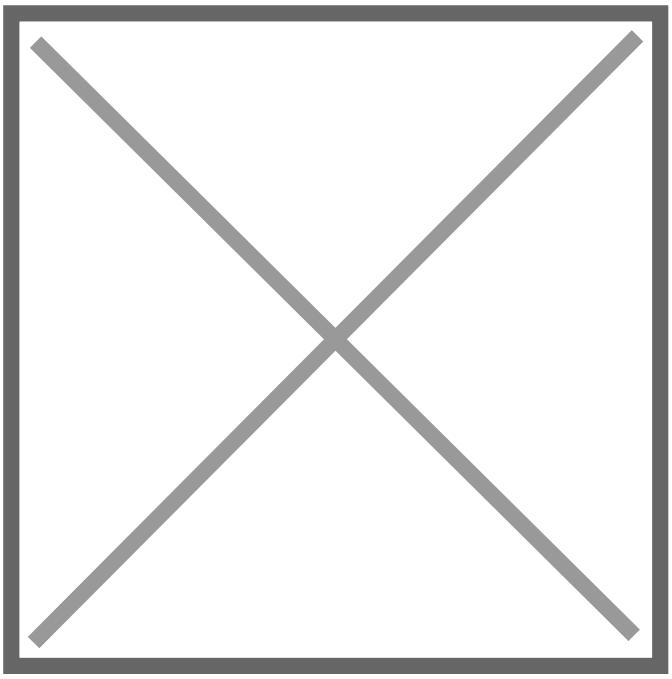
Windows Active Directory Info

- Adding Email Alias to AD Synced User
- Fix “The trust relationship between this workstation and the primary domain has failed.”

Adding Email Alias to AD Synced User

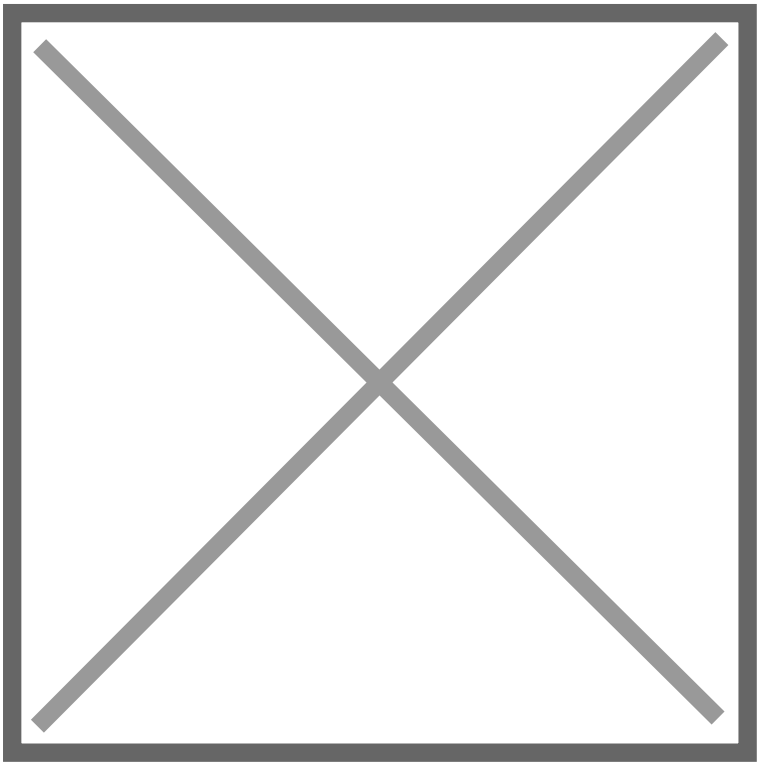
If you are using Office 365 with Azure AD Connect (or the older DirSync) you know that some changes to accounts cannot be made via the O365 admin portal. For instance, if someone gets married and changes their name, you may wish to add a new email address for them.

