In **part 1** the first layer of SRP protection was introduced.
It is related to **'Designated File Types'** list (DFT) and **ShellExecute()** API function. If the file extension is on DFT list, then ShellExecute() can prevent that file from opening.
But, if you choose to open HTA file directly, using the command:
'mshta.exe %Userprofile%\Desktop\How2BeReach.hta'
then ShellExecute() will be skipped.

Now, Windows has two possibilities:
1. Allow opening the file.
2. Use another mechanism to call into SRP.

In most cases, Windows will follow the 'point 1'. So, users cannot open the files from DFT list by clicking the file icon on the Desktop, pressing Enter, choosing 'Open' or 'Open with ...' from Explorer context menu. This prevents fooling users to open double extension malware files like IamInnocentFile.pdf.exe (with PDF icon). The user can still try to open the file using the application that supports opening it (sponsor). In the case of the above malware, the PDF viewer will run (with file opening error), and IamInnocent-File.pdf.exe will not be opened.
But still, the command 'mshta.exe %Userprofile%\Desktop\How2BeReach.hta' will be executed to infect the computer.

In some special cases, Windows will use the 'point 2' solution, anyway. There are some **Privileged Objects**: **'CMD Host', 'Windows Script Host', and 'Windows Installer'**. They are prepared to call into SRP about supported files. So, in the case of the 'How2Be-Reach.vbs' file, it can be blocked, because 'Windows Script Host' can call into SRP about VBS files (even when VBS is not on DFT list).

It is worth mentioning, that above privileged objects can control file execution independently of ShellExecute() and DFT list. File blocking by 'Script Hosts' or 'Windows Installer' is also more comprehensive as compared to ShellExecute(). For example, the How2BeReach.vbs can be run by:
**(A)** double click from the Desktop
**(B)** executing the command: 'wscript.exe %Userprofile%\How2BeReach.vbs'.
If we add VBS extension to DFT list, then:
ShellExecute() can block the (A) but cannot block (B).
Privileged Objects can block both (A) and (B).
Finally, Privileged Objects can block supported files even when those files have changed extensions.

One can ask a question: what is the (B) protection for? Users, mostly run files by clicking, not by executing commands. That is true, but the Office document macro or autorun.inf on DVD disk, can execute commands. The same is true when the system or appli-

cation is exploited, etc.

From the above considerations, we can conclude, that scripts: BAT, CMD, JS, JSE, VBS, VBS, WSF, WSH, and MSI files can be blocked in two independent ways:
* **by extension (DFT list, ShellExecute());**
* **by Privileged Objects ('CMD Host', 'Windows Script Host', and 'Windows Installer').**

The same applies to COM and SCR files that use ShellExecute() to find out, that they are just a special kind of EXE files.
The native Windows executables EXE files are always executed without invoking Shell Execute(). Instead of this, the API function **CreateProcess()** is used, and it can also call into SRP to block COM, EXE (and SCR) files.

There's another API function, that can call into SRP. The **LoadLibrary()** is invoked when EXE file is executed, and tries to load binary libraries (DLL and OCX files). SRP can be configured, either to allow or block loading libraries from specified locations. This layer of protection can stop many DLL attacks, when system files are used to load malicious libraries from the User Space. For example (paths omittet for simplicity):
InstallUtil.exe /logfile= /LogToConsole=false /U malicious.dll
regsvcs.exe malicious.dll
regasm.exe /U malicious.dll
regsvr32 /s /u malicious.dll
regsvr32 /s malicious.dll
rundll32 malicious.dll,EntryPoint

**SRP have three essential enforcement settings:**
* No Enforcement - ignores : DFT list, ShellExecute(), CreateProcess(), LoadLibrary() calls. SRP will only answer the calls from 'CMD Host', 'Windows Script Host' and 'Windows Installer'.
* Skip DLLs - ignores only LoadLibrary() calls - other security layers: DFT list, ShellExecute(), CreateProcess(), CMD Host, Windows Script Host, and Windows Installer calls are not ignored.
* All files - activates all SRP security layers.

**The protection applied by: CreateProcess(), LoadLibrary(), CMD Host, Windows Script Host, and Windows Installer, can work even when files have changed extensions.**

We see that 'All files' enforcement setting is the most powerful, but one should remember, that programs which use binary libraries located in the User Space, will not run properly. So, all those libraries have to be whitelisted.

In Windows Pro, there are also two other options in enforcement window:

- ★ 'Apply software restriction policies to the following users:' with two settings: (1) All users, and (2) All users except local administrators. The first applies SRP to all users, including local administrators, so SRP can control processes which try to run with Administrative Rights. The second applies SRP to all users, except local administrators, so elevated processes can bypass SRP.
- ★ ‘When applying software restriction policies:’ with two settings: (1) Enforce certificate rules, and (2) Ignore certificate rules.

In Windows Home (Vista and later versions), the above options are usually set to :
‘All users except local administrators’ + ‘Ignore certificate rules’. Those are the default settings in programs Simple Software Restriction Policies and Hard_Configurator.

**End of part 2.**