УДК 004.45:004.49

Мельник Н.Д., Паршуков С.С.

ВИКОРИСТАННЯ ТЕХНОЛОГІЙ WMI ДЛЯ ЗАБЕЗПЕЧЕННЯ ЦІЛІСНОСТІ ТА КОНФІДЕНЦІЙНОСТІ СЦЕНАРІЇВ В ЗАДАЧАХ АВТОМАТИЗАЦІЇ ПРОЦЕСІВ УПРАВЛІННЯ ОПЕРАЦІЙНИМИ СИСТЕМАМИ СІМЕЙСТВА *WINDOWS*

Анотація:

У статі мова йтиме про забезпечення цілісності та конфіденційності сценаріїв у задачах автоматизації процесів управління Windows- подібними ОС. Конфіденційність та цілісність забезпечується системою ЕЦП та центром сертифікації.

Аннотация:

В статье речь пойдет об обеспечении целостности и конфиденциальности сценариев в задачах автоматизации процессов управления Windows- подобными ОС. Конфиденциальность и целостность обеспечивается системой ЭЦП и центром сертификации.

Abstract:

In this sex will discuss about the integrity and confidentiality of scenarios in the problems of automation of management processes Windows - like OS. Confidentiality and integrity is provided by EDS system and certification center.

Вступ

Із зростанням кількості задач, що підлягають автоматизації зростає потреба у збереженні цілісності та конфіденційності даних автоматизованих систем, тим більше, якщо мова йде про застосування на практиці програмних сценаріїв типу Windows Script Host. Windows Script Host – являє собою мову сценаріїв, свого роду певний програмний код, який виконується, компілюється відразу в ОС сімейства Windows. WSH разом з WMI (Windows Management Instrumentation) створюють потужний інструмент для автоматизації процесів управління ОС сімейства Windows.

Застосування ЕЦП над сценаріями в задачах автоматизації процесів управління ОС сімейства *Windows*

Система ЕЦП створює можливість встановлення походження сценарію а також цифровий підпис захищає сценарій від підробки (редагування).

Більше того, ***WSH*** може виконувати сценарії з цифровим підписом, блокуючи при цьому виконання сценаріїв з недійсним цифровим підписом.

Система цифрового підпису обов'язково має включати: авторизований центр сертифікації ключів; систему керування ключами (шаблони ключів, сертифікати); ключі, що будуть використовуватися при виконанні відповідних задач.

Для контролю цілісності виконуваних сценаріїв потрібно виконати низку дій: отримати відповідний сертифікат, написати програмний сценарій, який буде накладати цифровий підпис на потрібні виконувані файли. Цілком ймовірно, що у центрі видачі сертифікатів не знайдеться потрібного шаблону сертифіката для цифрового підпису. Тому слід самостійно установити шаблон сертифіката для цифрового підпису, завантаживши в

консоль *MMC* оснастку *Certificate Authority,* і обрати опцію *Manage* контейнера *Certificate Template* так, як це зображено на рис. 1.

Обравши опцію Manage, отримаємо весь перелік доступних шаблонів для нашого центру сертифікації. Щоб не створювати шаблон заново, створимо дублікат для вже існуючого шаблону і надамо йому можливості цифрового підпису – *Code Signing*. Щоб створити дублікат для вже існуючого шаблону, достатньо натиснути правою кнопкою мишки на один із шаблонів і обрати опцію *Dublicate*, так, як це показано на рис. 2.

