Wysyłanie poczty ze skryptu

Poziom: średniozaawansowany **Dotyczy: Windows PowerShell 5.1, PowerShell 7+**

Czas: ~20 min

`Send-MailMessage` *jest oznaczone przez Microsoft jako przestarzałe od kilku lat, ale wciąż działa i wciąż jest tym, co większość istniejących skryptów administracyjnych realnie wywołuje — ta strona pokazuje, jak go używać świadomie, i czym warto je docelowo zastąpić.*

Skróty użyte na tej stronie (9)

Skrót	Co znaczy
API	Application Programming Interface: sposób, w jaki programy rozmawiają ze sobą
CSV	w Windowsie: Cluster Shared Volume, dysk widoczny naraz dla wszystkich węzłów klastra; w danych: Comma-Separated Values, plik z wartościami oddzielonymi przecinkami
HTML	HyperText Markup Language: format tekstowy stron WWW, tu używany jako format raportu GPO do przeglądania w przeglądarce
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy
OAuth	Open Authorization: standard logowania jednej usługi do drugiej bez podawania jej hasła, przez token wydany po zalogowaniu w przeglądarce
SMTP	Simple Mail Transfer Protocol: protokół wysyłania poczty
SSL	Secure Sockets Layer: dawna nazwa szyfrowania, dziś formalnie TLS
STARTTLS	włączenie szyfrowania w trakcie już nawiązanej, nieszyfrowanej sesji pocztowej
TLS	Transport Layer Security: szyfrowanie połączeń, np. w HTTPS

Fiszka

ZMIENIA SYSTEM

Wysłanie prostej wiadomości e-mail: Send-MailMessage

```
$EmailFrom = "AdresNadawcy@NazwaDomeny"
$EmailTo = "AdresOdbiorcy@NazwaDomeny"
```

```
$SMTPServer = "AdresSerwera"
$SMTPPort = 587
$SMTPUsername = "NazwaKontaSMTP"
$SMTPPassword = "HasloDoSkrzynki"
$Subject = "TematWiadomosci"
$Body = "TrescWiadomosci"

$SMTPPasswordSecure = ConvertTo-SecureString $SMTPPassword -AsPlainText -Force
$Credential = New-Object System.Management.Automation.PSCredential ($SMTPUsername,
$SMTPPasswordSecure)

Send-MailMessage -From $EmailFrom -To $EmailTo -Subject $Subject -Body $Body -SmtpServer
$SMTPServer -Port $SMTPPort -Credential $Credential -UseSsl
```

Uruchamiamy skrypt z poziomu PowerShella. Budujemy poświadczenia do skrzynki SMTP tym samym mechanizmem `PSCredential`, co opisany na stronie Poświadczenia i bezpieczne hasła w skryptach — hasło jawnym tekstem w zmiennej `$SMTPPassword` jest tu pokazane dla czytelności przykładu, w skrypcie produkcyjnym powinno pochodzić z zaszyfrowanego pliku albo z magazynu `SecretStore`, nie z linijki kodu.

Przykład:

```
PS C:\Skrypty> .\wyslij-mail.ps1
PS C:\Skrypty>
```

`Send-MailMessage` przy powodzeniu nie wypisuje niczego na ekran — brak komunikatu jest sygnałem sukcesu, błąd wysyłki zawsze zwraca czerwony wyjątek.

SMTP = Simple Mail Transfer Protocol: protokół wysyłania poczty

ZMIENIA SYSTEM

Wysłanie wiadomości z treścią HTML

```
$Body = "<h2>Raport dzienny</h2><p>Wszystkie zadania zakończone <b>poprawnie</b>.</p>"

Send-MailMessage -From $EmailFrom -To $EmailTo -Subject $Subject -Body $Body -BodyAsHtml
-SmtpServer $SMTPServer -Port $SMTPPort -Credential $Credential -UseSsl
```

`-BodyAsHtml` mówi klientowi pocztowemu odbiorcy, żeby zinterpretował treść jako znaczniki HTML zamiast wyświetlić je jako zwykły tekst — przydatne przy automatycznych raportach z tabelami i pogrubieniami zamiast surowego tekstu.

