Uwierzytelnianie sesji remoting – info

Jak uwierzytelnić użytkownika na systemie zdalnym

Pobieranie poświadczeń użytkownika - Get-Credentials

Zagrożenia podczas korzystania z remotingu

Co to jest mutual authentication

Przesyłanie poświadczeń do systemów zdalnych

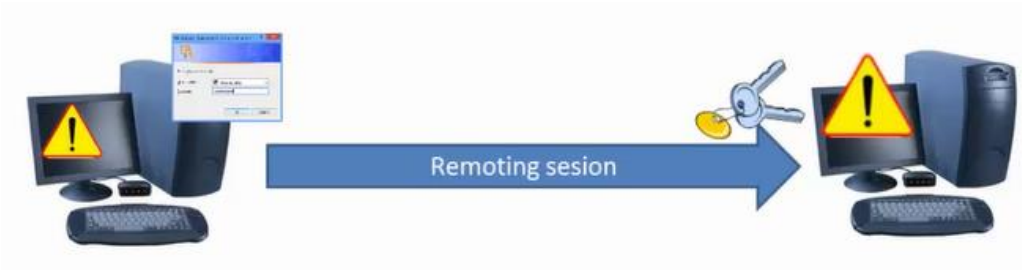
TrustedHosts

Jak to się dzieje, że komputer zdalny pozwala na uruchomienie przysłanych poleceń

Serwer, który obsługuje polecenie musi sprawdzić czy użytkownik, który wysłał polecenie jest do tego uprawniony

Za pobranie informacji o użytkowniku i haśle odpowiada strona klienta, tzw user credentials które będziemy nazywali poświadczeniami mogą być wygenerowane w oparciu o nazwę aktualnie zalogowanego użytkownika, co ułatwia domena Windows.

Może być tak że użytkownik do pracy na co dzień ma nisko uprzywilejowane konto a w momencie, gdy chce wykonywać prace administracyjną używa specjalnego dodatkowego konta, w takim przypadku można po stronie klienta pobrać informacje o użytkowniku i haśle wygenerować w oparciu o nie nowe poświadczenia które zostaną przesłane do serwera zdalnego. Serwer zdalny otrzymując poświadczenia z maszyny lokalnej podejmuje decyzje czy pozwolić albo odmówić wykonania przesłanego polecenia.



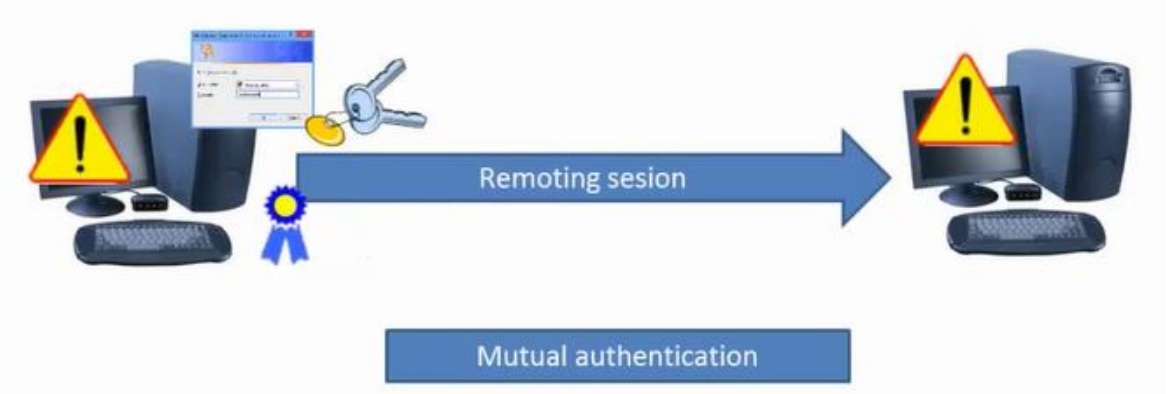
Czy to wystarczy do zapewnienia bezpieczeństwa połączenia? Czy komputer lokalny nie jest narażony na atak?