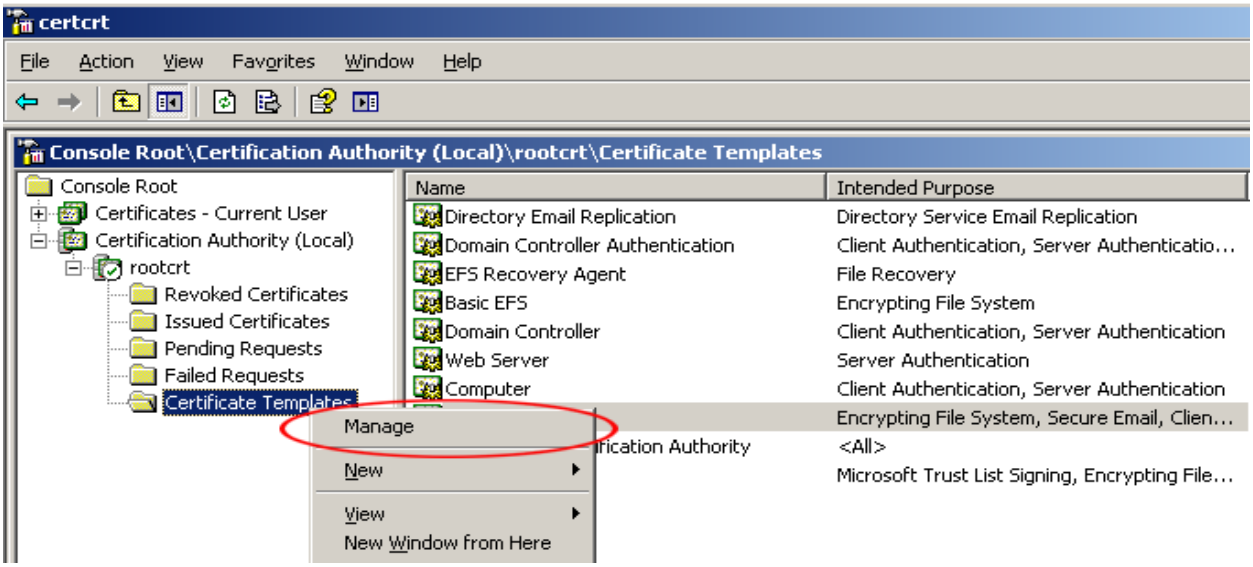


Рис. 1. Створення нового шаблону сертифікатів

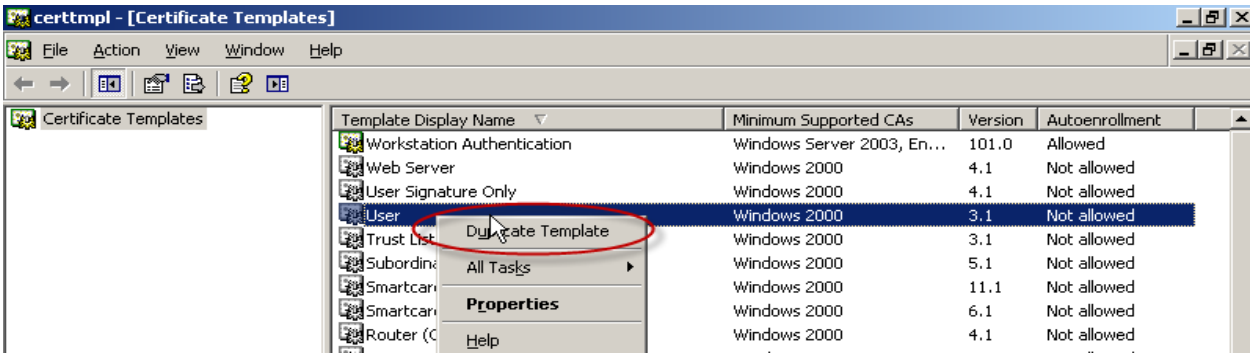


Рис. 2. Створення дублікату шаблону

Далі переходимо до діалогових вікон редагування нового шаблону. Спочатку назвемо його зрозумілою та відмінною від інших назвою – *code_signing_template.* Далі необхідно перейти на вкладку *Extensions*, де внесемо зміни до одного з доступних розширень – *Application Policies* (Політика застосування), добавивши нову політику застосування – Code Signing, так , як це показано на рис. 3 та рис. 4.

Таким чином, ми отримали новий шаблон сертифікатів, який включатиме можливість цифрового підпису. Тепер необхідно отримати сертифікат у центрі сертифікації згідно з даним шаблоном. Для цього зайдемо через веб-браузер на центр сертифікації, авторизувавшись там з правами адміністратора.

Дотримуючись зрозумілого діалогового набору сторінок, здійснюємо запит на видачу сертифіката. За замовчуванням для користувача буде доступний користувацький

сертифікат *User Certificate*, тому необхідно перейти на розширений доступ до видачі сертифікатів, так, як це показано на рис. 5 та рис. 6.

