
791 04 30: OCTET STRING 'schulze-hewett@infoseccorp.com'
: }
: }
: }
: }
823 30 13: SEQUENCE {
825 06 9: OBJECT IDENTIFIER
: sha1withRSAEncryption (1 2 840 113549 1 1 5)
836 05 0: NULL
: }
838 03 257: BIT STRING 0 unused bits
: 52 F5 64 A0 B5 71 8C 93 16 DC D5 FA 38 21 9B 8E
: 14 01 E7 6F 8F D1 97 5A 76 03 A6 3F 76 B5 0D F5
: 0F 5B 6B FC 85 AA D0 EB AE 86 E2 3E 6D FC A4 0B
: 3C C8 47 A0 A4 35 B2 2E 83 A5 79 DA 09 16 67 AD
: B8 95 58 C5 40 04 9A B7 DC 67 9F 02 09 89 04 CC
: 95 FE B7 8D 4A 34 23 C9 31 0A 51 2E F0 FC 1A E5
: 1D 46 DA 7F D1 05 85 0B 7F 06 C4 3B 59 67 C9 26
: 69 32 3C 68 D7 F4 D5 1A E2 5A 99 8B C5 CA 84 B0
: [ Another 128 bytes skipped ]
: }
: }
1099 31 572: SET {
1103 30 568: SEQUENCE {
1107 02 1: INTEGER 1
1110 30 126: SEQUENCE {
1112 30 102: SEQUENCE {
1114 31 11: SET {
1116 30 9: SEQUENCE {
1118 06 3: OBJECT IDENTIFIER countryName (2 5 4 6)
1123 13 2: PrintableString 'US'
: }
: }
1127 31 11: SET {
1129 30 9: SEQUENCE {
1131 06 3: OBJECT IDENTIFIER stateOrProvinceName (2 5 4 8)
1136 13 2: PrintableString 'IL'
: }
: }
1140 31 35: SET {
1142 30 33: SEQUENCE {
1144 06 3: OBJECT IDENTIFIER organizationName (2 5 4 10)
1149 13 26: PrintableString 'Information Security Corp.'
: }
: }
1177 31 18: SET {
1179 30 16: SEQUENCE {
1181 06 3: OBJECT IDENTIFIER localityName (2 5 4 7)
1186 13 9: PrintableString 'Deerfield'
: }
: }
1197 31 17: SET {
1199 30 15: SEQUENCE {
1201 06 3: OBJECT IDENTIFIER commonName (2 5 4 3)
1206 13 8: PrintableString 'ISC Root'
: }
: }
1216 02 20: INTEGER
: 14 D8 EC 7F B0 61 E3 BB 78 0A 13 AD 04 50 95 4D
: 5F 55 61 14
: }

```