Czy ktoś jakoś nie może przechwycić poświadczeń wprowadzanych przez użytkownika?

Potencjalny haker mógłby wstawić swój własny serwer podszywający się pod prawdziwy serwer zdalny, wyślemy mu poprawne poświadczenia i to dotyczące mocnego konta administratora, każdy wolałby uniknąć takiej sytuacji, dlatego klient musi sprawdzić czy komputer, do którego będą przesyłane poświadczenia jest serwerem, do którego chcemy się podłączyć. Uwierzytelnienie komputerów w AD nie jest problemem, tam każda stacja posiada swoje konto komputera, mamy więc środki, aby uwierzytelnić serwer i uniknąć ataku

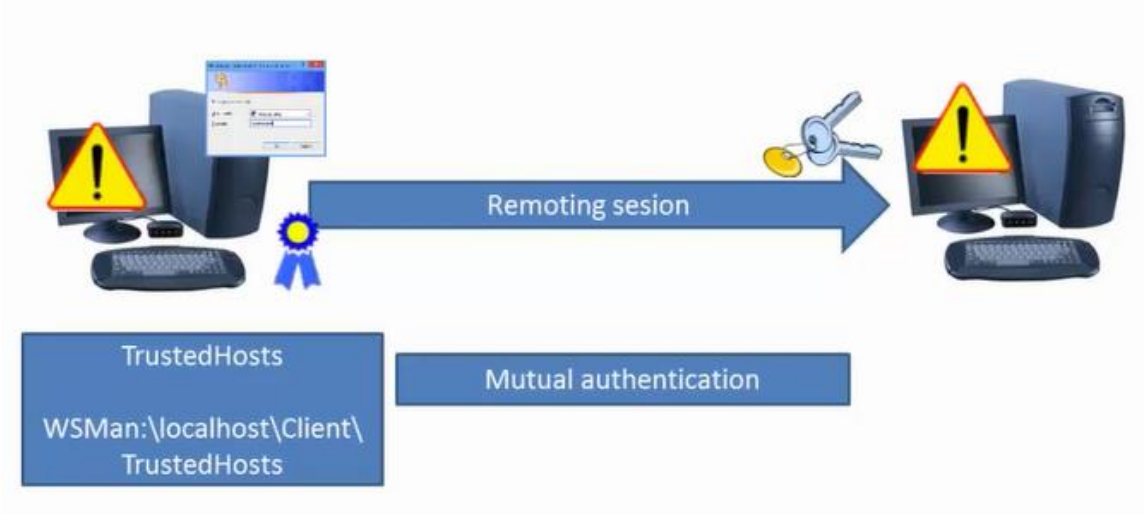
Problem pozostaje, gdy komputery nie są w domenie lub pozostają w dwóch nie ufających sobie domenach, wtedy pomocny jest certyfikat dla komputera

Informacje zaszyfrowane kluczem prywatnym komputera można odesłać do komputera klienckiego który mając klucz publiczny serwera przekonuje się, że jest to serwer, z którym miało być nawiązane połączenie, ten proces to wzajemne uwierzytelnianie (mutual authentication) Dopiero po uwierzytelnieniu obu stron poświadczenia użytkownika mogą być bezpiecznie przesłane.



Gdy pracujesz na komputerach nie znajdujących się w domenie lub pozostają w dwóch nie ufających sobie domenach, a konfiguracja https lub instalacja certyfikatów miała by być niemożliwa to mechanizm mutual authentication można wyłączyć, ale osłabia to bezpieczeństwo.