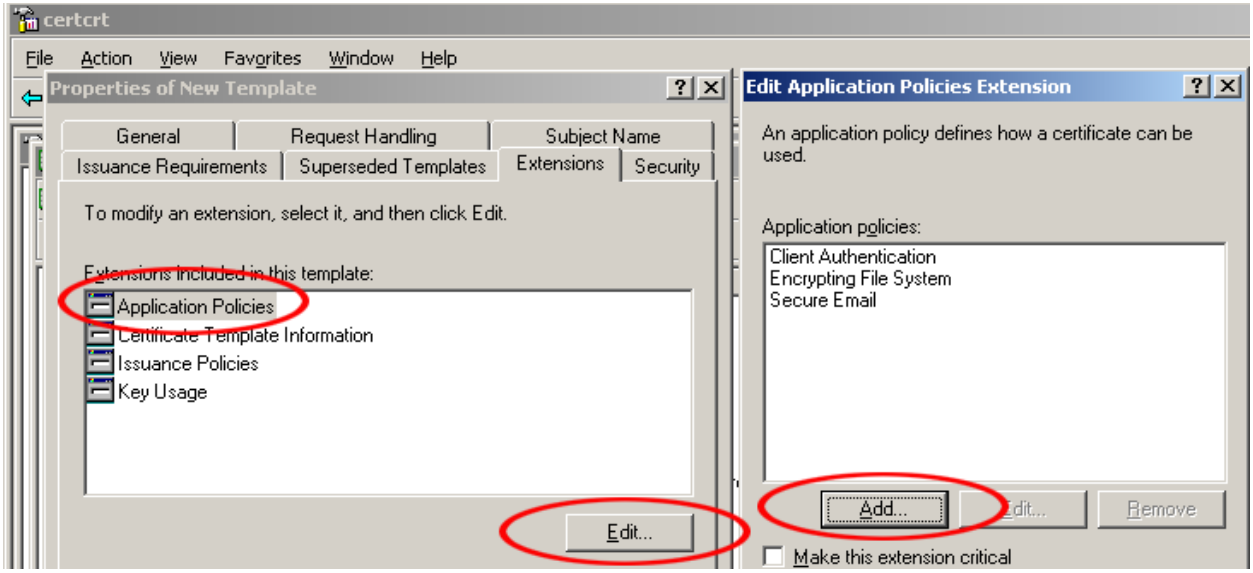


Рис. 3. Редагування політики застосувань для відповідного шаблону

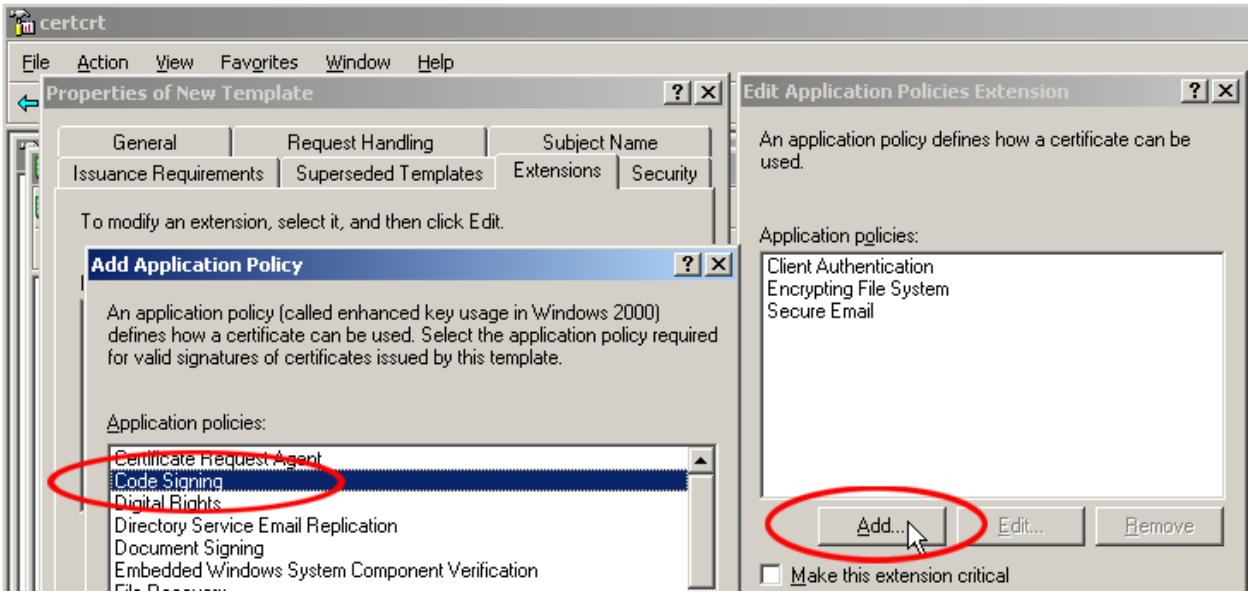


Рис. 4. Додаємо нову політику застосування для даного шаблону

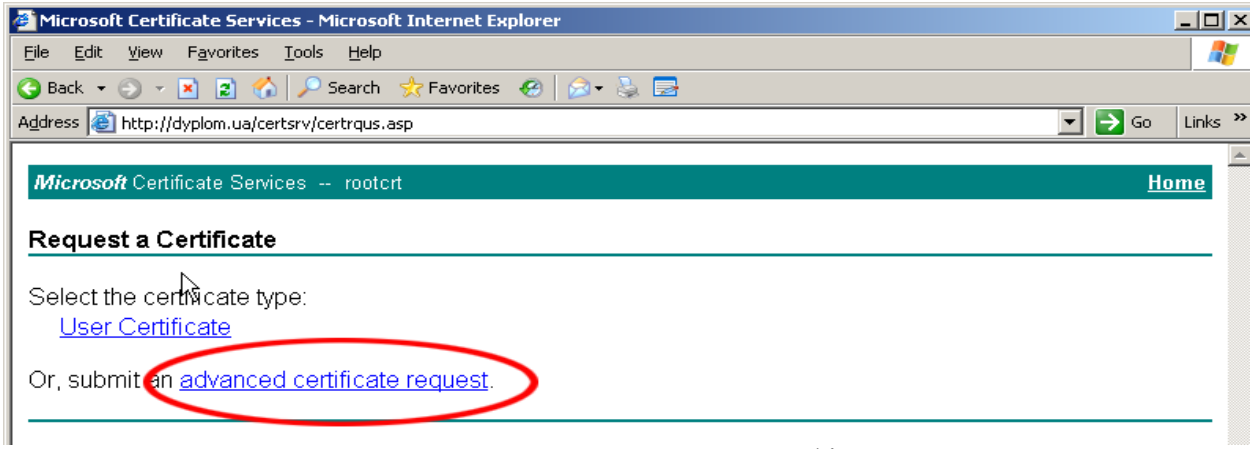


Рис. 5. Розширений запит сертифіката

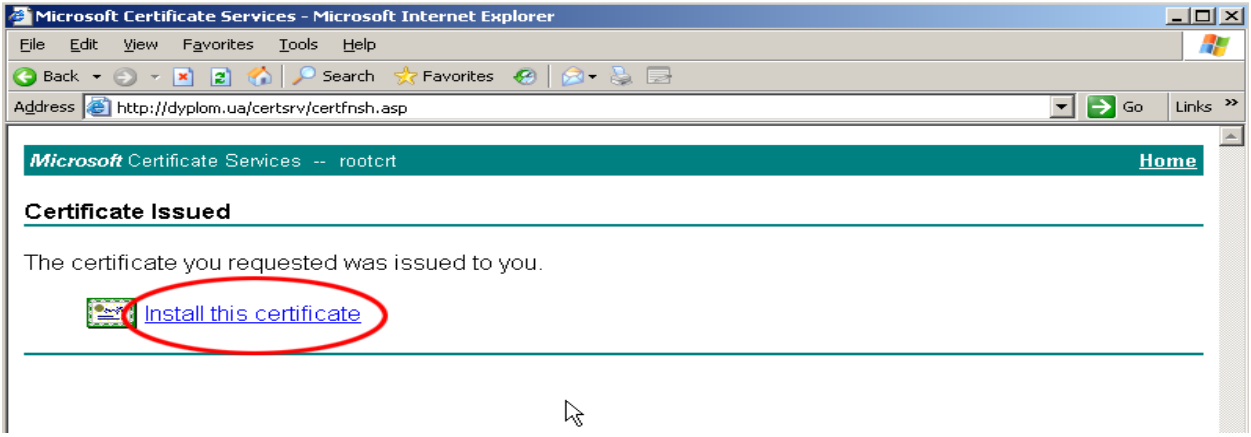


Рис. 6. Вікно створення запиту і отримання сертифіката

Далі потрібно вказати шаблон сертифіката, призначити йому ім'я, для прикладу, ***my_code_signing,*** і здійснити установлення. Після цього новий сертифікат з можливістю цифрового підпису буде доступним для перегляду даному користувачу у його сховищі сертифікатів у контейнері – ***Personal -> Certificates***, так, як показано це на рис. 7.

