



 master ▾


 


Go to file


<> Code ▾





 .gitattributes


 .gitignore


 Invoke-DNSUpdate.p...

 LICENSE


 Powermad.ps1


 Powermad.psd1


 Powermad.psm1

 README.md

README


 BSD-3-Clause license








About


PowerShell MachineAccountQuota and DNS exploit tools


 Readme

 BSD-3-Clause license

 Activity

 1.2k stars

 31 watching

 175 forks

Report repository

Releases


No releases published


Packages


No packages published

Contributors 

3







Page 1 of 11

# PowerShell MachineAccountQuota and DNS exploit tools

## Wiki

- <https://github.com/Kevin-Robertson/Powermad/wiki>

## Blog Post

- [Beyond LLMNR/NBNS Spoofing - Exploiting Active Directory-Integrated DNS](#)

## Functions

- [MachineAccountQuota Functions](#)
- [DNS Functions](#)
- [Dynamic Updates Functions](#)
- [ADIDNS Functions](#)
- [Miscellaneous Functions](#)

## MachineAccountQuota Functions

The default Active Directory ms-DS-MachineAccountQuota attribute setting allows all domain users to add up to 10 machine accounts to a domain. Powermad includes a set of functions for exploiting ms-DS-MachineAccountQuota without attaching an actual system to AD.

### Get-MachineAccountAttribute

This function can return values populated in a machine account attribute.

## Languages

- PowerShell 100.0%

**Example:**

- Get the value of 'description' from a machine account names 'test'.

```
Get-MachineAccountAttribute -MachineAccount test -  
Attribute discription
```

## Get-MachineAccountCreator

This function leverages the ms-DS-CreatorSID property on machine accounts to return a list of usernames or SIDs and the associated machine account. The ms-DS-CreatorSID property is only populated when a machine account is created by an unprivileged user.

**Example:**

- Get a list of all populated ms-DS-CreatorSID attributes.

```
Get-MachineAccountCreator
```

## Disable-MachineAccount

This function can disable a machine account that was added through New-MachineAccount. This function should be used with the same user that created the machine account.

**Example:**

- Disable a machine account named test.

```
Disable-MachineAccount -MachineAccount test
```

## Enable-MachineAccount

This function can enable a machine account that was disabled through Disable-MachineAccount. This function should be used with the same user that created the machine account.

**Example:**

- Enable a machine account named test.

```
Enable-MachineAccount -MachineAccount test
```

## New-MachineAccount

This function can add a new machine account directly through an LDAP add request to a domain controller and not by impacting the host system's attachment status to Active Directory.

The LDAP add request is modeled after the add request used when joining a system to a domain. The following (mostly validated by the DC) attributes are set:

- objectClass = Computer
- SamAccountName = Machine account name with trailing \$
- userAccountControl = 4096
- DnsHostName = FQDN
- ServicePrincipalName = 2 HOST and 2 RestrictedKrbHost SPNs using both the FQDN and account name
- unicodePwd = the specified password

A new machine account can be used for tasks such as leveraging privilege provided to the 'Domain Computers' group or as an additional account for domain enumeration, DNS exploits, etc. By default, machine accounts do not have logon locally permission. You can either use tools/clients that accept network credentials directly or through the use of 'runsas /netonly' or @harmj0y's Invoke-UserImpersonation/Invoke-RevertToSelf included with PowerView.

- <https://github.com/PowerShellMafia/PowerSploit/tree/dev/Recon>

Machine accounts created with standard users will have the mS-DS-CreatorSID populated with the standard user's SID.

Note that ms-DS-MachineAccountQuota does not provide the ability for authenticated users to delete added machine accounts from AD. Elevated privilege will need to be acquired

to remove the account if you want to avoid passing the task off to your client.

Examples:

- Add a new machine account

```
New-MachineAccount -MachineAccount test
```

- Use the added account with runas /netonly

```
runas /netonly /user:domain\test$ powershell
```

## Remove-MachineAccount

This function removes a machine account with a privileged account.

Example:

- Remove a machine account named test with domain admin credentials

```
Remove-MachineAccount -MachineAccount test -  
Credential $domainadmin
```

## Set-MachineAccountAttribute

This function can populate some attributes for an account that was added through New-MachineAccount, if a user has write access. This function should be used with the same user that created the machine account.

Here is a list of some of the usual write access enabled attributes:

- AccountDisabled
- description
- displayName
- DnsHostName
- ServicePrincipalName
- userParameters

- userAccountControl
- msDS-AdditionalDnsHostName
- msDS-AllowedToActOnBehalfOfOtherIdentity
- SamAccountName

#### Examples:

- Remove the trailing '\$' from the SamAccountName attribute

```
Set-MachineAccountAttribute -MachineName test -  
Attribute SamAccountName -Value test
```

- Use the modified account with runas /netonly

```
runas /netonly /user:domain\test powershell
```

## Invoke-AgentSmith

This function leverages New-MachineAccount to recursively create as many machine accounts as possible from a single unprivileged account through MachineAccountQuota. See the following blog post for details:

- <https://blog.netspi.com/machineaccountquota-transitive-quota>

## DNS Functions

By default, authenticated users have the 'Create all child objects' permission on the Active Directory-Integrated DNS (ADIDNS) zone. Most records that do not currently exist in an AD zone can be added/deleted.