Wyłączenie mutual authentication polega na tym, że po stronie **klienta** na wirtualnym napędzie WSMan: odpowiadającym za konfiguracje remotingu w \localhost\Client\ we właściwości TrustedHosts należy dopisać te nazwy komputerów do których połączenie będzie się odbywać bez zwrotnego sprawdzenia tożsamości komputera zdalnego,



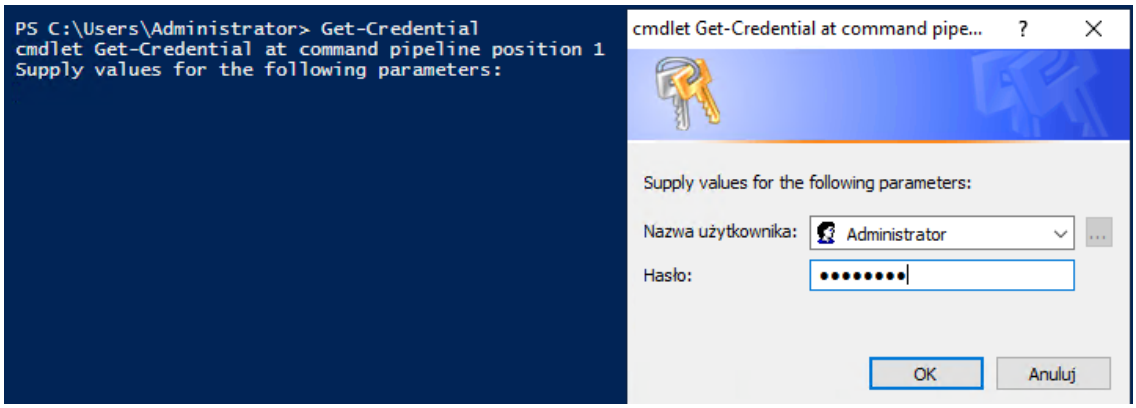
jeśli wpiszesz tam gwiazdkę to oznaczać to będzie, że połączenia do wszystkich komputerów mają się odbywać bez sprawdzenia ich tożsamości, to duża luka w zabezpieczeniach dlatego aby zminimalizować ryzyko, jeżeli

zdecydujesz się korzystać z trustedhosts to wymień tam nazwy komputerów jedna po drugiej lub wpisz maskę definiującą listę zaufanych komputerów.

```
PS C:\Users\Administrator> Invoke-Command -ComputerName serwer -ScriptBlock { Get-Process -Name notepad }
[serwer] Connecting to remote server serwer failed with the following error message : O
dmowa dostępu. For more information, see the about_Remote_Troubleshooting Help topic.
    + CategoryInfo          : OpenError: (serwer:String) [], PSRemotingTransportExcept
   ion
    + FullyQualifiedErrorId : AccessDenied,PSSessionStateBroken

PS C:\Users\Administrator> Set-Item WSMan:\localhost\Client\TrustedHosts -Value "serwer" -Force

PS C:\Users\Administrator> get-Item WSMan:\localhost\Client\TrustedHosts


   WSManConfig: Microsoft.WSMan.Management\WSMan::localhost\Client

Type            Name                             SourceOfValue   Value
----            ----                             -------------   -----
System.String   TrustedHosts                                    serwer
```

Wyślemy poświadczenia do komputera zdalnego poświadczenia wprowadzone przez lokalnego użytkownika

Zapytam o poświadczenia lokalnego użytkownika.