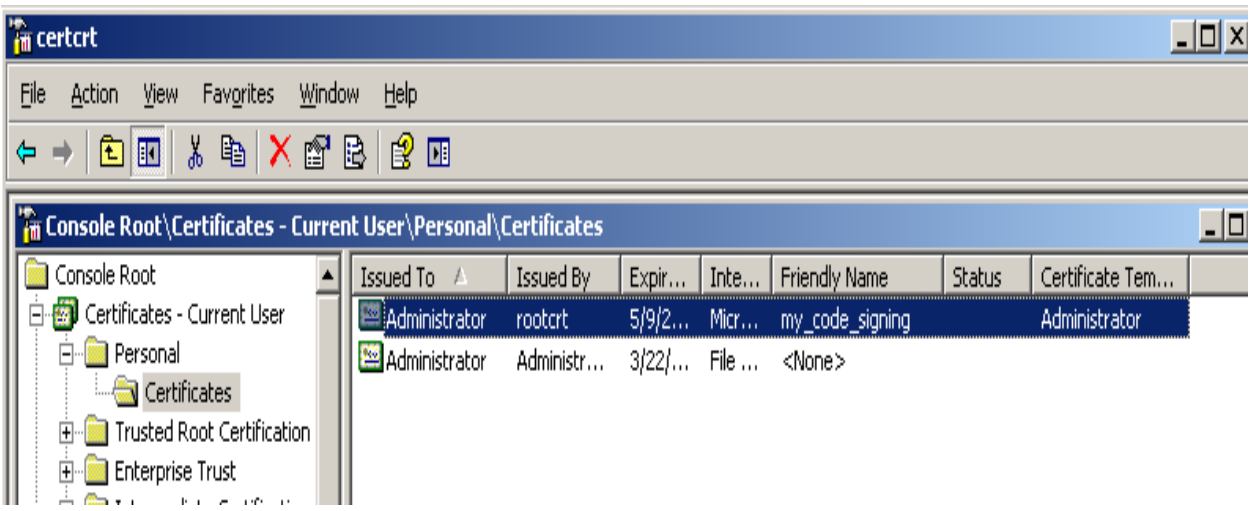


Рис. 7. Сховище виданих сертифікатів

Таким чином, ми отримали новий сертифікат, одною з політик призначення якого є створення цифрового підпису для програмного продукту. Тепер потрібно здійснити сам цифровий підпис. Для цього ми можемо написати програний сценарій, який виконуватиме таку задачу, або ж скористатися вбудованою можливістю для здійснення цифрового підпису в програмному середовищі *Primal Script*. Розглянемо обидва варіанти. Програмний сценарій створення цифрового підпису наведено нижче.

