Most Common Mistakes in Active Directory and Domain Services – Part 2

Omer Eldan Active Directory January 8, 2019 5 Minutes

In this blog post, we will continue to explore some of the most common mistakes in Active Directory and Domain Services.
In the previous post we covered the first three mistakes, and today we'll go over another three interesting issues. Enjoy your reading 🙂

Series:

- Part 1
- **Part 2**
- Part 3

Mistake #4: Keeping the Forest and Domain Functional Levels at a Lower Version

For various reasons, customers are afraid of dealing with the Forest and Domain Functional Levels (FFL and DFL in short).

Because the FFL and DFL purpose and impact are not always clear, people avoid changing it and sometimes maintain a very old functional level like Windows Server 2008 or even Windows Server 2003.

The Forest and Domain Functional Levels reflect the lowest Domain Controller version within the forest and the domain.
In other words, this attribute is telling the Domain Controllers that all DCs in the Domain or Forest are running

an OS equal to or higher than the functional level. For example, a functional level of Windows Server 2012R2 means that all DCs are running a Windows Server 2012R2 OS and above.

The functional level is used by the Active Directory to understand whether it's possible to take advantage of new features that require the Domain Controllers to be at a minimum OS version.
The FFL and DFL are also used to prevent promoting an old Domain Controller version in the domain, as it might, theoretically, affect the usability of new AD features being used by newer OS versions.

An old Forest/Domain Functional Levels may prevent you from using some very useful Active Directory features like Active Directory Recycle Bin, Domain-Based DFS namespaces, DFS Replication for SYSVOL and Fine-Grained Password Policies.
In this link, you can find the full list of Active Directory Features in each functional level.

It's also worth mention that you can roll back the FFL and DFL all the way down to Windows Server 2008R2 using the Set-ADForestMode and Set-ADDomainMode PowerShell cmdlets. See the example below:

```
#Changing The Forest Level
$CurrentForest = Get-ADForest
Set-ADForestMode -Identity $CurrentForest -Server $CurrentForest.SchemaMaster -ForestMode Windows20
#Changing The Domain Level
$CurrentDomain = Get-ADDomain
Set-ADDomainMode -Identity $CurrentDomain.Name -Server $CurrentDomain.PDCEmulator -DomainMode Windo
```

view raw

ChangeForestAndDomainFunctionalLevels.ps1

hosted with ❤ by GitHub

Bottom Line: Forest and Domain Functional Levels are used internally by the Domain Controllers and don't affect which operating systems can be used by clients (workstation and servers).
Older functionally and features are still supported in newer functional levels, so you shouldn't notice any differences, and everything is expected to continue to work as before.
If (for some reason) you still have concerns about certain applications, contact the vendor for clarification.

Do It Right: Backup your AD environment (using Windows Server Backup or any other solution you've got), upgrade the FFL and the DFL in your test environment and then in production.

Mistake #5: Use DNS as an Archive by Disabling DNS Scavenging

DNS is one of the most important services in each environment. It should be running smoothly and be up to date so it can resolve names to IP address correctly with no issues.

Yet, there are some cases when customers think about DNS as an archive for old and unused servers' names and IP addresses. In those cases, administrators disable the DNS Scavenging option to prevent old DNS records from being deleted. This is a bad habit because it could easily lead to a messy DNS with duplicated and irrelevant records, where A Records point to IP addresses which do not exist anymore, and PTRs refer to old computers deleted long time ago.

For those of you who don't know, DNS Scavenging is a DNS service responsible for cleaning-up old and unused DNS records which are not relevant anymore, based on their **timestamp**.
When DNS record is being updated or refreshed by a DNS client, its timestamp gets updated with the current date and time.
DNS Scavenging designed to delete records that their timestamp is older than the 'Refresh' + 'No Refresh' intervals (which are configured in the DNS zone settings). Pay attention that static DNS records are not being scavenged at all.