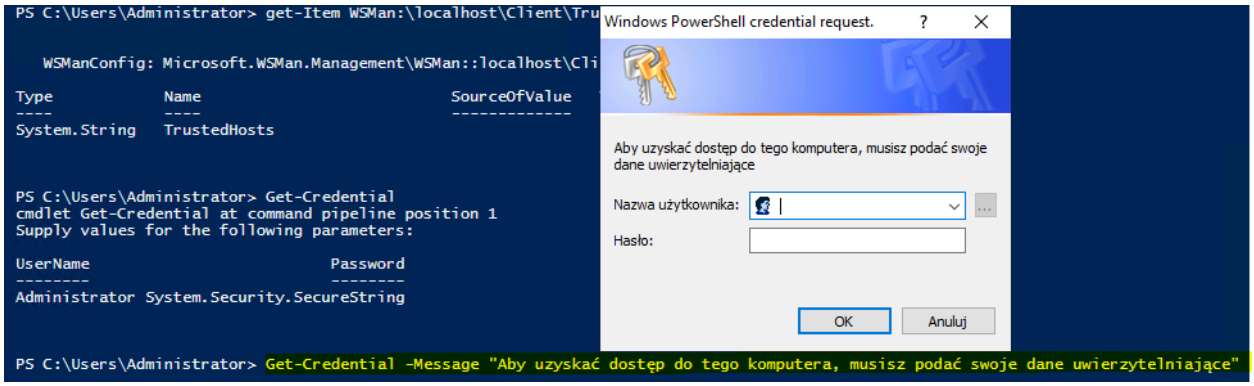


Zwracany jest obiekt który można wykorzystać do połączenia z zdalnym komputerem.

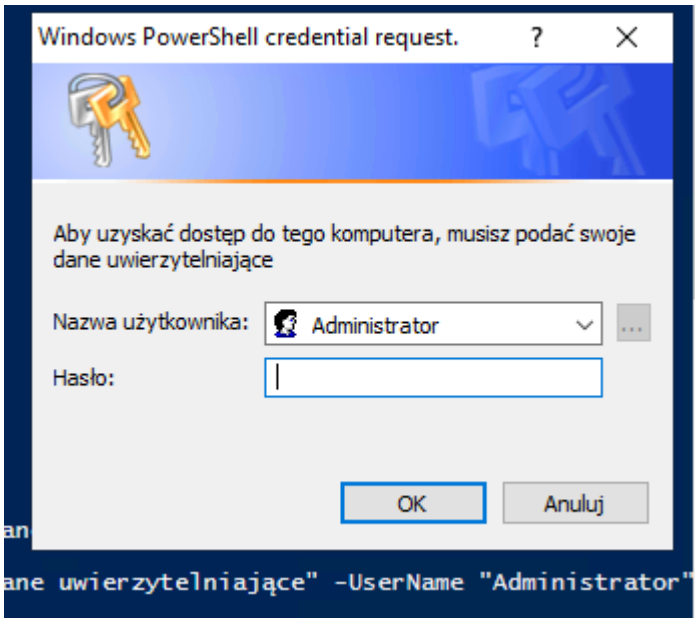
```
PS C:\Users\Administrator> Get-Credential
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

UserName                       Password
--------                       --------
Administrator System.Security.SecureString
```

Jeśli podasz parametr -Message to okienko pytające o dane użytkownika będzie instruować użytkownika jakie poświadczenia mają być wprowadzone

"Aby uzyskać dostęp do tego komputera, musisz podać swoje dane uwierzytelniające"