1. *Option Explicit*
2. *Dim FSO, Path, Filename, objSigner*
3. *Set FSO = CreateObject("Scripting.FileSystemObject")*
4. *Path = FSO.GetAbsolutePathName(".\programa1.vbs")*
5. *Set objSigner = WScript.CreateObject("Scripting.Signer")*
6. *objSigner.SignFile Path,"MyCodeSigning"*

7. Wscript.Echo "Вітаємо. Файл – C:\myscript\programa1.vbs успішно підписаний цифровим підписом"

Алгоритм роботи програмного сценарію зводиться до об'явлення відповідних перемінних (рядки 1 та 2), отримання абсолютного шляху до файла з програмним сценарієм, який підлягає підпису (рядок 3 та 4), створення нового об'єкта з можливістю підпису (рядок 5), здійснення самого процесу підпису, вказавши шлях до файла та ім'я сертифіката, що використовується для підпису (рядок 6).

Для того, щоб отримати змогу здійснювати цифровий підпис в програмному середовищі *Primal Script,* необхідно здійснити певні налаштування: так в оснастці *Tools -> Options -> Script-Security* слід вказати шлях до попередньо експортованого сертифіката. Докладний приклад можна побачити на рис. 8.

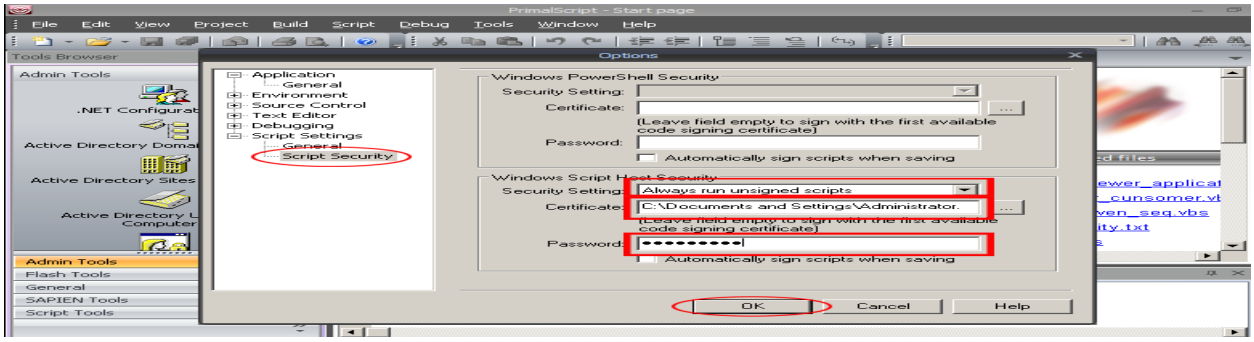


Рис. 8. Вказуємо шлях до вже експортованого сертифіката

Для експортування сертифіката слід скористатися командою діалогового меню – *Export*, потім вказати шлях експортування та пароль на цей сертифікат. Приклад показано на рис. 9.

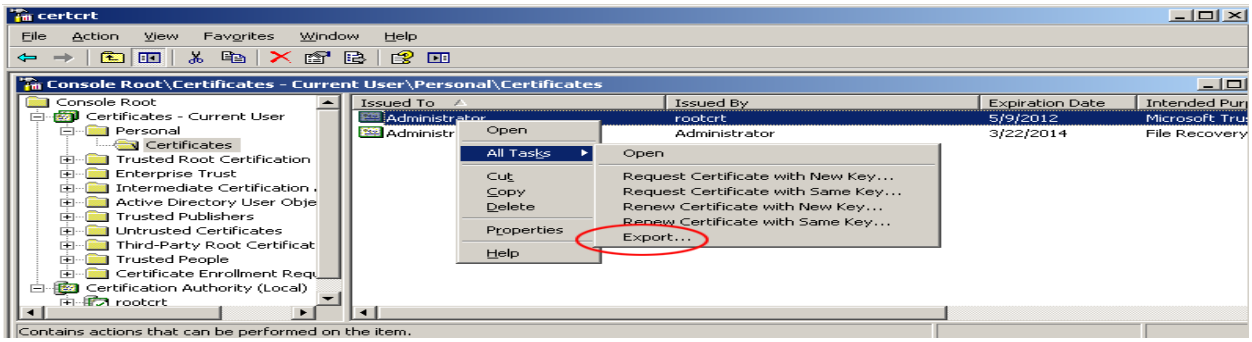


Рис. 9. Експортування сертифіката

Процес підписування програмного коду здійснюється шляхом використання вбудованої команди програмного середовища *Primal Script: Script -> Sign Script*. Приклад показано на рис. 10.