If you try to add an alias (second email address) to an account, you will get an error similar to this:



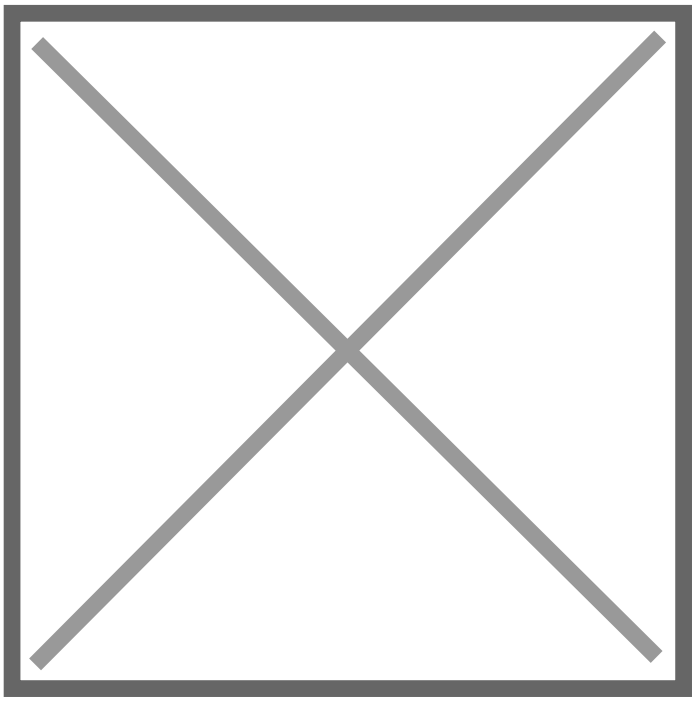
This error has made many people think they need to keep an Exchange Server up and running on their local network. Thankfully, that's not the case. You can easily add an alias via Active Directory Users and Computers (ADUC).

To do this, open ADUC and find the User you want to modify. Make sure that Advanced Features is checked, under View on the top menu. Double click on the User then click on the Attribute Editor tab.



Scroll down to the Proxy Address field and double click to open it for editing. It may be blank, which is fine, or it may already have some information in it. If it’s blank your first step is to add the existing email account in the format **SMTP:email@testemail.com**. Make sure to capitalize SMTP as that’s how the default account is determined. For the alias account you want to add, use the format: **smtp:aliasemail@testemail.com**. You can add as many aliases as needed, just be sure that they all use lower case for smtp.

After entering the information, it should look something like this:



When done click OK until you are out of ADUC and then sit back and be patient. The cloud side will synchronize and show the new alias, but it isn’t always fast. You can do a manual sync via Azure AD Connect / DirSync, but even then it can take some time to appear on the O365 side of things.

Fix “The trust relationship between this workstation and the primary domain has failed.”

> The trust relationship between this workstation and the primary domain has failed.
> The trust relationship between this workstation and the primary domain has failed.

Repair a computer’s corrupted domain trust relationship with PowerShell, no restart required.

What causes a domain computer to lose its trust relationship?

If you’ve been working in an Active Directory environment long enough, you are bound to encounter a domain-joined computer whose membership becomes invalid. Around our office, we call this “falling off the domain” (which seems to be a common term, though I haven’t found its origin).

The two scenarios I have encountered most often when this happens are:

1. A VM or system restore has been performed from a backup prior to a computer’s AD password being changed.
2. Someone accidentally adds a computer of the same name to the domain, thereby invalidating the current computer.

Windows events related to a computer falling off the domain

When one of these two scenarios occurs, you will see a logon error of *“The trust relationship between this workstation and the primary domain has failed.”* You will also see the following events in the Windows System log of the computer with the broken trust relationship:

Event 3210 – Error – NETLOGON

This computer could not authenticate with \\<DC NAME>, a Windows domain controller for domain <DOMAIN>, and therefore this computer might deny logon requests. This inability to authenticate might be caused by another computer on the same network using the same name or the password for this computer account is not recognized. If this message appears again, contact your system administrator.

Event 8019 – Warning – DNS Client Events

The system failed to register host (A or AAAA) resource records (RRs) for network adapter with settings:

<HOST SETTINGS>

The reason the system could not register these RRs was because of a security related problem. The cause of this could be (a) your computer does not have permissions to register and update the specific DNS domain name set for this adapter, or (b) there might have been a problem negotiating valid credentials with the DNS server during the processing of the update request.