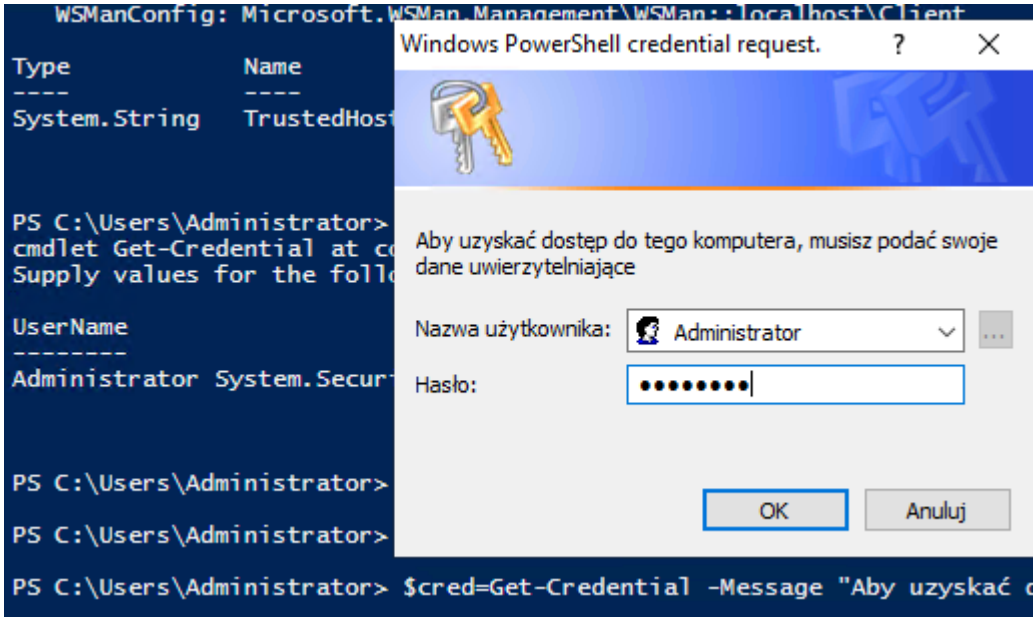


Dodanie -UserName "Administrator"

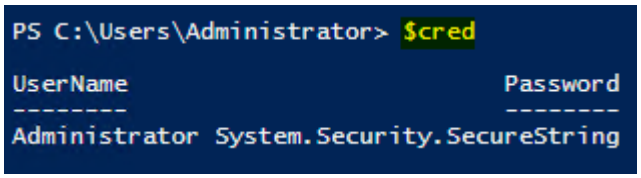


Nazwa wstępnie wypełniona

Komenda $cred=Get-Credential -Message "Aby uzyskać dostęp do tego komputera, musisz podać swoje dane uwierzytelniające" -UserName "Administrator" zapyta a następne zapisze w zmiennej $cred wartość poswiadczeń



To wyświetlam zawartość tej zmiennej



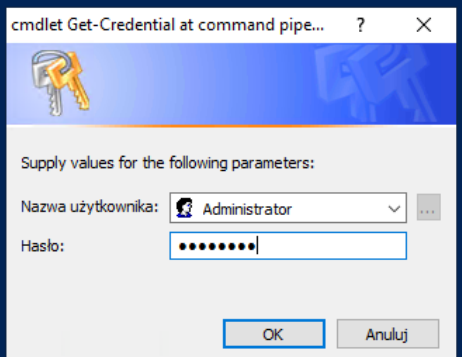
Modyfikując komendę Invoke-Command połączę się do komputera zdalnego i zapytam tam o listę procesów korzystając z pobranego poświadczenia użytkownika -Credential $cred

Przy założeniu, że na komputerze serwer uruchomiłem notatnik komenda dział

```
PS C:\Users\Administrator> Invoke-Command -ComputerName serwer -Credential $cred -ScriptBlock { Get-Process -Name notepad }

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName                                 PSComputerName
-------  ------    -----      -----     ------      --  -- -----------                                 --------------
    243      14     2924      15608       0,02    4476   2 notepad                                     serwer
```

Jeżeli nie chcemy korzystać z zmiennej to polecenie Get-Credential możemy wbudować do Invoke-Command

```
PS C:\Users\Administrator> Invoke-Command -ComputerName serwer -Credential (Get-Credential) -ScriptBlock { Get-Process -Name notepad }
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
```




```
PS C:\Users\Administrator> Invoke-Command -ComputerName serwer -Credential (Get-Credential) -ScriptBlock { Get-Process -Name notepad }
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName                                 PSComputerName
-------  ------    -----      -----     ------      --  -- -----------                                 --------------
    238      13     2568      13292       0,02    4476   2 notepad                                     serwer
```

Spróbujmy zatrzymać proces na komputerze zdalnym

```
PS C:\Users\Administrator> Invoke-Command -ComputerName serwer -Credential $cred -ScriptBlock { Get-Process -Name notepad } | Stop-Process
Stop-Process : Cannot find a process with the process identifier 4476.
At line:1 char:100
+ ... ntial $cred -ScriptBlock { Get-Process -Name notepad } | Stop-Process
+                                                            ~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (4476:Int32) [Stop-Process], ProcessCommandException
    + FullyQualifiedErrorId : NoProcessFoundForGivenId,Microsoft.PowerShell.Commands.StopProcessCommand
```

Uwaga Stop-Process w powyższej komendzie został uruchomiony lokalnie. Dlaczego? Ponieważ polecenie Stop-Process działające lokalnie może zatrzymywać procesy działające tylko lokalnie. Z potoku otrzymało iD procesu zdalnego, na szczęście takiego procesu nie było lokalnie, bo został by zakończony.