Ознакою успішного проведення цифрового підпису є наявність відповідного набору символів разом із текстом програмного коду. Приклад програмного коду, захищеного цифровим підписом, наведено нижче:

1. *wscript.echo "your code is already sign"*
2. *" SIG " Begin signature block*

3. " SIG " MIIIGgYJKoZIhvcNAQcCoIIICzCCCAcCAQExCzAJBgUr
4. " SIG "
DgMCGgUAMGcGCisGAQQBgjcCAQSgWTBXMDIGCisGAQQB

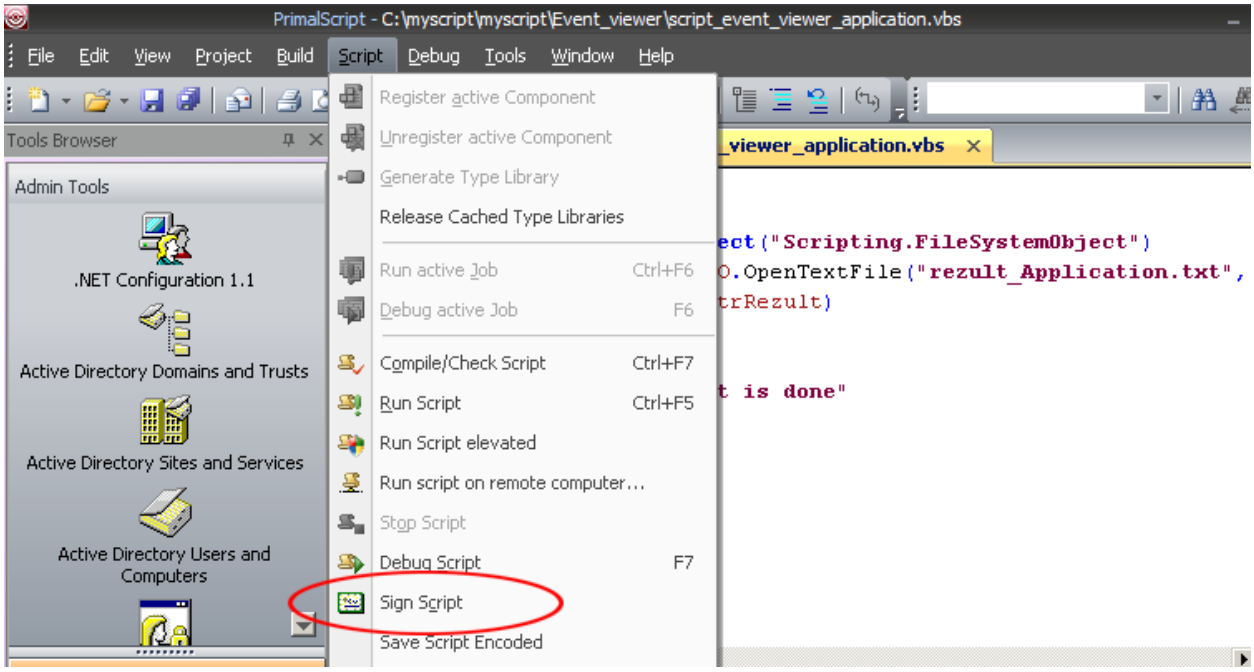


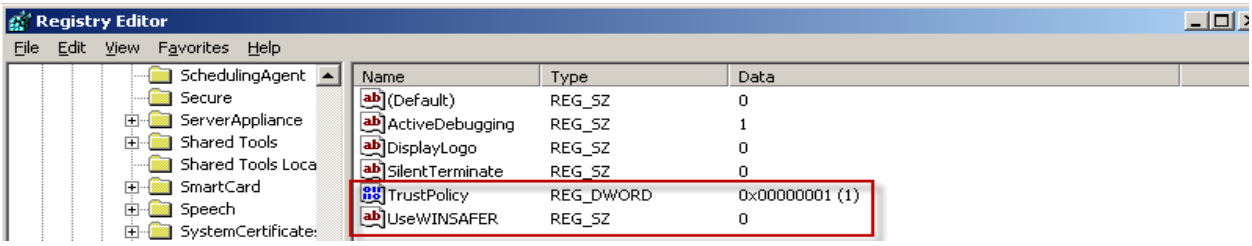
Рис. 10. Підписування програмного сценарію

5. " SIG " gjcCAR4wJAIBAQQQTvApFpkntU2P5azhDxfrqwIBAAIB
6. " SIG "
AAIBAAIBAAIBADAhMAkGBSsOAwIaBQAEFEeG5eZmpGlE
7. " SIG "
5aBa4TMav/JIUgBNoIIGIjCCBh4wggUGoAMCAQICCmEN
8. " SIG " tfP+NsYxcDJuY5XkWSlk8LYw0E7IWGQLmo5PwFVxChM=
9. " SIG " End signature block

Перший рядок програмного коду відповідає за вивід інформаційного повідомлення, про те, що код успішно підписано цифровим підписом, а наступні рядки 2…9 якраз і являють собою той спеціальний код, який перевірятиметься при запуску сценарію.