HTML = HyperText Markup Language: format tekstowy stron WWW, tu używany jako format raportu GPO do przeglądania w przeglądarce

TYLKO ODCZYT

Wysłanie wiadomości z załącznikiem

```
Send-MailMessage -From $EmailFrom -To $EmailTo -Subject $Subject -Body $Body `
    -Attachments "SciezkaDoPliku" -SmtpServer $SMTPServer -Port $SMTPPort -Credential
$Credential -UseSsl
```

`-Attachments` przyjmuje jedną ścieżkę albo tablicę kilku ścieżek naraz (`-Attachments "plik1.csv", "plik2.log"`) — typowe zastosowanie to dołączenie raportu CSV wygenerowanego wcześniej w tym samym skrypcie, na przykład tego z fiszki Zapisanie listy połączeń do pliku CSV ze strony o sieci.

CSV = w Windowsie: Cluster Shared Volume, dysk widoczny naraz dla wszystkich węzłów klastra; w danych: Comma-Separated Values, plik z wartościami oddzielonymi przecinkami

TYLKO ODCZYT

Wysłanie wiadomości do wielu odbiorców naraz

```
$odbiorcy = @("AdresOdbiorcy1@NazwaDomeny", "AdresOdbiorcy2@NazwaDomeny")

Send-MailMessage -From $EmailFrom -To $odbiorcy -Cc "AdresDoWiadomosci@NazwaDomeny" -
Subject $Subject -Body $Body -SmtpServer $SMTPServer -Port $SMTPPort -Credential
$Credential -UseSsl
```

`-To` przyjmuje tablicę adresów tak samo jak pojedynczy tekst — `-Cc` i `-Bcc` działają analogicznie, każdy jako osobny parametr.

> **Uważaj**

- Microsoft oznaczył `Send-MailMessage` jako przestarzałe (`deprecated`), bo polecenie nie wspiera nowoczesnych mechanizmów uwierzytelniania (OAuth 2.0, wymagane dziś przez Microsoft 365 i Gmail) i opiera się na starszej bibliotece .NET (`System.Net.Mail`), której Microsoft sam już nie rozwija ani nie łata pod kątem nowych standardów bezpieczeństwa transportu.
- Uwierzytelnianie logujące się samym loginem i hasłem (bez OAuth) do skrzynki Microsoft 365 przestaje działać wraz z wygaszaniem uwierzytelniania podstawowego (Basic Authentication) przez Microsoft — `Send-MailMessage` w tym scenariuszu wymaga konta SMTP z wyjątkiem albo przekierowania przez lokalny przekaźnik SMTP.
- Wiadomość bez `-UseSsl` na porcie 587 zwykle zawiedzie z błędem serwera odmawiającego uwierzytelnienia w połączeniu jawnym — `-UseSsl` uruchamia STARTTLS, opisane niżej.

OAuth = Open Authorization: standard logowania jednej usługi do drugiej bez podawania jej hasła, przez token wydany po zalogowaniu w przeglądarce · **SMTP** = Simple Mail Transfer Protocol: protokół wysyłania poczty · **STARTTLS** = włączenie szyfrowania w trakcie już nawiązanej, nieszyfrowanej sesji pocztowej

Z życia wzięte

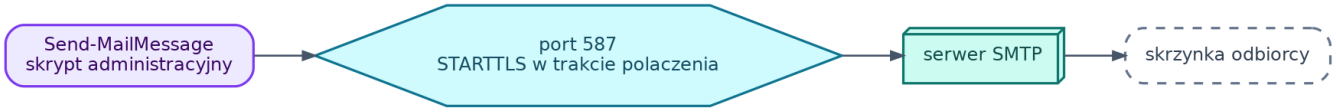
Z ŻYCIA WZIĘTE

My, zespół IT, mamy od dawna skrypt na wsus01 wysyłający co noc raport z zainstalowanych aktualizacji przez `Send-MailMessage` na skrzynkę Microsoft 365. Pewnego dnia raporty przestają przychodzić, a skrypt zwraca błąd uwierzytelnienia, mimo że hasło konta się nie zmieniło. Okazuje się, że dział bezpieczeństwa Microsoftu wygasił uwierzytelnianie podstawowe dla tej dzierżawy, a `Send-MailMessage` nigdy nie potrafiło zalogować się przez OAuth 2.0.

Po co to istnieje

Skrypt administracyjny prędzej czy później potrzebuje kogoś powiadomić — że backup się nie powiódł, że dysk się zapełnia, że audyt z poprzedniej strony wykrył coś wartego uwagi. `Send-MailMessage` przez lata było jedynym wbudowanym w PowerShella poleceniem robiącym to bez instalowania niczego dodatkowego, stąd jego wszechobecność w istniejących skryptach mimo formalnego statusu przestarzałego.