## Dynamic Updates Functions

### Invoke-DNSUpdate

This function can be used to add/delete dynamic DNS records if the default setting of enabled secure dynamic updates is

configured on a domain controller. A, AAAA, CNAME, MX, PTR, SRV, and TXT records are currently supported. Invoke-DNSUpdate is modeled after BIND's nsupdate tool when using the '-g' or 'gsstsig' options.

#### Examples:

- Add an A record

```
Invoke-DNSUpdate -DNSType A -DNSName www -DNSData  
192.168.100.125
```

- Delete an A record

```
Invoke-DNSUpdate -DNSType A -DNSName  
www.test.local
```

- Add an SRV record

```
Invoke-DNSUpdate -DNSType SRV -DNSName  
_autodiscover._tcp.test.local -DNSData  
system.test.local -DNSPriority 100 -DNSWeight 80 -  
DNSPort 443
```

## ADIDNS Functions

---

### Disable-ADIDNSNode

This function can tombstone an ADIDNS node.

#### Example:

\*Tombstone a wildcard record.  
`Disable-ADIDNSNode -Node \*

### Enable-ADIDNSNode

This function can turn a tombstoned node back into a valid record.

#### Example:

- Enable a wildcard record.

```
Enable-ADIDNSNode -Node *
```

## Get-ADIDNSNodeAttribute

This function can return values populated in a DNS node attribute.

Example:

- Get the value populated dnsRecord attribute of a node named test.

```
Get-ADIDNSNodeAttribute -Node test -Attribute  
dnsRecord
```

## Get-ADIDNSNodeOwner

This function can return the owner of an ADIDNS Node.

Example:

- Get the owner of a node named test. `Get-ADIDNSNodeOwner -Node test`

## Get-ADIDNSPermission

This function gets a DACL of an ADIDNS node or zone.

Examples:

- Get the DACL for the default Active Directory-Integrated Zone from a domain attached system.

```
Get-ADIDNSPermission
```

- Get the DACL for a DNS node named test from a domain attached system.

```
Get-ADIDNSPermission -Node test
```

## Get-ADIDNSZone

This function can return ADIDNS zones.



#### Examples:

- Get all ADIDNS zones.

```
Get-ADIDNSZone
```

## Grant-ADIDNSPermission

This function adds an ACE to a DNS node or zone DACL.

#### Example:

- Add full access to a wildcard record for "Authenticated Users".
- Add full access to a wildcard record for "Authenticated Users".

```
Grant-ADIDNSPermission -Node * -Principal  
"authenticated users"
```

## New-ADIDNSNode

This function adds a DNS node to an Active Directory-Integrated DNS (ADIDNS) Zone through an encrypted LDAP add request.

#### Example:

- Add a wildcard record to a ADIDNS zone and tombstones the node.

```
New-ADIDNSNode -Node * -Tombstone
```

## New-DNSRecordArray

This function creates a valid byte array for the dnsRecord attribute.

#### Example:

- Create a dnsRecord array for an A record pointing to 192.168.0.1.

```
New-DNSRecordArray -Data 192.168.0.1
```

## New-SOASerialNumberArray

This function gets the current SOA serial number for a DNS zone and increments it by the set amount.

Example:

- Generate a byte array from the current SOA serial number incremented by one.

```
New-SOASerialNumberArray
```

## Rename-ADIDNSNode

This function can rename a DNS node.

Example:

- Renames a DNS node named test to test2.

```
Rename-ADIDNSNode -Node test -NodeNew test2
```

## Remove-ADIDNSNode

This function can remove a DNS node.

Example:

- Removes a wildcard node.

```
Remove-ADIDNSNode -Node *
```

## Revoke-ADIDNSPermission

This function removes an ACE to a DNS node or zone DACL.

Example:

- Remove the GenericAll ACE associated with the user1 account.

```
Revoke-ADIDNSPermission -Node * -Principal user1 -  
Access GenericAll
```

## Set-ADIDNSNodeAttribute

This function can append, populate, or overwrite values in a DNS node attribute.

Example:

- Set the writable description attribute on a node named test.

```
Set-ADIDNSNodeAttribute -Node test -Attribute  
description -Value "do not delete"
```

## Set-ADIDNSNodeOwner

This function can set the owner of a DNS Node. Note that a token with SeRestorePrivilege is required.

Example:

- Set the owner of a node named test to user1.

```
Set-ADIDNSNodeOwner -Node test -Principal user1
```

# Miscellaneous Functions

## Get-KerberosAESKey

This function can generate Kerberos AES 256 and 128 keys from a known username and password. This can be used to test pass the hash in invoke-DNSUpdate