If DNS Scavenging is disabled in your environment for a while, I suggest running the PowerShell script below before enabling it in order to better understand which records are going to be removed as part of the scavenging process.
The script checks any Dynamic DNS Record and decided whether it's:
• A stale record which responded to ping.
• A stale record which doesn't respond to ping.
• An updated record (not stale).

```
Function Create-DNSScavengingRecordsReport
{
    <#The script checks any Dynamic DNS Record and decided whether it's:
        1)A stale record which responded to ping.
        2)stale record which doesn't responded to ping.
        3)An updated record (not stale).#>
    $DC = (Get-ADDomainController).Name
    $DNSRoot = (Get-ADDomain).DNSRoot
    $DNSRecords = Get-DnsServerResourceRecord –ComputerName $DC –ZoneName $DNSRoot
    $DateThershold = (Get-Date).AddDays(-14)
    $DNSArray = @()
    ForEach ($DNSRecord in $DNSRecords)
    {
        If ($DNSRecord.RecordType -eq "A" -and $DNSRecord.Timestamp -ne $Null -and $DNSRecord.Host
        {
            $Computer = $DNSRecord.HostName
            $ComputerIP = $DNSRecord.RecordData.IPv4Address.IPAddressToString
            $ComputerOS = "Null"
            Try
```

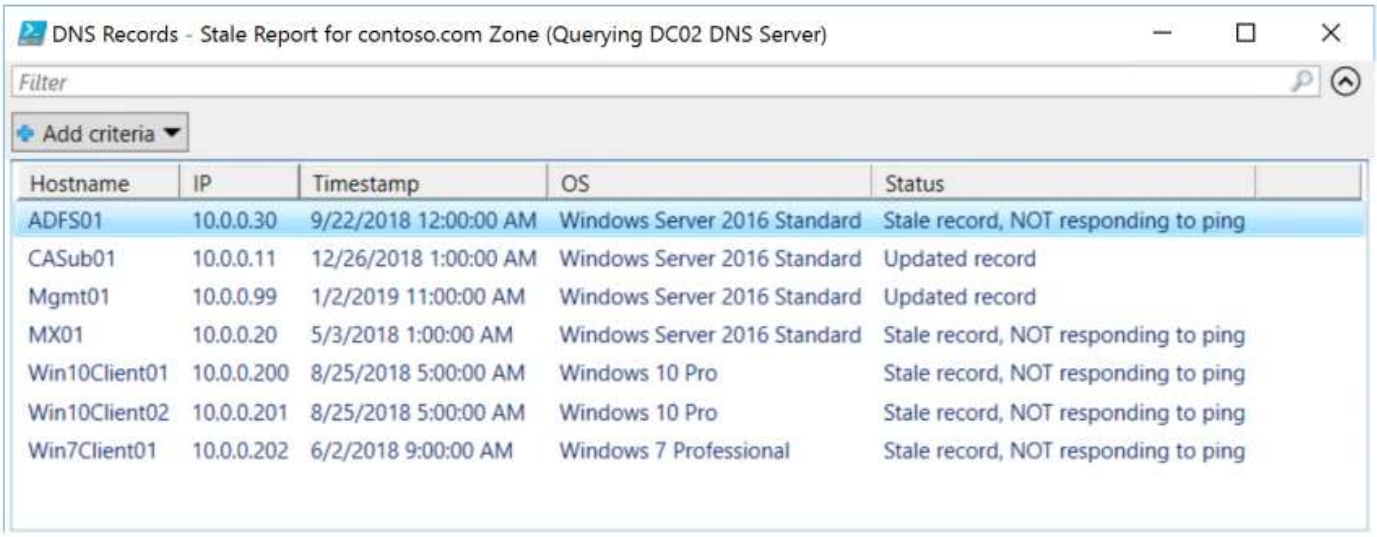
```
            {
                $ADComputer = Get-ADComputer $Computer -Properties OperatingSystem -ErrorAction St
                $ComputerOS = $ADComputer.OperatingSystem
            }
            Catch
            {
                Write-Host "The computer object could not be retreived from Active Directory. Skip
            }
            $Ping = Test-Connection -ComputerName $DNSRecord.HostName -Count 1 -ErrorAction Silent
            $DNSObject = New-Object -TypeName PSObject
            Add-Member -InputObject $DNSObject -MemberType 'NoteProperty' -Name 'Hostname' -Value
            Add-Member -InputObject $DNSObject -MemberType 'NoteProperty' -Name 'IP' -Value $Compu
            Add-Member -InputObject $DNSObject -MemberType 'NoteProperty' -Name 'Timestamp' -Value
            Add-Member -InputObject $DNSObject -MemberType 'NoteProperty' -Name 'OS' -Value $Compu
            If (($DNSRecord.Timestamp) -lt $DateThershold)
            {
                If ($Ping)
                {
                    $Status = "Stale record, responding to ping"
                }
                Else
                {
                    $Status = "Stale record, NOT responding to ping"
                }
            }
            Else
            {
                 $Status = "Updated record"
            }
            Add-Member -InputObject $DNSObject -MemberType 'NoteProperty' -Name 'Status' -Value $S
            $DNSArray += $DNSObject
        }
    }
    $DNSArray | Out-GridView -Title "DNS Records – Stale Report for $DNSRoot Zone (Querying $DC DN
}
```