> **Pojęcie: STARTTLS.** Mechanizm, w którym połączenie SMTP zaczyna się jako zwykłe, nieszyfrowane połączenie na porcie 587, a dopiero po nawiązaniu klient i serwer negocjują przejście na szyfrowanie TLS w trakcie tej samej sesji, zanim padnie jakiekolwiek polecenie z danymi uwierzytelniającymi albo treścią wiadomości. To odróżnia port 587 od portu 465 (SMTPS), gdzie szyfrowanie TLS obejmuje połączenie od pierwszego bajtu, bez etapu negocjacji w locie. `-UseSsl` w `Send-MailMessage` uruchamia właśnie STARTTLS na porcie 587 — mimo nazwy parametru sugerującej czyste SSL, w praktyce to najczęściej STARTTLS, bo to ten wariant obsługuje zdecydowana większość serwerów SMTP w 2026 roku.



Skrypt laczy sie na port 587, negocjuje STARTTLS z serwerem SMTP i dopiero wtedy dostarcza wiadomosc (kliknięcie otwiera pełny rozmiar)

Jak to działa

Powód formalnego wycofania `Send-MailMessage` przez Microsoft nie jest kosmetyczny — to polecenie nie potrafi uwierzytelnić się przez OAuth 2.0, a to jest dziś jedyny mechanizm, który Microsoft 365 akceptuje domyślnie dla nowych skrzynek. Skrypty, które nadal chcą wysyłać pocztę przez Microsoft 365 z uwierzytelnieniem loginem i hasłem, wymagają jawnie włączonego wyjątku uwierzytelniania podstawowego dla konkretnej skrzynki (coraz rzadziej dostępnego w nowych dzierżawach) albo przekierowania przez lokalny przekaźnik SMTP, który sam obsłuży uwierzytelnienie OAuth po swojej stronie.

Czym zastępujemy Send-MailMessage. Trzy realistyczne kierunki, w zależności od tego, dokąd wysyłamy pocztę i jak bardzo skrypt ma być przyszłościowy:

Zamiennik	Kiedy sięgamy
MailKit (biblioteka .NET)	skrypt PowerShellowy wysyłający przez dowolny serwer SMTP z pełnym wsparciem nowoczesnego uwierzytelniania i szyfrowania, bez zależności od konkretnego dostawcy poczty
Microsoft Graph API	wysyłka przez skrzynkę Microsoft 365/Exchange Online z uwierzytelnieniem OAuth 2.0, zgodnie z kierunkiem, w jaki idzie sam Microsoft
`curl` z SMTP	najprostsze, jednorazowe wysłanie z poziomu skryptu bez dociągania żadnej dodatkowej biblioteki — `curl` obsługuje protokół SMTP natywnie od dawna

Wysłanie wiadomości przez curl z protokołem SMTP

```
curl.exe --url "smtps://AdresSerwera:465" `
    --ssl-reqd `
    --mail-from "AdresNadawcy@NazwaDomeny" `
    --mail-rcpt "AdresOdbiorcy@NazwaDomeny" `
    --user "NazwaKontaSMTP:HasloDoSkrzynki" `
    --upload-file "SciezkaDoPliku.eml"
```

`curl` wymaga gotowej treści wiadomości w formacie `.eml` (nagłówki `From`/`To`/`Subject` i treść, rozdzielone pustą linią) podanej przez `--upload-file` — nie buduje wiadomości za nas tak, jak robił to `Send-MailMessage` przez parametry `-Subject`/`-Body`. To rozwiązanie sprawdza się jako szybki test łączności z serwerem SMTP albo jako awaryjny sposób wysyłki tam, gdzie nie chcemy instalować żadnej dodatkowej biblioteki .NET.

Częste błędy

Kontynuowanie użycia `Send-MailMessage` w nowym skrypcie bez świadomości, że jest przestarzałe. Działa dziś, ale opiera się na bibliotece, której Microsoft już nie rozwija — nowy skrypt pisany od zera, zwłaszcza jeśli ma wysyłać przez Microsoft 365, oszczędza sobie migracji w przyszłości, jeśli od razu sięgnie po MailKit albo Graph API.

Brak `-UseSsl` przy porcie 587. Serwer SMTP odrzuca próbę uwierzytelnienia w otwartym, nieszyfrowanym połączeniu, zanim w ogóle sprawdzi poprawność hasła — komunikat błędu bywa mylący i sugeruje problem z poświadczeniami, podczas gdy przyczyną jest brak negocjacji STARTTLS.

Umieszczenie hasła do skrzynki SMTP wprost w treści skryptu. Dokładnie ten sam problem, co opisany na stronie Poświadczenia i bezpieczne hasła w skryptach — konto SMTP używane tylko do wysyłki automatycznych powiadomień jest równie atrakcyjnym celem jak każde inne, a wyciek pozwala rozsyłać pocztę w imieniu organizacji.

