Protokoll zu DNS Task

Inhaltsverzeichnis

[Aufgabenstellung (Records): 2](#_Toc132044763)

[Script: 2](#_Toc132044764)

[„A“: 3](#_Toc132044765)

[„MX“: 3](#_Toc132044766)

[„NS“: 4](#_Toc132044767)

[Ergebnis: 4](#_Toc132044768)

[Aufgabenstellung (Namensauflösung): 5](#_Toc132044769)

[Script: 5](#_Toc132044770)

[Ergebnis: 6](#_Toc132044771)

# Aufgabenstellung (Records):

* Lege die Einträge der csv-Datei als Records auf deinem DNS-Serve an.
* Schreibe ein Powershell-Script für die Anlage.

## Script:

$list = Import-CSV -Path "C:\dns\_records.csv" -Delimiter ";"

foreach($csv in $list){

    $name = $csv.name

    $ipv4 = $csv.ipv4

    $type = $csv.type

  if($type -eq "A"){

    Add-DnsServerResourceRecordA -Name $name -ZoneName "suljevic.at" -AllowUpdateAny -IPv4Address $ipv4 -TimeToLive 01:00:00

  }

  if($type -eq "MX"){

    Add-DnsServerResourceRecordMX -Preference 10 -Name $name -TimeToLive 01:00:00 -MailExchange $ipv4 -ZoneName "suljevic.at"

  }

  if($type -eq "NS"){

    Add-DnsServerResourceRecordA -IPv4Address $ipv4 -Name $name -ZoneName "suljevic.at";

    Add-DnsServerResourceRecord -NS -ZoneName suljevic.at -Name suljevic.at -NameServer $($name + ".suljevic.at.");

  }

}

Der Inhalt von der CSV Datei wird ausgelesen:

$list = Import-CSV -Path "C:\dns\_records.csv" -Delimiter ";"

Danach wird der Inhalt in eine foreach Schleife durchgegangen

Die if Abfragen schauen, ob der **$type** gleich einem **A, MX** oder **NS** entspricht

### „A“:

Add-DnsServerResourceRecordA -Name $name -ZoneName "suljevic.at" -AllowUpdateAny -IPv4Address $ipv4 -TimeToLive 01:00:00

**-Name** 🡪 Gibt einen Hostnamen an

**-ZoneName** 🡪 Gibt den Namen einer DNS-Zone an

**-AllowUpdateAny** 🡪 Gibt an, dass jeder authentifizierte Benutzer einen Ressourceneintrag mit demselben Eigentümernamen aktualisieren kann.

**-IPv4Address** 🡪 Gibt ein Array von IPv4-Adressen an.

**-TimeToLive** 🡪 Gibt den Time to Live in Sekunden für einen Ressourceneintrag an. Andere DNS-Server verwenden diese Zeitspanne, um zu bestimmen, wie lange ein Datensatz zwischengespeichert werden soll.

### „MX“:

Add-DnsServerResourceRecordMX -Preference 10 -Name $name -TimeToLive 01:00:00 -MailExchange $ipv4 -ZoneName "suljevic.at"

**-Name** 🡪 Gibt den Namen der Host- oder untergeordneten Domäne für den Mail-Exchange-Datensatz an.

**-ZoneName**  🡪 Gibt den Namen einer DNS-Zone an

**-TimeToLive** 🡪 Gibt den Time to Live in Sekunden für einen Ressourceneintrag an. Andere DNS-Server verwenden diese Zeitspanne, um zu bestimmen, wie lange ein Datensatz zwischengespeichert werden soll.

**-Preference** 🡪 Gibt eine Priorität von 0 bis 65535 für diesen MX-Ressourceneintrag an. Ein Dienst versucht, Mail-Server in der bevorzugten Reihenfolge vom niedrigsten Prioritätswert zum höchsten Prioritätswert zu kontaktieren.

**-MailExchange** 🡪 Gibt einen FQDN für einen Mail-Exchanger an

### „NS“:

Da es keinen **Add-DnsServerResourceRecordNS** gibt muss man stattdessen **Add-DnsServerResourceRecordA** und **Add-DnsServerResourceRecord** nutzen

Add-DnsServerResourceRecordA -IPv4Address $ipv4 -Name $name -ZoneName "suljevic.at";

Add-DnsServerResourceRecord -NS -ZoneName suljevic.at -Name suljevic.at -NameServer $($name + ".suljevic.at.");

**-NameServer** 🡪 Gibt den Nameserver einer Domäne an.

## Ergebnis:

Das fertige Ergebnis kann man dann unter **DNS🡪Forward Looking Zone 🡪 suljevic.at/ebertz.at** sehen

Abbildung - DNS Manager

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

# Aufgabenstellung (Namensauflösung):

* Schreibe ein Powershell-Script, dass einen FQDN auflöst und den Ablauf der Namensauflösung beginnend bei einem DNS-Root-Server im Detail beschreibt.

Beim Beispiel wurde [www.amazon.de](http://www.amazon.de) als FQDN genutzt

## Script:

$address = "www.amazon.de"

nslookup $address 8.8.8.8

nslookup.exe -type=NS . 8.8.8.8

nslookup.exe -type=A b.root-servers.net 8.8.8.8

#Damit man die Ip-Adresse in eine Variable speichert

$firstOutput = nslookup.exe -type=A b.root-servers.net 8.8.8.8 | Select-String -Pattern "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}" | Select-Object -Last 1

$ip\_address = $firstOutput -replace ".\*?(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}).\*", '$1'

nslookup.exe -type=NS de. $ip\_address

$output = nslookup.exe -type=A c.gtld-servers.net 8.8.8.8 | Select-String -Pattern "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}" | Select-Object -Last 1

$ip\_address = $output -replace ".\*?(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}).\*", '$1'

nslookup.exe -type=NS amazon.de. $ip

$Secondoutput = nslookup.exe -type=A ns4.p31.dynect.net 8.8.8.8 | Select-String -Pattern "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}" | Select-Object -Last 1

$ip\_address = $Secondoutput -replace ".\*?(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}).\*", '$1'

nslookup.exe -type=A www.amazon.de. $ipAddress

Mit dem Befehl wird die IP-Adresse, die man von nslookup.exe -type=A c.gtld-servers.net 8.8.8.8, in eine Variable gespeichert

$output = nslookup.exe -type=A c.gtld-servers.net 8.8.8.8 | Select-String -Pattern "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}" | Select-Object -Last 1

$ip\_address = $output -replace ".\*?(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}).\*", '$1'

## Ergebnis:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung - Namensauflösung von www.amazon.de

Abbildung - Namensauflösung von www.amazon.de

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung



Abbildung - Ergebnis der Namensauflösung

Am Ende ist die IP-Adresse von [www.amazon.de](http://www.amazon.de) zu sehen 18.66.21.162