view raw

Create-DNSScavengingRecordsReport.ps1

hosted with ❤ by GitHub

The script's output should look like this:



Bottom Line: DNS Scavenging is NOT the place to save all your ancient names and IP addresses. If you required to save this information, use some CMDB tool or any other platform design for this. DNS is an operational service that should response fast and reliable with the correct and relevant values only.

Do It Right: Enable DNS scavenging and get rid of those old and unused records.

Mistake #6: Using a DHCP Failover Without Configuring DDNS Update Credentials

DHCP Failover is a well-known feature that was released back in September 2012 with Windows Server 2012. The DHCP Failover provides high availability mechanism by entering two DHCP servers into a failover relationship.

When the option “Always dynamically update DNS records” in the DHCP properties is selected, the DHCP server updates the DNS with A and PTR records of DHCP clients using its own computer credentials (e.g. ‘DHCP01’ computer object).

When a DHCP Failover is configured, this can become an issue:
When the first DHCP server (e.g. **DHCP01**) in a DHCP Failover is registering a DNS record, it becomes its owner and gets the relevant permissions to update the record when needed.
If the second DHCP server (e.g. **DHCP02**) in a DHCP Failover will try to update the same record (because DHCP01 is unavailable for the moment), the update will fail because it doesn't have the required permissions to update the record.
Pay attention that if your DNS zones are configured with "Nonsecure and secure" dynamic updates (which standing aginst the best practices), security permissions on DNS records are not enforced by any mean, and records can be updated by any client, including your DHCP servers.

To resolve this, you can configure DDNS update credentials and enter the username and password of a dedicated user account you created for this purpose (e.g. SrvcDHCP).
In general, no special permissions are required.
The DHCP servers will always use this credential when registering and updating DNS records.

Before changing the DNS dynamic update credentials, you may consider changing the ownership and the permissions of **existing** DNS records to include the new user account, especially if your DHCP environment is running for a long time.

In order to complete this, you can use the PowerShell script below.
The script examines each DNS record and displays a table with records that meet all of the following conditions:

1. The DNS record is a dynamic record.
2. Record's current owner is a DHCP server.
3. Record's type is A or PTR.

If approved by the user, the script updates the selected records with the new owner and add the user account to the records ACL with a 'full control' permission.

Bottom Line: Using a DHCP Failover without configuring DNS dynamic update credentials will result in DNS update failures when one DHCP server will try to update records that were registered by the second DHCP server.

Do It Right: If you are using DHCP Failover, you should configure DNS dynamic updates credentials on both DHCP servers.

In the next (and last) blog post we'll talk about a few more issues and warp up this series.

Author

 Omer Eldan

Tagged: DHCP, DNS

You must log in to post a comment.

Search This Blog

Search ...

Tags

Deep Technical

Microsoft Delivery Approach

Industry Solutions

Categories

Active Directory

AVD

Azure

Azure Active Directory

Azure MFA

Azure Monitor

Azure Sentinel

BI and Analytics

Certification

Defender

DevOps

Failover Clustering

Group Policy

Hyper-V

Identity

Infrastructure as Code

Intune

KMS

KQL

Log Analytics

Logic Apps

Microservices

Microsoft 365 Defender

Microsoft Authenticator Application

Microsoft Defender for Cloud Apps

Microsoft Defender for Endpoint

Microsoft Defender for Identity

Microsoft Endpoint Manager

Microsoft Intune

Microsoft Sentinel

Office 365

OMS

Performance

PowerShell

Security

SIEM

SOAR

Soft Skills

System Center

Uncategorized

Windows

WSUS

Follow Blog via Email

Enter your email address to follow this blog and receive notifications of new posts by email.

Email Address

Follow

About

Contact Us

Disclaimer