□□ Ćwiczenia

>

> Ćwiczenia poniżej wymagają dostępu do testowego konta SMTP — nie wysyłamy testowej poczty przez produkcyjną skrzynkę firmową bez wyraźnej potrzeby.

Ćwiczenie 1 — zbudowanie wiadomości krok po kroku

```
$Credential = Get-Credential -Message "Poświadczenia do testowego konta SMTP"

Send-MailMessage -From "test@zielona.example" -To "test@zielona.example" `
    -Subject "Test $(Get-Date -Format 'yyyy-MM-dd HH:mm')" `
    -Body "Wiadomość testowa wysłana ze skryptu." `
    -SmtpServer "AdresSerwera" -Port 587 -Credential $Credential -UseSsl
```

Wysyłamy do siebie samych wiadomość testową z dynamicznym tematem zawierającym bieżącą datę i godzinę.

Odpowiedź

Jeśli polecenie zwraca błąd `Unable to connect to the remote server`, problem leży w łączności sieciowej albo w nazwie serwera SMTP — warto sprawdzić `Test-NetConnection -ComputerName AdresSerwera -Port 587` ze strony Sieć z wiersza poleceń przed dalszym diagnozowaniem. Jeśli połączenie się udaje, ale uwierzytelnienie zawodzi, przyczyną bywa brak `-UseSsl` albo konto SMTP wymagające nowoczesnego uwierzytelniania, którego `Send-MailMessage` nie obsługuje.

Ćwiczenie 2 — porównanie portów

```
Test-NetConnection -ComputerName "AdresSerwera" -Port 587
Test-NetConnection -ComputerName "AdresSerwera" -Port 465
Test-NetConnection -ComputerName "AdresSerwera" -Port 25
```

Sprawdzamy, które z trzech klasycznych portów SMTP są w ogóle otwarte na danym serwerze pocztowym.

Odpowiedź

Port `587` z STARTTLS jest dziś standardowym portem do wysyłki uwierzytelnionej poleceniem klienckim. Port `465` (SMTPS) zakłada szyfrowanie TLS od pierwszego bajtu połączenia, bez etapu negocjacji. Port `25` jest tradycyjnie zarezerwowany dla komunikacji serwer-serwer i coraz częściej blokowany dla ruchu wychodzącego z sieci klienckich przez dostawców internetu właśnie po to, żeby utrudnić rozsyłanie spamu bezpośrednio z zainfekowanych stacji roboczych.

Pytania kontrolne

1. Dlaczego skrypt wysyłający pocztę przez konto Microsoft 365 przestał nagle działać, mimo że hasło się nie zmieniło?

Odpowiedź

Najbardziej prawdopodobną przyczyną jest wygaszenie uwierzytelniania podstawowego (Basic Authentication) po stronie Microsoft 365 dla tej dzierżawy albo tej konkretnej skrzynki — `Send-MailMessage` loguje się loginem i hasłem, nie potrafi przeprowadzić uwierzytelnienia OAuth 2.0, którego Microsoft dziś wymaga domyślnie. Rozwiązaniem jest migracja na Graph API z uwierzytelnieniem aplikacyjnym albo skonfigurowanie lokalnego przekaźnika SMTP obsługującego OAuth po swojej stronie.

2. Czym różni się port 465 od portu 587 w kontekście szyfrowania połączenia SMTP?

Odpowiedź

Port `465` (SMTPS) zakłada połączenie szyfrowane TLS od samego początku, zanim padnie jakiekolwiek polecenie protokołu SMTP. Port `587` zaczyna połączenie jawnym tekstem i dopiero w jego trakcie klient i serwer negocjują przejście na szyfrowanie przez polecenie `STARTTLS` — `-UseSsl` w `Send-MailMessage` na porcie 587 uruchamia właśnie ten drugi mechanizm, mimo że nazwa parametru sugeruje czyste SSL/TLS od początku.

Powiązane: Poświadczenia i bezpieczne hasła w skryptach · Sieć z wiersza poleceń · Skrypty PowerShell: struktura i dobre praktyki

Źródła: `Get-Help Send-MailMessage` · dokumentacja Microsoft Learn, „Send-MailMessage is not recommended" i „MailKit documentation"

Revision #40
Created 2025-03-10 14:03:06 UTC by Tomasz Konieczny
Updated 2026-10-26 14:06:58 UTC by Tomasz Konieczny