Завдяки проведенню низки послідовних дій було успішно виконано задачу з накладання цифрового підпису на виконувані сценарії. А тепер слід навчити ОС (для прикладу візьмемо Windows Server 2003) перевіряти сценарії на наявність цифрового підпису безпосередньо перед запуском відповідного сценарію. Хотілося б зауважити, що незалежно від того, ким буде запущено сценарій на виконання: користувачем подій, груповою політикою, безпосередньо користувачем, – він обов'язково перевірятиметься на відповідність цифровому підпису. Отже, для того, щоб сценарій перевірявся перед запуском, слід встановити відповідні параметри в реєстрі ОС, а саме: перейти на вкладку *My_Computer\ HKEY_LOCAL_MACHINE\ SOFTWARE\ Microsoft\Windows Script*

Host\Setting і встановити для змінних *UseWINSAFER* та *TrustPolicy* значення 0 та 1 так, як це зображено на рис. 11.



Таким чином, при запуску сценарію, який не має цифрового підпису, користувач сповіщатиметься інформаційним повідомленням, наведеним на рис. 12.

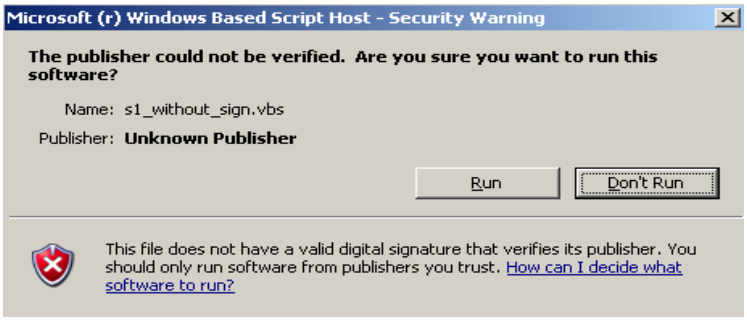


Рис. 12. Інформаційне вікно, яке сповіщає про небезпеку запуску неперевіреного сценарію щодо відповідності чи наявності цифрового підпису

Користувач самостійно приймає рішення щодо запуску чи відміни виконання цього сценарію. Таким чином, успішно виконано задачу з накладання ЕЦП на виконувані сценарії і перевірки відповідності цих ЕЦП ОС перед запуском самого сценарію.

Реалізація кодування програмного коду

Для створення скритності виконуваних сценаріїв, своєрідного маскування, що захистить програмний код від несанкціонованого ознайомлення, скористаємося вбудованою в технологію *WMI* можливістю так званого кодування символів програмного коду. Приклад та пояснення алгоритму наведено нижче:

```
1.  Dim FSO, Path, objFile, strDate, scriptname, codescriptname,
strCoded, MyEncoder, rez
2.  Set FSO = CreateObject("Scripting.FileSystemObject")
3.  Path = FSO.GetAbsolutePathName(".\v1.vbs")
'################################## Читання вмісту файла
4.  scriptname="v1.vbs"
5.  Set objFile = FSO.OpenTextFile(scriptname, 1, True)
6.  strDate = objFile.ReadAll()
7.  objFile.Close()
'################################## Зміна назви
8.  Set MyEncoder = WScript.CreateObject("Scripting.Encoder")
9.  codescriptname = Replace (scriptname, ".vbs", ".vbe")
10. strCoded = MyEncoder.EncodeScriptFile (".VBS", strDate, 0, "")
'##################################Створення нового файла
11. Set objFile = FSO.OpenTextFile(codescriptname, 2, True)
```

12. objFile.Write (strCoded)
13. objFile.Close()

У першому рядку програмного сценарію проводиться об'явлення змінних, які використовуватимуться надалі. Потім потрібно отримати абсолютний шлях до файла, в якому знаходиться сценарій, вміст якого підлягає кодуванню – рядок 2 та 3. Після цього зчитуємо вміст даного файла і його ім'я у відповідні змінні – рядки 4, 5, 6, 7. Попри зміну самого вмісту файла, змінюється його назва, точніше розширення. Стандартне розширення сценарію має вигляд **.vbs*, а закодованого **.vbe,* тому у рядку 9 і проводиться зміна його імені, а у рядку 10 – кодування вмісту файла. Далі створюємо новий, закодований, файл (рядки 11, 12, 13). Приклад вихідного (відкритого коду) сценарію і закодованого подано на рис. 13 та рис. 14.

