Using RLM with Visual Basic 6

Visual Basic 6 programs may call RLM functions to implement software licensing functionality. There are a few extra steps that must be taken beyond what's needed in C/C++ programs to make it work however. This is a description of those steps.

Step 1: Download and build the RLM SDK

This process is described in the RLM Getting Started Guide available here:
http://www.reprisesoftware.com/kits/RLM_Getting_Started_Guide.pdf
One of the products of the SDK build is rlm.dll, a dll containing the RLM functions in executable form. It is these functions that will be called from VB6.

Step 2: Build an interface DLL

Because Visual Basic and C use different conventions for calling functions, and because C can be made to use Basic's conventions but the reverse is not true, you must build a simple interface layer in C which accepts function calls in Basic using Basic's conventions and calls C using C's conventions. This layer is very simple. For each RLM function call to be made from Basic, write a C wrapper function. Each function should:

- Have a name <rlm-function>_stdcall and invoke the corresponding <rlm-function>
- Declare arguments identical to the corresponding RLM function
- Include "__stdcall" between the function's return type and t he function name (stdcall is what tells the compiler to expect to be called with Basic calling conventions)

Here is an example rlmstdcall.c, using 6 fundamental RLM functions:

```
#include "license.h"

RLM_HANDLE
__stdcall rlm_init_stdcall(const char *licdir, const char *argv0, const char *lic){
        return(rlm_init(licdir, argv0, lic));
}
int __stdcall rlm_stat_stdcall(RLM_HANDLE rh) {
       return(rlm_stat(rh));
}
RLM_LICENSE
__stdcall rlm_checkout_stdcall(RLM_HANDLE rh, char *prod, char *ver, int count){
        return(rlm_checkout(rh, prod, ver, count));
}
int __stdcall rlm_checkin_stdcall(RLM_LICENSE license) {
       return rlm_checkin(license);
}
int __stdcall rlm_license_stat_stdcall(RLM_LICENSE license) {
        return rlm_license_stat(license);
}
int __stdcall rlm_close_stdcall(RLM_HANDLE handle) {
        return rlm_close(handle);
 }
```

An exports definition file must also be created, naming each wrapper function. Here is an example rlmstdcall.def, exporting the 6 wrapper functions from the example above:

```
EXPORTS

 rlm_init_stdcall
 rlm_stat_stdcall
 rlm_checkout_stdcall
 rlm_checkin_stdcall
 rlm_license_stat_stdcall
 rlm_close_stdcall
```

With the C source file rlmstdcall.c and the exports file rlmstdcall.def in place, the dll rlmstdcall.dll is built as follows (this assumes that the c file and the def file are in the x86_w1 directory on the RLM SDK):

```
$ cl /c /MD /I..\src /Forlmstdcall.obj rlmstdcall.c
$   link   /nologo   /dll   /out:rlmstdcall.dll   /def:rlmstdcall.def
rlmstdcall.obj rlm.lib
```

Step 3: Create Basic function declarations for the RLM functions to be called

For each RLM function to be called from Basic, write a Basic function declaration for it. The declaration specifies the data type of each argument and how to pass it, and the function's return type. Arguments should be passed by value, using the "ByVal" specifier. C and Basic types used by RLM are as follows:

C type	Basic Type
int	long
char *	String
Any handle (RLM_HANDLE, RLM_LICENSE, etc)	long

Here is an example of the Basic function declarations of the 6 RLM functions used in previous examples:

```
Private Declare Function rlm_init_stdcall Lib
"c:\temp\7.0BL4\x86_w1\rlmstdcall.dll" (ByVal path As String, ByVal appPath As
String, ByVal license As String) As Long

Private Declare Function rlm_stat_stdcall Lib
"c:\temp\7.0BL4\x86_w1\rlmstdcall.dll" (ByVal handle As Long) As Long

Private Declare Function rlm_checkout_stdcall Lib
"c:\temp\7.0BL4\x86_w1\rlmstdcall.dll" (ByVal handle As Long, ByVal product As
String, ByVal version As String, ByVal count As Long) As Long

Private Declare Function rlm_checkin_stdcall Lib
"c:\temp\7.0BL4\x86_w1\rlmstdcall.dll" (ByVal license As Long) As Long

Private Declare Function rlm_license_stat_stdcall Lib
"c:\temp\7.0BL4\x86_w1\rlmstdcall.dll" (ByVal license As Long) As Long

Private Declare Function rlm_closet_stdcall Lib
"c:\temp\7.0BL4\x86_w1\rlmstdcall.dll" (ByVal handle As Long) As Long
```

Step 4: Add RLM function calls to your Basic program

Here is a small example Basic program using some of the functions from the previous examples:

```
Dim path As String
Dim nullstring As String
Dim handle As Long
Dim license As Long
Dim product As String
Dim ver As String
Dim stat As Integer

path = "5053@localhost"
nullstring = ""
product = "test1"
ver = "1.0"

handle = rlm_init_stdcall(path, nullstring, nullstring)
stat = rlm_stat_stdcall(handle)
If stat = 0 Then
       license = rlm_checkout_stdcall(handle, product, ver, 1)
       rlm_checkin_stdcall(license)
       rlm_close_stdcall(handle)
End If
```