Naprawa to potok w całości w bloku -ScriptBlock

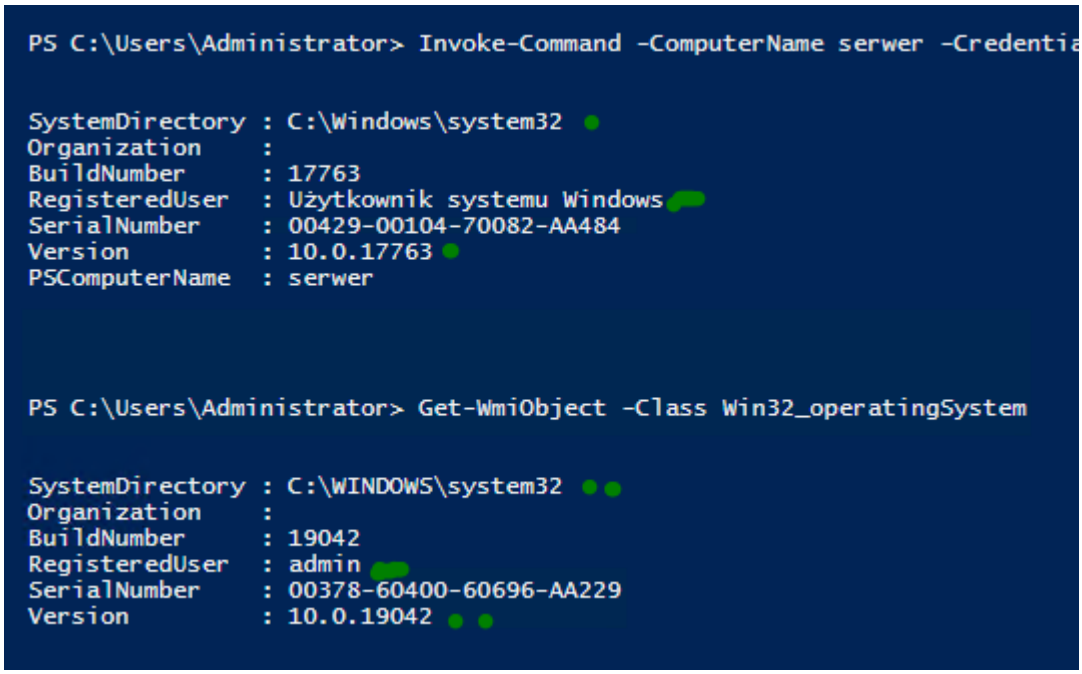
```
PS C:\Users\Administrator> Invoke-Command -ComputerName serwer -Credential $cred -ScriptBlock { Get-Process -Name notepad | Stop-Process }
```

Tak samo można wysyłać inne komendy PowerShell

```
Invoke-Command -ComputerName serwer -Credential $cred -ScriptBlock { Get-WmiObject -Class Win32_operatingSystem }
```

```
PS C:\Users\Administrator> Invoke-Command -Compu


SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 17763
RegisteredUser  : Użytkownik systemu Windows
SerialNumber    : 00429-00104-70082-AA484
Version         : 10.0.17763
PSComputerName  : serwer
```

Dla porównania wynik tej samej komendy na komputerze zdalnym i lokalnym



Komedy uruchomiły się na różnych komputerach

Dowiedziałeś się jak uwierzytelnić użytkownika na systemie zdalnym

Wiesz, jak pobierać poświadczenia użytkownika - Get-credentials

Znasz zagrożenia podczas korzystania z remotingu

Wiesz co to jest mutual authentication

Umiesz przesyłać poświadczenia do systemów zdalnych

Wiesz co to jest i jak używać TrustedHosts