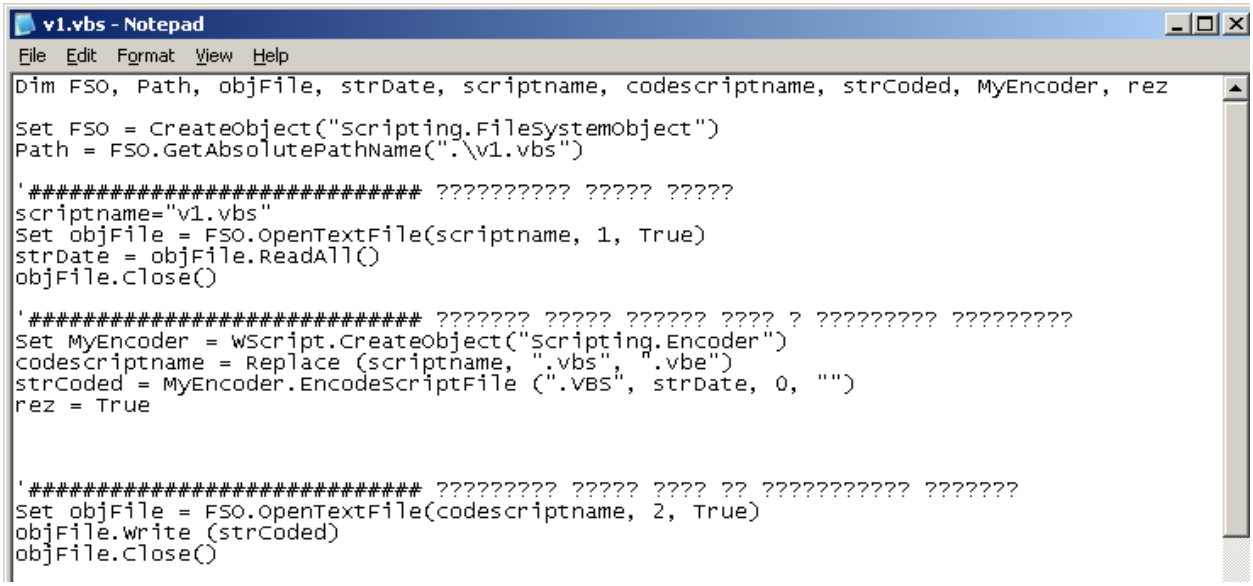


Рис. 13. Програмний код відкритого для ознайомлення сценарію

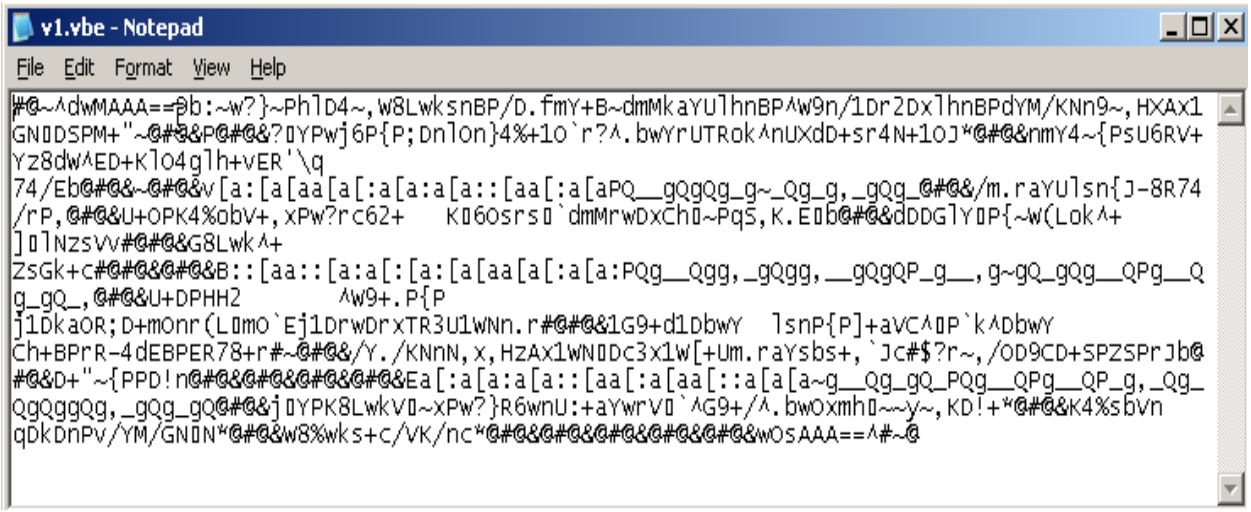


Рис. 14. Програмний код недоступного, закодованого, для перегляду сценарію

При здійсненні кодування міняється також і ім’я, а точніше розширення виконуваного файла з *.vbs на *.vbe. Розкодування здійснюється автоматично при виконанні сценарію в ОС Windows.

Висновки

Розглянувши вбудовані технології WMI, слід зазначити, що є можливість вирішення питання цілісності та конфіденційності шляхом написання виконуваних сценаріїв в ОС сімейства Windows. Конфіденційність сценаріїв досягається шляхом накладання цифрового підпису. Ключі для реалізації системи ЕЦП отримуємо в центрі видачі сертифікатів центра сертифікації Windows. Тобто можемо сказати про високий рівень реалізації ЕЦП. Щодо самого маскування вихідного коду, тут є великий недолік. Маскування (кодування) проводиться вбудованими механізмами ОС, які легко обійти, використавши відповідне програмне забезпечення.

Література:

1. *Шетка П.* Microsoft Windows Server 2003. Практическое руководство по настройке сети. — СПб.: Наука и Техника, 2006. — 608 с.

2. *Попов А.В.* «Командные файлы и сценарии Windows script host». – Спб.: БХВ-Петербург, 2002 — 320с.

3. *www.wikipedia.org* [Електронний ресурс].