Event 130 – Warning – Time-Service

NtpClient was unable to set a domain peer to use as a time source because of failure in establishing a trust relationship between this computer and the '<DOMAIN>' domain in order to securely synchronize time. NtpClient will try again in 15 minutes and double the reattempt interval thereafter. The error was: The trust relationship between this workstation and the primary domain failed. (0x800706FD)

Repair computer's trust relationship with domain

In the past, your option for fixing a computer's trust relationship with the domain was to remove it from the domain, reboot, re-add it to the domain, and reboot. Not exactly a seamless operation, especially if the system is remote.

In PowerShell 3.0, Microsoft introducted the cmdlet **Test-ComputerSecureChannel**. It is not telling from the name, but this cmdlet can not only check whether a computer's domain trust is still valid, but it can repair it if it is not!

Using Test-ComputerSecureChannel to check and repair domain trust

relationship

Here is how it works. On my afflicted computer, I am going to open an elevated admin PowerShell session. First thing I am going to do is check the current status of the computer's domain trust relationship. I can do this by issuing the naked cmdlet.

```
PS C:\Windows\system32> Test-ComputerSecureChannel
False
```

OK. We confirmed our suspicions based on the Windows events we saw in the log. Now let's work the magic of repairing the trust. I am going to run **Test-ComputerSecureChannel** with the **-Repair** parameter.

```
PS C:\Windows\system32> Test-ComputerSecureChannel -Repair
Test-ComputerSecureChannel : Cannot reset the secure channel password for the computer account i
Operation failed with the following exception: The user name or password is incorrect.
.
At line:1 char:1
+ Test-ComputerSecureChannel -Repair
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : OperationStopped: (Win10A:String) [Test-ComputerSecureChannel], Ir
   n
    + FullyQualifiedErrorId : FailToResetPasswordOnDomain,Microsoft.PowerShell.Commands.TestComp
   nd
```

Why did it fail? Because if you recall earlier, when I attempt to login using a domain user, I was getting the error message "*The trust relationship between this workstation and the primary domain has failed.*" I am currently signed into the computer using a local Administrator account. I need to use the **-Credential** parameter and pass in credentials for a domain user that has the rights to add a computer to the domain. To do this, I will use **Get-Credential** and store it in a variable called **$cred**. I will then pass that variable into the **Test-ComputerSecureChannel** cmdlet.

Here is a demo video, as well as the commands and output below.

Video Player

00:09

00:32

```
PS C:\Windows\system32> $cred = Get-Credential

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
```

```
PS C:\Windows\system32> Test-ComputerSecureChannel -Credential $cred -Repair
True
PS C:\Windows\system32> Test-ComputerSecureChannel
True
```

You can see the last cmdlet runs returned **True** to confirm the domain trust relationship is now valid. I was then able to log out of the computer and log back in with a domain user successfully. No restart required!

Reference

- Test-ComputerSecureChannel | docs.microsoft.com

```
PS C:\Users\aaron> Get-Help Test-ComputerSecureChannel

NAME
    Test-ComputerSecureChannel

SYNOPSIS
    Tests and repairs the secure channel between the local computer and its domain.


SYNTAX
    Test-ComputerSecureChannel [-Confirm] [-Credential <PSCredential>] [-Repair] [-Server <Strir
    [<CommonParameters>]


DESCRIPTION
    The Test-ComputerSecureChannel cmdlet verifies that the channel between the local computer a
    working correctly by checking the status of its trust relationships. If a connection fails,
    parameter to try to restore it. Test-ComputerSecureChannel returns $True if the channel is v
    $False if it is not. This result lets you use the cmdlet in conditional statements in functi
    get more detailed test results, use the Verbose parameter.

    This cmdlet works much like NetDom.exe. Both NetDom and Test-ComputerSecureChannel use the N
    perform the actions.


RELATED LINKS
    Online Version: http://go.microsoft.com/fwlink/?LinkId=821645
    Checkpoint-Computer
    Reset-ComputerMachinePassword
    Restart-Computer
    Stop-Computer

REMARKS
    To see the examples, type: "get-help Test-ComputerSecureChannel -examples".
    For more information, type: "get-help Test-ComputerSecureChannel -detailed".
```

```
For technical information, type: "get-help Test-ComputerSecureChannel -full".
For online help, type: "get-help Test-ComputerSecureChannel -online"
```